Initialisation and Multi-Layer Clustering of Spatial Data

A Better Method

Department of Computer Science
Software Engineering
Aalborg University
Spring 2011

Aalborg University
Department of Computer Science
Selma Lagerlöfs Vej 300
9220 Aalborg
Telephone:(45)96358080
http://www.cs.aau.dk

Title:
Initialisation and Multi-Layer Clustering
of Spatial Data
A Better Method

Theme:
Cluster Analysis

Project timeframe:
SW10, 1st February 2011 - $9th^{th}$ June, 2011

Project group:
f11d627a

Group members:
Simon Piepgras Lyager
Michael Wodstrup Vandborg

Supervisor:
Manfred Jaeger

Abstract:

In this report, an approach to spatial clustering based on Markov Random Fields will be proposed. The report has two main foci, multi-layer clustering, performed on geofloral data, as well as improved initialisation, performed on image data.
A superior technique for clustering of spatial data in multiple layers is designed, using logistic regression, with the capabability to handle both categorical and numerical data.
A group of initialisation techniques for image clustering is developed using histograms and selection heuristics, with a focus on the initial colour model prior to segmentation having as different models for each segment as possible.

Copies: 4

Total pages: 104

Appendices: 3

Supplements: 1

Paper finished: 8th June, 2011.

The content of this paper is freely available, but publication is only permitted with explicit permission from the authors.

Signatures

[illegible]

Simon Piepgras Lyager

Michael Wodstrup Vandborg

Preface

This report is written by a group of software engineer students at the Computer Science department of Aalborg University during the spring semester SW10 2011. The project started February 1st 2011 and ended June 9th 2011. The overall topic is *Clustering Analysis.*

The report and the work done herein is partly based on the report *Clustering Analysis of Spatial Data*[1], written in the previous semester. Reading the previous report may benefit the reader's understanding of the content of this report.

References are written in square brackets like: [2], and their reference can be found at the end of the report.

We would like to thank Manfred Jaeger for supervising this project.

Contents

1 Introduction **1**

1.1 Motivation 2

I Analysis **3**

2 Data **4**

2.1 Image Data 4

2.2 Geofloral Data Set 4

3 Markov Random Field **7**

3.1 General 7

3.2 Markov Property 7

3.3 Neighbourhood systems 7

3.4 Defining the Joint Distribution 9

4 Segmentation **11**

4.1 Specialising the Markov Random Field Model 11

4.2 Ising Model 12

4.3 Observed Values 13

4.4 Using the Specialised Markov Random Field Model 14

5 Multinomial MRF **16**

5.1 Model 16

5.2 Using the Multinomial Markov Random Field Model 18

6 Logistic Regression **19**

7 Histogram Initialisation **21**

8 Likelihood Value **26**

II Implementation **30**

9 Architecture **31**

9.1 Data 31

9.2 Site 32

9.3 Potential 32

10 SPSS **33**

10.1 Implementation 33

10.2 Submitting Data to SPSS 34

10.3 Executing the Logistic Regression 35

10.4 Retrieving Warnings and Errors 36

10.5 Performance Issues 37

11 Histogram Initialisation **38**

III Experimentation **40**

12 Histogram Initialisation **41**

12.1 Synthetic Images 41

12.2 Real World Images 46

13 Multi-Layer Geofloral Segmentation **54**

13.1 Synthetic Data Experiments 54

13.2 Real World Experiments 62

IV Conclusion **68**

14 Conclusion **69**

15 Future Development **70**

V Appendices **72**

Appendix A Similarity of Segmentations **73**

A.1 Implementation 73

Appendix B Domain Information on Switzerland **76**

Appendix C Characteristic Plants **77**

Bibliography 80

VI Supplements 81

Appendix D Factorial Clustering with an Application to Plant Distribution Data 82

1 Introduction

Clustering involves analysing a given data set and finding subsets, referred to as clusters that have similar attributes. A subcategory of this is spatial clustering, i.e. clustering of data where data points correspond to points on a given 2D grid. Clustering of spatial data can be applied in numerous different scenarios. Examples of usage range from preprocessing images for optical character recognition (OCR) to recognition of objects such as houses and roads in satellite pictures. It has also been applied within the field of medicine, where brain MR images can be automatically segmented.[3]

We propose a multi-layer spatial clustering model based on a Markov Random Field model, where observed variables are modelled by either a multinomial model or a logistic regression model. The intent is to separate factors of the data into individual layers, e.g. for spatial botanical data, one layer could describe the pH-value of the soil and another layer could describe average temperatures.

