RICE UNIVERSITY

Learning the Structure of High-Dimensional Manifolds with Self-Organizing Maps for Accurate Information Extraction

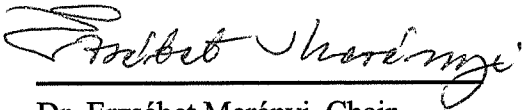
by

Lili Zhang

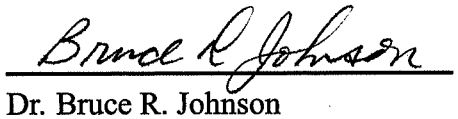
A THESIS SUBMITTED
IN PARTIAL FULFILLMENT OF THE
REQUIREMENTS FOR THE DEGREE

Doctor of Philosophy

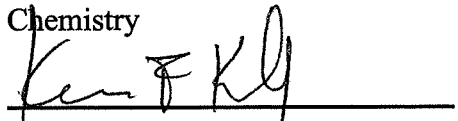
APPROVED, THESIS COMMITTEE



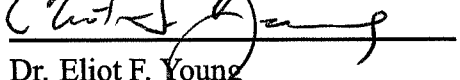
Dr. Erzsébet Merényi, Chair
Research Professor of Statistics and Electrical and Computer Engineering



Dr. Bruce R. Johnson
Distinguished Faculty Fellow of Chemistry



Dr. Kevin F. Kelly
Associate Professor of Electrical and Computer Engineering



Dr. Eliot F. Young
Principle Scientist, Department of Space Studies, Southwest Research Institute

HOUSTON, TEXAS
MAY 2011

ABSTRACT

Learning the Structure of High-Dimensional Manifolds with Self-Organizing Maps for Accurate Information Extraction

by

Lili Zhang

This work aims to improve the capability of accurate information extraction from high-dimensional data, with a specific neural learning paradigm, the Self-Organizing Map (SOM). The SOM is an unsupervised learning algorithm that can faithfully sense the manifold structure and support supervised learning of relevant information from the data. Yet open problems regarding SOM learning exist. We focus on the following two issues.

1. Evaluation of topology preservation. Topology preservation is essential for SOMs in faithful representation of manifold structure. However, in reality, topology violations are not unusual, especially when the data have complicated structure. Measures capable of accurately quantifying and informatively expressing topology violations are lacking. One contribution of this work is a new measure, the Weighted Differential Topographic Function ($WDTF$), which differentiates an existing measure, the Topographic Function (TF), and incorporates detailed data distribution as an importance weighting of violations to distinguish severe violations from insignificant ones. Another contribution is an interactive visual tool, TopoView, which facilitates the visual inspection of violations on the SOM lattice. We show the effectiveness of the combined use of the $WDTF$ and TopoView through a simple two-dimensional data set and two hyperspectral images.

2. Learning multiple latent variables from high-dimensional data. We use an existing two-layer SOM-hybrid supervised architecture, which captures the manifold structure in its SOM hidden layer, and then, uses its output layer to perform the supervised learning

of latent variables. In the customary way, the output layer only uses the strongest output of the SOM neurons. This severely limits the learning capability. We allow multiple, k, strongest responses of the SOM neurons for the supervised learning. Moreover, the fact that different latent variables can be best learned with different values of k motivates a new neural architecture, the Conjoined Twins, which extends the existing architecture with additional copies of the output layer, for preferential use of different values of k in the learning of different latent variables. We also automate the customization of k for different variables with the statistics derived from the SOM. The Conjoined Twins shows its effectiveness in the inference of two physical parameters from Near-Infrared spectra of planetary ices.

Acknowledgments

Without help and support along the way, Ph.D. would have been an insurmountable task for me. I cherish the opportunity here to thank all those who have constituted my memorable years at Rice.

First, I would like to thank my parents for their support during my entire life time, in whatever I decide, I do and I expect.

Deep thanks to my husband, Qing, who has been there through the best and the worst. His support helped me survive the hardest times in these years.

I would like to thank my committee members, Dr. Bruce Johnson, Dr. Kevin Kelly and Dr. Eliot Young, for their roles in my successful completion. It is a great honor having them on my committee. I thank Dr. Eliot Young also for his helpful discussions in our collaborative work.

Many thanks goes to our collaborators and colleagues. I thank Dr. William Grundy for his always prompt responses to my inquiries and helpful discussions. I thank Dr. Thomas Villmann for his valuable discussions regarding my research, and especially for his heartfelt encouragement before my very first formal presentation in WSOM09'.

I appreciate the gracious permissions from Dr. Jim Campbell, Dr. Teuvo Kohonen, Dr. John Kerekes, Dr. Juha Vesanto, Dr. Esa Alhoniemi, Dr. Kadim Taşdemir and Dr. Erzsébet Merényi, for the use of their published figures or results in my thesis.

Thanks to many wonderful students and friends at Rice: Xinya Zhang, Lei Ren, Lina

Xu, Kadim Taşdemir, Brian Bue, Michael Mendenhall, to name a few. I have enjoyed the moments with your company.

Finally, I wish to take the opportunity to acknowledge and thank my advisor, Dr. Erzsébet Merényi. Her patience, mentorship and friendship have made my Ph.D. experience extremely fruitful.

This work was partially supported by grants NNG06GE95G and NNG05GA63G from the Applied Information Systems Research Program, NASA, Science Mission Directorate.

Contents

List of Figures

List of Tables

Chapter 1

Self-Organized (unsupervised) learning for understanding complicated high-dimensional data

1.1 Challenges in information extraction from real world data

The data collected to characterize a real world problem, process, or object, are often high-dimensional thanks to the new sensory technologies that improve our perception of the world, and the modern computerized systems that are capable of collecting and storing huge amounts of data. The high dimensionality of the acquired data on one hand enables a wealth of information, on the other hand poses specific difficulties for information extraction methods. One prime example of such data is high-resolution spectra of planetary surfaces taken by modern imaging spectrometers. These spectrometers can resolve radiation in narrow band passes and take measurements at hundreds of wavelengths simultaneously. Each acquired spectrum is a high-dimensional vector of measurements at different wavelengths. When the measurements cover a contiguous spectral range, the acquired spectra are *hyperspectral*. By generating such spectra for all pixels in a contiguous scene, the

spectrometers produce a *hyperspectral image* (Fig. 1.1). Each pixel in such an image is a high-dimensional feature vector (spectrum), which provides a unique fingerprint of the material in the spatial area represented by that pixel. These spectral fingerprints can help unravel the chemical and physical properties of the materials. For example, a hyperspectral image can afford discrimination among many different surface materials, such as soil constituents and plant species in terrestrial regions or ice species on the surfaces of outer Solar System bodies. A hyperspectral image taken in wavelength ranges sensitive to surface physical conditions (e.g., temperature and grain size) can also be used to derive these physical parameters.

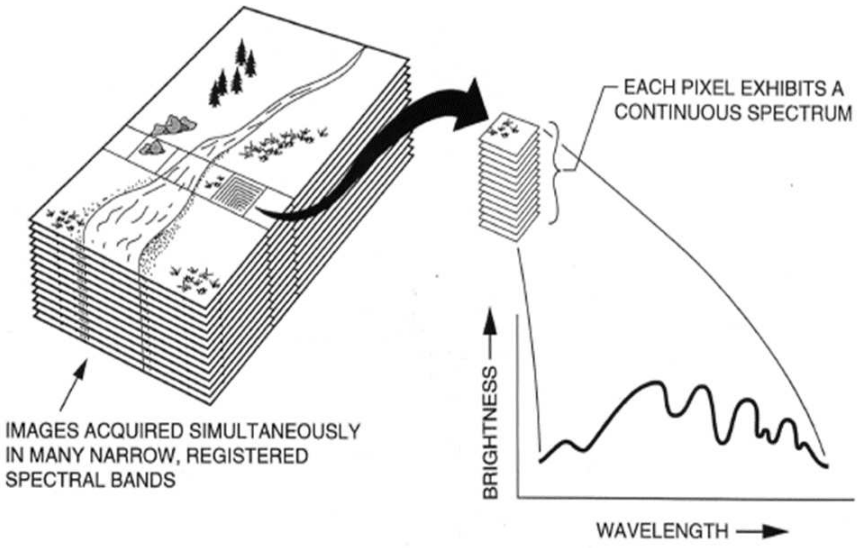


Figure 1.1: Illustration of a hyperspectral image. Each pixel in this image is a high-dimensional feature vector (spectrum) of the material in the spatial area represented by this pixel. Figure reproduced from [1], with permission of J. B. Campbell.

However, the inference of pertinent information from this kind of data is difficult. Since the underlying relations between the information of interest and the data are often too complicated to solve analytically, numerical solution to dynamical optimization is a commonly used traditional method. This method can handle low-dimensional data well, but loses its power for high-dimensional data, due to the so-called “curse of dimensionality” [2].

The exponentially increasing hypervolume in response to the growth of the data dimension makes the optimization process slow, and often unsuccessful, in finding optimal results. Feature extraction methods, used as data preprocessing, may help relieve the difficulties caused by high dimensionality. However, they can cause loss in information. In the applications with hyperspectral data, several feature extraction algorithms have been found unable to preserve relevant information [3]. Instead of paying great efforts to find good feature extraction algorithms to cooperate with the traditional dynamical optimization approach, we can alternatively use a more intelligent approach, machine learning algorithms, which *learn* the relations (or build models) between the information of interest and the data, from examples. No prior dimension reduction is needed for some machine learning algorithms. The models learned can then be easily used for fast information prediction from new data.

Another important fact that makes real world data demanding is their complicated structure. The redundancy in information, or dependance between the dimensions, influences the structure of the data. The subspace occupied by the data samples is called a *data manifold*. Due to partial and often nonlinear dependance between the dimensions, the data manifold can be *highly structured* [4]. This means there can be many clusters in the data, with various sizes and extremely varied statistical properties (variance, skewness, etc.). A potential challenge arising from such sophisticated data is to differentiate among the clusters, as the ability to do so may improve the level of detail of the information inferred from the data. In view of these, unsupervised learning algorithms that can correctly capture the structure of the data, become attractive.

1.2 Unsupervised learning

Our brains receive a massive flow of sensory information every day without explicit supervision, but can still develop capabilities to effectively determine the frequency of the occurrence of the incoming messages (density distribution) and the similarities between them (topology). *Unsupervised machine learning*, namely the learning by computer algorithms without extraneous supervision, mimics this natural process. In the scenario of neural machine learning, weighted connections between simple processing units (the artificial neurons) are iteratively modified in response to input patterns with a predefined learning rule. The final configuration of the connection weights often helps reveal some aspect of the structure of the data manifold, such as the distribution of clusters. This capability is important for understanding highly structured data.

Unsupervised machine learning is useful also because it can help with *supervised machine learning*, which means the learning of the mapping from the data to their labels from training samples. Supervised learning builds a model between the input data and the outputs through minimization of the errors in the outputs. As a result, success not only depends on the capability of the algorithm, but also the quality of the training labels. For example, mislabeling in the training samples or labeling that does not cover the functional relationships in sufficient detail may hinder good performance of supervised learning. Unsupervised learning algorithms can be helpful in such situations. Unsupervised learning captures the manifold structure, which is an objective piece of knowledge of the data. With this knowledge, the inconsistency in the labeling of the training samples can be detected and novelty (clusters that are not distinguished by the training labels) may be discovered. Therefore, it is possible to improve the capability of a supervised learning algorithm by incorporating knowledge previously learned through an unsupervised method.

There are two types of unsupervised learning, parametric and non-parametric. Para-

metric methods, which use prior assumptions on the data properties, have major limitations when dealing with highly structured data. For example, the most famous clustering method, k-means, favors (hyper-) spherical cluster shapes and requires a predefined number of clusters. Another well-known parametric method is mixture modeling, which customarily assumes the probability density functions (*pdf*) as Gaussian kernels. Real world data, however, often contain non-spherical and non-Gaussian clusters, hence these two methods can yield incorrect results. In addition, estimation of parameters in parametric modeling becomes problematic for high-dimensional data because the requirement for excessively large number of data samples due to the high dimensionality is often unmet. Conversely, non-parametric methods are applicable regardless of the complexity of the manifold structure and regardless of the dimensionality of the data. For real problems where the right models and parameters for the data at hand are often unavailable, non-parametric methods can often achieve better results than parametric methods. In this thesis, we focus on the Self-Organizing Map (SOM), which is a powerful non-parametric unsupervised learning paradigm.

1.3 The Self-Organizing Map (SOM), a powerful unsupervised learning paradigm

The Self-Organizing Map (SOM) [5] was invented by Kohonen in the 1980s. The SOM is a specific artificial neural network paradigm. An *artificial neural network* is a modeling approach inspired by biological neural networks. In an artificial neural network, a large number of artificial neurons are connected by weights in a certain topology. The artificial neurons usually perform relatively simple functions, such as computing a weighted sum of its incoming signals. The connection weights between the artificial neurons are adapted

iteratively through learning. Different paradigms of artificial neural networks differ in any of the three essential aspects, i.e., the network topology, the function performed by the processing units and the learning rule that guides the adaptation of the weights. The detailed network topology and learning mechanism of the SOM will be discussed in Section 2.1. We will call artificial neural networks as neural networks, and call artificial neurons as neurons in the thesis.

The SOM is also an adaptive vector quantization (VQ) algorithm. The purpose of vector quantization is to represent a data set with a relatively small set of *prototypes* [6, 7]. The quantization process facilitates data compression in information transmission and storage. When a proper representation of similarity relationships between the prototypes is provided, the quantization prototypes can also be used for clustering [5, 8, 9]. The clustering performance, of course, is dependent on the goodness of the quantization algorithm, as well as on the effectiveness of the representation of similarities of the prototypes. The SOM is exceptional among adaptive vector quantizers in this sense because it not only places its prototypes in the data space to accurately sense the manifold structure but also organizes the SOM prototypes according to their similarities on a rigid (customarily 1- or 2-dimensional) lattice simultaneously. The combination of these two capabilities make the SOM powerful in faithful representation of the complicated structure of real world data. The representation of manifold structure on a rigid lattice is an indispensable component of the SOM, i.e., topology preservation, which has a profound significance in biological systems.

Topology preserving (or topographic) maps are not artifacts; rather, they are widely observed in biological nervous systems, such as the retinotopic map in the visual cortex [10, 11], the somatotopic map in the somatosensory cortex [12], and the tonotopic map in the auditory cortex [13]. The cells on the cortex are topographically organized according

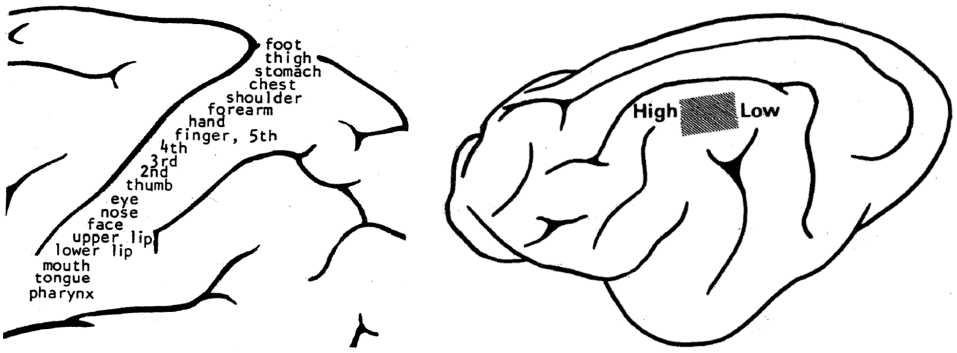


Figure 1.2: Examples of topographic maps in brains. Figures reproduced from Fig. 2.8 (page 100) and Fig. 2.9 (page 101) in [5], with kind permission of both Springer Science+Business Media and T. Kohonen. **Left**: The somatotopic map. The organization of the cells on the cortex reflects the spatial order of the body locations where the sensory signals are collected. **Right**: The tonotopic map (of cat). The cells that respond to acoustic signals are organized according to the frequencies of the tones perceived.

to the similarities of the signal patterns the brain receives. For example, sensory stimuli are organized in the somatotopic map according to the different locations of the body where the signals are received (Fig. 1.2, left). Sound signals are organized in the tonotopic map with respect to the acoustic frequencies of tones perceived (Fig. 1.2, right). This topology preservation property plays an important role in neural information storage, retrieval and processing. In 1973, von der Malsburg introduced a self-organizing process to model the local ordering of visual cortical cells [14]. His work pioneered the computer simulation of self-organization and a few related studies followed [15, 16]. In the early 1980s, a clear and intuitive algorithm, the Self-Organizing Map (SOM or Kohonen's SOM), was proposed by Kohonen [5], capturing much attention. The essential ingredient of the SOM algorithm is a neighborhood function, which gives rise to a global order in the map by local interactions between neighboring neurons.

Because of its appealing and advantageous properties, the SOM algorithm has been extensively studied and widely used by researchers and practitioners across a broad range of fields, including engineering, science, medicine, biology, economics [17, 18]. Up to now, there have been more than 7000 publications of its successful applications (http://www.

cis.hut.fi/research/som-bibl/). Some examples are text and web mining, hyperspectral image analysis, and microarray data analysis. Moreover, the SOM has been shown especially powerful in the learning of complicated data and the information extraction from these data in comparison to more traditional methods [4]. In this thesis work, we focus on the applications to hyperspectral imagery, or high-dimensional spectra without spatial context.

1.4 Contributions of this work to learning with SOMs

The SOM is a powerful tool for the analysis of high-dimensional complicated data. However, there are still open problems such as the evaluation of topology preservation in the map, cluster extraction from a learned SOM, optimized use of the SOM's knowledge in supervised learning of latent variables from high-dimensional data. Due to incomplete answers to these problems, the power of the SOM is not fully exploited in many situations. The goal of this work is to improve the learning of manifold structure with SOMs for precise information extraction. Our contributions include the following:

1. Development of a suite of advanced measures of topology preservation for SOMs;
2. Development of an interactive visualization tool to monitor topology preservation in SOMs;
3. Proposition of a novel SOM-hybrid supervised architecture, "Conjoined Twins", that optimizes the inference of latent variables from high-dimensional data;
4. Application of the "Conjoined Twins" to the inference of surface physical parameters from high-resolution Near-Infrared spectra of ices of the Pluto-Charon system.

A brief summary of each contribution is given in Sections 1.4.1–1.4.4. Detailed elaboration follows in Chapters 3–4.

1.4.1 Advancing the measures of topology preservation

We propose refined measures of topology preservation based on the Topographic Function (TF), introduced by Villmann *et al.* in 1997 [19]. The TF is better than other existing measures because it adopts an advanced graph distance metric, the induced Delaunay graph defined by Martinetz and Schulten in [20], to determine the neighboring relationships of the SOM prototypes in the data space. We find that a differential form of the TF provides a clearer view of topology violations (defects in topology preservation) than the TF for different scopes of violations in the map, therefore propose the Differential Topographic Function (DTF) as an alternative. In addition, we explicitly incorporate an often underutilized piece of information, the detailed data distribution around the SOM prototypes, into the DTF to distinguish the severe violations that result from improper or immature learning from the unimportant ones produced by noise. The TF and the DTF are insufficient in this sense. We call the new measure Weighted Differential Topographic Function ($WDTF$), which is a more precise evaluation of the quality of topology violations than the TF and the DTF.

1.4.2 An interactive visualization tool to monitor topology preservation in SOMs

While the measures provide a summary of the quality of topology preservation in SOMs, it is helpful for the user to visually locate the problematic areas in the map. Motivated by this idea, we develop an interactive tool we call TopoView, for visual inspection of the topology preservation in the SOM lattice. TopoView provides a set of thresholding abilities such that different subsets of violations meaningful for different applications can be inspected. Together with the newly proposed measures, TopoView can help diagnose the cause and the severity of the topology violations in the SOM.

1.4.3 A novel SOM-hybrid supervised architecture, "Conjoined Twins", that optimizes the inference of latent variables from high-dimensional data

An SOM-hybrid supervised architecture is a supervised neural network architecture, where an SOM is its hidden layer (hence the term "SOM-hybrid"). In the architecture, the layer above the SOM hidden layer uses the responses of the SOM neurons (not the SOM prototypes) to help with the extraction of information from the data. In the most frequent setup, only the strongest response from the SOM neurons is used (this is called Winner-Takes-All, or WTA). A two-layer SOM-hybrid supervised architecture, which contains an SOM hidden layer and an output layer, has been shown successful for classification problems in remote sensing applications [21, 22]. This architecture can also be used for inference of latent variables. However, we find that the WTA mode can severely limit the inference capability. We propose to use the first k ($k > 1$) strong responses of the SOM neurons (k-Winners-Take-All, or kWTA) such that we are able to infer certain latent variables better. Moreover, to solve the dilemma that different latent variables can be best learned with different values of k, we propose a new architecture we call "Conjoined Twins", where multiple copies of the output layer (multiple "heads") share the SOM ("body") and preferentially use different values of k for the learning of different latent variables. In addition, we automate the determination of k for different latent variables based on the statistics of the SOM.

1.4.4 Application of the "Conjoined Twins" to the inference of surface physical parameters from Near-Infrared spectra of ices of the Pluto-Charon system

We apply the innovative Conjoined Twins architecture to the inference of surface physical parameters from Near-Infrared spectra of ices in the Pluto-Charon system. The physical parameters can be nonlinearly dependent on each other and have much subtler influence on the spectral shapes than the chemical composition (different ice species) does. The accurate inference of the physical parameters is thus difficult. The Conjoined Twins has been shown effective for the inference of two physical parameters, temperature and grain size, from spectra of crystalline H2O ice with accuracies useful for scientific studies of diurnal temperature changes on Pluto and Charon.

The new measures of topology preservation, the interactive tool, TopoView, and the novel SOM-hybrid supervised architecture, the Conjoined Twins, proposed in the thesis are implemented, tested and documented by the author. See Appendix B for brief introductions of the software implementation of these new tools.

1.5 Outline of the thesis

Chapter 2 provides introductions to the basic concepts and customary procedures that will be used in the development and demonstration of new ideas and tools in the following chapters. We first introduce the SOM algorithm, including Kohonen's original version [5] and the Conscience variant [23]. We then review the important property of the SOM, namely topology preservation, and explain how topology preservation helps with the discovery of the manifold structure. Through two 2-dimensional data sets, we demonstrate the need for

advanced similarity metrics to assist the correct interpretation of the structure of data from the SOM, and show the usefulness of two such similarity metrics, the induced Delaunay graph [20] and the connectivity matrix [24], proposed in previous research.

In Chapter 3, we first address the need for informative tools to evaluate or monitor topology preservation in SOMs. In the review of existing measures, we discuss their pros and cons to justify our choice of the distance metrics and the formula used in the new measures we propose. Through a two-step improvement to the Topographic Function (TF), we propose the Weighted Differential Topographic Function ($WDTF$), a clearer and more accurate representation of the quality of topology preservation than the TF. Next, we introduce the interactive monitoring tool, TopoView, and describe several meaningful ways to filter out unimportant topology violations. At the end, we demonstrate the combined use of the $WDTF$ and TopoView through a simple 2-dimensional data set, a 194-dimensional real hyperspectral image of a volcanic field, and a 210-dimensional synthetic hyperspectral image of an urban area.

In Chapter 4 we start by addressing the challenges in the inference of latent variables from complicated data. To approach this inference problem we focus on an existing SOM-hybrid supervised neural architecture. We describe its network structure and the so-called winner-takes-all (WTA) mode, which is the customary use of the SOM's knowledge in the supervised learning by this architecture. In the WTA mode, the output layer only uses the strongest response of the SOM neurons. After discussing the effects of the WTA mode, we generalize the method such that multiple (k) strongest responses of the SOM neurons can participate in the supervised learning, which we call k-winners-take-all (kWTA) mode. Through application to the inference of surface physical parameters from Near-Infrared spectra of ices in the Pluto-Charon system, we find that different values of k can be beneficial to the inference of different latent variables. This motivates a novel neural architecture

we call Conjoined Twins, in which we allow the simultaneous use of different values of k as optimized for different latent variables. We describe the concept of the Conjoined Twins and show the effectiveness of the approach for inferring temperature and grain size from the high-dimensional spectra of crystalline water ice.

Chapter 5 summarizes the thesis work and discusses future directions.

Chapter 2

The Self-Organizing Map and its use for structure detection

2.1 The Self-Organizing Map (SOM) algorithm

2.1.1 The Kohonen SOM algorithm

The Self-Organizing Map (SOM) is a neural learning algorithm that maps a d-dimensional data manifold $M \subset R^d$ to a low-dimensional lattice A of N neurons. Data samples mapped to an SOM neuron $i \in A$ constitute the *receptive field*, RF_i, of that neuron. The mapping is formed in a topologically ordered way so that the structure of the high-dimensional data manifold is correctly manifested in the low dimensional lattice.

The following describes the fundamental network topology and the algorithm of the SOM. The most popular choice of the lattice structure is a 2-dimensional rectangular lattice, as illustrated in Fig. 2.1. (Another popular lattice type is hexagonal lattice, for which we will show an example at the end of this chapter.) Each neuron $i \in A$ has a d-dimensional weight vector, $\mathbf{w}_i$, assigned to it. The SOM weight vectors are iteratively adapted by an unsupervised learning algorithm proposed by Kohonen [5]. Each learning iteration consists of two steps: *competition* and *synaptic adaptation*, as described in eqs. 2.1 and 2.2. In the competition step, an SOM neuron, c, is selected as the SOM winner or best matching unit

(BMU) for an input vector $\mathbf{x}$ randomly chosen from M such that

$$\| \mathbf{w}_c - \mathbf{x} \|^2 \leq \| \mathbf{w}_j - \mathbf{x} \|^2 \quad \forall j \in A \tag{2.1}$$

In the synaptic adaptation step, all weight vectors, $\mathbf{w}_j$, are updated as

$$\mathbf{w}_j^{new} = \mathbf{w}_j^{old} + \alpha(t) h_{c,j}(t) (\mathbf{x} - \mathbf{w}_j^{old}) \tag{2.2}$$

where $\alpha(t)$ is a learning rate that decreases with time t. $h_{c,j}(t)$ is a neighborhood function that defines how much an SOM neuron j should learn from the input $\mathbf{x}$, for which the BMU is neuron c. The neighborhood function shows the $cooperation$ between the BMU and the rest of the neurons in the lattice A. $h_{c,j}(t)$ is commonly chosen as a Gaussian kernel in the Kohonen SOM (eq. 2.3). Other frequent choices can be a uniformly distributed spherical kernel (eq. 2.4), or, for rectangular SOM lattices, a uniformly distributed box kernel (eq. 2.5), centered over the BMU. In the formulae of the neighborhood functions, $\mathbf{r}_j$ denotes the coordinates of neuron j in the lattice. The neighborhood size (or radius), $\sigma(t)$, needs to be large at the beginning and diminish with time t, to help avoid global distortions in the map (lattice A).

$$\text{Gaussian kernel:} \qquad h_{c,j}(t) = \exp\left(\frac{-\|\mathbf{r}_j - \mathbf{r}_c\|_E^2}{2\sigma(t)^2}\right) \tag{2.3}$$

$$\text{Uniformly distributed spherical kernel:} \quad h_{c,j}(t) = \begin{cases} 1 & \|\mathbf{r}_j - \mathbf{r}_c\|_E \leq \sigma(t) \\ 0 & \|\mathbf{r}_j - \mathbf{r}_c\|_E > \sigma(t) \end{cases} \tag{2.4}$$

$$\text{Uniformly distributed box kernel:} \quad h_{c,j}(t) = \begin{cases} 1 & \|\mathbf{r}_j - \mathbf{r}_c\|_{max} \leq \sigma(t) \\ 0 & \|\mathbf{r}_j - \mathbf{r}_c\|_{max} > \sigma(t) \end{cases} \tag{2.5}$$

$\| \cdot \|_E$ represents the Euclidean norm. $\| \cdot \|_{max}$ represents the maximum norm (or city block

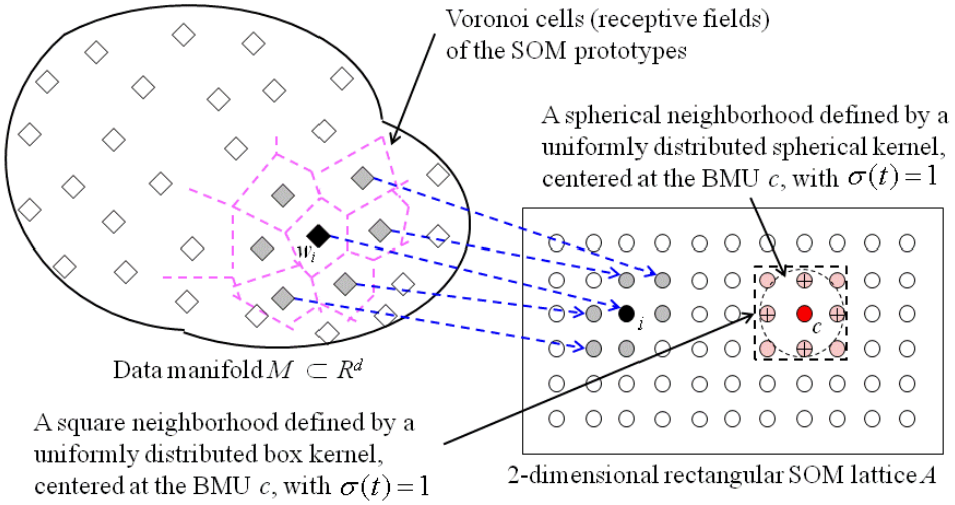


Figure 2.1: The SOM places a given set of prototypes optimally in the data space to represent a data manifold, and simultaneously organizes the prototypes in a rigid lattice according to their similarities. This figure provides an example of a 2-dimensional rectangular SOM A learned with a d-dimensional data manifold M. Circles in the SOM lattice are neurons. Diamonds are the learned SOM prototypes projected back into the data space. Blue dashed arrows relate some prototypes to their associated SOM neurons. Magenta dashed lines delineate the Voronoi cells (or receptive fields) of some SOM prototypes. Each prototype is the centroid of its Voronoi cell. When the SOM converges, it forms a topologically ordered mapping between M and A. This is illustrated for the prototype $\mathbf{w}_i$ (solid black diamond). 6 prototypes (solid grey diamonds) whose Voronoi cells share an edge with $\mathbf{w}_i$'s Voronoi cell are neighbors of $\mathbf{w}_i$ in the data space. The respective 6 neurons (solid grey circles) are neighbors of the neuron i (solid black circle) in the lattice A. In addition, this figure also shows the effects of two neighborhood functions, a uniformly distributed spherical kernel (eq. 2.4) and a uniformly distributed box kernel (eq. 2.5), on SOM learning. The spherical kernel with neighborhood size $\sigma(t) = 1$ allows the BMU c (red circle) and 4 other neurons (pink circles with "+" inside) in the dashed circle to adapt their prototypes. The box kernel with $\sigma(t) = 1$ allows the BMU c as well as 8 other neurons (pink circles) in the dashed box to adapt their prototypes.

distance). For two d-dimensional vectors $\mathbf{a} = [a_1, a_2, ..., a_d]$ and $\mathbf{b} = [b_1, b_2, ..., b_d]$, these two norms are defined by eq. 2.6 and eq. 2.7, respectively.

$$\text{Euclidean norm:} \quad \|\mathbf{a}-\mathbf{b}\|_E = (\sum_{i=1}^{d} |a_i - b_i|^2)^{\frac{1}{2}} \quad \mathbf{a}, \mathbf{b} \in R^d \tag{2.6}$$

$$\text{Maximum norm:} \quad \|\mathbf{a}-\mathbf{b}\|_{max} = \max_{i=1}^{d} |a_i - b_i| \quad \mathbf{a}, \mathbf{b} \in R^d \tag{2.7}$$

Fig. 2.1 illustrates the effects of two neighborhood functions, the uniformly distributed spherical kernel and the uniformly distributed box kernel functions, both with neighborhood size $\sigma(t) = 1$, on the synaptic adaptation phase of SOM learning. The uniformly

distributed spherical kernel with $\sigma(t) = 1$ defines a round neighborhood (in the dashed circle), around the BMU (red circle), in the 2-dimensional SOM lattice A. This means in this specific learning step, in addition to the BMU, 4 other neurons (pink circles with "+" inside) are simultaneously activated to update their weight vectors. The uniformly distributed box kernel with $\sigma(t) = 1$ defines a square neighborhood (in the dashed box), inside of which 8 pink neurons are activated together with the BMU.

After the SOM converges, i.e., the weight vectors no longer change significantly, the weight vectors $\mathbf{w}_i$ become the vector quantization prototypes of the data manifold M. We will refer to the SOM weight vectors as SOM prototypes from now on. The data space can be partitioned with respect to the SOM prototypes as in eq. 2.8. This is the so-called *Voronoi tessellation* V. The partitions V_i are called *Voronoi cells*, where the prototypes $\mathbf{w}_i$ are the centroids.

$$V_i = \{\mathbf{x} \in R^d : \|\mathbf{x} - \mathbf{w}_i\| \leq \|\mathbf{x} - \mathbf{w}_j\| \quad \forall\, j \in A\} \qquad i \in A \tag{2.8}$$

Note that the definition is given in the context of the SOM here, but Voronoi tessellation can be done with any set of prototypes, obtained in any way.

Voronoi cells V_i coincide with RF_i, the receptive fields of neurons i (or receptive fields of prototypes $\mathbf{w}_i$). The prototype $\mathbf{w}_i$ represents all data samples in its Voronoi cell V_i and the neuron i is the BMU of those data samples. The number of data samples mapped to a neuron i is called the *mapping density* or the *size of the receptive field* $|RF_i|$ of neuron i. The neurons with zero mapping density are called *empty neurons*, and the prototypes assigned to them are called *empty prototypes*.

Ideally the SOM algorithm forms a topology preserving mapping between the data manifold M and the SOM lattice A. The mapping is *topology preserving* when the mapping in both directions, $A \rightarrow M$ and $M \rightarrow A$, is neighborhood preserving. This means that

adjacent neurons in the lattice A should represent nearby data samples in the data space, and nearby data samples should map to adjacent neurons or to the same neuron in A. Since the SOM is a vector quantization algorithm, topology preservation can be interpreted on the level of prototypes as follows. The prototypes that belong to neurons *adjacent in the lattice* A should also be *adjacent in the data manifold* M, and vice versa. In this definition, the adjacency relationships between neurons or prototypes depend on the distance (similarity) metric used. The distances between SOM neurons in the lattice are usually computed by the Euclidean norm (eq. 2.6). Two neurons i and j are *adjacent* or *immediate lattice neighbors* when $\|\mathbf{r}_i - \mathbf{r}_j\|_E = 1$. For a 2-dimensional rectangular SOM lattice shown in Fig. 2.1, this definition results in 4 immediate neighbors (pink circles with "$+$" inside) for the neuron c (red circle). A more common distance metric used for a rectangular lattice is the maximum norm (eq. 2.7). With this metric, neurons i and j are defined as *adjacent* or *immediate lattice neighbors* when $\|\mathbf{r}_i - \mathbf{r}_j\|_{max} = 1$. This results in 8 immediate neighbors (pink circles) for the neuron c in a 2-dimensional rectangular lattice, as seen in Fig. 2.1. In the thesis, maximum norm is the default distance metric to determine immediate lattice neighbors. From now on, when two prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$ are assigned to two neurons which are immediate neighbors in the SOM lattice, we will directly call these two prototypes *immediate lattice neighbors* or *adjacent prototypes in the lattice*. To talk about topology preserving mapping, we also need to define the adjacency of prototypes in the data space. Such definition was put forward by Martinetz and Schulten (1994) as follows. Prototypes are *adjacent* or *neighboring* if their Voronoi cells share an edge, and hence they are also called *Voronoi neighbors*. For example, in Fig. 2.1, the prototype $\mathbf{w}_i$ (solid black diamond) has 6 adjacent prototypes (solid grey diamonds) in the data space. With adjacency defined in both the lattice and the data space, we can now illustrate topology preservation in a well organized SOM. The 6 prototypes neighboring prototype $\mathbf{w}_i$ are

associated with 6 grey neurons in the lattice. Blue dashed arrows help relate some prototypes to their respective neurons. The 6 grey neurons are adjacent to the neuron i in the lattice A. This means that the Voronoi neighbors of prototype $\mathbf{w}_i$ are also its immediate lattice neighbors. This topological ordering of the prototypes in the SOM lattice according to their similarities in the data space mimics the organization of the brain cells on the cortex according to the similarities between the signals received by these cells (Fig. 1.2).

2.1.2 The Conscience SOM variant

In this work we use the Conscience variant [23] of the original Kohonen's algorithm for two reasons. One is the ability of the Conscience algorithm to achieve equiprobablistic mapping (also called maximum entropy mapping). This means each SOM prototype represents approximately equal number of data samples. The resulting SOM provides the best possible approximation of the data *pdf* with the given number of prototypes. Equiprobablistic mapping is also optimum for information transfer, for which the Kohonen SOM is suboptimal. The other advantage of using the Conscience algorithm is the economy of computation compared to the Kohonen SOM, owing to the use of a fixed small neighborhood size for $h_{c,j}(t)$ in eq. 2.2.

The Conscience algorithm achieves an equiprobablistic mapping through the addition of a bias, b_j for each neuron j, to the distance between the prototype $\mathbf{w}_j$ and the input vector $\mathbf{x}$, in the competition step. The BMU c is found such that

$$\| \mathbf{w}_c - \mathbf{x} \|^2 - b_c \leq \| \mathbf{w}_j - \mathbf{x} \|^2 - b_j \;\; \forall j \in A \tag{2.9}$$

The bias b_j is computed from the winning frequency p_j of the neuron j.

$$b_j = \gamma(t) \times (1 - (N \times p_j)) \tag{2.10}$$

p_j is updated in each iteration as

$$p_j^{new} = p_j^{old} + \beta(t) \times (\delta_{c,j} - p_j^{old}) \tag{2.11}$$

where $\delta_{c,j}$ is the Kronecker delta, $\beta(t)$ and $\gamma(t)$ are user-specified parameters. As a result, a neuron that wins with larger than average frequency will be discouraged from winning by an increase in its bias. This added heuristic "conscience" thus helps achieve equiprobablistic mapping.

With the added biases, the Conscience algorithm can use a fixed and small neighborhood for the synaptic adaptation step (eq. 2.2) in the learning. An example of such neighborhood function used for a rectangular SOM lattice is the uniformly distributed box function with $\sigma(t) \equiv 1$ (eq. 2.5). In a 2-dimensional rectangular SOM lattice, this neighborhood function will activate 8 more neurons (pink circles) in addition to the BMU (red circle) to update their prototypes (Fig. 2.1). The small neighborhood size significantly lightens the overall computational burden in spite of the increased number of operations in eqs. 2.9–2.11.

