UNIVERSITÀ DEGLI STUDI DI UDINE

DIPARTIMENTO di
Ingegneria Elettrica, Gestionale e Meccanica

CORSO DI DOTTORATO DI RICERCA in
Ingegneria Industriale e dell'Informazione

CICLO XXV

TESI DI DOTTORATO DI RICERCA

Unsupervised spectral pattern recognition by Self-Organizing Maps

Relatore
Dott. Ing. Roberto Carniel

Dottorando
Luca Barbui

Anno Accademico 2013/2014

Contents

Abstract 1

I Theory 3

1 Self Organizing Maps 5

1.1 SOM architecture 6

1.2 SOM algorithm 8

1.2.1 Code vectors initialisation 12

1.2.2 Competition process 12

1.2.3 Cooperation process 13

1.2.4 Adaptive process 15

1.3 Quality map measures 18

1.3.1 Average quantisation error 18

1.3.2 Weighted distortion measure 18

1.3.3 Topographic error 19

1.4 Map visualisation 20

1.5 Correlation as discriminant function 23

1.6 Cluster recognition 26

1.7 Toroidal maps 27

II Applications 29

2 Volcanic regime detection 31

2.1 Procedure 31

2.2 Raoul Island 36

2.2.1 A SOM for Raoul 38

2.2.2 Raoul data projection and interpretation 41

2.3 Ruapehu 46
2.3.1 A SOM for Ruapehu 47
2.3.1.1 The 4 October 2006 event 49
2.3.1.2 The 27 September 2007 eruption 50
2.3.2 Ruapehu data clustering and interpretation 50
2.3.3 An alternative SOM analysis for Ruapehu 56
3 HVSR tecnique improvement by SOM 59
3.1 The environmental tremor 60
3.2 HVSR method 61
3.2.1 The quarter wave law 64
3.3 A SOM analysis for Salta city 66
3.3.1 Stations 5 & 6 71
3.3.2 Station 7 76
3.3.3 Station 8 77
3.3.4 Station 9 78
3.3.5 Station 3 80
3.3.6 Station 4 81
Appendix 91
A R software script 91
A.1 Time Series and Spectrograms Plotting 92
A.2 Plotting Splitted Time Series 95
A.3 Averaged Spectrum 96
A.4 HVSR (Horizontal to Vertical Spectral Ratio) 98
A.5 SOM map 101
A.6 Umatrix 105
A.7 Plotting the dendrogram and choosing the dissimilarity threshold 108
A.8 Plotting the clustering U-matrix 109
A.9 Cluster 109
A.10 SPCL 111

List of Figures

1.1 Illustration of a SOM neural network. The SOM projects the information of an n-dimensional feature space into a bi-dimensional grid of neurons, whereby the dimension are reduced. After training and cluster recognition SOM map should be able to identify an input activating just the corresponding area onto the map in function of input's features. The Kohonen layer tipically consists of a localised region of active neurons against the quiet background (loosely based on Klose [2006], fig. 1*a*). 7

1.2 Network size and topology type of a SOM are chosen before training begins. Notice overall shape difference for SOMs with identical size (side length equal to 5 in both cases), but different topology type. 9

1.3 The effect of the neighborhood function in the SOM algorithm. Starting from a perfect arrangement of the weights of a square grid (full lines), the weights nearest to the current input (indicated with the cross) receive the largest updates, those further away smaller updates, resulting in the updated grid (dashed lines). 10

1.4 Plot of the neighborhood function built according to the Gaussian function defined by equation 1.3 (*1.4a* and *1.4b*). Black dots placed along the x-axis (1.4b) shows how the influence of the BMU on neurons in its neighborhood decreases away, on the map, from the same BMU. The value of the neighborhood radius decreases with the increase of the number of iterations t (1.4c), so the width of the Gaussian function decreases more and more towards the center, reducing the number of neurons that constitute the neighborhood of the BMU. At the limit, in the final iterations of the SOM process, only the BMU is active on the map. 14

1.5 Graphical representation of α, as a function of the the number of iterations t. In fig. *1.5a* the diagram of function 1.7 is drawn for t equals to 100 iterations, $\alpha_0 = 0.5$ for τ between $\tau = 20$ and $\tau = 40$, increasing the value of τ in four steps. In fig. *1.5b* the diagram of function 1.8 is drawn for t equals to 100 iterations, $\alpha_0 = 0.5$ for τ between $\tau = 5$ and $\tau = 20$, increasing the value of τ in three steps. Any iteration with $\alpha = 0$ has no sense, since the network does not improve its organisation when the learning rate is equal to zero. 15

1.6 Graphical representation of a generic input vector $\mathbf{x}_i(t)$ and the corresponding BMU's code vector $\mathbf{w}_j(t)$. The updated code vector $\mathbf{w}_j(t+1)$ has been calculated using the eq. 1.6 (1.6a). In fig. 1.6b $\mathbf{x}_i(t)$ and $\mathbf{w}_j(t)$ have been normalised with respect to the sum of their own values. Fig. 1.6c proposes the comparison between the code vector $\mathbf{w}_j(t+1)$ in fig. *1.6a* and the same code vector drawn in fig. 1.6b. 16

1.7 Staying in the tri-dimensional input space, figure shows the position of the data (red crosses) and the position of the SOM map's nodes (blue circles) at the initialisation step (1.7a) and for the trained map (1.7b). 17

1.8 Figure shows an example of true neighbors (C and D) and a couple of false neighbors (A and B) projecting the two-dimensional space in a mono-dimensional space. 20

1.9 In fig. *1.9a* is shown the "classic" U-matrix, gray tone of each cell that contains a neuron is in reliance on the average dissimilarity from the neuron's code vector to the code vectors of the neighboring neurons. The more the dissimilarity between code vectors is higher, the more is dark the gray tone of the cell. In fig. 1.9b is shown the entire U-matrix using the improves suggested in this work, for the same example considered in fig. 1.9a. Each cell coloured in gray tone shows the dissimilarity level between a couple of neurons, the cells placed on the diagonals show the average dissimilarity of the two couples of neurons. 22

1.10 In fig. 1.10a the 12 edges solid is shown. Each one edge is described by its three direction cosins, the dataset is built repeating 50 times each one of the 12 vectors of length three. In fig. 1.10b the SOM map and the obtained dataset clustering are shown by the improved U-matrix representation. . . . 24

1.11 Toroidal SOM map visualisation. This kind of topology aims to remove the edges of the map, since neurons along edges of flat maps do not have the same number of neighbors as the others, resulting in an inhomogeneous training process. This is done by virtually sticking together upper and lower edges vertically and left and right edges horizontally, without any data modification. 27

2.1 Flow diagram of the proposed methodology for the data analysis. 34

2.2 Fig. 2.2a shows the location map of Raoul Island, showing position of seismic station RAO and major volcanic features. Fig. 2.2b shows a photo of the Green Lake located in the center of Raoul Island, it has been taken from the website *http://www.geonet.org.nz/*. 36

2.3 Original Helicorder recording acquired at seismic station RAO on Raoul Island, showing the pulsating eruption seismic signal on March 16, 2006 UT. Each line represents 6 minutes of data. The earthquake seen at the very end of the Helicorder recording is a tectonic event. 38

2.4 Spectrogram of the seismic data recorded at seismic station RAO on Raoul Island from March 15, 2006 to March 20, 2006 UT. 39

2.5 Trend of the averaged projection error calculated as the mean Euclidean distance between each input vector and its BMU code vector on the SOM map for each iteration of the training process. The x axis shows the number of iterations, each vertical leap of the graph before the vertical dotted line correspond to a learning rate value and neighborhood function width reduction. After the dotted line the graph shows the error reduction gained keeping the neighborhood function value equal to one (see 1.2.4), so each input vector modifies just its BMU code vector. 40

2.6 The SOM map obtained at the end of a training phase and visualised using the U-matrix method [Ultsch and Siemon, 1990]. At the last iteration of the SOM training process the whole data set is provided in input to the map and the code vectors are modified for the last time, preserving the information on how many and which spectra end up in each neuron. 40

2.7 Example of the projection onto the SOM map of a group of spectra. Each day is subdivided into 112 groups of the same size (120 spectra). Fig. 2.7a and 2.7b show the distribution of the group of spectra onto the map in two alterative ways (it can be noticed that the map is the same of that in fig. 2.6, cause it's not modified during the projection phase). Fig. 2.7c shows for each input vector (x axis) the corresponding neuron onto the map (y axis) and Euclidean distance value between the input vector and the code vector of the corresponding neuron. 42

2.8 Raoul distribution analysis of the seismic tremor acquired during March 16 (UT time). As the map is bidimensional, two map coordinates, "x" and "y", are needed to represent the position of a node on the map. The green line marks the explosion event at the Green Lake. 43

2.9 Raoul distribution analysis of the seismic tremor acquired during March 17 (UT time). 44

2.10 Raoul distribution analysis of the seismic tremor acquired during March 18 (UT time). 44

2.11 Raoul distribution analysis of the seismic tremor acquired during March 19 (UT time). 45

2.12 Location map showing Ruapehu volcanic massif (elevation 2797 m) with topographic contours. Contours are at 500 m intervals relative to sea level and the innermost contour is at 2500 m. Ruapehu volcano is on the southernmost margin of the Taupo Volcanic Zone (TVZ) (inset). Broadband (FWVZ) and short-period seismic sensor (DRZ) are shown as triangles. . . 46

2.13 Time series (top) and spectrogram (bottom) of seismic data recorded at FWVZ seismic station from 7.00 to 12.00 UTC, 4 October 2006. Spectrograms were produced using 2048 sample windows and a 1000 sample time step for data digitised at 100 Hz. The 5 hour interval was used to train the SOM. This interval includes a phreatic event at approximately 09:24 UTC. . 48

2.14 Waveforms for example volcanic earthquakes. A) 4 October 2006, B) 27 September 2007. The onset of the 25 September 2007 eruption is shown by the black arrow. 49

2.15 Dendrogram summarising the clustering of nodes of the SOM produced by training using the spectrogram of the seismic data recorded at FWVZ station from the interval shown in fig. 2.13. 51

2.16 Modified U-matrix for clustering purposes. SOM showing by colours the nodes assigned to each of the clusters after the training for 2006 data. The map should be seen as toroidal, i.e. the top side connects to the bottom, the left side connects to the right. 51

2.17 Sample spectrum characteristic of "blue tremor", obtained averaging only time windows of 2006 seismic data belonging to Cluster 1. 52

2.18 Sample spectrum characteristic of "green tremor", obtained averaging only time windows of 2006 seismic data belonging to Cluster 2. 53

2.19 Sample spectrum characteristic of "red tremor", obtained averaging only time windows of 2006 seismic data belonging to Cluster 3. 54

2.20 Time series of seismic data recorded at FWVZ station surrounding the 4 October 2006, 09.24 UT phreatic event, colour-labelled according to the clusterization of 2006 SOM. Each horizontal line represents 5 min. The transition from a "green tremor"-dominated regime to a "blue tremor"-dominated one through an interval of "red tremor" is evident. 55

2.21 Time series of seismic data recorded at FWVZ station surrounding the 27 September 2007, 08:26 phreatic event, colour-labelled according to the clusterization of 2006 SOM. Please note that SOM was not re-trained with any 2007 data. Each horizontal line represents 5 min. The transition from a "green tremor"-dominated regime to a "blue tremor"- dominated one through an interval of "red tremor" is evident. 55

2.22 Moment of inertia (y axis) trend for the groups of spectra of the days 23, 24 and 25 September 2007. The spectrogram of each day has been subdivided into 140 groups and each group have 120 spectra, corresponding to a time window of about 10 minutes. The thick gray line at the right side of the figure indicates the moment at which the seismic station stops to acquired tremor and get destroyed, the dotted line indicates when the explosion starts. 56

2.23 Distribution of the spectra of the groups from 6 to 11 of the 25 September. 57

2.24 Distribution of the spectra of the groups from 21 to 26 of the 25 September. 58

2.25 Distribution of the spectra of the groups 31 and 32 of the 25 September. . . 58

3.1 Simplified structure of a sedimentary basin and entities involved in the definition of the QTS. 62

3.2 Comparison between the horizontal component H_f, the vertical component V_f and the spectral ratio $QTS = {}^{H_f}/{}_{V_f}$ (loosely based on Nakamura [2000]). 63

3.3 Comparison between the H/V ratio obtained from environmental tremor and from seismic events (strong motion field). The graph *3.3a* shows the comparison of the frequencies f_0, the graph *3.3b* shows the comparison of the amplitudes $A(f_0)$. Loosely based on SESAME [2005]. 65

3.4 Satellite photo of the valley of Salta. The red dot indicates the area involved in the acquisitions of environmental tremor (from Google Earth). 66

3.5 Isopach map of the Lerma valley (loosely based on **?**). 67

3.6 Red dots show the measurement points where the three-component seismometer Mark L4 - 1sec has been placed to acquire the environmental tremor. Each number correspond to the seismic station number in the following presentation of the SOM results. Yellow dots show the position of a pair of the available drillings (see fig. 3.7). 68

3.7 Available drillings in the valley of Lerma (as supplied by the National University of Salta). 69

3.8 Microzonation map of the Lerma valley area on which the city of Salta is located (as supplied by the National University of Salta). Red dots indicate the measurement points (see also fig.). 70

3.9 Station N6 - spectrograms for the EW, NS and UD component respectively and the corresponding averaged spectrum. It's easy to note the high content of anthropogenic noise. 71

3.10 Station N6 - H/V spectrograms for the EW and NS component respectively and at the right side the corresponding mean spectrum is shown. 72

3.11 Station N6, EW component. Fig. 3.11a shows the dendrogram obtained computing the weighted cross-correlation matrix among the code vectors of the neurons and the clusterized SOM map (3.11b) that shows the seven detected clusters for the chosen dissimilarity threshold on the dendrogram. . 73

3.12 Station N6, EW component. Fig. 3.12a, 3.12c and 3.12d show the mean and the median spectrum for each one of the three largest clusters (cluster 1, 2 and 3 respectively) in the map trained with the H/V spectral ratios in EW direction. Fig. 3.12b shows data distribution over the three principal clusters. 74

3.13 Station N6, NS component. Fig. 3.13a shows the mean and the median spectrum for the cluster 1 in the map trained with the H/V spectral ratios in NS direction. Fig. 3.13b shows data distribution over the principal cluster. 75

3.14 Station N6 - EW component. Figures at the left side show the mean and the median spectrum for the two largest clusters (cluster 3 and 5 respectively) in the map trained with the rotated H/V spectral ratios. The *b*-value of the Konno-Ohmachi smoothing function has been fixed to a value equal to 20. The other two graphs (right side) show the data distribution over the two considered clusters. 75

3.15 Station N7 - EW component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 1) and for the cluster 3, the third one in order of size. Cluster 2, the second one in order of size, is characterized by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters. 76

3.16 Station N7 - NS component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 1) and for the cluster 5, the third one in order of size. Cluster 2, the second one in order of size, is characterized by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters. 77

3.17 Station N8 - EW component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 3) and for the cluster 1, the third one in order of size. Cluster 2, the second one in order of size, is characterized by a smooth peak at 2.9 Hz and by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters. 78

3.18 Station N8 - NS component. Figures at the left side show the mean and the median spectrum for the two largest cluster (cluster 2 and 1). Cluster 3, the only other one, is characterized by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters. 78

3.19 Station N9 - EW component. Figures at the left side show the mean and the median spectrum for the two largest cluster (cluster 1 and 2). The other two graphs (right side) show the data distribution over the two considered clusters. 79

3.20 Station N9 - NS component. Figures at the left side show the mean and the median spectrum for the two largest cluster (cluster 1 and 2). The other two graphs (right side) show the data distribution over the two considered clusters. 79

3.21 Station N3 - EW component. Figures at the left side show the mean and the median spectrum for the three largest cluster (cluster 1, 2 and 3). The other three graphs (right side) show the data distribution over the three considered clusters. 80

3.22 Station N3 - NS component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 3), the only interesting one. The other graph (right side) shows the data distribution over the considered cluster. 81

3.23 Station N4 - EW component. Figures at the left side show the mean and the median spectrum for the three largest cluster (cluster 1, 2 and 3). The other three graphs (right side) show the data distribution over the three considered clusters. 82

3.24 Station N4 - NS component. Figures at the left side show the mean and the median spectrum for the two largest clusters (cluster 3 and 1). The other graphs (right side) show the data distribution over the considered two clusters. 82

Abstract

This work consists of three main parts. In chapter 1 the Self-Organizing Maps (SOMs), proposed by T. Kohonen (1982), are analysed in their particular features. In order to apply the proposed SOM method to the analysis of geophysical data, the architecture and processes involved in the training of a SOM map are examined in detail. In the second part of the first chapter some methods to detect errors in the topological map organization and to compare, in terms of quality, different trained maps obtained using the same dataset are discussed. A useful and commonly used method to visualize a SOM map (the U-matrix, Unified distance matrix) is shown and a procedure is proposed for the automatic detection of clusters on the map, choosing a dissimilarity threshold below which the code vectors of the map neurons have features that may be considered statistically similar. A new discriminant function (the weighted cross-correlation coefficient) is then suggested instead of the classic Euclidean distance; it will be applied in each part of the SOM process to calculate the dissimilarity between any pair of vectors (input vectors and/or code vectors). The first chapter ends with a brief discussion about the advantages of using toroidal maps rather than flat maps.

The second part of this work (chapters 2 and 3) aims to show the results of some applications of the proposed SOM method to analyse geophysical data. In particular, in chapter 2, the SOM process is used to study the dynamical regimes of volcanic systems, starting from the tremor acquired at the Raoul volcano (Kermadec Islands, New Zealand) and Ruapehu volcano (North Island, New Zealand).
The Self-Organising Maps allow an automatic pattern recognition, as independent as possible from any a priori knowledge. In the training phase, volcanic tremor spectra are randomly presented to the network in a competitive iterative process. Spectra are then projected, ordered by time, onto the map. Every spectrum will take up a node on the map and their time evolution on the map can highlight the existence of different regimes and the transitions between them. We show a practical application on data recorded at Raoul Island during the period around the March 2006 phreatic eruption which reveals both a diurnal anthropogenic signal and the post-eruption system excitation.
SOMs are then applied to assess the low level seismic activity prior to small scale phreatic events at Ruapehu volcano New Zealand. A hierarchical clusterization of the SOM nodes follows the map training phase and the data projection onto the trained map. Two Ruapehu events were examined: a phreatic event on 4 October 2006 which displaced the crater lake producing a 4 m high wave on the lake edge, and the more energetic 25 September 2007 phreatic eruption. The SOM analysis provides a classification of tremor spectral patterns that cluster into three regimes, labelled by colours. The pattern for both eruptions is consistent with a pre-eruption spectral pattern including enhanced spectral energy in the range of 4 to 6 Hz. This gives way to spectra having broader energy between 2 and 6 Hz, just prior to the eruption. The post eruption pattern includes spectral peaks at generally

lower frequencies of 2 to 4 Hz. Clusterization into only three groups yields highly non-unique solutions which cannot explain the variety of processes operating at Ruapehu over long time periods. However, it is noteworthy that the SOM map trained with 2006 data can correctly process also data recorded in 2007. In particular, the approach highlights remarkable similarities that may be explained by a pattern of slow pressurisation under a hydrothermal or magmatic seal, followed by seal failure and subsequent de-pressurisation for the two events studied.
In the third chapter the SOM method has been applied to the HVSR technique (or H/V spectral ratio or Nakamura's method) with the intent to improve this method that allows the identification of the fundamental frequency that characterizes the sedimentary deposits of a site in a cheap and relatively easy way. The main issue is that difficulties in data interpretation occur especially in non-invasive geophysical techniques and/or when the data are multidimensional, non-linear and highly noisy. Another important task is to ensure an efficient automatic data analysis, in order to allow a data interpretation as independent as possible from any a priori knowledge. The application of the proposed SOM method to ensure a more reliable identification of the peak of the H/V function within the microzonation project for the Salta city area (Argentina) is discussed. SOM results are represented as two-dimensional maps, with a non-parametric mapping that projects the high dimensional original dataset in a fashion that provides both an unsupervised clustering and a highly visual representation of the data.

The third part of this work consists in an appendix that shows the scripts for the implementation of the proposed SOM method. The SOM process has been entirely implemented using the free software environment R (*http://www.r-project.org/*). Some packages were already available in the repositories of R software (*http://cran.r-project.org/web/packages/*) and were adapted to implement some parts of the process, some other parts have been originally developed from scratch.

Part I

Theory

Chapter 1

Self Organizing Maps

Self Organizing Maps (SOMs) are a particular kind of unsupervised artificial neural networks. Introduced in 1982 by Teuvo Kohonen, they are used in the analysis and visualization of data sets of high dimension. Quoting the man who conceived and formalised the SOM process, we can say that “the main applications of the SOM are in the visualisation of complex data in a two dimensional display, and creation of abstractions like in many clustering techniques” [Kohonen, 2001].

Maps are not only tools of visualization, they also represent an analysis tool. Appropriate display of clusters can give the analyst an insight that it is impossible to get from reading tables of output or simple summary statistics. For some tasks, appropriate visualization is the only tool needed to solve a problem or confirm a hypothesis, even though we do not usually think of maps as a kind of analysis.

The basic principle of the SOM is that our knowledge organisation at higher levels is created during learning by algorithms that promote self-organisation in a spatial order. SOMs are then neural networks that pay attention to spatial order. A SOM consists of a grid of a predetermined number of equally spaced nodes, it’s aimed at the display of the sample input data, of arbitrarily large size, in a particularly simple structure and small size. The grid is usually a bi-dimensional grid, but also three-dimensional grids can be used. Referring to the bi-dimensional grids, generally the number of nodes varies from a few hundred to a maximum of a few thousand, depending on the nature of the sample. In correspondence of each node of the grid is placed a so-called "neuron" (by analogy with the structure of the cerebral cortex). This is nothing more than a vector with the same size of the vectors that constitute the data of the sample to be analysed, connected with the other neurons to form the network. The SOM network evolves during the process and does it modifying, for any given input, the neuron (and therefore the vector that corresponds to the unit on the grid) most similar to it and neurons belonging to the grid area defined as “neighborhood” of that neuron.

The SOM architecture form can be thought as the representation of data features that assume the form of a self-organizing feature map, geometrically organized on a grid. In the pure form, the SOM defines an “elastic net” of points (each one is described into the n-dimensional input space by an n-dimensional vector) that are fitted to the input data space to approximate its density function in an ordered way. The algorithm takes thus a set of n-dimensional objects as input and maps them onto nodes of a bi-dimensional grid, resulting in an orderly feature map. To make sure that the display of the input data features is significant, the SOM aims at preserving the topological properties of the data set, so that similar neurons will occupy adjacent positions on the map. Similarly two input data, considerably dissimilar, have to belong to different neurons on the map and these neurons have to occupy two positions not close on the map.

Each input data is associated with one and only one neuron that represents its image on the grid. Whereas neurons are distributed in the input space as entities of the same size of the input data, they constitute an approximation of the distribution of the sample: the more densely populated regions corresponds to a higher number of neurons on the map and vice versa.

SOM applications are multiple and potentially related to each field of study where the aim is to identify complex and non-linear statistical relationships in the input space, by means of simple geometric relationships on the map. Different studies applied SOM for example to improve the analysis and visualization of gene expression microarray data in human cancer [Hautaniemi et al., 2003], or to visualize performance data of a GSM network analysing degradations in signaling and traffic channel capacity of the network [Lehtimaki and Raivio, 2005], or again to video surveillance involving moving object detection, tracking and normal/abnormal event recognition [Dahmane, 2005].

1.1 SOM architecture

As previously mentioned, the main goal of a SOM is to associate to a sample of arbitrarily large size, a map, performing a sort of projection of the data on to the map in such a way as to ensure that they are organized according to a topological order, that is not predetermined, but reflects the intrinsic organization of the sample itself. SOM architecture defines a flexible network of points, called neurons, each neuron is associated with a weight vector (also called code vector) having the same dimension of data. Each neuron occupies a different node of the grid and the whole SOM network, allowing the modification of the weight vectors according to the input data, self-organizes to approximate at best the sample's density function. Thus SOM algorithm receives in input a set of n-dimensional objects, each of which can potentially go to activate any of the neurons. Neurons are fully connected together and activation, as well as the consequent change of the weight vectors, does not generally involve only the single activated unit, but a neighborhood area.

Various properties of the brain were used as an inspiration for a large set of algorithms and computational theories known as artificial neural networks (ANN; Haykin, 1998). Such algorithms have shown to be successful, however a vital aspect of biological neural networks was omitted in the algorithm's development. This was the notion of self-organisation and spatial organisation of information within the brain. In 1981 Kohonen proposed a method which takes into account these two biological properties and presented them in his SOM algorithm Kohonen, 1981.

The SOM algorithm generates a map representing a scaled version of n-dimensional input data, this map is due to the algorithm's inspiration from the way that mammalian brains are structured and operate in a data reducing and self-organised fashion. In the brain, during growth, a subdivision takes place in specialised areas for input stimuli. Similarly in the SOM, during learning, takes place the creation of a map subdivided into regions, each region responding to a particular feature of the input space. An example from the biological domain is the somatotopic map within the human brain, containing a representation of the body and its adjacent and topographically almost identical motor map responsible for the mediation of muscle activity.

Moreover similar types of information (usually sensory information) are held in close spatial proximity to each other in order for successful information fusion to take place as well as to minimise the distance when neurons with similar tasks communicate. For example sensory information of the leg lies next to sensory information of the sole. The fact that similarities in the input signals are converted into spatial relationships among the responding neurons, provides the brain with an abstraction ability that suppresses trivial details and only maps the most important properties and features along the dimensions of the brain's map [Ritter et al., 1992]. A group of neighboring neurons is organized in a high activity "bubble", well distinguished from all other neurons in a low activity state. As we shall see, this biological phenomena is modeled in a Kohonen network when the weight vectors update is extended to the neighborhood of the winner neuron (also called Best Matching Unit, BMU), so defined because among other units of the network, that neuron is the most similar to the input data.

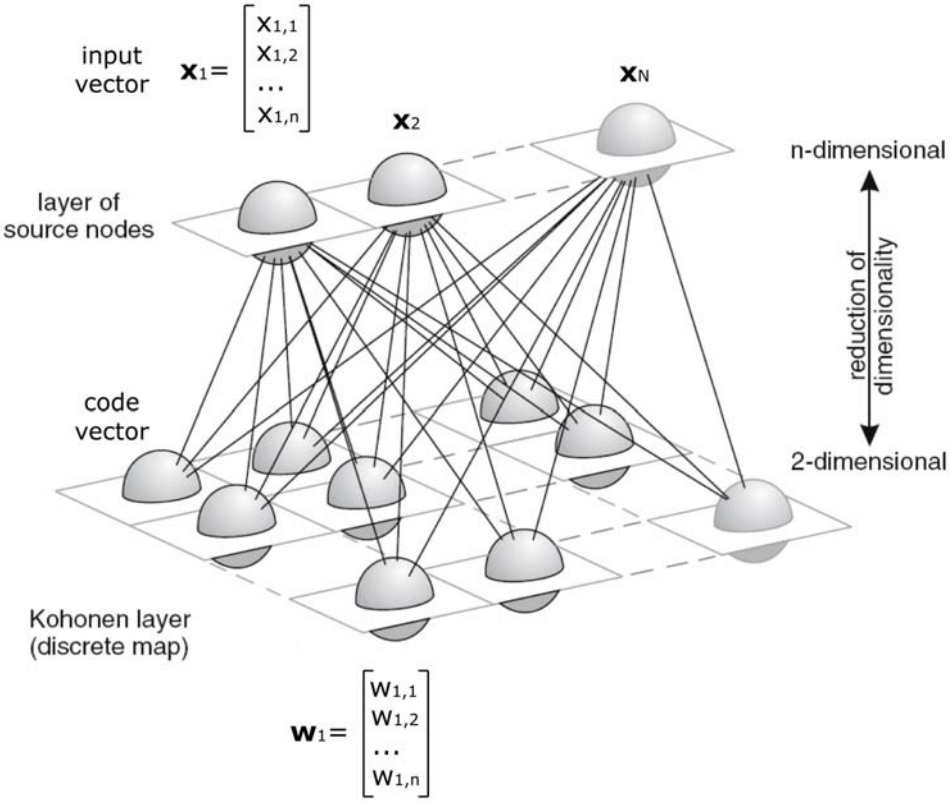

input vector
$\mathbf{x}_1 = \begin{bmatrix} x_{1,1} \\ x_{1,2} \\ \dots \\ x_{1,n} \end{bmatrix}$
$\mathbf{x}_2$
$\mathbf{x}_N$
n-dimensional
layer of source nodes
reduction of dimensionality
code vector
2-dimensional
Kohonen layer (discrete map)
$\mathbf{w}_1 = \begin{bmatrix} w_{1,1} \\ w_{1,2} \\ \dots \\ w_{1,n} \end{bmatrix}$

Figure 1.1: Illustration of a SOM neural network. The SOM projects the information of an n-dimensional feature space into a bi-dimensional grid of neurons, whereby the dimension are reduced. After training and cluster recognition SOM map should be able to identify an input activating just the corresponding area onto the map in function of input's features. The Kohonen layer tipically consists of a localised region of active neurons against the quiet background (loosely based on Klose [2006], fig. 1a).

The first step in the creation of a SOM is to determine its size and topology. The SOM's size is given by the number of neurons, determined as the product of the chosen length of the two sides of the map. It has been shown that while self-organizing maps with a small number of nodes behave in a way that is similar to K-means[1], larger maps rearrange data

[1] K-means clustering is a method of vector quantisation that is popular for cluster analysis in data mining. K-means clustering aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean, serving as a prototype of the cluster. This results in a partitioning of the data space into Voronoi cells, with the aim to minimise the within-cluster sum of

in a way that is fundamentally topological in character. Moreover it must be noted that SOM and K-means algorithms are rigorously identical when the radius of the neighborhood function in the SOM equals zero. In this case the update only occurs in the winning unit just as happens in K-means [Bodt et al., 1997].

There are three principal options to choose the size of the SOM map. The first one is to choose a very large map on which the number of neurons is greater than the number of input patterns [Ultsch, 1993]. The second and usually the most common option is to build a medium sized map, smaller than the number of input patterns, but still large enough to have a few units representing each cluster existing in the data [Kohonen, 2001]. Finally, there is also the possibility to choose small maps where the number of units is drastically smaller than the number of input vectors, usually with only one unit for each expected cluster [Bação, 2004]. The relevance of the choice of the size of the SOM is such that it can be argued that SOMs of significantly different sizes constitute different tools, which may be used to perform different tasks.
When opting for a larger map the underlying assumption is that we wish to explore in detail the underlying distribution of the data. By using more units than input patterns it is possible to obtain very large U-matrices (1.4) on which distances between input patterns can easily be identified. This can be seen as a strictly exploratory exercise. The data reduction, in this case, is solely based in projecting the n-dimensional space onto a bi-dimensional space.
The decision to build a medium sized map can be seen as a compromise, in the sense that although reducing the number of dimensions and creating clusters, it still enables the user to understand the basic (or broad) distribution of the data, eventually leading to further and more severe reductions.
Finally, small maps are used when the user is interested in clustering data without concerns about the detailed analysis of its distribution. In this case the primary objective is to form clusters of input patterns which are as similar as possible, aiming at a one step substantial data reduction. In this context the U-matrix is of little value, and component planes become more relevant as they allow a simple description of the resulting clusters.

Two topology types are frequently used. The first, and more frequently encountered, is the square topology, where each neuron is connected to four neighbouring neurons at a unitary distance and four neurons at a distance equal to $\sqrt{2}$. The second possibility is the use of a hexagonal topology, with six neighbours to every neuron, each one at a unitary distance. The choice of a topology rather than the other involves a variation of the "neighborhood relations", the relevance of which will be more clear by analysing the SOM learning process. Is important to note that at the edges of the map neurons have a neighborhood less numerous and this implies that these neurons are potentially activated fewer times.

1.2 SOM algorithm

The SOM belongs to the category of the unsupervised competitive learning networks. It is called competitive learning because there is a set of nodes that compete with one another to locate the Best Matching Unit (BMU, the neuron with the most similar code vector to

squares. K-means has problems when clusters are of differing sizes and densities, have concavity shapes and the dataset has outliers. Due to the effect of the neighborhood parameter which forces units to move according to each other in the early stages of the SOM process, SOM is less prone to local optima than K-means [Bação et al., 2005].

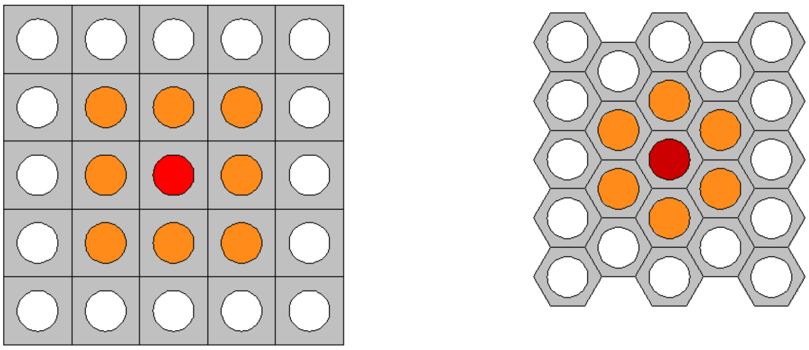


Figure 1.2: Network size and topology type of a SOM are chosen before training begins. Notice overall shape difference for SOMs with identical size (side length equal to 5 in both cases), but different topology type.

the input) that becomes active. Therefore the competitive learning means that a number of nodes is comparing the same input data with their code vectors and the node with the best match (say, "winner") is then tuning itself to that input; in addition the BMU activates its topographical neighbours in the network to take part in tuning to the same input. The more a node is distant from the winning node the weaker the learning. The level of similarity between the input data and each code vector is measured typically using the Euclidean distance as the discriminant function. In this work an unconventional choice of the discriminant function is proposed.

It is also called unsupervised learning because no information concerning the correct clusters is provided to the network during its training. Like any unsupervised clustering method, the SOM can be used to find clusters in the input data, and to identify an unknown data vector with one of the clusters. Moreover, the SOM represents the results of its clustering process in an ordered bi-dimensional space. A mapping from a n-dimensional data space onto a bi-dimensional grid of nodes is thus defined. Such a mapping can effectively be used to visualise metric ordering relations of input data.

Neurons code vectors are modified according to two functions:

- during learning, not only the weight vector of the winning neuron is updated, but also those of its map neighbors and, thus, they end up responding to similar inputs. This is achieved with the *neighborhood function*, which is centered at the winning neuron, and decreases with the grid distance to the winning neuron (the BMU). During training, the radius of this function will usually decrease, so that each unit will become more isolated from the effects of its neighbors. It is important to note that many implementations of SOM decrease this radius to 1, meaning that even in the final stages of training each unit will have an effect on its nearest neighbors, while other implementations allow this parameter to decrease to zero. The degree of lateral interaction between stimulated neuron and neighboring neurons is usually described by a Gaussian function;
- the *learning function*, which defines the degree of modification of the activated neurons, allowing these to adapt to the input. As the radius of the neighborhood func-

tion, also this function decreases monotonically with the number of iterations of the process, so the maximum learning rate occurs in the initial stage of the SOM algorithm.

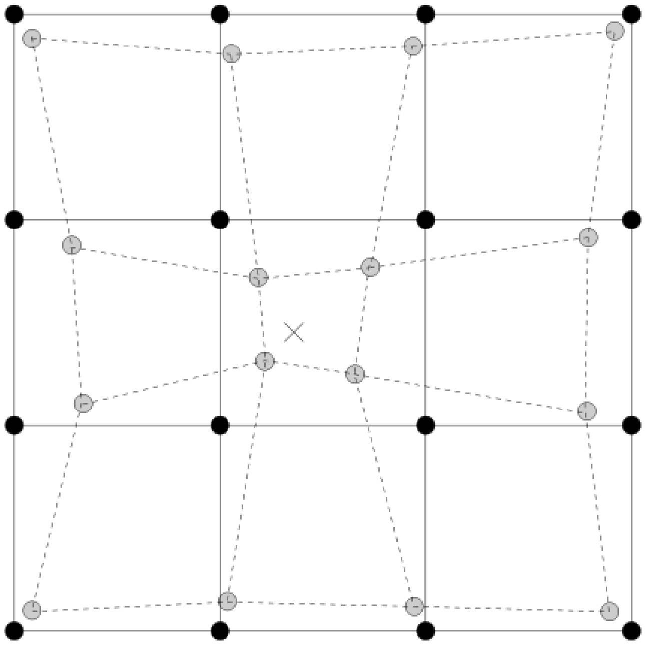


Figure 1.3: The effect of the neighborhood function in the SOM algorithm. Starting from a perfect arrangement of the weights of a square grid (full lines), the weights nearest to the current input (indicated with the cross) receive the largest updates, those further away smaller updates, resulting in the updated grid (dashed lines).

When an input activates a neuron and then it becomes the BMU, a correspondence between the input data and the winner neuron is created, among all the neurons on the map, that neuron is the one that projects the input onto the map with the best approximation. It is also recalled that the process leads to train the map preserving the topological properties and thus similar neurons occupy adjacent nodes on the grid, it follows that the SOM leads to the creation of clusters that split the sample and make viewable the intrinsic relations existing between the data. This kind of neural networks is called “unsupervised” since any kind of information is provided during the training phase of the map and about the correct and expected data clustering. Furthermore, unlike for example K-means algorithm , the number of clusters is not given in advance.

There are two high-level stages of the algorithm that ensure a successful creation of a map. The first stage is the global initialisation stage in which we start with a map of predefined size and not organized at all, with neurons of random nature; a sort of a rough and incomplete estimation of the input data distribution. Initialisation may be done using completely random values (which usually involves to slow convergence), or using values obtained from randomly selected input patterns [Kohonen, 2001]. Once a desired number of data is used for such initialisation step, the algorithm proceeds to the fine-tuning stage, where the effect of the input data on the topography of the map is monotonically decreasing with time, while individual neurons and their neighborhood are activated and thus fine tuned to increase their discriminant function value towards to the present input.

The original algorithm developed by Kohonen includes initialisation followed by three steps which are repeated until the process goes through many iterations until it converges, i.e. the adjustments approach zero:

1. *competitive process* implies to find the neuron whose code vector is closest to the input vector in the n-dimensional space. This neuron wins the competition and it's called the BMU;

2. *cooperative process* to identify and activate neighboring neurons;

3. *adaptive process* implies to adjust the weights of the winning neuron the nodes close neurons, so that they become still closer to the input vector in the n-dimensional space.

It is possible to define a number of iterations in advance, alternatively map training process can be stopped when no more substantial changes to the code vectors are carried out. At each iteration the entire data set is processed and each data is fed to the map just one time and in random order.

Formally, the SOM process may be described by the following algorithm:

```
Let:  X   be the set of $n$ training patterns $x_1, x_2 \ldots x_n$ ($n$ is the number of
          input vectors);
      W   be a $p \times q$ grid of units $w_{i,j}$ ($p \times q$ is the number of neurons),
          where i and j are their coordinates on that grid ;
      $\alpha$   be the learning rate, assuming values in $]0,1[$ , initialised to a
          given initial learning rate;
      $\sigma$   be the radius of the neighborhood function $h(w_{i,j}, w_{BMU}, \sigma)$
          initialised to a given initial radius.

while $\alpha > 0$
{
  for k=1 to n
  {
    for $w_{i,j} \in W$ (be i=1,2..p and j=1,2..q)
    {
      calculate the distance $d_{i,j} = f_{DISCRIMINANT\,FUNCTION}\{x_k;\ w_{i,j}\}$
      select the unit $w_{i,j}$ that minimises $d_{i,j}$ as the BMU
        update each unit  $w_{i,j} \in W$ : $w_{i,j} = w_{i,j} + \alpha \cdot h(w_{i,j}, w_{BMU}, \sigma) \cdot d_{i,j}$
      decrease the value of $\alpha$ and $\sigma$
    }
  }
}
```

Once the SOM map has accomplished the training phase it should be organised in such a way to be able to analyse the same data set used during the learning process. Moreover is also possible to use the same map to analyse other samples whose data are of the same type as those of the sample used for the training. Each time that a new data is projected onto the map, the network finds on the grid the neuron whose code vector has the minimum dissimilarity from the input vector, then the output value is no more than the location of that neuron and therefore its cluster.

1.2.1 Code vectors initialisation

In literature the issue that SOM algorithm is very sensitive to code vectors initialisation and then the solution obtained by SOM strongly depends on that, is often emphasised [Attik et al., 2005]. Certainly all existing SOM initialisation methods do not guarantee to obtain the best minimal solution, for this reason clusters shape and number of neurons constituting each cluster may vary. During the performed data analysis it has never happened that starting from two different map initialisations the process reaches to a completely different map organisation. Of course clusters position on the map vary and a slight difference in the number of neurons in each cluster has been noticed sometimes, but the process every time converged to a minimum in terms of dissimilarity between each input vector and the corresponding code vector onto the map. Moreover for a chosen threshold on the dendrogram (see 1.6) the number of clusters doesn't vary and neither the peculiar characteristics of the sample found in the clusters. This good result is probably because of the care that has been taken in the choice of the map size and in the choice of training parameters.
Two initialisation approaches are usually implemented for clustering methods, namely supervised initialisation approach and unsupervised initialisation approach. The first one is a supervised selection which assumes that a subset of samples data can be labeled in accordance with a tentative classification scheme. The second is generally preferred because of its better computational behaviour, without the manual effort to label unlabeled subset of data, since usually little or nothing is known about the intrinsic structure of the sample.

Random initialisation approach was found to be a preferable initialisation approach for its simplicity. This approach is not necessarily the best approach for producing a rapid convergence to a stable state, but it doesn't assume any a priori knowledge about the sample. It can be performed choosing for each component of each code vector a value randomly from the range of values observed in the data set. Otherwise sample data can be directly used assigning at each code vector a randomly chosen data taken from the sample.

1.2.2 Competition process

At iteration t the input $\mathbf{x}(t)$ is sorted randomly, but without repetition, between the data of the whole sample X, so that when t equals the number of sample data, each of these has been presented to the map one and only one time. Generally the same set X is submitted to the SOM more times.

For each input $\mathbf{x}(t) \in \mathbb{R}^n$, each neuron on the map uses the discriminant function to compute its value of similarity with the input. The neurons compete with each other in this way, the winner is the neuron that has the greatest similarity value.

Let be $\mathbf{x}_i(t) = [x_{i,1}(t), \, x_{i,2}(t) \ldots x_{i,n}(t)]$ the i-th input vector chosen randomly in, at t-th process iteration. Let be $\mathbf{w}_j(t) = [w_{j,1}(t), \, w_{j,2}(t) \ldots w_{j,n}(t)]$ the code vector of the j-th neuron on the grid. In order to find the most similar neuron to the input vector, the scalar products $\mathbf{w}_j(t) \cdot \mathbf{x}_i(t)$ ($j = 1, \, 2 \ldots n_w$ and n_w is the number of neurons onto the map) are compared, the neuron whose code vector maximises the scalar product value is chosen to be the BMU. In other words, the maximum value of the scalar product corresponds to the minimum Euclidean distance:

$$||\mathbf{x}_i(t) - \mathbf{w}_j(t)|| = \sqrt{\sum_{m=1}^{n} [x_{i,m}(t) - w_{j,m}(t)]^2} \tag{1.1}$$

The winner neuron is so defined because it verifies the following condition:

$$||\mathbf{x}_i(t) - \mathbf{w}_{c(x)}|| = \min_j \{\, ||\mathbf{x}_i(t) - \mathbf{w}_j(t)|| \,\} \tag{1.2}$$

The Euclidean distance is the typical choice of discriminant function, but it isn't the only possible similarity measure, some experiments with the correlation coefficients are carried out. Euclidean distance measure is very restrictive in comparison to the human perception of time series. A time series and its translated copy appear dissimilar under the Euclidean distance (because the comparison is made pointwise), whereas a human would perceive both series as similar. As the human perception is tolerant to translational effects, using the cross correlation distance would be a better choice than Euclidean distance. Moreover a measure of similarity should be invariant under admissible data transformations, which is to say changes in scale too.

It has been demonstrated [de Gelder et al., 2001] that various similarity criteria of two functions, $f(y)$ and $g(y)$, including the sum of squared differences, the correlation coefficient and the overlap integral, are related to the cross-correlation function $R_{f,g}(\tau)$ at $\tau = 0$.

$$R_{f,g}(\tau) = \int f(y)g(y+\tau)\,dy$$

Thus, they cannot provide any information about patterns that are shifted relative to each other. In this work a generalised expression for similarity proposed by [de Gelder et al., 2001] , $S_{f,g}$, which is based on a weighted cross-correlation function, a weighting function $z(\tau)$ normalised with the product of the two weighted autocorrelation functions:

$$S_{f,g} = \frac{\int z(\tau)R_{f,g}(\tau)\,d\tau}{\sqrt{\int z(\tau)R_{f,f}(\tau)\,d\tau \int z(\tau)R_{g,g}(\tau)\,d\tau}}$$

$z(\tau)$ is a triangular weighting function of width defined as $z(\tau) = 1 - {}^{|\tau|}/_{h}$ if $|\tau| < h$ and $z(\tau) = 0$ if $|\tau| \geq h$. The BMU is the neuron $w_{c(x)}$ that maximises the value of the function $S_{f,g}$. The overall effect of the competition process is that the continuous input space X of activation patterns is projected onto the map discrete output space M.