We will focus on two types of spatial data; image and geofloral data. For image data, we will primarily focus on initialisation, i.e. learning the observed variable model that will be used for the first clustering iteration. We have chosen image data for initialisation problems, because for most images we are able to intuitively identify a segmentation. Due to this it is easier to verify whether the initialisation is meaningful, e.g. for an image of a house, a meaningful initialisation could separate the house and background. For geofloral data, we will focus on clustering with a multi-layer spatial model. This is because geofloral data depends on a multitude of factors, e.g. temperatures, pH-values, precipitation, topography, etc.

This report builds on our previous SW9 project, *Cluster Analysis of Spatial Data*[1], in which we already developed a clustering method based on a Markov Random Field model that could successfully cluster single layers. A multinomial multi-layer model was proposed but proved to be unable to separate factors as intended. Additionally, we used initialisation based on a random segment configuration. We will perform tests with approaches from this report as well as those from the previous to showcase any improvements.

1.1 Motivation

We develop a multi-layer spatial clustering model with observed variables modelled using logistic regression. We expect the more sophisticated method to be able to separate factors into different layers as intended. Additionally, we develop initialisation methods using histograms and heuristics.

In our previous work in this field, a basic clustering method was developed for images and geofloral data. Additionally, attempts were made to develop a multi-layer clustering model, with little success. In this report, we intend to improve upon these methods to create a functional multi-layer clustering model.

Additionally, in the previous report, random initialisation was found to cause scenarios, where our clustering algorithm would be unable to escape a local maximum. In this report we will attempt to improve upon the initialisation methods using histograms and heuristics. We will focus on images as there is an intuitive method of approximately verifying results.

Part I

Analysis

2 Data

This chapter describes the data used for both types of segmentation. Note that this chapter was originally a part of the preceding report on this subject, *Cluster Analysis of Spatial Data*[1], with minor changes to reflect the modifications in the code since.

2.1 Image Data

The system is capable of handling Bitmap (BMP) and Portable Network Graphics (PNG) files, as these are natively supported in C#. Further support of image formats would be trivial to implement, but as they do not provide additional knowledge, we have chosen to not implement further support.

When we access image data, the image is stored in an instance of the native `Bitmap` class. A given pixel can then be retrieving by providing an X and Y values, indicating the location of the pixel in the image. We create observed variables from the image data by iterating over the X and Y coordinates, creating an instance of the `Site` class for each pixel. Each of these instances contains the observed colour of their respective pixel and a unique ID constructed from the X and Y value. Each pixel also contains references to its neighbours.

2.2 Geofloral Data Set

The data set used was a combination of two sources. One was a data file from *Swiss Web Flora*[4], containing distribution data for 2697 plants over 565 regions, with the possible values no data, rare, frequent, registered until 1984, registered until 1995, registered until 1998 and registered until 2000. Plants that were not categorised as frequent in any region were discarded, leaving 2398 plants in the data set. Of the 565 regions, 350 are categorised as below the treeline and 215 as above the treeline. However, this information is not used during clustering.

Additionally, we had a graphics file in the XFIG format containing the physical shapes of the regions. This was created as a part of *Clustering Bio-*

geographic Data Using Relational Model[5]. The regions mapped in the XFIG file can be seen in Figure 2.1.