2.2 Structure detection from learned Self-Organizing Maps

When an SOM has converged, we may detect the manifold structure from the learned SOM. In this thesis work, structure detection mainly refers to identification of clusters because in our applications most of the scientific goals are related to the finding of meaningful clusters in the data. However, the reader should be aware that the SOM is an algorithm that learns the manifold structure regardless of whether the data has clusters in it. For instance, an SOM can perfectly learn a manifold with uniform distribution, in which case, naturally, no cluster will be detected from the SOM.

In this section we will explain the relation between topology preservation and faithful representation of manifold structure, describe the visualization methods that help with interactive cluster extraction from the SOM, and introduce two advanced distance metrics that assist correct understanding of the manifold structure from the SOM.

2.2.1 Topology preservation is essential for structure detection

The essence of unsupervised learning is the grasp of the relationships among data samples. Topology is one of such relationships, expressing the neighboring relationships in the data manifold. The SOM aims to preserve the topology of a high-dimensional data manifold in a low-dimensional lattice, which makes it unique among vector quantization algorithms. Fig. 2.2 provides an example of a topology preserving map (left) and an example of a "twisted" map (right), which is "twisted" and does not preserve the topology of the data. Both SOMs were trained with a data set drawn from a 2-dimensional uniform distribution in a square area. In Fig. 2.2, the learned SOM prototypes are plotted as circles in the data space. To make the lattice structure easy to see, we connect two prototypes if their respective neurons are adjacent in the SOM lattice, by the Euclidean norm criterion $\|\cdot\|_E = 1$. In Fig. 2.2, left, the topology of the SOM lattice coincides with the topology of the data manifold, which indicates topology preservation. The SOM in Fig. 2.2, right, however, has a "twist", which can lead to an incorrect detection of two separate clusters from the SOM while this data set in fact has no clusters in its structure. "Twists" in the map that prevent topology preservation are called *topology violations*. Since topology preservation is defined as neighborhood preservation in both mapping directions, there are two types of topology violations, forward and backward [19]. *Forward topology violations* occur when two prototypes are Voronoi neighbors in the data space while they are not immediate lattice neighbors in the SOM. *Backward topology violations* occur when two prototypes are

immediate lattice neighbors in the SOM, while they are not Voronoi neighbors in the data space.

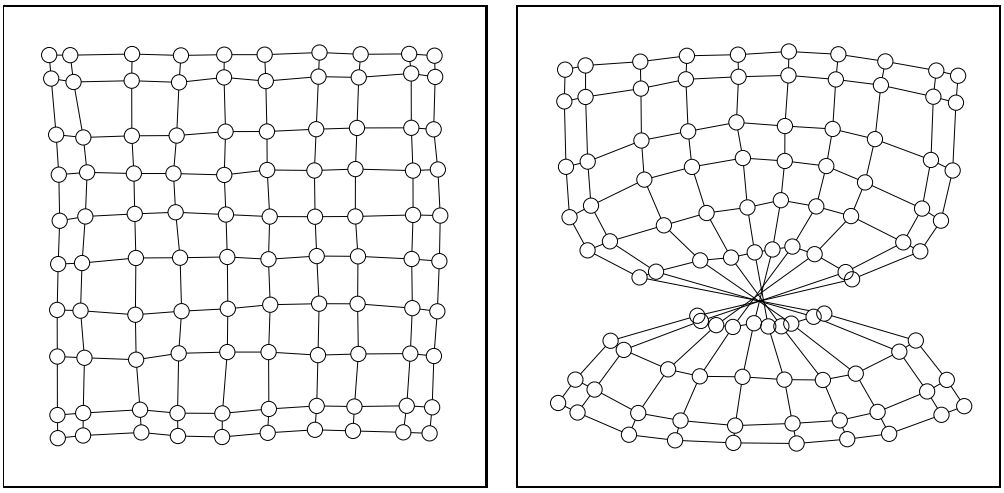


Figure 2.2: Prototypes of 10×10 SOMs (circles), mapped to data space after learning converged. The data set learned by the SOMs is generated from a uniform distribution in a square area. Two prototypes are connected if they are adjacent in the SOM lattice. **Left**: Topology is preserved in the SOM. **Right**: The SOM is twisted in the data space. Topology is not preserved.

Topology preservation in the SOM is essential for correct detection of the manifold structure. A topology-preserving map can be viewed as an ordered display of data. Assuming perfect topology preservation, the adjacency relationships between the neurons in the SOM lattice faithfully reflect the adjacency between their respective prototypes in the data space. However, topology preservation alone is not sufficient for detection of structure because the lattice distances between the prototypes do not reflect the dissimilarities between them. Thus, similarity metrics and visualization schemes to display the similarity relationships across the prototypes on the lattice are important tools for structure detection from the SOM.

2.2.2 Structure detection with (modified) U-matrix and advanced similarity metrics

U-matrix and mU-matrix

The U-matrix (unified distance matrix), proposed by Ultsch and Siemon [25] is a distance matrix widely used for visualization of the dissimilarity relationships between SOM prototypes on the SOM lattice. For a given prototype, its Euclidean distances to its immediate lattice neighbors are computed and averaged. To visualize this average distance to immediate lattice neighbors, the SOM is often visualized as a lattice of grid cells, which represent neurons, as seen in Fig. 2.3, middle and right. The average distance of the prototype to its lattice neighbors can be expressed as a grey scale intensity, of the cell of that prototype, proportional to the average distance. (No example of U-matrix visualization is shown here.) The U-matrix visualization and its variants [25, 26] have been shown effective for relatively large SOMs learned with small data sets that have a low number of clusters, but when a small SOM is used to learn a large data set containing many clusters, the averaging of the distances can smear the cluster boundaries and cause the loss of small clusters.

We use a high-resolution version of the U-matrix, the modified U-matrix (mU-matrix), introduced by Merényi in the 1990s, and described in [27]. The mU-matrix removes the averaging of the distances in the original U-matrix. It shows the distances of a given prototype to all of its immediate lattice neighbors separately as “fences” on the border of the grid cells including the diagonals. Fig. 2.3, middle and right, gives an example of the mU-matrix visualization for a rectangular SOM lattice, showing the distances from each prototype to its 8 immediate lattice neighbors. This representation also leaves room for displaying additional information in the grid cells ([21, 28]). One example of such information can be the mapping density (the number of data samples mapped to each neuron). Fig. 2.3, middle, is an example of visualizing both the mU-matrix and the mapping density. The data

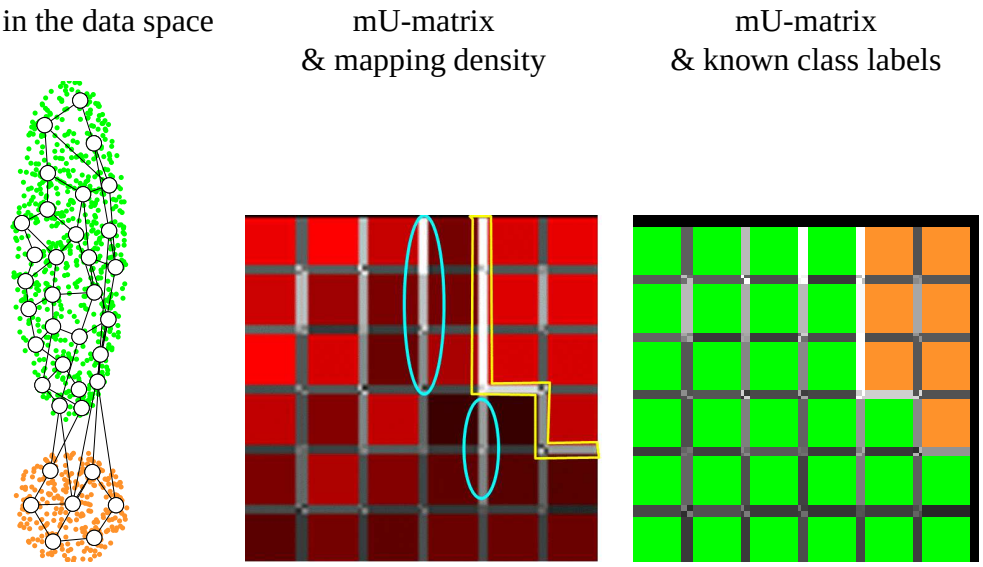


Figure 2.3: Possible visualizations of a 6×6 SOM learned with a 2-dimensional "exclamation mark" data set. In the middle and right, the SOM is shown as a lattice of grid cells, each of which represents an SOM neuron. **Left**: Data samples are shown as small green and orange dots. The colors represent two classes, the upper and lower parts of this "exclamation mark" data manifold, respectively. Open circles are the SOM prototypes projected back into the data space. The prototypes are connected according to the SOM lattice structure. **Middle**: The SOM overlain with the mU-matrix, which shows the distances, in the data space, of each prototype to its immediate lattice neighbors, as "fences" on the borders of the grid cells. The grey scale intensities of the "fences" are proportional to the distances they represent. White is large distance. The intensities of red, of the grid cell of each neuron, indicate the mapping density (number of data samples mapped to each neuron). The conspicuously high fences outlined by the yellow lines correspond to the discontinuity between the two clusters in the data. This can be seen if compared with the SOM on the right, which is overlain with the known class labels. The other relatively high fences, such as those in cyan ovals, result from twists in the map, which can be seen in Fig. 2.4. **Right**: The SOM overlain with the mU-matrix and the known class labels (colors).

learned by the SOM is a 2-dimensional "exclamation mark" data set (small filled dots in Fig. 2.3, left). The mapping density of each neuron is shown as proportional intensity of monochrome red in its grid cell in Fig. 2.3, middle. When class labels are available, we can also overlay this piece of information on the SOM. For example, each neuron (cell) can be color-coded to the majority class label of the samples in its receptive field. The overlain known class labels can help compare the SOM's knowledge with truth. The class labels, of course, are not used in SOM learning. For example, we have the data labels for the "exclamation mark" data set, as seen in Fig. 2.3, left. Two different class labels, color coded as

green and orange, represent data samples in the upper and lower parts of the “exclamation mark”, respectively. The SOM in Fig. 2.3, right, shows these known class labels as well as the mU-matrix. With the overlain class labels, we can see two clearly separated clusters in the SOM, an orange cluster with 7 prototypes and a green cluster with 29 prototypes. This cluster structure coincides with what we can observe from the mU-matrix visualization in Fig. 2.3, middle, where a strong “fence”, outlined by yellow lines, separates the prototypes in the upper right corner from the rest. This shows that the SOM has successfully learned the cluster structure.

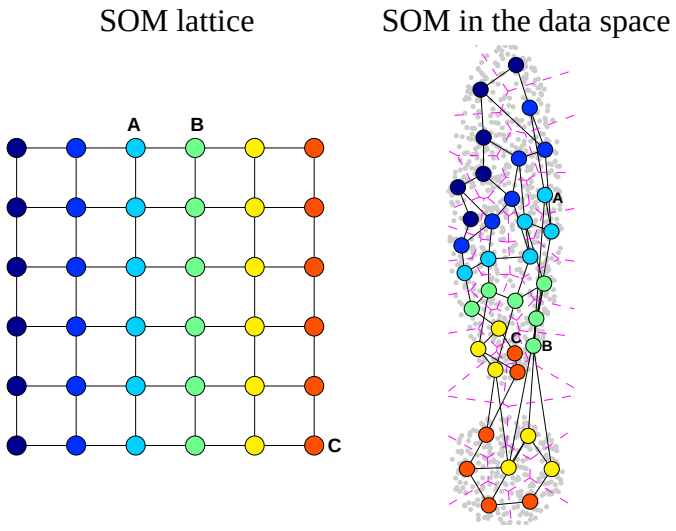


Figure 2.4: Visualization of the SOM learned with the 2-dimensional “exclamation mark” data set (grey filled dots). SOM prototypes are shown as circles, connected according to the lattice structure and colored according to their positions in the lattice. The coloring of the prototypes makes it easy to relate the locations of the prototypes across the lattice space and the data space. **Left**: The SOM prototypes are visualized in the lattice space. **Right**: The SOM prototypes are projected back into the data space. The Voronoi tessellation of the data space with respect to the prototypes are shown as magenta lines. Between the prototypes A and B, there is a backward topology violation, because A and B are immediate lattice neighbors but they are not Voronoi neighbors in the data space. Between the prototypes B and C, there is a forward topology violation, because B and C are Voronoi neighbors in the data space while they are not immediate lattice neighbors.

The use of (m)U-matrix for cluster identification, however, is nontrivial for two reasons. First, the determination of cluster boundaries based on “fences” often relies on interactive thresholding of the fence values. Second, the procedure is based on the assumption of perfect topology preservation, while topology violations are not unusual in real applica-

tions. To illustrate these two difficulties, we use the same “exclamation mark” data set. In Fig. 2.3, middle, besides the fences which indicate the discontinuity in the manifold, we also see several other relatively high fences within the two clusters, such as those in cyan ovals. Expert knowledge is needed here to inspect the prototypes to determine whether they are different enough (fences are high enough) to form meaningful subclusters. Another possible cause of the high fences can be topology violations in the map. Plotting the SOM prototypes in the data space and connecting them according the lattice structure, as in Fig. 2.3, left, is helpful for capturing the twists in the map. To make the twists in the map easier to see, we color the prototypes according to their relative lattice locations, in Fig. 2.4. We find that the square (6×6) map stretches and folds itself to some extent to fit in the elongated manifold shape. An example of stretching can be seen between two immediate lattice neighbors A and B (Fig. 2.4, left), which are forced to be apart, i.e., non-adjacent, in the data space (Fig. 2.4, right). This indicates a backward topology violation between A and B. An example of folding can be seen in the bottom of the upper part of the “exclamation mark”, where an orange and a yellow string of prototypes entangle in the data space (Fig. 2.4, right) while they are well separated in the lattice (Fig. 2.4, left). This folding causes forward violations, such as between prototypes B and C. B and C are Voronoi neighbors as their Voronoi cells share a common border, as seen on the right, but they are not immediate lattice neighbors, as seen on the left. The inspection of the SOM prototypes in the data space successfully helps diagnose topology violations for this case, but this strategy is restricted to 1-, 2- and 3-dimensional data. Methods to compare the neighboring relationships between the prototypes in the data space and in the lattice, regardless of the data dimensionality, are desirable, so that we can understand the manifold structure and find topology violations from SOMs learned with high-dimensional data. Better similarity metrics have been proposed in [20, 24] for more accurate structure detection. These will

be discussed next.

Induced Delaunay graph

The dual of the Voronoi tessellation (eq. 2.8) is the *Delaunay graph*, denoted by D. The Delaunay graph expresses the adjacency of the Voronoi cells, thus it can be represented by a binary adjacency matrix. For an SOM lattice A with N prototypes, the binary adjacency matrix is of size $N \times N$, and can be written as:

$$D(i,j) = \begin{cases} 1 & V_i \text{ and } V_j \text{ share a common border} \\ 0 & \text{otherwise} \end{cases} \qquad i,j \in A \qquad (2.12)$$

$D(i,j)$ denotes an edge of the graph whose vertices are the prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$. $D(i,j) = 1$ means that $\mathbf{w}_i$ and $\mathbf{w}_j$ are connected by an edge. An illustration of the Voronoi tessellation and the Delaunay graph is given for the SOM learned with the "exclamation mark" data set in Fig. 2.5, left.

Martinetz and Schulten pointed out in their paper [20] that the Delaunay graph could not correctly represent the connectedness in the data manifold. (This is confirmed by the "exclamation mark" data set in Fig 2.5, left. The discontinuity between the upper and lower parts of the manifold is not shown by the Delaunay graph.) Martinetz and Schulten therefore introduced the notion of the *induced Voronoi tessellation* $\widetilde{V}$, and its dual, the *induced Delaunay graph* $\widetilde{D}$ in [20]. The induced Voronoi tessellation $\widetilde{V}$ (the induced Delaunay graph $\widetilde{D}$) is the intersection of the regular Voronoi tessellation V (the regular Delaunay graph D) and the data manifold M. By incorporating the manifold shape into its definition, $\widetilde{V}$ ($\widetilde{D}$) can faithfully represent the connectedness in the manifold regardless of the complexity of the manifold shape. Applying the theory to a set of N learned SOM prototypes, we obtain N induced Voronoi cells. The induced Voronoi cell of the prototype $\mathbf{w}_i$, $\widetilde{V}_i$, is

defined by

$$\widetilde{V}_i = \{\mathbf{x} \in M : \|\mathbf{x} - \mathbf{w}_i\| \leq \|\mathbf{x} - \mathbf{w}_j\| \quad \forall\, j \in A\} \qquad i \in A \tag{2.13}$$

where A is the SOM lattice. The induced Delaunay graph $\widetilde{D}$ is a set of edges connecting the prototypes whose induced Voronoi cells share a common border, as in eq. 2.14.

$$\widetilde{D}(i,j) = \begin{cases} 1 & \widetilde{V}_i \text{ and } \widetilde{V}_j \text{ share a common border} \\ 0 & \text{otherwise} \end{cases} \qquad i,j \in A \tag{2.14}$$

$\widetilde{D}(i,j) = 1$ indicates the prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$ are connected by an edge in the induced Delaunay graph. Martinetz and Schulten also showed that, under certain circumstances, $\widetilde{D}$ can be effectively constructed through Hebbian learning [29], in which the synaptic weight between two neurons are reinforced if the activation of one neuron fires the other repeatedly. By the Hebbian learning, a *connection* (an edge) between two prototypes is constructed if these two prototypes form a pair of BMU and second BMU for at least one data sample. For an SOM with N prototypes, $\widetilde{D}$ then can be constructed by

$$\widetilde{D}(i,j) = \begin{cases} 1 & \mathbf{w}_i \text{ and } \mathbf{w}_j \text{ form a pair of BMU and second BMU} \\ & \text{for at least one data sample.} \\ 0 & \text{otherwise} \end{cases} \qquad i,j \in A \tag{2.15}$$

Fig. 2.5, middle, provides an illustrations of the induced Delaunay $\widetilde{D}$, with the SOM prototypes that learned the "exclamation mark" data set. $\widetilde{D}$ makes the discontinuity in the data obvious with a disconnect in the graph. Since the induced Delaunay graph $\widetilde{D}$ is a more accurate representation of the manifold structure than the regular Delaunay graph D [20], it is used to define the adjacency between the prototypes. Two prototypes, $\mathbf{w}_i$ and $\mathbf{w}_j$, are

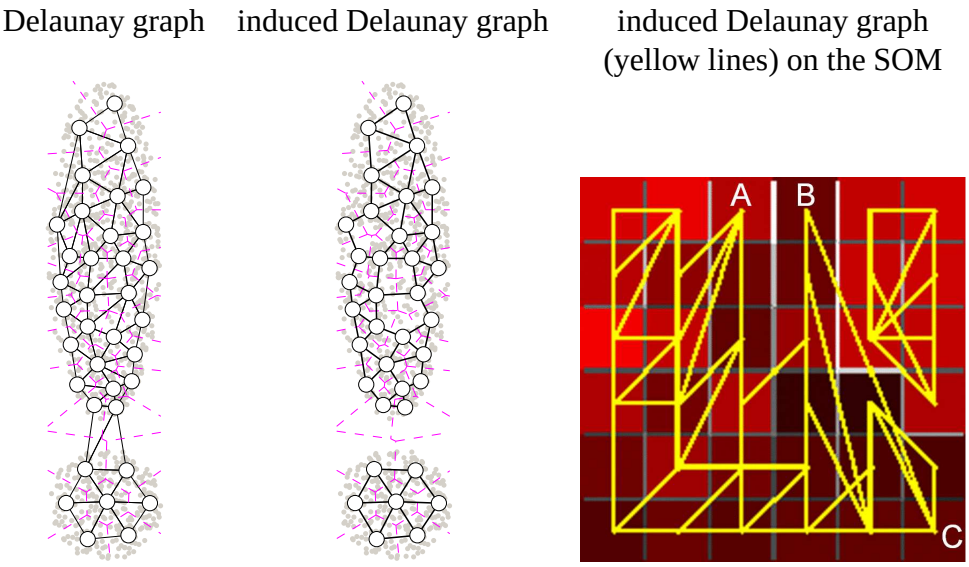


Figure 2.5: Illustration of Delaunay graph and induced Delaunay graph with the 2-dimensional "exclamation mark" data set (gray filled dots). Open circles represent the 36 learned SOM prototypes projected back into the data space. The prototypes are the centroids of the Voronoi cells, which are delineated by dashed magenta lines. **Left**: The Delaunay graph (black lines) does not help separate the two disconnected parts in the data set. **Middle**: The induced Delaunay graph (black lines) highlights the discontinuity in the manifold. **Right**: The induced Delaunay graph (yellow lines) drawn on the SOM, which is also overlain with the mU-matrix and the mapping density as in Fig. 2.3, middle. An example of a backward and a forward topology violation can be seen between the prototypes A and B, and between the prototypes B and C, respectively.

defined as *adjacent*, *neighboring*, *connected* prototypes, or *Voronoi neighbors* in the data space when $\widetilde{D}(i,j) = 1$ [20].

For data sets with no more than 3 dimensions, we can plot the induced Delaunay graph $\widetilde{D}$ in the data space to inspect the connectedness between the prototypes as in Fig. 2.5, middle. For data sets with more than 3 dimensions, it is impossible to visualize $\widetilde{D}$ in the data space, but we can drape $\widetilde{D}$ over the SOM by connecting grid cells with an edge when they represent two connected prototypes in $\widetilde{D}$. Fig. 2.5, right, provides an example by showing $\widetilde{D}$ as yellow lines on the SOM. With $\widetilde{D}$ overlain on the SOM, the topology violations, i.e., the inconsistency in the neighboring relationships between the prototypes across the lattice space A and the manifold space M, are visible. For example, in Fig. 2.5, right, the lack of connection between the prototypes A and B, which are immediate lattice neigh-

bors, indicates a backward topology violation. The connection between the prototypes B and C, which are not immediate lattice neighbors, indicates a forward topology violation. These observations agree with what we found by inspecting the SOM in the data space, in Fig. 2.4. This idea of overlaying the SOM lattice with the induced Delaunay graph was proposed by Taşdemir and Merényi in [30]. Here it enables the development of a versatile interactive tool, TopoView, which is one contribution of this work and will be described in Section 3.4. The comparison of the topologies defined by $\widetilde{D}$ and A was developed into a measure of topology violation, the Topographic Function (TF), by Villmann *et al.* [19]. We improve the TF to more refined measures, as will be discussed in Section 3.3.

Connectivity matrix ($CONN$ matrix)

In the induced Delaunay graph $\widetilde{D}$, an edge can be established between two prototypes by even a single data sample that selects these two prototypes as the BMU and the second BMU. As a result, noise or outliers in the data can easily obscure the discontinuities in the manifold structure. This situation is illustrated in Fig. 2.6 through the 2-dimensional "Clown" data set created by Vesanto and Alhoniemi in [31]. As seen in Fig. 2.6, top left, this data set has clusters of different sizes, shapes, and densities, to mimic a clown's face. The wide variation in the statistical properties of the clusters as well as overlaps between the clusters make the extraction of structure difficult, especially the extraction (or separation) of the three small subclusters in the left eye. A 17$\times$19 hexagonal SOM was used to learn this data set by the authors of [31]. In Fig. 2.6, both a variant of the U-matrix visualization (bottom left) and the induced Delaunay graph (top middle) help separate the coarse structure of this "Clown" data manifold (the two eyes, the nose, the mouth and the body), but neither of them is able to delineate the three subclusers in the left eye. For more precise structure identification (including distinction of noise from relevant information),

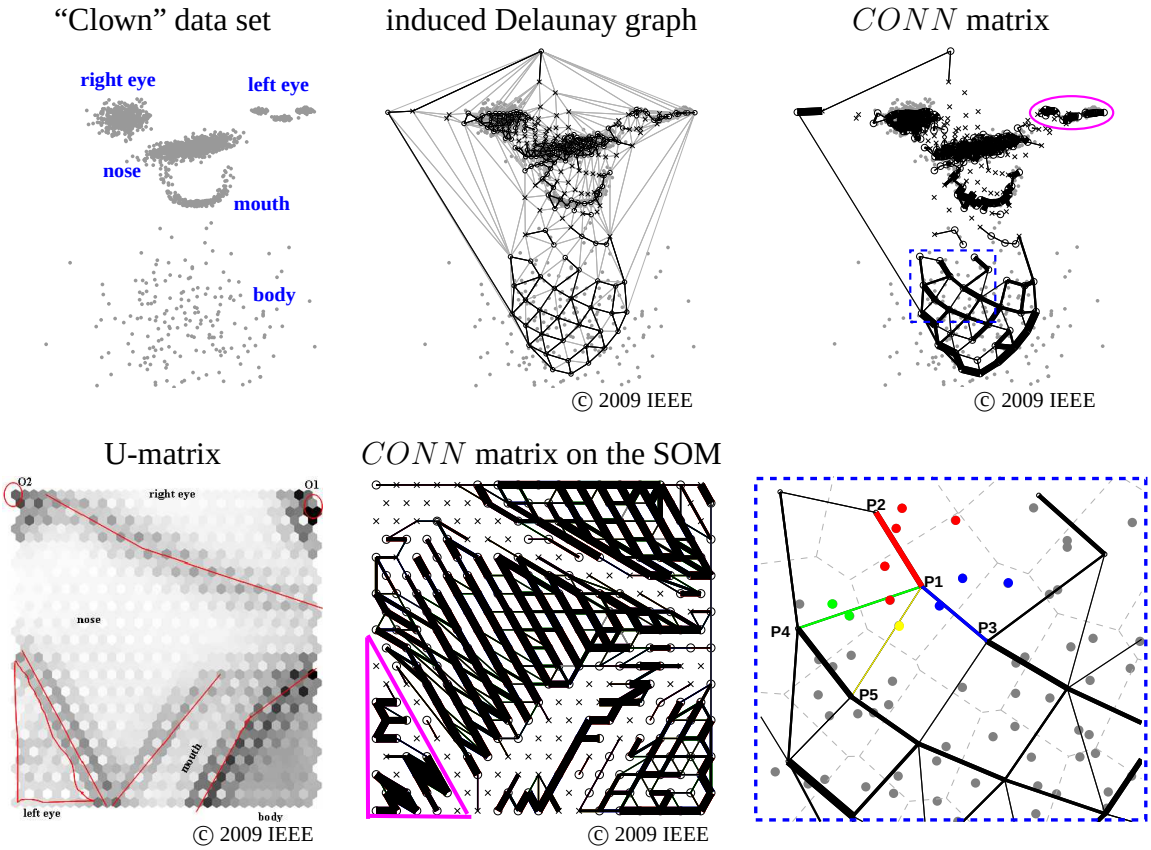


Figure 2.6: Visualization of the 2-dimensional “Clown” data set from [31] and the 17×19 hexagonal SOM learned with it. Solid dots are data samples. Open circles and crosses are non-empty and empty prototypes, respectively. Top middle, top right, bottom left and bottom middle are reproduced from [24], with kind permissions of IEEE and the authors, K. Taşdemir and E. Merényi. **Top left:** The 2-dimensional “Clown” data set from [31]. **Top middle:** The induced Delaunay graph (black lines) visualized in the data space highlights most of the discontinuities in the manifold structure, which are not delineated by the Delaunay graph (gray lines). **Top right:** The $CONN$ matrix, visualized in the data space, make detailed structures (e.g., the three subclusters in the left eye) emerge. Part of the clown’s body, in the dashed square, is magnified in the bottom right. **Bottom left:** A variant of the U-matrix visualization of the SOM. The SOM neurons are shown as hexagons. The grey scale intensity of the additional hexagon between each pair of neurons adjacent in the lattice indicates the Euclidean distance, in the data space, between the two respective prototypes. A darker gray indicates greater dissimilarity. This visualization delineates the coarse cluster structure. **Bottom middle:** The $CONN$ matrix drawn on the SOM lattice separates the large clusters, and also makes the three subclusters in the left eye (in the magenta triangle) emerge. **Bottom right:** Magnified detail in the visualization of the $CONN$ matrix from boxed area in the clown’s body (top right). The first to fourth ranking Voronoi neighbors of prototype P1 are P2, P3, P4 and P5, which have strengths 5, 3, 2 and 1, respectively. To make this easy to see, we color the data samples that contribute to each of the 4 connections, the same as their corresponding connections.

Taşdemir and Merényi proposed a new idea in [30, 24]. They defined the *connectivity matrix* (CONN matrix), which assigns weights to the edges of the induced Delaunay graph as in eq. 2.16, thereby emphasizing the connections that are established by a large number

of samples.

$$CONN(i,j) = \#\{\mathbf{x} \in M : \mathbf{w}_i \text{ and } \mathbf{w}_j \text{ form a pair of BMU and second BMU for } \mathbf{x}\} \tag{2.16}$$

$CONN(i,j)$ is called the *connection strength* between the prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$. The $CONN$ matrix reflects the anisotropic data distribution in the Voronoi cells of the prototypes, as seen in Fig. 2.6, bottom right. This information can be used to interpret the similarity relationships between the prototypes: the stronger two prototypes are connected, the more similar these two prototypes should be, or the more common information of the data they share. With the $CONN$ matrix, the discontinuities obscured by noise can emerge. In Fig. 2.6, top right, the separations between the three subclusters in the left eye (in the magenta oval) become obvious. When drawn on the SOM lattice, as in Fig. 2.6, bottom middle, the $CONN$ matrix is equally helpful in separating the three subclusters in the left eye (in the magenta triangle). Owing to the detailed topology information represented by the $CONN$ matrix, we will use it in the improvement of measures of topology preservation in Chapter 3.

The $CONN$ matrix contains additional information that can be used to express local relationships between the prototypes. These are the rankings of the Voronoi neighbors to each prototype according to their respective connection strengths, proposed by Taşdemir and Merényi in [24]. For example, in Fig. 2.6, bottom right, P2, P3, P4 and P5 are the first to the fourth ranking Voronoi neighbor of the prototype P1, and the respective 4 connections are the first to the fourth ranking connection to P1. The connection strengths and the rankings can be visualized together, as in the CONNvis visualization [24], simultaneously providing a view of global and local connectedness in the manifold. Although rankings were not used in the "Clown" illustration in Fig. 2.6, we will use them later in the customization of the use of the SOM's knowledge for supervised learning in Chapter 4.

Chapter 3

New tools for monitoring the faithfulness in representation of manifold structure by SOMs

Material based on:

- L. Zhang and E. Merényi, "Weighted Differential Topographic Function: a refinement of Topographic Function", *Proc. 14th European Symposium on Artificial Neural Networks (ESANN 2006) Bruges, Belgium, April 26–28*, 7–12, 2006.
- E. Merényi, K. Taşdemir, and L. Zhang, "Learning highly structured manifolds: harnessing the power of SOMs", Chapter in *Similarity based clustering*, Lecture Notes in Computer Science (Eds. M. Biehl, B. Hammer, M. Verleysen, T. Villmann), Springer-Verlag, LNAI 5400, 138–168, 2009.

3.1 Measuring the goodness of SOMs

Generally, there are two criteria for quantifying the goodness of the mapping formed by the SOM algorithm. One is the accuracy of the mapping, or how closely the prototypes follow the *pdf* or local structure of the input manifold. This criterion is commonly used for vector quantization algorithms. The other criterion is the quality of topology preservation, which is an important property that enables the correct identification of manifold structure from the SOM. Both criteria are important but neither is quite straightforward to evaluate. We will briefly review the two criteria next with an emphasis on measuring the quality of topology preservation, which is one focus of this thesis work.

One of the commonly chosen measures of mapping accuracy is the *quantization error* E_{qe} (eq. 3.1), which is calculated as the sum of the squared distances from data samples to their respective closest prototypes, over all data samples [5].

$$E_{qe} = \sum_{i \in A} \sum_{\mathbf{x} \in V_i} \|\mathbf{x} - \mathbf{w}_i\|^2 \tag{3.1}$$

where the prototype $\mathbf{w}_i$ is the centroid of the Voronoi cell V_i. This measure quantifies the quality of approximation of the *pdf* of the data, which is one aspect of SOM learning. The other aspect of the SOM, topology preservation, is also important and related measures are desirable.

As we have illustrated in Section 2.2, topology preservation is an essential property of an ideal SOM, and this property is necessary for correct interpretation of the manifold structure. However, in reality topology violation is not unusual. There are two common reasons for the occurrence of topology violations. First, parametrization of the learning process influences the development of the map. A simple example is that a neighborhood function with a too rapidly shrinking size or with a too small initial size can cause twists in the map (e.g., Fig. 2.2, right). As another example, in the Conscience algorithm, the additional "conscience" component brings in two more parameters, $\beta(t)$ and $\gamma(t)$ (eqs. 2.10 and 2.11), to the system. Topology violations can occur if the parameters are not scheduled in proper ranges or not suitable for each other. Second, while we constrain the dimension of the SOM lattice to 2 and enjoy the convenience of visual inspection and easy digestion, the dimensional mismatch between the lattice and the data space may result in topology violations. When the data has larger dimension than the SOM lattice, the map has to fold itself to better fill the manifold space, in compensation for the insufficiency in dimension. Imagine that, when a 1-dimensional SOM learns a 2-dimensional distribution, the sting of SOM prototypes has to distort itself to span the 2-dimensional data manifold. In SOMs

learned with real data, which can not only be high-dimensional, but may also have complicated structure, topology violations are common and may occur at all steps during the learning. All the above issues underline the need for measuring and monitoring of topology preservation in SOMs, for achieving good learning. The measures should be sensible quantifications of the topology violations. Preferably the measures should also be normalized in a certain way such that they can be used to compare different SOMs or to monitor an SOM at different learning steps. In addition, visual inspection of topology violations regardless of data dimension is desirable, because the visualization of a learned map in the data space (as seen in Fig. 2.4, right) has limited applicability (data dimension ≤ 3). With these tools, we can then find the map that represents the manifold structure most faithfully, from maps resulting from runs with different SOM sizes, learning parameters, or learning steps.

Quantification of the quality of topology preservation, however, is nontrivial. According to the definition of topology preservation, as introduced in Chapter 2, there are three basic elements in the design of a measure: distance metrics used to quantify neighborhood relations in the data space and in the lattice space; a proper mathematical interpretation of perfect topology preservation; and the quantification of topology violations. We will review pervious measures according to these three aspects.

3.2 Review of previous measures of topology preservation

Distance metric to quantify neighborhood relations

We remind the reader to distinguish the distances in the data space $M \in R^d$ and in the lattice A. For example, the Euclidean distance between two d-dimensional prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$ in the data space is $\|\mathbf{w}_i - \mathbf{w}_j\|_E$, while the Euclidean distance between them in the lattice indicates $\|\mathbf{r}_i - \mathbf{r}_j\|_E$, where $\mathbf{r}_i$ and $\mathbf{r}_i$ are the locations of the prototypes in the SOM

lattice A.

In both the data space and the SOM lattice, the most frequently used metric is the Euclidean distance (eq. 2.6). Although the Euclidean metric works well for data with continuous and linear distribution, it is unable to correctly express the neighborhood relationships for data with discontinuities and nonlinearities. An example of such a situation is given in Fig. 3.1. A 1-dimensional SOM of seven prototypes is used to learn a 2-dimensional uniform distribution in a thin "horseshoe" (delineated by solid lines). In Fig. 3.1, the learned prototypes (black dots) are plotted in the data space and connected to their respective immediate lattice neighbors by dashed lines. We can see that the prototypes are organized in the data space such that they represent the curved shape of the "horseshoe" manifold. The discontinuity between the two ends of the "horseshoe" is correctly represented by the SOM. The Euclidean distance, however, is unable to express this discontinuity. It will indicate that the prototypes 1 and 7 are Voronoi neighbors in the data space because they are the closest to each other. This will lead to the false conclusion that the topology is violated in the SOM: the seeming Voronoi neighbors, prototypes 1 and 7, are not adjacent in the SOM lattice. Therefore, the measures that rely on the Euclidean metric can falsely penalize the seeming topology violations, which are actually caused by discontinuities and nonlinearities in the manifold. This drawback is remedied in the Topographic Function (TF), proposed by Villmann *et al.* [19]. The TF adopts a graph distance metric based on the induced Delaunay graph [20] to faithfully express the connectedness (adjacency) between SOM prototypes in the data space. This distance metric was later applied to other measures, such as the improved Topographic Product (improved TP) [32] and the Topographic Error (TE) [33]. In addition, for rectangular SOM lattices, the TF also uses two different metrics in the lattice: maximum norm $\|\cdot\|_{max}$ (eq. 2.7) for evaluation of forward violations (i.e., prototypes that are neighbors in the data space M are not adjacent in the

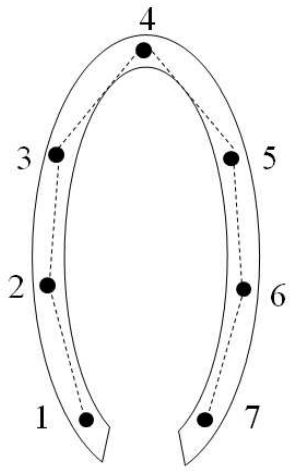


Figure 3.1: The SOM prototypes (black dots) that represent a 2-dimensional "horseshoe" distribution. The prototypes are projected back into the data space and connected according the SOM lattice structure.

lattice A) and Euclidean norm $\|\cdot\|_E$ for evaluation of backward violations (i.e., prototypes that are adjacent in the lattice A are not neighboring in the data space M). The purpose of using two metrics is to tolerate small distortions in the map, which do not prevent topology preservation. We provide details on the TF in Section 3.2.2.

Mathematical definition of perfect topology preservation

While seemingly sharing a common intuition about topology preservation, people rely on different mathematical definitions of perfect topology preservation when proposing measures. The different definitions can be based on three different types of similarity in the mapping between A and M: *metrics*, *ranking* and *continuity* [34].

Perfect topology preservation based on *metrics*, the strictest similarity type, requires the preservation of pairwise metric distances in the mapping. One measure based on this definition is the Pearson correlation [35], which calculates the correlation coefficient of the pairwise distances in the data space and in the lattice space, across all data samples. A mapping that preserves all distances produces a value of 1 for the correlation coefficient.

This measure is suitable for algorithms that are designed to preserve the metric distances, such as metric-multidimensional scaling (MDS) algorithms [36] (e.g., Sammon's mapping [37]). For the SOM, however, it is obviously unsuitable, since SOM learning by design is not intended to preserve distances.