1.2.3 Cooperation process

The winning neuron $w_{c(x)}$ determines the spatial location of a topological neighbourhood of neurons on the map that have to be excited. A neighbourhood function determines how strongly the neurons are connected to each other. This function must be uni-modal, with the lateral distance $d_{c(x),j} = \|r_c - r_j\|$ computed on the SOM map. The distance is calculated between the BMU neuron $w_{c(x)}$ and each generic neuron w_j (r_c and r_j determine the position of the two neurons on the map). A typical choice of $h_{c(x),j}$ is the Gaussian function:

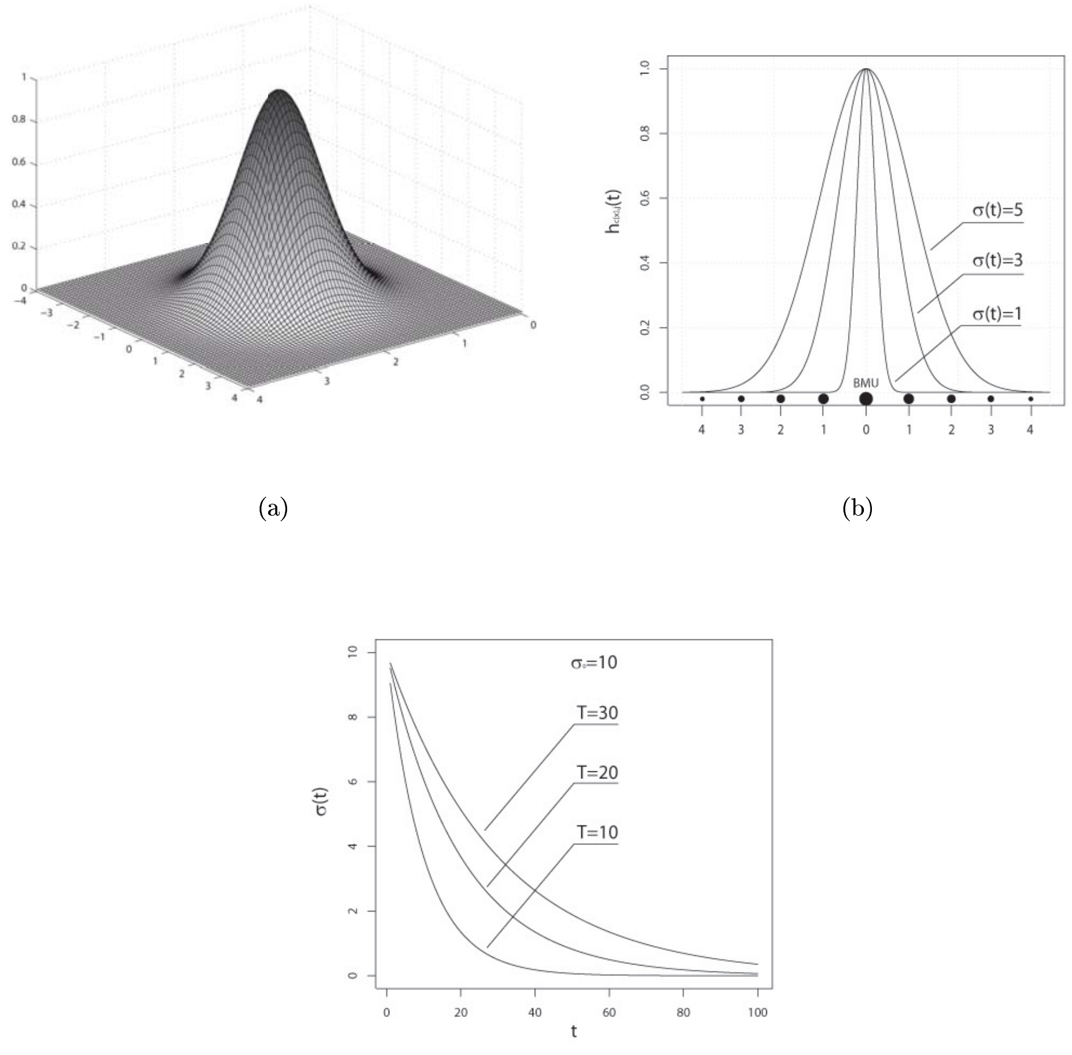
$$h_{c(x),j}(t) = \exp\left(-\frac{d^2_{c(x),j}}{2\sigma(t)^2}\right) \tag{1.3}$$

The value of the neighbourhood function also depends on the discrete time t that identifies the iteration number: at every step t, the whole dataset will be (re)processed by the network.

It is noteworthy that at each step the dataset is re-ordered randomly and the radius $\sigma(t)$ of the neighbourhood function is decreased monotonically in order to facilitate convergence. The neighborhood radius $\sigma(t)$ defines the width of the neighborhood function and contributes to establish the degree of influence that the BMU exerts on neurons that the cooperative process has defined as constituents of its neighborhood. A popular choice for the effective width of the neighbourhood function is the exponential decay:

$$\sigma(t) = \sigma_0 \cdot \exp\left(-\frac{t}{T}\right) \tag{1.4}$$

where σ_0 is the value of σ at the beginning of the SOM algorithm, T is a time constant and t identifies the iteration number.



(a)

(b)

(c)

Figure 1.4: Plot of the neighborhood function built according to the Gaussian function defined by equation 1.3 (*1.4a* and *1.4b*). Black dots placed along the x-axis (1.4b) shows how the influence of the BMU on neurons in its neighborhood decreases away, on the map, from the same BMU. The value of the neighborhood radius decreases with the increase of the number of iterations t (1.4c), so the width of the Gaussian function decreases more and more towards the center, reducing the number of neurons that constitute the neighborhood of the BMU. At the limit, in the final iterations of the SOM process, only the BMU is active on the map.

A possible alternative to the Gaussian function is the use of a simple rectangular function defined as:

$$h_{c(x),j}(t) \quad = \quad \begin{cases} constant & \text{when}\, d \leq \sigma(t) \\ 0 & \text{when}\, d > \sigma(t) \end{cases} \tag{1.5}$$

1.2.4 Adaptive process

An adjustment of the code vectors of excited neurons is carried out in order to reinforce the answer of the BMU neuron for similar input patterns, slowly allowing the map to be partitioned into consistent clusters. By using discrete-time formalism, the adaptive process usually takes place according to the following model:

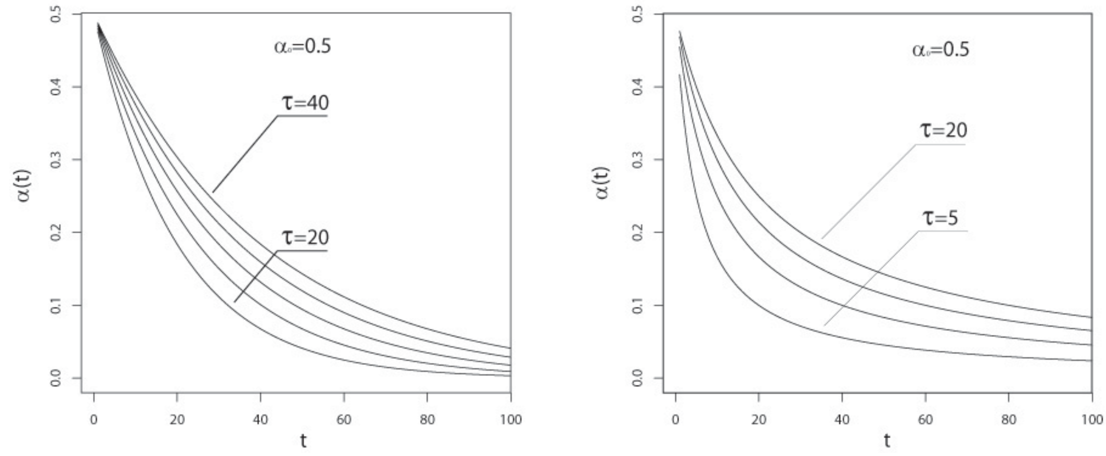
$$\mathbf{w}_j(t+1) = \mathbf{w}_j(t) + \alpha(t) \cdot h_{c(x),\,j}(t) \cdot [\mathbf{x}_i(t) - \mathbf{w}_j(t)] \tag{1.6}$$

where $\mathbf{w}_j(t) = [w_{j,1}(t),\, w_{j,2}(t) \ldots w_{j,n}(t)]$ is the code vector of the j-th neuron at the iteration t and $x_i(t)$ is the input vector. $\mathbf{w}_j(t+1)$ is the updated code vector that will take part in the competition process at iteration $t+1$. The parameter $\alpha(t)$ $(0 \leq \alpha(t) \leq 1)$ is the learning-rate factor, a monotonic function which also decreases with the regression step t in order to facilitate convergence. Two suitable functions are shown below:

$$\alpha(t) = \alpha_0 \cdot \exp\left(-\frac{t}{\tau}\right) \tag{1.7}$$

$$\alpha(t) = \alpha_0 \cdot \frac{\tau}{\tau + t} \tag{1.8}$$

where t is, as usual, the iteration number and τ is a new time constant (fig. 1.5).



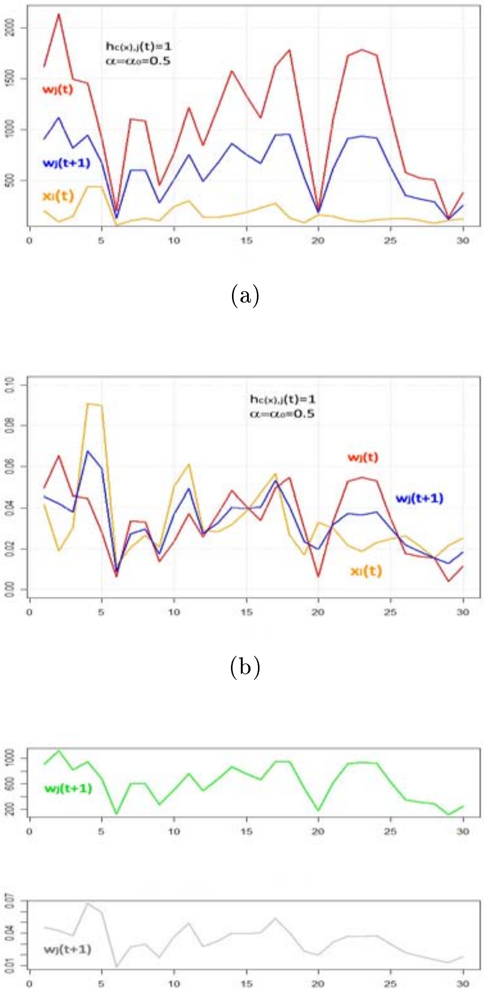
(a)

(b)

Figure 1.5: Graphical representation of α, as a function of the the number of iterations t. In fig. *1.5a* the diagram of function 1.7 is drawn for t equals to 100 iterations, $\alpha_0 = 0.5$ for τ between $\tau = 20$ and $\tau = 40$, increasing the value of τ in four steps. In fig. *1.5b* the diagram of function 1.8 is drawn for t equals to 100 iterations, $\alpha_0 = 0.5$ for τ between $\tau = 5$ and $\tau = 20$, increasing the value of τ in three steps. Any iteration with $\alpha = 0$ has no sense, since the network does not improve its organisation when the learning rate is equal to zero.

For an explanatory purpose a couple of generic vectors of size 30 are chosen. They would represent the code vector $\mathbf{w}_j(t)$ of the *BMU* (1.6a, red line) for the input vector $\mathbf{x}_i(t)$ (1.6a, yellow line). If, as supposed, $\mathbf{w}_j(t)$ is the code vector of the neuron that has won the competition process, then the neighborhood function $h_{c(x),j}(t)$ is equal to one (1.6b). If this is the first iteration of the SOM process, then the learning rate α is maximum and equal to α_0. In the fig. 1.6a the blue line represents the updated code vector $\mathbf{w}_j(t+1)$, modified in accordance with the equation 1.6.

In fig. *1.6b* the input vector $\mathbf{x}_i(t)$ and BMU's code vector $\mathbf{w}_j(t)$ are normalised with respect to the sum of their values. The updated code vector $\mathbf{w}_j(t+1)$ has been calculated using the eq. 1.6. Fig. *1.6c* suggests the comparison between the code vector $\mathbf{w}_j(t+1)$ that has been calculated from the not normalised vectors and the same code vector updated from the vectors $\mathbf{x}_i(t)$ and $\mathbf{w}_j(t)$ normalised with respect to the sum.



(a)

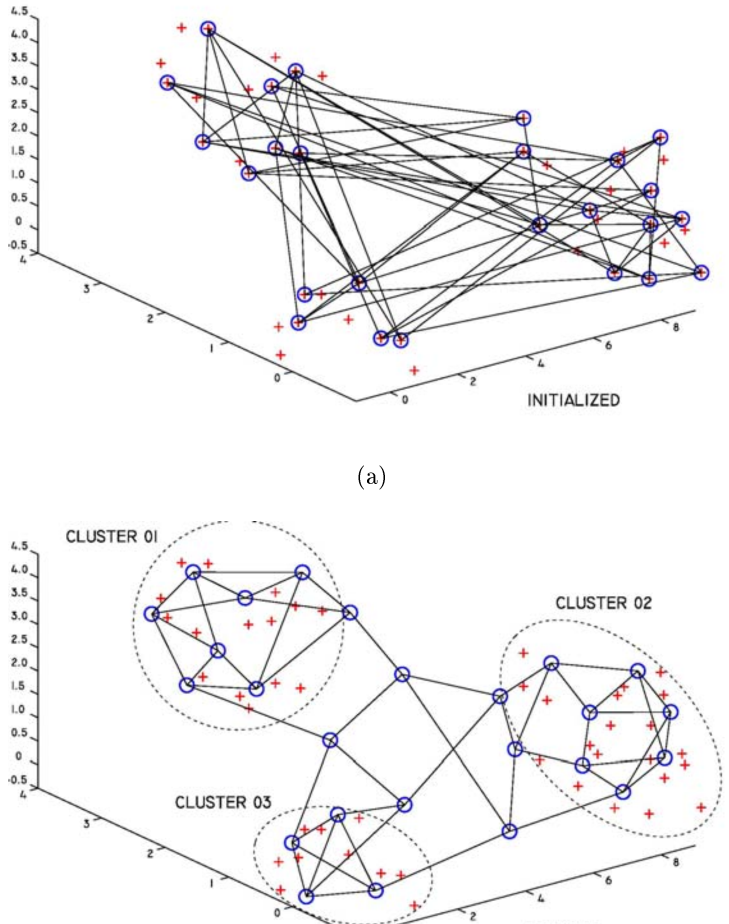
(b)

(c)

Figure 1.6: Graphical representation of a generic input vector $\mathbf{x}_i(t)$ and the corresponding BMU's code vector $\mathbf{w}_j(t)$. The updated code vector $\mathbf{w}_j(t+1)$ has been calculated using the eq. 1.6 (1.6a). In fig. 1.6b $\mathbf{x}_i(t)$ and $\mathbf{w}_j(t)$ have been normalised with respect to the sum of their own values. Fig. 1.6c proposes the comparison between the code vector $\mathbf{w}_j(t+1)$ in fig. *1.6a* and the same code vector drawn in fig. 1.6b.

The goal of the SOM process is to minimise the distance of each data input from the corresponding neuron on the map. The following example aims to show the organisation of the map in the input space. For this purpose a data set in a tri-dimensional input space has been chosen and the SOM map has been initialised choosing the code vectors randomly among the same input data. The simple data set is composed by three clusters of data points. The clusters are uniformly distributed about the points $(0; 0; 0)$, $(3; 3; 3)$ and $(9; 0; 0)$ with a maximum deviation from the center of 1 unit. The three clusters have an amount of 10, 15 and 20 points respectively.

Since the dimensionality of the input space (tri-dimensional) is larger than the dimensionality of the SOM (bi-dimensional), the map will try to balance the competing errors in how well it maps the data points versus how well it maps the topology, as if the SOM process would try to bend a sheet of paper to fill the interior of an empty cube.



(a)

(b)

Figure 1.7: Staying in the tri-dimensional input space, figure shows the position of the data (red crosses) and the position of the SOM map's nodes (blue circles) at the initialisation step (1.7a) and for the trained map (1.7b).

In fig. 1.7a red crosses show the input data and the blue circles show the position of SOM map's nodes in the input space. The map is squared and its sides dimension are $5x5$. SOM process has been iterated 100 times and the learning-rate factor σ varied between $0,5$ and $0,01$. In fig. 1.7b the trained map has completed the SOM process and it's well "stretched" in the input space.

1.3 Quality map measures

One of the most remarkable properties of the SOMs is the conservation of the dataset's topological order during the projection of the data on the Kohonen grid.

Two widely used SOM quality measures try to evaluate the vector projection, which is sometimes referred to topology preservation, and vector quantisation, related to data compression techniques. There is a tradeoff between these two measures, because increasing projection quality usually decreases the projection properties. Topology preservation is a property that is not easy to define and then to measure, since usually a major reduction of dimensionality is performed, some information is necessarily lost in the projection process. Usually a measure of proximity between input and output spaces is performed.

Moreover even if the SOM process is called "unsupervised", since it's fully automated from the first step, the map initialisation, to the last iteration of the training process, some decisions have to be taken about for example the dimension of the map, the shape of the neighborhood function and the functions that describe the decrease of the neighborhood radius and learning rate, together with the choice of their initial and final values.

For these reasons sometimes the need to process multiple times the same data set to get different SOMs arises. However there is the need to measure the SOMs quality, both in case of need to compare the maps to choose the one better structured and potentially more suitable for the data analysis, and in the case that just one map has been trained.

1.3.1 Average quantisation error

The average quantisation error is traditionally related to all forms of vector quantisation and clustering algorithms. Thus, this measure completely disregards map topology. The average quantisation error is computed by determining the average distance of the input vectors to the corresponding code vectors, so an average measure of similarity between each data and its winning neuron on the map (the BMU) is computed:

$$E_q = \frac{1}{N} \cdot \sum_{i=1}^{N} ||\mathbf{x}_i - \mathbf{w}_{c(x),\,x_i}|| \tag{1.9}$$

where N is the data set size, $\mathbf{x}_i$ is the input data and $\mathbf{w}_{c(x),\,x_i}$ is the code vector of the BMU (the subscripts $c(x)$, x_i indicates that the neuron with the code vector $\mathbf{w}$ is elected to be the BMU by the input $\mathbf{x}_i$).

For any given data set, the quantisation error can be reduced by simply increasing the number of map nodes, because then the data samples are distributed more sparsely on the map. Because of the tradeoff between vector quantisation and projection properties of the SOM, changing the training process such that the quantisation error is lowered usually leads to distortion of the map's topology.

1.3.2 Weighted distortion measure

The weighted distortion measure can be used to compute an error value for the whole map. Averaging the quantisation error over all data leads to the distortion or intraclass sum of squares (which are different names for the same error, used respectively in the information theory domain and by statisticians). The distances are weighted by the neighborhood function, since neurons close to the BMU play a more important part in the calculation:

$$E_d = \frac{1}{N} \cdot \sum_{i=1}^{N} \sum_{k=1}^{K} h_{c_i,k} \cdot ||\mathbf{x}_i - \mathbf{w}_k||^2 \tag{1.10}$$

where N is the number of inputs, K is the number of neurons, c_i is the best matching unit (BMU) of input $\mathbf{x}_i$ and $h_{c_i,k}$ is the neighborhood function calculated evaluating the distance between the BMU and neuron k on the grid, the neighborhood radius value has to be chosen in advance to calculate the weighted distortion measure.

This measure has been proven to be the local energy function for SOM in the case of a discrete data set and fixed neighborhood function [Kohonen, 1991]. Vesanto et al. [2003]showed that it can be decomposed into several components that evaluate the quantisation error and topological preservation separately. Therefore, this measure is able to evaluate the overall quality of a SOM.

1.3.3 Topographic error

Unlike the two previous measures, assess the topographical order means defining an index to evaluate if the map is topologically ordered, then check whether similar code vectors belong to neurons arranged in adjacent positions on the grid and vice versa.

The topographic error is the most simple of the topology preservation measures. For the whole dataset, the respective best and second-best matching units are determined. If these are not adjacent on the map grid, this is considered an error Kohonen [2001]. The total error is then normalised to a range form $[0; 1]$, where 0 means perfect topology preservation.

The topographic error is then defined as follows:

$$E_t = \frac{1}{N} \cdot \sum_{i=1}^{N} u(\mathbf{x}_i) \tag{1.11}$$

where N is the number of input vectors (the dataset size) and $u(\mathbf{x}_i)$ is a function so defined:

$$u(\mathbf{x}_i) = \begin{cases} 1 & \text{if } BMU_1(x_i) \,\text{and}\, BMU_2(x_i) \ \text{ are not closer on the map} \\ 0 & \text{otherwise} \end{cases}$$

The topographic error is affected by the choice of the neighborhood function and by its amplitude update during the SOM training, moreover the E_t value is related to the dataset heterogeneity and map dimension.

This kind of topographic error is called Forward Projection Error and it appears when a couple of similar input vectors $\mathbf{x}_l$ and $\mathbf{x}_m$ that are closed in the input space, are allocated to a couple of neurons $\mathbf{w}_s$ e $\mathbf{w}_t$ that are not closed on the map. Thus two or more distinct clusters on the map should merge in one, since the similarity of their data.

There exists a second type of error called Backward Projection Error and it appears when a couple of neurons $\mathbf{w}_s$ e $\mathbf{w}_t$, that are closed on the grid, are the image of a couple of input vectors $\mathbf{x}_l$ e $\mathbf{x}_m$ that are rather dissimilar and then they stay not closed in the input space. So happens that dissimilar data are projected into a unique cluster.

The concept of topographic error has a reference to the error that occurs when an embedding is attempted in a too low dimension in respect to the input space dimension (for a full discussion, please refer to Barazza [2004]). The False Nearest Neighbor method (FNN) can be used to find the lower embedding dimension [2] sufficient to reconstruct a n-dimensional

[2] In mathematics the set X is "*embedded*" of the set Y when exist an injective function $f : X \to Y$ and it preserves the topology, so a couple of entities that are near (distant) in the first set, are near (distant) in the second set too. This kind of function is then a morphism. Some examples of embedding is the existing relation between natural numbers and whole numbers, between whole numbers and rational numbers, between rational numbers and real numbers and between real numbers and complex numbers.

space in a m-dimensional space ($m < n$) preserving the topological properties of the n-dimensional space. Two entities are called false nearest neighbor when they are distant in the higher space and their projection on the lower space makes them neighbors.

Increasing the embedding dimension from m to $m+1$ the false nearest neighbor are found out, they corresponds to that couple of entities that are not more near using the embedding dimension $m+1$. Similarly, the percentage of FNN, for a given embedding dimension m, can be interpreted as the error that is committed using a space of this size for the reconstruction of the original n-dimensional space [Barazza, 2004].

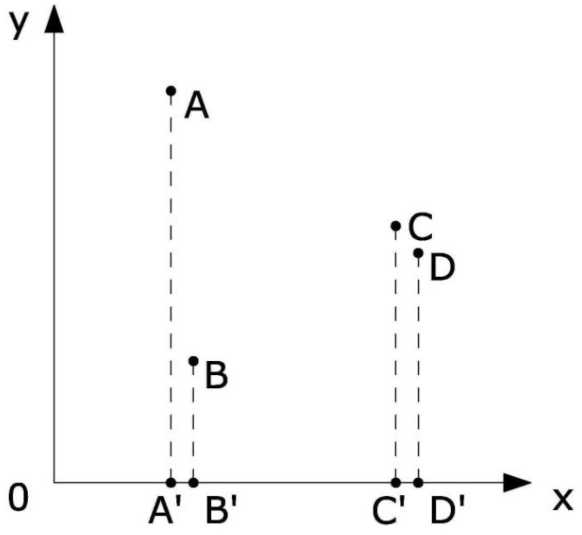


Figure 1.8: Figure shows an example of true neighbors (C and D) and a couple of false neighbors (A and B) projecting the two-dimensional space in a mono-dimensional space.

It seems that the hexagonal maps preserve the topology respect to this error better than the rectangular ones. This observation arises from a peculiarity of the topographic error measure rather than a better organisation of hexagonal maps respect to rectangular ones [Uriarte and Martín, 2006]. In fact the topographic error in rectangular maps increases due to nearby diagonal units. Although the topographic error does not consider the diagonal neuron neighbours, it does not really mean that the map does not conserve the local relations. It could be said that even if diagonal units are not properly neighbours, they are “special no neighbours”, but the topographic error does not make any difference between different kinds of adjacent units. This could be one of the reasons why this error devaluates the rectangular maps.

1.4 Map visualisation

SOMs can be used in two different manners. Neurons can be identified with clusters in the data space as it happens in k-means.
Otherwise the map space is regarded as a tool for the projection and visualisation of the high dimensional data space, with peculiarity that these kind of maps are composed by a large number of neurons, consisting of thousands neurons. Such SOMs allow the emergence of intrinsic structural features of the data space. These SOMs are self-organised projections of high dimensional data onto a two dimensional map, preserving topological relationships of the input space. The U-matrix (Unified distance matrix) is the canonical tool for the display of the distance structures of the input data on the SOM map [Ultsch, 1992].

Moreover the U-matrix can help to understand if the map have a suitable organisation for data projection, since the U-matrix shows the SOM map partition in clusters.

Following the original idea of Ultsch and Siemon [1990], the U-matrix is constructed on top of the map and has the aim to identify clusters measuring the similarity of each couple of code vectors associated with neurons that are neighboring on the grid. The SOM map visualisation becomes easy and intuitive, with a modest computational effort even with a high number of neurons.

Let n be a neuron on the map and $N_{neigh}(n)$ be the set of immediate neighbors on the map, $\mathbf{w}_n$ the weight vector associated with neuron n, then :

$$U_{height}(n) = \sum_{m \in N_{neigh}(n)} d[\mathbf{w}_n - \mathbf{w}_m] \tag{1.12}$$

where $d[\mathbf{w}_n - \mathbf{w}_m]$ is the same kind of distance used in the SOM algorithm to construct the map. The vectors $\mathbf{w}_n$ and $\mathbf{w}_m$ are the code vectors of the neuron n and m that are neighboring on the grid.

The U-matrix is a display of the U-heights on top of the grid positions of the neurons on the map. An U-matrix is usually displayed as a grey level picture or as three dimensional landscape. On a good quality map dark areas (transition zones) should separate, as far as possible, the light areas (clusters). Similarly, using a three-dimensional map, ridges (transition zones) should separate valleys (clusters).

The equation 1.12 calculates the sum of the distances between the neuron n and the set $N_{neigh}(n)$ composed by its neighbors. In this work the U-matrix has been improved to show the distance between the neuron n and each of its neighbouring neurons. So the cell corresponding to the neuron n show by a number and a circle the amount of input vectors projected in that neuron, while the cells around are coloured in gray scale to show the distances between n and the neighbouring neurons. Each one cell assumes a gray tonality in accordance to the singular distance:

$$U_{height}(n; m) = d[\mathbf{w}_n - \mathbf{w}_m]$$

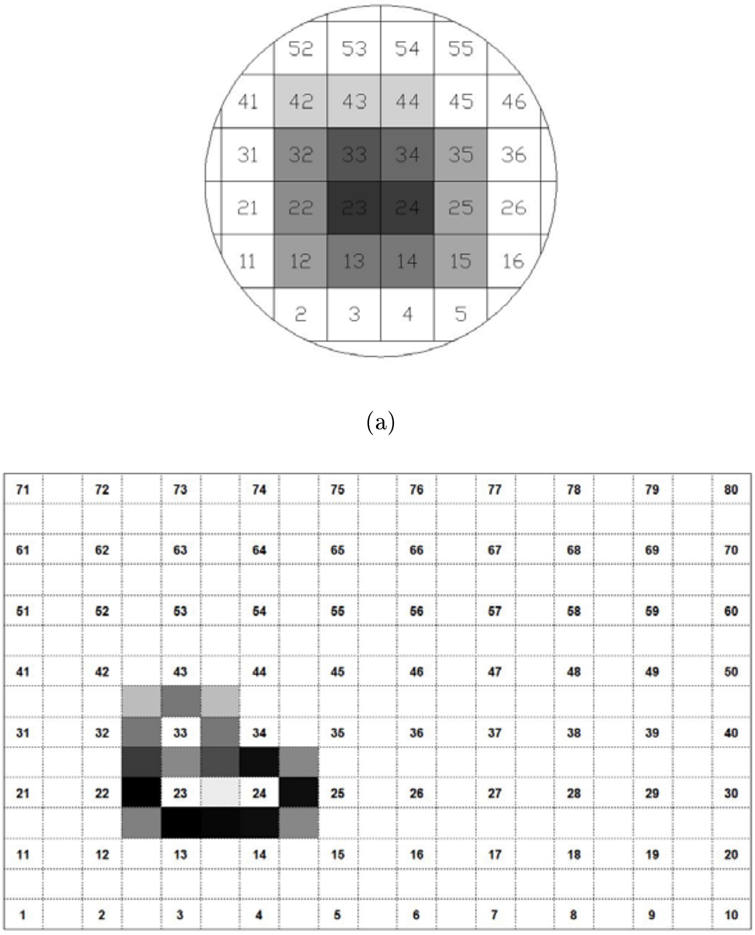
where $m \in N_{neigh}(n)$.

For illustrative purposes let M be a rectangular matrix of dimensions $10x8$ and each one of its 80 code vectors it's a vector of length 4 so defined $w_j = [1, 1, 1, 1]$, except the code vectors of the neurons 23, 24 and 33 that are so defined:

$$\begin{bmatrix} w_{23} \\ w_{24} \\ w_{33} \end{bmatrix} = \begin{bmatrix} 7.3 & 7.6 & 7.7 & 8.0 \\ 7.4 & 7.2 & 7.3 & 7.2 \\ 4.1 & 4.6 & 4.6 & 4.8 \end{bmatrix}$$

The distances matrix M_d is built using the same kind of distance used in the SOM algorithm at each couple of code vectors of neighboring neurons on the grid. In the case study exemplify the follow matrix is achieved:

$$M_d = \begin{array}{cccccccccc}
 & w_1 & \dots & w_{23} & w_{24} & \dots & w_{33} & \dots & w_{80} & \\
\left[\right. & 0 & \dots & & & \dots & & \dots & & \left.\right] w_1 \\
 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
 & 13.31 & \dots & 0 & & \dots & & \dots & & w_{23} \\
 & 12.55 & \dots & 0.98 & 0 & \dots & & \dots & & w_{24} \\
 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
 & 7.07 & \dots & 6.25 & 5.54 & \dots & 0 & \dots & & w_{33} \\
 & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\
\left[\right. & 0 & \dots & 13.31 & 12.55 & \dots & 7.07 & \dots & 0 & \left.\right] w_{80}
\end{array}$$



(a)

(b)

Figure 1.9: In fig. *1.9a* is shown the "classic" U-matrix, gray tone of each cell that contains a neuron is in reliance on the average dissimilarity from the neuron's code vector to the code vectors of the neighboring neurons. The more the dissimilarity between code vectors is higher, the more is dark the gray tone of the cell. In fig. 1.9b is shown the entire U-matrix using the improves suggested in this work, for the same example considered in fig. 1.9a. Each cell coloured in gray tone shows the dissimilarity level between a couple of neurons, the cells placed on the diagonals show the average dissimilarity of the two couples of neurons.

Both the U-matrices, the "classic" one and the one obtained by the introduced improvements, are displayed in fig. 1.9. In both figures are detectable three neurons whose weight vectors are easily distinguished from the vectors of the weights of the other units of the grid. Moreover is instantly displayed also the relationships of similarity between each pair of adjacent neurons.

An U-matrix displays the local distance structure of the data set.

The U-matrix delivers a "landscape" of the distance relationships of the input data in the data space. Properties of the U-matrix are (taken from Ultsch [2003]):

- the position of the projections of the input data points reflect the topology of the input space, this is inherited from the underlying SOM algorithm;
- weight vectors of neurons with large U-heights are very distant from other vectors in the data space;
- weight vectors of neurons with small U-heights are surrounded by other vectors in the data space;
- projections of the input data points are typically found in depressions;
- outliers in the input space are found in "funnels";
- "mountain ranges" on a U-matrix point to cluster boundaries;
- "valleys" on a U-matrix point to cluster centers.

The U-Matrix realises the emergence of structural features of the distances within the data space. Outliers, as well as possible cluster structures can be recognised for high dimensional data spaces. The proper setting and functioning of the SOM algorithm on the input data can also be visually checked.

A further example of the SOM process application to a simple dataset and the related U-matrix visualisation of the SOM map is shown below.

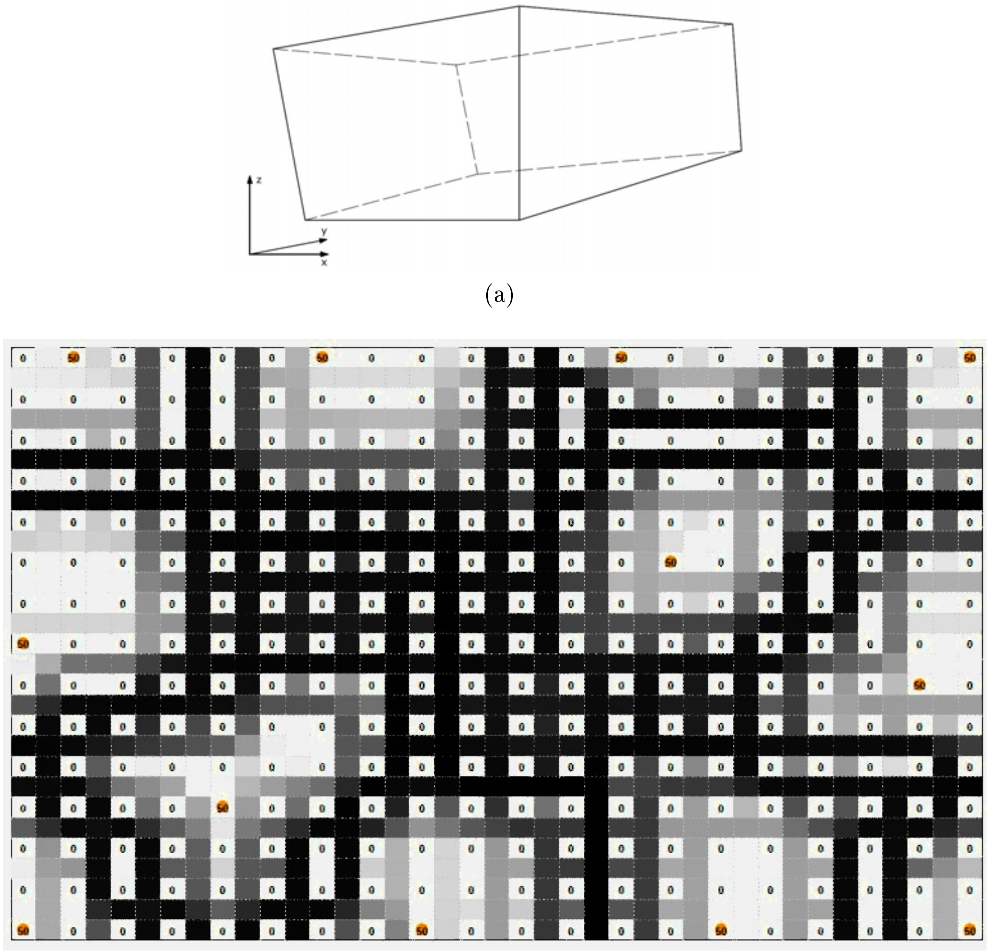
Fig. *1.10a* shows a solid in the three-dimensional space, it's composed by 12 edges with different orientation in the 3D space. The orientation of each one edge can be described by the direction cosines, then a SOM map has been trained using these vectors of length equal to three and the dataset is composed by these vectors repeated each one 50 times. In fig. 1.10b the U-matrix shows the SOM trained map, is easy to notice the 12 clusters, equals to the number of the edges of the solid.

1.5 Correlation as discriminant function

In general, clustering algorithms are used to group some given objects defined by a set of numerical properties in such a way that the objects within a group are more similar than the objects in different groups. Therefore a particular clustering algorithm needs to be given a criterion to measure the similarity of objects, how to cluster the objects into groups.

SOM process is entirely based on similarity measure between input vectors and code vectors. Moreover the U-matrix is based on similarity measure between code vectors of the neighboring neurons. It follows that a proper choice of the discriminant function is fundamental to run a capable SOM algorithm.

The Euclidean distance is the most popular choice, but it's not always the best choice. It takes into account the difference between two samples directly, based on the magnitude of changes in the sample levels. This distance type is usually used for data sets that



(a)

(b)

Figure 1.10: In fig. 1.10a the 12 edges solid is shown. Each one edge is described by its three direction cosins, the dataset is built repeating 50 times each one of the 12 vectors of length three. In fig. 1.10b the SOM map and the obtained dataset clustering are shown by the improved U-matrix representation.

are suitably normalised or without any special distribution problem, since this distance measure suffers from a high sensitivity to outliers.

Correlation tends to detect the difference in shapes, rather than to determine the magnitude of differences between two objects.

Correlation between variables is a measure of how well the variables are related. The correlation coefficient ranges from -1 to 1. A value of 1 implies that a linear equation describes the relationship between the variables perfectly, with all data points lying on a line for which one variable increases as the other one increases. A value of -1 implies that all data points lie on a line for which one variable decreases as the other one increases. A value of 0 implies that there is no linear correlation between the variables.

Data dependencies are often described by values of Pearson correlation, since this measure is closely connected to linear regression analysis via the residual sum of squares to the fitted line. Pearson correlation describes the degree of linear dependence of vectors **x** and **w** by:

$$r(\mathbf{x}_i, \mathbf{w}_j) = \frac{\sum_{m=1}^{n}[(x_{i,m} - \mu_x) \cdot (w_{j,m} - \mu_w)]}{\sqrt{[\sum_{m=1}^{n}(x_{i,m} - \mu_x)^2] \cdot [\sum_{m=1}^{n}(w_{j,m} - \mu_w)^2]}} \tag{1.13}$$

where, looking at the SOM process, $\mathbf{x}_i(t) = [x_{i,1}(t),\, x_{i,2}(t) \dots x_{i,n}(t)]$ is an input vector and $\mathbf{w}_j(t) = [w_{j,1}(t),\, w_{j,2}(t) \dots w_{j,n}(t)]$ is the code vector of one of the neurons. Data standardisation makes Pearson correlation invariant to rescalings of whole data vectors by common multiplication factors and to additive component offsets. In other words, the favorable invariance feature of Pearson correlation results from implicit data normalisation realised by equation 1.13.

There is a relation between the Euclidean distance and the Pearson correlation. The first step consists to express the correlation of the vectors $\mathbf{x}$ and $\mathbf{w}$ in terms of covariance using the scalar product $\langle \cdot, \cdot \rangle$, then it's necessary to discard the mean value of the vectors $\mathbf{x}$ and $\mathbf{w}$ and yield unit variance (z-score):

$$\mathbf{x}_i^z = (\mathbf{x}_i - \mu_x \cdot \mathbf{1})/\sqrt{\mathrm{var}(\mathbf{x}_i)}$$

$$\mathbf{w}_i^z = (\mathbf{w}_i - \mu_w \cdot \mathbf{1})/\sqrt{\mathrm{var}(\mathbf{w}_i)}$$

it follows the expression of the correlation of $\mathbf{x}$ and $\mathbf{w}$:

$$r(\mathbf{x}_i, \mathbf{w}_j) = \left\langle \mathbf{x}_i^z, \mathbf{w}_j^z \right\rangle /(n-1) \tag{1.14}$$

where:

$$\left\langle \mathbf{x}_i^z, \mathbf{w}_j^z \right\rangle = \sum_{m=1}^{n} (x_m^z \cdot w_m^z) \tag{1.15}$$

since $r(\mathbf{x}_i, \mathbf{w}_j) = r(\mathbf{x}_i^z, \mathbf{w}_j^z)$. When this notation is applied to the squared Euclidean distance of $\mathbf{x}_i^z$ and $\mathbf{w}_j^z$ this yields:

$$d^2(\mathbf{x}_i^z, \mathbf{w}_j^z) = \sum_{m=1}^{n} (x_m^z - w_m^z)^2 = \langle \mathbf{x}_i^z, \mathbf{x}_i^z \rangle - 2 \cdot \left\langle \mathbf{x}_i^z, \mathbf{w}_j^z \right\rangle + \left\langle \mathbf{w}_j^z, \mathbf{w}_j^z \right\rangle = 2 \cdot (n-1) \cdot (1 - r(\mathbf{x}_i, \mathbf{w}_j))$$

since $\langle \mathbf{x}_i^z, \mathbf{x}_i^z \rangle = \left\langle \mathbf{w}_j^z, \mathbf{w}_j^z \right\rangle = 1$.

Thus, when data are normalised to discard the mean value and yield unit variance, Pearson correlation can be easily expressed as distance the square of Euclidean distance. However the normalisation is a crucial step. In optimisations operating on dynamic data, static pre-computation by the z-score transform is not available for computational improvements over equation 1.13 [Strickert and Seiffert, 2007].

It has been demonstrated [de Gelder et al., 2001] that various similarity criteria of two functions, $f(y)$ and $g(y)$, including the sum of squared differences, the correlation coefficient and the overlap integral, are related to the cross-correlation function $R_{f,g}(\tau)$ at $\tau = 0$.

$$R_{f,g}(\tau) = \int f(y)\, g(y+\tau)\, dy$$

Thus, they cannot provide any information about patterns that are shifted relative to each other. de Gelder et al. [2001] proposed then a generalised expression for similarity, $S_{f,g}$, which is based on a weighted cross-correlation function, a weighting function $z(\tau)$ normalised with the product of the two weighted autocorrelation functions:

$$S_{f,g} = \frac{\int z(\tau)\, R_{f,g}(\tau)\, d\tau}{\sqrt{\int z(\tau)\, R_{f,f}(\tau)\, d\tau \quad \int z(\tau)\, R_{g,g}(\tau)\, d\tau}}$$

$z(\tau)$ is a triangular weighting function of width defined as $z(\tau) = 1 - {|\tau|}/{h}$ if $|\tau| < h$ and $z(\tau) = 0$ if $|\tau| \geq h$. Thus in SOM process the BMU is the neuron $w_{c(x)}$ that maximises the value of the function $S_{f,g}$.

1.6 Cluster recognition

A considerable improvement has been applied to the original algorithm (Carniel et al. 2009, 2012), it's aim at simplifying result presentation and interpretation, as proposed also by other authors (Vesanto et al. 2000, Messina and Langer 2011), introducing an automatic cluster recognition method.

After the data are projected onto the map, an automatic cluster analysis procedure is applied, again using a weighted cross-correlation to compare the code vectors of each pair of neurons. First the weighted cross-correlation matrix is calculated, then the hierarchical clustering performed by the *agnes* function (R environment, *cluster* package) is applied [Kaufman and Rousseeuw, 1990]. The clustering method that was selected within the *agnes* function is the "average" ([unweighted pair-]group average method, UPGMA). Each observation is thought to be a small "one-node" cluster, that is then progressively merged with similar alternative clusters, until only one large cluster remains which contains all the observations. At each stage the two most similar clusters are combined. The result is a figure called a dendrogram where the clustering procedure described above can be seen proceeding from bottom to top.

A few examples are shown in the chapters dedicated to the SOM approach applications.

It's good to take in mind that the clustering procedure applied to clustering detection does not act on the original data vectors, but on the code vectors, that are already "prototypes" of a class of data vectors, so this is not equivalent to a direct clustering of data but adds a further level of "abstraction" and should therefore better extract characteristic features of the data.

The clustering procedure can be interrupted at any vertical level by choosing an appropriate similarity threshold, which directly determines a certain number of clusters on the map to which nodes are assigned. The choice of the threshold is not straightforward and provides another degree of freedom for the fine tuning of the SOM analysis, because there isn't a single optimal threshold for any application or dataset.
Choosing a low threshold increases the number of clusters, and each of them becomes highly specialised to recognise a particular kind of pattern, so the SOM can recognise very subtle variations in the data. However, this can often lead to the assignment of data windows to clusters having lower coherence in time evolution, hence consistent regimes may not be recognised.
For this reason it is advised to carry out several analyses with different thresholds before choosing the threshold that best optimises time coherence while maintaining acceptable "resolution" in terms of distinction between different spectral patterns.