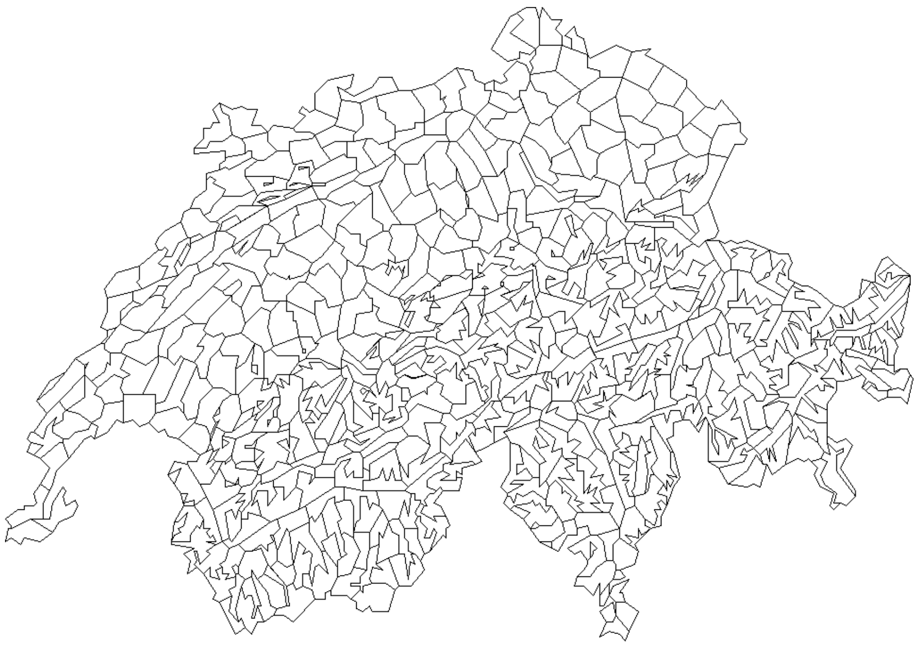


Figure 2.1: A drawing of Switzerland divided into 565 regions with shapes from the XFIG file[5]

Both the plant data and the XFIG file were combined into a single, custom format. This is an XML file, following a specific pattern.

First, there is an XML declaration. Following that, there is an outer tag called MRFData. Contained within is the metadata for the file, usually simply the amount of plants, and the ID of each. Additionally, we have an array of sites, each in its own Site tags. These have an ID attribute, corresponding to the region number in the plant data and XFIG file. Inside these are three groupings, Points, Neighbours and DiscreteVariables. Points have child nodes called Point with an X and Y attribute. Neighbours have Neighbour children with an ID attribute corresponding to the ID of the neighbouring site. DiscreteVariable have an ID and Value attribute, corresponding to the ID and value for the site in the plant data file.

This format maps closely to the attributes of the `Site` class. The points form a polygon for illustration purposes, the neighbours and plants form the data necessary for segmentation.

An example of the format is seen in Listing 2.1.

```xml
<?xml version="1.0">
<MRFData>
    <MRFMetaData>
        <DiscreteVariableCount>4</DiscreteVariableCount>
        <DiscreteVariables>
            <DiscreteVariable ID="3" Name="Lycopodium Annotinum" />
            <DiscreteVariable ID="22" Name="Equisetum telmateia " />
            <DiscreteVariable ID="55" Name="Athyrium distentifolium " />
            <DiscreteVariable ID="59" Name="Cystopteris montana " />
        </DiscreteVariables>
    </MRFMetaData>
    <Sites>
        <Site ID="108" AboveTreeLine="false">
            <Points>
                <Point X="765" Y="5535" />
                <Point X="945" Y="5265" />
                <Point X="1170" Y="5130" />
                <Point X="990" Y="5355" />
                <Point X="765" Y="5535" />
            </Points>
            <Neighbours>
                <Neighbour ID="107" />
            </Neighbours>
            <DiscreteVariables>
                <DiscreteVariable ID="3" Value="3" />
                <DiscreteVariable ID="22" Value="3" />
                <DiscreteVariable ID="55" Value="3" />
                <DiscreteVariable ID="59" Value="3" />
            </DiscreteVariables>
        </Site>
    </Sites>
</MRFData>
```

Listing 2.1: Example of the geofloral data format.

3 Markov Random Field

In this chapter, we will describe the unordered graph model, Markov Random Field (MRF). Note that this chapter was originally a part of the preceding report on this subject, *Cluster Analysis of Spatial Data*[1].

3.1 General

In general, an MRF is an undirected graph consisting of random variables that satisfy the Markov property. The Markov property will be explained in Section 3.2. Thus, an MRF is a probabilistic model where dependencies between the random variables can be mapped. A neighbourhood system for an MRF relates the random variables to each other.[2] A variety of neighbourhood systems are explained in Section 3.3. Given any MRF, a joint probability distribution can be calculated for it.