Perfect topology preservation measured by *ranking* is a relatively relaxed definition, which requires the preservation of the rankings of pairwise distances. We will call these measures ranking-based measures for short. One example is the Spearman's correlation coefficient ρ [38], which calculates the correlation of the rankings of the pairwise distances in the data space and in the lattice space. Another example is the Topographic Product (TP) [39], which relies on the ratios of the distances between each prototype and its neighbors of the same rank, in the data space and in the lattice space, respectively. Details of the TP will be reviewed in Section 3.2.1. Rather than ranking neighbors for the prototypes, another example, the König's measure, ranks neighbors for each data sample according to their distances to the sample in the input and output spaces and sets credit scores according to the differences in the ranking [40]. Some other measures in this category are the improved TP [32] and the Directional Product (DP) [41]. One difficulty with the ranking-based measures is the large number of ties among the pairwise distances between the neurons in the SOM lattice. For example, in a 2-dimensional rectangular SOM, a neuron has 8 equally close immediate lattice neighbors. These measures need to correctly decide the order of them to avoid false penalty for nonexistent topology violations. However, in implementation, a random order or a predefined order is often used to rank the ties (the neighbors that are equally close), for the sake of computational cost. As a result, the measures can be incorrect.

A more suitable definition of perfect topology preservation is based on *continuity*, which focuses on the neighborhood structure rather than on the neighbor ranking. It re-

quires the preservation of the immediate (nearest) neighbor(s) in the mapping, and therefore prevents the struggle of ordering the large number of ties in the ranking-based measures. In SOM learning, perfect topology preservation means that the SOM prototypes neighboring in the data space M should be also neighboring (adjacent) in the lattice A, and vice versa, as was introduced in Section 2.1. Two example measures based this interpretation are the Zrehen-measure [42] and the V-measure [43]. The Zrehen-measure sums up the number of "intruders", in the data space, between each pair of lattice-neighbor prototypes. The V-measure assigns a penalty score for each non-lattice neighbor whose distance to a given prototype is smaller than the distance between the prototype and any of its immediate lattice neighbors. Both the Zrehen-measure and the V-measure, however, capture only the forward violations, but not the backward violations. A better measure in this category is the TF [19], which accumulates violations in both mapping directions (Section 3.2.2).

Quantification of topology violations

After perfect topology preservation is mathematically defined, as discussed above, the final step in the design of a measure is to formulate a cost function that quantifies topology violations. One fundamental difference between cost functions is the level of the details in the information incorporated in them. The incorporated information can be on the level of prototypes, on the level of data distribution, or on a mixed level of the two.

Cost functions that depend only on the prototypes are computationally economical because the number of prototypes is often much smaller than the number of data samples. Examples of this type of measure are the TP, the DP, Spearman's ρ and the Zrehen-measure, which evaluate the inter-prototype violations and ignore the relationships between data samples. Without exploiting the detailed relationships among the data samples, these measures may be insufficient when dealing with noisy or complicated data.

Alternatively, the cost function can be formulated to depend on the relationships among the data samples only. For example, a cumulative histogram method was proposed by De Bolt *et al.* to capture a statistical view of the neighborhood status of the system [44]. The histogram shows the percentage of sample pairs as a function of pairwise distance. The authors used the histogram of an unordered map as a baseline to evaluate the reliability of any given SOM. The more dissimilar the histogram of an SOM was to the histogram of the unordered map, the more reliable the map was considered. However, it is unclear how the difference in the histogram can be interpreted quantitatively in terms of topology violations.

Another possibility is the joint consideration of the prototypes and the relations of data samples in the cost function. For example, the TF screens the topology violations by checking the neighboring relationships between the prototypes across the data space and the lattice. This screening involves only the inter-prototype relationships. However, the neighboring relationships across the prototypes are determined by the induced Delaunay graph, which can be constructed from the data distribution and the prototypes. The TF therefore implicitly utilizes the detailed data distribution. Another measure, TE [33], not only implicitly uses the data distribution the same way as the TF does, but also explicitly uses it in the cost function. It computes a percentage of data samples that contribute to violations, and thereby provides a detailed view, on the level of data distribution. However, the TE does not show the quality of topology preservation as a function of the scope of violations, which makes it less informative than the TF. Another interesting example of this type of measure is the Kaski-Lagus measure [45], which adds up the quantization error and a minimum-path distance between the BMU and the second BMU, across all data samples. The minimum path was defined as the shortest path in the data space, consisting of a string of prototypes, each of which is an immediate lattice neighbor of its predecessor

in the path. This measure, however, was only examined for 1-dimensional maps, but not for high-dimensional data, in [45].

We now discuss the Topographic Product (TP) and the Topographic Function (TF). The TP is one of the earliest measures. It uses the Euclidean norm as its distance metric. The TF is a more advanced measure, which uses a better distance metric than the Euclidean norm. Through the introduction of these two measures, we want to show the reader how the TF is better than the TP and therefore justify our choice of the TF as the basis for developing new measures. The author also implemented these two measures in our software environment at Rice University. We will compare them with our proposed measures through applications later in the chapter.

3.2.1 Topographic Product (TP), one of the earliest measures

The basic idea of the TP, proposed by Bauer and Pawelzik [39], is to establish, for any neuron $j \in A$, two ordered lists of the neurons $i \in A$ ($i \neq j$), according to the distances between the neurons in the lattice space and the distances between the corresponding prototypes in the data space, respectively. That is to say, with respect to the neuron j, we rank the neurons i ($i \neq j$), as the first, second, ... lattice neighbors of j according to $\|\mathbf{r}_i - \mathbf{r}_j\|_E$. Let $n_p^A(j)$ denote the neuron index of the *pth lattice neighbor* of neuron j in the SOM lattice A. Similarly, we rank the neurons i ($i \neq j$), as the first, second, ... neighbors of j in the data space according to $\|\mathbf{w}_i - \mathbf{w}_j\|_E$. Let $n_p^M(j)$ denote the neuron index of the *pth neighbor in the data space M*. Define two ratios Q_1 and Q_2 as follows:

$$\begin{aligned} Q_1(j,p) &= \frac{\|\mathbf{w}_j - \mathbf{w}_{n_p^A(j)}\|_E}{\|\mathbf{w}_j - \mathbf{w}_{n_p^M(j)}\|_E} \\ Q_2(j,p) &= \frac{\|\mathbf{r}_j - \mathbf{r}_{n_p^A(j)}\|_E}{\|\mathbf{r}_j - \mathbf{r}_{n_p^M(j)}\|_E} \end{aligned} \tag{3.2}$$

The TP is then defined as an average over all prototypes and over neighbors of all ranks:

$$TP = \frac{1}{N \times (N-1)} \sum_{j=1}^{N} \sum_{p=1}^{N-1} log[(\prod_{l=1}^{p} Q_1(j,l) \times Q_2(j,l))^{\frac{1}{2p}}] \tag{3.3}$$

By design, the sign of the TP indicates the relation between the dimensions of the input space and output SOM lattice space. A positive value of the TP indicates too low dimension of the output (lattice) space for the input data, and a negative value of the TP indicates too high dimension of the output space. A near-zero TP value is supposed to correspond to an approximate dimensional match. However, since the TP uses the Euclidean metric to quantify the similarity between the prototypes, seeming violations caused by nonlinearities in the manifold are incorrectly penalized. The TP in such cases may not indicate the true quality of topology preservation. We will show examples later in this Chapter.

3.2.2 Topographic Function (TF), a measure that treats nonlinearities correctly

The TF, by Villmann *et al.* [19], uses the induced Delaunay graph $\widetilde{D}$ (introduced in Section 2.2.2) to characterize the neighboring relationships between the prototypes. A graph distance metric, denoted by $\|\cdot\|_{\widetilde{D}}$, is used to compute the inter-prototype distances in the data space. The graph distance between two prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$ in the data space, $\|\mathbf{w}_i - \mathbf{w}_j\|_{\widetilde{D}}$, is defined as the length of the minimum path between them in $\widetilde{D}$. The graph distance between any two prototypes connected by an edge in $\widetilde{D}$ is defined as 1. Fig. 3.2 illustrates a minimum path between two prototypes in the "exclamation mark" data set. Since we can visualize the induced Delaunay graph both in the data space and in the lattice space, as shown in Fig. 2.5, middle and right, we show the minimum path (blue line segments) between the prototypes A and B in both the spaces, as well, in Fig. 3.2. The

prototypes along the path are numbered so that we can relate them across the two spaces. The minimum path between A and B has a length of 5, so the graph distance between A and B is 5. The graph distance between the prototypes B and C is obviously 1, because they are immediately connected by an edge in $\widetilde{D}$.

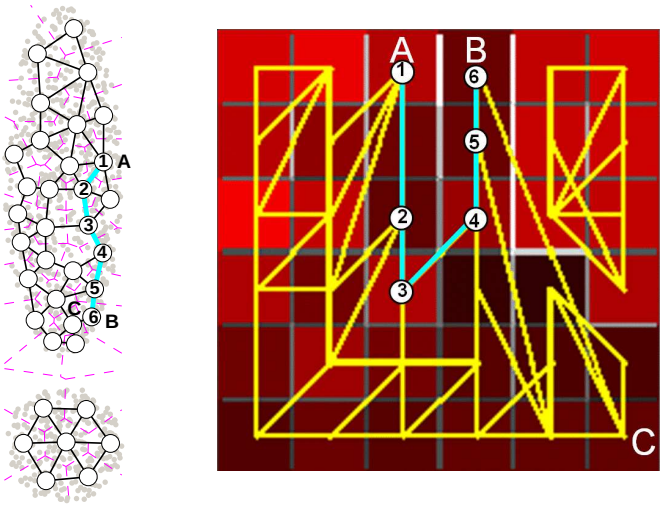


Figure 3.2: Illustration of a minimum path (green line segments) between two prototypes A and B, in the data space (on the **left**) and in the SOM (on the **right**), respectively, through the "exclamation mark" data set. The induced Delaunay graph is shown as black lines in the data space and as yellow lines in the SOM. The minimum path between A and B has a length of 5, so the the graph distance between them is 5. The prototypes along the path are numbered so that we can relate these prototypes across the data space and the lattice space.

Next we will review how the TF was defined for a commonly used rectangular lattice in [19]. First, the authors defined forward and backward violations rigorously. A *forward topology violation* is defined between two prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$, which are immediate neighbors in the data space ($\|\mathbf{w}_i - \mathbf{w}_j\|_{\widetilde{D}} = 1$) but have a maximum distance (city block distance) larger than 1 in the SOM lattice ($\|\mathbf{r}_i - \mathbf{r}_j\|_{max} > 1$). Forward topology violations are also called *violating connections*. A *backward topology violation* is defined between $\mathbf{w}_i$ and $\mathbf{w}_j$ when they are immediate lattice neighbors in the SOM lattice ($\|\mathbf{r}_i - \mathbf{r}_j\|_E = 1$) but have a graph distance in the data space larger than 1 ($\|\mathbf{w}_i - \mathbf{w}_j\|_{\widetilde{D}} > 1$). The reader may notice that the mathematical interpretations of "immediate lattice neighbors"

are different in the above two definitions. The definition of forward violations uses the Euclidean distance, while the definition of backward violations uses the maximum distance. This helps avoid penalizing unimportant forward violations caused by slight distortions in the map. More details can be found in [19]. The authors then defined a function $f_i(fl)$ for each neuron i, where fl is called the *folding length* of a topology violation, and it represents the *scope* or *range* of violation.

$$f_i(fl) \stackrel{def}{=} \begin{cases} \#\{j|\ \|\mathbf{r}_i - \mathbf{r}_j\|_{max} > fl \wedge \|\mathbf{w}_i - \mathbf{w}_j\|_{\tilde{D}} = 1\} & 1 \leq fl \leq \max\limits_{i,j \in A} \|\mathbf{r}_i - \mathbf{r}_j\|_{max} \\ \#\{j|\ \|\mathbf{r}_i - \mathbf{r}_j\|_{E} = 1 \wedge \|\mathbf{w}_i - \mathbf{w}_j\|_{\tilde{D}} > |fl|\} & -\max\limits_{i,j \in A} \|\mathbf{w}_i - \mathbf{w}_j\|_{\tilde{D}} \leq fl \leq -1 \end{cases} \tag{3.4}$$

A positive fl denotes the folding length of a forward topology violation between two prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$, which are immediate neighbors in the data space ($\|\mathbf{w}_i - \mathbf{w}_j\|_{\tilde{D}} = 1$) but have a maximum norm of fl in the SOM lattice ($\|\mathbf{r}_i - \mathbf{r}_j\|_{max} = fl$). A forward topology violation is equivalent to a *folding* of the SOM lattice in the data space, i.e., two distant prototypes in the lattice are folded together in the data space. fl indicates the *range* of that folding. This is why fl is called folding length. Similarly, a negative fl denotes the folding length of a backward topology violation between two prototypes $\mathbf{w}_i$ and $\mathbf{w}_j$, which are immediate lattice neighbors in the SOM lattice ($\|\mathbf{r}_i - \mathbf{r}_j\|_E = 1$) but have a graph distance of $|fl|$ in the data space ($\|\mathbf{w}_i - \mathbf{w}_j\|_{\tilde{D}} = |fl|$). Likewise, a backward topology violation is equivalent to a *folding* of the data manifold in the SOM lattice. For example, in the SOM learned with the "exclamation mark" data set, as seen in Fig. 3.2, we can easily see a forward violation between B and C ($\|\mathbf{w}_B - \mathbf{w}_C\|_{\tilde{D}} = 1$ and $\|\mathbf{r}_B - \mathbf{r}_C\|_{max} > 1$) and a backward violation between A and B ($\|\mathbf{r}_B - \mathbf{r}_C\|_{max} = 1$ and $\|\mathbf{w}_B - \mathbf{w}_C\|_{\tilde{D}} > 1$). The folding length of the forward violation between B and C is 5 since $\|\mathbf{r}_B - \mathbf{r}_C\|_{max} = 5$. The folding length of the backward violation between A and B is also 5 since $\|\mathbf{w}_A - \mathbf{w}_B\|_{\tilde{D}} = 5$.

In the positive domain, $f_i(fl)$ counts the number of forward topology violations be-

tween a given prototype $\mathbf{w}_i$ and other prototypes, with folding length larger than fl. In the negative domain, $f_i(fl)$ counts the number of backward topology violations between a given prototype $\mathbf{w}_i$ and other prototypes with folding length larger than $|fl|$. The TF is then computed as an average of $f_i(fl)$ across all neurons $i \in A$ in [19], as in eq. 3.5, where N is the total number of neurons. $TF(0)$ is defined as the sum of $TF(-1)$ and $TF(1)$, which was interpreted as the total number of violations, including both the forward and the backward violations, in the lattice [19].

$$TF(fl) \stackrel{def}{=} \begin{cases} \frac{1}{N}\sum_{i\in A} f_i(fl) & fl > 0 \\ TF(1) + TF(-1) & fl = 0 \\ \frac{1}{N}\sum_{i\in A} f_i(fl) & fl < 0 \end{cases} \tag{3.5}$$

A large fl corresponds to long-range folding, which we will also call *global violation*. Similarly, a small fl indicates short-range folding, which we will also call *local violation*. These definitions of the global and local violations are qualitative. We will provide their rigorous definitions, proposed by Taşdemir and Merényi in [24], later in Section 3.4. The largest fl in the positive domain that holds a non-vanishing value of the TF indicates the longest folding length of forward violations, and the largest $|fl|$ in the negative domain with a nonzero value of the TF corresponds to the longest folding length of backward violations. Villmann *et al.* also proposed to normalize fl to $[-1, 1]$, to allow comparison of SOMs with different lattice structures [19].

3.3 New, refined measures

3.3.1 Differential Topographic Function (DTF)

The TF is an integral function because of the inequalities (“$>$”) used in the definition (eq. 3.4). It is not informative when we are interested in the number of violations with a specific folding length, fl. This motivates us to define a differential form of the TF, which we name the Differential Topographic Function (DTF) [46]. Similarly to eq. 3.4, we first define a function $g_i(fl)$ for each neuron i:

$$g_i(fl) \stackrel{def}{=} \begin{cases} \#\{j|\ \|\mathbf{r}_i - \mathbf{r}_j\|_{max} = fl \wedge \|\mathbf{w}_i - \mathbf{w}_j\|_{\widetilde{D}} = 1\} & 2 \leq fl \leq \max\limits_{i,j \in A} \|\mathbf{r}_i - \mathbf{r}_j\|_{max} \\ \#\{j|\ \|\mathbf{r}_i - \mathbf{r}_j\|_{E} = 1 \wedge \|\mathbf{w}_i - \mathbf{w}_j\|_{\widetilde{D}} = |fl|\} & -\max\limits_{i,j \in A} \|\mathbf{w}_i - \mathbf{w}_j\|_{\widetilde{D}} \leq fl \leq -2 \end{cases} \tag{3.6}$$

The DTF is then computed as the average of $g_i(fl)$ over all neurons as in eq. 3.7. Obviously, the DTF can also be obtained directly by the first difference of the TF. The DTF enables the comparison of the numbers of violations, which are also called the *extents of violations*, across different scopes of violations.

$$DTF(fl) \stackrel{def}{=} \frac{1}{N} \sum_{i \in A} g_i(fl) = \begin{cases} TF(fl-1) - TF(fl) & fl \geq 2 \\ TF(fl+1) - TF(fl) & fl \leq -2 \end{cases} \tag{3.7}$$

To illustrate the detailed information revealed by the DTF and to compare it with the TF, we use an 8-class 6-dimensional synthetic spectral image created by Merényi, and described in [27] (Fig. 3.3). Each of the 128×128 pixels in the image is a 6-dimensional vector. Two of the classes have 4096 data samples (pixels) each, two others have 2048, and the remaining four classes have 1024 data samples. Approximately 10% Gaussian noise was added to each of the 8 representative spectra to create within-class variations. For

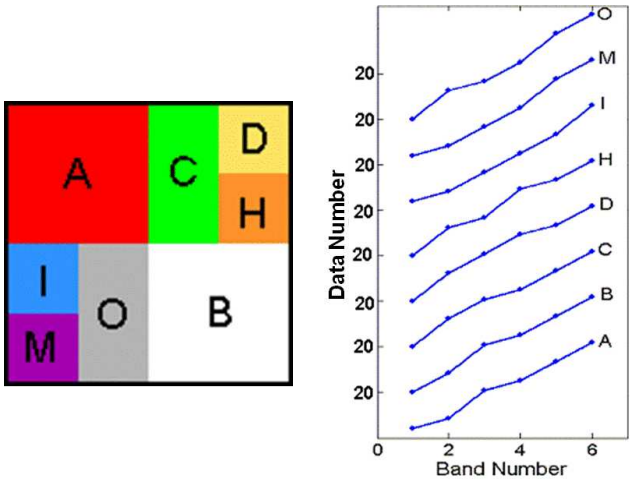


Figure 3.3: The 8-class 6-dimensional (6-band) synthetic spectral image data set. Figures reproduced from http://terra.ece.rice.edu/data_example/data.html, with permission of E. Merényi. **Left**: Spatial distribution of the 8 classes in the 128×128 image, overlain with known labels (colors). **Right**: Mean signatures (means of the data samples) for each class, vertically offset for clarity.

complete description of this data set, see [27]. A 15×15 SOM trained with this data set is shown in Fig. 3.4. In our evaluation of the measures and tools in Section 3.3–3.4 (Fig. 3.4–3.6), we use this SOM and the cluster structure, provided by Merényi. The resulting SOM is visualized with the mU-matrix in Fig. 3.4. We remind the reader that the mU-matrix visualization shows the Euclidean distance in the data space between each pair of lattice-neighbor prototypes, as a fence between the two SOM grid cells that represent the two respective prototypes. In Fig. 3.4, left, the intensity of the monochrome red color in a grid cell is proportional to the mapping density in that cell. We can see that the SOM is separated clearly into 8 clusters by double-fenced "corridors". The black cells in those corridors represent empty neurons (with no data mapped to them). In Fig. 3.4, right, the known class labels are overlain on the SOM, so that we can compare the clusters that emerge from the mU-matrix visualization with the ground truth. The known labels are, of course, not used in SOM learning. From this comparison, we can conclude that the SOM has learned the cluster structure of this synthetic data set well.

Next we use the TF and the DTF to evaluate the quality of topology preservation of

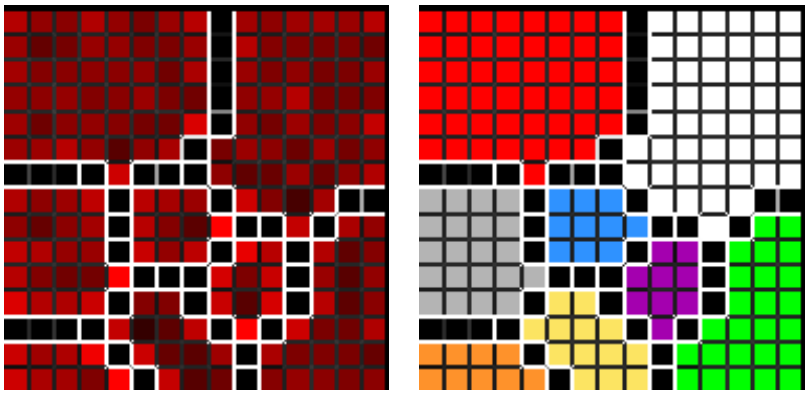


Figure 3.4: The mU-matrix visualization of the SOM learned with the 8-class 6-dimensional synthetic data set (Fig. 3.3). The SOM is shown as a lattice of grid cells. The Euclidean distance in the data space between any two prototypes that are immediate lattice neighbors is shown as a fence on the boundary of the two respective grid cells of the two prototypes, in a gray scale intensity proportional to the distance. White is large distance. Figures reproduced from [27], with kind permissions of IEEE and E. Merényi. **Left**: Each cell is shaded by an intensity of red proportional to the number of data samples mapped to the corresponding neuron. Black cells represent empty neurons. The cluster boundaries emerge through the mU-matrix (white fences). **Right**: The known class labels are overlain on the grid cells.

the SOM. Both measures are computed in two ways, once with all neurons included, and once with the empty neurons excluded. The two TFs, computed with and without empty neurons, overlap in the positive domain ($fl > 0$), in Fig. 3.5, top left. The largest positive fl with a nonzero TF value is 7, indicating the largest folding length of all forward violations is 8. In the negative domain ($fl < 0$), the exclusion of empty neurons makes the TF vanish for all negative values of fl. This means that all the backward violations are related to empty neurons. To help understand the measures, we visualize the induced Delaunay graph $\widetilde{D}$ on the SOM in Fig. 3.5, bottom right. Any pair of prototypes connected by a yellow line segment are Voronoi neighbors in the data space. $\widetilde{D}$ clearly delineates the discontinuities between the 8 clusters. The empty neurons (black cells) have no connection to their immediate lattice neighbors, which indicates that they cause the backward violations, as seen from the TF in Fig. 3.5, top left. Looking more closely, we see that these empty neurons have no connection to any other neurons. Mathematically this can be expressed as: the graph distance from any empty neuron to any other neuron is infinity. This explains

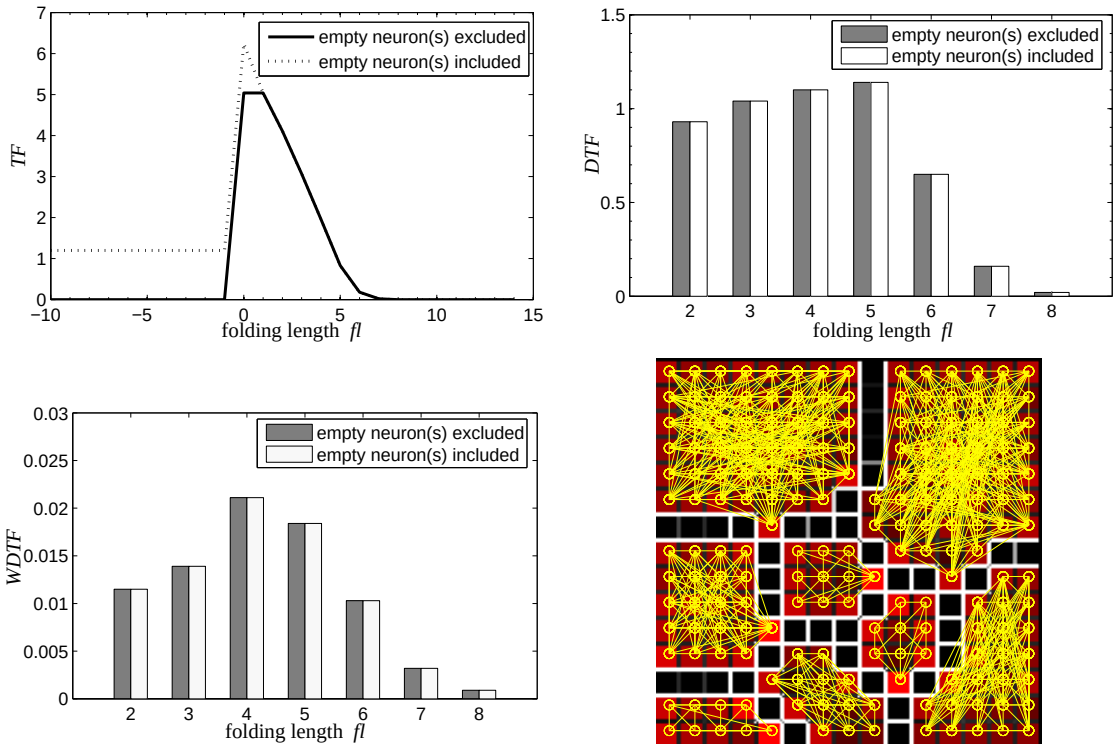


Figure 3.5: Measuring topology violations in the SOM learned with the 8-class 6-dimensional synthetic data set, with different measures, the TF, the DTF and the $WDTF$. All measures are calculated with and without empty neurons, respectively. **Top left**: The TF. **Top right**: The DTF. **Bottom left**: The $WDTF$. **Bottom right**: The induced Delaunay graph (yellow lines) is overlain on the SOM to help understand the values of the TF, DTF and $WDTF$.

the constant value shown in the negative domain of the TF when the empty neurons are included. The TF does not express the extent of violations (the number of violations) for a specific folding length, fl, while the DTF (Fig. 3.5, top right) clearly shows the relative extents of violations across different fl. The backward violations that involve the empty neurons are not shown by the DTF because the folding length of these violations is infinity ($fl = -\infty$).

In general, there are two types of empty neurons. One type is sometimes called interpolating neuron because it is often found at the boundaries of clusters. These empty neurons learned from the data, but were left empty at the end of the learning because the representations of data samples in the SOM contracted to a lighter group of prototypes as the SOM

converged. The empty neurons (black cells) shown in Fig. 3.4 are examples of interpolating neurons. These empty neurons help with the identification of clusters. The violations they induce do not hinder correct understanding of the manifold structure and are therefore negligible. The other type of empty neuron is one that remains inactive throughout the learning, i.e., it never had the chance to adapt its prototype. (No example for this type of empty neurons is shown here.) These empty neurons constitute the unused part of the map, and should be excluded from N, the total number of neurons, in the computation of the measures. The above discussions motivate us to exclude all empty neurons in the TF, the DTF and the other new measures that will be introduced next.

3.3.2 Normalized Differential Topographic Function ($NDTF$)

In the course of learning, the placement of the SOM prototypes in the data space is adjusted iteratively, resulting in the change of the connections across the prototypes. The total number of connections can then be different at different time steps of the SOM. We define the Normalized Differential Topographic Function ($NDTF$) [46] by incorporating the total number of connections as a normalization factor:

$$NDTF(fl) = \frac{N \times DTF(fl)}{2C} \qquad fl = 2, 3, ..., \max_{i,j \in A} \|\mathbf{r}_i - \mathbf{r}_j\|_{max} \tag{3.8}$$

where C is the total number of connections in the SOM. Normalized by $2C$, the $NDTF(fl)$ indicates the percentage of the connections at each folding length fl. It enables the comparison of the quality of topology preservation at different time steps of the SOM.

The $NDTF$ is defined only for forward violations ($fl > 0$) because backward violations can be easily detected from the SOM with the help of the mU-matrix and are hence not as detrimental as forward violations in cluster extraction. We remind the reader that, although manifold learning is not equivalent to cluster extraction, in most of our real world

applications the scientific goals are related to the finding of meaningful clusters in the data. Backward violations manifest in strong dissimilarities between immediate lattice neighbors (shown as high fences in the mU-matrix visualizations) and/or empty neurons on the cluster boundaries, as in Fig. 3.4, which actually help locate the clusters. In contrast, forward violations can lead to incorrect clustering. Imagine a map folds itself in the data space, so two prototypes that are distant in the lattice can actually represent similar data samples from the same cluster. However, since the two prototypes are separate in the lattice, they appear to represent data from two different clusters. Therefore, we choose to focus on the forward violations in the development of measures in this work.

3.3.3 Weighted Differential Topographic Function ($WDTF$)

The TF, the DTF, and the $NDTF$ express the extent of violations by a count of the violations at each folding length fl. However, these measures do not distinguish *severe* violations induced by a large number of data samples from violations caused by a few noisy samples. (We remind the reader that a connection can be induced by even a single sample, as in eq. 2.15.) In cases where data sets have noise and outliers, these measures thus do not reflect the *severity of violations*, which is quantified by the number of data samples involved in the violations. Motivated by this, we further resolve the DTF with an *importance weighting* of the connections to construct a new measure we call Weighted Differential Topographic Function ($WDTF$) [46] as follows. We first define a function $h_i(fl)$ for each neuron i:

$$h_i(fl) \stackrel{def}{=} \sum_{\substack{\|\mathbf{r}_i-\mathbf{r}_j\|_{max}=fl \\ \|\mathbf{w}_i-\mathbf{w}_j\|_{\tilde{D}}=1}} CONN(i,j) \qquad fl = 2,3,...,\max_{i,j\in A}\|\mathbf{r}_i-\mathbf{r}_j\|_{max} \tag{3.9}$$

where $\|\cdot\|_{\widetilde{D}}$ is the graph distance in the induced Delaunay graph. $CONN$ is the connectivity matrix, first proposed in [30]. The $CONN$ matrix expresses the connection strengths between the prototypes. The $WDTF$ is then computed for a given folding length fl as the sum of $h_i(fl)$, across all neurons, and the sum is normalized by the total number of data samples, P, as in eq. 3.10. The $WDTF$ expresses the severity of violations with folding length fl in terms of the percentage of contributing data samples.

$$WDTF(fl) = \frac{1}{2P}\sum_{i\in A} h_i(fl) \qquad fl = 2, 3, ..., \max_{i,j\in A}\|\mathbf{r}_i - \mathbf{r}_j\|_{max} \tag{3.10}$$

Note that the $WDTF$ is defined for forward violations (positive fl) but not for backward violations (negative fl) because of the lack of the counterpart of the $CONN$ matrix to quantify the connection strengths of backward violations. Nevertheless, this does not prevent the $WDTF$ from being a useful measure because forward violations are usually more harmful than backward violations for correct cluster extraction (as discussed in Section 3.3.2).

For the 8-class 6-dimensional synthetic data set, the $WDTF$ displays the severity of violations with folding lengths from 2 to 8 (Fig. 3.5, bottom left). If compared with the DTF in Fig. 3.5, top right, the $WDTF$ provides a more accurate evaluation of the relative importances of violations across different folding lengths. For example, while the DTF raises a red flag for $fl = 5$, indicating the maximum number of connections at this folding length, the $WDTF$ shows that the most severe violations occur at $fl = 4$, where the most data samples are contributing.

3.4 A new interactive visual monitoring tool – TopoView

In addition to the new measures discussed above, we also develop a useful interactive tool, TopoView, which allows to show meaningful subsets of connections (edges in the induced Delaunay graph) on the SOM lattice for capturing serious topological problems [4]. This is a similar visualization as seen in Fig. 2.5, right, and Fig. 3.5, bottom right. The crucial distinguishing function of TopoView is a set of versatile thresholding capabilities to filter out unimportant (weak) violations that may result from noise and outliers. It thereby shows the relevant or statistically significant sets of connections with improved visual clarity, compared to the plain visualization of the induced Delaunay graph on the SOM.

The graphical user interface of TopoView and a summary of thresholding keywords and their functionalities are given in Appendix B.2. We discuss here several basic and useful thresholding capabilities. For example, the subsets of connections to be shown by TopoView are selected by the user with a threshold for connection strength, a threshold for folding length, a choice of the category of connections (all, violating, or non-violating connections). The threshold for connection strength can be automatically computed by TopoView as a user-specified statistics of the connections, such as the mean strength of all connections or the mean strength of all violating connections. A useful threshold for folding length is the one that separates *global* and *local* violations. We remind the reader that at the end of Section 3.2.2 we described global and local violations qualitatively, as foldings in the map with long and short folding lengths, respectively. Taşdemir and Merényi proposed in [24] a rigorous definition for the folding length, l_{min}, that separates local and global violations. They computed l_{min} from the maximum number of Voronoi neighbors, m, to any prototype in the manifold, by eq. 3.11, for a rectangular SOM lattice [24]. Violations with $fl > l_{min}$ were defined as *global violations*, and those with $fl \leq l_{min}$ were

defined as *local violations* in [24].

$$l_{min} = \min\{l : m \leq \sum_{l'=1}^{l} 8l'\} \tag{3.11}$$

The argument in the computation of l_{min} from m by eq. 3.11 is that the m Voronoi neighbors of a prototype $\mathbf{w}_i$ should arrange themselves into the "tightest" SOM neighborhood of $\mathbf{w}_i$ in a topology preserving map (SOM lattice). In a rectangular SOM lattice, a prototype has 8 first-tier neighbors (8 equally closest neighbors), 16 second-tier neighbors (16 equally second closest neighbors), and so on. Eq. 3.11 therefore determines the smallest neighborhood size, l_{min}, that can accommodate m Voronoi neighbors. Another useful function of TopoView is to show the connectedness between clusters in the SOM, i.e., the similarity between clusters. The pre-requisite of using this function is the availability of cluster labels of the prototypes. The cluster labels can either come from ground truth or result from clusters extracted from the SOM. We define *inter-cluster* and *intra-cluster connections* as the connections with end points (prototypes) in the same cluster and in different clusters, respectively [4]. TopoView allows the user to show these two types of connections separately. This can be helpful in the evaluation of the correctness of learning or the validity of the cluster labels. For example, when the cluster labels are from ground truth and there are many strong connections between two different clusters, it may indicate either topology violations in the map or mislabeling. When the cluster labels result from cluster extraction from the SOM, strong connections between two clusters suggest that these two clusters may actually represent a single cluster.

To give an illustration of the thresholding functionalities of TopoView, we use the SOM learned with the 8-class 6-dimensional synthetic data set. As seen from the DTF and the $WDTF$ in Fig. 3.5, top right and bottom left, violations exist even at folding length 8 (i.e., nearly half the width of the SOM lattice), which suggests the map could be problem-

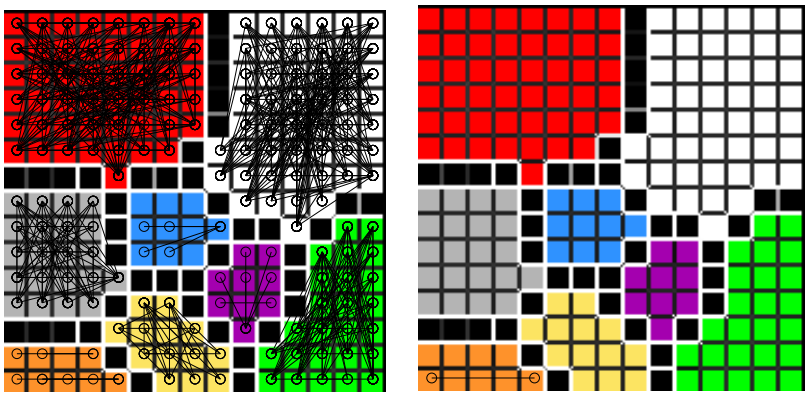


Figure 3.6: An example of displaying selected groups of violations with TopoView for the 15×15 SOM learned with the 8-class 6-dimensional synthetic data. The SOM upon which TopoView visualizes the connections, is reproduced from [27], with permissions of IEEE and Merényi. The violations are drawn on the SOM overlain with known class labels (colors) and mU-matrix. **Left**: All, 567, violations (black lines). **Right**: TopoView filters out connections with connection strength less than mean strength of all connections (15.8 in this example) and connections with folding length less than l_{min}, the maximum length of local violations (2 for this data set). There is one connection left with this thresholding method.

atic. TopoView then helps clarify where those violations are in the map. In Fig. 3.6, left, TopoView shows all violating connections (forward violations) as black lines on the SOM. All of these violations are located within the known clusters. 8 is the diameter of the two largest clusters (red and white) in the SOM. On the level of clustering, these violations are tolerable. To further emphasize the potentially serious violations, we use TopoView to filter out connections with low strengths or with short folding lengths. Here we set the threshold for connection strength as the mean strength of all connections (15.8 in this example), and set the threshold for folding length as the maximum length of *local* violations, l_{min} (2 in this example). l_{min} is computed from the maximum number of Voronoi neighbors of each prototype, m, by eq. 3.11. These two thresholds clear all connections shown in the left SOM except one violation in the lower left corner of the SOM, as seen in Fig. 3.6, right. This only violation is an intra-cluster violation, which does not prevent correct extraction of the clusters for this particular data set. In this example, TopoView helps with the quantitative analysis of the violations, from which we know that the SOM has achieved good

topological health for the purpose of cluster identification.

3.5 Applications of $WDTF$ and TopoView

Since SOMs learned with complicated data are often not free of violations, it is especially effective to use both the $WDTF$ and TopoView for the evaluation of topological conditions. The $WDTF$ provides a summary of the severity of violations at each folding length while TopoView provides localization of the violations in the SOM, for selected severity levels. Next, we will demonstrate the combined use of the tools on a 2-dimensional synthetic data set, a 194-dimensional real hyperspectral image, and a 210-dimensional synthetic hyperspectral image.

3.5.1 An explanatory example with a synthetic 2-dimensional 4-class Gaussian data set

We generate a 2-dimensional 4-class Gaussian data set to show an explanatory example of the use of the new tools. The data samples are drawn randomly from four Gaussian distributions with zero mean and unit variance, at four centers in a 2-dimensional space. The data samples are plotted in the data space in Fig. 3.7, left column, with their known class labels (colors) overlain. In Fig. 3.7, middle column, the SOM is overlain with mU-matrix and known class labels. Black cells represent empty neurons, which have no data samples mapped to them. During the evolution of the SOM three snapshots have been taken, at 1K (1000) steps (Fig. 3.7, top row), 3K steps (Fig. 3.7, center row) and 100K steps (Fig. 3.7, bottom row). In Fig. 3.7, left column, the black dots are the learned prototypes projected back into the data space. The prototypes are connected according to the SOM lattice structure.