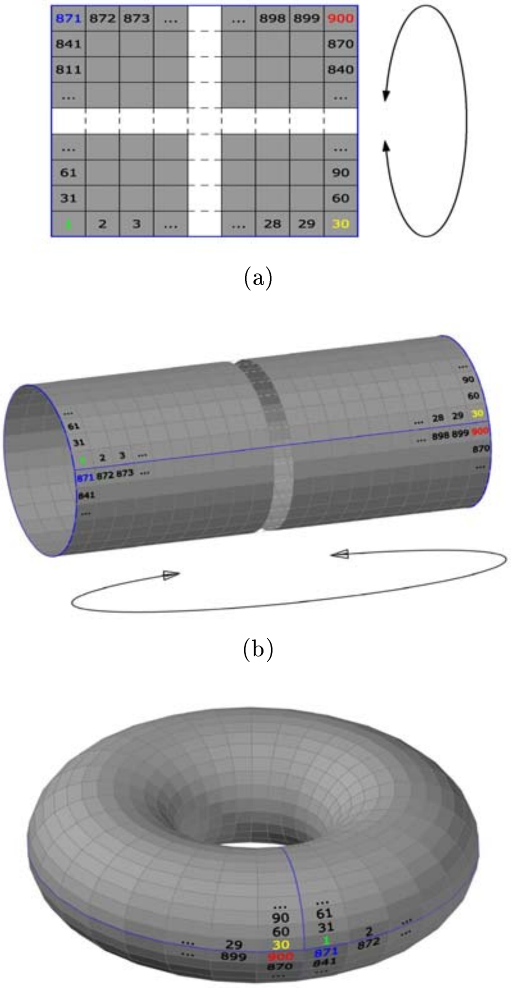
In order to better carry out these tests and present both test and final results in an easy to interpret way, clusters can be represented by different colours on the SOM and in the plots of time histories. At the chosen similarity threshold, the different clusters represent typical spectra with specific characteristics that can be distinguished from one another by the SOM.

1.7 Toroidal maps

The grid of neurons is usually embedded in a bi-dimensional space, it has the disadvantage that neurons at the borders of the map have very different mapping qualities than neurons in the center of the map. The reason for this is the different number of neighbors of center versus border neurons. This is important during the learning phase and structures the projection.

In many applications important clusters appear in the corners of such a planar map (see for example the previous fig.1.10). The embedding into a borderless grid, such as a torus, avoids such effects. All figures in the following are such tiled displays.

The solution adopted is then to remove the edges of the maps by transforming the flat maps into toroidal maps. This is done by virtually sticking together upper and lower edges vertically and left and right edges horizontally, without any data modification. The topological improvement is applied to the map lattice, not to the input vectors.



(a)

(b)

(c)

Figure 1.11: Toroidal SOM map visualisation. This kind of topology aims to remove the edges of the map, since neurons along edges of flat maps do not have the same number of neighbors as the others, resulting in an inhomogeneous training process. This is done by virtually sticking together upper and lower edges vertically and left and right edges horizontally, without any data modification.

Part II

Applications

Chapter 2

Volcanic regime detection

Since volcanoes can devastate and modify the scenery of wide areas, an eruption is likely to have a significant impact on population and anthropic activities. Volcanoes can remain in a quiescent state for a long period and the volcanic unrest is often prolonged over a period of months to years making the identification of precursory patterns of the uttermost importance.

The basic idea of this work is that if the volcanic system condition is related to some observed parameters and if the SOM process has produced a well organised map, with every cluster highlighting at least one distinctive feature with respect to the other clusters and clusters with similar features staying topologically close on the map, then the projection of the data onto the map, ordered by time, could detect possible modifications of the volcanic system condition.

2.1 Procedure

The parameter that has been analysed by the SOM process is the frequency content of the tremor acquired near a volcanic system. Signal acquisition has been carried out by a three components seismic single station. In the training phase volcanic tremor spectra are randomly presented to the network in a competitive iterative process. Spectra are then projected, ordered by time, onto the trained map. Every spectrum will take up a node on the map and the time evolution of their distribution on the map can highlight the existence of different regimes and the transitions between them.

Data processing described below have been carried out using acquired tremor spectrograms, a single input data for the SOM process is an amplitude spectrum with a given number of bins. Briefly a spectrogram is a visual representation of the frequency content of a signal, built from a sequence of spectra by stacking them together in time and by compressing the amplitude axis into a colour map. The final graph has usually time along the horizontal axis, frequency along the vertical axis, and the amplitude of the signal at any given time and frequency is shown as a colour level. A spectrogram is obtained splitting the acquired signal in temporal windows of a constant width, these windows can be overlapped by an interval shorter than the window width. An algorithm to compute the discrete Fourier transform to convert time to frequency is then applied at each temporal window. In this work the Fast Fourier Transform (FFT[1], Cooley and Tukey, 1965) has been applied in the

[1] The fast Fourier transform (FFT) is an optimised algorithm for the calculation of the Discrete Fourier

less recently analysis, where the objective was to observe the data distribution on the SOM map in terms of center of mass, largest node position and moment of inertia, the latter is calculated as:

$$M = \frac{1}{N}\sum_{i=1}^{N} N_{c_i} \cdot \sqrt{(x_{c_i} - x_{b_r})^2 + (y_{c_i} - y_{b_r})^2}$$

where N is the total number of input vectors projected onto the map at the iteration t, N_{c_i} is the number of input vectors projected onto the cell i of the map (with coordinates x_{c_i} and y_{c_i}) and the coordinates x_{br} and y_{br} define the position of the centre of the distribution. The moment of inertia assumes higher values when the vector projections fall farther from the centre of the distribution. On the contrary, a low value of the moment of inertia indicates a set of vectors very concentrated close to the centre of the distribution. This is physically analogous to the difference between an object whose mass is spread across a wide volume and one where the mass is mostly concentrated close to the barycentre.

In the most recent SOM analysis, where the objective was to obtain a coherent data clustering to observe how data, ordered by time, move onto the map through clusters, the Welch's method[2] [Welch, 1967] has been applied to obtain a spectral density estimation.

Spectrogram normalisation helps to avoid that the more energetic windows, in which the time series has been splitted, hide the spectra calculated on the less energetic windows. Moreover normalisation makes the Short Time Fourier Transform (STFT) spectrogram obey Parseval's energy conservation property, meaning that the energy in the STFT spectrogram equals the energy in the original time domain signal[3]. During the data analysis it has been noted that the spectrum normalisation enhances the SOM capability in data clustering, taking into account the whole range of frequencies for each time window, more or less energetic. Similar results have been obtained normalising the spectrum with respect to the sum of their amplitude values as with respect to their maximum amplitude value.

The SOM map training is achieved following the process described in section 1.2: a certain number of spectra, equal to the number of SOM map neurons, are sorted randomly from the whole available data set to complete the map training. Once the map is initialised, at each SOM process iteration the whole data set is given in input to the map, each spectrum

Transform (DFT) and its inverse (the inverse Fourier transform). The FFT is suitable to compute a DFT of time windows whose length is a power of 2 ($N = 2^n$). Many software and hardware (dedicated microcircuit) are available for its calculation. The strategy on which the FFT algorithms are based on is known as divide and conquer, the calculation of a DFT on windows of length N is reduced to compute a DFT on shorter windows (decreasing the number of operations needed to compute the DFT).

[2] The Welch's method consists in splitting the time series into shorter time windows with possible overlapping, computing the Power Spectral Density (PSD) of each window, and then averaging the PSD estimates. The averaging of PSD tends to decrease the variance of the estimate compared to a single PSD estimate on the entire dataset, although overlap between segments tends to introduce redundant information. This effect is reduced using of a non-rectangular window, which reduces the importance given to the end extremities of the overlapped segments with a Hamming window being a common choice. However the combined use of short data records and non-rectangular windows results in reduced resolution of the estimator, i.e. there is a tradeoff between variance reduction and resolution, but this variance reduction can be seen as a filter for the noise at high frequency.

[3] For periodic signals, Parseval's theorem shows how the signal's power is distributed among the harmonic components. Similarly, for aperiodic signals of finite energy a Parseval's inference indicates how the square of the Fourier transform is an energy density revealing the amount of energy at each of the frequencies. The plot of the square of the Fourier transform is the so called "energy spectrum" of the finite energy signal and it displays how the energy is distributed over frequency.

only one time and in random order. At each step the neighborhood function becomes narrowest and the learning rate value decreases. At the end of the training phase the SOM map is ready for the data projection phase. Each spectrum, that has contributed or not to the training phase, is then projected onto the map, ordered by time.

Once the SOM map is trained and the data are projected on the Kohonen grid output space, it is necessary to analyse data distribution on the bi-dimensional grid. Two main ways has been taken into account in this work.

The first one deals with the map as a bi-dimensional projection of a surface that is stretched into the n-dimensional input space to approximate the data distribution. The original idea is to find some useful parameter able to describe how the spectra corresponding to a selected time window have been projected among the neurons of the map. For this purpose the position on the map of the center of mass of the distribution, its moment of inertia and the position of the neuron with the largest number of projected input are computed. Data analysis is then performed projecting on the map groups of data of constant dimension (a certain constant number of temporally consecutive spectra) corresponding to a few minutes of acquired signal. For each one of these groups the parameters mentioned above are calculated to detect in which area of the map data are located and to obtain a measure of how much they are scattered. These three parameters allow to overcome one of the greatest issues of this in-time analysis, the large number of maps that have to be analysed and the consequent difficulty to have an overall view of the data set was a possible limit of the post-processing phase, it's achieved by summarising results by that three key variables over time.

The second way to analyse the data distribution starts from an automatic cluster recognition. The dendrogram (see section 1.6) is calculated starting from the weighted cross-correlation matrix computed among the whole code vectors set. Each code vector is thought to be a small "one-node" cluster, that is then progressively merged with similar alternative clusters, until only one large cluster remains which contains all the code vectors. At each stage the two most similar clusters are combined to obtain the so called dendrogram. Choosing a threshold for the similarity value, in relation of the chosen discriminant function properties, the dendrogram is horizontally cut and a certain number of clusters is detected on the map. Each spectrum is associated with one of the neurons on the map and then it's associated to one of the detected clusters. Is then possible to analyse data distribution looking at which cluster each spectrum, ordered by time, is associated to.

As outlined above, the data projection phase is conceived to analyse the temporal evolution of the volcanic system condition, monitoring the frequency content by the SOM pattern recognition capability. Each spectrum of a group, equivalent to a fixed time length window of the acquired signal, will take up a position on the map related with its frequency content and map topology. Since the code vectors are not modified, the map does not change its topology during the projection phase.

In order to keep the map unchanged during the projection phase, the neighbourhood function value and/or the learning-rate factor have to be fixed at the null value. For each group of spectra projected onto the trained map, the discriminant function calculates the dissimilarity between each input vector and the correspondent BMU code vector. If this value is low for each input vector of the whole data set, then the map is well organised, and it is possible to explore the whole input space. Moreover when data that have not contributed to the training phase are projected onto the map, some of these input vectors can have a great dissimilarity value from their BMUs. This occurrence highlights singular data, potentially related to unusual conditions of the volcanic system, which cannot find a suitable strong similarity to a code vector on the map.

The possibility to project on the map the data that have not contributed to the training phase, keeping the map unchanged until the projection errors are moderate, allows to carry out a real-time analysis, projecting the new spectra as soon as the signal is acquired and pre-processed. This property of the proposed SOM process allows to highlight the existence of dynamic regimes that are consistent and coherent over time, identifying possible precursors of paroxysmal phases of the volcanic system.

Figure 2.1 shows the flow diagram of the proposed SOM process. The first step concerns the pre-processing phase, so the spectrogram is calculated from the acquired signal and spectra are eventually normalised and/or smoothed (see section [sub:SOMforRuapehu]) and if necessary cut in the frequency range of interest. The next steps are the SOM map training phase and the data projection phase. The post-processing phase concerns the analysis of the distribution of the projected spectra onto the trained SOM map. Data distribution is described by geometrical parameters as the center of mass, the moment of inertia and the position of the neuron with the largest number of projected input or by a clustering analysis.

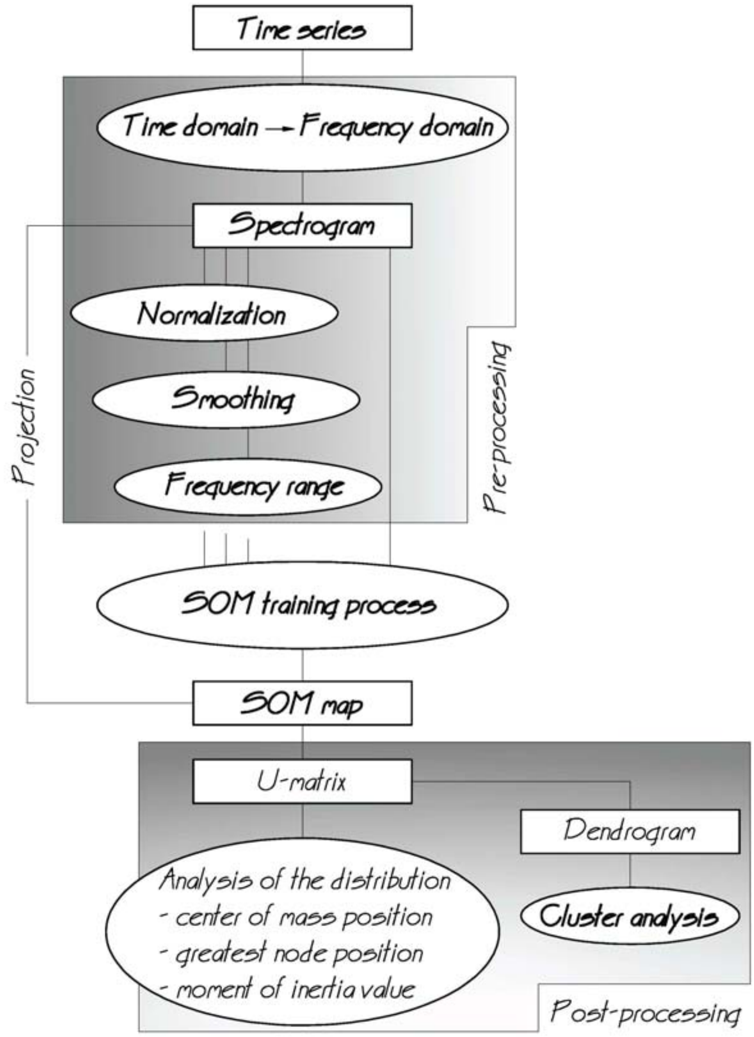


Figure 2.1: Flow diagram of the proposed methodology for the data analysis.

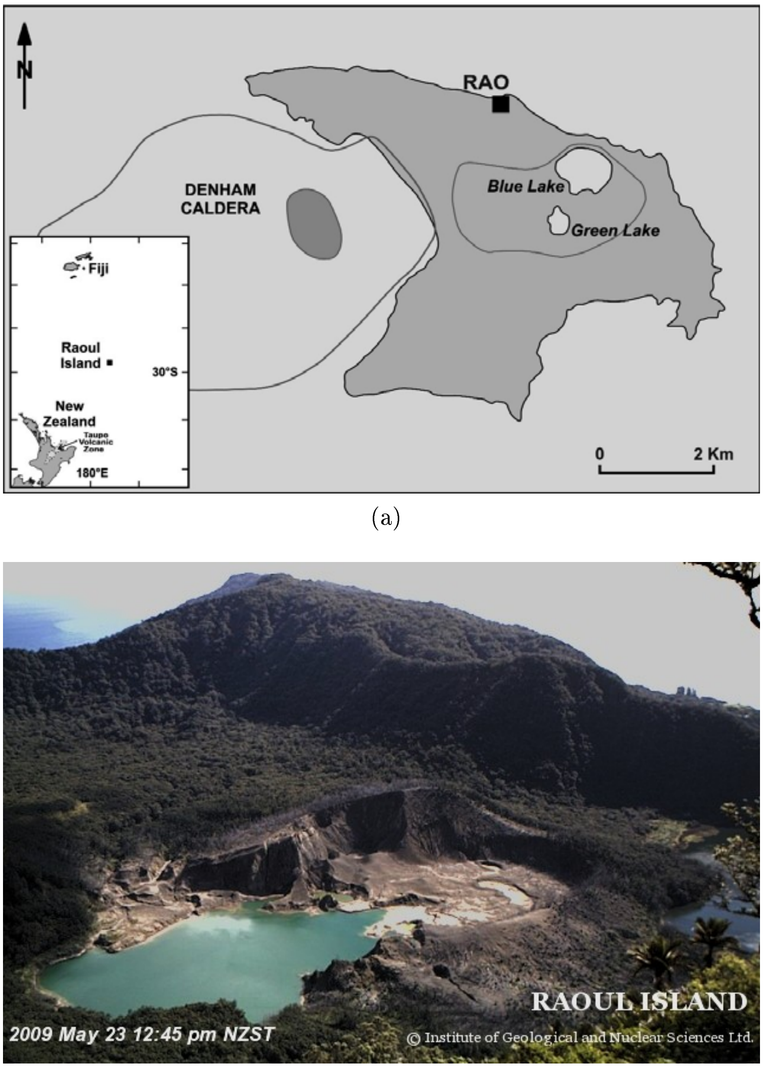
The outlined method has been applied to analyse the tremor acquired in proximity of the Green lake, one of two tiny crater lakes on Raoul Island in the Kermadec Islands (New Zealand), and the tremor acquired by a seismic station placed not so far from the Crater lake of Ruapehu volcano (Taupo volcanic zone, New Zealand).

The study of the volcanic regimes of Raoul is a part of the research program funded by PRIN 2007PTRC4C_002 at the University of Udine (R. Carniel and L. Barbui) and New Zealand Ministry of Science and Innovation (MSI) funding (A.D. Jolly). The EQC funded GeoNet Project provided data used in the project.

The EQC funded GeoNet Project and the New Zealand Volcano Database (VoD) provided data used for Ruapehu volcanic tremor analysis, while the EQC Project "Application of speaker recognition methods to volcanic systems" funded the specific application of SOMs presented here.

2.2 Raoul Island

Raoul Island is the largest and northernmost of the Kermadec Islands, it's far away 900 km from 'Ata Island of Tonga in SSW direction and 1100 km from New Zealand's North Island, in NNE direction. Raoul volcanic system has been the source of vigorous volcanic activity during the past several thousand years that was dominated by dacitic[4] explosive eruptions.



(a)

(b)

Figure 2.2: Fig. 2.2a shows the location map of Raoul Island, showing position of seismic station RAO and major volcanic features. Fig. 2.2b shows a photo of the Green Lake located in the center of Raoul Island, it has been taken from the website *http://www.geonet.org.nz/*.

Of the two calderas that belong to the Raoul volcanic system (fig. 2.2a) the oldest one (dating back to the Holocene[5]) is positioned in the center of the island, it measures about

[4] Dacite is an igneous volcanic rock and usually forms as an intrusive rock such as a dike or sill. Because of the moderate silica content, dacitic magma is quite viscous and therefore prone to explosive eruption (as occurred for example at Mount St. Helens in which dacite domes formed and exploded).

[5] The Holocene is a geological epoch which began at the end of the Pleistocene (about 11000 years ago)

3.5 km by 2.5 km and it forms the stratovolcano Raoul. The most recent caldera is located in the bay of Denham and was formed about 2200 years ago, following a violent dacitic volcanic explosion. It measures about 6.5 km to 4 km wide, with the major axis parallel to the oceanic rift[6] called Havre Trough. During the nineteenth and twentieth century, some eruptions on the island of Raoul affected simultaneously both calderas and consisted of phreatic eruptions classified as small to moderate in intensity. Some of these eruptions have seen the consequent formation of temporary islands in Denham caldera.

Raoul Island (fig. 2.2) provides a useful case study of the proposed SOM approach because volcanic activity was monitored by a single seismic sensor and activity at the volcano includes a range of volcanic activity including large magmatic eruptions [Healy et al., 1965, Lloyd and Nathan, 1981, Worthington et al., 1999, Smith et al., 2010], and small phreatic events [Christenson et al., 2007]. In addition, a small semi-permanent population of New Zealand Department of Conservation (DoC) staff is present. Hence, improvements in near-real time event discrimination may have positive benefits to the local population.

On March 17, 2006 (08:21 NZST) [March 16, 2006 (20.21 UT)], a small phreatic eruption occurred. The event was preceded by a swarm of small earthquakes located 10-20 km from the seismic sensor [Christenson et al., 2007] on March 12, decaying to background seismicity by March 16 NZST. The eruption seismic signal (fig. 2.3) was composed of several pulses and had a duration of 8 minutes [Christenson et al., 2007]. The spectrogram of the days from March 17, 2006 (NZST) to March 20, 2006 (NZST) is shown in fig. 2.4.

On March 16 (20.21 UT), the phreatic eruption was recorded on the local seismograph. Apart from the possible seismic precursors, no other significant changes that concern for example the water level in the lake or its temperature, even during the 24 hours before the eruptive event, were observed. According to the records of the seismometer installed on the island, the eruption seems to last longer than 30 minutes, even if the period of maximum intensity lasts about 5-10 minutes. After the explosion, the level of seismic activity has doubled, but starting from March 23, the number of earthquakes has been reduced to 10-20 per day. No thermal anomaly was recorded by the MODIS satellite system during the month of March.

Unfortunately, the eruption took the life of DoC ranger Mark Kearney, who was in the volcanic crater at the time of the eruption.

The goal of this analysis is to examine retrospectively the activity of this event and determine if improvements in seismic data processing and interpretation can be made.

In proximity of the Green lake have been noticed fumaroles of modest temperature (boiling point) and a slight seepage of surface brackish hydrothermal water (hot springs) along the beach of Oneraki, outside the caldera. The gases have a composition typically hydrothermal, which therefore does not suggest the presence of a magmatic source and rule out the possibility that exists an unique phase of the steam directly from the source magma to the surface, but rather suggest the presence of hydrothermal brackish water in depth. Observations on the eruptions that have affected the caldera of Raoul and the Bay Denham indicate the existence of an important active system located around the island.

and continues to the present. The Holocene is part of the Quaternary period.

[6] A rift is a linear zone where the Earth's crust and lithosphere are being pulled apart and is an example of extensional tectonics.

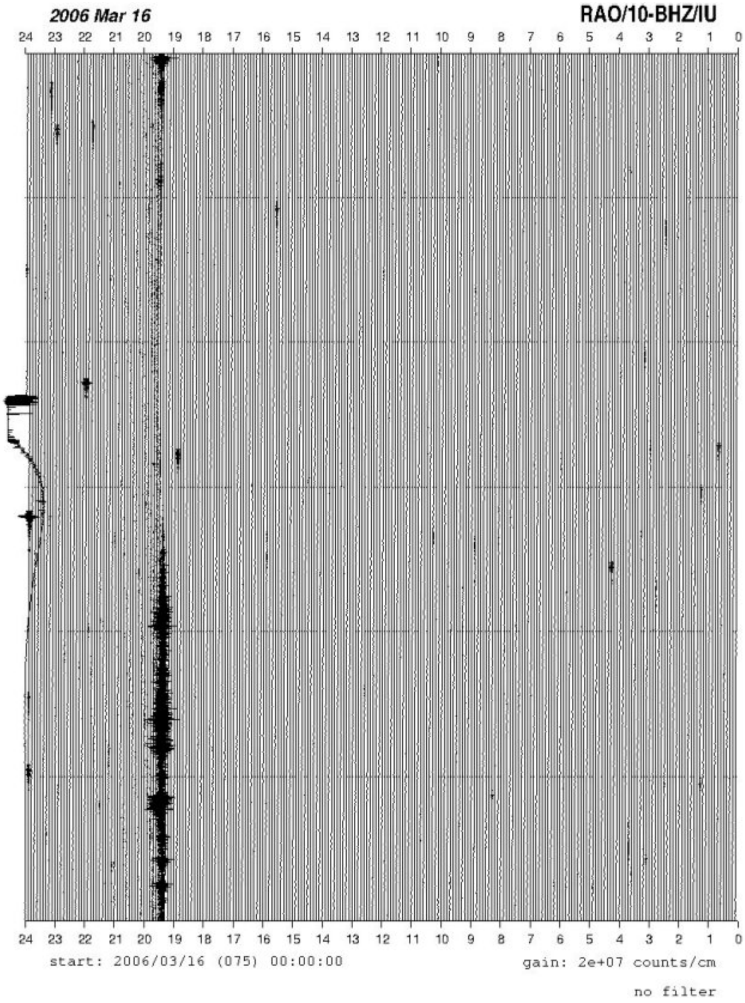


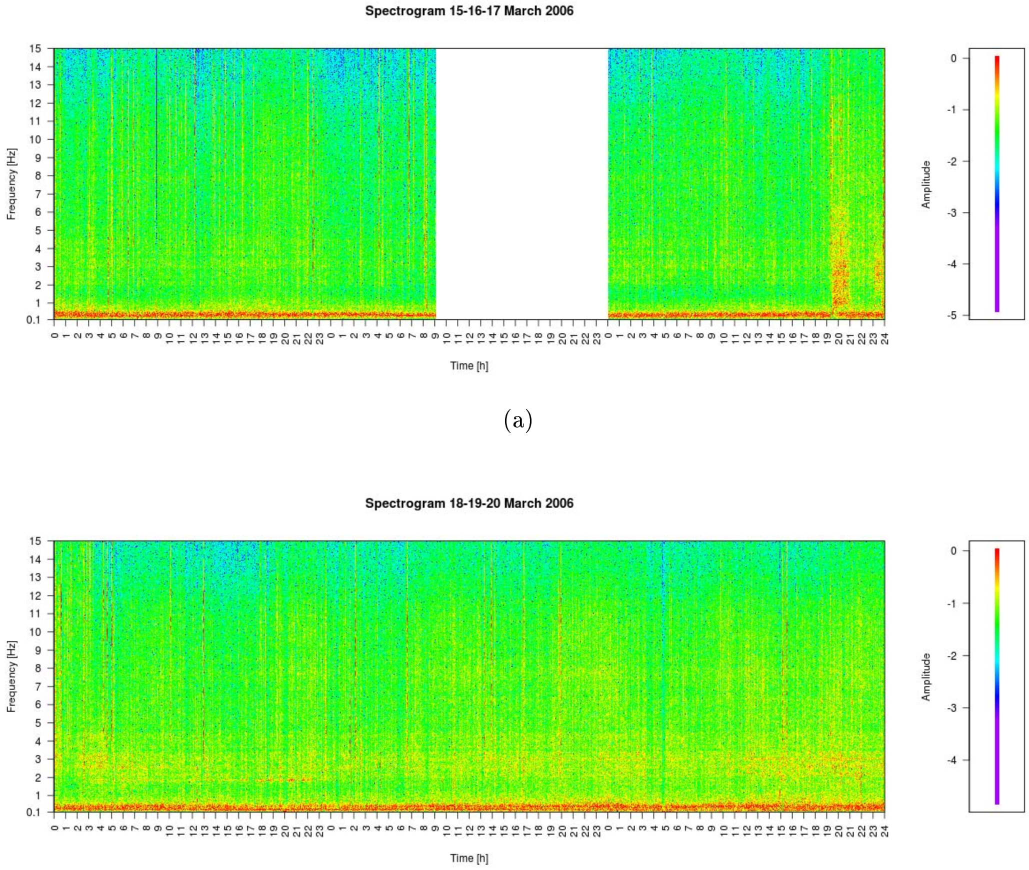
Figure 2.3: Original Helicorder recording acquired at seismic station RAO on Raoul Island, showing the pulsating eruption seismic signal on March 16, 2006 UT. Each line represents 6 minutes of data. The earthquake seen at the very end of the Helicorder recording is a tectonic event.

2.2.1 A SOM for Raoul

SOM algorithm has been applied to the tremor frequency spectra calculated from the acquired signal sampled at 40 Hz. Signal has been split in time windows of 512 points (corresponding to a time window of 12.8 seconds), with a 50% of overlap and Fast Fourier Transform (FFT) has been applied to each time window to calculate the corresponding spectrum in a frequency, the range of interest for the SOM analysis is up to 15 Hz.

Each spectrum is an input vector for the SOM analysis, the process is split in two phases: the training phase and the data projection phase. The training phase has been carried out using the spectra of the signal acquired during the March 16, 17, 18 and 19 (UT time), excluding the spectra corresponding to the volcanic eruptive events, because the SOM trained map must be able to recognise both the hypothetical dynamical regimes occured during the signal acquisition as well as every singular event that has not contributed to train the map, such as a volcanic explosion.

During the training phase the whole training data set, composed of about ten thousand spectra, is presented to the network 100 times. Spectra are normalised with respect to the sum of their own frequency values, so each spectrum has values in the range [0;1]. At



(a)

(b)

Figure 2.4: Spectrogram of the seismic data recorded at seismic station RAO on Raoul Island from March 15, 2006 to March 20, 2006 UT.

each iteration t, the network processes the training data set examining each input vector in random order. As the number of iterations t increases, the map topological order is improved. Obviously, this is a finite process and a large number of iterations causes an increase in the computation time, so, after many tests, a good compromise value for t has been found at 100 iterations.

Two opposed needs compete to choose the map size: a larger map increases the number of clusters that could arise; moreover a larger number of neurons generally increases the separation between these clusters and so the final resolution of the map. On the other hand, a large map decreases the SOM capability to provide an overview of the data set structure and increases the computational effort. A SOM map of dimensions 20x15 has been chosen.

In this case the SOM analysis has been carried out choosing the Euclidean distance as the discriminant function of the whole process. Fig. [fig:RaoulMeanDistance] shows the mean Euclidean distance value of each input vector to its closest unit on the map (the BMU) for each iteration (the whole data set is presented to the network 100 times) of the SOM process.

Over the course of the training process iterations the learning rate value has been decreased linearly from the starting value $\alpha_i = 0.1$ to the final value $\alpha_f = 0.001$. The neighborhood radius value decreases linearly from the starting value $\sigma_i = 10$ to the final value $\sigma_f = 0$ at an half of the chosen number of iterations (see the dotted vertical line in fig. 2.5) and it's

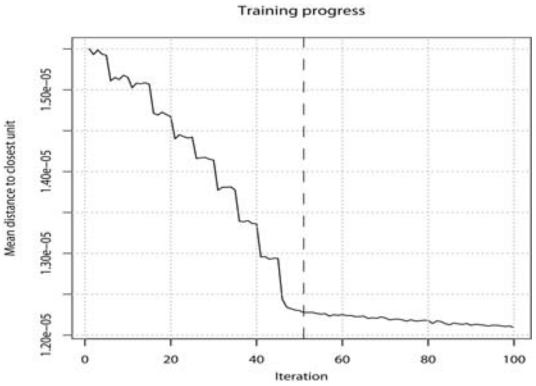


Figure 2.5: Trend of the averaged projection error calculated as the mean Euclidean distance between each input vector and its BMU code vector on the SOM map for each iteration of the training process. The x axis shows the number of iterations, each vertical leap of the graph before the vertical dotted line correspond to a learning rate value and neighborhood function width reduction. After the dotted line the graph shows the error reduction gained keeping the neighborhood function value equal to one (see 1.2.4), so each input vector modifies just its BMU code vector.

kept equal to zero till the last iteration. So during the second part of the training process each input vector modifies just its BMU code vector, neurons are no more affected by the mutual interactions, so the convergence to a minimum averaged error value is optimised.

At the end of the training phase the need to visualise the SOM map arises (see 1.4). The U-matrix visualisation method [Ultsch and Siemon, 1990] has been chosen and the code for the R software environment [Team, 2010] has been written (see A.6).

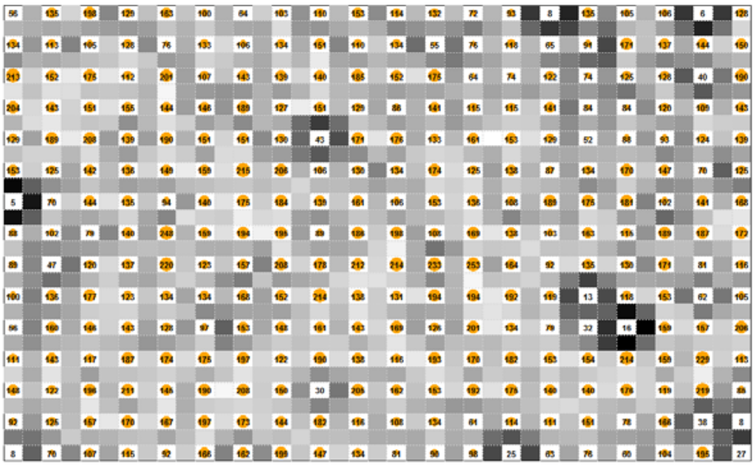


Figure 2.6: The SOM map obtained at the end of a training phase and visualised using the U-matrix method [Ultsch and Siemon, 1990]. At the last iteration of the SOM training process the whole data set is provided in input to the map and the code vectors are modified for the last time, preserving the information on how many and which spectra end up in each neuron.

Map nodes correspond to white cells, each node contains one neuron, the number and the circle dimension in the cell indicates how many input vectors have been mapped into the node. Cells which do not contain a number use a gray scale to show the similarity of the nearby neurons code vectors: light gray indicates a strong similarity and vice versa. In this way the light areas on the map indicate possible clusters consisting of more than one node (fig. 2.6).

2.2.2 Raoul data projection and interpretation

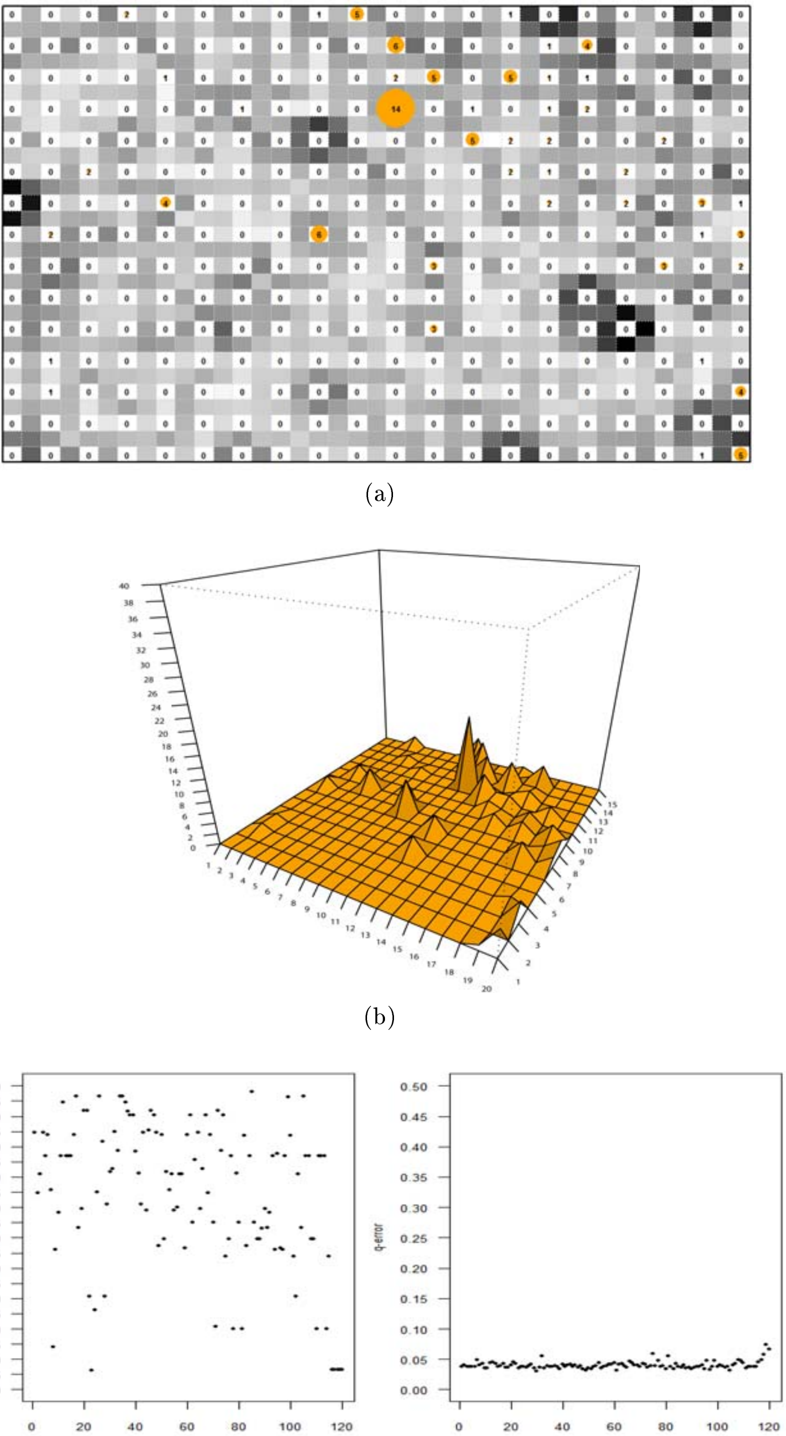
The data projection phase is conceived to analyse the temporal evolution of the volcanic system condition, monitoring the frequency content by the SOM pattern recognition capability. Each spectrum will take up a position on the map related with its frequency content and map topology. Since the code vectors are not modified, the map does not change its topology during the projection phase.

At each step of the projection phase, a group of 120 spectra (corresponding to a window length of about 13 minutes) is projected onto the map. In order to keep the map unchanged during the projection phase, the neighbourhood function value and/or the learning-rate factor have to be fixed at the null value. Fig. 2.7 shows an example of the projection of a group of spectra onto the trained SOM map.

As previously mentioned, one of the greatest issues of this analysis is the large number of maps that have to be analysed and the consequent difficulty to have an overall view of the data set. To avoid this problem the distributions property are summarised by three parameters, so each point (or pair of points) in fig. 2.8, 2.9, 2.10 and 2.11 show the position of the centre of mass, the position of the neuron with the largest number of input vectors and the value of the moment of inertia for one group of spectra. It is important to underline that the seismic tremor data acquired during March 16 has not been used in order to train the map.

It has been verified that the error defined as the Euclidean distance between each input vector and its BMU onto the map always stays low, this assures that the SOM training produced a well organised map, able to explore the input space data used during the training as well the data that did not aid the map topology organisation. Only after the explosion at the Green Lake the mean error shows a slight increase that lasts about one hour and a half (fig. 2.8). Considering the graphs of the day of the explosion (fig. 2.8) and the three days after (fig. 2.9, 2.10 and 2.11), it can be noticed that the trend on the position of the centre of mass, the position of the greatest node and the value of the moment of inertia for the groups from approximately the 20th to the 40th is considerably different to the mean trend. This means that the spectra of those groups have been projected into a restricted (looking at the moment of inertia graph) area onto the map, implying a strong coherency of such spectra, which show almost the same shape within each of those groups. Moreover, the area of the map involved is different from the “usual” areas, due to a different mean frequency content. This excursion occurred each day analysed and at the same time of the day, hence we interpret the excursion as resulting from an anthropogenic source. It’s important to note however that the island was evacuated on the day following the eruption. We surmise that the diurnal cycle for groups 20-40 represents some relict cultural noise effect, such as an automated scheduled activity, that persisted even after the human population departed. A natural source for the diurnal cycle is considered unlikely. Moreover, during the one hour and half after the explosion the data has been arranged into an area of the map that was previously unfilled. This transition can be interpreted as a result of the explosion and post explosion system excitation.

The time evolution of summary parameters of the SOM analysis, such as the centre of mass, the position of the neuron with the largest number of input vectors and the moment of inertia of the data distribution can provide information about the existence of relatively stable regimes and about the transitions between them by either slow or abrupt volcanic processes as expressed by transitions through the areas on the map.



(a)

(b)

(c)

Figure 2.7: Example of the projection onto the SOM map of a group of spectra. Each day is subdivided into 112 groups of the same size (120 spectra). Fig. 2.7a and 2.7b show the distribution of the group of spectra onto the map in two alterative ways (it can be noticed that the map is the same of that in fig. 2.6, cause it's not modified during the projection phase). Fig. 2.7c shows for each input vector (x axis) the corresponding neuron onto the map (y axis) and Euclidean distance value between the input vector and the code vector of the corresponding neuron.

Similar volcanic regime changes have been noticed for Ambrym Volcano (Vanuatu Islands) [Carniel et al., 2003]. In the seismic tremor acquired there during a one month period, two regimes were observed, characterised by durations of a few days, that differ in terms of frequency content and more generally in terms of volcanic activity with different levels of tremor energy, degassing processes and small explosions at the lava lake. A similar alternation of two main regimes has been observed at the Stromboli Volcano (Sicily, Italy) with time scales going from minutes to weeks [Carniel and Iacop, 1996, Ripepe et al., 2002]. The Erta Ale lava lake (Ethiopia, Africa) also showed a similar alternation of volcanic activity regimes [Harris et al., 2005, Jones et al., 2006].

At Raoul Island, we observe a transition in spectral characteristics at the onset of the eruptive activity into a spectral regime that lasted for approximately 1.5 hours. Christenson et al. [2007] surmised that the proximal cause of the eruption was due to the failure of a shallow hydrothermal seal which became pressurised by gas released from a deeper magmatic carapace. The carapace itself is surmised to have failed due to a swarm of hybrid and volcano-tectonic earthquakes [Lahr et al., 1994] occurring on March 12, which released gas from magma and caused pressurisation beneath the hydrothermal seal. If this model is correct, then the failure of the hydrothermal seal was instantaneous and included no precursors seen in either the observed spectra or the SOM analysis. However, the SOM highlights the post-failure system excitation which is possibly due to the re-equilibration of the hydrothermal system.

At Raoul Island, the SOM method allows recognition of diurnal pattern in seismic data that may be anthropogenic in nature, and are not readily apparent in visual spectrogram analysis. The SOM did not reveal precursors to the eruption in Raoul Island seismicity, but it highlighted the post-failure system excitation.

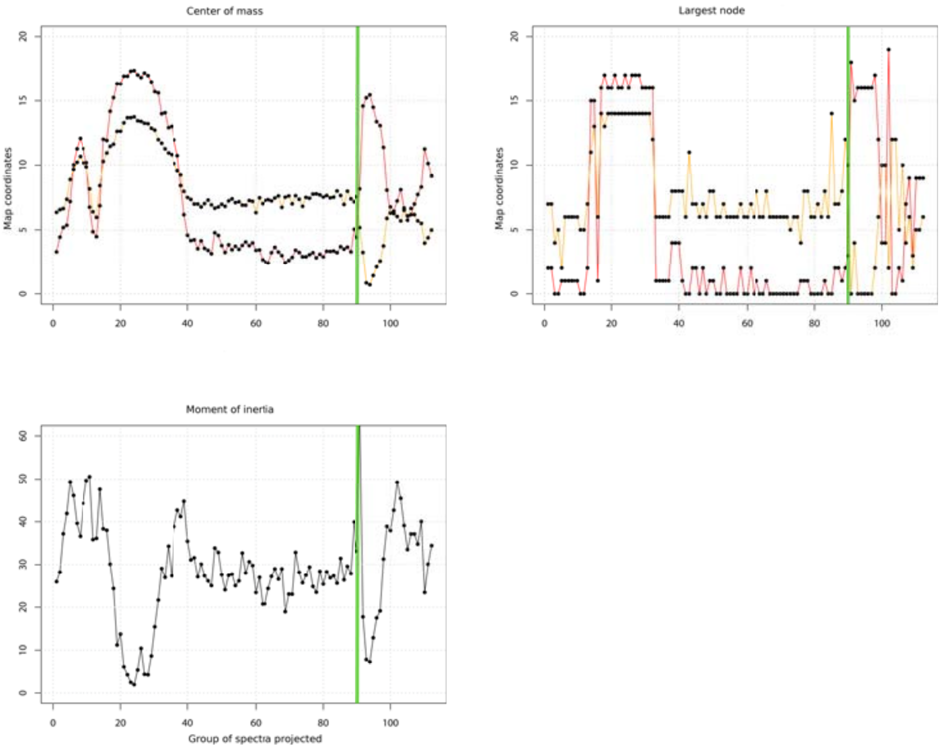


Figure 2.8: Raoul distribution analysis of the seismic tremor acquired during March 16 (UT time). As the map is bidimensional, two map coordinates, “x” and “y”, are needed to represent the position of a node on the map. The green line marks the explosion event at the Green Lake.

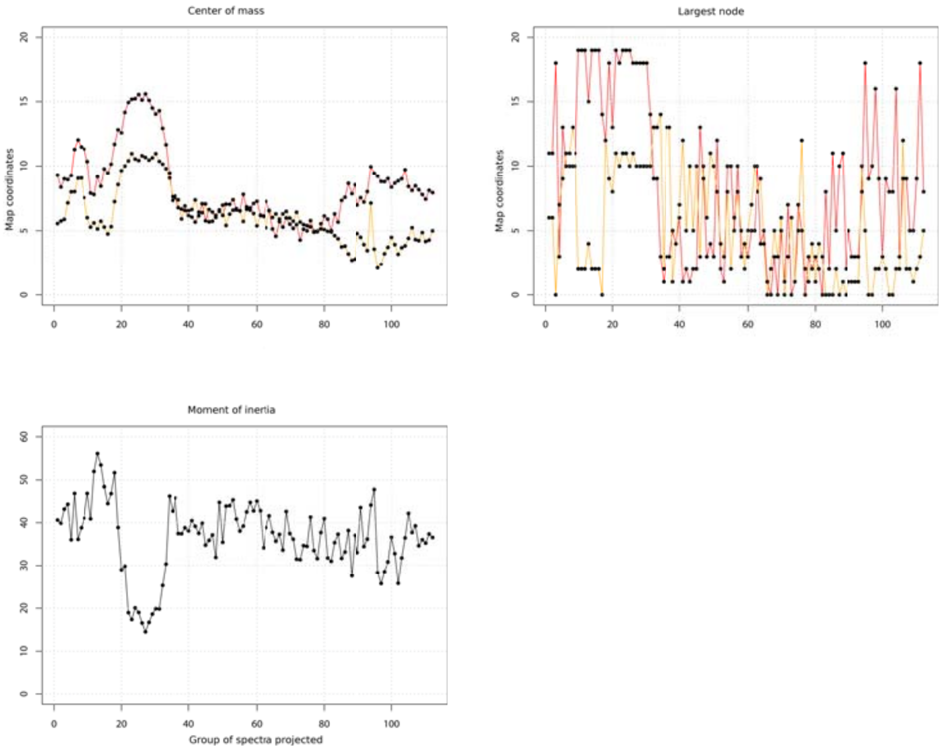


Figure 2.9: Raoul distribution analysis of the seismic tremor acquired during March 17 (UT time).