3.2 Markov Property

In order for an undirected graph to be classified as an MRF it must fulfil the *Markov property* that can be seen in Equation 3.1, where x_i denotes a hidden variable, x_k is any hidden variable in x other than x_i and N_i is the neighboorhood of i. It states that the sites, S, in an MRF are conditionally independent of each other, given their neighbourhood. It follows from this that given all other hidden variables, each non-adjacent site in an MRF is conditionally independent.

$$P(x_i|\forall x_k \in x, x_i \neq x_k) = P(x_i|N_i) \tag{3.1}$$

3.3 Neighbourhood systems

The sites, S, in an MRF are related to each other through a neighbourhood system, N. This system can be defined in a variety of ways and we will now examine these neighbourhood systems and the cliques they can contain. In general, an MRF is a graph with vertices and edges. Formally, a neighbourhood system can be defined as in Equation 3.2[2], where $i \notin N_i$ and $i \in N_{i'} \iff i' \in N_i$. As can be seen, each site, i, has a unique neighbourhood, N_i, associated with it.

$$N = \{N_i | \forall i \in S\} \tag{3.2}$$

Depending on the placement of the sites, different neighbourhoods are defined. In Figure 3.1(a), the sites in an MRF are placed on a regular lattice. Figure 3.1(b) and Figure 3.1(c) illustrate a four and eight neighbourhood system for Figure 3.1(a), respectively. Dark grey nodes indicate sites that are part of the neighbourhood of site e.

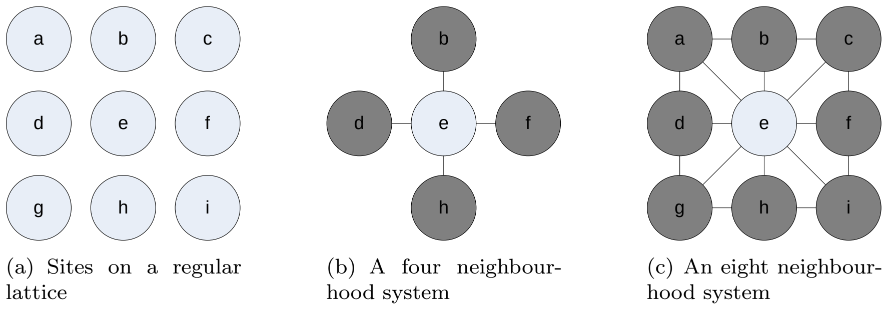

a
b
c
d
e
f
g
h
i
(a) Sites on a regular lattice
b
d
e
f
h
(b) A four neighbourhood system
a
b
c
d
e
f
g
h
i
(c) An eight neighbourhood system

Figure 3.1: Neighbourhood systems on a regular lattice.

A four neighbourhood system on a regular lattice can be defined as in Equation3.3.[2, P.23] Note that the sites near the edges have three neighbours, the sites at the corners have two.

$$N_{x,y} = \{(x-1, y), (x+1, y), (x, y-1), (x, y+1)\} \tag{3.3}$$

Neighbourhoods can also be defined for irregular sites. In this case, the neighbourhood of a site, N_i, is of arbitrary size. One method of finding the neighbours of a site, i, is to select all sites within a distance, r, of i. However for certain spatial data other methods are required, for instance when handling geographical data, where region borders define neighbours.

An MRF consisting of irregular sites is illustrated in Figure 3.2. As some sites have multiple neighbours and other sites only have a single neighbour, this will result in different cliques within the neighbourhood systems. The neighbourhood for a few sites of Figure 3.2 can be found in Table 3.1.

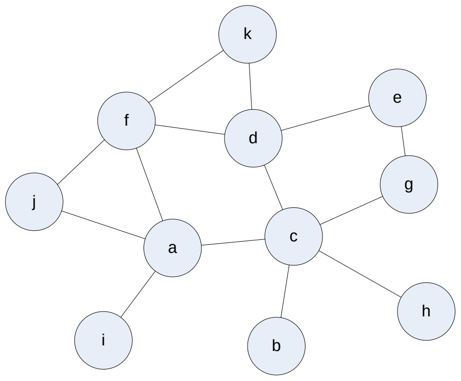


Figure 3.2: An MRF consisting of irregular sites.