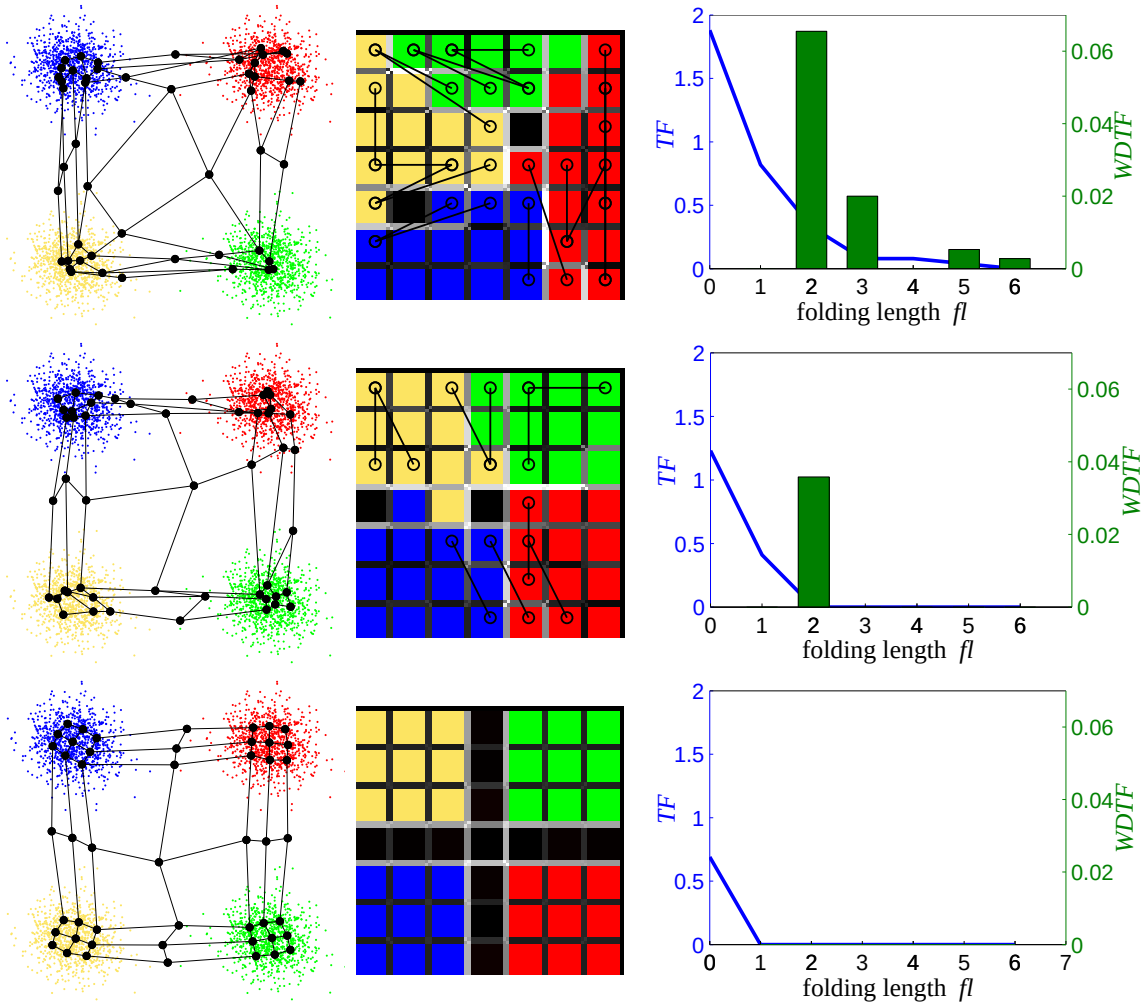


Figure 3.7: The evolution of the SOM as it learns a synthetic 2-dimensional 4-class Gaussian data set. Three snapshots are shown at 1K, 3K and 100K steps, from top to bottom. **Left**: The SOM prototypes (black dots) are plotted in the data space and connected according to the SOM lattice structure. Data samples (small dots) are color coded according to their known class memberships. **Middle**: All violating connections are shown as black lines, over the SOM. The SOM is also overlain with the known class labels (colors) and the mU-matrix. **Right**: The TFs (blue lines) and the $WDTF$s (green bars).

It is evident, from the visualization of the SOM prototypes in the data space (left column) and from the SOM overlain with mU-matrix and known class labels (middle column), that the SOM improved as the learning step increased. At 1K steps, the SOM appears twisted in the data space (Fig. 3.7, top left). We can see an obvious twist of the map at the upper right corner (red cluster), where a chain of prototypes is arranged in the shape of a "horseshoe". In the top middle SOM, the obvious high (white) fences within the 4 known

clusters show that the SOM has not yet learned the structure well. As the SOM evolves, the within-cluster high fences are gradually relieved. At 100K steps, all within-cluster high fences disappear, and the cluster structure delineated by the double-fenced empty (black) corridors agrees perfectly with the overlain known class labels (Fig. 3.7, bottom middle). In the data space, the prototypes are nicely placed according to the manifold shape (Fig. 3.7, bottom left). These all indicate the improvement of the SOM in topology preservation throughout this learning.

We next discuss how the measures and TopoView reflect this improvement in the SOM. At 1K steps, TopoView expresses the "horseshoe" twisting (in the red cluster on top left) by a set of connections along the right side of the SOM (Fig. 3.7, top middle). From the connection statistics shown by both the TF and the $WDTF$ (Fig. 3.7, top right), we know the existence of violations up to folding length 6. This means that the end neurons of the chain on the right side of the SOM, and some other non-lattice-neighbor prototypes in between must be connected. The $WDTF$ also shows that the long-range violations, at $fl = 5, 6$, are relatively weak, compared with the short-range ones at $fl = 2, 3$. From the TF, however, the detailed extent of violations at each folding length cannot be seen due to the TF's integral property. As the SOM evolves, the set of long-range connections (in the red cluster) along the right side of the SOM disappears at 3K steps (Fig. 3.7, center middle), which means the "horseshoe" changed to a shape that better approximates the spherical cluster. Finally, TopoView shows the SOM free of violating connections at 100K steps in Fig. 3.7, bottom middle. The $WDTF$ vanishes in Fig. 3.7, bottom right. The residual in the TF comes from the inconsequential backward violations between the empty neurons and their lattice neighbors. Since TopoView, the TF, and the $WDTF$ all show the improvement in topology preservation, we can conclude that these tools indeed reflect the true topological health of the SOM.

For comparison with the $WDTF$, we also compute the TP for these three snapshots of the SOM in Table 3.1. In all three stages, the TP has a small, near zero value, indicating an approximate dimensional match between the input and the output spaces. However, the change in the value of the TP along the SOM's evolution does not reflect the truth. The TP deviates the most from 0 at 100K steps, indicating the worst topological health at this learning step, while in fact at 100K steps the SOM is the best among the three snapshots as seen in Fig. 3.7, left and middle column. This indicates that the TP is not a very helpful and accurate measure.

Table 3.1: The Topographic Product (TP) calculated for the three stages of learning, at 1K, 3K and 100K steps, of the SOM learned with the 2-dimensional 4-class Gaussian data set.

Learning steps	1K	3K	100K
TP	-0.03189	-0.02192	-0.03305

3.5.2 A study with a 194-dimensional hyperspectral image

In this section, we demonstrate the power of the $WDTF$ and TopoView through a 194-dimensional real image data set. This data set is a noisy remote sensing VIS-NIR (0.4 – 2.5 μm) hyperspectral image of the Lunar Crater Volcanic Field (LCVF), Nevada, USA. In this LCVF area, remote sensing images are taken yearly for extensive field studies. The 614$\times$420 image (257,880 pixels) we use is a subsection of the image collected by AVIRIS (the Airborne Visible Near-infrared Imaging Spectrometer of NASA/JPL) [48, 49] on April 5, 1994 at 18:22 GMT. The spatial resolution is 17m/pixel. AVIRIS measures spectral radiance values in 224 bandpasses, 30 bands of which were removed due to excessive noise and overlaps in the detector channels. The remaining 194 image bands comprise the hyperspectral image used in this work. The challenge in the study of the LCVF area is the large number of surface cover types to be detected and distinguished [3, 47, 50].

A natural color composite of the LCVF image is shown in Fig. 3.8, with 23 cover types marked at representative locations by class labels. (We refer to [3, 47] for details of these cover types.) Merényi trained a 40×40 SOM to learn this image, and, after 300K learning steps, identified 32 distinct cover types through interactive cluster extraction from the SOM with the help of the mU-matrix visualization [3]. The extracted clusters are shown in the SOM, and mapped back into the spatial image, in Fig. 3.9. The verification of the extracted clusters was based on accumulated ground truth from comprehensive and independent field studies, and previous analysis, done by others in a number of works [51, 50].

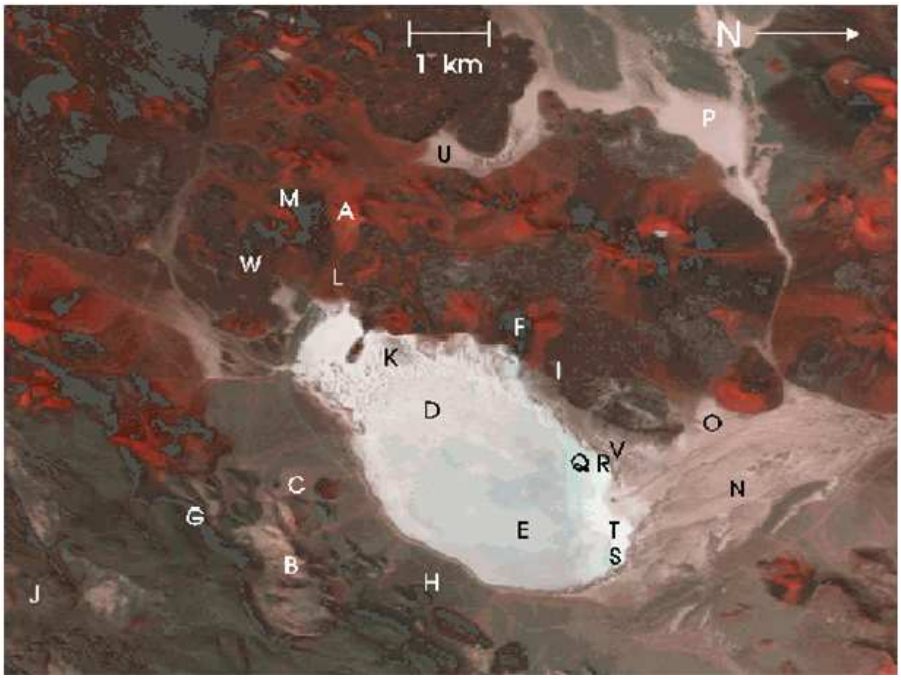


Figure 3.8: A natural color composite of the Lunar Crater Volcanic Field (LCVF). Figure from [3], courtesy of E. Merényi. 23 character labels indicate different cover types of geologic interest. We refer to [3, 47] for details of these cover types.

Because from the 40×40 SOM of the LCVF image 32 verified cover types were extracted successfully, we can assume a reasonably high degree of topology preservation in this SOM. We will apply the $WDTF$ and TopoView to the SOM to more closely examine the SOM in this respect. Then, we will use the same tools to compare two stages of learning

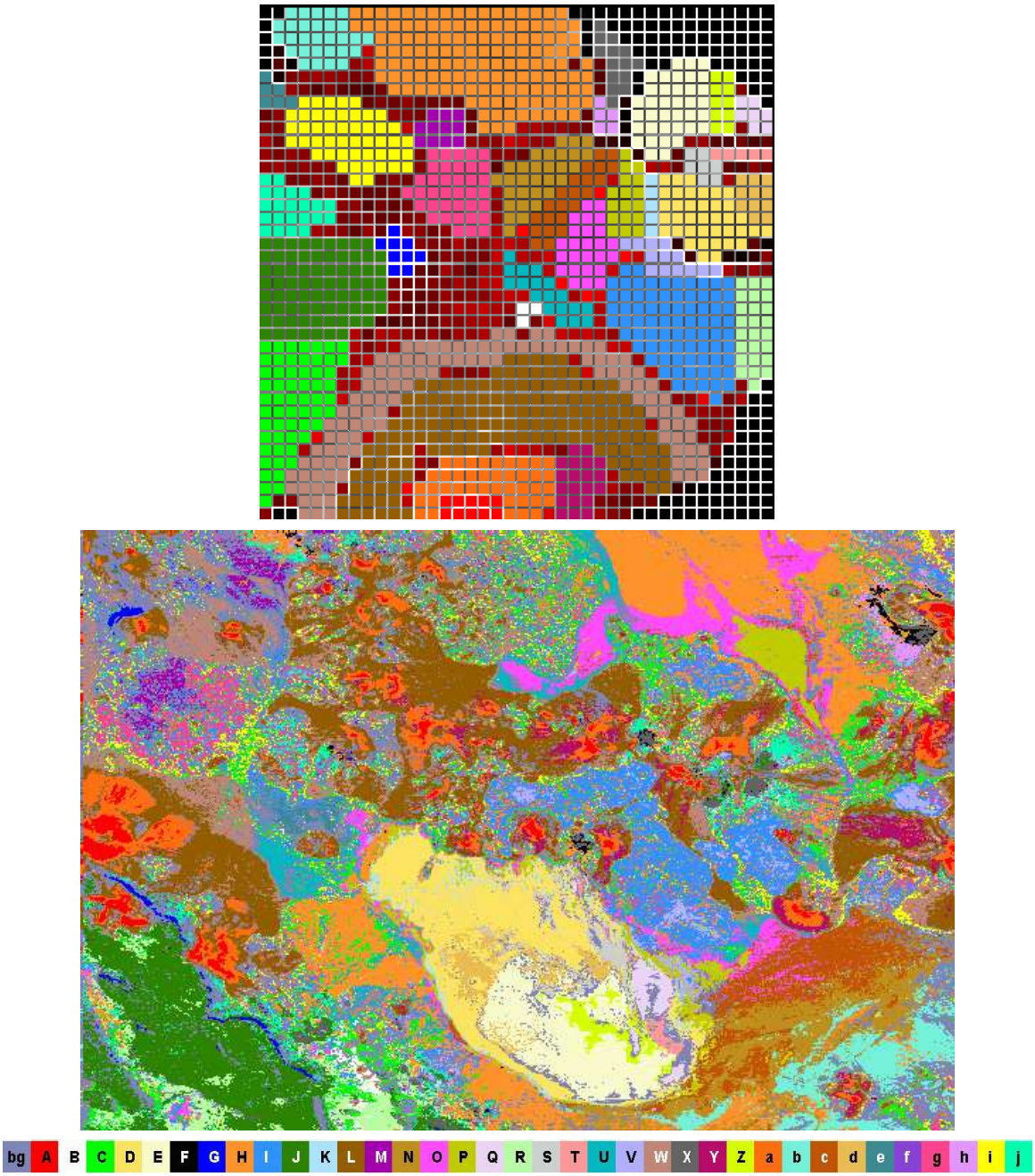


Figure 3.9: The 40×40 SOM learned with the LCVF image for 300K steps. Figures from [3], reused here with kind permissions of both Springer Science+Business Media and E. Merényi. **Top**: The cluster labels (colors) were extracted by Merényi from the SOM by using the mU-matrix visualization [3]. **Bottom**: The clusters mapped back into the spatial image. Each color corresponds to a different surface cover type, whereas medium grey indicates background "bg" (unclustered) pixels.

in this SOM, one after 300K learning steps, and the other after 8M (8,000,000) steps. The SOMs and the extracted cluster structure in these evaluations are provided by Merényi.

Evaluation of topology preservation in the SOM of the LCVF image after 300K learning steps

We use our new measures and interactive visualization tool to evaluate topology preservation of the SOM that learned the LCVF image for 300K steps (Fig. 3.9, top). In Fig. 3.10, left, we compare the TF with the DTF. From the TF, we can see that violations exist up to folding length 39, the largest possible folding length for the 40×40 SOM. The DTF shows the average number of violating connections per neuron at each folding length, providing a clearer view of the extents of violations at different folding lengths fl. For example, we can see from the DTF that each neuron has approximately 1.5 violations with folding length 2 and the average number of violating connections at the extreme long ranges (27 $\leq fl \leq$ 39) is less than 0.2. In Fig. 3.10, right, we compare the TF with the $WDTF$. The $WDTF$ expresses the severity of violations at different folding lengths by the percentage of contributing data samples. For example, approximately 2.6% of the data samples (0.026 in the figure) participate in the formation of the violations with $fl = 2$. Less than 1% of the data samples (0.01 in the figure) are involved in the violations at each folding length that is larger than 6. In addition to the statistical view of the violations provided by the DTF and the $WDTF$, our interactive tool, TopoView, can show the locations and the orientations of violations selected by the user. In Fig. 3.11, left, TopoView visualizes all, 521, violating connections, whose strengths are larger than the mean strength of all connections (which is 15 in this case), on the SOM. Interestingly, most of the violations follow the shapes of the boundaries of the clusters identified in [3]. For example, in the bottom annular area in the SOM, the connections profile the boundaries of five adjacent clusters, which represent

geologically similar surface cover types (i.e., compositional, and thus spectrally similar cover types). For example, the red cluster maps peaks of cinder cones and the dark orange cluster maps flanks of cinder cones. In the upper right area of the SOM, almost all violations follow the same direction. These can be results of the foldings within those clusters or manifestations of close relationships between the clusters along that direction (D, E, S, P, etc.). These clusters form a series of continuously varying signatures (not shown here).

To distinguish the inter-cluster violations, which are potentially harmful to correct clus-

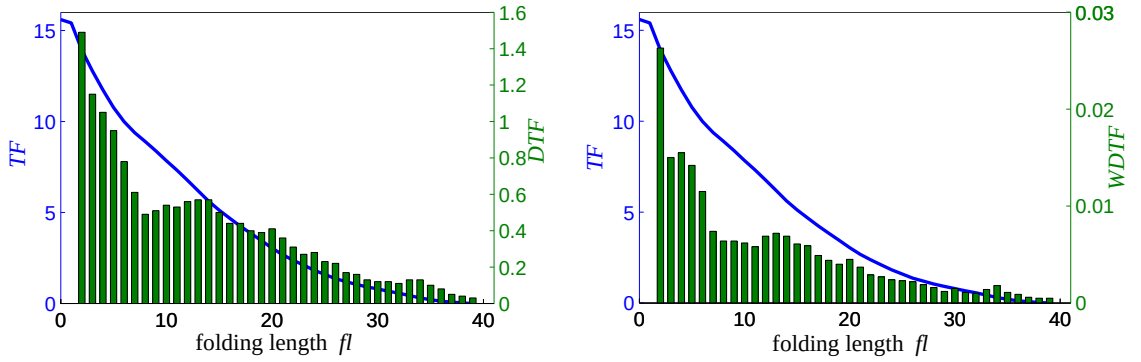


Figure 3.10: The TF, the DTF and the $WDTF$ for the 40×40 SOM learned with the 194-dimensional hyperspectral image of the Lunar Crater Volcanic Field (LCVF) area after 300K learning steps. **Left**: The DTF provides a clearer view of the extents of violations (average number of violations per neuron) at different folding lengths, fl, while the TF does not show this information obviously. **Right**: The $WDTF$ shows the severity of violations at each folding length as the percentage of contributing data samples.

ter extraction, we apply an additional "inter-cluster" filter. This results In Fig. 3.11, right, where 165 violations pass the filtering conditions. They involve only approximately 1% of the total connections. In close inspection of those 165 violations, we find that although two prototypes are connected by a violation, the difference between them is still large enough to separate them into different clusters. These violations do not indicate topological problems, rather, they can be caused by either noise in data, or the mixtures of signatures from different cover types in some image pixels. These qualitative evaluations confirm that the topology preservation in the SOM is good.

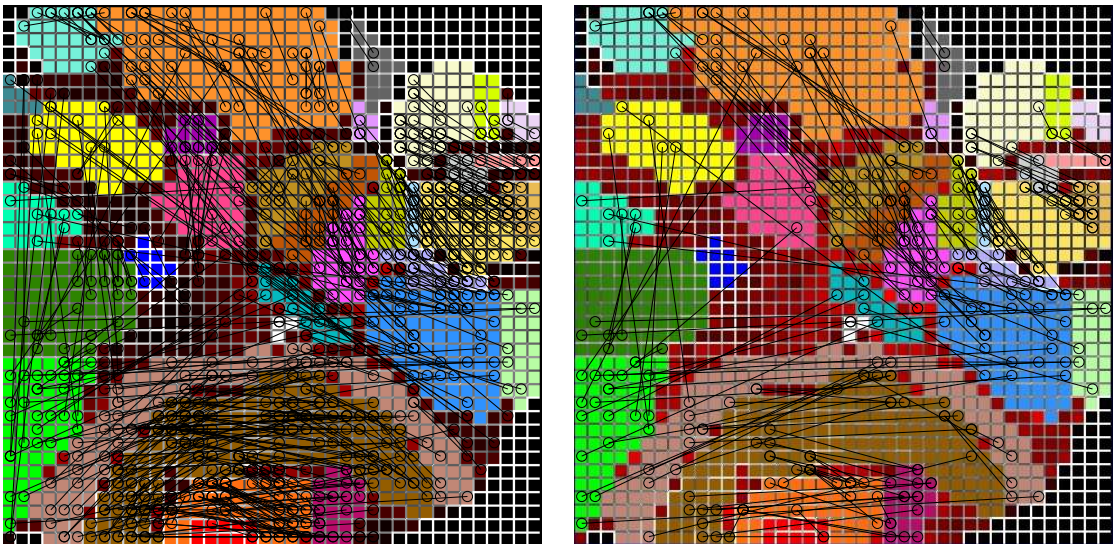


Figure 3.11: TopoView visualization of selected subsets of connections (black lines) on the SOM of the LCVF image. The SOM is also overlain with extracted class labels in [3] and the mU-matrix visualization. **Left**: TopoView shows all, 521, violating connections with strength larger than the mean strength of all connections (which is 15 in this case). **Right**: TopoView shows all, 165, inter-cluster violating connections with strength larger than the mean strength of all connections (which is 15 in this case).

Comparison of the SOMs after 300K and 8M learning steps

For the assessment of the expressiveness of the new measures, the $NDTF$ and the $WDTF$, as compared to the TF and the TP, we continued the learning of the SOM from the previous stage, 300K steps, to 8M steps and compare these two stages of learning. The TPs for both the SOMs after 300K and 8M steps are small numbers, as seen in Table 3.2. The decrease in the magnitude of the TP indicates an improvement of topology in the SOM, however, as a single-number measure, the TP does not provide any insight how much better the SOM really became. In Fig. 3.12, top, the TFs after 300K and 8M steps are similar, with a slight decrease for $fl < 17$ and increase for $fl \geq 17$. The TF thus provides no conclusive result in the comparison. The $NDTF$s (middle) show the changes in the numbers of violations at different folding lengths. With longer learning time (8M steps), there are small, $< 10\%$, decreases in the $NDTF$ at certain short ranges (e.g., $fl = 2$, 10, 11, 13, 14 and 19), while at most of the remaining folding lengths, including the the longest

ones ($fl > 30$), the $NDTF$ increases. This overall growth in the number of violations, especially at long ranges, is a warning of possibly worsened topological health after more learning steps. In contrast, the $WDTF$ (bottom) leads to the opposite conclusion. The longer learning time quenches many high peaks in the $WDTF$, especially at short ranges, by $> 30\%$ at $fl = 2$ and by $20 - 30\%$ at folding lengths between 3 and 14 (indicated by arrows). Moreover, the changes at long ranges are negligible, showing no loss in good topological health. We can conclude from the $WDTF$ that the topology preservation improved as the SOM learned longer. From the above, we see that the conclusion from the $NDTF$ contradicts the conclusion from the $WDTF$. The $NDTF$ suggests that the quality of the map dropped slightly because the number of violations increased at the long ranges and decreased by a tiny amount at the short ranges. The $WDTF$, however, favors the SOM after longer training time, because it shows that the violations at all ranges became much weaker after longer training time, which alleviated the overall severity of violations.

Table 3.2: The Topographic Product (TP) calculated for the two stages of learning, at 300K and 8M learning steps, of the SOM learned with the 194-dimensional LCVF image.

Learning steps	300K	8M
TP	-0.14870	-0.14237

A detailed statistical analysis of the connections between SOM prototypes supports our conclusion obtained with the $WDTF$ (Table 3.3). The percentage of the violating connections in the SOM decreases from 73.6% to 70.9%. The total strength of the non-violating connections (percentage of data samples contributing to the non-violating connections) increases from 80.3% to 83.7%. In addition, the average strength of the violating connections decreases from 4.1 to 3.6. These facts indicate that longer learning time (8M steps) is beneficial to the topological health in this case, and thereby confirms the analysis results from the $WDTF$. This means that the $WDTF$ is capable of expressing the topological quality

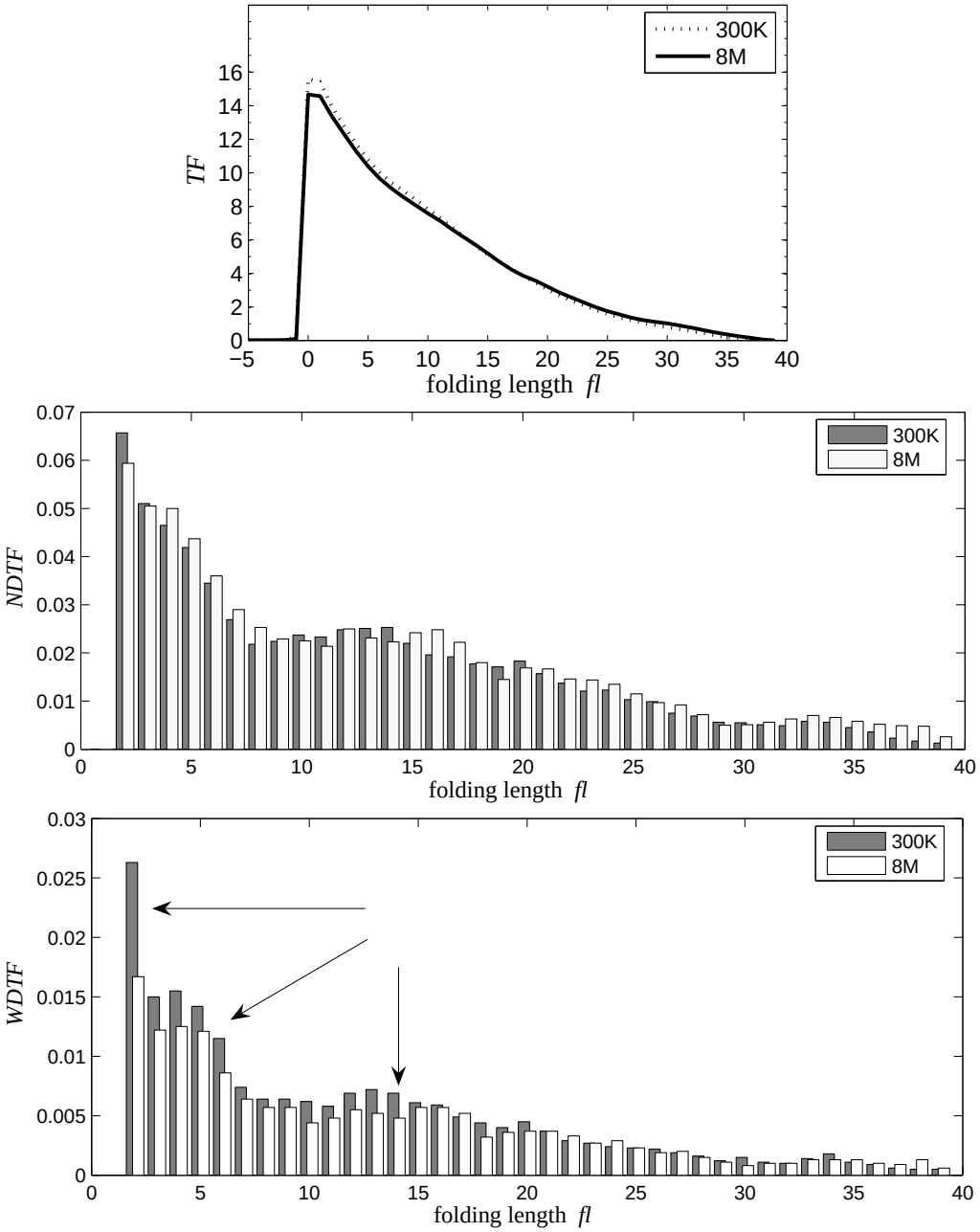


Figure 3.12: Comparison of two learning stages, 300K steps and 8M steps, of the SOM of the LCVF image with the TF, the $NDTF$ and the $WDTF$. **Top**: The two TF curves for the two learning stages are similar, with no pronounced difference. The TF provides no conclusive comparison. **Middle**: The $NDTF$ shows an increase at most of the folding lengths, including the longest ones ($fl > 30$), and a less than 10% decrease at a few folding lengths (e.g., $fl = 2$, 10, 11, 13, 14 and 19). This indicates a possibly worsened topological health in the SOM. **Bottom**: The $WDTF$ shows obvious, 20–30%, decreases in the severity of violations at several folding lengths (indicated by arrows) after longer learning time. At other folding lengths, there is no significant change in the $WDTF$. This indicates an overall improvement in the map.

Table 3.3: Statistics of the connections for the SOM learned with the LCVF data, at two different stages of learning, after 300K learning steps and after 8M steps, respectively. Percentage representations of the values are in parentheses. The numbers in bold face indicate the improvement of topology preservation in the SOM after longer learning steps, as discussed in the text.

Learning steps	300K		8M	
Number of empty neurons	119		3	
Number of data samples *	257875		257870	
Number of all connections	16750		16443	
Number of violating connection	12327	(73.6%)	11659	(**70.9**%)
Number of non-violating connections	4423	(26.4%)	4784	(**29.1**%)
Average strength of all connections	15.4		15.7	
Average strength of violating connections	4.1		**3.6**	
Average strength of non-violating connections	46.8		45.1	
Number of data points in the non-violating connections	207162	(80.3%)	215879	(**83.7**%)

* The data samples that induce the connections to empty neurons are excluded from this statistics, because we consider empty neurons to be sources of inconsequential violations, as discussed in Section 3.3.2.

of the SOM more accurately than the $NDTF$ for complicated real data sets.

3.5.3 A study with a 210-dimensional synthetic hyperspectral image

We give a second demonstration of the use of the new tools through a 210-dimensional hyperspectral urban image we call "RIT image", which was synthetically generated via rigorous radiative transfer modeling called the DIRSIG procedure at the Rochester Institute of Technology [53, 54]. The image comprises 400×400 pixels, each of which is a spectrum in the 0.38–2.4 μm wavelength window. In spite of its synthetic nature, the RIT image is amazingly realistic with noise, illumination geometry, spectral variations, and other attributes incorporated, in the simulation. The scene, rendered as a natural color composite in Fig. 3.13, top, appears indistinguishable from a real scene. Importantly, the availability of material labels on the pixel level makes this data set suitable for objective evaluation of analysis results. There are over 70 different surface materials in the image, including vegetation, various roof shingles, sidings, building materials, road pavings and

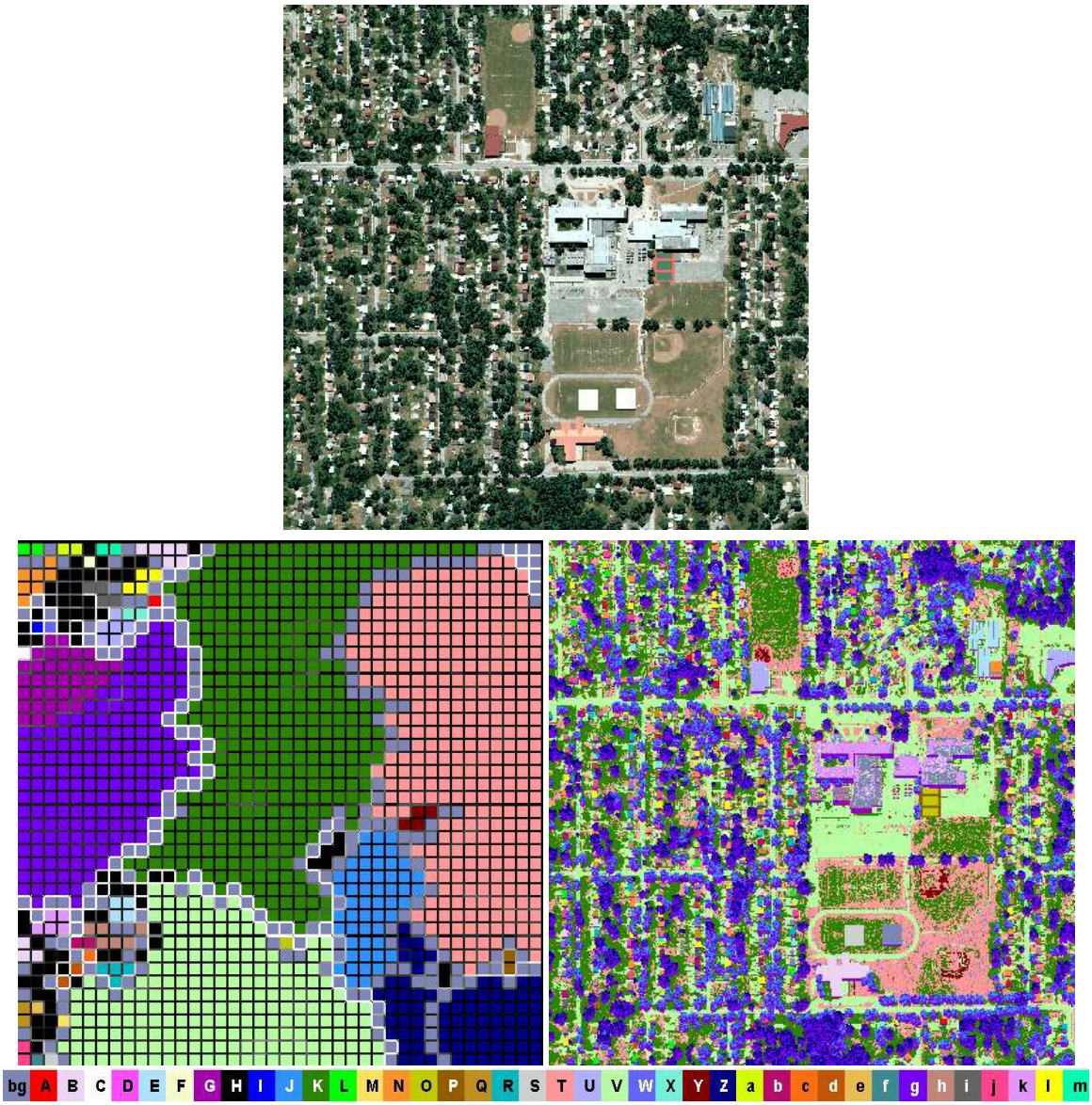


Figure 3.13: The 210-band synthetic hyperspectral RIT image and the SOM learned with it. Figures from [52], courtesy of E. Merényi. **Top**: A natural color composite of the RIT image. **Bottom left**: The SOM of the RIT image (after 3M learning steps), overlain with cluster labels that have been identified by Merényi *et al.* in [52]. The SOM is also overlain with the mU-matrix. Cells with the color of the background, "bg", are empty neurons (no data mapped to them), most of which appear along cluster boundaries. Black cells indicate prototypes whose cluster labels are not shown in this representation due to color limitation in the SOM visualization software. **Bottom right**: Clusters mapped back to the spatial image.

car paints. From the SOM learned with this image after 3M steps, Merényi *et al.* identified groups of prototypes representing these different surface materials, in [52], and Merényi provided the SOMs and the clusters for this study of the tools for evaluation of topology

preservation. Fig. 3.13 shows the identified clusters in the SOM (bottom left) and the cluster labels mapped back to the spatial image (bottom right), respectively. We refer to [52] for descriptions of the clusters and the surface materials they represent.

We will first compare two snapshots of the SOM learned with the RIT data, at 500K and 3M steps, with the WDTF and TopoView. Second, we will use TopoView to help evaluate the clusterings from two SOMs of the RIT data.

Comparison of the SOMs learned after 500K and 3M learning steps

We compare two learning stages (at 500K and 3M steps) of the SOM of the RIT data. First we will confirm that the topological quality of the SOM improved by scrutinizing the statistics of the connections in the SOM. Second we will apply the measures and tools to the evaluation of the two learning stages of the SOM, to show that the $WDTF$ and TopoView can correctly reflect the improvement in the SOM.

We conduct a detailed statistical analysis of the connections between SOM prototypes (Table 3.4) to compare the two learning stages. The percentage of the violating connections in the SOM decreases from 31.0% to 28.1%. The total strength of the non-violating connections (percentage of data samples contributing to the non-violating connections) increases from 93.9% to 94.9%. In addition, the average strength of the violating connections decreases from 12.6 to 11.7. These facts indicate that longer learning time (3M steps) is beneficial to the topological health in this case.

We next use the measures to evaluate the SOMs. In Fig. 3.14, we can see the topology preservation improved from 500K to 3M steps. The TF shows obvious decrease in the short-range violations (near the peak $fl = 0$), but not conclusive for the long ranges. The $NDTF$ differentiates the view in the TF: the violations with the shortest range, $fl = 2$, decrease by one third; the numbers of violations at other folding lengths have small

Table 3.4: Statistics of the connections for the SOM learned with the RIT data set, at two different stages of learning, after 500K learning steps and after 3M steps, respectively. Percentage representations of the values are in parentheses. The numbers in bold face indicate the improvement of topology preservation in the SOM after longer learning steps, as discussed in the text.