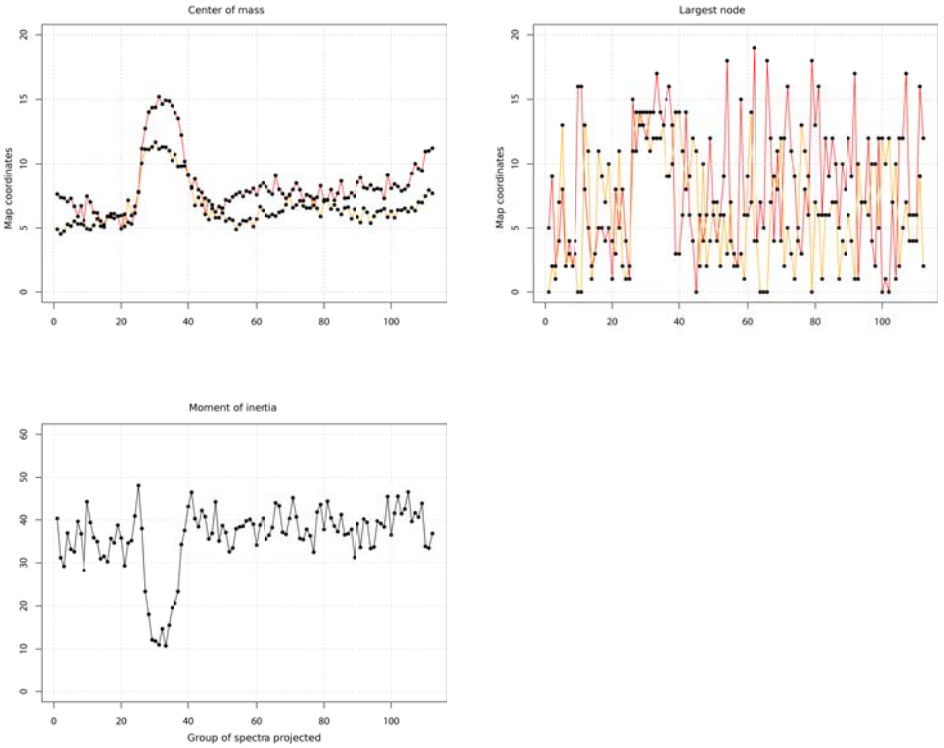


Figure 2.10: Raoul distribution analysis of the seismic tremor acquired during March 18 (UT time).

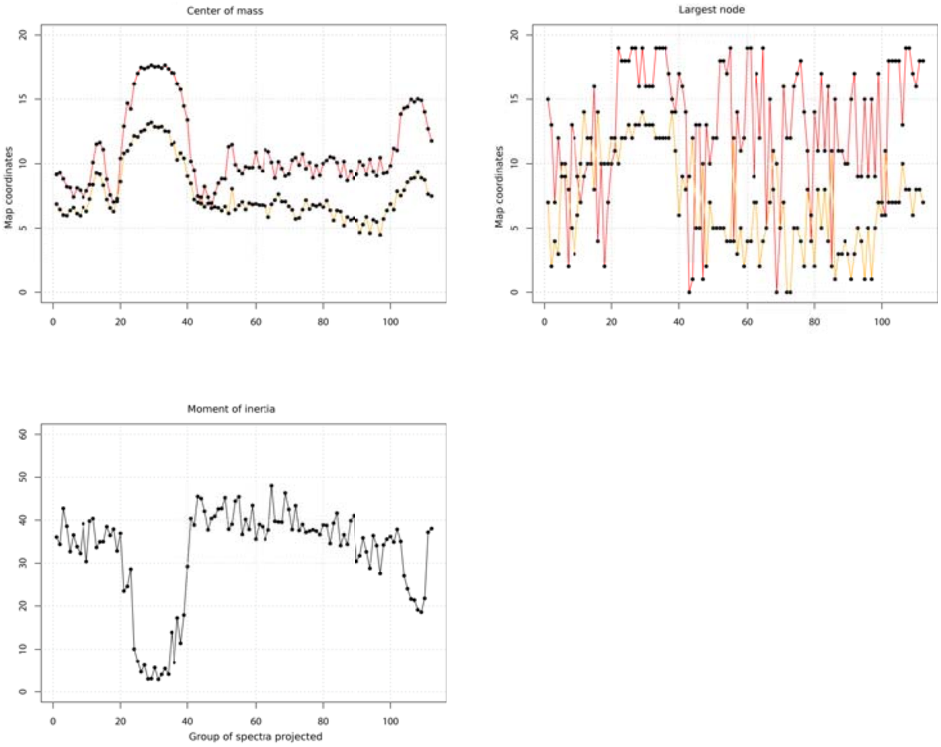


Figure 2.11: Raoul distribution analysis of the seismic tremor acquired during March 19 (UT time).

2.3 Ruapehu

Ruapehu volcano (fig. 2.12) is one of New Zealand's most dangerous volcanoes, due to the proximity of recreation activities near the active vent and to its frequent and often unpredictable eruptions. The volcano lies at the southern portion of the Taupo Volcanic Zone [Hackett and Houghton, 1989] and has produced 8 significant eruptions since 1945, including magmatic, phreato-magmatic and phreatic events. Eruptions occurring in 1969, 1975, 1988, 2006 and 2007 occurred without evident precursory activity [Sherburn et al., 1999, Mordret et al., 2010].

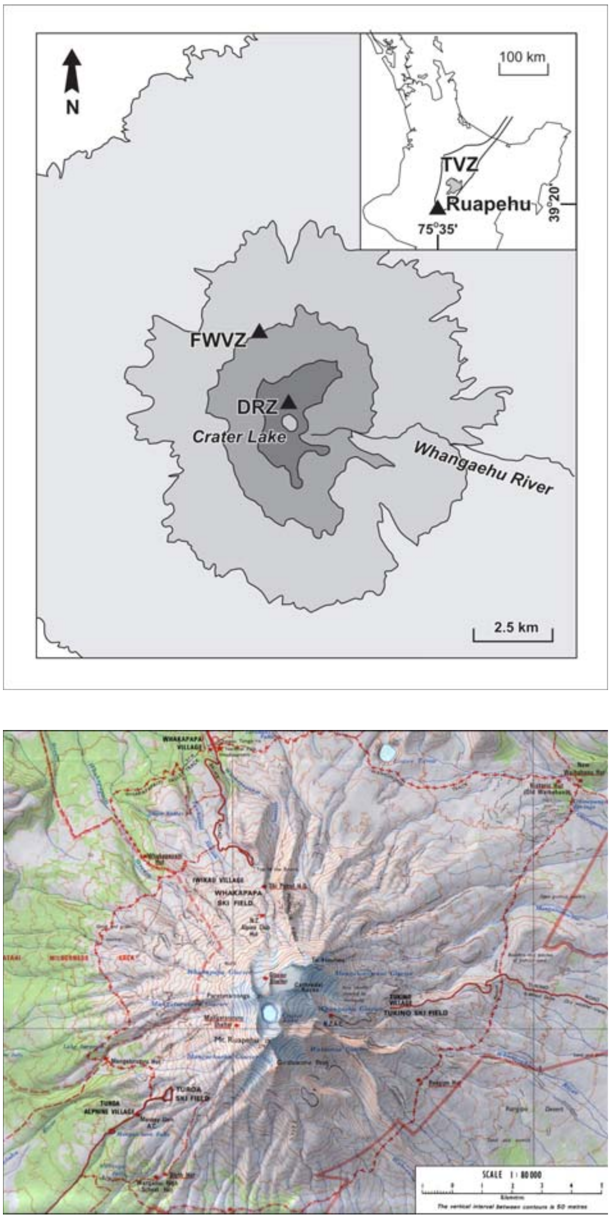


Figure 2.12: Location map showing Ruapehu volcanic massif (elevation 2797 m) with topographic contours. Contours are at 500 m intervals relative to sea level and the innermost contour is at 2500 m. Ruapehu volcano is on the southernmost margin of the Taupo Volcanic Zone (TVZ) (inset). Broadband (FWVZ) and short-period seismic sensor (DRZ) are shown as triangles.

Recent improvements in the seismic monitoring systems around the volcano now allow the application of increasingly sophisticated volcano monitoring techniques and research activities. This work presents a SOM analysis of the evolution of tremor that occurred before, during and immediately after two small phreatic events at Ruapehu volcano, it focuses on recent events because the data has improved with the addition of broadband digital seismic systems. Both volcanic events, occurring on 4 October 2006 [Mordret et al., 2010] and 25 September 2007 [Christenson et al., 2010, Jolly et al., 2010, Kilgour et al., 2010] have documented surface expressions and are thought to be phreatic in nature.

It's shown here how the dynamics of the volcanic system for these eruptive events can be better understood by application of SOM cluster analysis and offer a new interpretation of these patterns in the context of both surface observations and the seismological features associated with mostly sub-surface processes.

The task becomes once again that of highlighting the existence of dynamic regimes that are consistent and coherent over time and/or identifying them as possible precursors of paroxysmal phases. For this SOM cluster analysis several enhancements have been introduced to the algorithm originally proposed by Carniel et al. [2012] a discussion of the results in the context of phreatic eruptive activity and shallow hydrothermal models for Ruapehu follows at the end of this section. The section 2.3.3 shows an alternative SOM analysis of the data set related to the event of the 25 September 2007 acquired by another seismic station, the one nearest to the vent that has been destroyed during the same explosion. It doesn't involve the cluster recognition, but it analyses the position that data, ordered in time, take onto the SOM map during the projection phase, with particular attention to the moment of inertia of the distribution of each group in which the data set is subdivided.

2.3.1 A SOM for Ruapehu

With respect to the original algorithm [Carniel et al., 2012] applied to the Raoul data set (see sections 2.2.1 and 2.2.2), the classic FFT has been replaced by the Welch's method [Welch, 1967] implementation [Lees, 2012] in the R environment [Team, 2010]. This allows a better balance between time and frequency resolution (*evolfft* function, *RSEIS* package) and damping of noise effects with an original Konno–Ohmachi smoothing function (Konno and Ohmachi [1998], see A).

The acquired signal is split into time windows of length *Nfft* (e.g. *Nfft* = 2048 points in the following analysis), then the *Nfft* window is split into sub-windows of length *Ns* (e.g. *Ns* = 1024 points in the following analysis) overlapped of length *Nov* (e.g. *Nov* = 920 points in the following analysis). On one hand the frequency resolution is controlled by the *Nfft* parameter, on the other hand the time resolution is controlled by the sub-window length (*Ns* parameter). The *evolfft* function first resets the mean of each sub-window to zero, then applies a cosine taper window and extends each sub-window to *Nfft* length, adding zeros and finally applying a multivariate Fast Fourier Transform (FFT) to the matrix composed of the sub-windows arranged in columns. Using a taper window means that the signal near the time being analysed will have higher weight.

The input vectors for the SOM are still the individual columns of the spectrogram. Each spectrum is smoothed using the function proposed by Konno–Ohmachi:

$$f_s(f;f_{ns}) = \left[\frac{\sin\left(\log\left(\mathrm{f}/\mathrm{f}_{\mathrm{ns}}\right)\right)^b}{\left(\log\left(f/f_{ns}\right)\right)^b}\right]^4$$

where the smoothed frequency value f_s depends on its original value f_{ns}, on the bin number f and on the band width parameter value b. An example of such input spectrogram is shown in fig. 2.13. Each spectral column becomes an individual input to the SOM learning process, where the discriminant function is now based on the weighted cross-correlation index (see section 1.5) instead of the Euclidean distance used in Carniel et al. [2012]. This better captures spectral shapes rather than point values. Moreover, the patterns are now compared only in a frequency range of interest, this means that the SOM map can be trained using just the frequency range that is considered most relevant, but the time evolution of resulting spectra can be subsequently examined in the entire frequency range. This is equivalent to saying that the feature vectors are built using only part of the available spectrogram [Langer et al., 2011]. The frequency range of the training can be chosen on the basis of previous knowledge or simply by trial and error, evaluating the resulting map organisation in terms of spatial coherence and therefore clustering, and on the base of temporal coherence of the resulting "regimes".

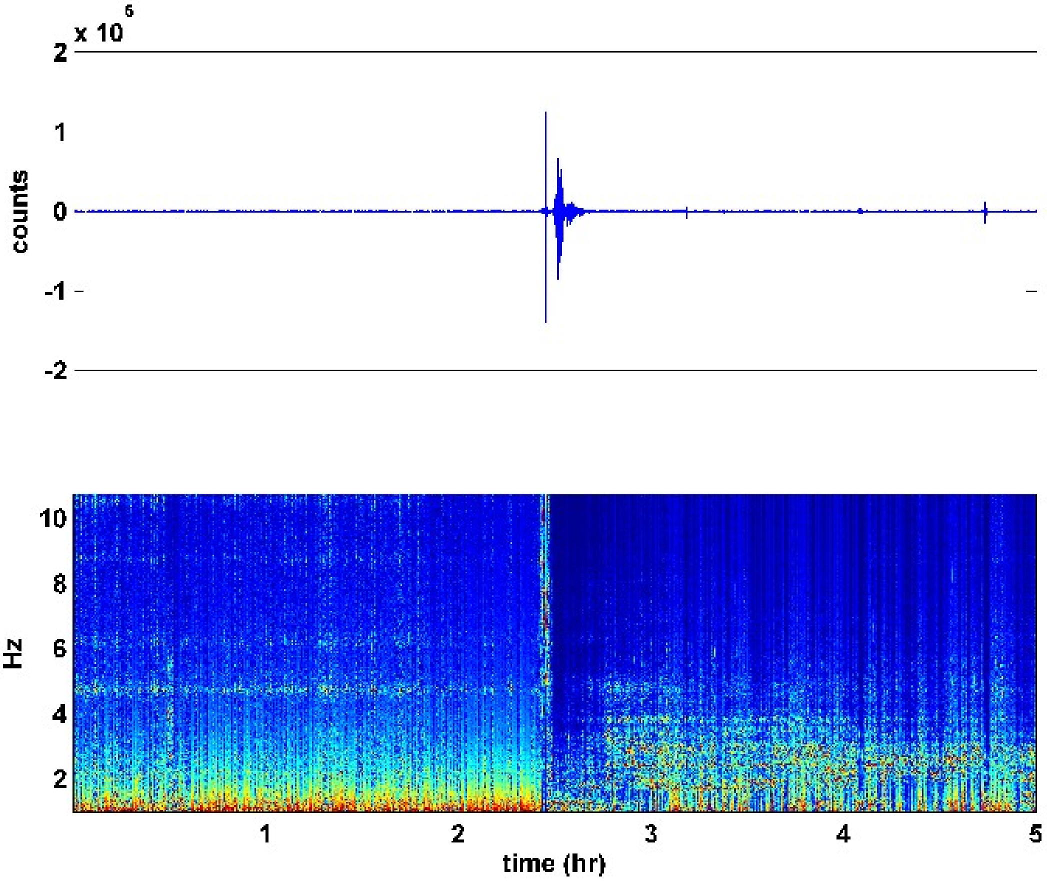


Figure 2.13: Time series (top) and spectrogram (bottom) of seismic data recorded at FWVZ seismic station from 7.00 to 12.00 UTC, 4 October 2006. Spectrograms were produced using 2048 sample windows and a 1000 sample time step for data digitised at 100 Hz. The 5 hour interval was used to train the SOM. This interval includes a phreatic event at approximately 09:24 UTC.

In order to avoid the fact that neurons along edges of flat maps do not have the same number of neighbours as the others, resulting in an inhomogeneous training process, the SOM map is toroidal (see section 1.7).

A SOM was trained using the spectrograms (fig. 2.13) of the seismic data for a 5 hour period surrounding the event of 4 October 2006 (from 7:00 to 12:00 UT).

The two events examined exhibit a range of phreatic responses from mild lake level changes to a relatively large phreatic explosion. It follows a briefly description of them based on seismic data recorded at station FWVZ (Guralp CMG-40T 60s broadband station at

Ruapehu volcano) which is sampled at 100 Hz and is located about 3 km from the vent. Although FWVZ is not the closest station to the crater, it has been used because it remained operational for both phreatic events. In fact, as mentioned above, the closest station (DRZ) was destroyed during the 2007 events so that no data are available after that phreatic explosion (fig. 2.12).

2.3.1.1 The 4 October 2006 event

A seismic transient was detected on 4 October 2006, 22:24 NZDT (09:24 UT) based on recordings of the seismic station FWVZ (fig. 2.14A). Subsequent visual observations occurred 2 days later and confirmed that a small eruptive event had occurred based on thin lake edge deposits generated from a 4 to 5 m high wave. The volcanic earthquake's magnitude was 2.9 ML, based on a fixed location at the volcano summit, but produced no airwave and no ash ejection into the atmosphere [Mordret et al., 2010]. Based on these observations, the event is interpreted to have been a small subaqueous eruptive event, that was followed by a slow (over 13 days) about 1.5 m rise of the lake level [Mordret et al., 2010] which contributed to the 18 March 2007 tephra dam collapse and lahar [Manville and Cronin, 2007, Carrivick et al., 2009]. In subsequent detailed analysis, the seismic transient was found to contain no very long period seismic component (VLP) and the relative seismic velocity during the days surrounding the eruption was found to be lower than the long term average value (about 0.8%) based on an ambient noise analysis. This suggests a localised extension of the volcanic edifice [Mordret et al., 2010].

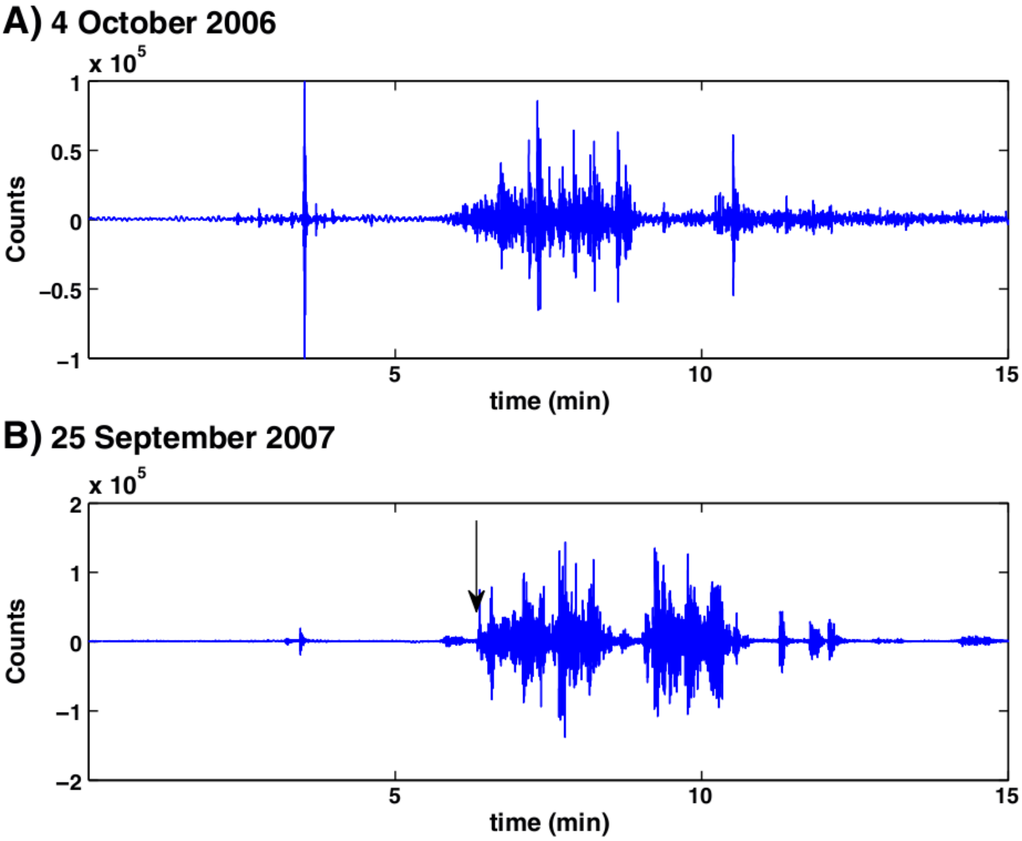


Figure 2.14: Waveforms for example volcanic earthquakes. A) 4 October 2006, B) 27 September 2007. The onset of the 25 September 2007 eruption is shown by the black arrow.

2.3.1.2 The 27 September 2007 eruption

Ruapehu erupted again on 27 September 2007 at 20:26 NZDT (08:26 UT) producing a steam cloud that rose to 4600 m above the crater lake, as well as northward directed ballistic and surge deposits [Kilgour et al., 2010]. The eruption produced a complex seismo-acoustic sequence including broad-spectrum eruption tremor (fig. 2.14B) with a strong VLP component [Jolly et al., 2010]. The earthquake had a local magnitude of 3.2 ML and a moment magnitude of 4.0 MW. The acoustic signal was generally low frequency, with similar spectral characteristics to the VLP seismic component (6–25 s period). This event was much more energetic than the 2006 event [Jolly et al., 2010], causing injuries, damaging the seismic station closest to the crater (DRZ) and triggering two lahars down the Whakapapa and Whangaehu drainages [Kilgour et al., 2010].

2.3.2 Ruapehu data clustering and interpretation

Another improvement to the original algorithm [Carniel et al., 2012] is aimed at simplifying result presentation and interpretation, as proposed also by other authors [Vesanto et al., 2000, Messina and Langer, 2011]. After the data are projected onto the map, an automatic cluster analysis procedure is applied, again using a weighted cross-correlation to compare the code vectors of each pair of neurons.

First the weighted cross correlation matrix is calculated, then the hierarchical clustering performed by the *agnes* function (R environment, *cluster* package) is applied [Kaufman and Rousseeuw, 1990]. The clustering method that was selected within the *agnes* function is the “average” ([unweighted pair-]group average method, UPGMA). Each observation is thought to be a small “one-node” cluster, that is then progressively merged with similar alternative clusters, until only one large cluster remains which contains all the observations. At each stage the two most similar clusters are combined.

The result is a figure called a dendrogram where the clustering procedure described above can be seen proceeding from bottom to top. An example is shown in fig. 2.15, which is derived from the spectrogram of the vertical component of the volcanic tremor shown in fig. 2.13. It should be noted that the clustering procedure does not act on the original data vectors, but on the node vectors, that are already “prototypes” of a class of data vectors, so this is not equivalent to a direct clustering of (spectral) data but adds a further level of “abstraction” and should therefore better extract characteristic features of the data.

The clustering procedure can be interrupted at any vertical level by choosing an appropriate similarity threshold, which directly determines a certain number of clusters on the map to which nodes are assigned. The choice of the threshold is not straightforward and provides another degree of freedom for the fine tuning of the SOM analysis, because there isn’t a single optimal threshold for any application or dataset. Choosing a low threshold increases the number of clusters, and each of them becomes highly specialised to recognise a particular kind of pattern, so the SOM can recognise very subtle variations in the data. However, this can often lead to the assignment of data windows to clusters having lower coherence in time evolution, hence consistent regimes may not be recognised. For this reason it is advised to carry out several analyses with different thresholds before choosing the threshold that best optimises time coherence while maintaining acceptable “resolution” in terms of distinction between different spectral patterns. In order to better carry out these tests and present both test and final results in an easy to interpret way, clusters can be represented by different colours on the SOM and in the plots of time histories. At the chosen similarity threshold, the different clusters represent typical spectra with specific characteristics that can be distinguished from one another by the SOM.

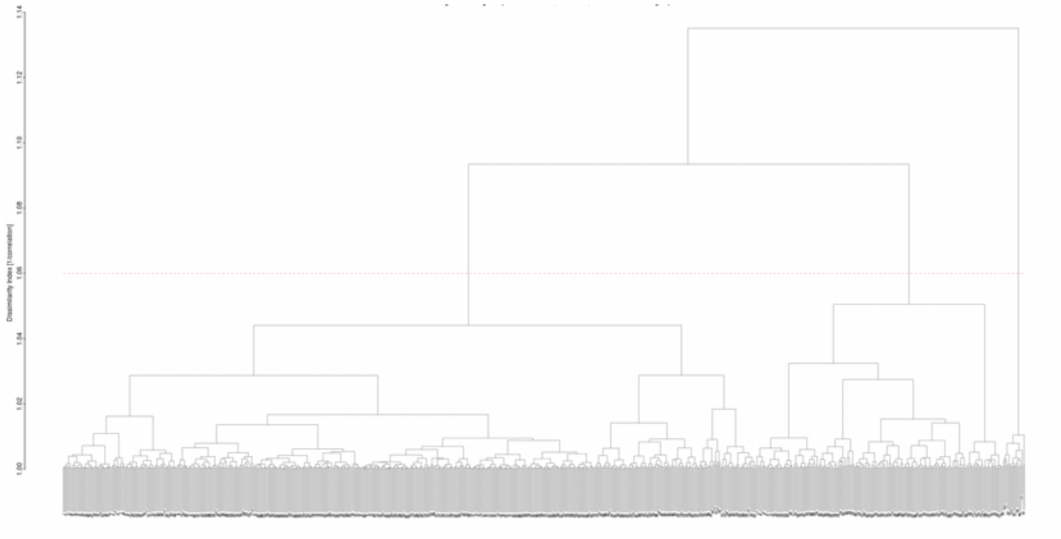


Figure 2.15: Dendrogram summarising the clustering of nodes of the SOM produced by training using the spectrogram of the seismic data recorded at FWVZ station from the interval shown in fig. 2.13.

Cutting the dendrogram of fig. 2.15 at a threshold of 1.060 generates 3 clusters, which have been labelled with the blue, green and red colours respectively. Examining the spatial distribution of the nodes corresponding to each cluster/colour on the SOM (fig. [fig:RuapehuUmatrixClustering]) we notice that the same colour is associated with nodes geometrically nearby on the map, which demonstrates a topologically consistent training result.

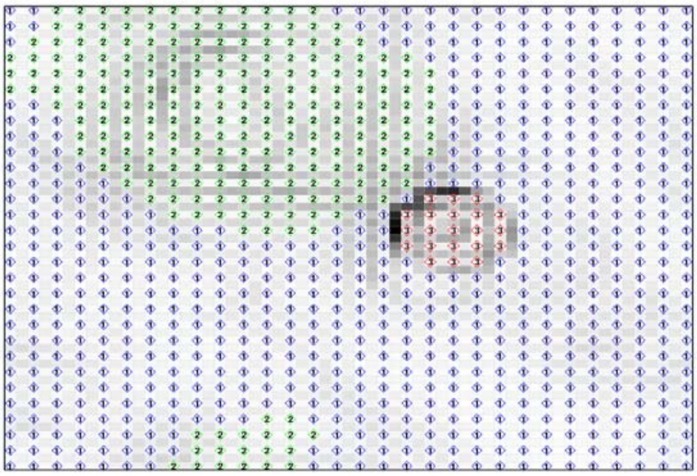


Figure 2.16: Modified U-matrix for clustering purposes. SOM showing by colours the nodes assigned to each of the clusters after the training for 2006 data. The map should be seen as toroidal, i.e. the top side connects to the bottom, the left side connects to the right.

The different clusters have typical spectra that characterise and distinguish them from each other. This is illustrated by average spectra computed over time windows belonging to Cluster 1 (fig. 2.17), Cluster 2 (fig. 2.18) and Cluster 3 (fig. 2.19) respectively. It's also possible return to the time domain, to detect when data enters each of the clusters. Long durations of the same colour indicate a consistent behaviour of the volcanic system, which it may be called a volcanic regime.

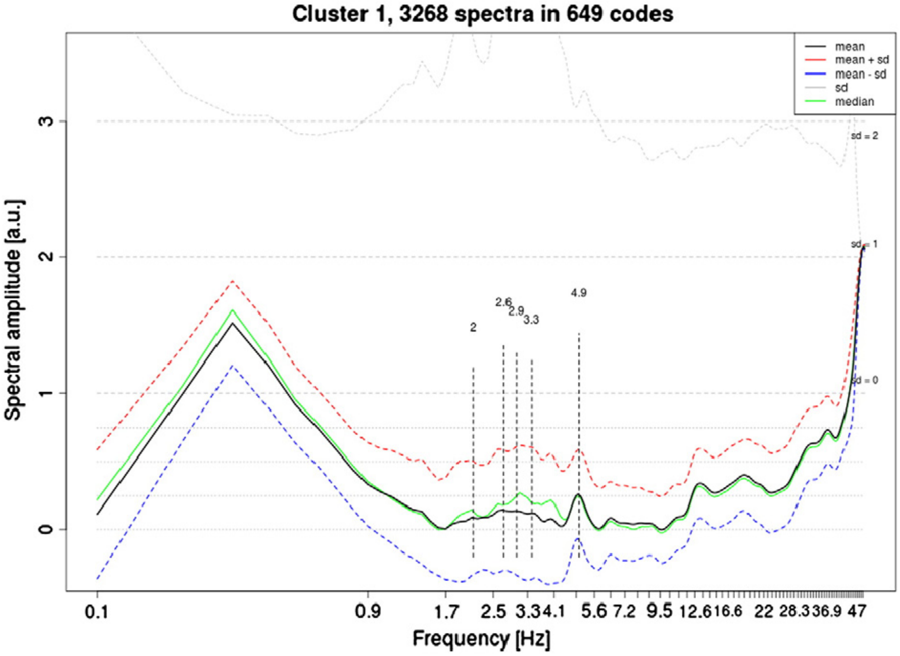


Figure 2.17: Sample spectrum characteristic of "blue tremor", obtained averaging only time windows of 2006 seismic data belonging to Cluster 1.

For instance, it can be noted a significant change of style (see fig. 2.20) shortly before the eruption, graphically characterised by the transition from an interval dominated by the green cluster, to another where the blue cluster is more dominant. This is accompanied by a relatively brief appearance of time windows assigned to the red cluster. It is worth noting that during the paroxysmal phase of the eruption, the tremor seems often to behave similarly to the "normal" green spectral cluster.

The same SOM map has been applied, without further training, directly to the data representing the volcanic tremor preceding the phreatic eruption of 25 September 2007. The interesting observation is that in 2007 (Fig. 2.21), again at station FWVZ, the behaviour is similar to that in 2006 (Fig. 2.20). This means that the SOM, which "learned" the regime transitions shown by the 2006 event, was able to recognise those same transitions for the much more energetic phreatic eruption of 2007.

SOM analysis based on seismic spectrograms had provided information about the existence of relatively stable volcanic regimes and highlight (gradual or abrupt) transitions between them. As a clustering automatic detection has been applied in the post-processing phase, the transitions between regimes can be represented by discrete colours, so of course a gradual transition would appear as a transition from a regime strongly dominated by colour A to another regime strongly dominated by colour B passing through a time window where neither colour dominates.

Volcanic regime changes on a time scale of a few days have been noticed at Ambrym volcano (Vanuatu Islands), characterised by the presence of lava lakes, where the transitions were sometimes associated to the occurrence of tectonic events [Carniel et al., 2003]. Erta Ale volcano in Ethiopia, also characterised by the presence of an active lava lake, showed transitions on a shorter time scale (tens of minutes) that could be correlated instrumentally and visually to two different regimes of "low" and "high" convection in the lava lake [Harris et al., 2005, Jones et al., 2006]. Similar transitions at similar timescales were observed

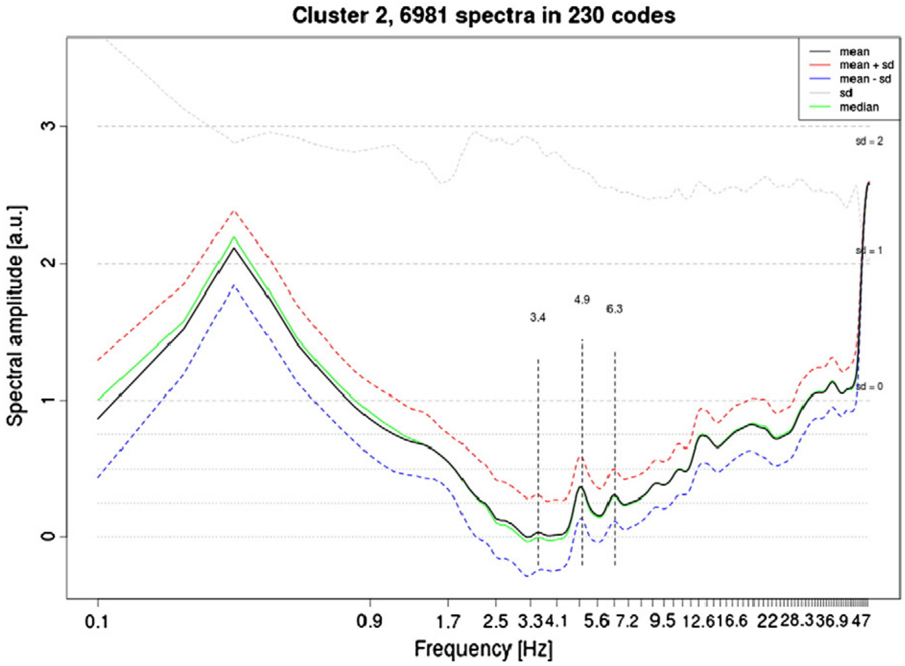


Figure 2.18: Sample spectrum characteristic of “green tremor”, obtained averaging only time windows of 2006 seismic data belonging to Cluster 2.

with a multiparameter approach – including seismic, infrasonic and thermal data – also at Stromboli volcano, characterised by regimes of strombolian activity and degassing of different intensities [Ripepe et al., 2002].

There also, transitions can be triggered over longer timescales by the occurrence of tectonic events [Carniel and Tarraga, 2006] or by the occurrence of paroxysmal events [Carniel and Cecca, 1999]. The analysis of seismic data recorded at Dallol, a geothermal field not far from Erta Ale in Ethiopia, showed that the alternation of regimes with different spectral characteristics can also be observed in volcanic areas showing only external geothermal activity [Carniel et al., 2010]. This observation was confirmed by the SOM analysis of seismic data recorded at Raoul Island [Carniel et al., 2012] close in time to the phreatic explosion of 17 March 2006 (NZST). This supports the observation that a SOM unsupervised pattern recognition scheme can allow the automatic recognition of these regimes. Moreover, by monitoring also the error in the classification, e.g. as the distance between each input vector and the correspondent BMU code vector, the SOM approach is potentially able to recognise outliers, i.e. anomalous states not previously observed, which may be a precursor to an impending paroxysmal phase.

Ruapehu’s shallow hydrothermal system has been previously described by using the shallow heat-pipe model [Hurst et al., 1991] which explains the efficient transport of heat within the shallow volcanic system, without an associated transfer of magmatic mass. Moreover, the model also explains the intermediate term cyclic behaviour during non-magmatic phases. More recently, Christenson et al. [2010] showed the importance of CO2 transport through the shallow Ruapehu system and its role in generating shallow hydrothermal seals through the formation of an elemental sulphur–anhydrite–natroalunite mineral assemblage. Christenson et al. [2010] suggest that seal formation could reduce permeability and promote pressurisation as gases continue to flow from depth. Hence the 2007 eruption could be a direct result of failure of the shallow hydrothermal seal, resulting in an interval of depressurisation. One interesting aspect of the 2007 eruption is suggested by the location of pre-eruption volcano-tectonic and VLP seismicity. Jolly et al. [2010] showed that these

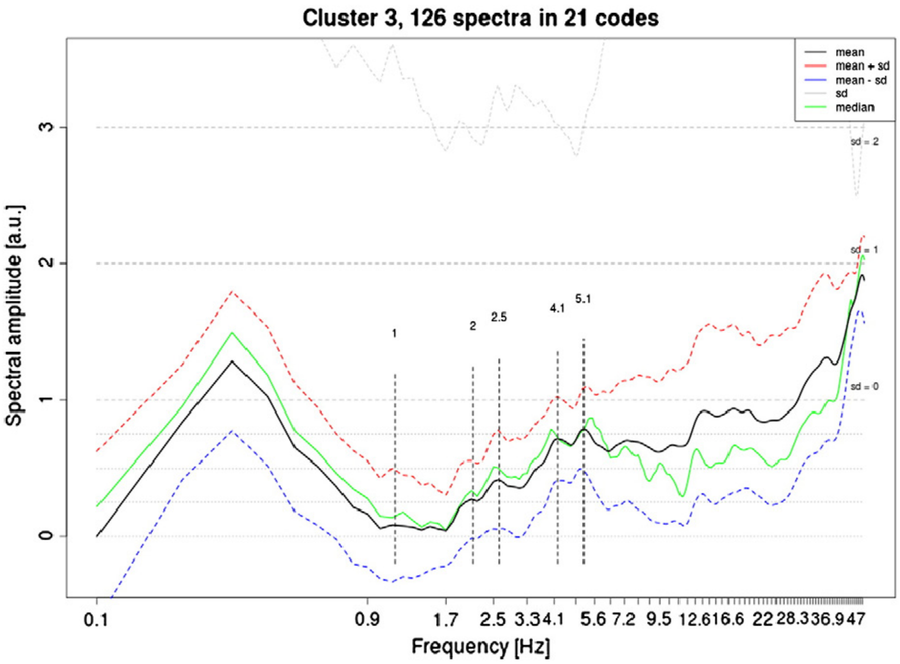


Figure 2.19: Sample spectrum characteristic of "red tremor", obtained averaging only time windows of 2006 seismic data belonging to Cluster 3.

events, which occurred about 1–10 min before the eruption, had depths on the order of 3–7 km, deeper than the hydrothermal seal. Evidence for shallower, presumably hydrothermal tremor, occurred in a period less than 1 min before the eruption and this tremor might be an alternate indicator of seal failure. In addition, Christenson et al. [2010] found evidence for a small amount of juvenile magmatic material in the 2007 eruption deposits.

Examining the results of the SOM analysis in the context of the observations from the 2006 and 2007 events, it can be noted the dominance of the 'green' tremor pattern prior to both eruptions and suggest that when the system is pressurised it produces spectra having features similar to those shown in fig. 2.18. In Fig. 2.18, the spectral peaks below 4 Hz are reduced compared to spectral peaks at 4.9 and 6.3 Hz — these frequencies can be considered approximate, as the exact frequencies of the spectral peaks in the time windows that fall into the 'green' cluster are not constant throughout the analysed period. Such patterns, observed in the 'green' cluster, may be then considered representative of a pressurisation phase in the context of the seal model.

Next, it can be observed a short period with broader spectral character (fig. 2.17) denoted by 'red' in the SOM analysis. This phase includes some discrete volcano-tectonic earthquakes whose appearance was recognised for both the 2006 and 2007 events [Mordret et al., 2010]. This broad spectrum 'red' phase is possibly associated with failure of either the shallow seal, a deeper magma carapace or both. In the latter case can occur both a failure at depth and a seal rupture at a shallow level. The question remains whether the process is a top–down depressurisation, beginning when the seal breaks and initiating the deeper ruptures [Christenson et al., 2010] or the other way around [Jolly et al., 2010], i.e. the occurrence of deeper ruptures starting a bottom–up decompression and causing the failure of the seal of the near-surface hydrothermal system.

Interestingly, the main eruption for both the 2006 and 2007 events have signatures most similar to the 'green' pre eruption pattern, while the post eruption sequence is clearly dominated by the 'blue' pattern. Typical spectra for blue phase contain stronger low frequency excitation in the frequency band 2 to 3 Hz. The spectral shift from higher frequencies to

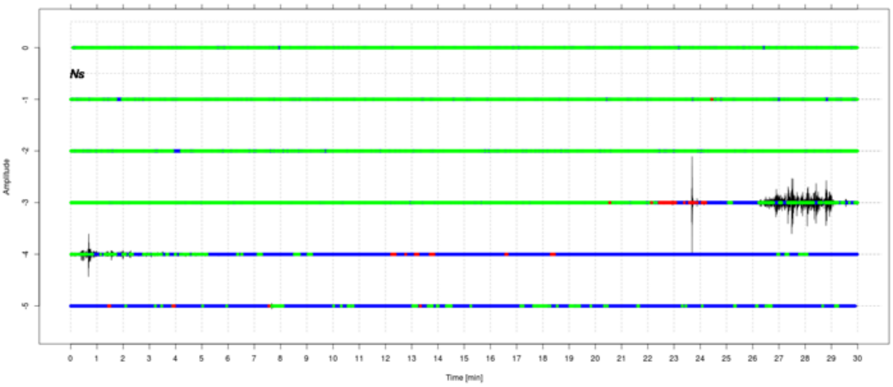


Figure 2.20: Time series of seismic data recorded at FWVZ station surrounding the 4 October 2006, 09.24 UT phreatic event, colour-labelled according to the clusterization of 2006 SOM. Each horizontal line represents 5 min. The transition from a “green tremor”-dominated regime to a “blue tremor”-dominated one through an interval of “red tremor” is evident.

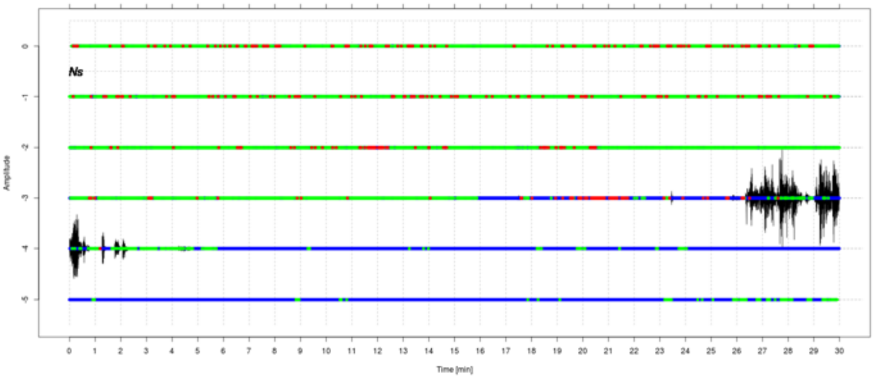


Figure 2.21: Time series of seismic data recorded at FWVZ station surrounding the 27 September 2007, 08:26 phreatic event, colour-labelled according to the clusterization of 2006 SOM. Please note that SOM was not re-trained with any 2007 data. Each horizontal line represents 5 min. The transition from a “green tremor”-dominated regime to a “blue tremor”- dominated one through an interval of “red tremor” is evident.

lower frequencies is consistent with the depressurisation pattern that has been modelled for magmatic systems [Neuberg and O’Gorman, 2002] and is directly applicable to tremor excitation seen here. It is uncertain if the pressurisation of a hydrothermal system below a mineralogical seal can produce a similar resonance pattern. The observed pattern is also consistent with ambient noise results obtained by Mordret et al. [2010], who showed that an anomalous reduction in relative seismic velocities occurred beginning 3–4 days prior to the 4 October 2006 event. The anomaly disappeared within 3–4 days and seismic velocities returned to background levels. It is uncertain if longer term changes to the Ruapehu hydrothermal system, such as those observed at the nearby Tongariro area in 2008 (e.g., Johnson and Savage, 2012), could be resolved using a spectral analysis approach. Such an analysis would have greater data processing requirements and the clusterization procedure would be more challenging in order to discern these changes from other (variable) background activity.

Because the proposed clusterization is limited to three clusters for any time window, the interpretation of the ‘colours’ is without doubt not unique. However, it has been observed that noteworthy and similar spectral evolutions are seen in the two example eruptions that have been recorded on the modern Ruapehu geophysical monitoring network. Obviously, the same colour can be interpreted differently in different eruptive contexts and even future phreatic eruptions may confirm this pattern or may follow another path. The work presented here however shows the potential value of the SOM approach to geophysical monitoring and highlights the need for more data describing examples of this Ruapehu event type.

2.3.3 An alternative SOM analysis for Ruapehu

The SOM training phase has been carried out using the data acquired by the nearest to the vent seismic station (fig. 2.12) during the days of the 23, 24 and 25 September 2007. The seismic station has been destroyed by a lahar induced by the explosion of the 25 September, so just the data acquired during the first 8 hours and 20 minutes are available for the day of the explosion. Still taking the approach for the detection of possible precursors of the volcanic crisis, the spectra calculated on the time windows related to the event of the 25 September are not used for the SOM map training, except for some spectra related to the beginning of the explosion (see the dotted line in fig. 2.22).