Site	C_1	C_2	C_3
a	$\{a\}, \{c\}, \{f\}, \{i\}, \{j\}$	$\{a, c\}, \{a, f\}, \{a, i\}, \{a, j\}$	$\{a, f, j\}$
e	$\{d\}, \{e\}, \{g\}$	$\{d, e\}, \{e, g\}$	$\{\}$

Table 3.1: A table containing examples of neighbourhood cliques for Figure 3.2.

3.4 Defining the Joint Distribution

A joint distribution calculation determines the probability of a specific configuration f being correct, i.e. the higher this is, the closer the configuration is to the right one.

The joint distribution probability of an MRF can be calculated, among other ways, by using a Gibbs distribution. This is due to the equivalence of MRFs and Gibbs distributions[2, S. 2.1.4] as per the Hammersley-Clifford theorem[6, Theorem 1].

First, let us look at how at how a Gibbs distribution is defined. This is shown in Equation 3.4. Z is called the partition function, and works as a normalising constant, while $U(f)$ is called the energy function. It is the sum of the clique potentials $V_c(f)$ over all possible cliques $\mathcal{C}$. In these, T is the temperature, in most cases assumed to be 1, which controls the sharpness of the distribution, and $\mathbb{F}$ is the set of all possible configurations.

$$\begin{aligned} P(f) &= Z^{-1} \cdot e^{-\frac{1}{T}U(f)} \\ Z &= \sum_{f \in \mathbb{F}} e^{-\frac{1}{T}U(f)} \\ U(f) &= \sum_{c \in \mathcal{C}} V_c(f) \end{aligned} \tag{3.4}$$

$P(f)$ calculates the probability of a configuration, f. To calculate it, we first need to calculate Z. However, when working with discrete values, $\mathbb{F}$ contains a combinatorial number, that quickly becomes too high to feasibly calculate. Instead, it is approximated using a number of techniques.[2, P.26]

4 Segmentation

This chapter describes the extensions and specialisations of the MRF model to enable using it for segmentation. Note that this chapter was originally a part of the preceding report on this subject, *Cluster Analysis of Spatial Data*[1].

4.1 Specialising the Markov Random Field Model

An MRF is often used in labelling problems such as image segmentation. This is due to the fact that an MRF is structured in a way that allows observable variables such as the intensity of an image or the presence of a plant to be easily represented.

An MRF for segmentation is organised in sites, each consisting of one node connecting to the other sites, called the hidden variable or x, and a number of nodes connected to this one, called the observed values or y. An example of this is shown in Figure 4.1. The observed values are the data known initially, e.g. the red, green and blue values of a pixel or the presence of different plants. The hidden variable is unknown initially, and represents the segment the site would belong to in an optimal configuration. The goal of segmentation is to find the best value of the hidden variable.

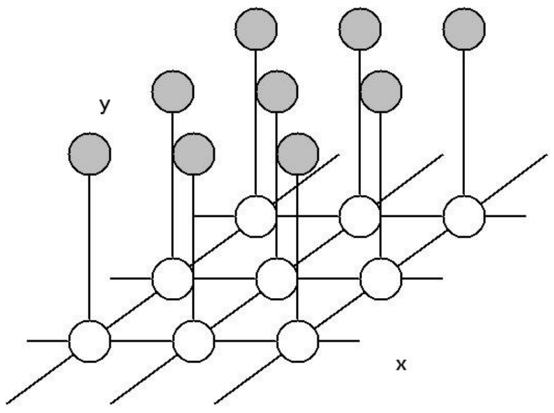


Figure 4.1: A Markov Random Field. The grey nodes represent observed variables. The white nodes represent hidden variables. Source: [7]

As seen in Figure 4.1, there are two distinct types of cliques. One is the type consisting of a hidden variable and an observed variable, $\{x_i, y_{i,k}\}$, where k indicates which observed value of the site i the clique contains. This will be called an observation clique from now on. The other is the type consisting of two hidden variables, $\{x_i, x_j\}$, where j is a neighbour of i. This will be called a neighbourhood clique. It should be noted that we only focus on 2-clique neighbourhood cliques, to reduce the computational complexity.