Learning steps	500K		3M	
Number of empty neurons	539		554	
Number of data samples *	141477		133154	
Number of all connections	2220		2055	
Number of violating connection	688	(31.0%)	577	(**28.1**%)
Number of non-violating connections	1532	(69.0%)	1478	(**71.9**%)
Average strength of all connections	63.7		64.8	
Average strength of violating connections	12.6		**11.7**	
Average strength of non-violating connections	86.7		85.5	
Number of data points in the non-violating connections	132812	(93.9%)	126376	(**94.9**%)

* The data samples that induce the connections to empty neurons are excluded from this statistics, because we consider empty neurons to be sources of inconsequential violations, as discussed in Section 3.3.2.

changes. With the $WDTF$, we not only see the same considerable decrease in the severity of the short-range violations (one third decrease at $fl = 2$, one sixth decrease at $fl = 3$ and one third decrease at $fl = 4$), but also find a general decrease at large folding lengths, with some exceptions (such as at $fl = 8$ and $fl = 13$). For this case, both of the $NDTF$ and the $WDTF$, which show an overall decrease in the extent and the severity of the violations, respectively, indicate that the topological health of the SOM improved after long learning time (3M steps). This agrees with what we conclude from the detailed statistical analysis of the connections above.

From the TopoView representations in Fig. 3.15 one can follow which violations disappear between the two snapshots. In Fig. 3.15, top row, we show all violations for the two snapshots of the SOM. We can see that the 3M snapshot is cleaner than the 500K snapshot. The total number of violations decreased from 688 to 577. The lower left corner of the SOM even became completely violation free. To view the strong violations, we set the threshold for connection strength as the mean strength of all connections in TopoView, as

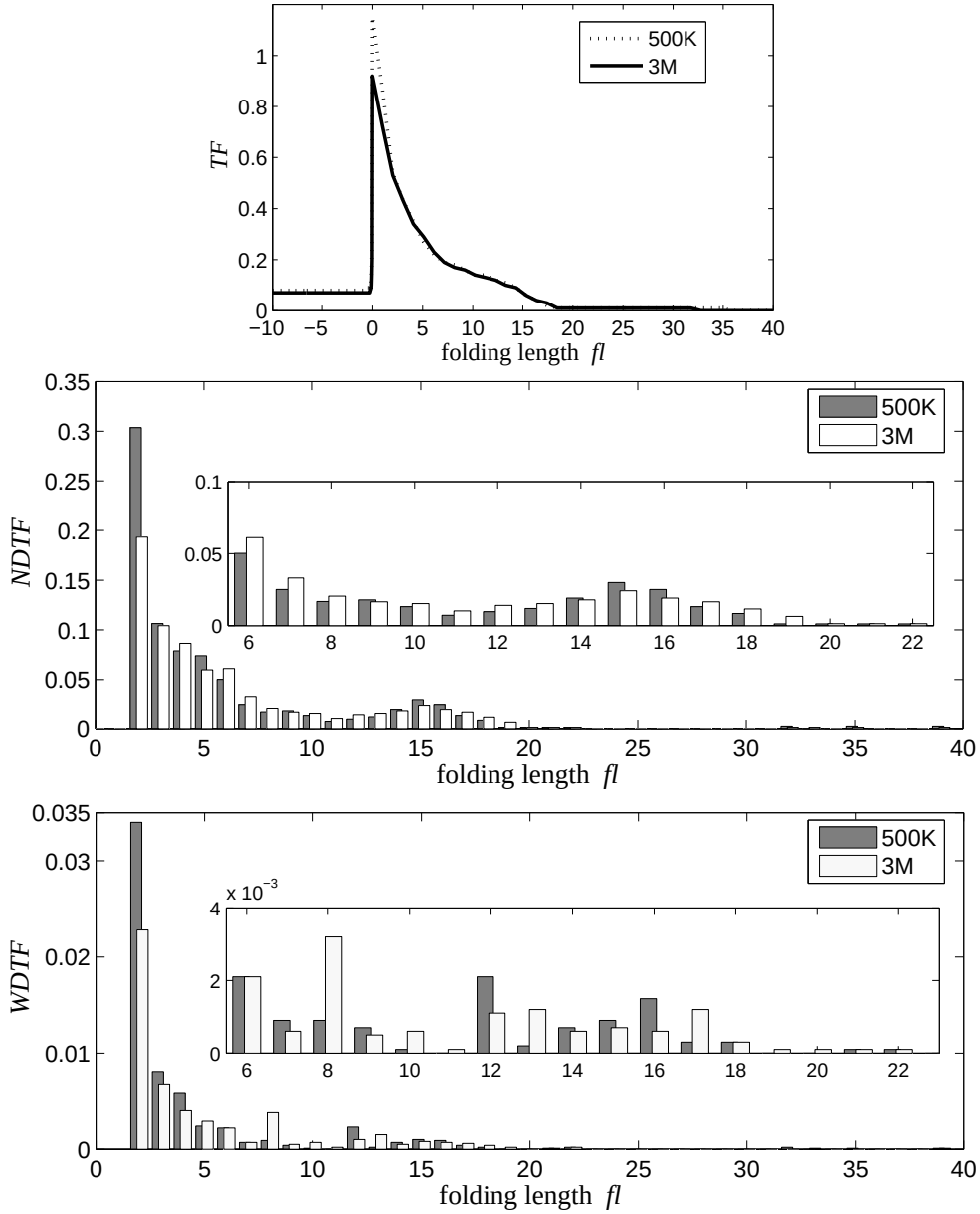


Figure 3.14: Comparison of two learning stages, at 500K steps and 3M steps, of the SOM of the RIT image with the TF, the $NDTF$ and the $WDTF$. **Top**: The TF shows a general decrease of violations at short folding lengths. Parts of the $NDTF$ and the $WDTF$ are magnified and shown in insets for clarity. **Middle**: The $NDTF$ indicates an overall decrease in the number of violations at most folding lengths after longer learning time. Some exceptions exist, i.e., at $fl = 6, 7, 10, 12$, where the number of violations increased slightly. **Bottom**: The $WDTF$ indicates a general decrease in the severity of violations (the number of contributing samples) at short folding lengths ($fl = 2, 3, 4$) and at most larger folding lengths ($fl = 7, 12, 15, 16$). Exceptions exist at some folding lengths, such as at $fl = 8, 13, 17$, where the severity of the violations increased.

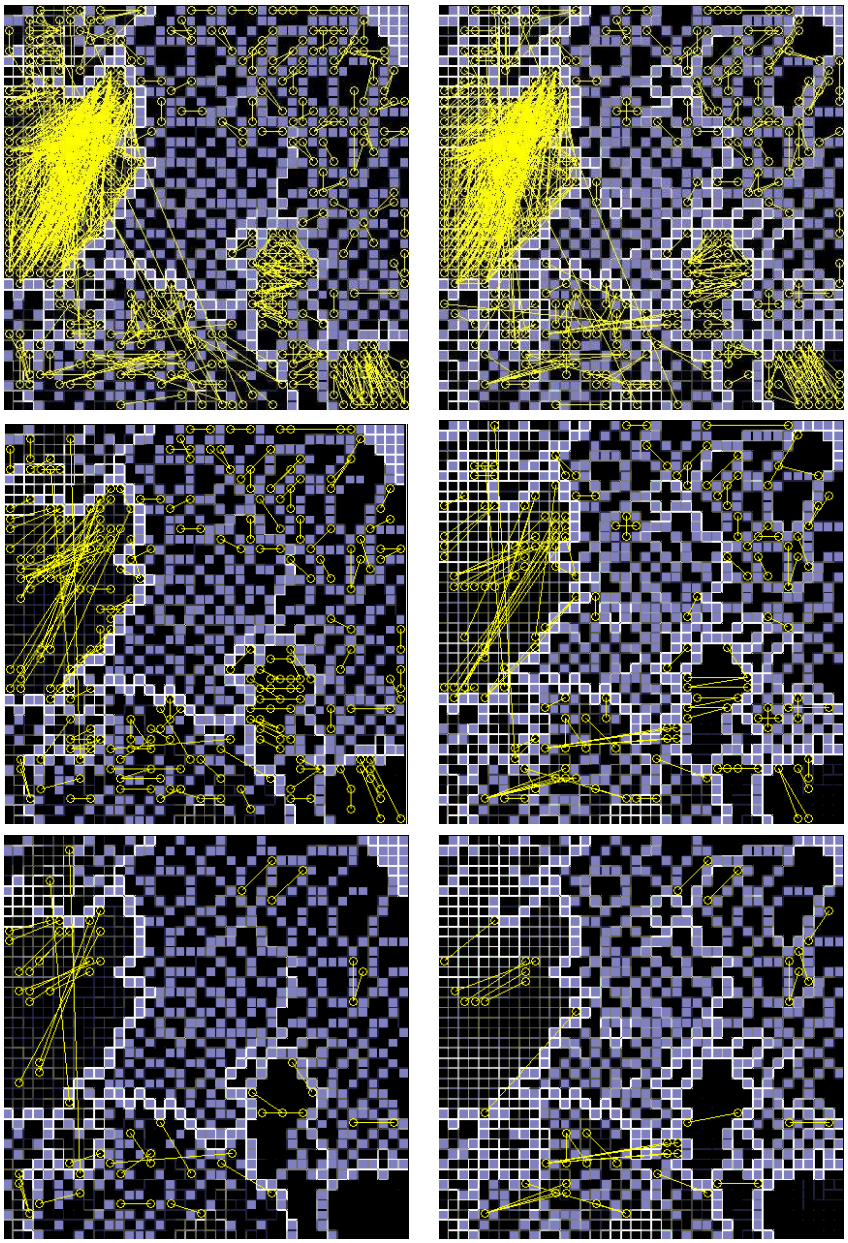


Figure 3.15: TopoView visualization of selected sets of violations on the SOM learned with the RIT image. A comparison is done between two learning stages of the SOM, at 500K (**left column**) and 3M steps (**right column**). Three different subsets of violations are selected to be shown by TopoView, from top to bottom. The same SOM as in Fig. 3.13 is superimposed with the mU-matrix visualization. Medium grey and black cells indicate empty and non-empty neurons, respectively. To make the violations easy to see, we do not show the cluster labels on the SOM. **Top row**: All violating connections are shown. **Middle row**: Violating connections with strength greater than the mean strength of all violating connections are shown. **Bottom row**: Global violating connections ($fl > 2$ for this data set) with connection strength greater than the mean strength of the fourth strongest connections of all prototypes are shown.

shown in Fig. 3.15, middle row. With this representation of violations, it is easier to see that in most part of the map the number of violations decreased as the SOM learned longer. In Fig. 3.15, bottom row, we show another representation of the "important" violations. We turn on the filters for both the connection strength and the folding length. We set the threshold for folding length as the maximum folding length of local violations, l_{min} (which is 2 in this example), to show only the global violating connections. In addition, we set the threshold for connection strength as the mean strength of all fourth ranking connections. The rationale to use the statistics of all fourth ranking connections is that the four highest ranking neighbors are usually the most important ones in a rectangular SOM lattice, as pointed out in [24]. This representation shows obvious improvements in the quality of topology preservation. For example, the two long connections from the upper left corner to the lower left corner in the SOM disappear. From all three views of the violations by TopoView in Fig. 3.15, we arrive at the same conclusion that the topological health of the SOM improved.

Through the above experiment, we have demonstrated the ability of the $WDTF$ and TopoView in correct reflection of the change in topological health of the SOM that learned complicated high-dimensional data sets.

TopoView assists the evaluation of clustering

We use TopoView to compare the clusterings from two different SOMs of the RIT image. The two SOMs were learned separately, but with the same parameters and both to 3M steps. Consequently the two SOMs are very similar with some minor differences. The two clusterings were produced from two methods. The clusters in the first SOM (Fig. 3.16, left column) were extracted from the mU-matrix in [52], and the clusters in the second SOM (Fig. 3.16, right column) were produced from CONNvis visualization, an interactive

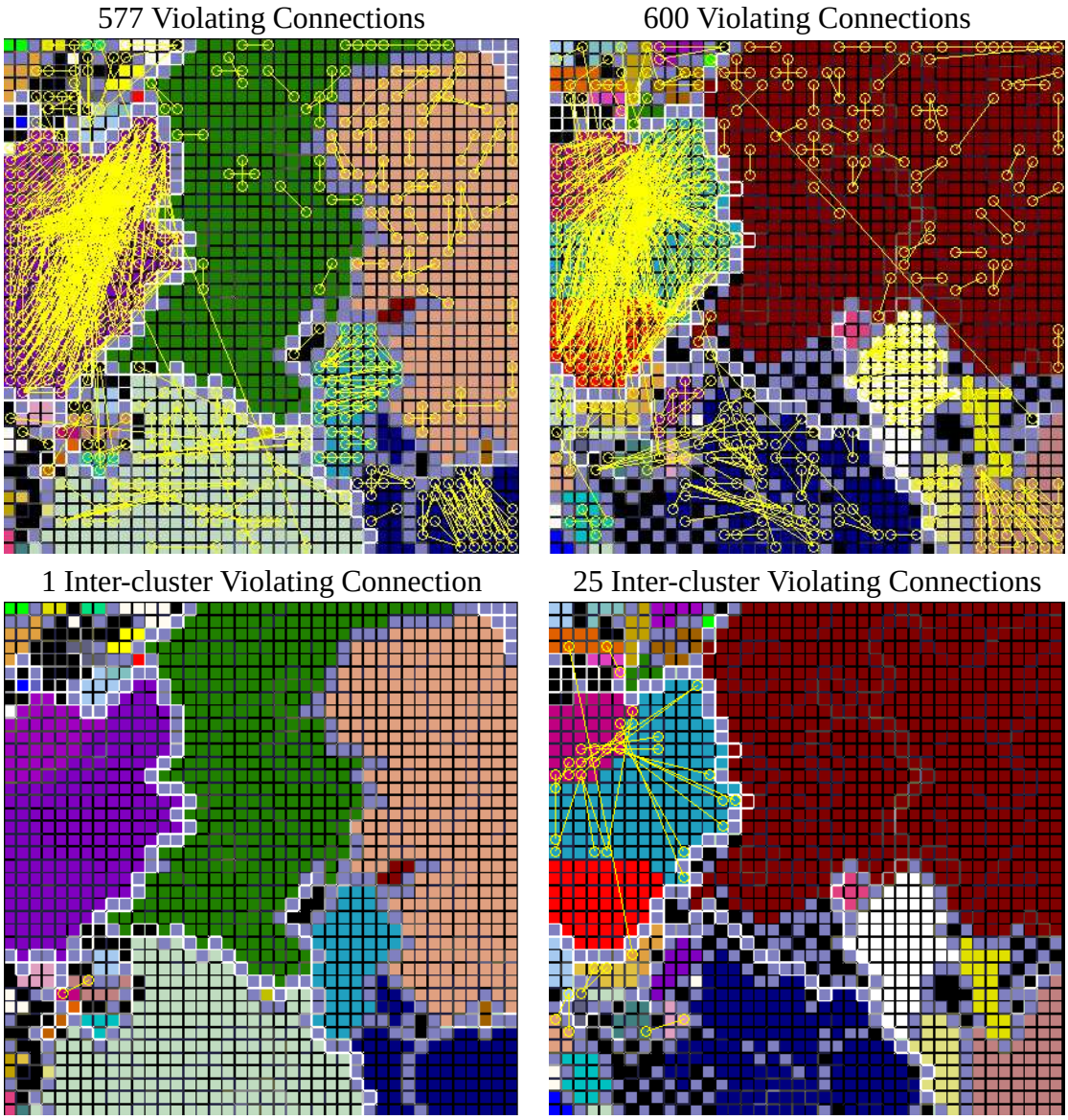


Figure 3.16: TopoView visualization of selected sets of violating connections (yellow lines) on the SOMs leaned with the RIT image. A comparison is done between two SOMs, which are similar with some minor differences, shown in the **left** and **right** columns. Both SOMs are overlain with extracted cluster labels and the mU-matrix. In the SOM in the left column, the extracted clusters are the same as in Fig. 3.13, bottom left. In the SOM in the right column, clusters were extracted with the help of CONNvis [24] in [4]. The color label of cluster V (light green) was removed to show the underlying scattered empty prototypes in [4]. Medium grey cells are empty neurons. Black cells do not indicate empty neurons or cluster "H". Clusters of those prototypes are not shown in this representation due to color limitation in the SOM visualization software. TopoView visualizes two selected sets of violations. **Top row**: All violating connections. **Bottom row**: All inter-cluster violating connections.

clustering method based on the $CONN$ matrix [24], in [4]. Because of the high similarity between the two SOMs, we can make comparative observations between the two clusterings. Since the cluster labels were assigned separately in the two SOMs, the same spectral clusters generally have different color labels in the two SOMs. However, the similarity in the layout helps relate them visually.

We overlay the TopoView visualization of violating connections (yellow lines) on these cluster representations. We first show all violating connections in Fig. 3.16, top row. The two SOMs have similar total numbers of violations (577 for the SOM on the left, and 600 for the SOM on the right), as well as similar locations of the violations. These facts, combined with the similar layout of the clusters in the SOMs, confirm the high similarity between the two SOMs. We also find that most violations occur within the clusters. The most disorganized clusters (those with many intra-cluster violations), such as the purple one at center left of the left SOM, are results of the high noise level in those clusters. In contrast, some other clusters, such as the dark green one in the middle of the left SOM, seem well-organized with few violations, due to the relatively small spectral variation in the cluster. Next, we compare the two clusterings by showing all inter-cluster violations in Fig. 3.16, bottom row, for the two SOMs, because the inter-cluster violations can be a warning of incorrect extraction of clusters. In the left SOM, only 1 inter-cluster violation is left after 3M steps, which has a folding length of 2 and connects two adjacent and similar clusters. In the right SOM, however, there are a handful of violations left, with folding lengths ranging from 2 to 24. The violations with long folding lengths cross 2 or 3 clusters. The inter-cluster violations could indicate any of the following situations. One is insufficient learning in the SOM. Another possibility is that noise induces those inter-cluster violations. In this case, since thresholding can help analyze the significance of the connection, we can further visually remove weak connections with a threshold for connection strength (not shown

here). A third possible reason is incorrect extraction of the cluster boundaries. For example, two connected clusters may be similar enough to be combined into one. To determine the cause of the inter-cluster violations, further investigation is needed. In response to different causes, different remediations should be used. If the cause is insufficient learning of the SOM, we can lengthen the training time or modify the learning parameters. If the cause is incorrect cluster extraction, we need to closely examine the relationships between the SOM prototypes and carefully modify the cluster boundaries. One good phenomenon in both SOMs is that they have very few inter-cluster violations across the small clusters on the upper and lower left corners. These clusters represent various roof materials and car paintings (as shown and discussed in [52]), which are some of the most interesting clusters in this image. The almost nonexistence of inter-cluster violations across those interesting clusters confirms that both clusterings achieve satisfactory precision (resolution of many small clusters) and accuracy (few confusions across the clusters) in extracting interesting information from the SOMs.

The above application of TopoView in evaluation of clusterings with the RIT image has demonstrated the usefulness of TopoView in providing valid and quantitative information about the correctness of clusterings.

In this chapter, we have refined and enriched an existing measure, the TF, to create a new measure, the $WDTF$, and have introduced an interactive visualization tool, TopoView, for inspecting selected sets of connections (both violating and non-violating connections) on the SOM. We have shown the usefulness of the two new tools through application to three data sets. The $WDTF$ can quantify the severity of topology violations more accurately than the previous measures, the TF and the TP. TopoView is a helpful complementary visualization tool to the $WDTF$, locating potentially harmful violations (the violations

that may confuse the correct understanding of manifold structure) via a set of thresholding capabilities. The experiments with the hyperspectral images especially highlight the advantages of the new tools. For complicated data sets such as the two hyperspectral images, topology violations often exist at all learning steps of the SOM, and the change in topological health of the SOM across different learning steps can be subtle. However, the $WDTF$ and TopoView have been shown powerful in the applications. They can evaluate and express the topology violations in a more refined manner than previous measures.

Chapter 4

A novel SOM-hybrid supervised learning architecture

Material based on:

- L. Zhang, E. Merényi, W. M. Grundy, and E. F. Young, "An SOM-hybrid supervised model for the prediction of underlying physical parameters from Near-Infrared planetary spectra", *Proc. 7th International Workshop on Self-Organizing Maps (WSOM 2009), Advances in Self-Organizing Maps, Jun 8–10, St. Augustine, FL,* Springer-Verlag, LNCS 5629, 362–371, 2009.
- L. Zhang, E. Merényi, W. M. Grundy, and E. F. Young, "Inference of surface parameters from Near-Infrared spectra of crystalline H_2O ice with neural learning", *Publications of the Astronomical Society of the Pacific,* 122:839–852, 2010 July.
- L. Zhang and E. Merényi, "Learning multiple latent variables with Self-Organizing Maps", *Proc. 2010 IEEE International Conference on Granular Computing, Silicon Valley, CA, August 14-16,* 2010.

In Chapter 3, we have developed the measures and tool that help evaluate the correctness of the representation of the manifold structure in the SOM. After obtaining a faithful map, we can use the SOM's knowledge for accurate information extraction. In this thesis work we target the inference of latent variables from high-dimensional data, as explained in Section 4.1. The neural architecture we use is an existing SOM-hybrid architecture, which incorporates the SOM into a supervised learning architecture. This architecture will be introduced in detail in Section 4.2.1. Motivated by initial experiments, we also develop an innovation to the SOM-hybrid architecture and propose a new architecture we call Conjoined Twins (Sections 4.2.2–4.2.3 and Section 4.3). This helps achieve the prediction

accuracy needed for the particular science problem we address, and, in general, provides a principled approach to accurate inference of multiple latent variables.

4.1 Inference of latent variables from high-dimensional data

Latent variables are variables that are not directly observed but are rather inferred (through a mathematical model) from variables that are directly measured, i.e., *observable variables*. The latent variables can be inferred from the observable variables because of their underlying relationship, which can be expressed by a function f:

$$\mathbf{x} = [x_1, x_2, ...x_d]^T = f(\mathbf{l}) = f([l_1, l_2, ..., l_L]^T) \tag{4.1}$$

where $\mathbf{x}$ is a vector of d observed variables, x_1, x_2, ..., and x_d, and $\mathbf{l}$ represents a vector of L latent variables, l_1, l_2, ..., and l_L. Each element of $\mathbf{x}$, or each observable variable, is affected by all latent variables l_1, l_2, ..., and l_L. This means that the latent variables l_j $(j = 1, 2, ..., L)$ have a global effect on the observable variables x_i $(i = 1, 2, ..., d)$. As an example, surface temperature of distant planetary bodies is a variable of interest to astronomers. Since it is impractical or impossible to directly measure the temperature for extended planetary surface regions, astronomers alternatively use reflectance spectra collected by telescopes or spacecraft as thermometers to infer the temperature. Here, temperature is a latent variable and the reflectance values measured at different wavelengths are observable variables. Temperature can influence the reflectance values globally, i.e., at many wavelengths.

The inference of latent variables from the observable variables can be considered as an

inversion problem. To find $\mathbf{l}$ is to find the inverse function f^{-1}.

$$\mathbf{l} = f^{-1}(\mathbf{x}) \tag{4.2}$$

In real problems f can be extremely complicated, and the analytical solution of the inverse function, f^{-1}, is often hard to obtain. Customary numerical approaches to regress this function can be ineffective because the "curse of dimensionality" [2] for high-dimensional data, and also because the form of f needs to be known or assumed. To deal with such regression problems, neural methods are often used. A well-known universal function approximator is the Multilayer Perceptron (MLP) trained with backpropagation rule (abbreviated as BP network) [55, 56, 57], which can deal with high-dimensional data well and does not need prior assumption of the form of the function f. A brief introduction of BP network's architecture and algorithm is given in Appendix C. By training a BP network, we build a black box model, which predicts an output $\mathbf{l}$ for any given input vector $\mathbf{x}$. However, the knowledge in the black box is hard to retrieve and interpret, while understanding the learned knowledge in the box is often desirable for assessment and improvement of the performance of the algorithm.

Motivated by the idea that the high-dimensional data (observable variables) can lie on a low-dimensional submanifold (with a low intrinsic dimension), another approach to find latent variables from observable variables is to embed the data manifold in a low-dimensional space (dimension reduction), as in a number of manifold learning algorithms (e.g., [36, 58, 59]). If the dimension of the low-dimensional mapping space is chosen appropriately, the latent variables are expected to be factorized into different dimensions in the resulting low-dimensional representation. According to [60], however, no study has been done to retrieve the values of the latent variables from the low-dimensional representations that resulted from these algorithms. They were used only for cluster identification. These

algorithms have other insufficiencies as well. A classical approach, principle component analysis (PCA), works well for linear submanifolds, but suffers when nonlinearities exist. Several nonlinear approaches have emerged, such as Isomap [58], locally linear embedding (LLE) [59] and Hessian LLE (hLLE) [61]. These algorithms have been demonstrated successfully for data sets with 3 latent variables (horizontal and vertical angles of face and illumination direction, in data sets of face images) [60]. However, to separate all latent variables, a necessary step for these algorithms is to estimate the intrinsic dimensionality in advance, which is a nontrivial task. In addition, latent variables that induce relatively small variations in the data can also be mistakenly treated as noise and eventually be lost in such dimension reduction, while those latent variables can carry interesting scientific meanings.

Our approach to the inference problem is through Self-Organizing Maps (SOM) [5], which as elaborated in Chapter 3 preserve the topology of the data in a low-dimensional representation without reducing the data dimension. The SOM prototypes store the high-dimensional information present in the data, avoiding loss of information caused by dimension reduction. Moreover, there is no need for prior estimation of intrinsic dimensionality. When the SOM has converged, further analysis of the learned SOM prototypes can help recover relevant information from the data, such as latent variables. We describe next, how the SOM's knowledge is used in a supervised neural network, for this purpose. The neural network we use is an SOM-hybrid architecture, which has an SOM as its hidden layer. After the SOM correctly captures the manifold structure through unsupervised learning, the output layer of the architecture combines the outputs from the SOM neurons into weighted sums to learn the relations between the latent variables and the input data, through supervised learning.

4.2 Supervised learning assisted by an SOM

4.2.1 SOM-hybrid supervised neural architecture

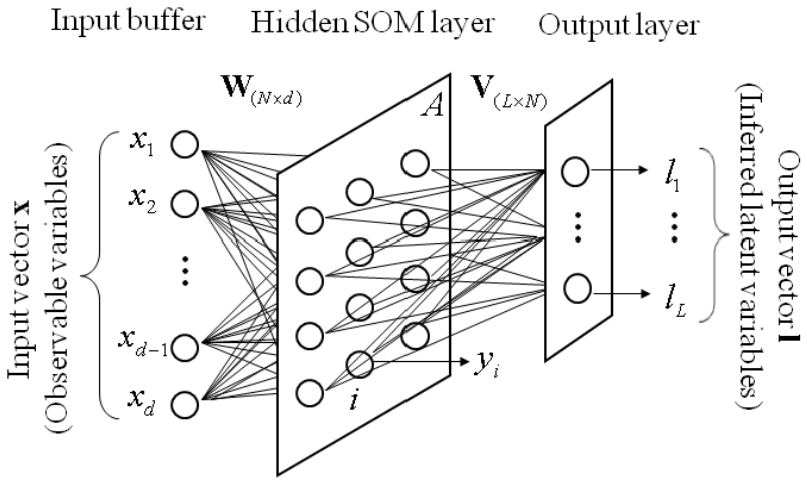


Figure 4.1: The SOM-hybrid neural architecture. It is a two-layer fully connected feedforward network with an SOM as its hidden layer. Each neuron i, in the SOM lattice A of N neurons, is connected to the input buffer with a d-element prototype $\mathbf{w}_i$ (the ith row vector of the $N \times d$ matrix $\mathbf{W}$). An $L \times N$ weight matrix $\mathbf{V}$ connects the output layer to the SOM.

The SOM-hybrid neural architecture we use is a two-layer fully connected feedforward network, as shown in Fig. 4.1. It takes an input vector $\mathbf{x}$ randomly from the d-dimensional data set in each learning step. This neural architecture is used in a two-phase procedure. In the first, unsupervised, learning phase, the SOM iteratively adjusts its N prototypes, $\mathbf{w}_i$, according to the SOM algorithm [5] as described in Section 2.1, while the output layer is idle. Upon the convergence of the SOM, a second, supervised, learning phase can be started, in which the output neurons are trained according to the Delta rule [63]. Each neuron p in the output layer combines the SOM outputs y_i into a weighted sum:

$$l_p = \sum_{i \in A} v_{pi} y_i \qquad p = 1, 2, ..., L \tag{4.3}$$

v_{pi} is the element in the pth row and ith column of weight matrix $\mathbf{V}$, which connects the

output layer and the hidden layer (Fig. 4.1). The output layer then iteratively adjusts $\mathbf{V}$ to minimize the total squared error in the outputs l_p by the delta rule:

$$\Delta v_{pi} = \alpha y_i (l_p^d - l_p) \tag{4.4}$$

where l_p^d is the desired output, α is a learning rate. During the supervised training of the output layer weights $\mathbf{V}$, the SOM can continue its unsupervised learning with a very small learning rate, for fine-tuning of the SOM prototypes. The outputs from the output layer correspond to the inferred knowledge from the input data. When this network is used to infer latent variables, the output vector, $\mathbf{l}$ ($= [l_1, l_2, ..., l_L]^T$), yields the inferred values of the latent variables. A good implementation of this architecture is available in Neural Works Professional II/Plus by NeuralWare [64].

This supervised architecture is suitable for the analysis of high-dimensional data mainly for two reasons. One is the ease of the SOM in the handling of high-dimensional data. No prior feature extraction (dimension reduction) is needed before the learning of the data. The other is the ability of the SOM to distinguish the subtle differences between high-dimensional feature vectors. These subtle differences are well reflected in the responses (the outputs) of the SOM neurons to the feature vectors. The SOM-hybrid supervised architecture exploits the SOM's knowledge by combining the SOM outputs into weighted sums (eq. 4.3) for supervised learning of information of interest (e.g, latent variables). This architecture has helped achieve good classification accuracies [21, 22]. For example, Howell *et al.* revised Tholen's taxonomy of asteroids by analyzing the clusters identified from an SOM of asteroid spectra. The revised taxonomy was demonstrated to be more self-consistent through supervised classification [21]. Another example is the accurate classification of a large number of clay-bearing soils with subtle spectral differences due to different clay species, for landslide hazard study from AVIRIS imagery [22].

In the SOM-hybrid architecture, the customary way of using the SOM outputs by the output layer is called the winner-takes-all mode (WTA), where only the output of the SOM winner is allowed to contribute to the weighted sums. The WTA mode works well for classification problems such as in [21, 22]. In this work we apply this architecture to a multi-variable regression problem, the inference of continuous latent variables. We generalize the WTA mode to k-winners-take-all (kWTA) mode, by which the SOM's knowledge can be better exploited and consequently the latent variables can be learned with higher inference accuracies than by relying on the WTA mode. This will be discussed next.

4.2.2 Greater exploitation of the SOM's knowledge: from Winner-Takes-All (WTA) to k-Winners-Take-All (kWTA)

The customary way of using the SOM outputs: Winner-Takes-All (WTA)

By the SOM formula (eq. 2.1), the output of an SOM neuron should be indicative of the similarity between the neuron's prototype and the input vector. In the most frequent implementations, the SOM output is either proportional to the inner product of the input vector and the SOM prototype [5], or inversely proportional to the distance between the input vector and the SOM prototype. Because of the topology preserving property of the SOM, the responses to an input vector are strongly localized in the map. This means that only a few neighboring SOM neurons have relatively large output values while other neurons generate negligible responses. A customary way of utilizing the SOM outputs is the winner-takes-all (WTA) mode, where a binary thresholding is applied to the SOM responses, assigning 1 to the best matching unit (BMU) c (determined in eq. 2.1) and 0 to the rest of the neurons as

$$y_i = \begin{cases} 1 & i = c \\ 0 & i \neq c \end{cases} \tag{4.5}$$

By this, the right side of eq. 4.3 is reduced to one term.

$$l_p = v_{pc} \quad p = 1, 2, ..., L \tag{4.6}$$

With a single term left in the weighted sum, the output layer of the network will be unable to distinguish between data samples that map to the same SOM neuron and will yield the same inferred values, l_p, for these samples. In problems where the number of different values a latent variable can take is much smaller than the number of the SOM neurons, N, the WTA mode can work successfully in differentiating these various values. However, when the latent variable is continuous, i.e., the number of possible values is much larger than N, the resolution of the inferred values is severely restricted by the WTA mode, which may prevent high inference accuracies.

A generalized way of using the SOM outputs: k-Winners-Take-All (kWTA)

To relieve the above limitation in the inference resolution caused by the WTA mode, we allow multiple (k) SOM outputs to be nonzero in eq. 4.3 (k-winners-take-all or kWTA). This can be justified by the SOM algorithm: the prototypes within the lattice neighborhood of the BMU learn concurrently from the same input vector (eq. 2.2). After the SOM has converged, the memory of a data sample is stored not only in its BMU but also in the neighboring neurons of its BMU. The inclusion of these neighbors into the supervised learning can help distinguish the samples that share the same BMU but represent different values of a latent variable.

The next question is which k SOM prototypes should be allowed to contribute to the supervised learning. Since the SOM outputs reflect the similarities between the prototypes and a given data vector, the larger the output of an SOM neuron, the more knowledge the neuron contains about that data vector. Therefore, a natural choice is to use the outputs

from the first k SOM winners, indexed as i_1, i_2, ..., and i_k, in the supervised learning, such that the most of the SOM's knowledge about the given sample can be utilized. The output, y_i, of each SOM neuron i is computed by

$$y_i = \begin{cases} 1 & i = i_1 (= c) \\ \frac{d_1}{d_1 + d_i} & i = i_2, i_3, ..., i_k \\ 0 & i \neq i_1, i_2, ..., i_k \end{cases} \tag{4.7}$$

where d_i is the Euclidean distance between the prototype $\mathbf{w}_i$ and the input data vector $\mathbf{x}$. We normalize y_i to make the outputs from the SOM sum up to 1, which is consistent with the WTA mode.

$$y_{i_q} = \frac{y_{i_q}}{\sum_{q=1}^{k} y_{i_q}} \quad q = 1, 2, ..., k \tag{4.8}$$

Each output from the output layer, l_p, is now expressed as a linear combination of k nonzero SOM outputs, i.e., eq. 4.3 reduces to k terms:

$$l_p = \sum_{q=1}^{k} v_{pi_q} y_{i_q} \quad p = 1, 2, ..., L \tag{4.9}$$

The WTA mode is obviously a special case of the kWTA mode ($k = 1$). NeuralWare's implementation in Neural Works Professional II/Plus [64] provides the special cases of $k = 1$ and $k = 3$. The latter is called "interpolating mode". The implementation of kWTA in our software allows the use of any given k.

The question follows: how to choose k, i.e., the number of SOM winners to use for best learning of latent variables? The best k obviously depends on the data set as well as the SOM (the SOM size and the maturity level of the SOM, i.e., how well the SOM converged), because these factors influence the way how the information of the data samples is distributed across the prototypes. This dependence on the data and the SOM provides the

opportunity to use a principled way to constrain k to a small range, in a fast and efficient way, as will be introduced next.

4.2.3 Theoretical upper bound of k

We can determine the upper bound of k, denoted by K, from the *relative importance* of the SOM winners. For any data sample, the BMU is the most important one, containing the most information about that sample. The importance of second, third, etc., winners can be evaluated by their similarities to the BMU. In the data space, we consider two prototypes to be similar (have common information about data samples) when they are Voronoi neighbors. An obvious theoretical upper bound is $m + 1$, i.e., $K \leq m + 1$, where m is the maximum number of Voronoi neighbors to any prototype. Usually, $m+1$ is already much smaller than the total number of SOM prototypes, N, but we can further tighten this bound to $K = \widetilde{m} + 1$, where $\widetilde{m}$ is the number of "important" neighbors. The important neighbors are the prototypes that are most strongly connected to the BMU. To determine $\widetilde{m}$, we use the ranking of Voronoi neighbors of each prototype according to their connection strengths, as defined by [24]. The connection strengths between the SOM prototypes can be represented by the $N \times N$ $CONN$ matrix, proposed in [30]. An illustration of the ranking of Voronoi neighbors according connection strengths is given in Fig. 2.6, bottom right. Among the four neighbors of prototype P1, the first ranking (the most similar) to the last ranking (the least similar) neighbors are P2, P3, P4 and P5, in decreasing order of connection strengths. After ranking the Voronoi neighbors of each prototype, we can quantify the importance of the neighbors of rank i by the average connection strength s_i to the ith ranking neighbors across all SOM prototypes.

$$s_i = \frac{1}{n_i} \sum_{\substack{p,q \in A \,\wedge\, \mathbf{w}_q \text{ is the} \\ i\text{th Voronoi neighbor of } \mathbf{w}_p.}} CONN(p, q) \tag{4.10}$$

where n_i $(i = 1, 2, ..., m)$ is the total number of ith ranking neighbors in the SOM. However, thresholding on s_i to determine the important SOM winners could be problematic. First, a consistent threshold that can be used across different SOMs and data sets is impossible because s_i is dependent on the size of the SOM, the size of the data set and how the data samples are distributed across the Voronoi cells. Second, n_i is also an important factor in the importance of neighbors. When the connections to ith ranking neighbors are weak (small s_i), but n_i is large, the neighbors of rank i may still be useful in the supervised learning, because a large n_i indicates the non-negligible participation of the ith neighbors in the representation of the data samples. In view of this, it is better to consider the combined effect of s_i and n_i in the thresholding. Therefore we propose thresholding on the percentage of data samples, $\%data_i$, involved in the connections of each rank i [65]:

$$\widetilde{m} = \max\{i : \%data_i > \mu\} \tag{4.11}$$

$$\%data_i = \frac{s_i \times n_i}{2P} \times 100\% \tag{4.12}$$

μ is a user-specified threshold. P is the total number of data samples. $s_i \times n_i$ can be interpreted as the accumulated strength in the connections to all ith ranking neighbors. Normalized by $2P$, it shows the importance of these neighbors by the percentage of the total connection strength involved. With the determined $\widetilde{m}$, we obtain K, the upper bound of k, as $\widetilde{m} + 1$.

After constraining k to a small range, $0 < k \leq K$, we need to perform the supervised training with different values of k within this range to find the best value of k that yields the highest inference accuracy. We currently know no better than this exhaustive search of k, but the theoretical upper bound K significantly narrows down the range of the search and greatly alleviates the computational cost.

The threshold μ to determine $\widetilde{m}$ in eq. 4.11 is of course data dependent. For example, it depends on the noise level of the data set. In Section 4.3 , where an application to a real problem is presented, we will discuss how μ is determined.

4.3 Conjoined Twins – a new architecture – motivated by a planetary science problem

In this section we will apply the SOM-hybrid neural architecture with the kWTA mode to the inference of two latent physical parameters from high-dimensional spectra of ices. We will show that the two physical parameters are best inferred with different values of k. This motivates the idea of a new architecture we call Conjoined Twins, which combines the use of different values of k in one architecture for the best learning of both parameters.