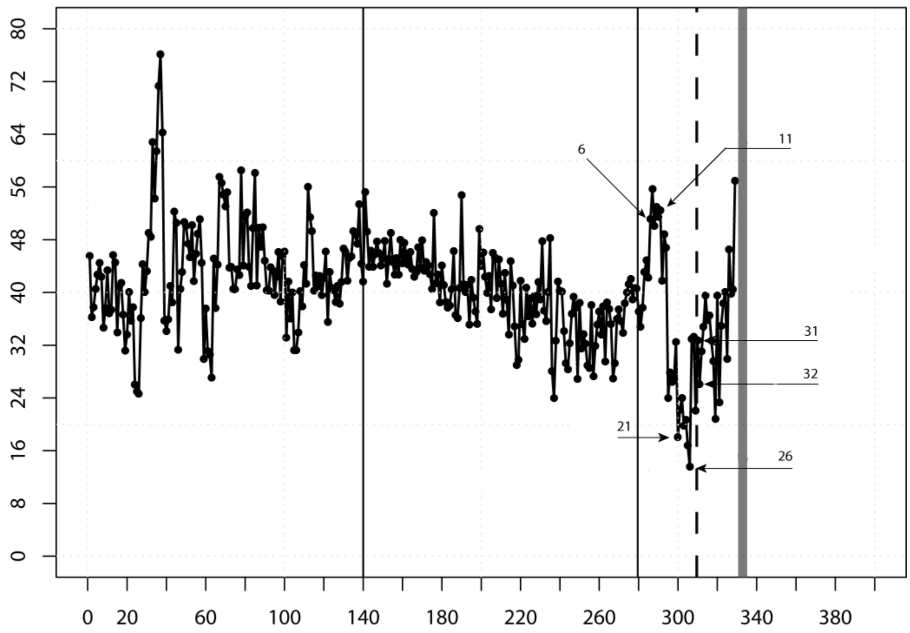


Figure 2.22: Moment of inertia (y axis) trend for the groups of spectra of the days 23, 24 and 25 September 2007. The spectrogram of each day has been subdivided into 140 groups and each group have 120 spectra, corresponding to a time window of about 10 minutes. The thick gray line at the right side of the figure indicates the moment at which the seismic station stops to acquired tremor and get destroyed, the dotted line indicates when the explosion starts.

Once the SOM map has completed the training phase, the spectrograms have been split into groups of 120 spectra (corresponding to a time window of 10 minutes about) and each group has been projected onto the map. For each group the data distribution has been analysed in terms of center of mass of the distribution and largest map's node position and in terms of moment of inertia value.

Fig. 2.22 shows the trend of the moment of inertia for the distributions of the data projected onto the SOM map. Each point corresponds to the moment of inertia value of one of the groups of 120 spectra in which the spectrograms are subdivided, corresponding to a time window of about 10 minutes.

The moment of inertia reaches the maximum value in respect to the groups from 6 to 11 (see 2.22). Fig. 2.23 shows where the data of the groups from 6 to 11 of the day 25 September 2007 are projected onto the SOM map. The groups from 6 to 11 are constituted by the spectra of the signal acquired from 4h 16' to 3h 25' before the explosion start (see the dotted line in fig. 2.22).Then the moment of inertia reaches the minimum value with the projection of the spectra of the group 26. Fig. 2.24 shows the projection of the spectra of the groups from 21 to 26. The groups from 21 to 26 are constituted by the spectra of the signal acquired from 1h 42' to 51' before the explosion starts (see the dotted line in fig. 2.22).

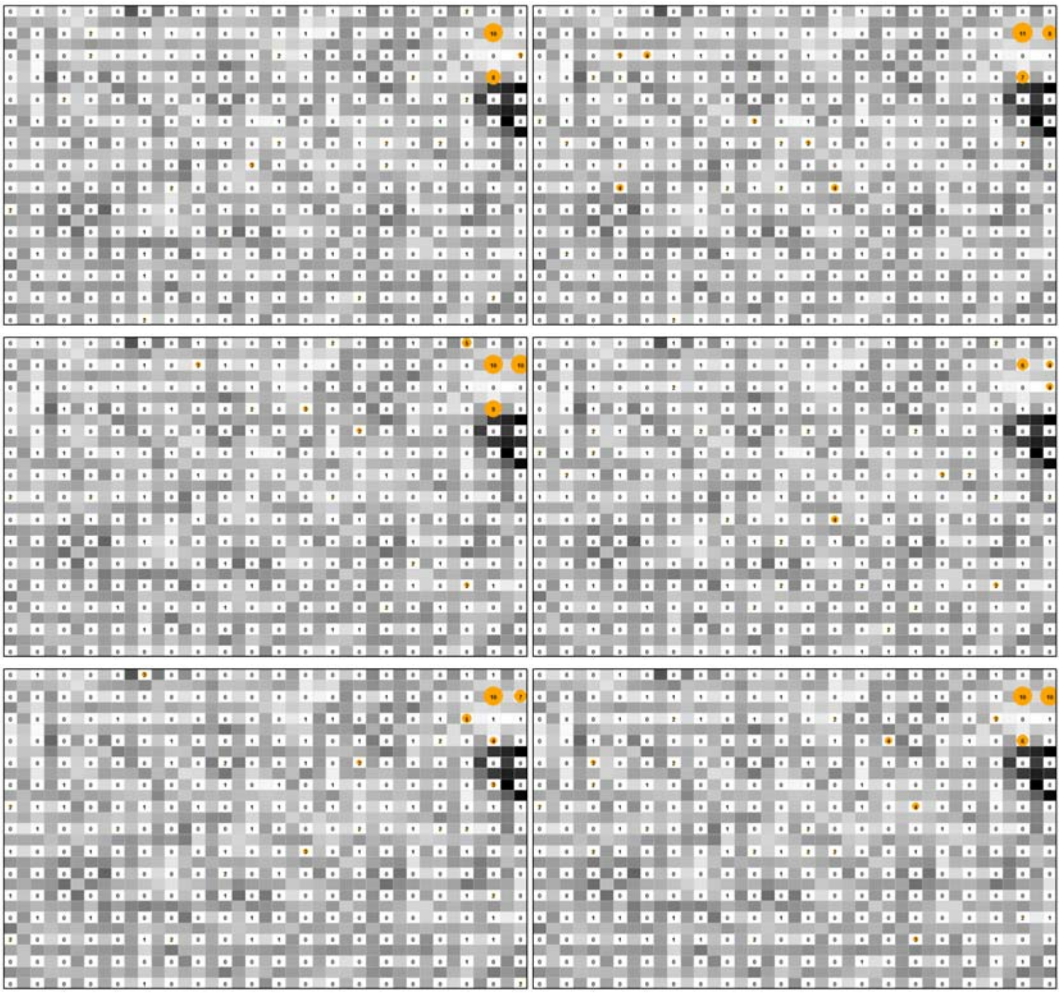


Figure 2.23: Distribution of the spectra of the groups from 6 to 11 of the 25 September.

Groups 31 and 32 (fig. 2.25) are constituted by the spectra of the signal acquired when the explosion started (see the dotted line in fig. 2.22). It can be noticed that these spectra are projected onto an area where any data had been projected before.

The SOM map analysis shows a meaningful trend of the moment of inertia value that highlight an unusual projection of the spectra onto the map, in respect to the previous groups projection, that starts from the 120 spectra of the group 6. That's occur from 4 hours about before the volcanic explosion start, highlighting the existence of dynamic regimes that are consistent and coherent over time and identifies a possible precursors of the paroxysmal phases.

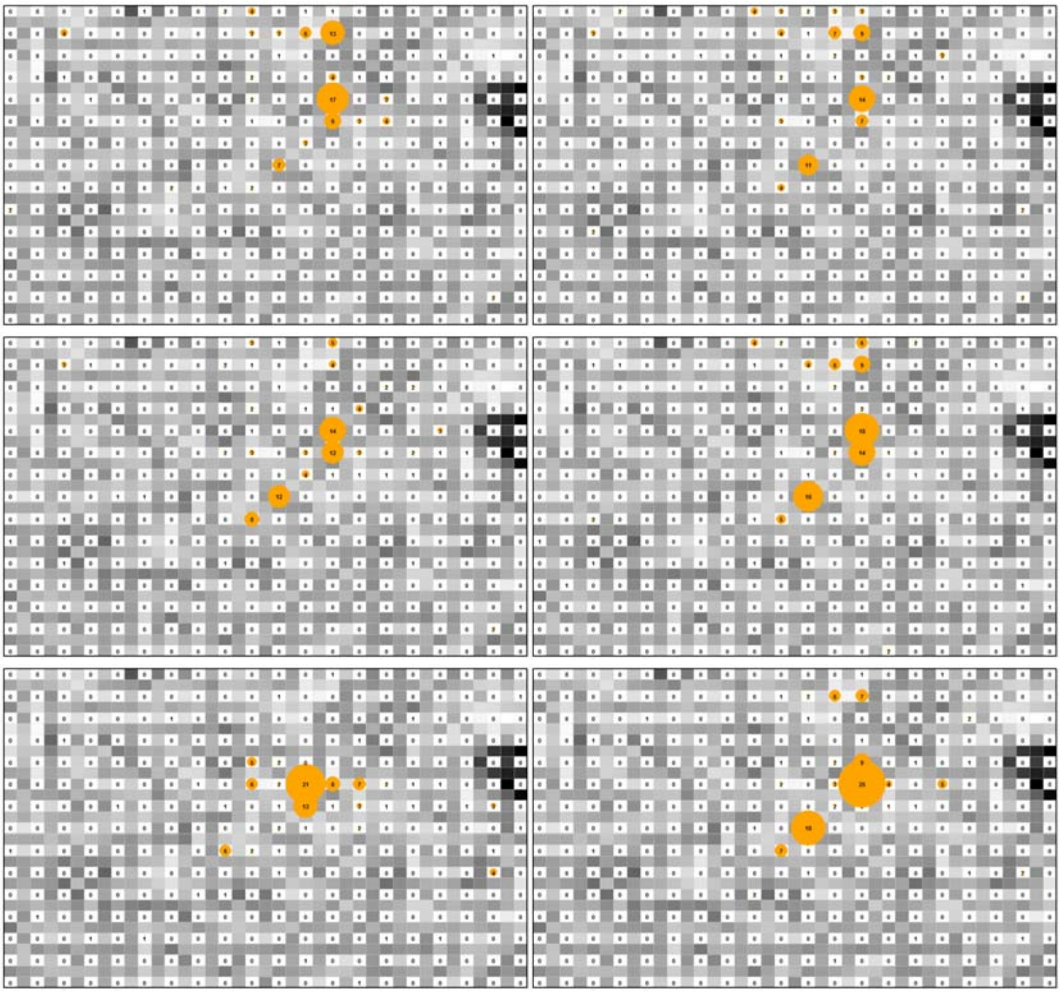


Figure 2.24: Distribution of the spectra of the groups from 21 to 26 of the 25 September.

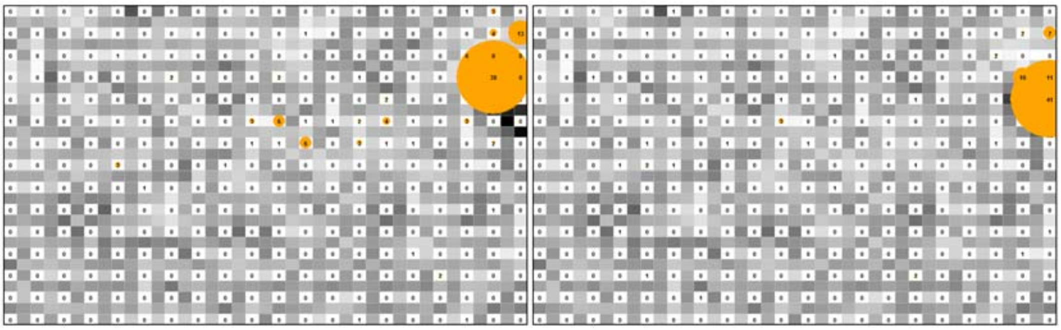


Figure 2.25: Distribution of the spectra of the groups 31 and 32 of the 25 September.

Chapter 3

HVSR tecnique improvement by SOM

The earthquake engineering has in recent decades developed rapidly, both in theoretical and experimental field. This development has also led to improve the finite element software products to carry out complex dynamic analysis of structures, supported by increasingly powerful computers that allow to consider non-linear materials and complex detailed models.

In respect of increasingly complex and detailed models of the structures, the new construction norms regarding the structural seismic response afford the opportunity to adopt techniques at various levels of complexity, but without giving any information or guidelines on how to obtain the parameters of interest to design new structures or to verify their performances and adequacy in case of earthquake.

The evaluation of damages due to seismic events shows that, for the same earthquake and the same structural typology of buildings, the effects can vary prominently among close sites. Geophysical studies during the last decades have shown that the rocks convey seismic motions without significant changes; on the other hand, less rigid sedimentary soils and particular configurations of the sedimentary dump can cause the amplification of the seismic waves near the surface, with potential disastrous consequences (e.g. the devastating effects of the Michoacan earthquake in Mexico City — more than 300 km away — in 1985). For this reason the rapid investigation methods based on environmental tremor analysis for site effects characterisation have assumed increasing relevance.

More attention should be paid to the interaction between soil and structure, even before the structure itself. The seismic design requires the estimation of the acceleration (or velocity, or displacement) to which the structure will be subject, at the various frequencies, in the event of an earthquake. These values are known at the bedrock[1], for example in terms of Peak Ground Acceleration (PGA) and reported onto probabilistic ground motion hazard maps. Since most of the buildings have their foundations on loose compressible soil, the need is to determine if the soil can modify the ground motion parameters (e.g. frequency, acceleration, velocity, displacement and duration) passing through the bedrock to the surface and in the case that some change occur, then what are the ground motion

[1] The so called "seismic bedrock" is a semi-infinite space of enough rigid material in which there isn't a significative modification of the seismic waves, moreover the soil that can be considered as the "seismic bedrock" is much more homogeneous in comparison with the layers of sediment lying over this semi-infinite space.

parameters at the surface. To evaluate the ground motion at the surface there are two main methods.

The first one involves a model of the soil, able to calculate the accelerogram at the surface starting from a given accelerogram at the bedrock. This implies that many parameters related to the soil structure (e.g. depth of the seismic bedrock, number of recognisable layers and their thickness over the bedrock, orientation of the geological horizons and geographic morphology of the surface) and to the soil mechanical properties have to be evaluated, each one introducing an error due to the complexity of the soil and to the means and methods of investigation. It's worth remembering that to determine the seismic response it's needed to define the propagation of seismic waves in the ground to a depth of the order of magnitude of the wavelength associated with the vibration frequency that can potentially damage the structures present on the area of interest. The estimation of the propagation velocity of seismic waves (S-waves in particular) and the way in which they decrease with the frequency (damping factor) in different geological units is of particular importance [Kramer, 1996, Lanzo and Silvestri, 1999]. These estimates can be obtained either from laboratory measurements on soil samples collected by drilling, or applying seismic prospecting techniques for in-situ measurements.

The second method to obtain information about the expected ground motion at the surface is to provide direct measures applying the so called passive seismic techniques. In this respect the Horizontal to Vertical Spectral Ratio method (HVSR or Nakamura's method - Nakamura [1989, 2000]) is a technique that allows to evaluate, in a cheap and relatively easy way, the fundamental frequency of a given site and so contributes to choose and verify the adequacy of the input parameters for the dynamical models of the designed structure.

Unfortunately the evaluation of the fundamental frequency (e.g. the peak of the H/V function) is not always easy. Problems can derive from directionality of seismic noise, that can make the estimate along different directions differ. This problem has been tackled in several previous papers [Carniel et al., 2006, 2009, Barazza et al., 2009]. Additionally, the acquired signal is split in time windows (upon which the Fast Fourier Transform - FFT - or the Welch's method are applied to obtain the corresponding frequency spectrum) but some of these windows can be affected by a considerable amount of noise, for example due to anthropogenic sources localised close to the seismic station. When this occur the sources can't considered randomly distributed and independent and the identification of these time windows allows an easier and more reliable location of the H/V function's peak. Moreover an important task is to ensure an efficient automatic data analysis, in order to allow a data interpretation as independent as possible from any a priori knowledge about the nature of the data and then about the area of investigation. For this reason the SOM process has been applied to the acquired tremor at the surface, as a suitable methodology for this purpose.

3.1 The environmental tremor

Passive seismic methods for site effects characterisation are essentially based on the study of the wavefield associated with the seismic environmental tremor. This tremor is present in every part of the Earth's surface due to natural and anthropogenic causes, and it's constituted by seismic phases that often have passed through significant portions of the Earth's crust. These features make the applicability of passive seismic measurements virtually unlimited in terms of depth of exploration. The main limitations are related to the presence of unknown sources that make the interpretation and analysis of data more complex as compared to active seismic techniques. In the case of passive measures, the results are usable for structural purposes only using statistical methods for the analysis of the average characteristics of the wavefield.

The environmental seismic tremor is related to a variety of possible causes, both natural (ocean waves, storms, wind, etc.) and artificial (vehicular traffic, industrial activities, power lines, etc.). Briefly, Bonnefoy-Claudet et al. [2004] suggest that:

- tremor at low frequency ($f < 0.5\,Hz$, $T > 2\,sec$), generated by ocean waves at a great distance, it is stable and consistent and is composed mainly of surface waves;
- tremor at intermediate frequencies ($0.5 \leq f \leq 1\,Hz$, $1 \leq T \leq 2\,sec$) is generated by waves on the shore at a short distance, by the wind or, in certain circumstances, by human activities. Stability is much smaller and the content in terms of surface waves is variable;
- the tremor with frequency $f > 1\,Hz$ ($T < 1\,sec$) is essentially of local origin and it's related to human activities. It's highly unstable in terms of both amplitude and the ratio of energy between body waves and surface waves from which it's composed.

Generally the tremor has a complex structure with an average constant amplitude values on the scale of tens of minutes, but it can vary depending on the time scale. From the point of view of spectral content, tremor amplitudes related to the frequencies higher than $1\,Hz$ show systematic variations between day and night, while the relative amplitudes of lower frequencies (also known as microseisms) remain constant, showing some variability with respect to the local structure of the substrate.

In the absence of controlled sources, seismic noise is therefore an essentially stochastic phenomenon that requires specific methods of analysis, both theoretical and experimental. If the essentially stochastic character of the phenomenon leads up to some problems of theoretical type, on the other hand it allows to reduce the complexity of the analysis using the statistics. For example, assuming that the noise is the result of a widespread distribution of sources that are activated randomly (think to an urban environment), the average structure of the noise will be statistically independent of the nature and location of the sources, while will be determined by the structure of the soil.

3.2 HVSR method

The spectral ratios method (HVSR or Nakamura's method) results from studies carried out in Japan in the 50s [Nogoshi and Igarashi, 1971, Ameri, 1980, Nakamura, 1989, 2000] and it has been involved in numerous applications, even if it remains at the center of a bitter scientific controversy, involving not secondary aspects of the procedure (see e.g. SESAME [2005], Tokimatsu et al. [1992], Lachet and Bard [1994], Bard [1999], Mucciarelli and Gallipoli [2001]). The application of the method is very simple and is based on the analysis of the relationship between the spectral ordinates (the weights of the frequency components of the signal in which it has been split) of seismic noise measured in the horizontal component (H) and vertical (V) for a certain time interval.

The basic idea of the procedure is that the relationship between the spectral components allows to eliminate the role of the source, hypothetically present to the same extent in both the vertical and the horizontal component, allowing to determine the filtering action of the soil on the seismic waves propagation. It was verified that the performance of the spectral ratio between the horizontal and vertical components of the tremor (H/V function) shows the maximum at the resonant frequencies for the S waves generated by seismic contrast impedance in the subsurface. The determination of these resonant frequencies plays an important role in seismic microzonation (Kramer, 1996) and provides important information about some characteristics of the velocity profile of the S waves.

It's important to note that the analysis of the spectral ratios H/V, by itself, is not sufficient to characterise the complexity of the seismic site effects and especially it's not able to determine, by itself, the absolute value of the seismic site amplification. However the HVSR method has proven to be suitable for evaluating the fundamental period of sedimentary deposits (particularly when the impedance of these is in great contrast with the impedance of the bedrock). Therefore the main application of spectral ratios technique is to allow a fast and quite simple, but not always completely reliable, identification of the fundamental frequency f_0 of different areas.

The HVSR technique allows an evaluation of the fundamental resonance frequency of a soft layer analysing the H/V function of the tremor acquired at the surface. Considering the typical geological structure of a sedimentary deposit (fig. 3.1), the tremor recorded at the surface can be considered as composed by surface waves and body waves. These waves are modified by the filtering action of the soft layer. Two spectra can be defined as related to measures of surface horizontal motion (H_f) and vertical motion (V_f). These spectra are related to the body waves spectra and surface waves spectra by the following formula:

$$\begin{cases} H_f = A_h \cdot H_b + H_s \\ V_f = A_v \cdot V_b + V_s \end{cases}$$

where A_h and A_v are the amplification factors of the horizontal and vertical motion of the body waves. H_b and V_b are the horizontal and vertical spectra of the motion at the bedrock and H_s and V_s are the horizontal and vertical spectra of the motion at the surface.

1

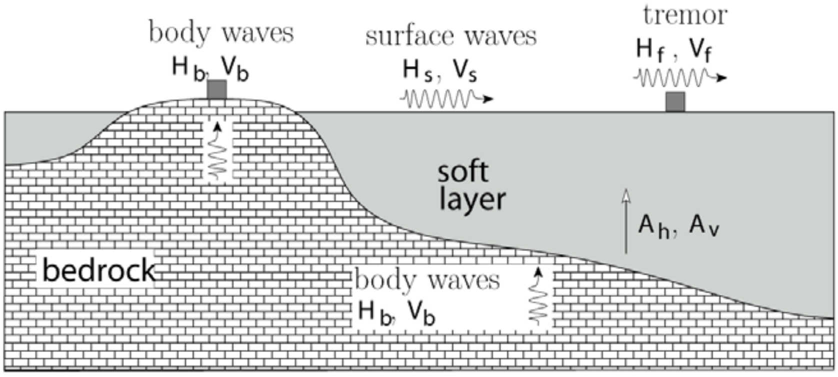


Figure 3.1: Simplified structure of a sedimentary basin and entities involved in the definition of the QTS.

Nakamura then defined the QTS (Quasi-Transfer Spectra) as:

$$QTS = \frac{H_f}{V_f} = \frac{H_b}{V_b} \cdot \frac{A_h + \dfrac{H_s}{H_b}}{A_v + \dfrac{V_s}{V_b}}$$

Fig. 3.2 proposes a comparison between the horizontal component H_f, the vertical component V_f and the spectral ratio QTS.

2

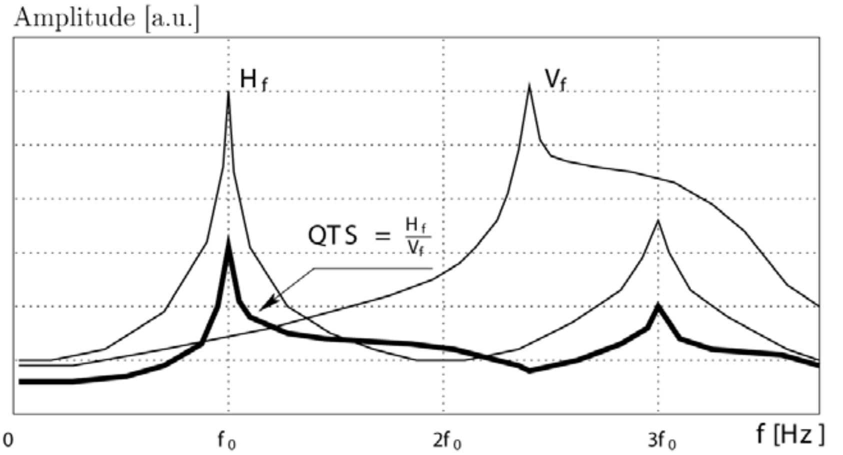


Figure 3.2: Comparison between the horizontal component H_f, the vertical component V_f and the spectral ratio $QTS = {}^{H_f}/_{V_f}$ (loosely based on Nakamura [2000]).

Assuming that the wave field that characterises the tremor consists of a combination of seismic waves (P, S and surface waves), to better understand the basis on which the HVSR technique is based on, it's useful to analyse separately the contributions of the volume waves and surface waves:

- The volume waves coming from remote sources reach the site from the bottom with angles of incidence close to the vertical. For these waves, the amplitudes of the horizontal and vertical components of the motion are controlled respectively by the motion phases longitudinal (P) and transverse (S).
  Due to the attenuation suffered during the journey, the contribution of these waves to the total field is presumably little if not at the lowest frequencies of vibration. The major contribution of the phases related to the volume waves would then come from local sources (traffic, industrial activities, etc.), probably randomly distributed around the site and acting independently. Only these local sources of noise are able to provide a strong contribution of volume waves in the high frequency range.

- Surface waves are generally characterised by a lower geometric attenuation, preserving considerable amplitudes even at a considerable distance from the source. So both the surface waves next to the area of measurement as remote sources take part in composing the seismic tremor. The characteristics of the ground motion associated to the surface waves depends on both the characteristics of the source and the characteristics of the subsurface between the source and the point of measure.
  Assuming the existence of many sources that act independently, it's reasonable to assume that, on average, the contributions of the different horizontal and vertical sources are equal, so, at a given time, the probability that the main stress is related to the horizontal motion is equal to the probability that the main stress is related to the vertical motion. Considering numerous independent sources, it is reasonable to assume that the "equivalent" source provides the same average contribution to the vertical and to the horizontal components of the motion.

Starting from these assumptions, the ratio between the vertical components (Rayleigh waves) and horizontal components (Rayleigh and Love waves) of the motion is conditioned only by the characteristics of the subsurface along the propagation path. In particular, if the soil is characterised by flat layers parallel and arranged horizontally, the ratio between the vertical and horizontal components of the motion is determined only by the velocity profile in correspondence of the measurement point.
Sources randomly distributed around the measurement point produce surface waves coming from different directions that are summed in the measurement point, then any form of phase coherence and polarisation is lost (that is commonly observed in practice). However this does not influence the amplitude ratios between the horizontal and vertical components of the motion, which is instead controlled by the local velocity profile. This makes the average ratio H/V of surface waves a parameter sensitive to the local structural conditions of the soil and thus substantially independent from the sources, allowing to correlate the shape of the function H/V to the soil characteristics at the measurement point.

3.2.1 The quarter wave law

Considering an ideal site (fig. 3.1) characterised by the presence of a soft layer above the bedrock, when seismic waves pass through the soft layer to reach the surface, they are modified according to the mechanical and geometrical characteristics of the layer. If the ideal soft layer is assumed to be homogeneous, elastic and isotropic, it's possible to calculate the fundamental resonance frequency according to the quarter wave law:

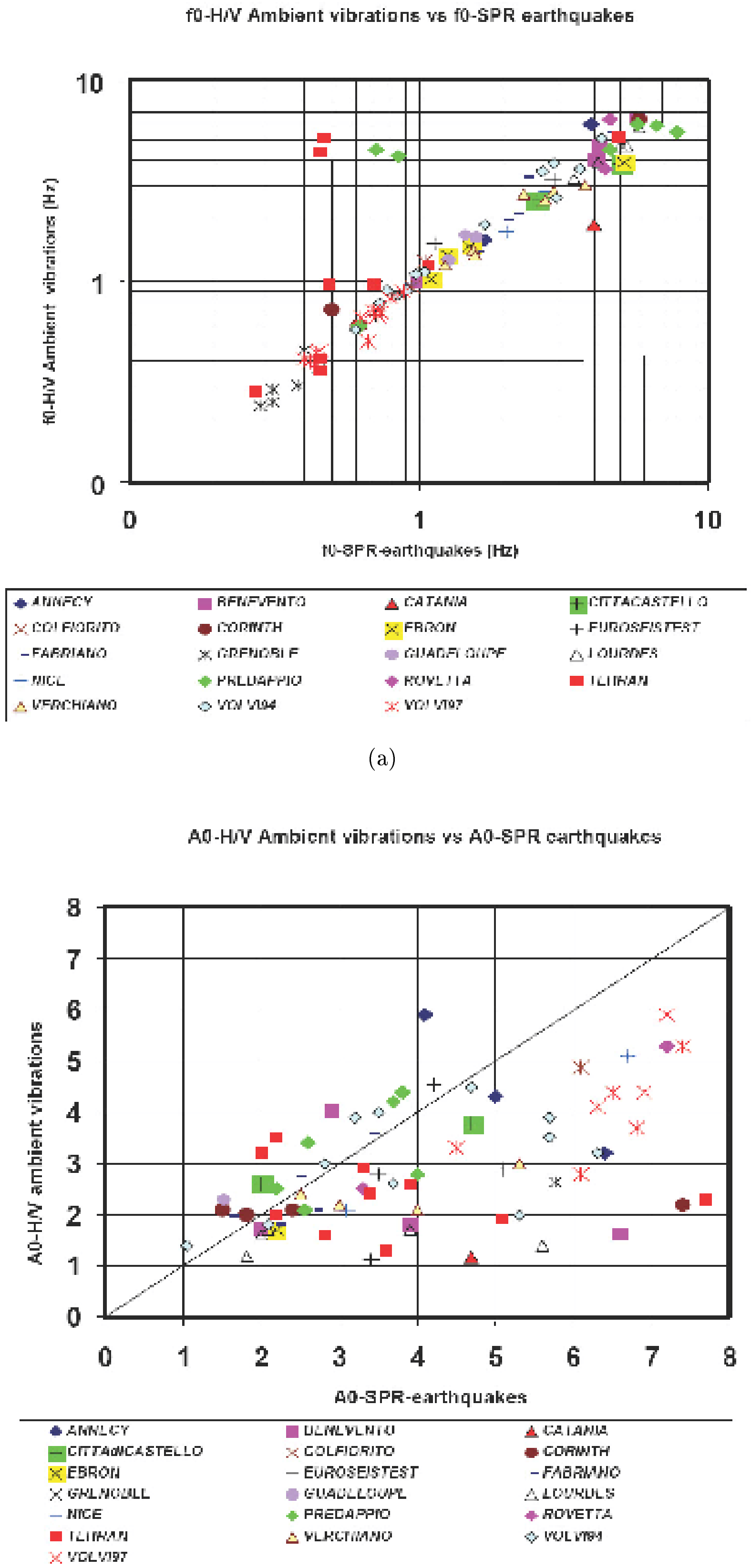
$$f_0 = \frac{V_s}{4 \cdot H} \tag{3.1}$$

where V_s is the mean velocity of the S waves and H is the depth of the soft layer.

The quarter wave law puts then in relation the natural frequency of resonance f_0 with the thickness H of a soft layer placed above a much more compact soil (bedrock) considered as a semi-infinite solid.

Assuming that the wavefield is dominated by surface waves, it appears that also the tremor spectral ratio H/V is essentially controlled by such waves and in particular by the ellipticity of Rayleigh waves [Tokimatsu, 1997]. In general it can be observed the presence of pronounced maximum in the H/V function at the resonance frequency f_0 of the S waves in the soft soils. In first approximation the value of the peak frequency is related to the ratio between the average speed V_s of the S waves in the soft layer and its thickness H, always according to eq. 3.1.

This interpretation also provides a justification for the lack of correspondence that is found in some cases between the amplitude of the maximum in the H/V function and the magnitude of the amplification of the seismic motion in correspondence of the resonance frequency (fig. 3.3).



(a)

(b)

Figure 3.3: Comparison between the H/V ratio obtained from environmental tremor and from seismic events (strong motion field). The graph *3.3a* shows the comparison of the frequencies f_0, the graph *3.3b* shows the comparison of the amplitudes $A(f_0)$. Loosely based on SESAME [2005].

3.3 A SOM analysis for Salta city

The city of Salta is located in the northern sector of the Lerma valley. The geomorphological characteristics of the area are principally a consequence of the presence of the Eastern Cordillera thrust. The hills that enclose the valley are formed mainly by outcrops of the bedrock dating back to the Superior Precambrian - Lower Paleozoic, while the Cenozoic rocks outcrop mainly within the valley and they are often fractured and with different immersion angles. Finally, the Quaternary deposits occupy the depressed area.

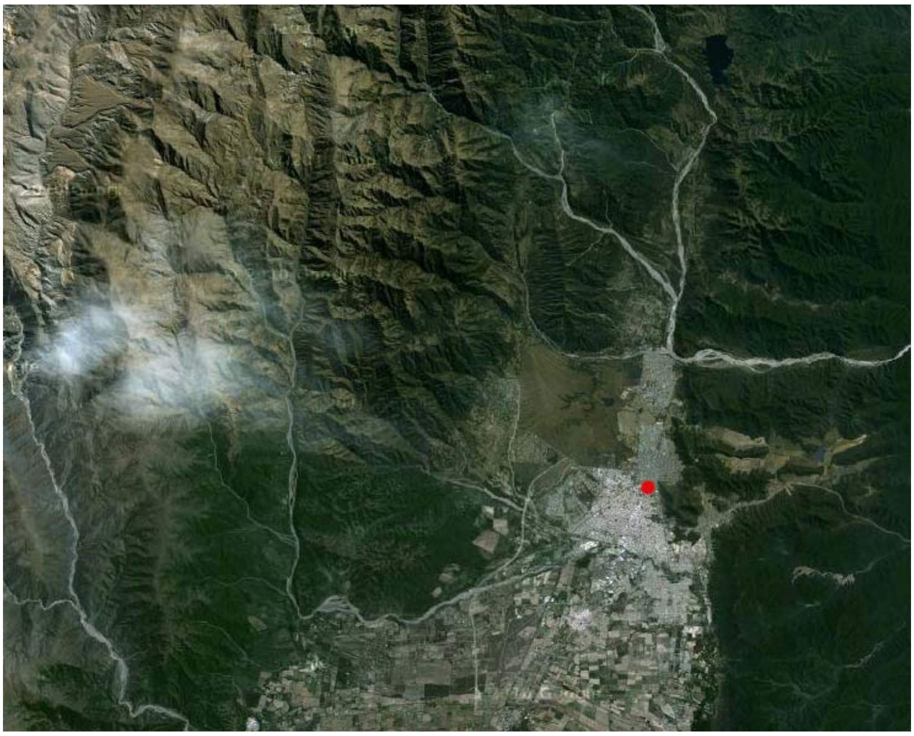


Figure 3.4: Satellite photo of the valley of Salta. The red dot indicates the area involved in the acquisitions of environmental tremor (from Google Earth).

This application of the SOM process can be considered an enhancement of the previous SOM analysis procedure [Carniel et al., 2009] to improve the HVSR technique for the seismic site response evaluation. Applying SOM to several seismic noise records, some problems and considerations have arisen about data pre-processing, SOM learning process and results interpretation. In the meanwhile, the National University of Salta (Argentina) is carrying out a project of microzonation of the city of Salta, since the National Institute for Seismic Prevention considers the seismic hazard of particular interest among the natural hazards for the city and its district, given the high population density of the area and the decisive influence of site effects due to the geo-morphology of the valley. As a case study, SOM has therefore been applied to the short and highly noisy seismic tremor acquisitions of Salta city, with the objective of an easier and more reliable determination of the H/V function's peak.

The seismic characterisation of the site is converted into a data cluster analysis where the proposed SOM process is applied to horizontal to vertical spectral ratios. The basic idea is to handle separately the spectral ratios in east–west (EW) and north–south (NS) directions and, since the main frequency is a feature of the site, it should be always present in all spectral ratios, unlike frequencies due to noise, which generally cover limited time intervals of the records and potentially imply incorrect peaks estimation of the H/V function.

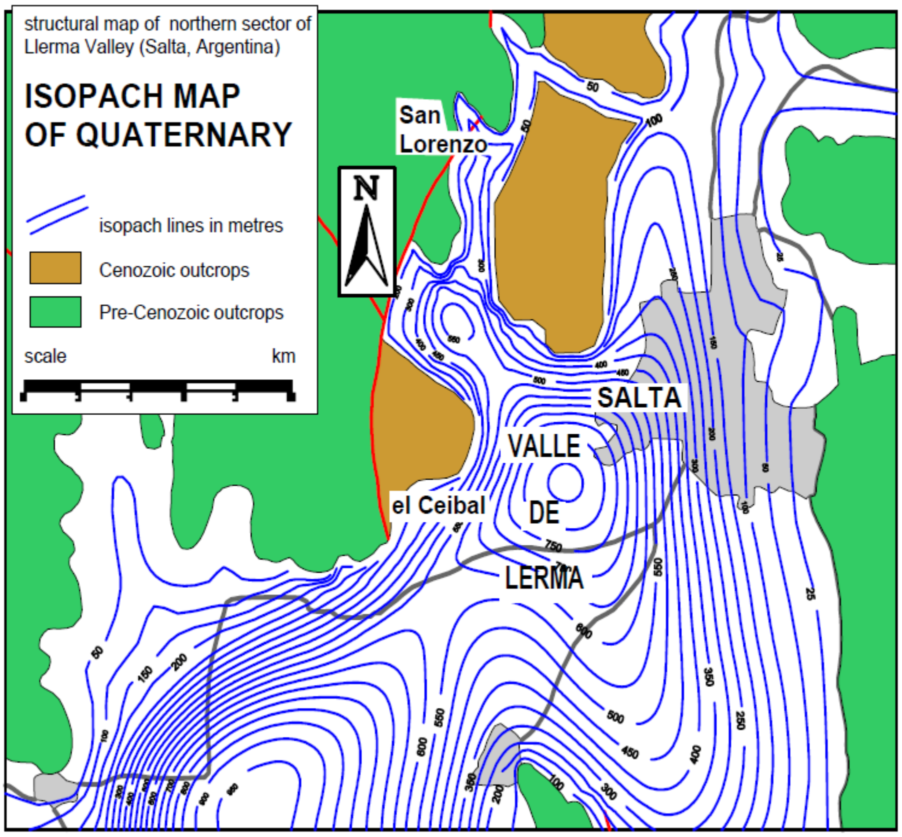


Figure 3.5: Isopach map of the Lerma valley (loosely based on ?).

The main improvements of the process regard both the procedure and the optimisation of the algorithms to reduce the computational effort and then the analysis duration. Briefly, in order to estimate the amplitude spectrum, the classic FFT has been replaced by the Welch's method [Welch, 1967] implementation [Lees, 2012] that allows to damp the noise effects, reaching also a good balance between time and frequency resolution (*evolfft* function, *RSEIS* package, Team [2010]). Moreover, Konno-Ohmachi smoothing function [Konno and Ohmachi, 1998] has been implemented. In the learning process, the discriminant function is based on a weighted cross-correlation index instead of the Euclidean distance. This allows to better consider the spectral shape rather than punctual H/V function values. Moreover, during the training, a proper choice of the frequency range allows to compare patterns only in the range of interest, optimising the map organisation and obtaining a more meaningful data cauterisation. With regard to topological properties, as previously discussed, flat maps have the disadvantage that neurons along their edges do not have the same number of neighbours as the others, resulting in a non homogeneous training process. The solution is to use a toroidal map, removing then the edges of flat SOM maps.

To comply with the objectives of the research program, one of the most significant information is to define the site frequency response, in order to evaluate the vulnerability of buildings on the basis of the type of construction and thus obtaining an assessment about the seismic hazard of the area. The HVSR technique should allows to highlight in a simple and economical way the fundamental period that characterises the sedimentary deposits of a certain area, without taking into account any information about the soil structure and without obtaining a reliable information about the effective site amplification in correspondence of the identified fundamental frequency.

The data acquisition has been carried out using a three-component seismometer Mark L4

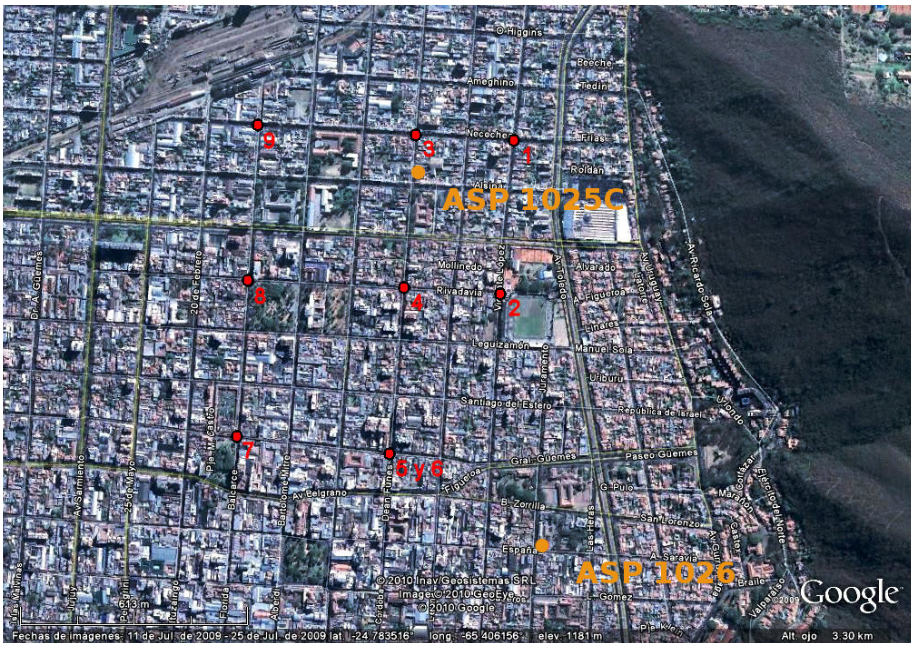
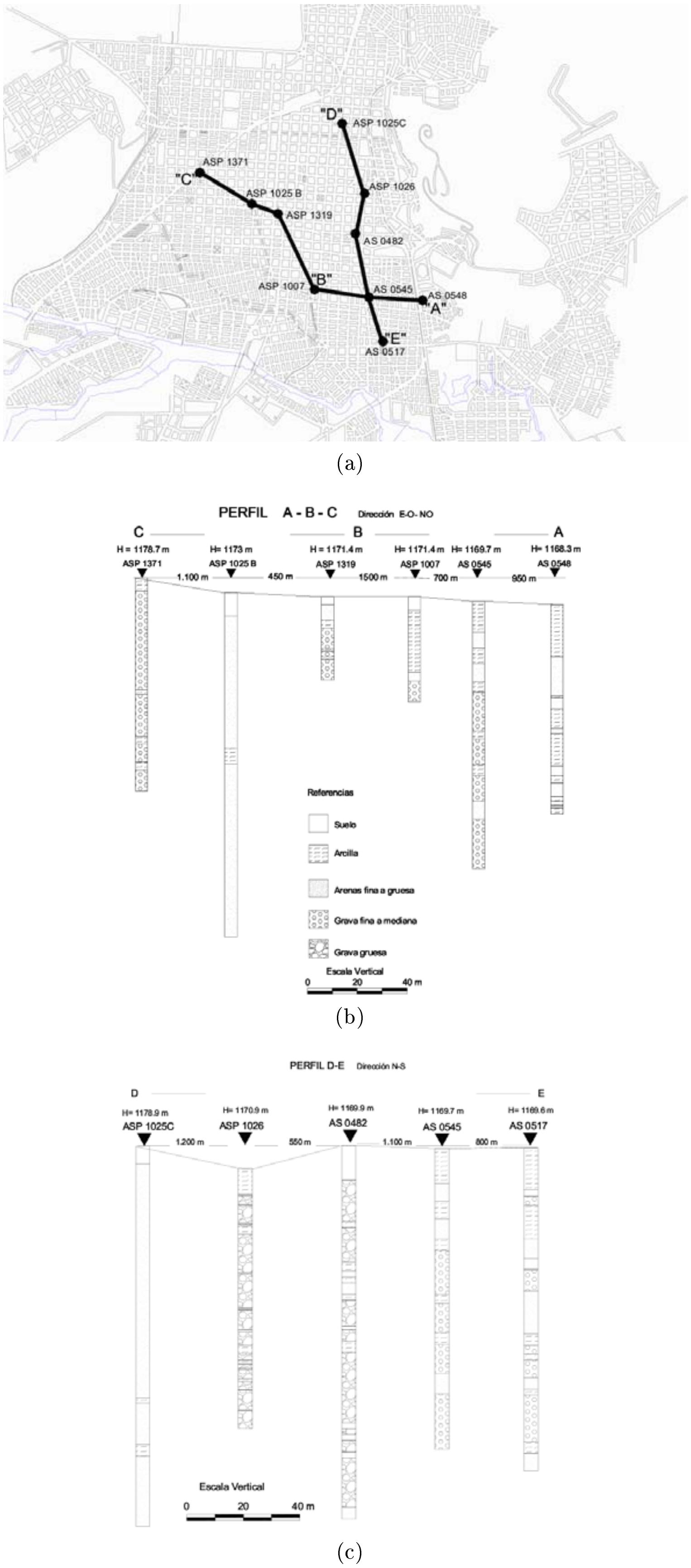


Figure 3.6: Red dots show the measurement points where the three-component seismometer Mark L4 - 1sec has been placed to acquire the environmental tremor. Each number correspond to the seismic station number in the following presentation of the SOM results. Yellow dots show the position of a pair of the available drillings (see fig. 3.7).

with own period of 1 second. For each measuring point, the acquisition has lasted for 3 minutes with a sampling frequency of 50 Hz. The measurement points have been arranged in a square grid of constant side equal to 300 m to define a detailed map showing the soil fundamental frequency response.