We define the neighbourhood clique potential as $V_C(x_i, x_j)$. We likewise define the observation potential as $V_C(x_i, y_{i,k})$.

We can specialise the Gibbs distribution w.r.t. segmentation by using these potentials in the energy function, as seen in Equation 4.1. It should be noted that the temperature $-\frac{1}{T}$ has been excluded, as it has no bearing on our use of the model. Additionally, it should be noted that since the neighbourhood and observation potentials refer to a specific site i, we sum over this variable as well.

$$\begin{aligned} P(f) &= Z^{-1} \cdot e^{U(f)} \\ Z &= \sum_{f \in \mathbb{F}} e^{U(f)} \\ U(f) &= \sum_i \sum_j V_c(x_i, x_j) + \sum_i \sum_k V_c(x_i, y_{i,k}) \end{aligned} \tag{4.1}$$

4.2 Ising Model

This section is based on [6, S. 1.3.1].

The Ising Model is a joint density model for binary data. It represents the probability of a given site having a given binary value, depending on which binary values its neighbours have. Traditionally, this is used for black and white images, but in our case, we will use it for whether or not a given site is a part of the segment.

For a given clique of two hidden variables, if they are the same, the potential of the clique is 1. If they are different, it is 0. This is represented in Equation 4.2, where $I_[x_i = x_j]$ is 1 if they are the same, and 0 if not. β is a weight, which changes the importance of the neighbourhood in relation to other cliques. Higher values lead to more coherence in clusters.

$$V_c(x_i, x_j) = \beta \cdot I_[x_i = x_j] \tag{4.2}$$

Following from Equation 3.4, if we insert this clique potential into a Gibbs distribution, we get Equation 4.3, where $i \backsim j$ represents each neighbour j of i. As should be evident, this is essentially the Ising model. It should be noted, that β has been moved out out the summation.

$$\begin{aligned} P(i) &= Z^{-1} \cdot e^{U(i)} \\ Z &= \sum_x e^{U(i)} \\ U(i) &= \beta \cdot \sum_{i \backsim j} I_[x_i = x_j] \end{aligned} \tag{4.3}$$

4.3 Observed Values

This section deals with specifying the observation potential $V_c(x_i, y_{i,k})$.

An observed value can be divided into two types, continuous and discrete. Continuous values fall within a range, e.g. the red component of a pixel will fall within the [0, 255] range. Discrete values are a set of labels, e.g. "true" and "false" for indicating the presence of a plant.

For both continuous and discrete values, we find the probability of a site having a hidden variable, given the observed variable.

The probability function for a continuous value is a Gaussian distribution, see Equation 4.4.

$$V_c(x_i, y_i) = \frac{1}{\sqrt{2\pi\sigma_s^2}} e^{\dfrac{(y_i - \mu_s)^2}{2\sigma_s^2}} \tag{4.4}$$

The probability function for a discrete value is shown in Equation 4.5.

$$V_c(x_i, y_i) = p_{x_i, y_i} \tag{4.5}$$

For both data types, the probability is based on learning the distribution of values of the variable across all sites in a segment. Equation 4.6 and 4.7 defines the mean and variance, respectively, of a continuous value, and Equation 4.8 defines the probability of a discrete value being a given label.

$$\mu_s = \frac{1}{n} \sum_{i:x_i=s} y_i \tag{4.6}$$

$$\sigma_s^2 = \frac{1}{n} \sum_{i:x_i=s} (y_i - \mu_s)^2 \tag{4.7}$$

In Equation 4.6 and 4.7, s is the segment ID and n is the number of sites tied to the segment.

$$p_{s,l} = \frac{1}{n} \sum_{\substack{i:x_i=s \\ y_i=l}} 1 \tag{4.8}$$

In Equation 4.8, l is the label the potential is being calculated for. A special case holds for binary discrete values. Only the probability for one label needs to be calculated, as the other one will be $p_{false} = 1 - p_{true}$.