4.3.1 Background on the planetary science problem

One intriguing problem in planetary astronomy is the modeling and interpretation of geological histories and current dynamical changes of Solar System objects. Current surface conditions of these objects, such as chemical composition and physical parameters (e.g., temperature and grain size) of the surface materials, often provide important clues for the unraveling of their geologic histories. However, it is impractical or hard to directly measure these surface parameters for extended surface areas of planets. Remote sensing spectroscopy has become a prime alternative approach, as spectroscopic instrumentation and techniques have been improved dramatically in the past decades. Current spectroscopic instruments are capable of acquiring spectral measurements at hundreds or thousands of contiguous bandpasses, whereby the detailed spectral features sensitive to surface parameters can be resolved. From these spectra, surface parameters can potentially be inferred

[66, 67].

The ultimate goal of the specific science problem we present is to infer surface parameters from Near-Infrared spectra of Pluto and Charon. The knowledge regarding the Pluto-Charon system is fairly limited due to the scarcity of observational data. NASA's New Horizons space mission, which is a one-way journey to the Kuiper belt and beyond, is expected to investigate the icy surfaces of remote planetary bodies such as Pluto, Charon, Nix, and Hydra [68]. In 2015 the onboard infrared imaging spectrometer [69] will map the surfaces of Pluto and Charon at 250 wavelengths from 1.25 to 2.5 μm. The resulting hyperspectral images will be used to unravel the surface conditions, such as ice species, distributions of different ice species, temperature and grain size of the ices, on the surfaces of Pluto and Charon. The work presented in this Chapter is a collaboration with W. M. Grundy (Lowell Observatory) and E. F. Young (Southwest Research Institute). The spectral data for the experiments were prepared by Grundy and Young to simulate conditions expected in the Pluto-Charon system.

The classification of different chemical compositions from spectra is not too difficult because specific chemical compositions are often manifested by specific spectral absorptions. Spectra of ices that possibly exist on Pluto (H_2O, N_2, CO_2, etc.) have fairly different spectral patterns. With an SOM learned with a data set containing spectra of 6 ice species, we succeeded with 100$\%$ classification accuracy in [70]. However, inference of continuous physical parameters, such as temperature and grain size, is challenging because of the following reasons. First, these physical parameters have global influence on the spectral shapes, as opposed to ions or molecular compounds, which cause *local* absorptions (limited to some wavelengths) and therefore may be determined from individual absorption bands. Second, significant changes in physical parameters can induce subtle variation in the spectral shapes. This demands algorithms capable of discerning the subtle differences

between spectra. Third, different surface parameters can interact nonlinearly. The disentanglement of different causes can be difficult. To illustrate these difficulties, we show some sample spectra of H_2O ice as a function of temperature and grain size, generated by W. M. Grundy [71], in Fig. 4.2. Temperature and grain size are two latent variables that have a global and intertwined influence on the spectral shapes. Both parameters can deepen the absorptions, e.g., at 1.3 μm and 1.65 μm, as seen from Fig. 4.2. Moreover, the change in the band depths is a nonlinear function of the parameters. From the band depths of single bands, it is hard to separate the influences of the two parameters, not to mention the accurate inference of the values of the parameters. However, if entire spectra, with multiple absorption bands, are used, the effects of temperature and grain size may be disentangled. In addition, temperature has a much more subtle effect on the spectral brightness than grain size. This results in many crossovers between spectra with different temperatures (Fig. 4.2, top). The subtle changes in spectral shapes caused by temperature makes the differentiation between temperatures difficult. A sensitive algorithm is needed to distinguish between these spectra such that accurate inference of temperatures can be made.

4.3.2 Approaches to the inference of latent surface parameters from spectra

Modeling the mapping from the surface parameters to the observable spectra is called a forward problem. It can be simulated numerically by radiative transfer models. The Hapke model, which describes the scattering and absorption of light in surfaces composed of particles of a given absorption coefficient, has now become a standard method for interpreting spectral surface reflectance data [72, 73, 74]. The Hapke model can produce model spectra for given sets of surface parameters. Conversely, inference of the surface parameters from the spectra is called an inversion problem, as shown in eq. 4.2. Since analytical so-

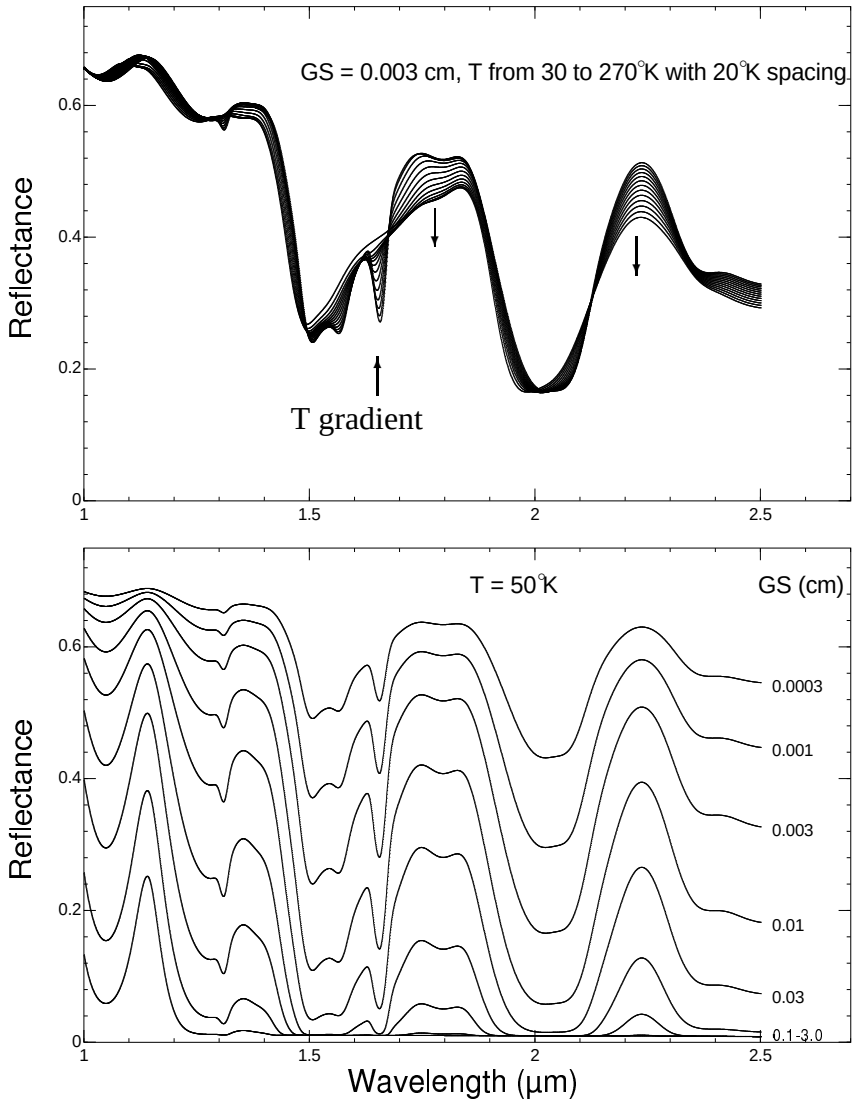


Figure 4.2: Sample synthetic spectra of crystalline H_2O ice. **Top**: Variation in the spectral shape as a function of temperature (T), for one fixed grain size (GS), 0.003 cm. **Bottom**: Variation in the spectral shape as a function of grain size, at 50 °K (Kelvin).

lutions are unavailable in general for a radiative transfer model, numerical or statistical methods are necessary to solve the inversion problem. Approaches to this inversion problem fall into three main categories: numerical optimization, look-up table and machine learning [75, 76]. Numerical optimization algorithms were the first to approach the inver-

sion problem and have been the most extensively used. These algorithms search for a best matching simulated spectrum for a real spectrum through forward search in the parameter space. They generate simulated spectra repeatedly, during the search, with parameters that are modulated to minimize a certain error function, which describes the quality of the match. Unfortunately, this approach can be inefficient because it performs the optimization for each spectrum, one by one, separately. It is thus impractical for a large spectral image (with large number of spectra). The look-up table approach expedites the numerical optimization by precomputing a large database of simulated spectra for a wide range of parameter values. The inversion problem is then reduced to searching the look-up table for a best match to a real spectrum. However, there are still issues such as how the gridding of the model parameters in the look-up table should be set. The third type, machine learning algorithms, aim to learn the mapping from the spectra to the parameters (the inverse function) through a training set of data (i.e., through supervised learning). The advantage is that once the mapping has been learned, it can be used to infer surface parameters from large data sets of spectra fast and easily. The training data can either be simulated spectra or real spectra with known surface parameters. Examples of machine learning algorithms that have been used are multilayer backpropagation (BP) neural network [76], support vector machine [77], and Gaussian Regularized version of Sliced Inverse Regression [78, 79]. The comparison of the three types of approaches in [76, 78] all showed that machine learning algorithms were more capable of achieving accurate inference accuracies than numerical optimization and look-up table approaches.

Since in this thesis work we infer two physical parameters of interest, namely temperature and grain size, here we review some related previous work. A simple and purely empirical method was proposed by Fink and Larson for retrieving H_2O ice temperatures from reflectance spectra [80]. They developed a calibration curve of a feature at 6056 cm^{-1}

($\sim$1.65 μm) and used it to determine ice temperatures for the Galilean satellites Europa, Ganymede, and the rings of Saturn. Their method was limited to objects that display H_2O ice absorptions, and was specific for the 1.65 μm feature. Another competing method to infer H_2O ice temperatures was discussed by Grundy *et al.* [71], involving construction of a suite of models with various free parameters, fitting them to the spectra with numerical optimization. This method fit the model to different segments of the spectra separately, and compared the resulting collection of best-fit temperatures. If most of the models and most of the spectral segments agreed on the temperature, that temperature was accepted as likely correct. When different models gave systematically different temperatures, the results were taken as probably meaningless. This technique worked well for applications where a small number of surface parameters are inferred for a small number of spectra. When the number of parameters of interest increases and the inference needs to be done for thousands of spectra, the optimization approach used may yield suboptimal results and the computational time can be tremendous.

We use a machine learning approach, specifically the SOM-hybrid neural architecture shown in Fig. 4.1, to learn latent surface parameters from spectra. We assess the capability of this neural architecture by inferring temperature and grain size from spectra of a single material, crystalline H_2O ice, because it is one of the most commonly found materials in the Solar System, for which a great deal of relevant data and experience have been accumulated. We focus on crystalline (as opposed to amorphous) H_2O ice because observations of Charon are consistent with crystalline ice [81] and because spectra of amorphous ice are virtually insensitive to temperature.

For the training of the SOM-hybrid neural architecture, we need a large number of training spectra for the learning of temperature and grain size. However, real spectra of Pluto and Charon with sufficient resolution in spatial and other aspects are scarce. In such

situations, and if available, realistic synthetic data (simulated data) or laboratory spectra can be used for training. For example, Gilmore *et al.* developed a carbonate identifier with a large number of laboratory spectra of carbonate and non-carbonate minerals by training a Backpropagation (BP) neural network [82]. The resulting autonomous system was successful in various simulated Martian scenarios. Ramsey *et al.* used both laboratory spectra and synthetic spectra to train a mineral identifier from Near-Infrared reflectance spectra, with a Bayesian approach [83]. In the experiments with laboratory and field spectra of a variety of solid and powdered rock samples, a recognition rate higher than what human experts could produce was shown. Similarly, we use simulated spectra for the training of our neural network (the SOM-hybrid neural architecture). We develop neural models that fit to the entire Near-Infrared spectral range (as opposed to piecewise models). Then, we assess the performance of the trained neural models with test sets of synthetic spectra. After the validity and reliability of the models are confirmed, follow-up work, beyond this thesis, will be the deployment of the trained models to infer unknown physical parameters from spectra taken from real planetary surfaces.

The synthetic spectra were produced and given to us by our collaborators W. M. Grundy and E. F. Young. The spectra were generated on a parameter grid through a radiative transfer code [84, 66] based on the Hapke model [72, 73]. The ice optical constants used in the Hapke model were also synthetically generated by Grundy with a model that fits to laboratory spectra with 17 temperature-dependent Gaussians [66]. The parameter grid has 126 temperatures with 2 Kelvin (°K) spacing between 20 and 270 °K, and 9 grain sizes logarithmically spaced from 0.0003 to 3.0 cm. This set of parameters covers a meaningful range of surface conditions for the Pluto-Charon system. The resolution of the parameters is also sufficient for scientific studies of the surface conditions of Pluto and Charon. For example, the gridding of temperature has a resolution of 2 °K, which is sufficient to resolve

the diurnal temperature changes ($\sim$ 20 °K) on Pluto. The spectral resolution, 230 band-passes in the Near-Infrared range (1–2.5 μm), is close to the resolution of the sensor used on the New Horizons spacecraft [69]. Sample spectra are shown in Fig. 4.2.

First, we will assess the neural modeling for noiseless spectra in Sections 4.3.3–4.3.5. The performance we achieve on noiseless data will serve as a benchmark in a noise sensitivity analysis we will give in Section 4.3.6. In the unsupervised training phase of the SOM-hybrid network, we use all available synthetic spectra of crystalline H_2O ice. (126 temperatures and 9 grain size yield a total of 1134 spectra.) In the supervised training phase, we conduct ten-fold jackknifing (cross-validation) to assess the performance of the trained predictive models. In each jackknife run, 1134 spectra are randomly split with a 1:9 ratio into a test and a training set. The prediction results are averages of 10 jackknife runs.

4.3.3 Manifold structure learned by the SOM

Since the unsupervised learning phase is important for assisting the fine discrimination of the spectral shapes in subsequent supervised learning, it is useful to examine how the converged SOM reflects the manifold structure and, specifically, what can be seen in terms of the influence by temperature and grain size.

We visualize the mU-matrix on the SOM in Fig. 4.3, left, and plot the prototypes (in their respective SOM grid cells) in Fig. 4.4 to find out how similar (or dissimilar) the SOM prototypes are. In both figures, the known grain size labels are overlain on the SOM. We remind the reader that these labels are not used in SOM learning. By layering the labels over the SOM, we can see whether the SOM clustering of the data coincides with the prior knowledge. In this case, we can see in Fig. 4.3, left, that the SOM is clearly separated into grain size clusters, typically by double-fenced (black) corridors of empty neurons, such as the diagonal one that separates the dark blue cluster from the yellow cluster. (We note

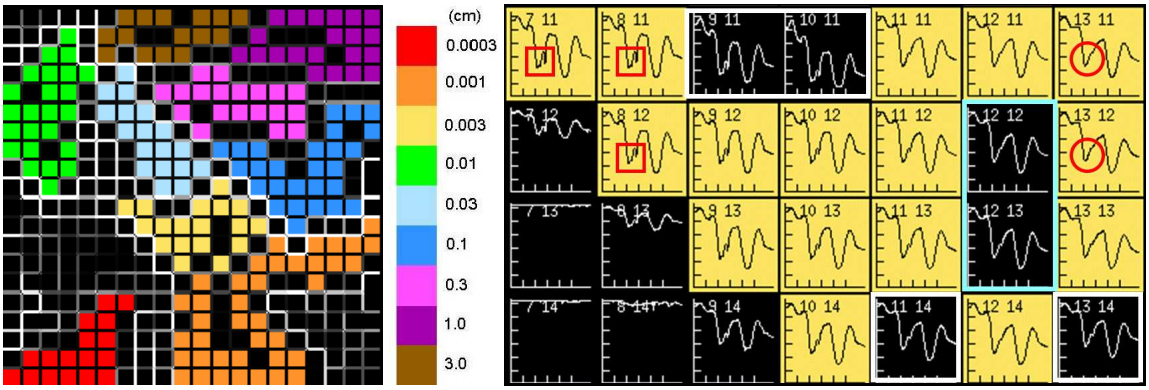


Figure 4.3: **Left**: The 20×20 SOM learned with the synthetic spectra of Pluto ices. Grid cells represent SOM neurons. In the SOM, we only color the neurons that represent spectra of crystalline H_2O ice. The colors indicate the known grain sizes as keyed at right. The "fences" between adjacent cells have grey scale intensities proportional to the Euclidean distances between the prototypes of the respective neurons (in feature space). White is large distance. The unlabeled (black) cells, such as those between the red and the green clusters, mostly indicate prototypes of spectra of ices other than H_2O ice, such as N_2 and CH_4 ice. This information is not shown here. Some black cells – typically in the narrow corridors between grain size groups, e.g., between the dark blue and the yellow clusters – are prototypes with no data mapped to them. Whether a prototype has data mapped to it is not shown in this representation. **Right**: Part of the yellow grain size group at left, magnified to show an example of how spectra are organized within a grain size group according to temperatures. Here, the prototypes are plotted in the SOM cells. A gradual change in the prototype shapes from left to right can be observed in response to increasing temperature. The red boxes and circles exemplify differences in temperature-dependent absorption features at low and high temperatures, respectively. The light blue and white boxes indicate the empty prototypes of this grain size group, inside and at the boundaries, respectively.

that although in this case clusters are delineated by double-fenced corridors, this is not a requirement for cluster separation.) This confirms that the (grain size) cluster structure has been perfectly learned by the SOM. In Fig. 4.4, we can see the variation in the spectral shapes across the grain size clusters caused by different grain sizes. The prototypes within the same grain size cluster are very similar, while the prototypes across different grain size clusters have more obvious differences. Looking more closely at the prototypes within each grain size cluster, in Fig. 4.4, we find that these prototypes are organized with respect to temperature. The temperature-dependent spectral features change in an orderly fashion from one end of the cluster to another (from top to bottom, left to right, or in other directions). Fig. 4.3, right, illustrates this for the 0.003 cm (yellow) grain size group, through an absorption feature at 1.65 μm. The prototypes learned from spectra with low temperatures

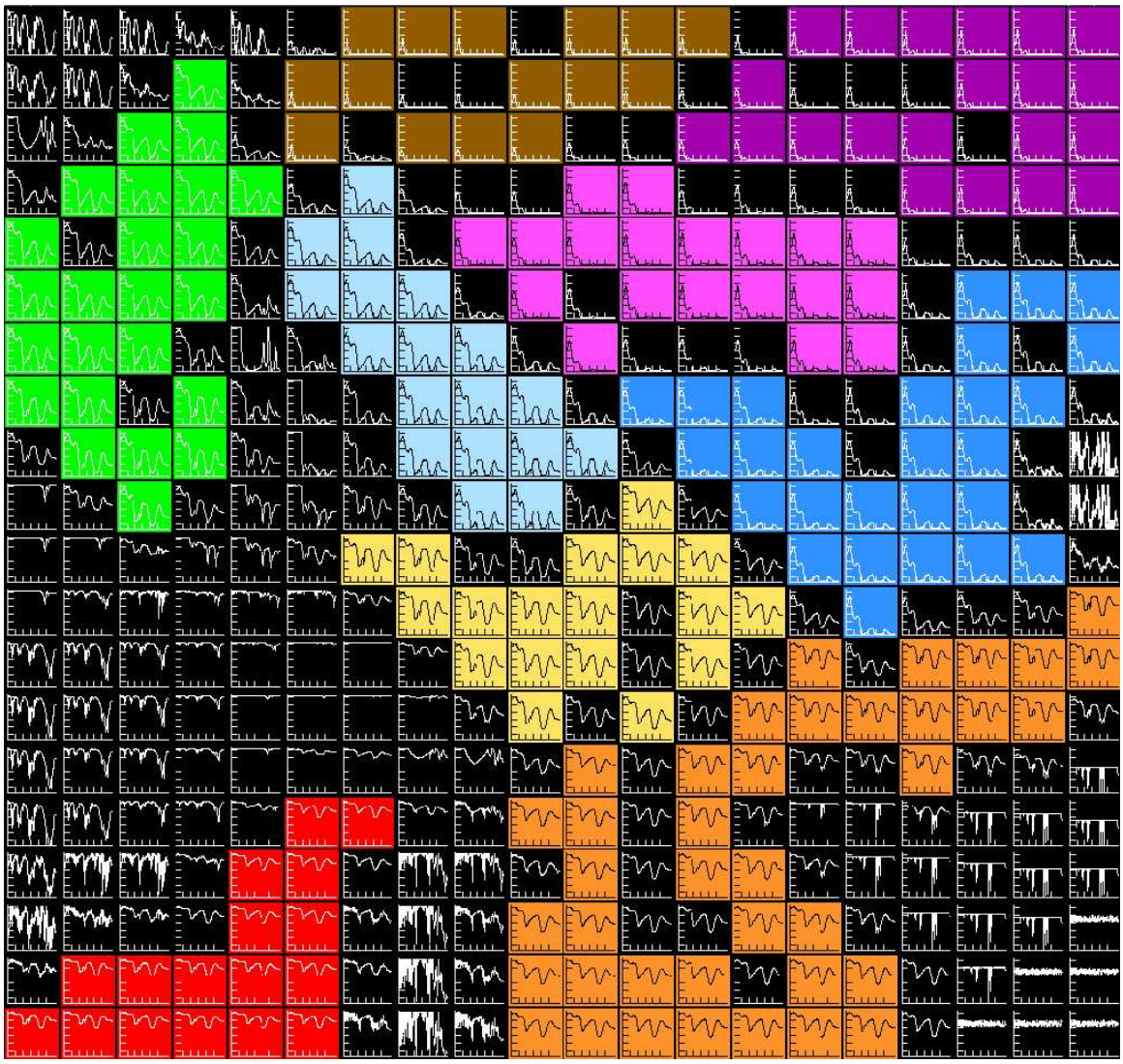


Figure 4.4: The learned prototypes plotted in their respective cells in the same SOM as in Fig. 4.3, left. The "fences" between adjacent SOM cells are not shown here for clarity. An orderly change in the temperature-dependent features in the prototypes can be observed from one end of each grain size cluster to another. This can be seen in more detail for the yellow grain size group in Fig. 4.3, right.

have a strong absorption at 1.65 μm (in red boxes). This feature gradually disappears toward the right for high temperatures (in red circles). The observation (from Figs. 4.3–4.4) that grain size has a more dominating effect on the SOM clustering than temperature can be explained by the influence of the two physical parameters on the spectral shape, as seen in Fig. 4.2. The difference in the spectral brightness is substantial between two grain size

categories (Fig. 4.2, bottom). In contrast, the changes in temperature cause much more subtle changes in spectral brightness. Temperature mainly causes shifts of band centers, as well as significant changes in relative band depths (Fig. 4.2, top). Therefore, we can conclude that the structure of this data manifold as suggested by the clustering in the SOM agrees with the spectral properties we know.

Since we have developed new tools to evaluate the faithfulness of SOM in the representation of manifold structures, instead of observing the SOM prototypes directly, we can use our new tools, the $WDTF$ and Topoview, to evaluate the quality of topology preservation in the SOM. Because our work focuses on the spectra of H_2O ice, in the calculation of the $WDTF$ and the TopoView visualization of connections the spectra of other ices are excluded. In Fig. 4.5, left, the $WDTF$ shows the existence of a few local topology violations, with small folding lengths, 2, 3 and 4. The violations with folding length 3 or 4 are extremely weak, with approximately 0.5% of contributing data samples ($0.5\% = 0.005$ in the figure). In Fig. 4.5, right, TopoView displays all connections (both violating and non-violating connections) on the SOM to help locate the violations and understand the connectedness of the manifold structure. We find that almost all connections are between immediate lattice neighbors, with folding length 1. These connections are non-violating connections and therefore not manifested in the $WDTF$. The small number of violating connections ($fl > 1$) are not only weak, as seen from the $WDTF$, but also confined within grain size clusters, reflecting the smearing of temperature representation across neighboring prototypes. These evaluations demonstrate the soundness of the map. This means that the SOM is mature enough, and we can proceed to the supervised learning phase of the SOM-hybrid neural architecture, for the learning of temperature and grain size.

The inspection of the manifold structure in the SOM also provides clues that the learning of temperature may need the help from Voronoi neighbors in the supervised learning

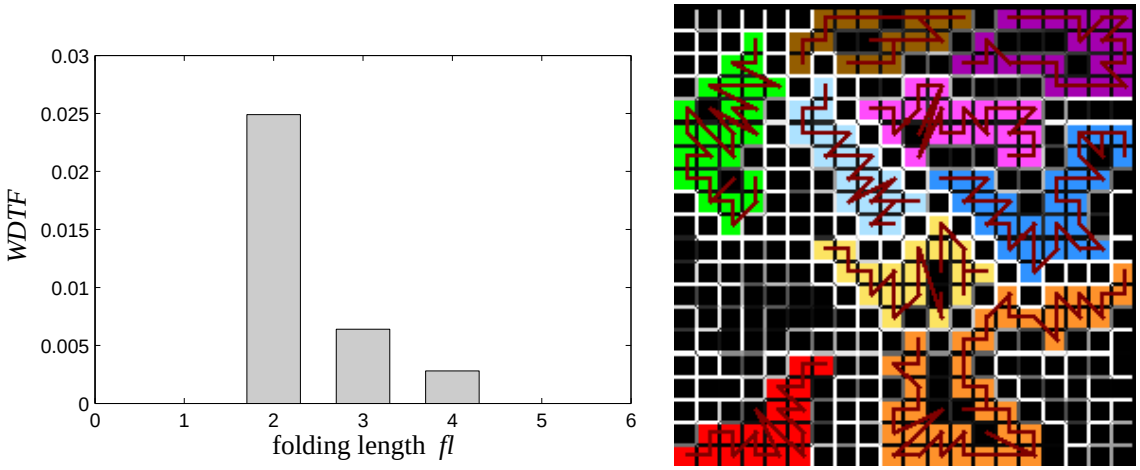


Figure 4.5: Evaluation of topological quality of the SOM learned with spectra of Pluto ice, by the $WDTF$ and TopoView. The counting of connections in the computation of the $WDTF$ as well as in TopoView only includes the spectra of crystalline H_2O ice, since our study here focuses on H_2O ice. **Left**: The $WDTF$. **Right**: TopoView visualizes all connections between prototypes on the SOM as maroon line segments. The SOM is overlain by the same color labels of 9 grain sizes as in Fig. 4.3.

(i.e., need the kWTA mode), while the learning of grain size does not need this help (i.e., the WTA mode is enough). Closer inspection of the SOM reveals that, without exception, all input spectra mapped to any prototype within a grain size cluster have the same grain size label (not shown in Fig. 4.3, left). Perfect learning of grain size thus can be easily achieved in the WTA mode, according to eq. 4.6. In contrast, with approximately 25–30 prototypes in a grain size cluster to represent 126 different temperatures, each prototype is forced to form an average (a mixture) of spectra across different temperatures. In the WTA mode, the output temperature value for any input spectrum that maps to a prototype is trained to approximate the average temperature represented by that prototype. This severely limits the resolution of the inferred values of temperature. However, the kWTA mode, which uses multiple, k, SOM winners in the weighted sum (eq. 4.9), may help better reconstruct a specific temperature.

4.3.4 Supervised learning of temperature and grain size with different $k \leq K$ from noiseless spectra

Before starting the supervised learning with different values of k, we need to first find the theoretical upper bound of k with the statistics of the connections to Voronoi neighbors, as described in Section 4.2.3, so that the exhaustive search for the best k can be constrained to a small range. From the statistics of the connections shown in Table 4.1, we know that the maximum number of Voronoi neighbors of any prototype, m, is 3. This tells us that the theoretical upper bound of k must not be larger than 4 ($K \leq m + 1 = 3 + 1$). From the statistics of the number and the average strength of connections of each rank i, namely n_i and s_i, we see that the third ranking neighbors are less important than the first two ranking neighbors in both quantity and connection strength. The connections to all third ranking neighbors involve only 0.3% of contributing data samples. With the criterion that combines n_i and s_i, in eq. 4.11, we threshold on the percentage of data samples contributing to the connections of rank i, $\%data_i$, with $\mu = 1\%$, and determine the number of important neighbors, $\widetilde{m}$, is 2. K, the upper bound of k, then equals 3 ($K = \widetilde{m} + 1 = 2 + 1$).

Table 4.1: Statistics of connections to Voronoi neighbors, from the highest to the lowest ranking, analyzed across SOM prototypes that represent spectra of H_2O ice. Spectra that represent other ices are excluded from this statistics.

	Neighbor ranking i			
	1	2	3	4
n_i	209	184	7	0
s_i	7.6	3.7	1.1	0
$\%data_i$	69.7	30.0	0.3	0

We choose the threshold μ based on the data property we know. The noisier the data, the larger μ we should choose. The data used here are synthetic spectra generated by a radiative transfer model [66]. The optical constants used in the model were obtained in laboratories, and can bring minor amount of noise into the spectra. To account for this

noise, we set the threshold as 1% based on an estimate of the influence of the noise on the data. We then justify the validity of the 1% threshold by repeating supervised learning with all values of $k \leq m + 1 = 4$ and comparing the resulting prediction accuracies. We find that with $k > K(= 3)$ (K determined by $\mu = 1\%$) the prediction accuracies are no better than the results achieved with the best k found within the range $0 < k \leq K$. This means that $\mu = 1\%$ is a valid threshold that does not cause the loss of any important neighbors. For this specific science problem, we may assume similar data properties across data sets. Therefore it is reasonable to use $\mu = 1\%$ for all other data sets generated for the same problem.

We then perform the supervised training for temperature and grain size with each $k \leq K = 3$. The prediction results of the learned models are shown as correlations between predicted and true values, in Fig. 4.6. Since both physical parameters have large ranges, we quantify the prediction accuracy as the percentage of test spectra for which the true parameter value was predicted with less than 5% relative error. The accuracies obtained with $k \leq 3$ are shown in Table 4.2, top row. The results confirm what we expected from the manifestation of the two parameters in the SOM: The WTA mode ($k = 1$) works perfectly for the inference of grain size, while the kWTA mode with $k = 3$ helps improve the prediction of temperature significantly, from 76.2% to 83.0%, compared to using $k = 1$. The relatively poor results for the prediction of temperature occur mostly at the extreme values, $\sim$20 °K and $\sim$270 °K (Fig. 4.6, left block) due to limited availability of synthetic training spectra with optical constants in these ranges. Since the temperatures on the surfaces of Pluto and Charon were estimated in literature to range between 50°K and 70°K, we can safely exclude these boundary effects. As seen in Table 4.2, bottom row, within the [50 °K, 240 °K] range temperatures are predicted with 91.8% accuracy.

From the above we find that different values of k, 1 and 3, are best for the inference of

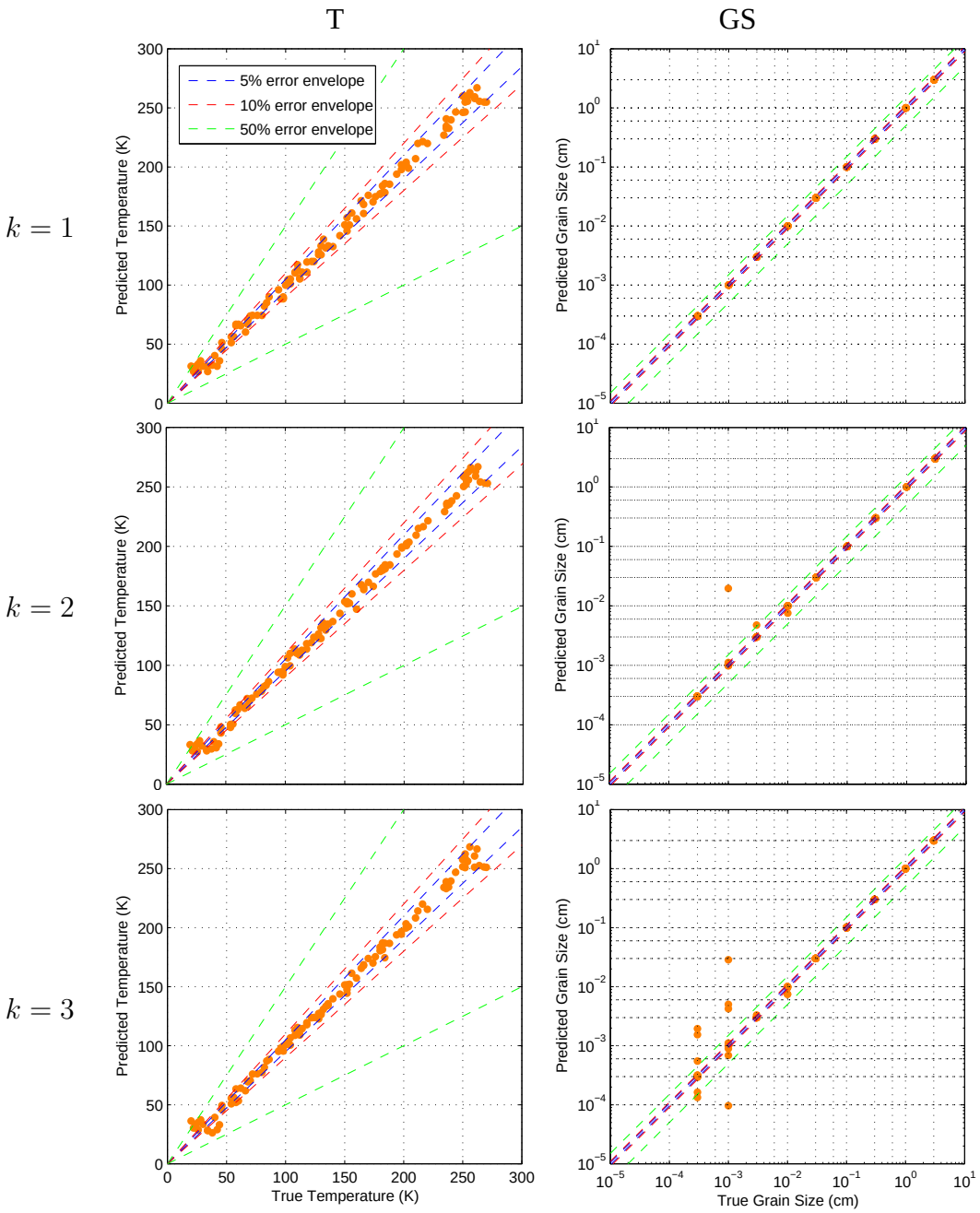


Figure 4.6: Correlation of predicted and true values of temperature, T (**left block**) and grain size, GS (**right block**). Data are shown as orange dots. Results are obtained with $k = 1$ (**top row**), $k = 2$ (**middle row**) and $k = 3$ (**bottom row**). The blue, red and green dashed lines indicate 5%, 10%, and 50% error envelopes for the prediction, respectively. The temperature has the smallest prediction error with $k = 3$. The prediction of grain size is best with $k = 1$.

Table 4.2: Prediction accuracies of grain size (GS) and temperature (T) with the spectra of H_2O ice, with $k \leq 3$, calculated for the whole data set with T∈[20 °K, 270 °K] and for the subset of data with T∈[50 °K, 240 °K], respectively. Results are averages of 10 jack-knife runs.

T (°K)		$k = 1$	$k = 2$	$k = 3$
20–270	GS	**100.0±0.0%**	85.4±4.6%	76.4±4.4%
	T	76.2±2.6%	80.1±2.7%	**83.0±2.7%**
50–240	GS	**100.0±0.0%**	–	73.7±4.7%
	T	82.3±3.7%	–	**91.8±1.2%**

grain size and temperature, respectively. Next, we do an experiment to investigate whether best results can be simultaneously achieved with a uniform k. A natural thought to improve the grain size prediction with $k = 3$ is to increase the grid resolution of the training data. We insert 9 additional grain sizes, evenly spanned on a logarithmic scale, between each two adjacent grain sizes in Fig. 4.2, bottom, to construct a data set with 81 grain sizes. We use an SOM of the same size (20×20) and an SOM of an increased size (40×40) to learn this data set. Due to color limitation, we group every 9 consecutive grain sizes, in ascending order, into 9 grain size supergroups. Fig. 4.7 shows the two SOMs, overlain with color labels of the 9 grain size supergroups. However, increasing the grain size resolution in the training data, or increasing both the grain size resolution and the SOM size to 40×40, is not helpful in this case, as seen from Table 4.3. With the same SOM size (20×20) and 9 times more grain sizes, we find that the grain size clusters do not separate clearly as in the SOM learned with the data set with 9 grain sizes. Each prototype is forced to represent not only a mixture of different temperatures, but also a mixture of different grain sizes. This can be observed from the connections that cross the boundaries of the grain size supergroups in Fig. 4.7, left. It also explains why multiple winners ($k = 3$) achieve higher accuracy (78.8%) for the inference of grain size in this case than a single winner ($k = 1$) (74.1% accuracy), in contrast to the case with 9 grain sizes (Table 4.2, top left block). The overall performance suggests that the variations in the data caused by the two physical

parameters are insufficiently represented in the small SOM, due to which they cannot be inferred as accurately as for the data set with 9 grain sizes. When increasing the size of the SOM to 40×40, we observe that 81 grain size clusters are separated from each other almost cleanly. This can be seen from 9 clear strings of connections, which represents 9 different grain sizes, in most of the supergroups (Fig. 4.7, right). For example, in the dark blue supergroup, 9 strings are nicely aligned, oriented along the vertical direction. The almost clean separation between grain size groups helps recover the inference accuracy of grain size to near perfection (97.8%) with $k = 1$. However, it is unable to recover to 100% accuracy because in the supergroups where entangled connection strings exist, such as indicated by the yellow oval, some of the prototypes still represent spectra from different grain size groups. The inference accuracy of temperature is also poorer than in the case of 20×20 SOM and the data set with 9 grain sizes. This can be explained by the fact that the 40×40 SOM allocates, on average, approximately 19 prototypes to each grain size group, which is around two thirds of the number ($\sim$ 30) allocated by the 20×20 SOM for each of the 9 grain size groups.

Table 4.3: Prediction accuracies of grain size (GS) and temperature (T) for two separate data sets, containing 9 and 81 grain sizes, respectively, with 20×20 and 40×40 SOMs, each with $k = 1$ and $k = 3$, respectively [85]. Results are averages of 10 jack-knife runs.