Researchers of the National University of Salta involved in the project of seismic microzonation project have encountered difficulties in the identification of the peak of the H/V function, in particular for some time series acquired at certain points of the grid. The HVSR technique can not provide satisfying results because of some negligence during the tremor acquisition phase, e.g. a poor soil-sensor coupling, but also environmental and weather conditions strongly influence the results, e.g. anthropogenic noise, blowing wind, wet soil etc. [SESAME, 2005].

In the case of Salta time series, the high anthropogenic noise, the density of the buildings and the short duration of the acquisitions definitely play a role. In order to provide an example of how the proposed SOM procedure is able to provide additional information with respect to the classical data analysis techniques, in the following are showing some results obtained by SOM analysis applied to the worst time series for which the interpretation of the H/V function peak is particularly uncertain. Fig. 3.8 shows the seismic microzonation proposed by the research group of the National University of Salta.



(a)

(b)

(c)

Figure 3.7: Available drillings in the valley of Lerma (as supplied by the National University of Salta).

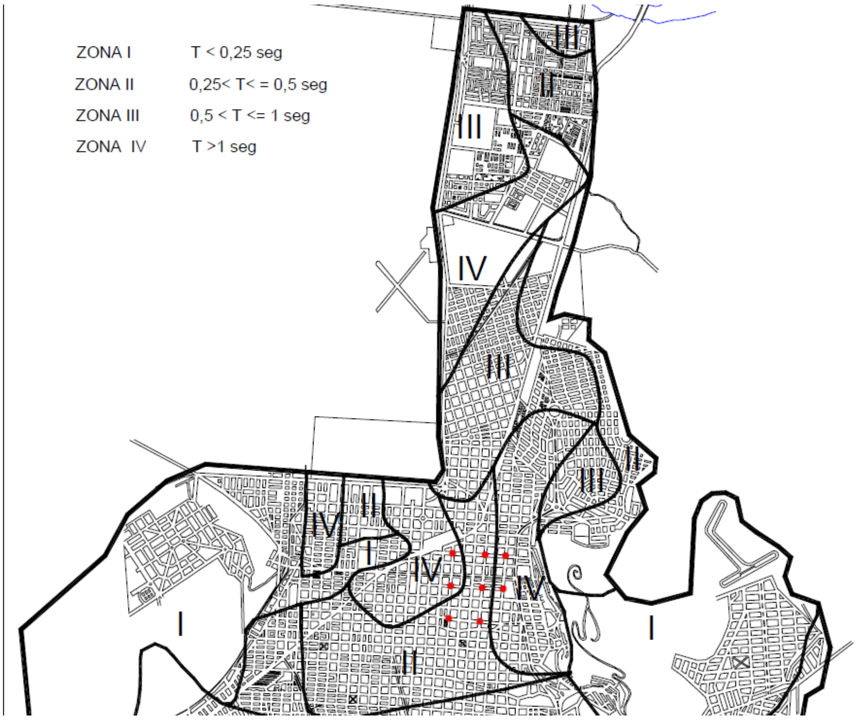


Figure 3.8: Microzonation map of the Lerma valley area on which the city of Salta is located (as supplied by the National University of Salta). Red dots indicate the measurement points (see also fig.).

3.3.1 Stations 5 & 6

As a first example, in the following the SOM analysis for the station 5 & 6 (at the same measurement point) is discussed. For the other stations (see fig. 3.6) the same procedure has been applied, with the same SOM parameters (SOM map dimension, learning rate and neighborhood radius, Konno-Ohmachi smoothing function value, number of process iterations etc.).

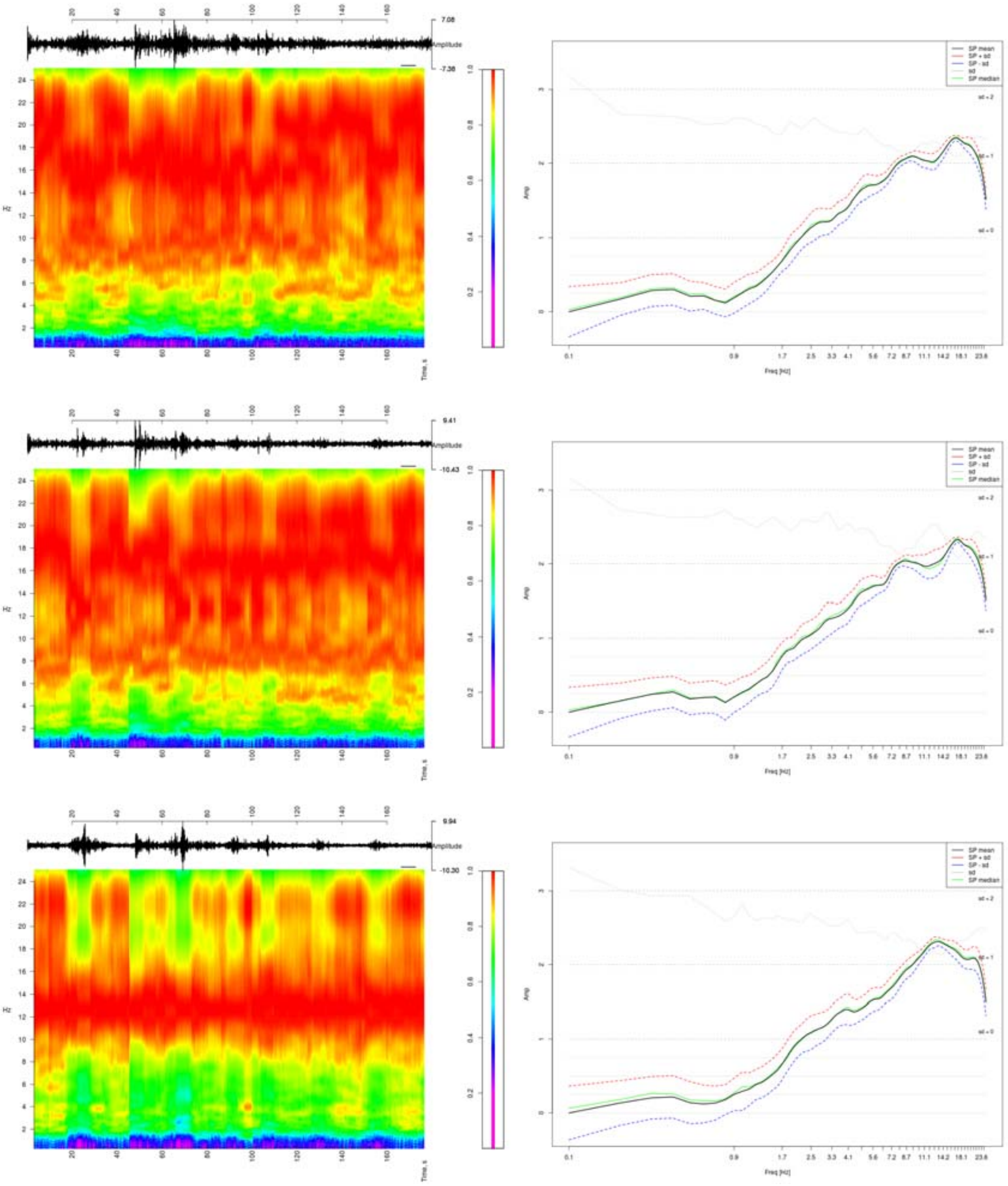


Figure 3.9: Station N6 - spectrograms for the EW, NS and UD component respectively and the corresponding averaged spectrum. It's easy to note the high content of anthropogenic noise.

In order to estimate the amplitude spectrum (fig. 3.9 shows the spectrograms in EW, NS and UD direction respectively) the Welch's method has been used with 1024 points in the FFT windows (*Nfft* parameter, *evolfft* function, *RSEIS* package), 320 points in the sub-windows (*Ns* parameter, *evolfft* function, *RSEIS* package) and 305 points of overlap

(*Nov* parameter, *evolfft* function, *RSEIS* package, Team [2010]). Subsequently the Konno-Ohmachi smoothing function has been applied to the spectra, with the *b*-value equal to 40. As the SESAME project [SESAME, 2005] suggestes, in the case of industrial origin tremor, the H/V peak should become sharper and sharper reprocessing the data with less and less smoothing, while this is not the case for a "site effects" peak linked with the soil characteristics. For the local narrow peaks that have an industrial origin, the reprocessing with different smoothing parameters shows it becomes narrower and narrower, with a larger and larger amplitude when the *b*-value (Konno-Ohmachi smoothing function parameter) is increasing.
Horizontal components rotation has been applied to investigate the differences between differently polarised horizontal components, since along valley edges a clear differences may appear between the parallel and transverse components with relation to the elongation valley axis. The rotation angle is the one that maximises the dissimilarity between the horizontal spectra components (EW vs NS).

Then the H/V spectral ratios have been calculated (fig. 3.10). Each one spectrogram (EW and NS component) is handled separately and each one spectra is an input vector for the SOM process.

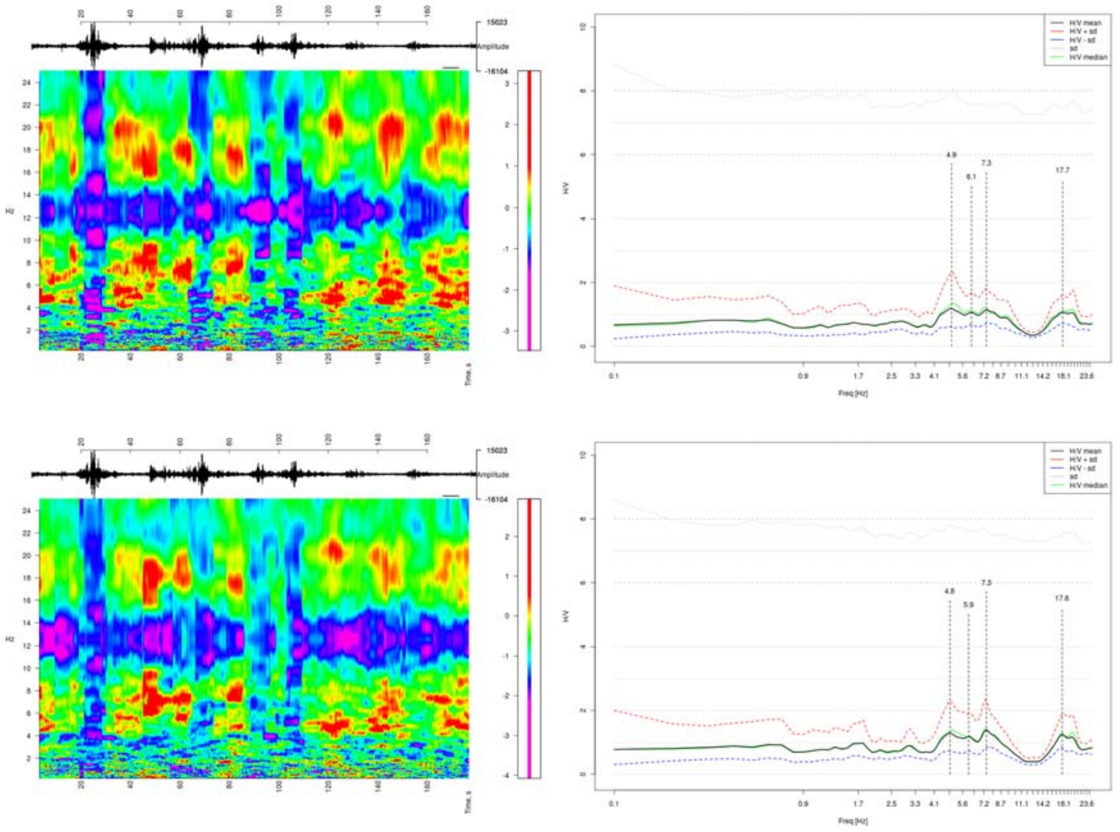
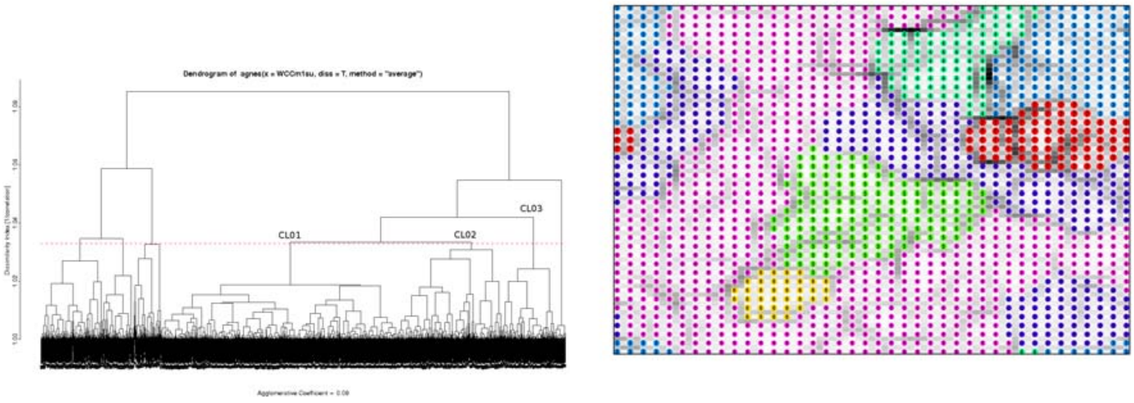


Figure 3.10: Station N6 - H/V spectrograms for the EW and NS component respectively and at the right side the corresponding mean spectrum is shown.

The SOM map has 1600 neurons (or grid nodes) and it's square and with a toroidal shape for the computational purposes, so it has any edges. During the learning process phase, the discriminant function is based on the weighted cross-correlation index instead of the Euclidean distance, as previously discussed. In order to analyse the spectra only in the range of interest, it was considered appropriate a choice of the frequency range between 0.7 Hz and 7 Hz, optimizing the map organization and obtaining a more meaningful data

clusterization.
The neighborhood radius starts from a value of 40 equal to the SOM map sides and it decreases over iterations to the value of -40, so after an half of iterations each input vector activates just the corresponding BMU on the map and its neighbours code vectors are not modified. The learning rate starts from a value of 0.15, decreasing over iterations to reach the final value equal to 0.001. The SOM training process lasts for 100 iterations, at each iteration the whole dataset is processed by the network.

Fig. 3.11 shows, for the EW component (the rotation of the horizontal component has not been applied in this case), the dendrogram (fig. 3.11a) obtained computing the weighted cross-correlation matrix among the code vectors of the neurons on the SOM grid. Choosing a threshold, corresponding to a certain dissimilarity value, a certain number of clusters is detected onto the map. In this case the threshold is placed at a dissimilarity value equal to 1.033, corresponding to seven clusters onto the SOM map (fig. 3.11b).



(a)

(b)

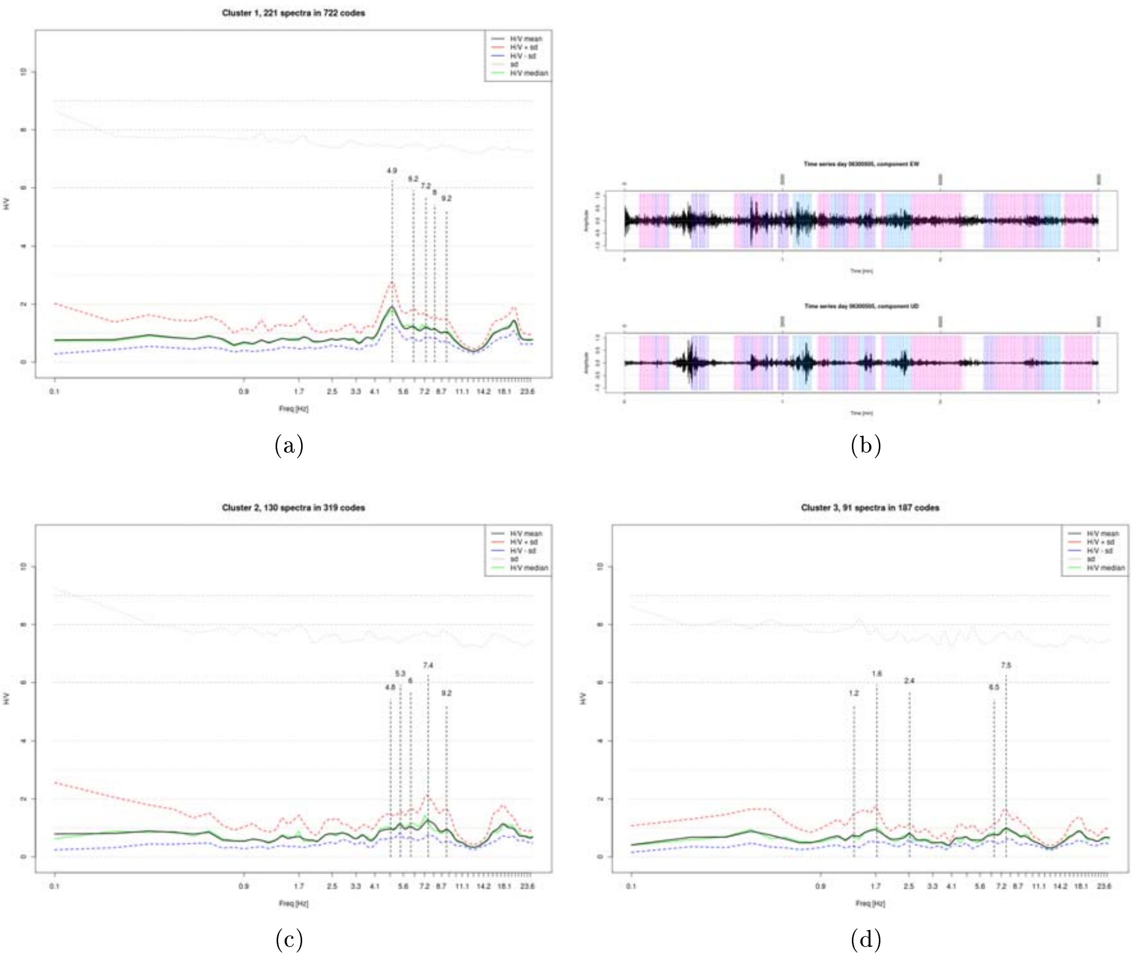
Figure 3.11: Station N6, EW component. Fig. 3.11a shows the dendrogram obtained computing the weighted cross-correlation matrix among the code vectors of the neurons and the clusterized SOM map (3.11b) that shows the seven detected clusters for the chosen dissimilarity threshold on the dendrogram.

The choice of the dissimilarity threshold is not unique and this provides another degree of freedom in the SOM analysis results, since there isn't a single optimal threshold for any dataset. Choosing a low threshold increases the number of clusters, and each of them becomes highly specialized to recognize a particular feature that caracterises the dataset. For this reason it is advised to carry out a few analyses with different thresholds before choosing the threshold that best optimises the coexistence of an overall view on the main features of the dataset and an acceptable "resolution" in terms of distinction between different spectral patterns.

The last step of the proposed SOM analysis consists in considering each one cluster or at least the main clusters which have the great number of neurons and consequently collect the great number of data. The procedure allows to return back in the time domain in order to analyse at which cluster each time window, on which the corresponding spectra has been calculated, is assigned. In this way it is possible to evaluate the temporal stability of each cluster, since the fundamental frequency is an intrinsic property of the site as opposed to the environmental noise that is reasonably less stable.

The three main clusters for the EW component analysis have been considered and shown in fig. 3.12. The cluster 1 is the one that occupies the largest portion of the map (fig.

3.11b) and it's also the most stable in time (fig. 3.12b, pink lines show the temporal windows of which spectrum is projected into the cluster 1) and with the largest number of occurrences (fig. 3.12a). The other clusters don't show the same temporal stability and are not comparable in terms of the number of neurons on the map and the data represented.



(a)

(b)

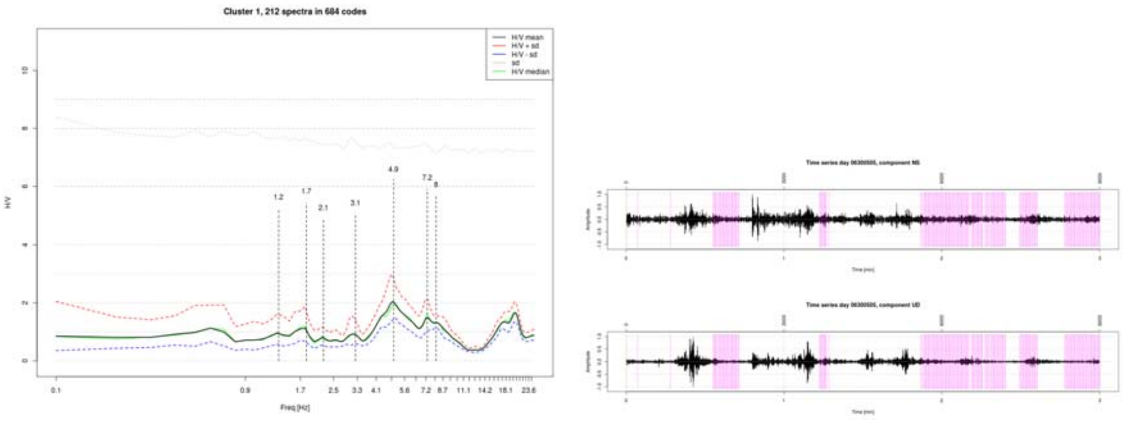
(c)

(d)

Figure 3.12: Station N6, EW component. Fig. 3.12a, 3.12c and 3.12d show the mean and the median spectrum for each one of the three largest clusters (cluster 1, 2 and 3 respectively) in the map trained with the H/V spectral ratios in EW direction. Fig. 3.12b shows data distribution over the three principal clusters.

In previous applications the SOM procedure allowed to identify the time windows with a not satisfying signal to noise ratio, since the objective was to remove these time windows to obtain spectral ratios less conditioned by transients due to environmental noise [Carniel et al., 2009]. After many applications and some improvements of the procedure, the idea of eliminating a part of the dataset to get a sub-sample composed by only the data that have presumably a better signal to noise ratio, to provide a more reliable H/V function, has been set aside. The aim of the SOM process is rather to decompose the dataset into an arbitrary number of sub-samples (clusters) of uniform features to carry on an evaluation on each one of these, e.g. in terms of frequency content and temporal stability.

The results of the same kind of analysis are presented also for the largest cluster in the NS direction (fig. 3.13). As fig. 3.12a and 3.13a show, the largest cluster in both directions is characterized by a H/V peak at about 4.9 Hz. It's reasonable then to suppose that this is the fundamental resonance frequency at that single recording point.



(a) (b)

Figure 3.13: Station N6, NS component. Fig. 3.13a shows the mean and the median spectrum for the cluster 1 in the map trained with the H/V spectral ratios in NS direction. Fig. 3.13b shows data distribution over the principal cluster.

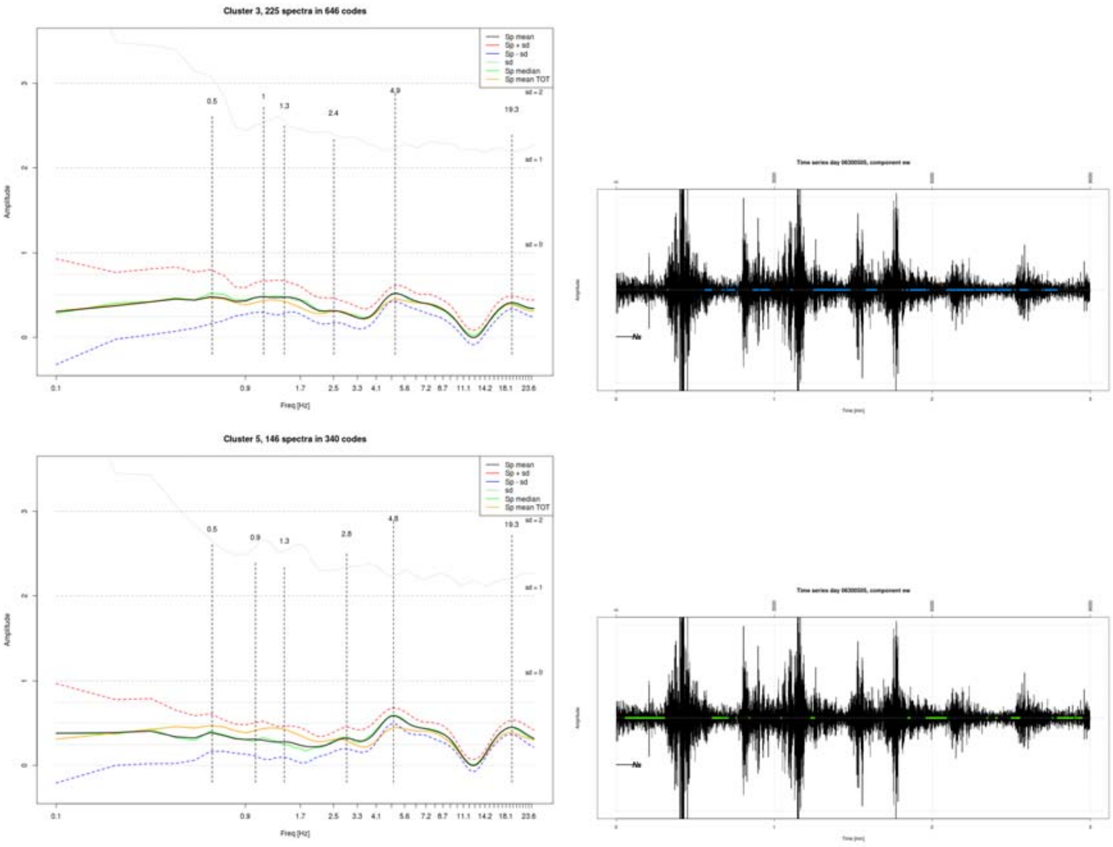


Figure 3.14: Station N6 - EW component. Figures at the left side show the mean and the median spectrum for the two largest clusters (cluster 3 and 5 respectively) in the map trained with the rotated H/V spectral ratios. The b-value of the Konno-Ohmachi smoothing function has been fixed to a value equal to 20. The other two graphs (right side) show the data distribution over the two considered clusters.

In order to verify the stability of the peak at 4.9 Hz of the H/V function, the horizontal components are rotated and the b-value of the Konno-Ohmachi smoothing function has been decreased to a value equal to 20 (a smoothing too strong according to the SESAME

[2005] guidelines). Fig. 3.14 shows that the peak at 4.9 Hz is still present and stable, it's therefore reasonable to assume that this is the fundamental frequency in proximity of the seismic station. This evaluation of the fundamental frequency response differs slightly from the evaluation proposed by the researchers of the National University of Salta (fig. 3.8).

3.3.2 Station 7

SOM analysis applied to the tremor acquired at the station 7 highlights a peak at about 3 Hz in EW component (horizontal components have been rotated by an angle equal to 60°), both in the largest cluster (cluster 1) and in the third one in order of size (cluster 3). Even if the cluster 3 has only one tenth of the number of H/V spectra projected in the cluster 1, it is quite stable in time (looking at the distribution of its spectra over time). For this reason it's reasonable to assume that the H/V peak at 3 Hz indicates the fundamental frequency for this measurement station.
In the NS component a large H/V peak (in the range 2.7 to 3.5 Hz) appears only in the cluster 5 (the third one in order of size). Since the data projected in this cluster concern only a limited time window (looking at the distribution of its spectra over time), the connotative feature of the cluster 5 can't be considered stable over time.

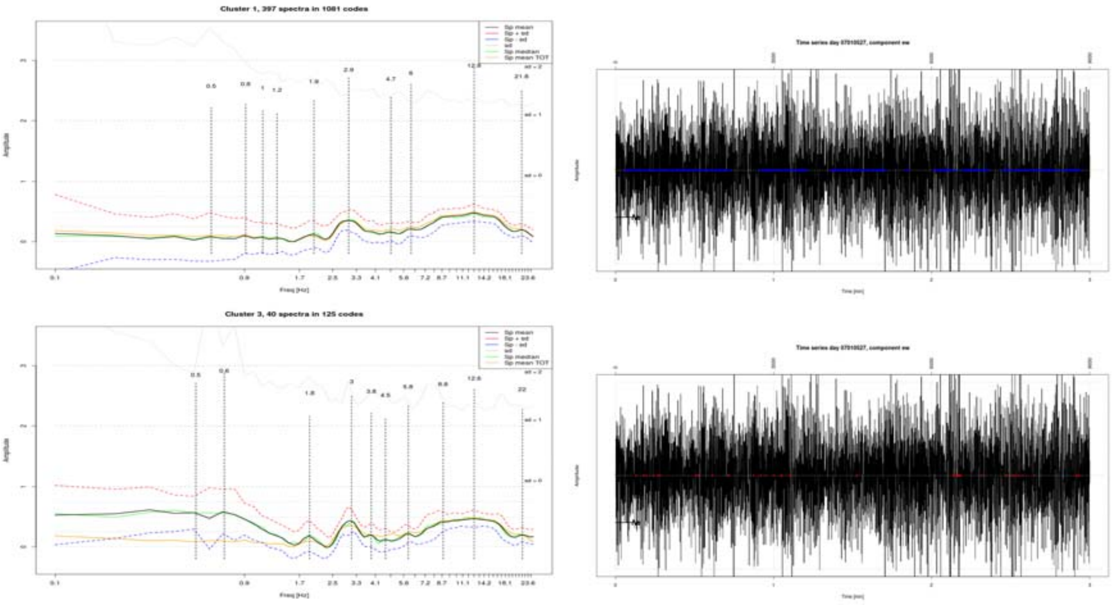


Figure 3.15: Station N7 - EW component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 1) and for the cluster 3, the third one in order of size. Cluster 2, the second one in order of size, is characterized by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters.

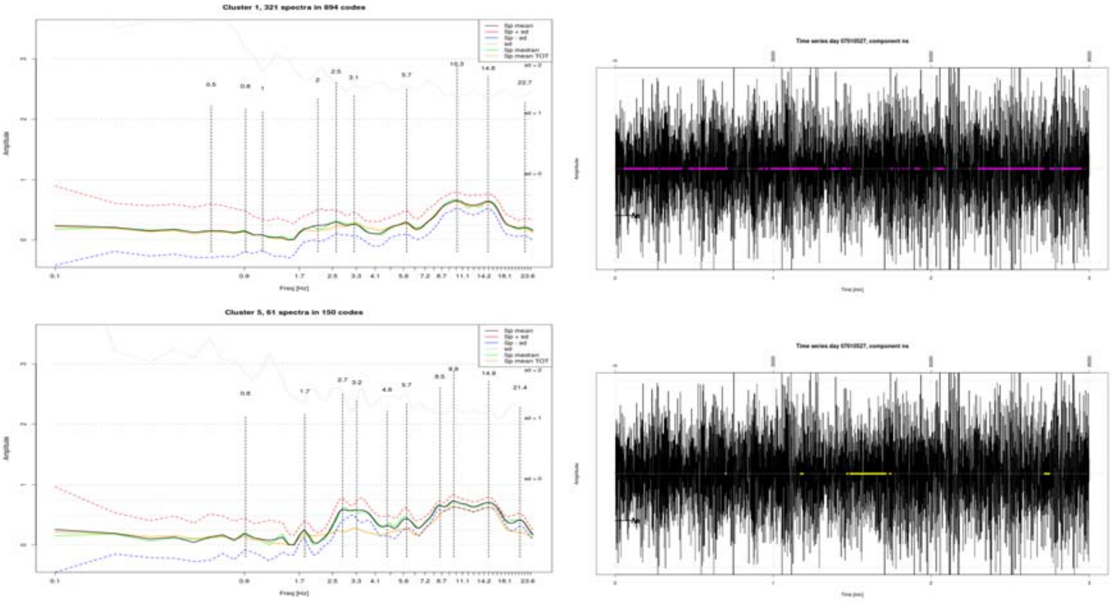


Figure 3.16: Station N7 - NS component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 1) and for the cluster 5, the third one in order of size. Cluster 2, the second one in order of size, is characterized by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters.

3.3.3 Station 8

SOM analysis applied to the tremor acquired at the station 8 highlights a quite large peak at about 2.3 Hz in EW component (horizontal components have been rotated by an angle equal to 70°) in the third cluster in order of size (cluster 1). The largest cluster (cluster 3) is characterized by a quite flat H/V function and some high frequency content. The cluster 2, the second one in order of size, is characterized by a smooth peak at 2.9 Hz and by high frequency content. The cluster 3 seems to be considered quite stable in time (looking at the distribution of its spectra over time), for this reason the H/V peak at 2.3 Hz is not devoid of interest.

Looking at the NS component something similar to the EW component can be noticed: the largest cluster (cluster 2) shows a quite flat H/V function, but the third one cluster in order of size highlights a large H/V peak (in the range 1.8 to 2.3 Hz) and it's quite stable over time.

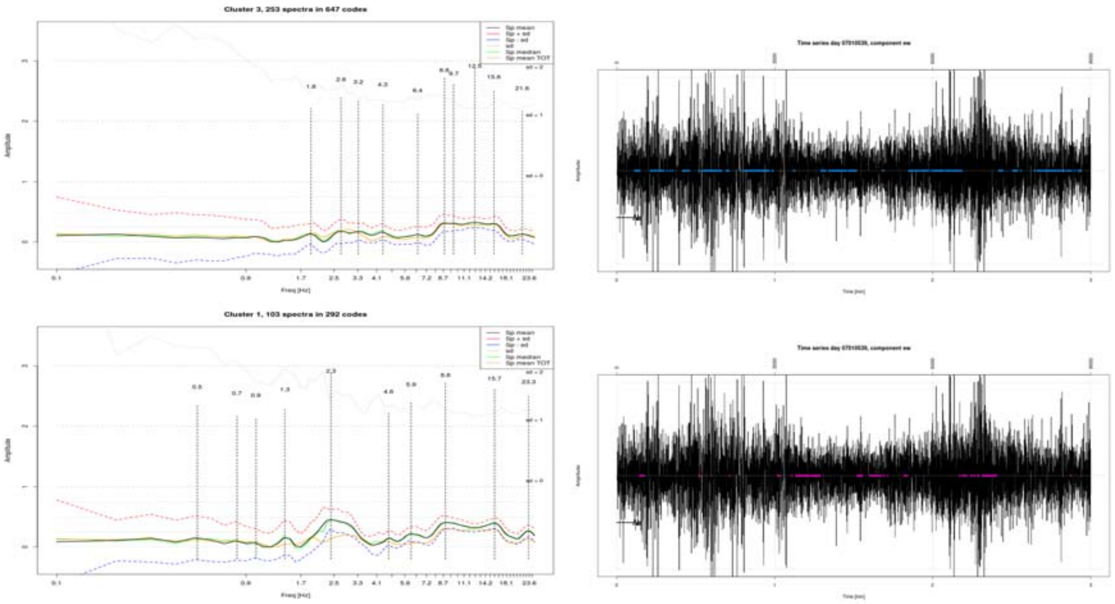


Figure 3.17: Station N8 - EW component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 3) and for the cluster 1, the third one in order of size. Cluster 2, the second one in order of size, is characterized by a smooth peak at 2.9 Hz and by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters.

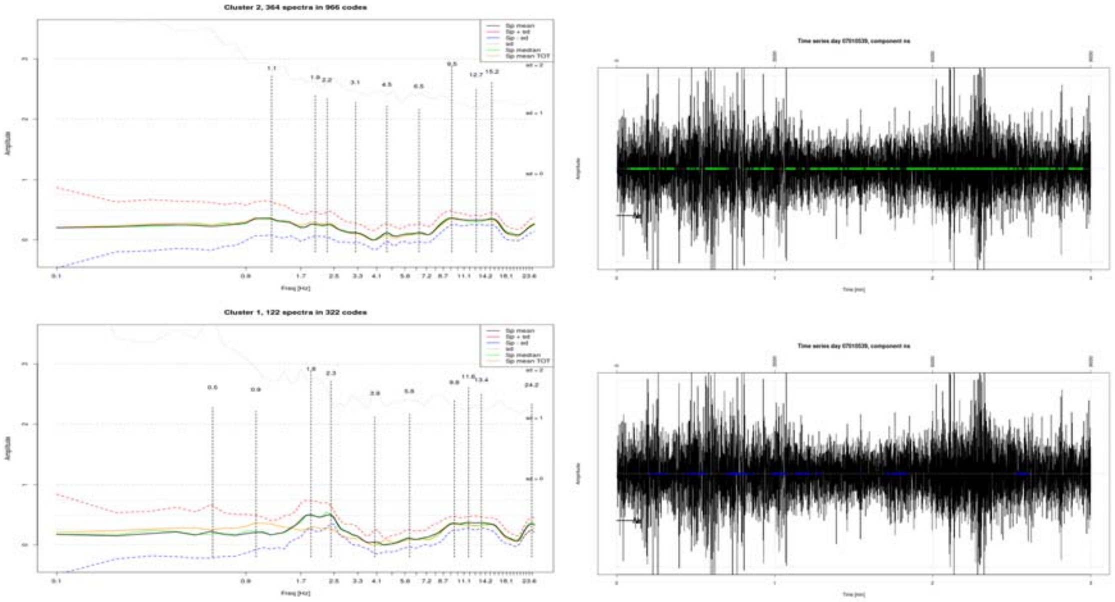


Figure 3.18: Station N8 - NS component. Figures at the left side show the mean and the median spectrum for the two largest cluster (cluster 2 and 1). Cluster 3, the only other one, is characterized by high frequency content. The other two graphs (right side) show the data distribution over the two considered clusters.

3.3.4 Station 9

SOM analysis applied to the tremor acquired at the station 9 shows not reliable results to characterize the site frequency response at the measurement station. A large flat peak centered at the frequency of 2.7 Hz and a narrow peak with low amplitude at 4.5 Hz characterize the EW component (horizontal components have been rotated by an angle equal to 40°), while a quite narrow but with low amplitude peak at 1.9 Hz characterizes the NS component.

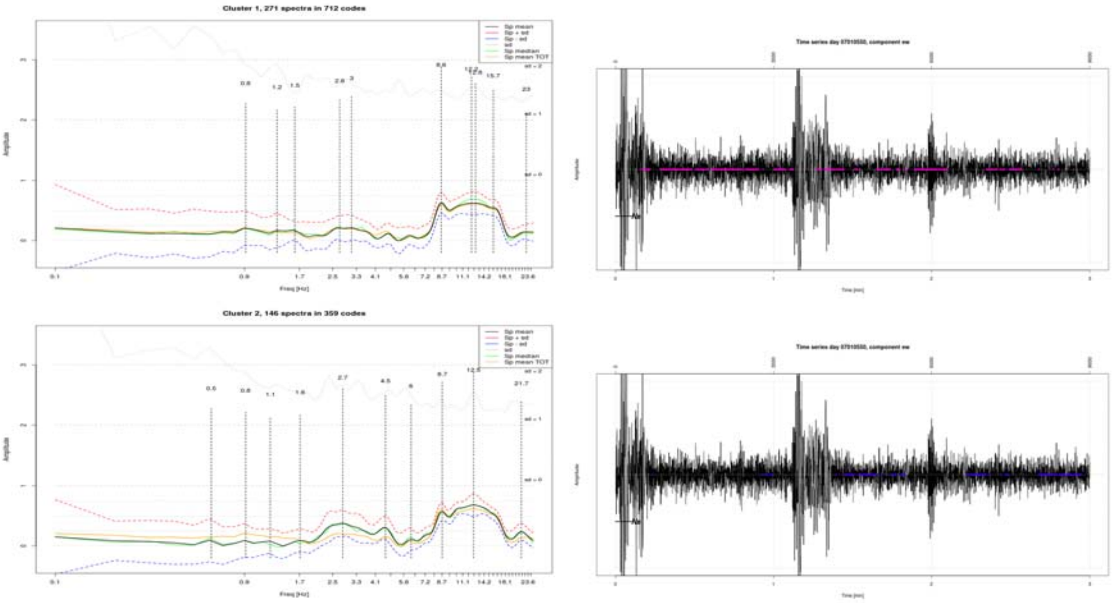


Figure 3.19: Station N9 - EW component. Figures at the left side show the mean and the median spectrum for the two largest cluster (cluster 1 and 2). The other two graphs (right side) show the data distribution over the two considered clusters.

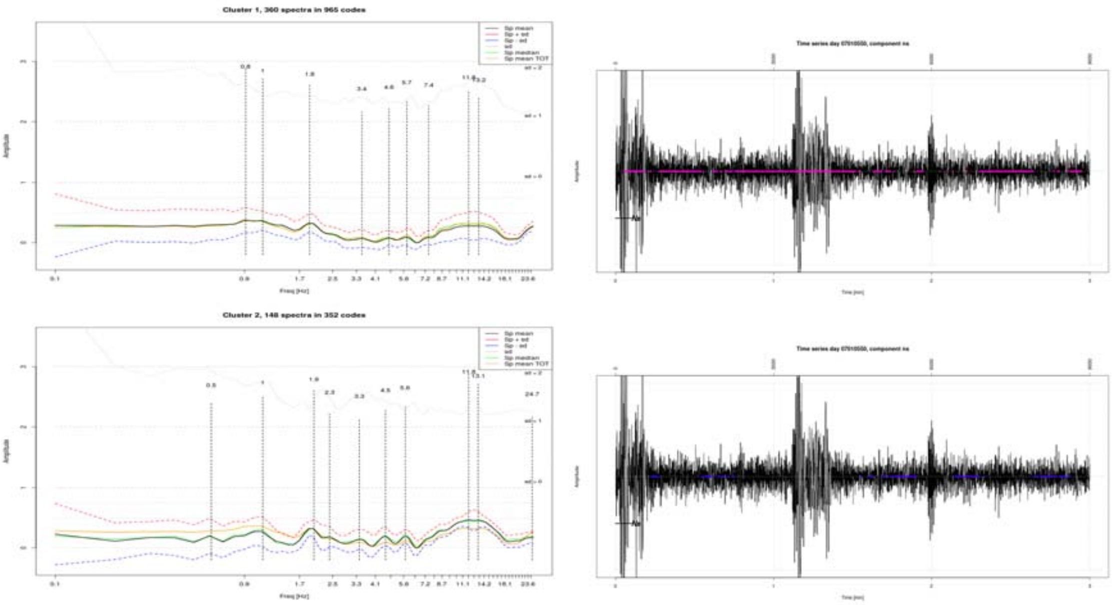


Figure 3.20: Station N9 - NS component. Figures at the left side show the mean and the median spectrum for the two largest cluster (cluster 1 and 2). The other two graphs (right side) show the data distribution over the two considered clusters.

3.3.5 Station 3

Analysing the spectral ratios of the tremor acquired at the station 3, SOM analysis shows that the EW component (horizontal components have been rotated by an angle equal to 80°) of the H/V function is characterized by a peak at about 3 Hz. In particular the spectral ratios projected in the cluster 2 enhance this feature and the cluster is quite stable over time.

SOM analysis applied to the spectral ratios of the NS component shows that the H/V function if quite flat. Lowering the dissimilarity threshold on the dendrogram to increase the number of clusters and then let each cluster becomes highly specialised to recognise a particular kind of pattern, no one of them highlights a realiable peak of the H/V function.

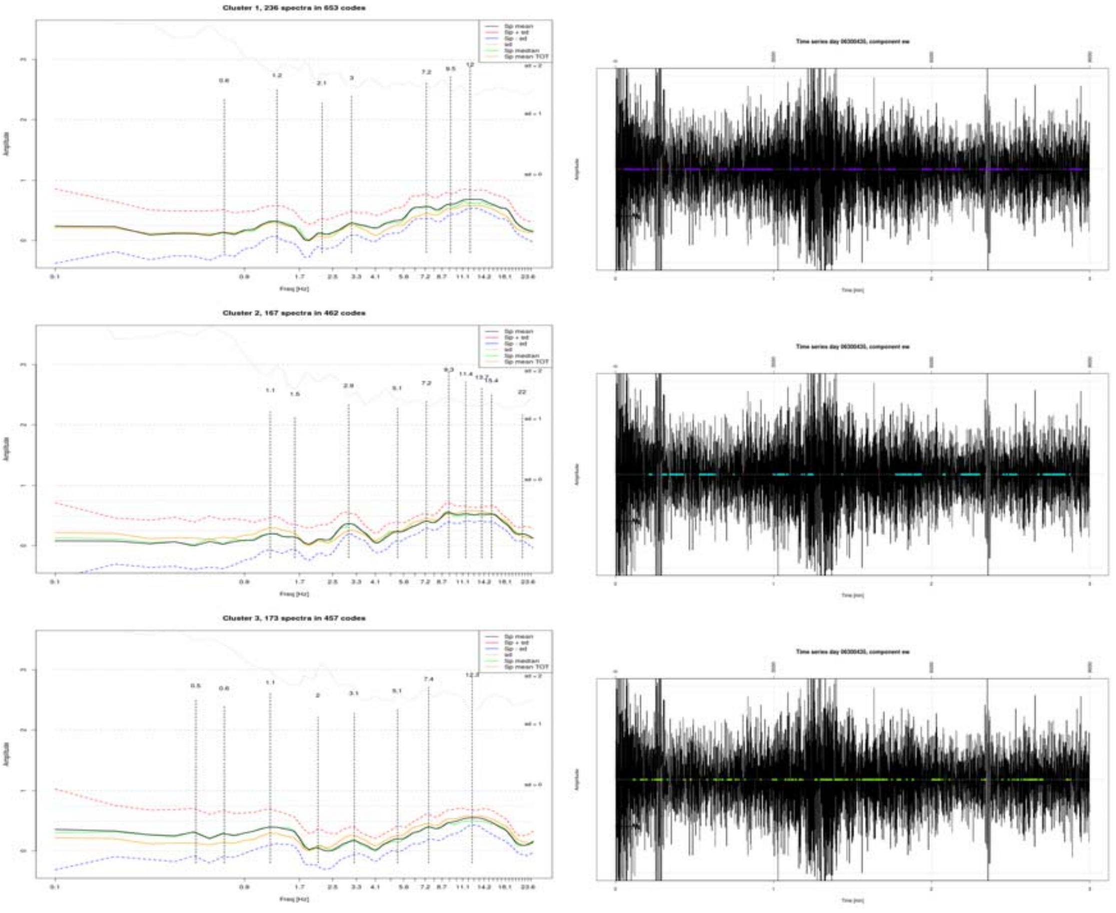


Figure 3.21: Station N3 - EW component. Figures at the left side show the mean and the median spectrum for the three largest cluster (cluster 1, 2 and 3). The other three graphs (right side) show the data distribution over the three considered clusters.