4.4 Using the Specialised Markov Random Field Model

Expanding on the specialised Gibbs distribution in Equation 4.1, we need to insert the potentials.

Given that the observation probabilities are not strictly speaking potentials, we need to extract $P(x)$ from $P(f) = P(x,y)$. This is done as shown in Equation 4.9.

$$P(x,y) = P(x)P(y|x) \tag{4.9}$$

With $P(x)$ depending only on the neighbourhood potential, it can be defined as a Gibbs distribution over these nodes as shown in Equation 4.10.

$$P(x) = Z^{-1}e^{\beta \cdot \sum_{i \backsim j} V_c(x_i, x_j)} \tag{4.10}$$

The probability of y given x is actually the observation probability as seen in Equation 4.11.

$$\begin{aligned} P(y|x) &= V_c(x,y) \\ P(y|x) &= \prod_i \prod_k V_c(x_i, y_{i,k}) \end{aligned} \tag{4.11}$$

Combining Equation 4.10 and Equation 4.11 and rewriting them so that they take the form of a Gibbs distribution can be seen in Equation 4.12.

$$\begin{aligned} P(f) &= Z^{-1}e^{\sum_i \sum_{i \backsim j} V_c(x_i, x_j)} \cdot \prod_i \prod_k V_c(x_i, y_{i,k}) \\ P(f) &= Z^{-1}e^{\sum_i \sum_{i \backsim j} V_c(x_i, x_j)} \cdot e^{\sum_i \sum_k ln(V_c(x_i, y_{i,k}))} \\ P(f) &= Z^{-1}e^{\sum_i \sum_{i \backsim j} V_c(x_i, x_j) + \sum_i \sum_k ln(V_c(x_i, y_{i,k}))} \end{aligned} \tag{4.12}$$

Here, f is a configuration, i.e. a set of values for the different variables in the graph. We need to find the set of segment values for the different hidden variables that gives us the highest result of $P(f)$.

This gives us the benefit of being able to eliminate Z, as this will be the same for all configurations, and as such, in direct comparisons it will have no bearing on the final result.

However, even with Z eliminated, the number of different configurations f needing to be checked is exponential with regards to the hidden variables, and as such, the method is computationally unfeasible.

To eliminate this problem, we use a heuristic composed of calculating P for a single site i. Again, we need to find the segment value giving the highest value, so the function becomes $P(x_i)$. This change, and the Z change, is reflected in Equation 4.13.

$$\begin{aligned} P(i) &= e^{N(i)+O(i)} \\ N(i) &= \beta \sum_{i \backsim j} I_[x_i = x_j] \\ O(i) &= \sum_k ln(V_c(x_i, y_{i,k})) \end{aligned} \tag{4.13}$$

For each iteration, we select a number of sites randomly, iterating over the segments, assuming that the hidden variable x_i is that segment, and calculate the probability using $P(i)$. After the probabilities for all segments over all selected sites have been calculated, the sites are set to new segments based on which segment has the highest probability.

Over a series of iterations, as the changes from one configuration to the next become less dramatic, the configuration will be converging to a local maximum probability distribution.

```
Randomly assign all sites to a segment
Estimate distributions of observed values
for k = 1 to $n_it$ ; Iterate iterations
  for i = 1 to $n_si$ ; Iterate sites
    Randomly select site
    for s = 1 to $n_se$ ; Iterate segments
      Calculate probabilities based on Equation 4.13
  Set segments of sites according to probabilities
  Re-estimate distributions
```

Listing 4.1: Pseudocode for MRF segmentation.

5 Multinomial MRF

This chapter describes the extension of the segmentation MRF model into a multi-layered Multinomial Markov Random Field (MMRF) model. Note that this chapter was originally a part of the preceding report on this subject, *Cluster Analysis of Spatial Data*[1].

5.1 Model

We will expand on the specialised MRF model from Section 4.4 in this section. We need to alter the potentials so that they properly take layers into account. As seen in Figure 5.1, the hidden variable from each layer of a site is connected to the observed variable(s) of that site. Furthermore, the hidden variables of a site are all connected. i.e. layer$_2$ in the figure is connected to layer$_1$ and layer$_3$ and so on.