		Data set with 9 grain sizes		Data set with 81 grain sizes	
		$k = 1$	$k = 3$	$k = 1$	$k = 3$
20×20 SOM	GS	**100.0±0.0%**	76.4±4.4%	74.1±1.5%	**78.8±1.7%**
	T	76.2±2.6%	**83.0±2.7%**	31.9±1.2%	**52.5±1.8%**
40×40 SOM	GS	–	–	**97.8±0.4%**	54.5±1.2%
	T	–	–	60.3±1.2%	**77.9±1.0%**

We can conclude from the above that larger SOM size and more grain size samples may not get us closer to better overall prediction with a uniform k. Although it is possible that with an SOM even larger than 40×40, or with many more training steps, we may be

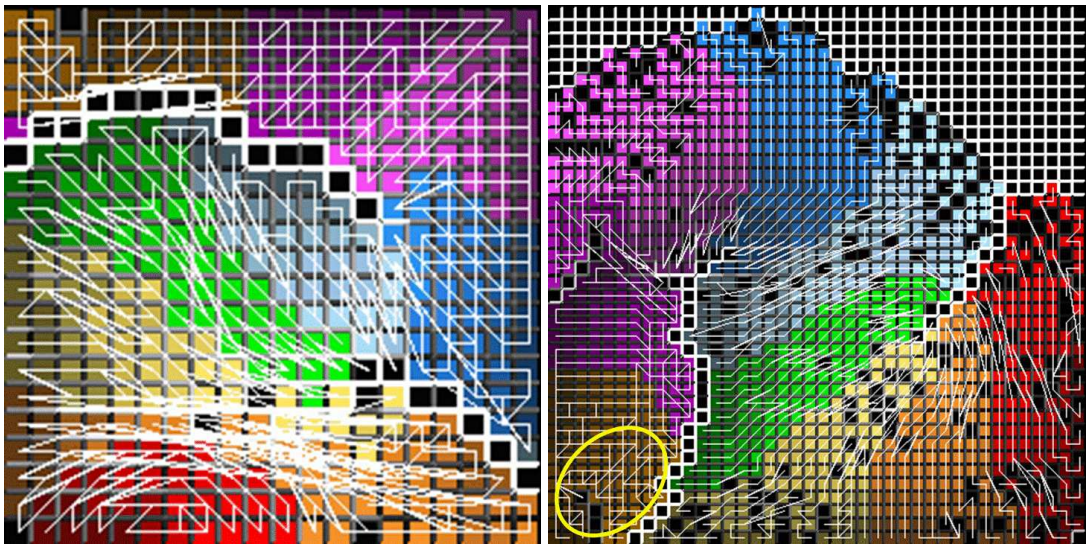


Figure 4.7: TopoView visualizes all connections between prototypes (thin white lines) on the two SOMs that learned the data set with 81 grain sizes. The same color coding of grain sizes is used as in Fig. 4.3, but here each color represents a grain size supergroup, which contains 9 consecutive grain sizes out of the 81. Dark to bright shading of colors expresses proportional, low-to-high, temperature values. **Left:** The 20×20 SOM. **Right:** The 40×40 SOM. The yellow oval indicates an example of the entanglements between the connections, resulting from undefined boundaries between grain size groups.

able to achieve the same accuracies as in Table 4.3, top left, the extra resources and time required make that approach undesirable for practical purposes. However, we can encode the use of two different values of k, 1 and 3, for the learning of temperature and grain size, into one architecture, which we call Conjoined Twins. This new architecture is then able to achieve the best accuracies simultaneously for the two physical parameters, while adding minimal overhead to the SOM-hybrid neural architecture (in Fig. 4.1). We will introduce the Conjoined Twins architecture in detail next.

4.3.5 Conjoined Twins: using different values of k for learning different latent variables

The Conjoined Twins architecture

The idea of the Conjoined Twins is to allow preferential use of different values of k for the inference of different latent variables [85, 70]. The Conjoined Twins architecture has the same structure as the SOM-hybrid architecture (Fig. 4.1), but with "twin heads", two copies of the output layer, as in Fig. 4.8. Both "heads" share the same "body" of knowledge in the SOM, but use it in customized ways. Head $\#1$ pulls the SOM output only from the BMU ($k = 1$) for the training of the output layer (eq. 4.6). This head becomes a grain size specialist achieving perfect prediction for grain size. The prediction of temperature from its second output neuron is discarded. Similarly, head $\#2$ specializes on temperature by drawing the outputs from the first three BMUs and forming a three-term weighted sum according to eq. 4.9 ($k = 3$). The grain size prediction from this head is discarded. With the Conjoined Twins we obtain high prediction accuracies for both parameters by minimal additional computational cost, compared to increasing the SOM size or adding more training steps, as discussed next.

Computational cost

The increase in computational cost with an additional "head" is relatively small compared to increasing the size of the SOM, for two reasons. First, the training of the SOM is typically longer (takes more training steps) than the training of the output layer. Second, the cost of each training step of the SOM is much larger than the cost of a training step in the output layer. With N SOM prototypes and dimension D, it takes $(3D + 6)N$ operations for the SOM to calculate the distances between an incoming input vector and all the prototypes, for winner selection (eqs. 2.9–2.11), and $4DN$ operations for updating the SOM prototypes

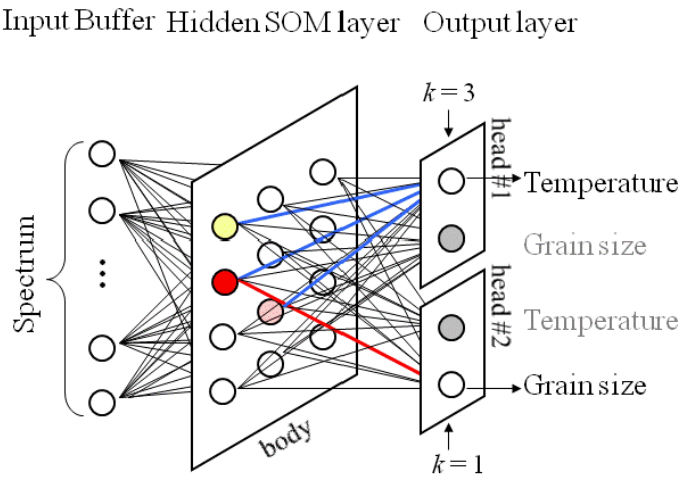


Figure 4.8: Conceptual diagram of the Conjoined Twins architecture for best inference of temperature and grain size. Head #1 uses the output from the BMU (red neuron in the SOM) to predict grain size. Head #2 uses, in addition, the second and third BMUs (pink and yellow neurons) to predict temperature.

$\mathbf{w}_j$ (eq. 2.2). For a 20×20 SOM ($N = 400$) that learns 230-dimensional data ($D = 230$), it takes 646,400 (=278,400+368,000) operations to learn from one input vector. For a 40×40 SOM ($N = 1600$), the number of operations is 2,585,600 (4 times larger) for one learning step. In contrast, adding a "twin head" carries a small overhead. In the WTA mode, one training step has $5N$ operations: N for setting SOM outputs (eq. 4.5), $2N-1$ for calculating y_p^{OUT} in the output layer (eq. 4.3), and $2N+1$ for updating v_{pi} (eq. 4.4). In the kWTA mode, one step costs $3k + 5N - 2$ operations, where k is the customized number of SOM winners to be used by this "head". This includes $k + N - 1$ operations for setting the SOM outputs (eq. 4.7), $2k - 1$ operations for normalizing the SOM outputs (eq. 4.8), $2N - 1$ operations for calculating y_p^{OUT} in the output layer and $2N + 1$ for updating v_{pi}. k is typically a small number ($\ll N$). For a 20×20 SOM, 2000 additional operations are needed for a "head" in the WTA mode, or 2007 operations for a "head" in the kWTA mode ($k = 3$). Hence, the extra computational cost of adding a "twin head" is negligible, and independent of the data dimensionality. This makes the Conjoined Twins approach especially suitable for high-dimensional data.

We can achieve 100% accuracy for the prediction of grain size in the kWTA mode (k=3) by running the supervised phase for $\sim$2 million steps, more than twice as long as with the Conjoined Twins (750,000 steps). This means the inclusion of 3 SOM winners can produce good prediction for both temperature and grain size, but more computational time is required by this machine with a single mode (a uniform k for supervised learning) to perform as well as the Conjoined Twins for the inference of grain size. For a 20$\times$20 SOM ($N = 400$) and 230-dimensional data ($D = 230$), it takes a total of $\sim$2.1$\times 10^{11}$ (=(276,000+2000+2007)$\times$750,000) operations for the Conjoined Twins to learn both parameters well. The machine with the kWTA mode with $k = 3$ needs $\sim$5.6$\times 10^{11}$ operations (=(276,000+2007+2007)$\times$2,000,000) to achieve similar results. Thus, the Conjoined Twins approach is a more economical solution to this parameter inference problem.

Inference results with Conjoined Twins

We achieve perfect, 100.0$\pm$0.0% prediction accuracy for grain size and 83.0$\pm$2.7% for temperature (Table 4.3, top left block), using the Conjoined Twins architecture. If we exclude the problematic end regions of the temperatures where adequate training data are unavailable, the prediction accuracy for temperature in the remaining 50–240 °K range is 91.8$\pm$1.2% (Table 4.2). For Charon, and for regions of Pluto free of N_2-ice, diurnal and latitudinal temperature variations of tens of °K are expected, with temperatures in the the 50–70 °K range. Our neural model is desired to retrieve temperatures with less than $\sim$3 °K error in order to resolve the temperature differences to $\sim$6 °K, which is sufficient for defining the tens of °K diurnal and latitudinal temperature changes. This will further help map the thermal inertia across these surfaces. Since 3 °K represents $\sim$ 5% error in the 50–70 °K temperature range and we already use 5% error envelope in the computation of prediction accuracy, we can assume such performance for 91.8$\pm$1.2% of the measured spectra with

temperature between 50–70°K, according to Table 4.2. This means that 91.8±1.2% of the predictions from our model will be useful to assist scientific analysis of the geological histories of Pluto and Charon.

4.3.6 Noise sensitivity analysis

As our ultimate goal is to infer surface parameters from real spectra acquired in space missions under various noise conditions, we conduct a noise sensitivity analysis to evaluate the robustness of our neural model. To address the noise conditions that are common for spectral measurements, our collaborators, Young and Grundy, added noise to the noiseless data set (1134 spectra), producing noisy versions of the data with seven different Signal-to-Noise Ratio (SNR) levels, SNR=256, 128, 64, 32, 16, 8 and 4. Properties of the added noise reflect their knowledge about the noise in real spectra [70]. They generated two batches of noisy data sets, one batch with one noisy version, the other with 10 noisy versions for each noiseless spectrum. We refer to these as NoisyData1 (7×1134=7938 spectra) and NoisyData10 ($7\times10\times1134$=79,380 spectra).

The noise sensitivity analysis consists of two parts.

First, we compare the models trained on the NoisyData10 data set with five SNR levels, infinity (inf), 256, 128, 64 and 32, as shown in Table 4.4. The inference capabilities of the resulting models are tested on data with eight different SNR levels, inf, 256, 128, 64, 32, 16, 8 and 4. For each case, we do 3 three-fold jackknife runs. The training set for each case comprises randomly selected two thirds of the spectra with the training SNR. The remaining one third of the spectra, together with the spectra with other SNR levels, make up the corresponding test set.

Second, we investigate the influence of the size of the noisy training set on the prediction accuracy by comparing the models trained with the NoisyData1 and the NoisyData10

data sets.

In both experiments, we reuse the 20×20 SOM that learned with the noiseless data (1134 spectra) and train the "twin heads" with the noisy data in the supervised learning phase. A rationale for reusing the SOM is that we expect to train the SOMs in our models mostly with synthetic data, thus, we have no limitation in using noiseless data. In addition, the use of the same SOM (trained with noiseless spectra) across all cases helps separate the effect of training the "twin heads" with different noisy data sets from the effect of training the SOM with noisy data. A follow-up task should be to assess what noise levels make significant difference in training the SOM.

Results with the NoisyData10 data set

The best results in Table 4.4 (numbers in bold face) show that the difference between the SNR levels of the training and the test data is important for the prediction of temperature. The training set with the highest SNR, namely the noiseless set, does not always generate the best predictions. For instance, from the noisy data with SNR=64, temperature is best inferred with the model trained on data with SNR=128. This makes sense because training with noisy data is similar to training with a larger variety of noiseless training samples that have the same variance as the noisy data. This helps the prediction from noisier samples which have an even larger variance than the training data. Table 4.4 suggests that the training sets with SNR 2–8 times as high as the SNR of the test data produce the highest accuracies. In contrast, for grain size, the noiseless training set produces the best prediction accuracy for all test sets with SNR$\geq$16. The markedly lower accuracies produced by the models resulting from noisy training data can be explained by noise-induced blurring of the boundaries between grain size groups. Two noisy spectra with different grain sizes can map to the same SOM prototype at the boundary of two clusters (such as the ones in the

white boxes in Fig. 4.3, right). This causes confusion during the training of the grain size specialist "head". For the test sets with SNR levels 4 and 8, the best results are produced by the models trained on data with SNR=64. However, for these two test sets, the advantage of the best models over others is small ($\leq$2.0% increase in accuracy), thus may not be conclusive.

Table 4.4: Prediction accuracies for temperature (T) and grain size (GS) tabulated for different SNR levels of the training and test data. Each prediction accuracy is an average of 3 jack-knife runs. The numbers in bold face are the best prediction accuracies for test data. Variances of all prediction accuracies are less than 2.7 for T, and less than 0.3 for GS, as shown in Table 4.9.

		T prediction accuracy (%)					GS prediction accuracy (%)				
Training SNR		inf	256	128	64	32	inf	256	128	64	32
Test SNR	4	45.5	46.0	**46.1**	46.0	**46.1**	87.4	87.7	88.5	**89.2**	87.8
	8	55.6	56.0	56.3	**56.4**	55.9	94.5	94.7	95.3	**95.9**	93.9
	16	64.7	65.8	**66.0**	65.9	65.6	**97.4**	**97.4**	97.1	**97.4**	93.9
	32	72.7	73.7	**74.0**	73.0	71.5	**98.8**	98.5	97.9	97.5	93.6
	64	78.4	**79.3**	**79.3**	77.0	74.3	**99.3**	98.8	98.2	97.5	93.3
	128	82.0	**82.4**	81.7	78.1	74.9	**99.8**	99.0	97.9	97.2	93.1
	256	**83.5**	83.3	81.8	78.2	74.6	**99.9**	99.0	97.9	97.2	92.8
	inf	**83.0**	82.5	80.8	76.1	72.3	**100.0**	99.0	97.8	97.1	93.0

inf: infinity

Comparison of results obtained with NoisyData10 and NoisyData1

Tables 4.5 and 4.6 show the prediction accuracies achieved with the NoisyData10 and the NoisyData1 data sets, respectively, for training SNR 32, 64 and 128. Their difference, in Table 4.7, indicates the improvements in prediction accuracies due to the larger sizes of the training sets. For the prediction of temperature, the improvement in accuracy is prominent when the test data set has an SNR level at least twice as high as the SNR level of the training data. When the SNR level of the training set is 8 times as large as the SNR level of the test data, the advantage of using 10 noisy versions for training vanishes. For the prediction of grain size, however, the tendency is consistent. The results with NoisyData10 are always better than with NoisyData1. One general conclusion from Table 4.7, for both parameters,

is that in most cases the noisier the training set the greater improvement in accuracy can be achieved with more (in this case 10 times more) noisy training spectra.

The above results demonstrate good consistency in the performance of the neural models under a wide range of noisy conditions. The statistics in Tables 4.4–4.7 will help choose the most suitable model for inference of temperature and grain size from real spectra when noise estimate for real data is available.

Table 4.5: Prediction accuracies produced with the NoisyData10 data set, containing 10 noisy versions for each noiseless spectrum. This table shows a subset of Table 4.4, for easy comparison with Table 4.6.

		T accuracy (%)			GS accuracy (%)		
Training SNR		128	64	32	128	64	32
Test SNR	4	46.1	46.0	46.1	88.5	89.2	87.8
	8	56.3	56.4	55.9	95.3	95.9	93.9
	16	66.0	65.9	65.6	97.1	97.4	93.9
	32	74.0	73.0	71.5	97.9	97.5	93.6
	64	79.3	77.0	74.3	98.2	97.5	93.3
	128	81.7	78.1	74.9	97.9	97.2	93.1
	256	81.8	78.2	74.6	97.9	97.2	92.8
	inf	80.8	76.1	72.3	97.8	97.1	93.0

Table 4.6: Prediction accuracies produced with the NoisyData1 data set, containing one noisy version for each noiseless spectrum. Entries are missing when the test and the training SNR levels coincide, because in these cases the single noisy version is included in the training set, leaving the test set empty.

		T accuracy (%)			GS accuracy (%)		
Training SNR		128	64	32	128	64	32
Test SNR	4	46.6	46.3	46.0	88.5	87.8	87.0
	8	56.7	57.0	56.9	94.4	92.4	90.9
	16	66.7	65.3	64.3	94.8	93.6	89.4
	32	73.2	71.7	–	97.3	96.5	–
	64	78.9	–	70.3	97.5	–	90.4
	128	–	76.2	70.5	–	94.9	89.0
	256	79.3	75.7	70.5	96.8	95.7	89.4
	inf	77.1	71.9	67.4	96.6	95.3	89.4

Table 4.7: The difference of Table 4.5 and Table 4.6, showing improvements in prediction accuracies by using 10 noisy versions instead of one. Values greater than 1.5% are in bold face.

		Improvement in T accuracy (%)			Improvement in GS accuracy (%)		
Training SNR		128	64	32	128	64	32
Test SNR	4	-0.5	-0.3	0.1	0	1.4	0.8
	8	-0.4	-0.6	-1.0	0.9	**3.5**	**3.0**
	16	-0.7	0.6	1.3	**2.3**	**3.8**	**4.5**
	32	0.8	1.3	–	0.6	1.0	–
	64	0.4	–	**4.0**	0.7	–	**2.9**
	128	–	**1.9**	**4.4**	–	**2.3**	**4.1**
	256	**2.5**	**2.5**	**4.1**	1.1	**1.5**	**3.4**
	inf	**3.7**	**4.2**	**4.9**	1.2	**1.8**	**3.6**

4.3.7 Comparison between Conjoined Twins and backpropagation (BP) network

Because backpropagation (BP) network [55, 56] is a well-known universal function approximator and it is popular in spectral classification problems [82, 86, 87], we compare it with the Conjoined Twins through the same planetary science problem, the inference of temperature and grain size from spectra of H_2O ice. A brief introduction of the BP network is given in Appendix C.

The BP network we use is a two-layer network with 2 neurons in the output layer and 40 neurons in the hidden layer. The 2 output neurons generate inferred values of temperature and grain size, respectively. The number of neurons to use in the hidden layer is determined by trial and error. We also preprocess the grain size values with a logarithmic filter before the network training. The filter maps the logarithmically spaced grain size values, which span 5 magnitudes, to a linear scale between -5 and 1. This logarithmic transformation improves the prediction accuracy of grain sizes from 47.4% to 99.9% for noiseless spectra with the BP network. The reason could be that the logarithmic transformation changed the shape (smoothness and steepness) of the error surface for this specific problem such that

the minimum error could be easily achieved by gradient descent. Inference accuracies of temperature and grain size are evaluated after 600 million training steps for each case with different training and test SNR values shown in Table 4.8. Compared with the Conjoined Twins, which takes 1 million training steps (250,000 steps for unsupervised learning and 750,000 steps for supervised learning) to achieve satisfactory prediction accuracies, the BP network takes much more steps. The supervised learning of the Conjoined Twins is faster than the BP network because when the SOM converged the supervised training involves only the output layer and the training is reduced to a linear regression, where the outputs of the output layer are weighted sums of the SOM outputs (eq. 4.3). In contrast, in the BP network the supervised learning involves the training of both the hidden layer and the output layer, and it is a nonlinear regression because an additional nonlinear transfer function is applied on the weighted sums in both the hidden layer and the output layer (eqs. C.1–C.3). These make the BP network require much more computational efforts to converge than the Conjoined Twins.

Table 4.8: Prediction accuracies achieved with a two-layer BP network for temperature (T) and grain size (GS) tabulated for different SNR levels of the training and test data. Each prediction accuracy is an average of 3 jack-knife runs. The numbers in bold face are the maximum prediction accuracies for test sets.

		T prediction accuracy (%)					GS prediction accuracy (%)				
Training SNR		inf	256	128	64	32	inf	256	128	64	32
Test SNR	4	**36.1**	33.6	34.3	33.6	34.3	36.8	25.3	28.3	27.9	**38.9**
	8	**44.8**	41.8	42.7	42.2	43.5	45.4	33.1	36.5	36.2	**48.8**
	16	**55.1**	52.0	53.1	51.9	54.0	55.5	42.9	46.4	46.9	**60.7**
	32	**67.0**	63.6	64.8	63.2	64.8	69.1	56.0	59.6	60.1	**74.1**
	64	**78.8**	73.8	74.5	73.3	71.5	84.3	70.9	74.8	74.5	**84.4**
	128	**88.8**	81.3	82.0	79.9	76.3	**95.7**	81.1	84.5	83.8	88.7
	256	**93.9**	83.8	84.4	82.3	77.8	**99.6**	85.1	87.8	86.9	89.7
	inf	**95.5**	84.5	84.9	83.0	78.5	**99.9**	86.2	88.8	88.6	89.8

From noiseless spectra of H_2O ice, the BP network achieves 95.5% and 99.9% accuracies for temperature and grain size, respectively (Table 4.8). This means that the BP

network can infer grain size similarly well as the Conjoined Twins (99.9% vs.100.0%), and it outperforms the Conjoined Twins by approximately 12% in the prediction for temperature (95% vs. 83%). Nevertheless, we note that the equally good performance for the prediction of grain size is due to the logarithmic transformation, which requires the prior knowledge of the data.

Moreover, from Table 4.8, we find that the BP network is less robust to noise than the Conjoined Twins. When the noise level of the test spectra increases, the best inference accuracies from models trained on spectra with different noise levels all drop more steeply compared to the Conjoined Twins for both physical parameters. For example, for test spectra with SNR=32, the BP network is worse than the Conjoined Twins (67.0% vs. 74.0% for temperature and 74.1% vs. 98.8% for grain size). The Conjoined Twins' remarkable robustness to noise results from the use of the SOM, which naturally mitigates the effect of noise owing to the vector quantization procedure. In contrast, the BP network has no mechanism to relieve the influence of noise in the input vector. To compare the reliability of the models from the two neural modeling approaches, we also calculate the statistics of the standard deviations of the inference accuracies in Tables 4.4 and 4.8. From Table 4.9, the Conjoined Twins architecture shows higher reliability with significantly smaller standard deviations of the prediction accuracies.

Table 4.9: Statistics of the standard deviations of the prediction accuracies shown in Table 4.4 and 4.8. The standard deviations are not shown in Table 4.4.

Neural modeling approach		mean	std	min	max
Conjoined Twins (Table 4.4)	T	0.2	0.4	0.0	2.7
	GS	0.1	0.1	0.0	0.3
Backpropagation (Table 4.8)	T	1.3	0.8	0.2	3.4
	GS	5.1	3.2	0.2	11.6

From the above, we conclude that the Conjoined Twins is better than the BP network for this particular inference problem, primarily for two reasons. First, the Conjoined Twins

does not need a preprocessing of data while the BP network depends on the logarithmic transformation to achieve high prediction accuracy for grain size. Second, the models produced by the Conjoined Twins architecture have higher degree of robustness to noise and higher reliability than the models produced by the BP network. These properties are important and desirable, especially in future deployments where physical parameters will be inferred from noisy spectra collected from real planetary surfaces. Additionally, the Conjoined Twins is more economical than the BP network. The supervised learning in the Conjoined Twins is much faster than in the BP network.

4.3.8 Conjoined Twins architecture for the inference of multiple latent variables

The Conjoined Twins architecture proposed above has two heads for the supervised learning of two physical parameters. This can be conceptually extended to an architecture with multiple, more than two, "heads" for the learning of multiple latent variables, as illustrated in Fig. 4.9. All "heads" rely on the same "body" of knowledge, the learned SOM, but each draws from a different, k_i, number of SOM winners in the weighted sum (eq. 4.9) for best learning of the latent variable l_i.

The essential part of building the Conjoined Twins architecture with multiple "heads" is the customization of k_i for each latent variable l_i. Procedurally, customization for more than two "heads" ($L > 2$) is the same as for two "heads" ($L = 2$), as conducted in Section 4.3.4. It should follow the two-step procedure we propose [65]. In the first step, we determine the collective upper bound, K, of k_i for all latent variables according to the statistics of the connections, by eq. 4.11 in Section 4.2.3. This is to find how many SOM winners are *sufficient* to represent the information in a data sample. The second step is to search for the best k_i for each latent variable l_i, i.e., the *necessary* smallest $k_i \leq K$ for the

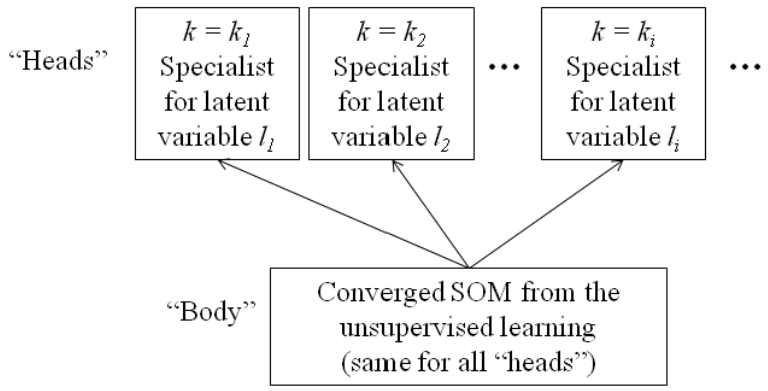


Figure 4.9: Conceptual diagram of the Conjoined Twins architecture for inference of multiple latent variables l_1, l_2,..., l_L. The Conjoined Twins architecture has multiple "heads", each of which preferentially uses a different number of SOM winners, k_i, to achieve the best inference accuracy for the latent variable l_i.

supervised learning of l_i. Since the search range of k_i has already been narrowed by K, we perform an exhaustive search by repeating the supervised training phase K times with $k = 1, 2, ..., K$ and selecting k_i with which we obtain the highest inference accuracy for l_i. After completing these two steps, we can "mount" the "heads" with different k to the SOM "body" such that the resulting Conjoined Twins architecture (Fig. 4.9) can infer all these latent variables with high accuracies simultaneously.

When the number of latent variables to be inferred is larger than two ($L > 2$), new problems might arise due to the increased level of interplay across the latent variables manifested in the SOM clustering. For example, more latent variables in the data indicates higher intrinsic dimensionality of the data. With increasing number of latent variables, the mismatch of the dimensionality between the data manifold and the 2-dimensional SOM lattice can increase accordingly. This may result in increasing level of topology violations in the SOM, and may consequently lead to poor supervised learning. In that case, more research will be needed to carefully evaluate the success of the Conjoined Twins, and more innovations may be required for issues related to more than two latent variables.

Chapter 5

Summary and future directions

Data collected for real world problems often pose considerable challenges for information extraction algorithms, due to the high dimension of the data as well as the convoluted dependencies across the data dimensions. Supervised machine learning algorithms can model the relationships, regardless of their complexity, between the high-dimensional data (observable variables) and certain information of interest (latent variables), when example data (labeled data) are available. Success, however, depends on the quality of the data labels. Unsupervised machine learning, which reveals the hidden patterns and regularities in the data, can provide additional objective information to support the supervised learning, and hence can help improve the capabilities of supervised learning algorithms (e.g., by detecting mislabeling).

This thesis work focuses on a powerful unsupervised neural learning paradigm, the Self-Organizing Map (SOM), which has been studied extensively and has been successful for the analysis of high-dimensional data in recent decades. The essential property of the SOM, topology preservation, enables a faithful representation of the structure of high-dimensional data manifolds on a low-dimensional lattice, from which relevant information of the data can be extracted. However, topology violations are not unusual, especially in

the learning of real data. One contribution of this work is the development of new measures and visual tool for evaluation of topology preservation in SOMs. Assuming good quality in SOM learning, we can further employ the learned SOM's knowledge for supervised learning of latent variables from data, by incorporating the SOM into a supervised architecture. Another contribution of this work is the proposition of an innovative supervised learning architecture, the Conjoined Twins, which enables the optimal uses of the SOM's knowledge for the inference of different latent variables. We summarize these contributions here briefly, along with further insights and outlooks.

New measures and visual tool for evaluation of topology preservation in SOMs

Topology preservation is an essential property of the SOM, but it can be lost for various reasons, e.g., improper parameterizations of learning. Deterioration of topology preservation can lead to incorrect understanding of the manifold structure, and consequently, inaccurate information extraction. Measures and tools that effectively monitor topology violations are hence desirable so that the user can make remediations to improve the learning.

We advanced the state-of-the-art by further developing one of the best measures available, the Topographic Function (TF). The TF is advantageous than other measures because it uses the induced Delaunay graph as the distance metric in data space, which correctly interprets the neighborhood relationships for data manifolds with nonlinearities and discontinuities, and because it displays both the forward and backward violations for different scopes of violations (Section 3.2.2). However, the TF has three drawbacks. 1. It is an integral function, counting the average number of violations with folding lengths larger than a given fl, while in most situations we may be more interested in its differential information, i.e., the number of violations at each specific fl. 2. The TF can not be used directly for comparison across SOMs learned with different data sets or SOMs with dif-

ferent sizes, because of the lack of normalization in its definition. 3. The TF makes no difference between potentially important violations, which are induced by a large number of data samples, and insignificant violations resulted from noise. The evaluation of topological health of an SOM by the TF therefore can be inaccurate and misleading.

To overcome these drawbacks, we improved the TF into a suite of new measures. First, we proposed the differential version of the TF, the Differential Topographic Function (DTF), which clearly shows the extent of violation (average number of violations) at each folding length, fl (Section 3.3.1). Next, we normalized the DTF by the total number of connections, showing the percentage of connections at each folding length (Section 3.3.2). This enables comparison across SOMs that have different total number of connections. In a further step, we used the connection strength (an element of the connectivity matrix, or $CONN$ matrix) as an importance weighting on each violation so that strong violations could be distinguished from weak ones. This resulted in the Weighted Differential Topographic Function ($WDTF$), which shows the severity of violations (the percentage of data samples contributing to the violations) at a given folding length, fl (Section 3.3.3). The $WDTF$ offers a more elaborate and accurate view of the relative severities across all scopes of violations than the $NDTF$, and it was shown to be applicable to the comparison across SOMs of different sizes, SOM at different learning steps, different data sets.

We also designed and implemented an interactive tool, TopoView, which enables the visual inspection of violations on the SOM, regardless of data dimension. TopoView shows the locations and orientations of the violations, which constitute an additional piece of information to the summary view from the measures. Moreover, we implemented a series of selection and thresholding capabilities in TopoView, providing the freedom to investigate various (sub)sets of violations such that the user can pinpoint the problematic areas in the map, determine the cause of the violations, and make appropriate remediations.

We showed the effectiveness of the combined use of the $WDTF$ and TopoView through an artificial data set and two hyperspectral images. For the 2-dimensional 4-class Gaussian data set (Fig. 3.7), the evolution of the $WDTF$ and the corresponding TopoView visualization of the connections illustrated how the $WDTF$ and TopoView reveal topological problems (Section 3.5.1). For real data, perfect topology preservation is likely impossible because of noise or intricate structure, as demonstrated by the two hyperspectral data sets. In such cases, zero violation is a too strict criterion for a healthy map in this situation. Rather, the user should focus more on the evaluation of the severe violations (mostly long-ranged and/or strong violations), which could be detrimental to the correct identification of manifold structure. Both the $WDTF$ and TopoView are helpful in this sense. The $WDTF$ puts more emphasis on strong violations. TopoView helps filter out the benign violations and makes the harmful ones obvious according to the user's specifications. With the LCVF data set (Fig. 3.8), we validated that the $WDTF$ was more accurate and informative than the $NDTF$ through investigation of the detailed statistics of the connections (Section 3.5.2). With the RIT data set (Fig. 3.13), we showed the usefulness of TopoView in the comparison of two SOM clusterings (Section 3.5.3). Inter-cluster violations are considered warnings of possible incorrect clustering. It is noteworthy that, while using the quantity and/or the strengths of inter-cluster violations as criteria for the evaluation of a clustering, the user should be aware of the dependency of the inter-cluster violations on the clustering. An extreme case is that when all SOM prototypes are assigned the same cluster label, no inter-cluster violations will be shown by TopoView because there is only one cluster. No inter-cluster violations here does not mean the clustering is perfect. Therefore, the user should be very careful when making judgement whether a clustering is good or not. Since inter-cluster violations reflect the similarities across the clusters, perhaps the user should combine the use of cluster validity indices that reflect the similarities of sam-

ples or prototypes within the clusters, and the use of TopoView, for a more comprehensive evaluation of quality of clusterings.

We can envision a few possible directions for further development of tools for topology measurement.

First, it would be good to also provide a definition of the $WDTF$ for the negative domain ($fl < 0$). We have defined the positive domain of the $WDTF$ in the thesis, focusing on forward violations ($fl > 0$), since backward violations ($fl < 0$) are not as detrimental as forward violations for cluster identification. Nevertheless, a quantification of the severities of the backward violations would make the definition of the $WDTF$ more complete. To find a meaningful and useful counterpart of the connection strength, in the negative domain, however, is not intuitive and requires more research in future work.

Second, it might be useful to combine the thresholding capabilities of TopoView into the $WDTF$, to show the severity of "harmful" violations across different scopes of violations. The "harmful" set of violations can be specified by the user. This means to combine the $WDTF$ and TopoView into one evaluation tool. Once the user makes a selection of thresholds, TopoView filters out the violations considered unimportant and the $WDTF$ would also exclude those unimportant violations from its computation.

In addition to the two possible improvements of the tools, an intriguing future direction is to incorporate the measures into the SOM learning algorithm for online feedback so that the SOM can intelligently correct itself and achieve best outcome automatically. The motivation is obvious: since the most appropriate learning parameters for fast convergence to the best map depend on the unique properties of the data in each application, the determination of the learning parameters can require laborious parameter exploration by the user. It would be desirable for these parameters to be tuned automatically during the training. AdSOM, proposed by Kiviluoto [33], was a variant of the SOM with locally adapting

neighborhood widths. The adaptation is controlled by the Topographic Error (TE), which is a measure of topology violations. For the neurons near the problematic region of the map, where forward violations exist, the neighborhood width is increased automatically to untwist the map. However, AdSOM was tested only with a 3-dimensional artificial data set, so whether it would benefit the learning of high-dimensional complicated data is unknown. Moreover, AdSOM did not realize full automation because the user still needed to tune the learning rate, $\alpha(t)$ in eq. 2.2. Auto-SOM, a more advanced algorithm proposed by Haese and Goodhill [88], automated the modulation of both the learning rate and the neighborhood width by a Kalman filter implementation of the SOM with a recursive parameter estimation method. Auto-SOM incorporated the TF in the control of the neighborhood size. The authors demonstrated the effectiveness of Auto-SOM with both artificial and real data. Future work can be development of an SOM algorithm similar to Auto-SOM, which utilizes the $WDTF$ instead of the TF as a more precise feedback to the control of the parameters.

Inference of multiple latent variables by customized exploitation of the SOM's knowledge in a new supervised architecture

We approached the inference of latent variables with supervised learning aided by an SOM. The neural architecture we proposed is the Conjoined Twins, a new architecture motivated by a planetary science application where two physical parameters were inferred from Near-Infrared spectra of water ice.

The Conjoined Twins was developed from an SOM-hybrid neural architecture, where the output layer retrieved the SOM outputs and combined them into weighted sums to learn to approximate the latent variables from the input data vectors. The customary use of the SOM outputs by the output layer is the Winner-Takes-All (WTA) mode, where the best

matching unit (BMU) has an output of 1 while the other SOM neurons have outputs of 0. The WTA mode has a hard limit, N, of the number of values of latent variables it can differentiate, where N is the number of SOM neurons. For data sets with continuous latent variables, this limit can prevent high prediction accuracies. To break this limit, we generalized the WTA mode to the k-Winners-Take-All (kWTA) mode, which makes better use of the SOM's knowledge by allowing multiple, k, SOM winners to have nonzero outputs and to contribute to the supervised learning.

The best value of k is obviously dependent on the data properties and the SOM (the SOM size and the learning step). To help determine k for an SOM learned with a specific data set, we proposed a theoretical upper bound, K, of k, which can be computed automatically from the statistics of the Voronoi neighbors. After this, we proposed to perform an exhaustive search of best k by performing the supervised learning with all values of $k \leq K$. The best k is the value that produced the highest prediction accuracy. Although the exhaustive search sounds computationally expensive, the theoretical upper bound of k allows to constrain the search of k in a small range. This makes our method of finding k a feasible solution.

Through the inference of temperature and grain size from high-dimensional spectra of ices, we found an interesting dependency of the best k on the latent variables. For different latent variables, different numbers of SOM winners contain the right amount of information for the recovery of this variable. In this specific application, temperature and grain size can be inferred best with $k = 3$ and $k = 1$, respectively. This motivated the new architecture, the Conjoined Twins, which is similar to the SOM-hybrid architecture, but has two copies of the output layer ("twin heads"). By allowing the different "twin heads" to use different k for the supervised learning, the Conjoined Twins can achieve the highest prediction accuracies for both latent variables simultaneously. With this application, we

showed that the Conjoined Twins achieved high and scientifically useful accuracies for temperature and grain size. Through a noise sensitivity analysis and a comparison to a competing neural approach, the Backpropagation (BP) network, we also confirmed a high degree of robustness to noise and exceptional reliability of the models produced with the Conjoined Twins. These properties of the models give us confidence that the Conjoined Twins is an effective solution to the inference problems in planetary spectral applications.

In this thesis, we inferred two physical parameters from spectra of H_2O ice as an initial assessment of our new approach. Future work should include the investigation of the performance of the Conjoined Twins model under increasing levels of realism in the spectra such as the inference of more than two physical parameters. Although conceptually the Conjoined Twins can be extended to have more than two "heads", and the same procedure can be used for the learning of more than two latent variables, the increased number of latent variables may cause new difficulties in the neural modeling. More research work is required in future projects to investigate the new issues and to improve the Conjoined Twins.

Our efforts in this thesis advance the capability of correctly learning the structure of high-dimensional, complicated data, and accurately retrieving knowledge from these data, with the SOM. Although our innovations were motivated by applications to hyperspectral data of planetary surfaces, the new tools should be applicable for the analysis of other types of high-dimensional data, such as data collected through medical trials. New data of course, can hold unforseen challenges, which may necessitate revisions of the current tools and further innovations.