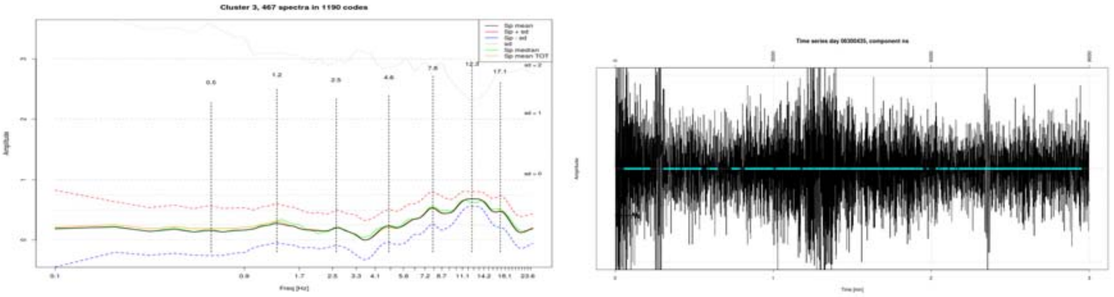


Figure 3.22: Station N3 - NS component. Figures at the left side show the mean and the median spectrum for the largest cluster (cluster 3), the only interesting one. The other graph (right side) shows the data distribution over the considered cluster.

3.3.6 Station 4

Analysing the spectral ratios of the tremor acquired at the station 4, the SOM process highlights for the EW component (horizontal components have been rotated by an angle equal to 65°) a result similar to the analysis result at the station3. The H/V function is characterized by a peak at about 3 Hz.

The largest cluster (cluster 3) in the SOM map for the NS component analysis show a large peak centered at 2.3 Hz, while the second one cluster in order of size (cluster 1) is characterized by low frequency content. Both these two main clusters highlight a low amplitude peak at 4.5 Hz. It's interesting to compare this low amplitude peak at 4.5 Hz with the peak at 4.9 Hz that characterizes the H/V funcion at the station 5 and 6.

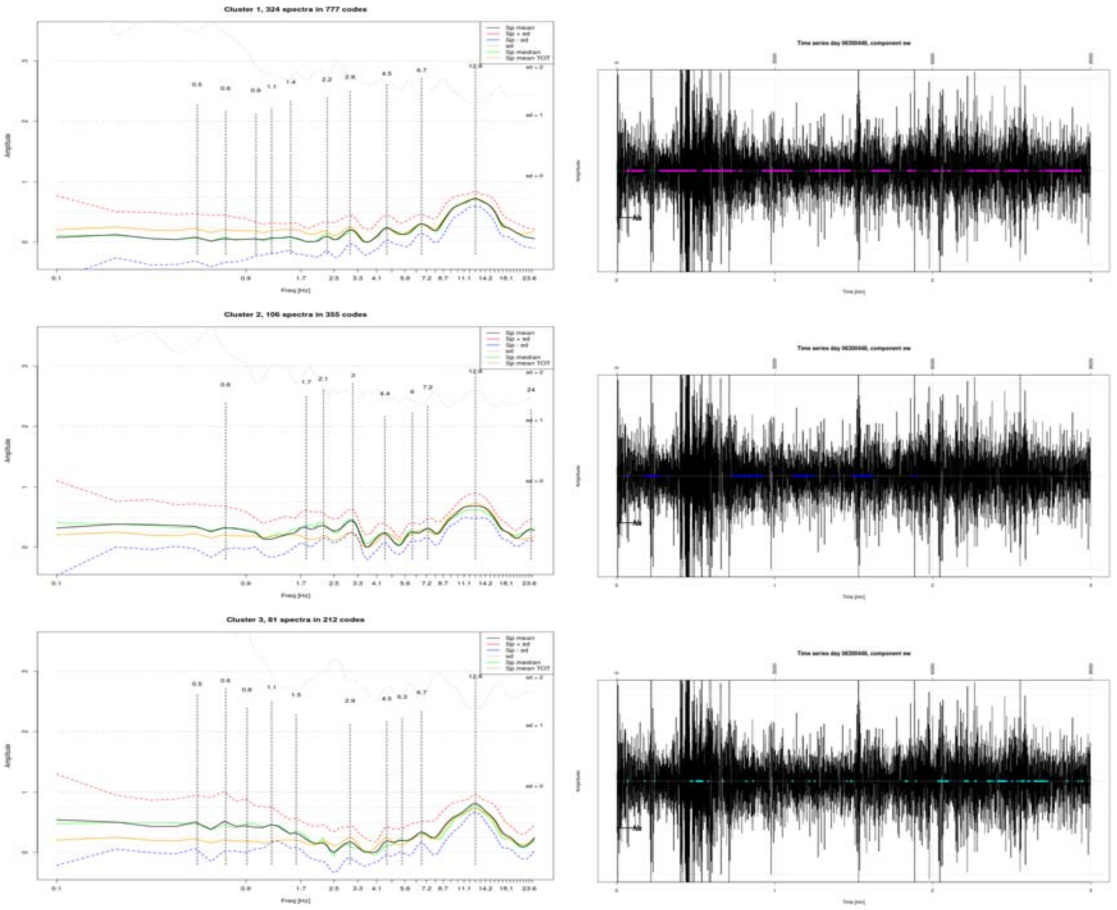


Figure 3.23: Station N4 - EW component. Figures at the left side show the mean and the median spectrum for the three largest cluster (cluster 1, 2 and 3). The other three graphs (right side) show the data distribution over the three considered clusters.

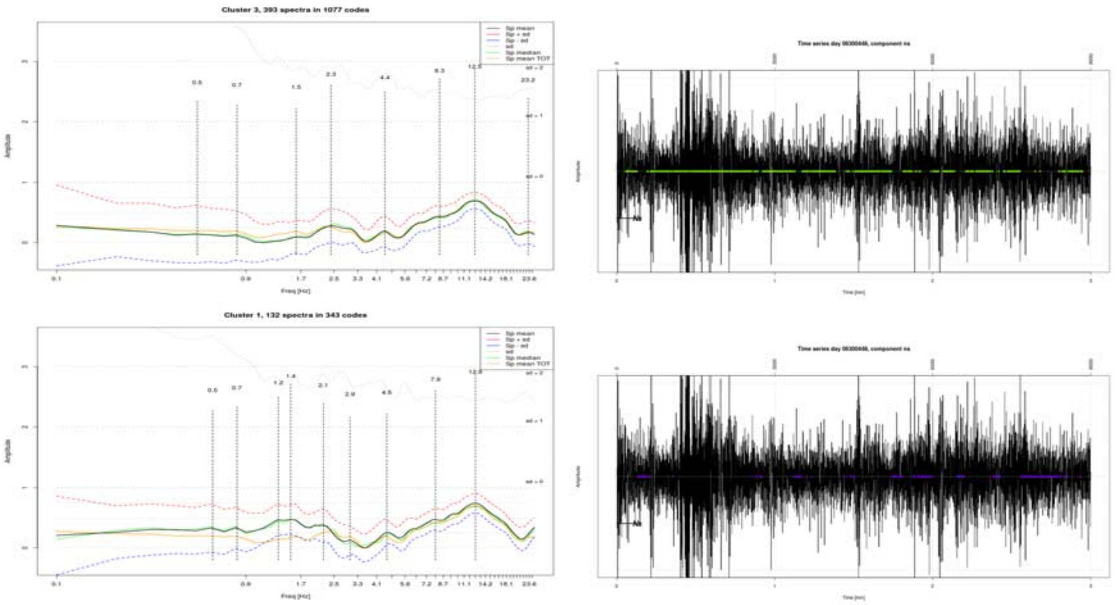


Figure 3.24: Station N4 - NS component. Figures at the left side show the mean and the median spectrum for the two largest clusters (cluster 3 and 1). The other graphs (right side) show the data distribution over the considered two clusters.

The SOM analysis discussed above does not completely confirm the microzonation proposed by the researchers of the National University of Salta for the area of interest (see fig. 3.8). In particular the low frequency range seems hard to inspect, since the tremor acquisitions are short, highly affected by anthropogenic noise and the tremor is acquired by a short-period seismometer.

Conclusions

This work aimed to investigate the possibility, both at a theoretical and practical level, to apply a particular kind of artificial neural network, the so called Self-Organizing Maps (SOMs), to improve the analysis of geophysical data. Data interpretation is one of the most important and thorny tasks in geosciences. Difficulties occur especially in non-invasive geophysical techniques and/or when the data that have to be analysed are multidimensional, non-linear and highly noisy. Another important task of the proposed SOM process is to ensure an efficient automatic data analysis, in order to allow a data interpretation as independent as possible from any a priori knowledge.

The SOM process provides a new capability to explore the input data space applying an unsupervised pattern recognition algorithm, and allows to recognise outliers as samples that have been projected onto unusual areas of the map. The analysis of the time evolution of a volcanic system through the SOM methodology offers then the possibility to highlight data patterns possibly related to a precursory activity of an upcoming volcanic crisis.

At Raoul Island, the SOM method allowed recognition of diurnal pattern in seismic data that may be anthropogenic in nature, and are not readily apparent in visual spectrogram analysis. The SOM did not reveal precursors to the eruption in Raoul Island seismicity, but it highlighted a transition in spectral characteristics at the onset of the eruptive activity into a spectral regime that lasted for approximately 1.5 hours. This frequency content transition seems to support the model proposed by Christenson et al. [2007] for which the proximal cause of the eruption was due to the failure of a shallow hydrothermal seal which became pressurised by gas released from a deeper magmatic carapace. The carapace itself is surmised to have failed due to a swarm of hybrid and volcano-tectonic earthquakes Lahr et al. [1994] occurring on March 12, which released gas from magma and caused pressurisation beneath the hydrothermal seal. If this model is correct, then the failure of the hydrothermal seal was instantaneous and included no precursors seen in either the observed spectra or the SOM analysis. The SOM analysis highlights the post-failure system excitation which is possibly due to the re-equilibration of the hydrothermal system.

By examining seismic spectra from a single station using a single-trained SOM approach, we find a consistent evolutionary pattern in two phreatic events that occurred at Ruapehu volcano. Results are interpreted in context with a model proposed by Christenson et al. [2010] which suggests that the hydrothermal system may be progressively sealed by precipitation of an elemental sulphur–anhydrite–natroalunite mineral assemblage. This is in turn associated with a pressurisation beneath the seal. The results are consistent with a pressurisation and failure of a sealed system, but cannot discriminate whether this seal failure is in the hydrothermal system or due to a deeper process. The results show the value of advanced spectral processing of data to elucidate details of the volcanic system that might otherwise not be detected. Future research should focus on analysis of additional

examples and expand the analysis to other data streams to examine the robustness of the present observations and to document alternative evolutionary patterns, if they exist. It is worthwhile to emphasise that the patterns learned by SOM only by looking at 2006 data were recognised again in 2007. Although as we have underlined the interpretation of each colour (pattern) is not unique, with a greater set of example events and a correspondingly more sophisticated training, the approach towards real-time recognition of such patterns may prove valuable from a monitoring perspective.
Moreover the alternative SOM analysis, based on the observation of the data distribution onto the map, carried out for the tremor acquired at Ruapehu during the days 23, 24 and 25 September 2007, highlighted the existence of dynamic regimes that are consistent and coherent over time and identifies a possible precursors of the paroxysmal phase.

SOM results are represented as two-dimensional maps, with a non-parametric mapping that projects the high dimensional original dataset in a fashion that provides both an unsupervised clustering and a highly visual representation of the data relationships.
The proposed analysis process has also been applied to improve the HVSR technique. The evaluation of the fundamental frequency (i.e. the peak of the H/V function) is not always easy. The acquired signal is splitted in time windows (upon which the corresponding frequency spectrum is calculated), but some of these windows can be affected by a considerable amount of noise, for example due to anthropogenic sources localised close to the seismic station. The identification of these time windows allows an easier and more reliable location of the H/V function's peak.
The seismic characterisation of the site is converted into a data cluster analysis where the proposed SOM process is applied to horizontal to vertical spectral ratios. The basic idea is to handle separately the spectral ratios in east–west (EW) and north–south (NS) directions. Being data projection on the map temporally ordered, considerations about data mapping into a cluster can help to understand if cluster properties are stable or not. Since the fundamental frequency is a feature of the site, it should always be present in all spectral ratios. On the other hand, peaks of H/V function due to environmental noise should not necessarily be stable in time.
The National University of Salta (Argentina) is carrying out a project of microzonation of the city of Salta, since the National Institute for Seismic Prevention considers the seismic hazard of particular interest among the natural hazards for the city and its district, given the high population density of the area and the decisive influence of site effects due to the geo-morphology of the valley. As a case study, SOM has therefore been applied to the short and highly noisy seismic tremor acquisitions of Salta city, with the objective of an easier and more reliable determination of the H/V function's peak.
The SOM analysis does not completely confirm the microzonation proposed by the researchers of the National University of Salta for the area of interest. In particular the low frequency range seems hard to inspect, since the tremor acquisitions are short, highly affected by anthropogenic noise and the tremor is acquired by a short-period seismometer. On the other hand the SOM analysis results highlight that the H/V function peak moves towards the low frequency taking in account the stations furthest from the edge of the valley. This seems to be consistent with the increasing thickness of the deposits towards the center of the valley.

Acknowledgements

I would like to express my special appreciation and thanks to my advisor Professor Roberto Carniel, you have been a mentor for me. I would like to thank you for encouraging me, giving me the opportunity to develop independently this project research, yet being present steadily during these years. Thanks for allowing me to express myself freely.

Besides my advisor, I would like to thank my referees, Professor Marco Mucciarelli and Professor Luciano Telesca, for their insightful comments and questions.

A special thanks to my family and my parents especially, thanks for believing in me and support me.

Thank you Paola, for without you, this thesis draft would be rather long and boring for sure. I want to thank you to make me open my eyes on forgotten aspects of my life, you helped me not to give up.

Thanks!

"One day I will find the right words, and they will be simple."

Jack Kerouac, *The Dharma Bums*

Appendix

Appendix A

R software script

The SOM process proposed in this work has been entirely implemented using the free software environment R [Team, 2010]. Some packages were already available in the repositories of R software (*http://cran.r-project.org/web/packages/*) for the implementation of some parts of the process, some other parts have been written from scratch. Here below is shown the original code written to implement the parts of the SOM process not available in the R repository and the scripts that call the appropriate R packages and the original code and that manage the graphics.

R is a free implementation of a dialect of the S language, the statistics and graphics environment for which John Chambers won the ACM Software Systems award. R was designed for interactive data analysis, blurring the distinction between users and programmers. The core is an interpreted computer language which allows branching and looping as well as modular programming using functions.

Since R is free and open, everybody can use R free of charge and can inspect the code and tinker with it (provided that the terms of the GNU General Public License ver.2, under which it is distributed, is fulfilled), so thousands of experts around the world have done just that and their contributions benefit the millions of people who use R. With thousands of contributors and more than two million users around the world, there's a wealth of community resources for R available on the Web, for help in just about every domain.

R is similar to other programming languages, like C, Java and Perl, in that it helps people perform a wide variety of computing tasks by giving them access to various commands. As an interactive language, as opposed to a data-in / data-out black-box procedures, R promotes experimentation and exploration, which improves data analysis. R doesn't restrict the user to choosing a pre-defined set of routines, the code contributed by others can be used in an open-source manner or can be the same user to extend R with its own functions.

Data visualization through charts and graphs is an essential part of the data analysis process, so R has excellent tools for creating graphics, from staples like bar charts and scatterplots to multi-panel lattice charts to brand new graphics of the user own devising. R has great graphical power, but it's not a point and click interface. This means that the user must uses typed commands to get it to produce the graphs. This can be a bit tedious at first and not very user-friendly, but once a list of useful commands (graphical functions and their parameters settings) has been learned and saved, this is a good starting point to begin to visualize data analysis results, quickly pasting that commands into the R command line and slightly modifying them to obtain original and meaningful graphics.

The code below is intended as supplementary to the basic R functions and packages. For each function originally written and shown below, an example of a call of it is shown, so as to remove any ambiguity about the form of arguments to be passed. Moreover any dependency to packages and functions (including the functions original written and shown below) which the written code references is explicated. Before calling an R function, it’s necessary to load the corresponding package (the one that includes that function), otherwise if the function has been originally written then it has to be declared in advance and loaded in the workspace.

A.1 Time Series and Spectrograms Plotting

TSSP function is written to plot the temporal series and to calculate and plot the corresponding spectrograms and the H/V spectral ratios, if the analysis that is carrying on is a HVSR analysis. *TSSP* takes from the folder “*/folder/dy*” the horizontal and vertical components that are R objects of the class “vector”, each one named as “fileTOT” and saved in the folder as “fileTOTew” or “fileTOTns” or “fileTOTud”.

```
TSSP <- function(folder, dy, component, Fc, doSP, Nfft, Ns, Nov, fl, cutTS,
    lenFIN, csU, csD, normMax, bKonno, splitTS, filter, flfilt, fhfilt,
    rotate, trwd)
{
  dyDIR <- sprintf('/%s/%08d', folder, dy)
  dir.create(path=dyDIR, showWarnings=F)
  spDIR <- sprintf('%s/%08d/Spectrograms', folder, dy)
  dir.create(path=spDIR, showWarnings=F)

  for(i in 1:length(component)) # time series components loading
  {
    strFT <- sprintf('fileTOT%s', component[i])
    tsIN <- sprintf('%s/%08d/%s', folder, dy, strFT)
    fT <- c()
    fT <- get(load(tsIN))

    if(is.finite(cutTS[1]) == T && is.finite(cutTS[2]) == T)
    {
      fT <- fT[cutTS[1]:cutTS[2]]
      fT <- fT - mean(fT, na.rm=T)      #zero mean
      fT <- fT / sd(fT, na.rm=T)        #unit variance
    }
    if(is.finite(cutTS[1]) == T && is.finite(cutTS[2]) == F)
    {
      fT <- fT[cutTS[1]:length(fT)]
      fT <- fT - mean(fT, na.rm=T)
      fT <- fT / sd(fT, na.rm=T)
    }
    if(is.finite(cutTS[1]) == F && is.finite(cutTS[2]) == T)
    {
      fT <- fT[1:cutTS[2]]
      fT <- fT - mean(fT, na.rm=T)
      fT <- fT / sd(fT, na.rm=T)
    }

    if(filter == T)     # Butterworth filter
    {
      wl <- flfilt/(Fc*0.5)
      wh <- fhfilt/(Fc*0.5)
      bttw <- butter(n=7, W=c(wl,wh), type='pass', plane='z')
      for(j in 1:length(component))
      {
        fT <- filter(filt=bttw$b, a=bttw$a, x=fT)
      }
    }
    assign(strFT, fT)
  }

  if(rotate == T)          # horizontal components rotation
  {
    SPwcc <- rep(NA, (90/5)+1)
    fiROT <- NA
    for(fi in seq(0,90, by=5))
    {
      fTOTrot1 <- fileTOTew*cos(fi*pi/180) - fileTOTns*sin(fi*pi/180)
      fTOTrot2 <- fileTOTew*sin(fi*pi/180) + fileTOTns*cos(fi*pi/180)
      SProt1 <- evolfft(fTOTrot1, dt=1/Fc, Nfft=Nfft, Ns=Ns, Nov=Nov, fl=fl, fh=Fc/2)
      SProt2 <- evolfft(fTOTrot2, dt=1/Fc, Nfft=Nfft, Ns=Ns, Nov=Nov, fl=fl, fh=Fc/2)
      SPmnrot1 <- rowMeans(SProt1$DSPEC)
      SPmnrot2 <- rowMeans(SProt2$DSPEC)
      SPwcc[fi/5+1] <- wcc(SPmnrot1, SPmnrot2, trwdth=trwd)
    }
    fiROT <- ((which.min(SPwcc)-1)*5)
    fTOTrot1 <- fileTOTew*cos(fiROT*pi/180) - fileTOTns*sin(fiROT*pi/180)
    fTOTrot2 <- fileTOTew*sin(fiROT*pi/180) + fileTOTns*cos(fiROT*pi/180)
    fileTOTew <- fTOTrot1
    fileTOTns <- fTOTrot2
    strROT <- sprintf('Rotazione␣delle␣componenti␣orizzontali␣pari␣a␣%d␣gradi', fiROT)
    print(strROT)
  }
  else
  {
    fiROT <- NA
  }

  for(i in 1:length(component))
  {
    strFT <- sprintf('fileTOT%s', component[i])
    fileTOT <- c()
    fileTOT <- get(strFT)
    nFIN <- ceiling(length(fileTOT)/lenFIN)
    for(x in 1:nFIN)     # time series plotting
    {
      sspDIR <- sprintf('%s/%08d/Spectrograms/sSP%02d', folder, dy, x)
      dir.create(path=sspDIR, showWarnings=F)
      strplot <- sprintf('%s/%08d%s.png', sspDIR, dy, component[i])
      png(filename = strplot, width = 1440, height = 720)
      fTOT <- fileTOT[((x-1)*lenFIN+1) : (x*lenFIN)]     # splitting the time series
      fTOT <- fTOT - mean(fTOT, na.rm=T)
      fTOT <- fTOT / sd(fTOT, na.rm=T)
      PlotTSsplit(data=fTOT, dy=dy, component=component[i], Fc=Fc, splitTS=splitTS)
      dev.off()

      if(doSP == T)       # spectrograms calculation and plotting
      {
        SP <- evolfft(fTOT, dt=1/Fc, Nfft=Nfft, Ns=Ns, Nov=Nov, fl=fl, fh=Fc/2)
        nfreq <- length(SP$freqs)
        freqs <- SP$freqs
        Wb <- c()
        vSP <- c()
        SPk <- matrix(nrow=nrow(SP$DSPEC), ncol=ncol(SP$DSPEC))

        if(is.finite(bKonno) == T)        # Konno-Ohmachi smoothing
        {
          print('Smoothing˽spettrogrammi˽con˽la˽funzione˽di˽Konno-Ohmachi')
          for(centf in 1:nfreq)
          {
            cf <- freqs[centf]
            h <- 1
            for(f in 1:nfreq)
            {
              varf <- freqs[f]
              Wb[h] <- ((sin(log10((varf/cf)^bKonno)))/(log10((varf/cf)^
                  bKonno)))^4
              if(is.nan(Wb[h]) == T)
              {
                Wb[h] <- 1
              }
              h <- h + 1
            }
            for(spcol in 1:ncol(SP$DSPEC))
            {
              vSP[spcol] <- mean(SP$DSPEC[,spcol] * Wb)
            }
            SPk[centf,] <- vSP
          }
          SP$DSPEC <- SPk
        }

        if(normMax == T)
        {
          print('Normalizzazione˽spettrogrammi:˽ogni˽spettro˽viene˽diviso˽
              per˽il˽suo˽valore˽max')
          for(c in 1:ncol(SP$DSPEC))
          {
            SP$DSPEC[,c] <- (SP$DSPEC[,c]/max(SP$DSPEC[,c], na.rm=T))
          }
        }

        spOUT <- sprintf('%s/SP.%s', sspDIR, component[i])
        save(SP, file=spOUT)
        cs <- rainbow(128)[110:0]
        cs <- c(cs, rep(cs[110],csU))
        cs <- c(rep(cs[1],csD), cs)
        imgSP <- sprintf('%s/SP.%s.png', sspDIR, component[i])
        png(filename=imgSP, width=960, height=720)
        par(cex=1, xaxt='s', yaxt='s', mai=c(0.9,0.7,0.9,2.7), las=2)
        # modify 'may' if white lines appear in the spectrogram plot
        plotevol(DEVOL=SP, log=1, fl=fl, fh=Fc/6.4, col=cs, ylog=F, ygrid=F,
             AXE=c(1, 2, 3, 4), CSCALE=F, WUNITS="Amplitude", STAMP=NULL,
            STYLE="fft")
        par(mai=c(0.9,12.3,2.15,0.5), new=T)      # color legend edges
        rgSP <- range(SP$DSPEC, na.rm=T)
        plot(rep(0,length(cs)), seq(rgSP[1],rgSP[2],length=length(cs)), pch
            = 15, col=cs, xaxt='n', xlab='', ylab='', las=3)
        box()
        dev.off()

        sSP <- sprintf('/sSP%02d', x)
        SPmd(folder=folder, dy=dy, sSP=sSP, SP=SP, component=component[i],
            Fc=Fc, ampLOG=T, pikFmin=1.0, pikFmax=10, pikN=10)
      }
    }
  }
  STRtxt <- sprintf('%s/%08d/Spectrograms/parametriSP.txt', folder, dy)
```

```r
  # parameters values storing in a .txt
  SPpar <- toString(SP$wpars)
  STRtext <- sprintf('Parametri␣per␣lo␣spettrogramma␣-␣pacchetto␣RSEIS,␣funzione␣evolfft␣-\nsample␣frequency:␣%f\nparameters␣Nfft,␣Ns,␣Nov,␣fl,␣fh:␣%s\ncolor␣scale␣extension[red-purple]:␣%f␣%f\nsmoothing␣Konno-Ohmachi:␣%f\nnormalizzazione␣spettro_i/max[spettro_i]:␣%s\nfilter␣%f␣to␣%f\nrotate␣%f␣degrees\n\n', Fc, SPpar, csU, csD, bKonno, normMax, flfilt, fhfilt, fiROT)
  cat(STRtext, file=STRtxt)
}
```

How to call *TSSP*, an example:

```r
source('/home/luca/Univ/PhD/BatchVolc/TSSP.R') # to load the TSSP function
folder <- '/home/luca/Univ/PhD/MudVolc'        # main folder for SOM analysis
dy <- 20060623
Fc <- 128       # sampling frequency
                # see http://cran.r-project.org/web/packages/RSEIS/RSEIS.pdf
Nfft <- 768     # fft lenght
Ns <- 384       # number of samples in a window
Nov <- 192      # number of samples of overlap per window
cutTS <- c((0*60*60*Fc+1),(20*60*Fc)) # to consider just a piece of the time series
lenFIN <- 20*60*Fc # length of single analysis window
splitTS <- 2*60*Fc # time series length for each graph row
                   # e.g. lenFIN=20x60xFc & splitTS=5x60xFc => time series splitted in 4 rows
trwd <- 16         # triangle width for the wcc, given in the number of data points

TSSP(folder=folder, dy=dy, component=c('ew','ns','ud'), Fc=Fc, doSP=T, Nfft=Nfft, Ns=Ns, Nov=Nov, fl=0.2, cutTS=cutTS, lenFIN=lenFIN, csU=15, csD=55, normMax=T, bKonno=60, splitTS=splitTS, filter=T, flfilt=1, fhfilt=15, rotate=T, trwd=trwd)
```

Required packages: *signal, RSEIS, wccsom*

Required functions: *butter* (*signal*), *filter* (*signal*), *evolfft* (*RSEIS*), *plotevol* (*RSEIS*), *wcc* (*wccsom*), *PlotTSsplit* (see par. A.2), *SPmd* (see par. A.3)

A.2 Plotting Splitted Time Series

PlotTSsplit is written to plot the time series subdivided into *n*-rows. *PlotTSsplit* is called by *TSSP* function.

```r
PlotTSsplit <- function(data, dy, component, Fc, splitTS)
{
  data <- data/max(abs(range(data, na.rm=T)))
  data1 <- (data[1:splitTS])*3
  range <- 0.5
  nTS <- floor(length(data)/splitTS)
  par(mai=c(1,1,1.9,1))
  plot(data1, type='l', ylim=c(-((nTS*2-1)*range), range), yaxt='n', xaxt='n', xlab='Time␣[min]', ylab='Amplitude')
  axis(side=1, at=seq(0, length(data1), by=Fc*60), labels=seq(0, floor(length(data1)/(Fc*60)), by=1), cex.axis=1)
  axis(side=3, at=seq(0, length(data1), by=Fc*60), labels=seq(0, length(data1), by=Fc*60), cex.axis=1, las=2)
  strmain <- sprintf('Time␣series␣day␣%08d,␣component␣%s', dy, component)
  title(main=strmain)
  segments(rep(0,5), c(-1,1,-0.5,0.5,0), rep(length(data1)*1.03,5), c(-1,1,-0.5,0.5,0), lty=2, col='gray')
  segments(seq(0, length(data1), by=Fc*60), rep(-range*(nTS*2-1),floor(length(data1)/(Fc*60))), seq(0, length(data1), by=Fc*60), rep(range, floor(length(data1)/(Fc*60))), lty=2, col='gray')
  for(i in 2:nTS)
  {
    din <- splitTS*(i-1) + 1
    den <- splitTS*i
    datai <- (data[din:den])*3
    datai <- datai - range*(i-1)*2
    lines(datai, type='l')
    segments(rep(0,3), c(-2*range*(i-1), -2*range*(i-1)+1, -2*range*(i-1)-1), rep(length(datai)*1.03,3), c(-2*range*(i-1), -2*range*(i-1)+1, -2*range*(i-1)-1), lty=2, col='gray')
  }
}
```

How to call *PlotTSsplit*, an example: *PlotTSsplit* can be called from *TSSP* function (see par. A.1 at line 89).

Required packages: none

Required functions: none (except the base functions for which a call is not required)

A.3 Averaged Spectrum

SPmd is written to calculate the averaged spectra of the corresponding spectrograms calculated by the TSSP function. *SPmd* is called by *TSSP* function.

```
SPmd <- function(folder, dy, sSP, SP, component, Fc, ampLOG, pikFmin, pikFmax, pikN)
{
    rangeSp <- c(-0.3,3.5)
    SPlog <- log(SP$DSPEC)
    SPmn <- rowMeans(SPlog)
    SPsd <- apply(t(SPlog), 2, sd)
    SPmnPsd <- SPmn + SPsd
    SPmnMsd <- SPmn - SPsd
    SPmd <- rep(NA, length=nrow(SPlog))
    for(rm in 1:nrow(SP$DSPEC))
    {
      SPmd[rm] <- median(SPlog[rm,], na.rm=T)
    }
    SPmn <- exp(SPmn)
    SPsd <- exp(SPsd)
    SPmnPsd <- exp(SPmnPsd)
    SPmnMsd <- exp(SPmnMsd)
    SPmd <- exp(SPmd)
    if(ampLOG == T)
    {
      SPmn <- log10(SPmn)
      addAMP <- abs(min(SPmn, na.rm=T))
      SPmn <- SPmn + addAMP
      SPmnPsd <- log10(SPmnPsd) + addAMP
      SPmnMsd <- log10(SPmnMsd) + addAMP
      SPmd <- log10(SPmd) + addAMP
    }
    plotSPmn <- sprintf('%s/%08d/Spectrograms%s/SPmnlog.%s.png', folder, dy,
        sSP, component)
    png(filename=plotSPmn, width=960, height=720)
    plot(SPmd[1:(nrow(SP$DSPEC)/2)], type='l', lwd=1.5, ylim=rangeSp, xlab='
        Freq␣[Hz]', ylab='Amp', xaxt='n', log='x', col='green')
    numf <- SP$numfreqs/2
    lines(rep(0, length=numf), col='gray', lwd=1.25, lty=3)
    lines(rep(0.25, length=numf), col='gray', lwd=1.25, lty=3)
    lines(rep(0.5, length=numf), col='gray', lwd=1.25, lty=3)
    lines(rep(0.75, length=numf), col='gray', lwd=1.25, lty=3)
    lines(rep(1, length=numf), col='gray', lwd=1.25, lty=2)
    text(length(SP$freqs), 1.1, 'sd␣=␣0', cex=0.9)
    lines(rep(2, length=numf), col='gray', lwd=1.25, lty=2)
    text(length(SP$freqs), 2.1, 'sd␣=␣1', cex=0.9)
    lines(rep(3, length=numf), col='gray', lwd=1.25, lty=2)
    text(length(SP$freqs), 2.9, 'sd␣=␣2', cex=0.9)
    ind <- seq(1, numf, by=2^3)
    lab <- c(length(ind))
    for(nIND in 1:length(ind))
    {
      lab[nIND] <- ((ind[nIND]-1) * (SP$freq[numf]-SP$freq[1]) / (numf-1)) +
          SP$freq[1]
    }
    axis(side=1, at=ind, labels=ceiling(lab*10)/10)
    lines(SPmnPsd[1:numf], type='l', lty=2, lwd=1.5, col='red')
    lines(SPmnMsd[1:numf], type='l', lty=2, lwd=1.5, col='blue')
    lines(SPsd[1:numf] + 1, type='l', lty=2, lwd=1, col='gray')
    lines(SPmn[1:numf], type='l', lwd=2.0, col='black')
    legend("topright", c("SP␣mean", "SP␣+␣sd", "SP␣-␣sd", "sd", "SP␣median")
        , lty=c(1), lwd=1.5, col=c('black', 'red', 'blue', 'gray', 'green'))

    # Picking the averaged spectrum (SPmn)

    limFd <- numf*pikFmin/(Fc/2)
    limFu <- numf*pikFmax/(Fc/2)
    limSPmn <- order(SPmn[limFd:limFu], decreasing=T)
    z1 <- 1
    z2 <- 1
    while(z1 < (pikN+1) && z2 <= length(limSPmn))
    {
      fm <- limSPmn[z2] + numf*pikFmin/(Fc/2) - 1
      z2 <- z2 + 1
      if(is.na(SPmn[fm-1]) == F && is.na(SPmn[fm+1]) == F)
      {
        if(SPmn[fm-1] <= SPmn[fm] && SPmn[fm+1] <= SPmn[fm])
        {
          segments(fm, -0.3, fm, 6.5/log(z1+7), col='black', lwd=1.3, lty=2)
          mtext(text=as.character(floor((fm*(Fc/2)/numf)*10)/10), at=fm,
              padj=6*log(z1+3), side=3)
          z1 <- z1 + 1
        }
      }
    }
    dev.off()
}
```

How to call *SPmd*, an example: *SPmd* can be called from *TSSP* function (see par. A.1 at line 153).

```
folder [previously declared]
```

```
dy [previously declared]
sSP [passed by TSSP] # the sub-window (see lenFIN in TSSP)
SP [passed by TSSP]  # the spectrogram calculated by TSSP
component [passed by TSSP]
Fc [previously declared]
ampLOG = TRUE to choose the logarithmic scale for the frequency scale
pikFmin the minimum frequency value from where the picking starts
pikFmax the maximum frequency value from where the picking stops
pikN    the maximum number of points detected by the picking

SPmd(folder=folder, dy=dy, sSP=sSP, SP=SP, component=component[i], Fc=Fc, ampLOG=T, pikFmin=1.0, pikFmax=10, pikN=10)
```

Required packages: none

Required functions: none (except the base functions for which a call is not required)

A.4 HVSR (Horizontal to Vertical Spectral Ratio)

HVSR is written to implement the Nakamura spectral ratio technique [Nakamura, 1989]. *HVSR* receives in input the spectrograms of the vertical component and of just one or both the horizontal components of the acquired signal and gives in output the spectrograms obtained dividing each spectrogram of the horizontal component by the spectrogram of the vertical component.

```
HVSR <- function(folder, dy, component, nFIN, Fc, fl, csU, csD, pikFmin, pikFmax, pikN, splitHV)
{
  for(i in 1:length(component))
  {
    if(component[i] == 'ew')
    {
      for(x in 1:nFIN)  # merging the spectrograms splitted by TSSP
      {
        if(x == 1)
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ew', folder, dy, x)
          load(spIN)
          SPew <- SP$DSPEC
        }
        else if(x > 1)
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ew', folder, dy, x)
          load(spIN)
          SPew <- cbind(SPew, SP$DSPEC)
        }
      }
    }
    else if(component[i] == 'ns')
    {
      for(x in 1:nFIN)
      {
        if(x == 1)
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ns', folder, dy, x)
          load(spIN)
          SPns <- SP$DSPEC
        }
        else if(x > 1)
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ns', folder, dy, x)
          load(spIN)
          SPns <- cbind(SPns, SP$DSPEC)
        }
      }
    }
    else  if(component[i] == 'ud')
    {
      for(x in 1:nFIN)
      {
        if(x == 1)
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ud', folder, dy, x)
          load(spIN)
          HV <- SP
        }
        else if(x > 1)
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ud', folder, dy, x)
          load(spIN)
          HV$DSPEC <- cbind(HV$DSPEC, SP$DSPEC)
          HV$sig <- c(HV$sig, SP$sig)
          SPtims <- seq(from=HV$tims[length(HV$tims)]+(HV$tims[2]-HV$tims[1]), by=HV$tims[2]-HV$tims[1], length.out=ncol(SP$DSPEC))
          HV$tims <- c(HV$tims, SPtims)
        }
      }
      SPud <- HV$DSPEC
    }
  }

  for(i in 1:length(component))
  {
    if(component[i] == 'ew')
    {
      HV$DSPEC <- SPew/SPud
      hvOUT <- sprintf('%s/%08d/Spectrograms/HV.ew', folder, dy)
      save(HV, file=hvOUT)
    }
    if(component[i] == 'ns')
    {
      HV$DSPEC <- SPns/SPud
      hvOUT <- sprintf('%s/%08d/Spectrograms/HV.ns', folder, dy)
      save(HV, file=hvOUT)
    }
    if(component[i] != 'ud')
    {
      SPmd(folder=folder, dy=dy, sSP='', SP=HV, component=component[i], Fc=Fc, ampLOG=T, pikFmin=1.0, pikFmax=14, pikN=10)
      cs <- rainbow(128)[110:0]
      cs <- c(cs, rep(cs[110],csU))
      cs <- c(rep(cs[1],csD), cs)
      hvDIR <- sprintf('%s/%08d/Spectrograms', folder, dy)
      imgHV <- sprintf('%s/HV.%s.png', hvDIR, component[i])
      png(filename=imgHV, width=960, height=720)
      par(cex=1, xaxt='s', yaxt='s', mai=c(0.9,0.9,0.9,2.5), las=2)
      # modify 'may' if white lines appear in the spectrogram plot
      plotevol(DEVOL=HV, log=1, fl=fl , fh=Fc/2, col=cs , ylog=F, ygrid=F, AXE=c(1, 2, 3, 4), CSCALE=F, WUNITS="Amplitude H/V", STAMP=NULL, STYLE="fft")
      par(mai=c(0.9,12.3,2.15,0.5), new=T)      # color legend edges
      rgHV <- range(HV$DSPEC, na.rm=T)
      plot(rep(0,length(cs)), seq(rgHV[1],rgHV[2],length=length(cs)), pch = 15, col=cs, xaxt='n', xlab='', ylab='', las=3)
      box()
      dev.off()
    }
  }

  if(splitHV == T)      # splitting the spectrograms
  {
    for(x in 1:nFIN)
    {
      for(i in 1:length(component))
      {
        if(component[i] == 'ew')
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ew', folder, dy, x)
          load(spIN)
          SPew <- SP$DSPEC
        }
        else if(component[i] == 'ns')
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ns', folder, dy, x)
          load(spIN)
          SPns <- SP$DSPEC
        }
        else  if(component[i] == 'ud')
        {
          spIN <- sprintf('%s/%08d/Spectrograms/sSP%02d/SP.ud', folder, dy, x)
          load(spIN)
          HV <- SP
          SPud <- HV$DSPEC
        }
      }
      for(i in 1:length(component))
      {
        if(component[i] == 'ew')
        {
          HV$DSPEC <- SPew/SPud
        }
        if(component[i] == 'ns')
        {
          HV$DSPEC <- SPns/SPud
        }
        if(component[i] != 'ud')
        {
          cs <- rainbow(128)[110:0]
          cs <- c(cs, rep(cs[110],csU))
          cs <- c(rep(cs[1],csD), cs)
          hvDIR <- sprintf('%s/%08d/Spectrograms/sSP%02d', folder, dy, x)
          imgHV <- sprintf('%s/HV.%s.png', hvDIR, component[i])
          png(filename=imgHV, width=960, height=720)
          par(cex=1, xaxt='s', yaxt='s', mai=c(0.9,0.9,0.9,2.3), las=2)
          # modify 'may' if white lines appear in the spectrogram plot
          plotevol(DEVOL=HV, log=1, fl=fl, fh=Fc/2, col=cs, ylog=F, ygrid=F, AXE=c(1, 2, 3, 4), CSCALE=F, WUNITS="Amplitude␣H/V", STAMP=NULL, STYLE="fft")
          par(mai=c(0.9,12.3,2.15,0.5), new=T)  # color legend edges
          rgHV <- range(HV$DSPEC, na.rm=T)
          plot(rep(0,length(cs)), seq(rgHV[1],rgHV[2],length=length(cs)), pch = 15, col=cs, xaxt='n', xlab='', ylab='', las=3)
          box()
          dev.off()
        }
      }
    }
  }
}
```

How to call *HVSR*, an example: *HVSR* can be called from *TSSP* function after the line 156 (see par. A.1)

```
folder  [previously declared]
dy      [previously declared]
nFIN    [returned by TSSP]
Fc      [previously declared]
fl the low frequency cut-off (plotevol function)
csU & csD to modify the spectrogram color scale
pikFmin the minimum frequency value from where the picking starts
pikFmax the maximum frequency value from where the picking stops
pikN is the maximum number of points detected by the picking
splitHV = TRUE to split the spectrograms in the corresponding nFIN windows

HVSR(folder=folder, dy=dy, component=c('ew','ns','ud'), nFIN=nFIN, Fc=Fc, fl=0.2, csU=45, csD=55, pikFmin=1.0, pikFmax=14, pikN=10, splitHV=T)
```

Required packages: *RSEIS*

Required functions: *plotevol* (*RSEIS*)