Notations

A	SOM lattice, 14
C	Total number of connections in the SOM, 50
D	Delaunay graph, 27
K	Upper bound of k in the kWTA mode, 87
L	Dimension of the vector of latent variables, $\mathbf{l}$, 79
M	Data manifold, 14
N	Number of neurons (or prototypes) in an SOM, 14
P	Total number of data samples, 51
RF_i	receptive field of neuron i, 14
V	Voronoi tessellation, 17
V_i	Voronoi cell of SOM prototype $\mathbf{w}_i$, 17
$\%data_i$	Percentage of data samples contributing to all ith ranking connections in the SOM, 87
$\mathbf{l} = [l_1, l_2, ..., l_L]^T$	Vector of latent variables, 79
$\mathbf{r}_i$	Lattice coordinates of neuron i in the SOM, 15
$\mathbf{w}_i$	Prototype (weight vector) of SOM neuron i, 14
$\mathbf{x} = [x_1, x_2, ..., x_d]^T$	Input data vector, 14
$\widetilde{D}$	Induced Delaunay graph, 27

$\widetilde{V}$	Induced Voronoi tessellation, 27
$\widetilde{V}_i$	Induced Voronoi cell of SOM prototype $\mathbf{w}_i$, 27
$\widetilde{m}$	Number of important Voronoi neighbors to any prototype, 87
c	Lattice index of the BMU, 14
d	Dimension of the input data manifold, 14, 79
fl	folding length, 43
$h_{c,j}(t)$	Neighborhood function used in the SOM algorithm, 15
k	Number of SOM winners used for supervised learning in the kWTA mode, 85
m	Maximum number of Voronoi neighbors to any prototype, 53, 87
n_i	Number of connections from all prototypes to their ith ranking Voronoi neighbors, 87
s_i	Average strength of all connections from prototypes to their ith ranking Voronoi neighbors, 87
y_i	Output of SOM neuron i, 82
$\mathbf{V}$	weight matrix connecting the output and the hidden layers of the SOM-hybrid neural machine, 81
$\mathbf{W}$	weight matrix connecting the hidden layer and the input buffer of the SOM-hybrid neural machine, 81

Index

Appendix A

Publications of the author

- L. Zhang and E. Merényi, "Learning multiple latent variables with Self-Organizing Maps", *Proc. 2010 IEEE International Conference on Granular Computing, Silicon Valley, CA, August 14-16*, 2010.

- L. Zhang, E. Merényi, W. M. Grundy, and E. F. Young, "Inference of surface parameters from Near-Infrared spectra of crystalline H_2O ice with neural learning", *Publications of the Astronomical Society of the Pacific*, 122:839–852, 2010 July.

- E. Merényi, K. Taşdemir, and L. Zhang, "Learning highly structured manifolds: harnessing the power of SOMs", Chapter in *Similarity based clustering*, Lecture Notes in Computer Science (Eds. M. Biehl, B. Hammer, M. Verleysen, T. Villmann), Springer-Verlag, LNAI 5400, 138–168, 2009.

- L. Zhang, E. Merényi, W. M. Grundy, and E. F. Young, "An SOM-hybrid supervised model for the prediction of underlying physical parameters from Near-Infrared planetary spectra", *Proc. 7th International Workshop on Self-Organizing Maps (WSOM 2009), Advances in Self-Organizing Maps, Jun 8–10, St. Augustine, FL*, Lecture Notes in Computer Science, Springer-Verlag, LNCS 5629, 362–371, 2009.

- E. Merényi, L. Zhang, and K. Taşdemir, “Min(d)ing the small details: discovery of critical knowledge through precision manifold learning and application to on-board decision support”, *Proc. IEEE Int’l Conference on Systems of Systems Engineering (IEEE SoSE 2007), April 16–18, San Antonio, TX*, 2007.

- L. Zhang and E. Merényi, “Weighted Differential Topographic Function: a refinement of Topographic Function”, *Proc. 14th European Symposium on Artificial Neural Networks (ESANN 2006) Bruges, Belgium, April 26–28*, 7–12, 2006.

Appendix B

Software implementation of the new tools proposed in the thesis

The author implemented the new tools proposed in this thesis in C/C++ on UNIX platform. The implementation includes the following four pieces of software:

- dtf, a module to compute a suite of measures of topology preservation
- TopoView, an interactive tool for visualization of selected sets of connections on the SOM
- CTwins, the Conjoined Twins supervised learning architecture
- Augmentation to a module CONNvis, for computation of Voronoi statistics used by CTwins

All four pieces of software were implemented in the environment developed and maintained by the Merényi group for SOM-based machine learning of complicated, high-dimensional data. Information about this environment and software capabilities is available at http://terra.ece.rice.edu. The input/output mechanisms, general structure, processing and housekeeping of these software take advantage of the standardized support services in this environment. A detailed user manual along with examples of use is provided with each individ-

ual module. Here we are giving an impression of the unique functions of dtf, TopoView, CTwins and the augmentation to CONNvis, and the controls the user can exercise.

B.1 dtf, a module to compute a suite of measures of topology preservation

The dtf module takes a data set and an SOM learned with it as inputs. The files that represent the learned SOM are products from any of the SOM-hybrid neural network modules (ann-SOMconsc, ann-SOMbdh, etc.) implemented by the Merényi group. The dtf module computes the TP, the TF, the DTF, the $NDTF$ and the $WDTF$ for the input SOM and writes these measures, together with some statistics of the connections, out to a file in a human readable format.

Example output file fragment:

```
## TP = -0.05115

## number of empty PEs = 41
## total number of connections = 1034
## total number of data points = 16384
## number of connections connecting non-empty PEs = 1034
## number of data points whose BMU and second BMU are non-empty = 16384
## TF = 0 for k>7, when all PEs are included.
## TF = 0 for k>7, when empty PEs are excluded.

#  k   norm_k    TF      TF      DTF      DTF      NDTF      NDTF      WDTF      WDTF
#                       (empty PEs       (empty PEs         (empty PEs          (empty PEs
#                        excluded)        excluded)          excluded)           excluded)
1      0.0714    5.04    5.04    0.00    0.00      0.0000    0.0000    0.0000    0.0000
2      0.1429    4.11    4.11    0.93    0.93      0.1015    0.3322    0.0115    0.0115
3      0.2143    3.07    3.07    1.04    1.04      0.1132    0.3701    0.0139    0.0139
4      0.2857    1.96    1.96    1.10    1.10      0.1199    0.3923    0.0211    0.0211
5      0.3571    0.83    0.83    1.14    1.14      0.1238    0.4049    0.0184    0.0184
6      0.4286    0.18    0.18    0.65    0.65      0.0706    0.2309    0.0103    0.0103
7      0.5000    0.02    0.02    0.16    0.16      0.0174    0.0569    0.0032    0.0032
8      0.5714    0.00    0.00    0.02    0.02      0.0019    0.0063    0.0009    0.0009
9      0.6429    0.00    0.00    0.00    0.00      0.0000    0.0000    0.0000    0.0000
...     ...             ...         ...         ...
```

The author has also provided Matlab scripts, which read in the output file and plot the measures as shown in Fig. 4.6 in Section 3.5.

B.2 TopoView, an interactive tool for visualization of selected sets of connections on the SOM

TopoView is a visualization tool for interactive selection and display of connections over the SOM. The inputs of TopoView include a data set and the SOM learned with it. Upon start, TopoView launches its graphical user interface as shown in Fig. B.1. The user can change the values of the keywords that control the selection and the thresholding of connections, in the keyword window (Fig. B.1, right). TopoView will refresh the connections drawn on the SOM in the display window (Fig. B.1, left) according to the user's choices.

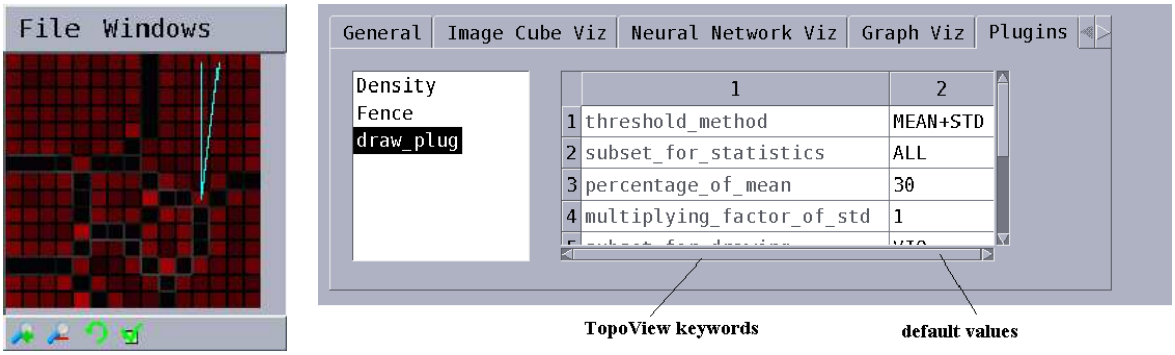


Figure B.1: The graphical user interface of TopoView. **Left**: The display window of TopoView. The selected subsets of connections will be shown on the SOM in this window. **Right**: The keyword window, in which the user can make selection and set thresholding of connections by specifying values for keywords that controls TopoView.

TopoView provides two main categories of keywords: keywords for selecting connections to be drawn and keywords for setting the drawing properties.

1. Keywords for selecting connections to be drawn:

   The user can select the connections by choosing a range for folding length, the type

of the connections, and a threshold for connection strength.

- The range for folding length is specified by an upper and a lower limit of folding length.
- The type of connections include all connections, violating connections, non-violating connections, inter-cluster connections, intra-cluster connections, inter-cluster violating connections and intra-cluster violating connections.
- The threshold for connection strength is computed automatically by TopoView from a statistical property of a specific set of connections. The user needs to indicate which set of connections, and what statistical property of this set of connections, are to be used as the threshold. The statistical property can be the mean connection strength multiplied by a constant a ($mean_{conn\_stren} \times a$) or the mean connection strength plus the standard deviation of the connection strengths multiplied by a constant b ($mean_{conn\_stren} + std_{conn\_stren} \times b$).

2. Keywords for setting the drawing properties:

   The user can modulate the following two properties of the lines drawn on the SOM, to ensure best visual clarity.

   - the line color of the connections
   - the line width of the connections

TopoView also computes the statistical properties (the number of connections, the mean connection strength and the standard deviation of connection strength) for different sets of connections and show them in the terminal window to help the user decide the thresholding method to use.

Example statistics of connections shown in the terminal window:

```
========= Statistics of Connections Subsets (dead PEs excluded) ==========
All #=1034  mean=15.845261  std=25.930155
Violating #=567   mean=2.292769   std=2.387171
non-violating #=467   mean=32.299786  std=38.516922
inter-cluster #=0     mean=n/a        std=n/a
intra-cluster #=1034  mean=15.845261  std=25.917613
inter-cluster violating #=0     mean=n/a        std=n/a
intra-cluster violating #=567   mean=2.292769   std=2.385065
============================================================
THRESHOLD = 41.775416
```

In addition, TopoView allows the choice of displaying clusters under the drawn connections. This makes it easy to visually separate inter-cluster and intra-cluster violations in the SOM, and therefore helps with the evaluation of clusterings.

B.3 CTwins, the Conjoined Twins supervised learning architecture

Before using the CTwins module for the supervised learning of multiple latent variables, the user needs to first perform the unsupervised learning of the data with any of the SOM learning modules (ann-SOMconsc, ann-SOMbdh, etc.) implemented by the Merényi group. Then the user runs another module, CONNvis, for computation of the statistics of Voronoi neighbors and determination of the theoretical upper bound, K, of k, as will be described in Section B.4. After the above two steps, the user can launch CTwins, which takes the learned SOM and K as its inputs. CTwins performs the supervised learning repeatedly with $k \leq K$ and finds the best value of k for the learning of each latent variable. The author implemented the repeated supervised runs with different k in CTwins by reusing the code for supervised learning in ann-SOMconsc and building an elaborate wrapper around it. After completing the supervised learning, CTwins saves all networks resulting from the supervised learning with different values of k, and generates a report file listing the best k for each latent variable and prediction accuracies it has achieved.

Example report file fragment:

(Temperature and Particle_size are the two latent variables learned in this example.)

```
...     ...             ...         ...         ...
# Parameter    Best_k  Test_accu(%)  Network
Temperature      3      80.5         ices-8class-v5-ss10.150000.H2Ojk10-tr3.k3.1000000.nnd
Particle_size    1     100.0         ices-8class-v5-ss10.150000.H2Ojk10-tr3.k1.1000000.nnd

# K = 4 (K is the upper limit of k, computed by Voronoi statistics in .vstat or given by user.)
# All training results produced with k =   1  2  3  4
# (Parameter names correspond to output neurons 1, 2, etc.)
#   k   Parameter        Test_accu(%)  Train_accu(%)  Test_RMSE  Train_RMSE
#   1
#       Temperature       74.3          80.2           5.2204     4.6388
#       Particle_size    100.0         100.0           0.0000     0.0000
#   2
#       Temperature       76.1          86.3           4.8691     4.2319
#       Particle_size     93.8          89.9           0.0048     0.0065
#   3
#       Temperature       80.5          85.6           5.2546     4.6256
#       Particle_size     84.1          78.9           0.0066     0.0058
#   4
#       Temperature       80.5          86.6           5.8512     5.1000
#       Particle_size     74.3          71.9           0.0110     0.0062
...     ...             ...         ...         ...
```

When the learned model is deployed for the inference of the latent variables from new data (data not used for the training and validation of the model), CTwins takes the list of best values of k in the above report file and the new data as inputs, and computes the prediction accuracy for each latent variable with its best value of k indicated in the report file, for the new data.

B.4 Augmentation to CONNvis, for computation of Voronoi statistics used by CTwins

This piece of software is an augmentation to the module CONNvis, which was implemented by Kadim Tasdemir and explained in [24]. The Voronoi statistics, required by CTwins for determination of K, the upper bound of k, are derivatives of the output of CONNvis. The author augmented the CONNvis module to compute and output the de-

tailed Voronoi statistics, as well as a suggested value of K, to a file. The CTwins module will read in the value of K from this output file.

Example output file:

```
# suggested_K_for_Conjoined_Twins = 3
# suggested_cut_rank = 2
# suggestion based on thresholding: %data>1%
#
# total_number_data_samples = 1134
# total_number_connections = 200
# mean_strength_all_connections =   5.67
# max_number_neighbors = 3
#
# norm_mean_stren = mean_stren/mean_strength_all_connections
# %data = mean_stren*nr_conn/(2*total_number_data_samples)*100%
# rank  nr_conn  mean_stren norm_mean_stren %data
   1       209    7.56            1.33      69.71
   2       184    3.69            0.65      29.94
   3         7    1.14            0.20       0.35
```

Appendix C

Backpropagation (BP) neural network

Feedforward neural networks trained with backpropagation (BP) method, or BP networks, are widely used for pattern recognition and function regression [55, 56, 57]. Since a 2-layer BP network with a nonlinear transfer function is capable of approximating any continuous function [56, 57], it can be used for the planetary science problem targeted in this thesis work (in Section 4.3). In Section 4.3.7, we compare the results from a 2-layer BP network with those from the Conjoined Twins, proposed in this thesis.

As shown in Fig C.1, the 2-layer BP network (BP network with 2 weight layers) takes a d-dimensional input vector $\mathbf{x}$, $[x_1, x_2, ..., x_d]$, by its input buffer, in each learning step. The network has M neurons in its hidden layer. Each neuron j in the hidden layer combines the inputs and a bias $x_0(=1)$ into a weighted sum and generates an output, x_j', as

$$x_j^{'} = f(\sum_{i=0}^{d} w_{ji}x_i) \quad j = 1, 2, ..., M \tag{C.1}$$

The bias term x_0 is analogous to the intercept term in a regression equation. w_{ji} is the weight between the neuron j in the hidden layer and the neuron i in the input buffer. f is a

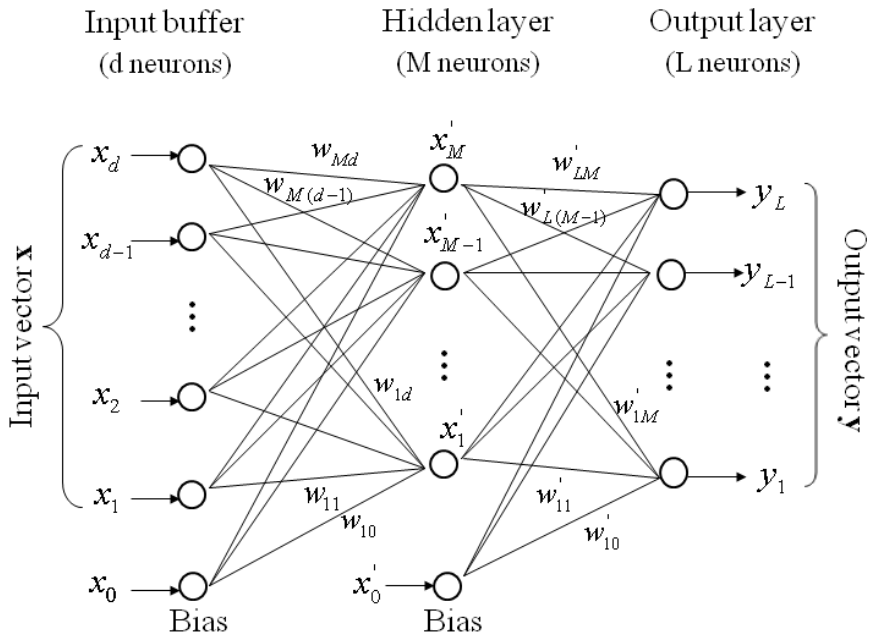


Figure C.1: A 2-layer backpropagation (BP) neural network.

transfer function, a typical choice of which can be

$$f(x) = \frac{1 - e^{-x}}{1 + e^{-x}} \tag{C.2}$$

Similarly, each neuron l in the output layer yields an output y_l with the signals from the hidden layer.

$$y_l = f(\sum_{j=0}^{M} w'_{lj} x'_j) \quad l = 1, 2, ..., L \tag{C.3}$$

w'_{lj} is the weight between the neuron l in the output layer and the neuron j in the hidden layer.

The initial weights are often chosen as small random numbers. The training of the network can be done either in online or batch mode. In online learning the weights of the network are updated every time an input vector, $\mathbf{x}^q$, has gone through the network

(eqs. C.1–C.3) and the error in the output layer, E^q, is computed.

$$E^q = \frac{1}{2}\sum_{l=1}^{L}(t_l^q - y_l^q)^2 \tag{C.4}$$

where t_l^q and y_l^q are the target (or desired) output and the actual output, respectively, from the l-th neuron in the output layer. In batch learning the network accumulates the errors in the output layer for an epoch of Q input vectors and then updates the weights. When the epoch size, Q, is 1, batch learning is equivalent to online learning. The total squared error for an epoch of Q input vectors, $\mathbf{x}^1, \mathbf{x}^2, ..., \mathbf{x}^Q$, is

$$E_{total} = \sum_{q=1}^{Q} E^q = \frac{1}{2}\sum_{l=1}^{L}\sum_{q=1}^{Q}(t_l^q - y_l^q)^2 \tag{C.5}$$

The gradient descent method is used to minimize E_{total}. The weights are modified after each epoch as

$$\begin{cases} w_{lj}^{'(new)} = w_{lj}^{'(old)} - \alpha\dfrac{\partial E_{total}}{\partial w_{lj}^{'}} \\ w_{ji}^{(new)} = w_{ji}^{(old)} - \eta\dfrac{\partial E_{total}}{\partial w_{ji}} \end{cases} \tag{C.6}$$

where α and η are learning rates, which decrease with time. By inserting eqs. C.1–C.3 into eq. C.5, we can rewrite the gradients in eq. C.6 as

$$\begin{cases} \dfrac{\partial E_{total}}{\partial w_{lj}^{'}} = -\sum_{q=1}^{Q}\delta_l^q x_k^{'q} \\ \dfrac{\partial E_{total}}{\partial w_{ji}} = -\sum_{q=1}^{Q}\delta_j^{'q} x_i^q \end{cases} \tag{C.7}$$

where

$$\begin{cases} \delta_l^q = (t_l^q - y_l^q)(1 - (y_l^q)^2)/2 \\ \delta_j^{'q} = \sum_{l=1}^{L}\delta_l^q w_{lj}^{'}(1 - (x_j^{'q})^2)/2 \end{cases} \tag{C.8}$$

The errors in the hidden layer are computed by propagating the errors backward from the output layer and distributing according to the second equation in eq. C.8. This is why the algorithm is called backpropagation. A simple stopping criterion for the training can be the completion of a certain number of epochs or the achievement of a specified small total error. However, these cannot guarantee that the network has converged (learned long enough) and that it has not overfitted the training data (learned too long). A better approach for quality control is to use a test set. The performance of the network on both the training and the test data should be monitored during the training. The stopping time should be the time point when the performance on the test data stops improving and begins declining.

Bibliography

[1] J. Campbell. *Introduction to remote sensing*. The Guilford Press, 1996.

[2] R. E. Bellman. *Adaptive Control Processes*. Princeton University Press, Princeton, NJ, 1961.

[3] E. Merényi. Precision mining of high-dimensional patterns with Self-Organizing Maps: Interpretation of hyperspectral images. In *Quo Vadis Computational Intelligence: New Trends and Appraoaches in Computational Intelligence. Studies in Fuzziness and Soft Computing*, volume 54. Physica-Verlag, 2000.

[4] E. Merényi, K. Taşdemir, and L. Zhang. Learning highly structured manifolds: harnessing the power of SOMs. In M. Verleysen T. Villmann M. Biehl, B. Hammer, editor, *Similarity based clustering*, LNAI 5400, pages 138–168. Springer-Verlag, 2009.

[5] T. Kohonen. *Self-Organizing Maps*. Springer-Verlag, Berlin Heidelberg New York, Third edition, 2001.

[6] R. M. Gray. Vector quantization. *IEEE Acoustics, Speech and Signal Processing Magazine*, 1:4–29, 1984.

[7] A. Gersho and R. M. Gray. *Vector quantization and signal processing*. Kluwer Academic Publlisher, Boston, 1992.

[8] C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[9] T. Martinetz, S. Berkovich, and K. Schulten. Neural gas network for vector quantization and its application to time-series prediction. *IEEE Trans. Neural Networks*, 4(4):558–569, 1993.

[10] G. G. Blasdel and G. Salama. Voltage-sensitive dyes reveal a modular organization in monkey striate cortex. *Nature*, 321:579–585, 1986.

[11] D. H. Hubel and T. N. Wiesel. Sequence regularity and geometry of orientation columns in the monkey striate cortex. *Journal of Comparative Neurology*, 158:267–294, 1974.

[12] J. H. Kaas, R. J. Nelson, M. Sur, C. S. Lin, and M. M. Merzenich. Multiple representations of the body within the primary somatosensory cortex of primates. *Science*, 204:521–523, 1979.

[13] N. Suga and W. E. O'Neill. Neural axis representing target range in the auditory cortex of the mustache bat. *Science*, 206:351–353, 1979.

[14] C. von der Malsburg. Self-organization of orientation sensitive cells in the striate cortex. *Kybernetik*, 14:85–100, 1973.

[15] D. J. Willshaw and C. von der Malsburg. How patterned neural connections can be set up by self-organization. *Proc. roy. Soc. Lond. B*, 194:431–445, 1976.

[16] C. von der Malsburg and D. J. Willshaw. How to label nerve cells so that they can interconnect in an ordered fashion. *Proc. Natl. Acad. Sci. U.S.A.*, 74:5176–5178, 1977.

[17] S. Kaski, J. Kangas, and T. Kohonen. Bibliography of Self-Organizing Map (SOM) papers: 1981-1997. *Neural Computing Surveys*, 1:102–350, 1998.

[18] M. Oja, S. Kaski, and T. Kohonen. Bibliography of Self-Organizing Map (SOM) papers: 1998-2001 addendum. *Neural Computing Surveys*, 3:1–156, 2003.

[19] T. Villmann, R. Der, M. Herrmann, and T. M. Martinetz. Topology preservation in self-organizing feature maps: exact definition and measurement. *IEEE Trans. Neural Networks*, 8:256–266, 1997.

[20] T. Martinetz and K. Schulten. Topology representing networks. *Neural Networks*, 7:507–522, 1994.

[21] E. S. Howell, E. Merényi, and L. A. Lebofsky. Classification of asteroid spectra using a neural network. *J. Geophys. Res.*, 99(E5):10,847–10,865, 1994.

[22] L. Rudd and E. Merényi. Assessing debris-flow potential by using AVIRIS imagery to map surface materials and stratigraphy in cataract canyon, Utah. In R.O. Green, editor, *Proc. 14th AVIRIS Earth Science and Applications Workshop*, Pasadena, CA, 2005.

[23] D. DeSieno. Adding a conscience to competitive learning. In *IEEE Int'l Conference on Neural Networks*, volume 1, pages 117–124, 1988.

[24] K. Taşdemir and E. Merényi. Exploiting the data topology in visualizing and clustering of Self-Organizing Maps. *IEEE Trans. Neural Networks*, 20(4):549–562, 2009.

[25] A. Ultsch and H. P. Siemon. Kohonen's self organizing feature maps for exploratory data analysis. In *Intl. Neural Network Conf. (Proc. INNC'90)*, pages 305–308, Dordrecht, Netherlands, 1990.

[26] M. A. Kraaijveld, J. Mao, and A. K. Jain. A nonlinear projection method based on Kohonens topology preserving maps. *IEEE Trans. Neural Networks*, 6(3):548–559, 1995.

[27] E. Merényi, A. Jain, and T. Villmann. Explicit magnification control of Self-Organizing Maps for "forbidden" data. *IEEE Trans. Neural Networks*, 18(3):786–797, May 2007.

[28] E. Merényi, E. S. Howell, A. S. Rivkin, and L. A. Lebofsky. Prediction of water in asteroids from spectral data shortward of 3 microns. *Icarus*, 129:421–439, 1997.

[29] D. O. Hebb. *The organization of behavior*. New York: Wiley, 1949.

[30] K. Taşdemir and E. Merényi. Considering topology in the clustering of Self-Organizing Maps. In *Proc. 5th Workshop on Self-Organizing Maps (WSOM 2005)*, pages 439–446, 2005.

[31] J. Vesanto and E. Alhoniemi. Clustering of the Self-Organizing Map. *IEEE Trans. Neural Networks*, 11:586–600, 2000.

[32] T. Villmann. Benefits and limits of the Self-Organizing Map and its variants in the area of satellite remote sensoring processing. In *Proc. ESANN'99 European Symposium on Artificial Neural Networks*, pages 111–116, 1999.

[33] K. Kiviluoto. Topology preservation in Self-Organizing Maps. In *Proceedings IEEE International Conference on Neural Networks*, pages 294–299, Bruges, 1996.

[34] H.-U. Bauer, M. Hermann, and T. Villmann. Neural maps and topographic vector quantization. *Neural Networks*, 12:659–676, 1999.

[35] A. Flexer. On the use of Self-Organizing Maps for clustering and visualization. *Intelligent Data Analysis*, 5:373–384, 2001.

[36] T. F. Cox and M. A.A. Cox. *Multidimensional scaling*. Chapman and Hall/CRC, 2001.

[37] J. W. Sammon Jr. A nonlinear mapping for data structure analysis. *IEEE Trans. Computers*, C-18(5):401–409, May 1969.

[38] J. C. Bezdek and N. R. Pal. An index of topological preservation for feature extraction. *Pattern Recognition*, 28(3):381–391, 1995.

[39] H.-U. Bauer and K. Pawelzik. Quantifying the neighborhood preservation of self-organizing feature maps. *IEEE Trans. Neural Networks*, 3:570–579, 1992.

[40] A. König. Interactive visualization and analysis of hierarchical neural projections for data mining. *IEEE Trans. Neural Networks*, 11(3):570–579, 2000.

[41] D. Vidaurre and J. Muruzábal. A quick assessment of topology preservation for SOM structures. *IEEE Trans. Neural Networks*, 18(5):1524–1528, 2007.

[42] S. Zrehen. Analyzing Kohonen maps with geometry. In St. Gielen and B. Kappen, editors, *Proc. ICANN'93 International Conference on Artificial Neural Networks*, pages 609–612, 1993.

[43] M. C. Su, H. T Chang, and C. H. Chou. A novel measure for quantifying the topology preservatin of Self-Organizing Feature Maps. *Neural Processing letters*, 15:137–145, 2002.

[44] E. De Bodt, M. Cottrell, and M. Verleysen. Statistical tools to assess the reliability of Self-Organizing Maps. *Neural Networks*, 15:967–978, 2002.

[45] S. Kaski and K. Lagus. Comparing Self-Organizing Maps. In C. Von der Malsburg and B. Sendhoff, editors, *Lecture Notes in Computer Science*, LNCS 1112, pages 809–814, 1996.

[46] L. Zhang and E. Merényi. Weighted Differential Topographic Function: A refinement of Topographic Function. In *Proc. ESANN 06'*, pages 13–18, Bruges, 2006.

[47] T. Villmann, E. Merényi, and B. Hammer. Neural maps in remote sensing image analysis. *Neural Networks, Special Issue on Self-Organizing Maps for Analysis of Complex Scientific Data*, 3–4(16):389–403, 2003.

[48] R. O. Green, J. E. Conel, J. Margolis, C. Chovit, and J. Faust. Inflight calibration and validation for Airborne the Visible/lnfrared Imaging Spectrometer (AVIRIS). In *Proc. 6th Annual Airborne Earth Science Workshop, JPL Pub 96-4*, 1996.

[49] R. O. Green and J. Boardman. Exploration of the ralationship between information content and signal-to-noise ratio and spatial rasolution. In *Proc. 9th AVIRIS Earth Science and Applications Workshop, Pasadena, CA*, 2000.

[50] W. H. Farrand. *VIS/NIR Reflectance Spectroscopy of Tuff Rings and Tuff Cones*. PhD thesis, University of Arizona, 1991.

[51] R. E. Arvidson, M. D.-B, and et al. Archiving and distribution of geologic remote sensing field experiment data. *EOS, Transactions of the American Geophysical Union*, 17(72):176, 1991.

[52] E. Merényi, K. Taşdemir, and W. Farrand. Intelligent information extraction to aid science decision making in autonomous space exploration. In W. Fink, editor, *Proc. DSS08 SPIE Defense and Security Symposium, Space Exploration Technologies*, volume 6960, page 69600M, 2008.

[53] J. R. Schott, S. D. Brown, R. V. Raqueno, H. N. Gross, and G. Robinson. An advanced synthetic image generation model and its application to multi/hyperspectral algorithm development. *Canadian Journal of Remote Sensing*, 25(2), June 1999.

[54] E. J. Ientilucci and S. D. Brown. Advances in wide-area hyperspectral image simulation. In *Proceedings of SPIE*, volume 5075, pages 110–121, May 5–8 2003.

[55] P. Werbos. PhD thesis, Harvard University, Cambridge, MA, 1974.

[56] D. E. Rumelhart, G. E. Hinton, and R. J. Williams. *Parallel Distributed Processing: Explorations in the Microstructure of Cognition, Vol. 1: Foundataions*, pages 318–362. MIT Press, Cambridge, 1986.

[57] K. Hornik. Approximation capabilities of multilayer feedforward networks. *Neural Networks*, 4:251–257, 1991.

[58] J. B. Tenenbaum, V. de Silva, and J. Langford. A global geometric framework for nonlinear dimensionality reduction. *Science*, 290(5500):2319–2323, 2000.

[59] S. Roweis and L. Soul. Nonlinear dimensionality reduction by locally linear embedding. *Science*, 290(5500):2323–2326, 2000.

[60] J. Lee and M. Verleysen. *Nonlinear Dimension Reduction*. Information Science and Statisics. Springer, 2007.

[61] D. L. Donoho and C. Grimes. Hessian eigenmaps: new locally linear embedding techniques for high-dimensional data. volume 100, pages 5591–5596, 2003.

[62] T. Kohonen. Description of input patterns by linear mixtures of SOM models. Technical Report E8, Helsinki University of Technology, 2007.

[63] B. Widrow and F. W. Smith. Pattern-recognizing control systems. In J. T. Tou and R. H. Wilcox, editors, *Computer and Information Sciences (COINS) Symposium Proceedings*, pages 288–317. Spartan Books, 1964.

[64] NeuralWare Inc. *Neural Computing – A Technology Handbook for Professional II/PLUS and NeuralWorks Explorer*. NeuralWare Inc., Pittsburgh, 2003.

[65] L. Zhang and E. Merényi. Learning multiple latent variables with Self-Organizing Maps. In *Proc. 2010 IEEE International Conference on Granular Computing, Silicon Valley, CA, August 14-16*, 2010.

[66] W. M. Grundy and B. Schmitt. The temperature-dependent Near-Infrared absorption spectrum of hexagonal H_2O ice. *J. Geophys. Res.*, 103:25809–25822, 1998.

[67] W. M. Grundy, B. Schmitt, and E. Quirico. The temperature dependent spectrum of methane ice I between 0.65 and 5 microns and opportunities for Near-Infrared remote thermometry. *Icarus*, 155:486–496, 2002.

[68] L. A. Young, S. A. Stern, H. A. Weaver, F. Bagenal, R. P. Binzel, B. Buratti, A. F. Cheng, D. Cruikshank, G. R. Gladstone, W. M. Grundy, D. P. Hinson, M. Horanyi, D. E. Jennings, I. R. Linscott, D. J. McComas, W. B. McKinnon, R. McNutt, J. M. Moore, S. Murchie, C. C. Porco, H. Reitsema, D. C. Reuter, J. R. Spencer, D. C. Slater, D. Strobel, M. E. Summers, and G. L. Tyler. New Horizons: Anticipated scientific investigations at the Pluto system. *Space Science Reviews*, 140:93–127, 2008.

[69] D. C. Reuter, S. A. Stern, J. Scherrer, D. E. Jennings, J. Baer, J. Hanley, L. Hardaway, A. Lunsford, S. McMuldroch, J. Moore, C. Olkin, R. Parizek, H. Reitsma, D. Sabatke, J. Spencer, J. Stone, H. Throop, J. V. Cleve, G. E. Weigle, and L. A.Young. Ralph:

A Visible/Infrared Imager for the New Horizons Pluto/Kuiper Belt Mission. *Space Science Reviews*, 140:129–154, 2008.

[70] L. Zhang, E. Merényi, W. M. Grundy, and E. F. Young. Inference of surface parameters from Near-Infrared spectra of crystalline H_2O ice with neural learning. *Publications of the Astronomical Society of the Pacific*, 122:839–852, July 2010.

[71] W. M. Grundy, M. W. Buie, J. A. Stansberry, and J. R. Spencer. Near-Infrared spectra of icy outer solar system surfaces: remote determination of H_2O ice temperatures. *Icarus*, 142:536–549, 1999.

[72] B. Hapke. Bidirectional reflectance spectroscopy: 1. theory. *J. Geophys. Res.*, 86:3039–3054, 1981.

[73] B. Hapke. *Theory of Reflectance and Emittance Spectroscopy*. Cambridge University Press, New York, 1993.

[74] D. P. Cruikshank, R. H. Brown, W. M. Calvin, T. L. Roush, and M. J. Bartholomew. Ices on the satellites of Jupiter, Saturn, and Uranus. In C. De Bergh B. Schmitt and M. Festou, editors, *Solar System Ices*, pages 579–606. Kluwer Academic Publishers, Dordrecht, 1998.

[75] D. S. Kimes, Y. Knyazikhin, J. L. Privette, A. A. Abuelgasim, and F. Gao. Inversion methods for physically-based models. *Remote Sensing Reviews*, 18:381–439, 2000.

[76] A. Pragnère, F. Baret, M. Weiss, R. Myneni, Y. Knyazikhin, and L. B. Wang. Comparison of three radiative transfer model inversion techniques to estimate canopy biophysical variables from remote sensing data. In T. I. Stein, editor, *IEEE 1999 International Geoscience and Remote Sensing Symposium 1999 (IGARSS'99) Proceedings*, volume 2, pages 1093–1095, 1999.

[77] S. S. Durbha, R. L. King, and N. H. Younan. Support vector machines regression for retrieval of leaf area index from multiangle imaging spectroradiometer. *Remost Sensing of Envioronment*, 107:348–361, 2007.

[78] C. Bernard-Michel, L. Gardes S. Douté, and S. Girard. Inverting hyperspectral images with Gaussian Regularized Sliced Inverse Regression. In *Proc. ESANN'2008, European Symposium on Artificial Neural Networks*, pages 463–468, 2008.

[79] C. Bernard-Michel, S. Douté, M. Fauvel, L. Gardes, and S. Girard. Retrieval of Mars surface physical properties from OMEGA hyperspectral images using regularized sliced inversion regression. *J. Geophys. Res.*, 114:E06005, 2009.

[80] U. Fink and H. P. Larson. Temperature dependence of the water-ice spectrum between 1 and 4 microns: Applications to Europa, Ganymede and Saturn's rings. *Icarus*, 24:411–420, 1975.

[81] M. E. Brown and W. M. Calvin. Evidence for crystalline water and ammonia ices on Pluto's satallite Charon. *Science*, 287:107–109, 2000.

[82] M. S. Gilmore, M. D. Merrilla, R. Casta, B. Bornsteinb, and J. P. Greenwood. Effect of Mars analogue dust deposition on the automated detection of calcite in Visible/Near-Infrared spectra. *Icarus*, 172:641–646, 2004.

[83] J. Ramsey, P. Gazis, T. Roush, P. Spirtes, and C. Glymour. Automated remote sensing with Near Infrared reflectance spectra: Carbonate recognition. *Data Min. Knowl. Discov.*, 6(3):277–293, 2002.

[84] W. M. Grundy. *Methane and Nitrogen Ices on Pluto and Triton: a Combined Laboratory and Telescope Investigation*. PhD thesis, University of Arizona, 1995.

[85] L. Zhang, E. Merényi, W. M. Grundy, and E. F. Young. An SOM-hybrid supervised model for the prediction of underlying physical parameters from Near-Infrared planetary spectra. In J.C. Príncipe and R. Miikkulainen, editors, *Proc. 7th WSOM Advances in Self-Organizing Maps*, LNCS 5629, pages 362–371, St. Augustine, FL, 2009.

[86] S. Mukherjee, U. Bhattacharya, S. K. Parui, R. Gupta, and R. K. Gulati. A multilayered backpropagation artificial neural network algorithm for UV spectral classification. *Astrophysics and Space Science*, 239(2):361–373, 1996.

[87] H. Du, W. Mei, and L. Shark. Classification of multispectral remote sensing image using an improved backpropagation neural network. In L. Zhou and C. Li, editors, *Proc. SPIE Electronic Imaging and Multimedia Systems II*, volume 3561, pages 403–408, 1998.

[88] K. Haese and G. J. Goodhill. Auto-SOM: Recursive parameter estimation for guidance of Self Organizing Feature Maps. *Neural Computation*, 13:595–619, 2001.