A.5 SOM map

SOMmap is written to carry on the training phase of a SOM map or to project the data on a trained SOM map. In both cases *SOMmap* provide the visualization of the map calling the *Umatrix* function.

```
SOMmap <- function(folder, dy, day, sSP, component, SPload, SPcutU, SPcutD, xdim, ydim, rlen, alpha_i, alpha_f, rad_i, rad_f, trwidth, toro, initDIAG, initCODE)
{
 # 'dy' is a numeric element of type 20010101 that indicates the day folder where to save the analysis results
 # 'day' is a numeric vector of elements of type 20010101 that indicates the day folders where to get the data (spectra) for the training and/or projection phase
 # 'sSP' is a character vector of elements of type '/sSP01' that indicates the sub-spectrograms of the spectrogram splitted by TSSP
  mapDIR <- sprintf('/%s/Maps', folder)
  dir.create(path=mapDIR, showWarnings=F)
  fldMAP <- sprintf("SOM%08d%s", dy, component)
  mapDIR2 <- sprintf('/%s/Maps/%s', folder, fldMAP)
```

```
dir.create(path=mapDIR2, showWarnings=F)

if(SPload == '')
{
  for(r in 1:length(day))      # merging the spectrograms of different days
  {
    for(s in 1:length(sSP))   # merging the sub-spectrograms of the same day
    {
      STRload <- sprintf('%s/%08d/Spectrograms%s/SP.%s', folder, day[r], sSP[s], component)
      if(file.exists(STRload) == T && r == 1 && s == 1) # first spectrogram to be merged
      {
        load(STRload)
        SPna <- SP$DSPEC[c(SPcutD:SPcutU),]    # cutting frequency in range of interest
        v_sp <- rep(NA, length=ncol(SPna))     # removing the NA spectra
        j <- 1
        v <- 0
        for(n in 1:ncol(SPna))
        {
          if(is.na(SPna[1,n]) == F)
          {
            v_sp[j] <- n
            j <- j + 1
            v <- v + 1
          }
        }
        SPtot <- matrix(nrow=nrow(SPna), ncol=v) # new matrix without NA spectra
        w <- 1
        for(i in 1:length(v_sp))
        {
          if(is.na(v_sp[i]) == F)
          {
            SPtot[,w] <- SPna[,v_sp[i]]
            w <- w + 1
          }
        }
      }
      else if(file.exists(STRload) == T && (r > 1 || s > 1))
      # spectrograms to be merged (except the first one)
      {
        load(STRload)
        SPna <- SP$DSPEC[c(SPcutD:SPcutU),]
        v_sp <- rep(NA, length=ncol(SPna))
        v <- 0
        j <- 1
        for(n in 1:ncol(SPna))
        {
          if(is.na(SPna[1,n]) == F)
          {
            v_sp[j] <- n
            j <- j + 1
            v <- v + 1
          }
        }
        SPnotNA <- matrix(nrow=nrow(SPna), ncol=v)
        w <- 1
        for(i in 1:length(v_sp))
        {
          if(is.na(v_sp[i]) == F)
```

```r
               {
                  SPnotNA[,w] <- SPna[,v_sp[i]]
                  w <- w + 1
               }
            }
            SPtot <- cbind(SPtot, SPnotNA)
         }
         else if(file.exists(STRload) == F)
         {
            strNAfile <- sprintf("File %s non trovato", STRload)
            print(strNAfile)
         }
      }
   }
}
else if (SPload != '')
{
   if(file.exists(SPload) == F)
   {
      print('Inserisci il percorso completo dove trovare lo spettrogramma (matrice righe=freq bins, colonne=spettri)')
   }
   load(SPload)
}
strSPtot <- sprintf("%s/Maps/%s/SP%08d.%s", folder, fldMAP, dy, component)
save(SPtot, file = strSPtot, precheck=T)

# SOM map initialization

if(initDIAG == T)      # diagonal initialization
{
   dimMap <- floor(nrow(SPtot)^0.5)
   if(xdim*ydim > nrow(SPtot))
   {
      strdim <- sprintf('Scegliere una mappa quadrata di lato massimo %d', dimMap)
      stop(strdim)
   }
   vdiag <- c(1)
   for(d in 2:(ydim-1))
   {
      for(dd in 0:(d-1))
      {
         vdiag <- c(vdiag,(d+dd*(xdim-1)))
      }
   }
   for(d in ydim:xdim)
   {
      for(dd in 0:(ydim-1))
      {
         vdiag <- c(vdiag,(d+dd*(xdim-1)))
      }
   }
   dd <- ydim - 1
   for(d in 2:ydim)
   {
      for(dd in 0:(ydim-d))
      {
         vdiag <- c(vdiag,(d*xdim + (dd)*(xdim-1)))
      }
   }
   m_diag <- diag(x=10, nrow=xdim*ydim, ncol=nrow(SPtot))
   m_init <- matrix(NA, nrow=xdim*ydim, ncol=nrow(SPtot))
```

```
    for(v in 1:length(vdiag))
    {
      m_init[vdiag[v],] <- m_diag[v,]
    }
    mSOM <- wccsom(t(SPtot), grid=somgrid(xdim,ydim,'rectangular'), rlen=
        rlen, alpha=c(alpha_i, alpha_f),
                      radius=c(rad_i,rad_f), trwidth=trwidth, toroidal=toro,
                          keep.data=T, init=m_init, FineTune=F)
  }
  else if(initCODE != '')      # initializing the SOM map using an existing
      map
  {
    STRinitCODE <- sprintf('%s/code', initCODE)
    if(file.exists(STRinitCODE) == T)
    {
      load(STRinitCODE)
      mSOM <- wccsom(t(SPtot), grid=somgrid(xdim,ydim,'rectangular'), rlen=
          rlen, alpha=c(alpha_i, alpha_f),
                        radius=c(rad_i,rad_f), trwidth=trwidth, toroidal=toro,
                            keep.data=T, init=code, FineTune=F)
    }
    else
    {
      print('In␣initCODE␣scrivi␣il␣percorso␣del␣file␣\'code\'␣da␣caricare␣[
          senza␣\'/code\']')
    }
  }
  else  # random initialization (using the dataset)
  {
    mSOM <- wccsom(t(SPtot), grid=somgrid(xdim,ydim,'rectangular'), rlen=
        rlen, alpha=c(alpha_i, alpha_f),
                      radius=c(rad_i,rad_f), trwidth=trwidth, toroidal=toro,
                          keep.data=T, FineTune=F)
  }
  code <- mSOM$code
  strcode <- sprintf('%s/Maps/%s/code', folder, fldMAP)
  save(code, file=strcode)
  changes <- mSOM$changes
  strchanges <- sprintf('%s/Maps/%s/changes', folder, fldMAP)
  save(changes, file=strchanges)
  classif <- mSOM$unit.classif
  strclassif <- sprintf('%s/Maps/%s/classif', folder, fldMAP)
  save(classif, file=strclassif)
  error <- mSOM$wccs
  strerror <- sprintf('%s/Maps/%s/error', folder, fldMAP)
  save(error, file=strerror)

  strtxt <- sprintf('%s/Maps/%s/parametriSOM.txt', folder, fldMAP)
  strday <- toString(day)
  strsSP <- toString(sSP)
  strtext <- sprintf('Parametri␣mappa␣SOM:\ndata␣[day␣&␣component]:␣%s␣%s\
      nsubSP:␣%s\ncode␣[path]:␣%s\ncode␣init␣diag:␣%s\ncode␣initialization:␣%
      s\nfrequency␣cut:␣%s␣%s\ninit␣diag:␣%s\nxdim:␣%s\nydim:␣%s\nn␣of␣
      iterations:␣%s\nalpha_i:␣%.4f\nalpha_f:␣%.4f\nneighborhood␣radius_i:␣%s
      \nneighborhood␣radius_f:␣%s\ntrwidth:␣%s\ntoroidal:␣%s\n\n', strday,
      component, strsSP, strcode, initDIAG, initCODE, SPcutD, SPcutU,
      initDIAG, xdim, ydim, rlen, alpha_i, alpha_f, rad_i, rad_f,
trwidth, toro)
  cat(strtext, file=strtxt)
  Umatrix(folder=folder, fldMAP=fldMAP, component=component, dy=dy, code=
      code, classif=classif, xdim=xdim, ydim=ydim, trwdth=trwidth, UmcPlot=F)
}
```

How to call *SOMmap*, an example:

```
folder [previously declared]
dy     [previously declared]
day <- c(dy)
subSP <- c(1:1) # if there is just one sub-window with the whole spectrogram
sSP <- sprintf('/sSP%02d', subSP)
cmp <- c('ns')
SPload              not null string to load an existing spectrogram
SPcutU & SPcutD  to cut the spectrogram in a range of frequencies
xdim & ydim         map dimensions
rlen                number of iterations for the SOM training process
alpha               learning rate for the SOM training process
rad                 neighborhood radius for the SOM training process
trwd                [previously declared]
toro = TRUE         for a toroidal SOM map
initDIAG & initCODE to force the SOM map initialization

SOMmap(folder=folder, dy=dy, day=day, sSP=sSP, component=cmp, SPload='',
    SPcutU=SPcutU, SPcutD=SPcutD, xdim=xdim, ydim=ydim, rlen=25, alpha_i=0.1,
     alpha_f=0.001, rad_i=rad, rad_f=-rad, trwidth=trwd, toro=T, initDIAG=F,
    initCODE='')
```

Required packages: *wccsom*

Required functions: *wccsom* (*wccsom*), *Umatrix* (see par. A.6)

A.6 Umatrix

Umatrix is written to visualize the SOM map implementing the U-matrix (Unified distance matrix) method proposed by Ultsch and Siemon (1990). *Umatrix* is called by *SOMmap* function.

```
Umatrix <- function(folder, fldMAP, component, dy, code, classif, xdim, ydim
    , trwdth, thrhold, UmcPlot)
{
  if(UmcPlot == F)
  {
    WCCm <- matrix( , nrow=nrow(code), ncol=nrow(code))
    for(j in 1:nrow(WCCm))
    {
      for(i in 1:ncol(WCCm))
      {
        WCCm[i,j] <- wcc(code[i,], code[j,], trwdth=trwdth)
        # weighted cross-correlation matrix
        WCCm[j,i] <- WCCm[i,j]
      }
    }
    if(min(WCCm, na.rm=T) < 0)
    {
      print('ATTENZIONE:␣indici␣di␣correlazione␣negativi␣in␣WCC!')
      stop('Error')
    }
    strWCCm <- sprintf('%s/Maps/%s/WCCm', folder, fldMAP)
    save(WCCm, file=strWCCm)

    WCCm1su <- 1/WCCm
    agn_WCCm <- agnes(WCCm1su, diss=T, method='average')
    # dendrogram calculation
  agnWCCmOUT <- sprintf('%s/Maps/SOM%08d%s/agn_WCCm_A', folder, dy, component)
  save(agn_WCCm, file=agnWCCmOUT)

  WCCm <- WCCm - min(WCCm, na.rm=T)    # WCCm[i;j] >= 0
  WCCm <- WCCm / max(WCCm, na.rm=T)    # max(WCCm[i;j]) = 1
  WCCm <- WCCm^25

  # Umatrix calculation:

  Uydim <- ydim*2-1
  Uxdim <- xdim*2-1
  Um <- matrix(, nrow=Uxdim, ncol=Uydim)
  for(j in 1:Uydim)   # considering each cell of the Umatrix
  {
    for(i in 1:Uxdim)
    {
      if((i+j) %% 2 != 0) # cells between a pair of code vectors
      {
        if(i %% 2 != 0)        # a code vector above another one
        {
          n2 <- ((i+j+1)/2 + (Uxdim %/% 2)*(j %/% 2)) # number of the above cell
          Dmj <- (n2-(Uxdim %/% 2)-1)
          Um[i,j] <- WCCm[n2, Dmj]      # dissimilarity value of the pair of codes
        }
        else  # a pair of code vectors side by side
        {
          n3 <- ((i+j-1)/2 + (Uxdim %/% 2)*(j %/% 2)) # number of the left cell
          Um[i,j] <- WCCm[n3,n3+1]      # dissimilarity value of the pair of codes
        }
      }
      else    # cells between 4 code vectors and cells of the code vectors
      {
        if(i %% 2 == 0)        # cells between 4 code vectors
        {
          n1 <- ((i+j)/2 + (Uxdim %/% 2)*(j %/% 2))   # number of the top-left cell
          Um[i,j] <- (WCCm[n1,(n1-(Uxdim %/% 2))] + WCCm[n1+1,(n1-(Uxdim %/% 2)-1)])/2
          # mean dissimilarity value of the 4 code vectors
        }
        else  # cells of the code vectors
        {
          Um[i,j] <- max(Um)
        }
      }
    }
  }
  strUm <- sprintf('%s/Maps/%s/Um', folder, fldMAP)
  save(Um, file=strUm)

  strplot <- sprintf('%s/Maps/%s/Um%08d.%s.png', folder, fldMAP, dy, component)
  # plotting the U-matrix
  png(filename = strplot, width = 960, height = 720)
  ncell <- nrow(Um)*ncol(Um)
  s <- seq(0, 1, length = ncell)      # gray tone scale
  image(z=Um, col=gray(s), xaxt='n', yaxt='n')
  # correspondence between gray tone and dissimilarity value
  grid(Uxdim,Uydim)

  codeclass <- rep(0, (xdim*ydim))
  for(k in 1:length(classif))
  # conversion form "classif" (kohonen package) to "code.sum[,3]" (wccsom package)
  {
    codeclass[classif[k]] <- codeclass[classif[k]] + 1
  # number of inputs projected onto a map cell
  }

  for(j in 1:Uydim)
  {
    for(i in 1:Uxdim)
    # drawing circles and numbers into each map cell
    {
      if((i %% 2 != 0) && (j %% 2 != 0))
      {
        ncode <- ((i+j)/2 + (Uxdim %/% 2)*(j %/% 2))
        x <- ((i-1) / (Uxdim - 1))
        y <- ((j-1) / (Uydim - 1))
        points(x, y, pch=20, cex=(codeclass[ncode]/max(codeclass))*5.0, col='orange')
        # circles width according to the number of inputs projected
        text(x, y, labels=codeclass[ncode], font = 2, cex = 0.8)
        # drawing the number of the projected inputs
      }
    }
  }
  box(lwd=2)
  dev.off()
}

# drawing clusters onto the U-matrix

if(UmcPlot == T)
{
  strUm <- sprintf('%s/Maps/%s/Um', folder, fldMAP)
  load(strUm)
  Uydim <- ydim*2-1
  Uxdim <- xdim*2-1
  ncell <- nrow(Um)*ncol(Um)
  strplot <- sprintf('%s/Maps/%s/agn_THR%.03f/UmClus%08d.%s.png', folder, fldMAP, thrhold, dy, component)
  png(filename = strplot, width = 960, height = 720)
  s <- seq(0, 1, length = ncell)
  # gray tone scale
  image(z=Um, col=gray(s), xaxt='n', yaxt='n')
  # correspondence between gray tone and dissimilarity value
  grid(Uxdim,Uydim)
  strloadagn <- sprintf('%s/Maps/%s/agn_WCCm_A', folder, fldMAP)
  load(strloadagn)

  clusEL <- Cluster(agn=agn_WCCm, threshold=thrhold, xdim=xdim, ydim=ydim)
  STRsaveCL <- sprintf('%s/Maps/%s/agn_THR%.03f/clusEL_A', folder, fldMAP, thrhold)
  save(clusEL, file=STRsaveCL)
  cs <- rainbow(max(clusEL))[max(clusEL):0]
  # max(clusEL) = number of the detected clusters
  for(j in 1:Uydim)
  {
    for(i in 1:Uxdim)
    {
          if((i %% 2 != 0) && (j %% 2 != 0))
          # is it a cell with a code?
          {
            ncode <- ((i+j)/2 + (Uxdim %/% 2)*(j %/% 2))
            # number of the code in that cell
            x <- ((i-1) / (Uxdim - 1))
            y <- ((j-1) / (Uydim - 1))
            points(x, y, pch=23, cex=log(clusEL[ncode]+5)*1.00, col=cs[clusEL[ncode]])
            # coloured rhombus according to the cluster number
            text(x, y, labels=clusEL[ncode], font = 2, cex = 0.9)
            # drawing the cluster number
          }
        }
      }
      box(lwd=2)
      dev.off()
  }
}
```

How to call *Umatrix*, an example: *Umatrix* can be called from *SOMmap* function, after the line 155 (see par. A.5)

```
folder [previously declared]
fldMAP [passed by SOMmap]
component <- cmp [previously declared]
dy      [previously declared]
code    [passed by SOMmap]
classif [passed by SOMmap]
xdim    [previously declared]
ydim    [previously declared]
trwidth [previously declared]
UmcPlot = TRUE to plot the cluster U-matrix (only if the clustering recognition has been completed)

Umatrix(folder=folder, fldMAP=fldMAP, component=component, dy=dy, code=code, classif=classif, xdim=xdim, ydim=ydim, trwdth=trwidth, UmcPlot=F)
```

Required packages: *wccsom*, *cluster*

Required functions: *wcc* (*wccsom*), *agnes* (*cluster*), *Cluster* (see par. A.9)

A.7 Plotting the dendrogram and choosing the dissimilarity threshold

Since the choosing of the dissimilarity threshold is not unique, this part of the code has to be run line by line.

```
agnWCCmOUT <- sprintf('%s/Maps/SOM%08d%s/agn_WCCm_A', folder, dy, cmp)
load(agnWCCmOUT)
par(cex=0.25, cex.axis=3, cex.lab=3, mgp=c(4.7,1,0), cex.main=3.5, cex.sub=3, mai=c(0.4,0.4,0.3,0), lwd=0.4) plot(agn_WCCm, which.plots=2, xlab='', ylab='Dissimilarity␣Index␣[1/correlation]')
# now choose a dissimilarity threshold
thr <- [the chosen dissimilarity value]
segments(1, thr , xdim*ydim, thr, lty=2, col='red') # drawing an horizontal line (the threshold) on the dendrogram
```

A.8 Plotting the clustering U-matrix

Since the dissimilarity threshold has been chosen, a certain number of clusters has been detected from the dendrogram. *Umatrix* function labels each area on the SOM map using the colour of the corresponding cluster.

```
codeIN <- sprintf('%s/Maps/SOM%08d%s/code', folder, dy, cmp)
load(codeIN)
DIRmap <- sprintf('SOM%08d%s', dy, cmp)

Umatrix(folder=folder, fldMAP=DIRmap, component=cmp, dy=dy, code=code,
    classif=classif, xdim=xdim, ydim=ydim, trwdth=trwd, thrhold=thr, UmcPlot=
    T) # plotting the clustering U-matrix
```

A.9 Cluster

Cluster is written as an internal function of *HVSR* function. *Cluster* returns the vector *clusEL* creating the link between each neuron of the map and its corresponding cluster.

```
Cluster <- function(agn, threshold, xdim, ydim)
{
  v_height <- sort(agn$height)
  # $height stores the dissimilarity values at which two clusters merge
  h <- 1
  v_rowmerge <- c()
  # storing the $merge row numbers above the chosen threshold
  for(m in 1:nrow(agn$merge))
  {
    if(v_height[m] >= threshold)
    {
      v_rowmerge[h] <- m
      h <- h + 1
    }
  }
  k <- 1
  w <- 1
  v_rowmergeout <- c()
  # single neurons above the chosen threshold
  v_rowmergeel <- c()
  # clusters that merge just above the chosen threshold (I order fusions)
  v_merge <- as.vector(t(agn$merge[c(min(v_rowmerge):nrow(agn$merge)),]))
  for(r in 1:length(v_merge))
  {
    if(v_merge[r] > 0 && v_merge[r] < min(v_rowmerge))
    # removing from $merge the rows with II order fusions
    {
      v_rowmergeel[k] <- v_merge[r]
      k <- k + 1
    }
    else if(v_merge[r] < 0)
    # single neurons above the chosen threshold
    {
      v_rowmergeout[w] <- v_merge[r]
      w <- w + 1
    }
  }
  ncluster <- length(v_rowmergeel)      # number of clusters of the I order
  strNcluster <- sprintf('Ci␣sono␣%d␣cluster', ncluster)
  print(strNcluster)
  print(v_rowmergeout)
  print(v_rowmergeel)

  clusEL <- rep(NA, length=(xdim*ydim))
  for(c in 1:length(v_rowmergeel))        # taking in account each I order cluster
  {
    z <- 1
    k <- 1
    v_cluster <- c()
    v_clusTMP1 <- agn$merge[v_rowmergeel[c],]
    v_clusTMP2 <- c()
    while(k == 1)
    {
      for(q in 1:length(v_clusTMP1))
      {
        if(v_clusTMP1[q] > 0 && is.na(v_clusTMP1[q]) == F)
        {
          v_clusTMP2[c(z,z+1)] <- agn$merge[v_clusTMP1[q],]
          z <- z + 2
        }
        else if(v_clusTMP1[q] < 0 || is.na(v_clusTMP1[q]) == T)
        {
          #print('Uno o più elementi di $merge < 0 o NA')
          #v_clusTMP2[c(z,z+1)] <- c(NA, NA)
          #z <- z + 2
        }
      }
      v_cluster <- c(v_cluster, v_clusTMP1)
      v_clusTMP1 <- v_clusTMP2
      v_clusTMP2 <- c()
      if(length(v_clusTMP1) == 0)
      # if v_clusTMP1 is empty, there are no more fusions
      {
        k <- 0
      }
      z <- 1
    }
    v_cluster <- as.vector(na.omit(v_cluster))
    for(ind in 1:length(v_cluster))
    # now in v_cluster there are just the single neurons [negative values]
    {
      if(v_cluster[ind] < 0)
      {
        clusEL[abs(v_cluster[ind])] <- c
      }
    }
  }
  return(clusEL)
}
```

How to call *Cluster*, an example: *Cluster* can be called from *Umatrix* function, after the line 129 (see par. A.6)

```
agn_WCCm [passed by Umatrix]
thrhold  [previously declared]
xdim     [previously declared]
ydim     [previously declared]

Cluster(agn=agn_WCCm, threshold=thrhold, xdim=xdim, ydim=ydim) # Cluster returns clusEL
```

Required packages: none

Required functions: none (except the base functions for which a call is not required)

A.10 SPCL

SPCL is written to plot the averaged spectrum for each cluster and to return back in the time domain in order to plot the time series and analyse at which cluster each time window, on which the corresponding spectra has been calculated, is assigned.

```r
SPCL <- function(folder, dy, sSP, component, Fc, threshold, ampLOG, pikFmin, pikFmax, pikN, splitTS, Ns, Nov)
{
  range <- 0.25 # the value must be half of the range value used in plotTSsplit
  rangeSp <- c(-0.3,3.5)
  classifIN <- sprintf('%s/Maps/SOM%08d%s/classif', folder, dy, component)
  load(classifIN)
  clusterIN <- sprintf('%s/Maps/SOM%08d%s/agn_THR%.03f/clusEL_A', folder, dy, component, threshold)
  load(clusterIN)

#  for(d in 1:length(day))
#  {
    for(s in 1:length(sSP))
    {
      spIN <- sprintf('%s/%08d/Spectrograms%s/SP.%s', folder, dy, sSP[s], component)
      load(spIN)
      Ns <- SP$wpars$Ns
      Nov <- SP$wpars$Nov
      sn <- as.numeric(substr(sSP[s], 5, 6))

      classifsSP <- classif[((sn-1)*ncol(SP$DSPEC)+1) : (sn*ncol(SP$DSPEC))]
      # taking in account just the part of classif correspondent to sSP[s]
      clusIN <- rep(NA, length(classifsSP))
      # clusIN classifies spectra (clusEL classifies codes)
      # max(clusEL) corresponds to the number of clusters
      for(cl in 1:length(clusEL))
      {
        if(is.na(clusEL[cl]) == F)
        # clusEL[cl] is the cluster number, cl is the code vector number
        {
          for(clIN in 1:length(classifsSP))
          # clIN is the input vector number
          # classifsSP[clIN] is the code number
          {
            if(classifsSP[clIN] == cl)
            {
              clusIN[clIN] <- clusEL[cl]
            }
          }
        }
      }
      sSPmapDIR <- sprintf('%s/Maps/SOM%08d%s/agn_THR%.03f%s', folder, dy, component, threshold, sSP[s])
      dir.create(path=sSPmapDIR, showWarnings=F)
      clusINout <- sprintf('%s/Maps/SOM%08d%s/agn_THR%.03f%s/clusIN_A', folder, dy, component, threshold, sSP[s])
      save(clusIN, file=clusINout)

    # Plotting the averaged spectrum, its standard deviation and the median

    SPlogTOT <- log(SP$DSPEC[ , ])
    SPmnTOT <- rowMeans(SPlogTOT, na.rm=T)
    SPmnTOT <- exp(SPmnTOT)
    if(ampLOG == 'y')
    {
      SPmnTOT <- log10(SPmnTOT)
      addAMP <- abs(min(SPmnTOT, na.rm=T))
      SPmnTOT <- SPmnTOT + addAMP
    }

    for(nC in 1:max(clusIN))
    {

      SPlog <- log(SP$DSPEC[ ,which(clusIN == nC)])
      # searching for the clusIN elements = nC to detect the spectra projected into the nC cluster
      SPmn <- rowMeans(SPlog, na.rm=T)
      SPsd <- apply(t(SPlog), 2, sd, na.rm=T)
      SPmnPsd <- SPmn + SPsd
      SPmnMsd <- SPmn - SPsd
      SPmd <- rep(NA, length=nrow(SPlog))
      for(rm in 1:nrow(SP$DSPEC))
      {
        SPmd[rm] <- median(SPlog[rm,], na.rm=T)
      }
      SPmn <- exp(SPmn)
      SPsd <- exp(SPsd)
      SPmnPsd <- exp(SPmnPsd)
      SPmnMsd <- exp(SPmnMsd)
      SPmd <- exp(SPmd)
      if(ampLOG == 'y')
      {
        SPmn <- log10(SPmn)
        addAMP <- abs(min(SPmn, na.rm=T))
        SPmn <- SPmn + addAMP
        SPmnPsd <- log10(SPmnPsd)
        SPmnPsd <- SPmnPsd + addAMP
        SPmnMsd <- log10(SPmnMsd)
        SPmnMsd <- SPmnMsd + addAMP
        SPmd <- log10(SPmd)
        SPmd <- SPmd + addAMP
      }
      plotSPmn <- sprintf('%s/Maps/SOM%08d%s/agn_THR%.03f%s/SPmnlog%s.%s.png', folder, dy, component, threshold, sSP[s], nC, component)
      png(filename=plotSPmn, width=960, height=720)
      nSP <- length(which(clusIN == nC))
      nCL <- length(which(clusEL == nC))
      plotMain <- sprintf('Cluster␣%d,␣%d␣spectra␣in␣%d␣codes', nC, nSP, nCL)
      plot(SPmd[1:(nrow(SP$DSPEC)/2)], type='l', lwd=1.5, ylim=rangeSp, xlab='Freq␣[Hz]', ylab='Amplitude', xaxt='n', log='x', main=plotMain, col='green')
      numf <- SP$numfreqs/2
      lines(rep(0, length=numf), col='gray', lwd=1.25, lty=3)
      lines(rep(0.25, length=numf), col='gray', lwd=1.25, lty=3)
      lines(rep(0.5, length=numf), col='gray', lwd=1.25, lty=3)
      lines(rep(0.75, length=numf), col='gray', lwd=1.25, lty=3)
      lines(rep(1, length=numf), col='gray', lwd=1.25, lty=2)
      text(length(SP$freqs), 1.1, 'sd␣=␣0', cex=0.9)
      lines(rep(2, length=numf), col='gray', lwd=1.25, lty=2)
      text(length(SP$freqs), 2.1, 'sd = 1', cex=0.9)
      lines(rep(3, length=numf), col='gray', lwd=1.25, lty=2)
      text(length(SP$freqs), 2.9, 'sd = 2', cex=0.9)
      ind <- seq(1, numf, by=2^3)
      lab <- rep(NA, length(ind))
      for(nIND in 1:length(ind))
      {
        lab[nIND] <- ((ind[nIND]-1) * (SP$freq[numf]-SP$freq[1]) / (numf-1)) + SP$freq[1]
      }
      axis(side=1, at=ind, labels=ceiling(lab*10)/10)

      lines(SPmnTOT[1:numf], type='l', lwd=2.0, col='orange')
      # drawing the mean spectrum of the whole spectrogram
      lines(SPmnPsd[1:numf], type='l', lty=2, lwd=1.5, col='red')
      lines(SPmnMsd[1:numf], type='l', lty=2, lwd=1.5, col='blue')
      lines(SPsd[1:numf] + 1, type='l', lty=2, lwd=1, col='gray')
      lines(SPmn[1:numf], type='l', lwd=2.0, col='black')
      legend("topright", c("Sp mean", "Sp + sd", "Sp - sd", "sd", "Sp median", "Sp mean TOT"), lty=c(1), lwd=1.5, col=c('black', 'red', 'blue', 'gray', 'green', 'orange'))

      # picking

      limFd <- numf*pikFmin/(Fc/2)
      limFu <- numf*pikFmax/(Fc/2)
      limSPmn <- order(SPmn[limFd:limFu], decreasing=T)
      z1 <- 1
      z2 <- 1
      while(z1 < (pikN+1) && z2 <= length(limSPmn))
      {
        fm <- limSPmn[z2] + numf*pikFmin/(Fc/2) - 1
        z2 <- z2 + 1
        if(is.na(SPmn[fm-1]) == F && is.na(SPmn[fm+1]) == F)
        {
          if(SPmn[fm-1] <= SPmn[fm] && SPmn[fm+1] <= SPmn[fm])
          {
            segments(fm, -0.2, fm, 6/log(z1+7), col='black', lwd=1.3, lty=2)
            mtext(text=as.character(floor((fm*(Fc/2)/numf)*10)/10), at=fm, padj=7*log(z1+5), side=3)
            z1 <- z1 + 1
          }
        }
      }
      dev.off()
    }

    # plotting the temporal series with cluster detection

    for(clu in 1:max(clusIN))
    {
      strTSplot <- sprintf('%s/Maps/SOM%08d%s/agn_THR%.03f%s/TScl.%s%d.png', folder, dy, component, threshold, sSP[s], component, clu)
      png(filename = strTSplot, width = 1440, height = 720)
      fTOT <- SP$sig - mean(SP$sig, na.rm=T)
      fTOT <- fTOT / sd(fTOT, na.rm=T)
      PlotTSsplit(data=fTOT, dy=dy, component=component, Fc=Fc, splitTS=splitTS)
      cs <- rainbow(max(clusIN))[max(clusIN):0]
      clusINcl <- which(clusIN == clu)

```

```
        for(pt in 1:length(clusINcl))
        {
          posSp <- Ns/2 + (Ns-Nov)*(clusINcl[pt] - 1)
          x <- posSp %% splitTS
          y <- floor(posSp/splitTS) * (-range) * 4
          points(x, y, pch=20, cex=1.5, col=cs[clu])
          segments(0, -range, Ns, -range, lwd=1)
          text(Ns*2.5, -range, labels='Ns', cex=1.5, font=3)
        }
        dev.off()
      }

      strTSplot <- sprintf('%s/Maps/SOM%08d%s/agn_THR%.03f%s/TScl.%s1to%d.png', folder, dy, component, threshold, sSP[s], component, max(clusIN))
      png(filename = strTSplot, width = 1440, height = 720)
      fTOT <- SP$sig - mean(SP$sig, na.rm=T)
      fTOT <- fTOT / sd(fTOT, na.rm=T)
      PlotTSsplit(data=fTOT, dy=dy, component=component, Fc=Fc, splitTS=splitTS)
      cs <- rainbow(max(clusIN))[max(clusIN):0]
      for(el in 1:length(clusIN))
      {
        posSp <- Ns/2 + (Ns-Nov)*(el - 1)
        x <- posSp %% splitTS
        y <- floor(posSp/splitTS) * (-range) * 4
        points(x, y, pch=20, cex=1.5, col=cs[clusIN[el]])
        segments(0, -range, Ns, -range, lwd=1)
        text(Ns*2.5, -range, labels='Ns', cex=1.5, font=3)
      }
      dev.off()
    }
#   }
}
```

How to call *SPCL*, an example:

```
folder   [previously declared]
dy       [previously declared]
subSP <- c(1:1)
sSP <- sprintf('/sSP%02d', subSP)
cmp      [previously declared]
Fc       [previously declared]
thr      [previously declared]
splitTS  [previously declared]
Ns       [previously declared]
Nov      [previously declared]

SPCL(folder=folder, dy=dy, sSP=sSP, component=cmp, Fc=Fc, threshold=thr, ampLOG='y', pikFmin=2, pikFmax=10, pikN=10, splitTS=splitTS, Ns=Ns, Nov=Nov)
```

Required packages: none

Required functions: *PlotTSsplit* (see par. A.2)

Bibliography

S. I. Ameri. Topographic organisation of nerve fields. *Bulletin of mathematical biology*, 42: 339–364, 1980.

M. Attik, L. Bougrain, and R. Alexandre. Self-organizing map initialization. *Artificial Neural Networks: Biological Inspirations - Icann 2005, Pt 1, Proceedings*, 3696:357–362, 2005.

F. Barazza. *Teoria ed applicazioni di tecniche dinamiche e di decomposizione in componenti principali di serie temporali geofisiche e biomeccaniche.* Facoltà di Ingegneria, Università degli studi di Udine, Tesi di Laurea. Relatori: R. Carniel, P. B. Pascolo, 2004.

F. Barazza, P. Malisan, and R. Carniel. Improvement of h/v technique by rotation of the coordinate system. *Communications in Nonlinear Science and Numerical Simulation*, 14:182–193, 2009.

P. Y. Bard. Microtremor measurements: a tool fo site effect estimation? *The effects of surface geology on seismic motion*, pages 1251–1279, 1999.

F. Bação. Clustering census data: comparing the performance of self-organising maps and k-means algorithms. In *KD-Net Symposium, Knowledge-based services for the public sector, Bonn*, 2004.

F. Bação, V. Lobo, and M. Painho. Self-organizing maps as substitutes for k-means clustering. *Computational Science - Iccs 2005, Pt 3*, 3516:476–483, 2005.

E. Bodt, M. Verleysen, and M. Cottrell. Kohonen maps versus vector quantization for data analysis. In *ESANN*, 1997.

S. Bonnefoy-Claudet, C. Cornou, J. Kristek, M. Ohrnberger, M. Wathelet, P. Y. Bard, D. Fäh, P. Moczo, and F. Cotton. Simulation of seismic ambient vibrations: H/v and array techniques on canonical models. *13th world conference in Earthquake Engineering, Vancouver*, 2004.

R. Carniel and M. Di Cecca. Dynamical tools for the analysis of long term evolution of volcanic tremor at stromboli. *Annali di Geofisica*, 42 (3):483–495, 1999.

R. Carniel and F. Iacop. Spectral precursors of paroxysmal fases of stromboli. *Annali di geofisica*, XXXIX(2):327–345, march 1996.

R. Carniel and M. Tarraga. Can tectonic events change volcanic tremor at stromboli? *Geophysical Research Letters*, 33 (20):L20321. http://dx.doi.org/10.1029/2006GL027690, 2006.

R. Carniel, M. Di Cecca, and D. Rouland. Ambrym, vanuatu (july-august 2000): spectral and dynamical transitions on the hours-to-days timescale. *Journal of Volcanology and Geothermal Research*, 128:1–13, 2003.

R. Carniel, F. Barazza, and P. Pascolo. Improvement of nakamura technique by singular spectrum analysis. *Soil Dynamics and Earthquake Engineering*, 26:55–63, 2006.

R. Carniel, L. Barbui, and P. Malisan. Improvement of hvsr technique by self organizing map (som) analysis. *Soil Dynamics and Earthquake Engineering*, 29:1097–1101, 2009.

R. Carniel, E. Muñoz Jolis, and J. Jones. A geophysical multi-parametric analysis of hydrothermal activity at dallol, ethiopia. *Journal of African Earth Sciences*, 58 (5): 812–819, 2010.

R. Carniel, L. Barbui, and A.D. Jolly. Detecting dynamical regimes by self-organizing map (som) analysis: an example from the march 2006 phreatic eruption at raoul island. new zealand kermadec arc. *Bollettino di GeofisicaTeorica ed Applicata*, http://dx.doi.org/10.4430/bgta0077, 2012.

J.L. Carrivick, V. Manville, and S. Cronin. Modelling the march 2007 lahar from mt ruapehu. *Bulletin of Volcanology*, 71 (2):153–169 http://dx.doi.org/10.1007/s00445–008–0213–2, 2009.

B. Christenson, C. Werner, A.G. Reyes, S. Sherburn, B.J. Scott, C. Miller, M.J. Rosenburg, A.W. Hurst, and K.A. Britten. Hazards from hydrothermally sealed volcanic conduits. *Eos, Trans. Am. Geophys. Un.*, 88:53–55, 2007.

B.W. Christenson, A.G. Reyes, R. Young, A. Moebis, S. Sherburn, J. Cole-Baker, and K. Britten. Cyclic processes and factors leading to phreatic eruption events: Insights from the 25 september 2007 eruption through ruapehu crater lake, new zealand. *Journal of Volcanology and Geothermal Research*, 191:15–32, 2010.

J. W. Cooley and J. W. Tukey. An algorithm for the machine calculation of complex fourier series. *Mathematics of Computation*, vol. 19, no. 90(51-52):297–301, December 1965.

J. Dahmane, M. Meunier. Real-time video surveillance with self-organizing maps. *Computer and Robot Vision*, pages 136–143, 2005.

R. de Gelder, R. Wehrens, and J.A. Hageman. A generalized expression for the similarity spectra: application to powder diffraction pattern classification. *Journal of Computational Chemistry*, 22 (3):273–289, 2001.

W.R. Hackett and B.F. Houghton. A facies model for a quaternary andesitic composite volcano: Ruapehu, new zealand. *Bulletin of Volcanology*, 51:51–68, 1989.

A. Harris, R. Carniel, and J. Jones. Identification of variable convective regimes at erta ale lava lake. *J. Volcanol. Geotherm. Res.*, 142:207–223, 2005.

Sampsa Hautaniemi, Olli Yli-harja, Jaakko Astola, Päivikki Kauraniemi, Anne Kallioniemi, and Maija Wolf Jimmy Ruiz. *Analysis and Visualization of Gene Expression Microarray Data in Human Cancer Using Self-Organizing Maps.* 2003.

Simon Haykin. *Neural networks: A comprehensive foundation.* 1998.

J. Healy, E.F. Lloyd, C.J. Banwell, and R.D. Adams. Volcanic eruption on raoul island, november 1964. *Nature*, 205:743–745, 1965.

A.W. Hurst, H.M. Bibby, B.J. Scott, and M.J. McGuinness. The heat source of ruapehu crater lake; deductions from the energy and mass balances. *Journal of Volcanology and Geothermal Research*, 46:1–20, 1991.

J.H. Johnson and M.K. Savage. Tracking volcanic and geothermal activity in the tongariro volcanic centre, new zealand, with shear wave splitting tomography. *Journal of Volcanology and Geothermal Research*, 223-224:1–10, 2012.

A.D. Jolly, S. Sherburn, P. Jousset, and G. Kilgour. Eruption source processes derived from seismic and acoustic observations of the 25 september 2007 ruapehu eruption - north island, new zealand. *Journal of Volcanology and Geothermal Research*, 191:33–45, 2010.

J. Jones, R. Carniel, A. J. L. Harris, and S. Malone. Seismic characteristics of variable convection at erta ale lava lake, ethiopia. *Journal of volcanology and geotermal research*, 153:64–79, 2006.

L. Kaufman and P.J. Rousseeuw. *Finding Groups in Data: An Introduction to Cluster Analysis (chapter 5).* 1990.

G. Kilgour, V. Manville, F. Della Pasqua, A. Graettinger, K.A. Hodgson, and G.E. Jolly. The 25 september 2007 eruption of mount ruapehu, new zealand: directed ballistics, surtseyan jets, and ice-slurry lahars. *Journal of Volcanology and Geothermal Research*, 191:1–14, 2010.

C. D. Klose. Self-organizing maps for geoscientific data analysis: geological interpretation of multidimensional geophysical data. *Computational Geosciences*, 10(3):265–277, 2006.

T. Kohonen. Self-organizing maps: Optimization approaches. in t. kohonen, k. makisara, o. simula, & j. kangas (eds.). In *Proceedings of ICANN91 International conference on artifial neural networks vol. 2 (pp. 981-990)*, 1991.

T. Kohonen. *Self-Organizing Maps.* Springer-Verlag New York, 2001.

Teuvo Kohonen. Automatic formation of topological maps of patterns in a self-organizing system. In *Proceedings of the 2nd Scandinavian Conference on Image Analysis (Espoo)*, 1981.

K. Konno and T. Ohmachi. Ground-motion characteristics estimated from spectral ratio between horizontal and vertical components of microtremor. *Bulletin of the Seismological Society of America*, 88:228–241, 1998.

S. L. Kramer. *Geotechnical Earthquake Engineering.* Prentice Hall, 1996.

C. Lachet and P.Y. Bard. Numerical and teoretical investigations on the possibilities and limitationd of nakamura's techinque. *Journal of Physics of the Earth*, 42(9):377–397, 1994.

J.C. Lahr, B.A. Chouet, C.D. Stephens, J.A. Power, and R.A. Page. Earthquake classification, location, and error analysis in a volcanic environment: implications for the magmatic system of the 1989-1990 eruptions at redoubt volcano, alaska. *J. Volcanol. Geotherm. Res.*, 62:137–151, 1994.

H. Langer, S. Falsaperla, A. Messina, S. Spampinato, and B. Behncke. Detecting imminent eruptive activity at mt etna, italy, in 2007-2008 through pattern classification of volcanic tremor data. *Journal of Volcanology and Geothermal Research*, 200:1–17, 2011.

G. Lanzo and F. Silvestri. *Risposta sismica locale. Teoria ed esperienze.* Hevelius, 1999.

L.M. Lees. Seismic time series analysis tools, rseis package for r software. *http://cran.r-project.org/web/packages/RSEIS/*, 2012.

Pasi Lehtimaki and Kimmo Raivio. *A SOM based approach for visualization of GSM network performance data.* 2005.

E.F. Lloyd and S. Nathan. Geology and tephrochronology of raoul island, kermadec group, new zealand. *New Zealand Geol. Surv.*, Bull. 95:105, 1981.

V. Manville and S.J. Cronin. Breakout lahar from new zealand's crater lake. *EOS. Transactions of the American Geophysical Union*, 88 (43), 2007.

A. Messina and H. Langer. Pattern recognition of volcanic tremor data on mt. etna (italy) with kkanalysis - a software program for unsupervised classification. *Computers & Geosciences*, http://dx.doi.org/10.1016/j.cageo.2011.03.015, 2011.

A. Mordret, A.D. Jolly, Z. Duputel, and N. Fournier. Monitoring of phreatic eruptions using interferometry on retrieved cross-correlation function from ambient seismic noise: results from mt. ruapehu, new zealand. *Journal of Volcanology and Geothermal Research*, 191:46–59, 2010.

M. Mucciarelli and M. Gallipoli. A critical review of 10 years of microtremor hvsr technique. *Bollettino Di Geofisica Teorica ed Applicata*, 42:255–266, 2001.

Y. Nakamura. A method for dynamic characteristic estimation of subsurface using microtremor on the ground surface. *Quarterly report of railway technical research institute*, 30(1):25–33, 1989.

Y. Nakamura. Clear identification of fundamental idea of nakamura's technique and its application. *Proceedings of the 12th World Conference of Earthquake Engineering*, 2000.

J. Neuberg and C. O'Gorman. A model of the seismic wavefield in gas-charged magma: application to soufriere hills volcano, montserrat, in the eruption of soufriere hills volcano, from 1991 to 1999. *In: Druitt, T.H., Kokelaar, B.P. (Eds.), Geol Soc. Mem.*, 21: 603–609, 2002.

M. Nogoshi and T. Igarashi. On the amplitude characteristics of microtremors. *Journal of the Seismological Society of Japan*, 24:24–40, 1971.

M. Ripepe, A. J. L. Harris, and R. Carniel. Thermal, seismic and infrasonic evidences of variable degassing rates at stromboli volcano. *Journal of Volcanology and Geothermal Research*, 118:285–297, 2002.

H. Ritter, T. Martinetz, and K.Schulten. *Neural computation and self-organizing maps: an introduction.* 1992.

SESAME. Guidelines for the implementation of the h/v spectral ratio technique on ambient vibrations measurements, processing and interpretation. *http://sesame-fp5.obs.ujf-grenoble.fr/*, 2005.

S. Sherburn, C.J. Bryan, A.W. Hurst, and J.H. Latter B.J. Scott. Seismicity of ruapehu volcano, new zealand, 1971-1996: a review. *Journal of Volcanology and Geothermal Research*, 88:255–278, 1999.

I. Smith, R.B. Stewart, R.C. Price, and T.J. Worthington. Are arc-type rocks the products of magma crystallisation? observations from a simple oceanic arc volcano: Raoul island, kermadec arc, sw pacific. *J. Volcanol. Geotherm.*, 190:219–234, 2010.

M. Strickert and U. Seiffert. Correlation-based data representation. *Dagstuhl Seminar Proceedings 07131*, 2007.

R Development Core Team. *R: a language and environment for statistical computing.* Vienna, Austria: R Foundation for Statistical Computing. Retrieved from http://www.R-project.org, 2010.

K. Tokimatsu. Geotechnical site characterization using surface waves. *Earthquake Geotechnical Engineering*, pages 1333–1368, 1997.

K. Tokimatsu, S. Tamura, and H. Kjima. Effects of multiple modes on rayleigh wave dispersion characteristic. *J.Geotechnical Engineering*, 118:1529–1543, 1992.

A. Ultsch. Self-organizing neural networks for visualization and classification. In *Proc. Conf. Soc. for Information and Classification, Dortmund*, 1992.

A. Ultsch. *Self-Organizing Neural Networks for Visualization and Classification*, chapter Information and Classification: Concepts, Methods, and Applications, pages 307–313. eds O. Opitz, B. Lausen and R. Klar, 1993.

A. Ultsch. U*-matrix: a tool to visualize clusters in high dimensional data. *University of Marburg, Department of Computer Science, Technical Report*, 36:1–12, 2003.

A. Ultsch and H.P. Siemon. Kohonen's self organizing feature maps for exploratory data analysis. *Proc. Intern. Neural Networks, Kluwer Academic Press, Paris*, pages 305–308, 1990.

E. A. Uriarte and F. D. Martín. Topology preservation in som. *Proceedings of world academy of science, engineering and technology*, 15:1–4, 2006.

J. Vesanto, J. Himberg, E. Alhoniemi, and J. Parhankangas. Som toolbox for matlab 5. report a57. Technical report, Helsinki University of Technology, Helsinki, Finland, 2000.

J. Vesanto, M. Sulkava, and J. Hollmn. On the decomposition of the self-organizing map distortion measure. In *In Proceedings of the workshop on self-organizing maps (pp. 11-16)*, 2003.

P.D. Welch. The use of fast fourier transform for the estimation of power spectra: a method based on time averaging over short, modified periodograms. *IEEE Trans. Audio and Electroacoust*, AU-15:70–73, 1967.

T.J. Worthington, M.R. Gregory, and V. Bondarenko. The denham caldera on raoul volcano: dacitic volcanism in the tonga-kermadec arc. *J. Volcanol. Geotherm.*, 90:29–48, 1999.