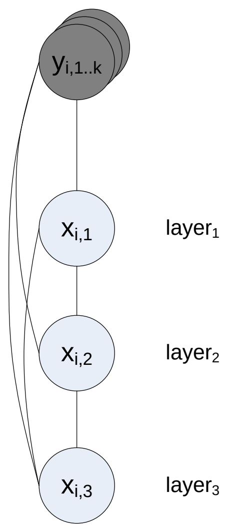


Figure 5.1: A site in a multinomial MRF with three layers.

The observation potential of a site will then depend upon the configuration of the hidden variable of each layer of that site. The model in Section 4.4 relied solely on the configuration of a single hidden variable for each site and thus the number of potentials for each site was the number of segments. In this model the potentials will be stored in an n-dimensional matrix, where n is the number of layers. The total size of a potential per site will be the product of the number of segments for each layer.

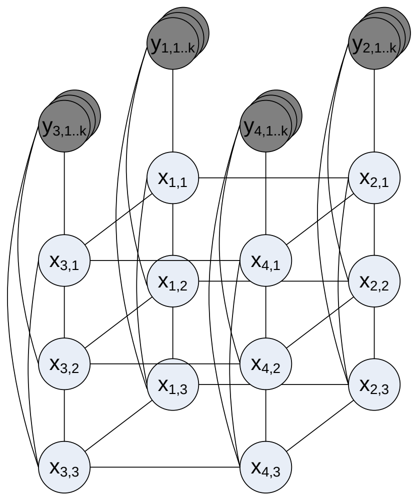


Figure 5.2: A multinomial MRF with three layers containing four sites.

The neighbourhood of a site will consist of independent neighbourhoods for each layer. Each layer has a neighbourhood potential weight, β_l associated with it. In Figure 5.2, an MRF consisting of four sites and three layers is depicted. In the figure, the neighbourhood potential for the site with an index of 3, would be the sum of the Ising model from Section 4.2 applied to each layer independently, with the appropriate weight applied.

See Equation 5.1 for the specialised multinomial MRF model, where l is the number of layers and $\mathbb{L}_i$ is a configuration of segment IDs in the different layers for the site i.

$$\begin{aligned}
P(f) &= Z^{-1} \cdot e^{N(f)+O(f)} \\
Z &= \sum_{f \in \mathbb{F}} e^{N(f)+O(f)} \\
N(f) &= \sum_i \sum_l \beta_l \sum_{i \backsim j} I_{[}x_{i,l} = x_{j,l}] \\
O(f) &= \sum_i \sum_k ln(V_c(\mathbb{L}_i, y_{i,k})) \\
\mathbb{L}_i &= \{x_{i,1}, x_{i,2} \dots x_{i,l}\}
\end{aligned} \tag{5.1}$$

5.2 Using the Multinomial Markov Random Field Model

The approach for using the MMRF model is similar to that for using the segmentation MRF model, seen in Section 4.4. The changes performed there are likewise performed on the MMRF model, resulting in the equation seen in Equation 5.2, where $\mathbb{L}_i$ is an ordered set for a site, i, consisting of a segment ID for each layer, i.e. $\mathbb{L}_i = (x_{i,1}, x_{i,2} \dots x_{i,l})$.

$$\begin{aligned} ln(P(\mathbb{L}_i)) &\approx N(\mathbb{L}_i) + O(\mathbb{L}_i) \\ N(\mathbb{L}_i) &= \sum_l \sum_{i \backsim j} I_[x_{i,l} = x_{j,l}] \\ O(\mathbb{L}_i) &= \sum_k ln(V_C(\mathbb{L}_i, y_{i,k})) \end{aligned} \tag{5.2}$$

We take the natural logarithm of both sides of the equation. This is in order to prevent an implementation issue, where the exponential of the sum of the potentials would result in an overflow.

Furthermore, instead of $P(x_i)$, the equation uses $P(\mathbb{L}_i)$, the set of all hidden variables in the site i. This is because we no longer look at single segmentation values, but the combination of segment values for the site i. It should be noted that like the problem with the single layer segmentation model, the number of combinations of segment IDs for the site is exponential with regards to the number of layers. However, as this is not expected to rise above 10, this is not expected to be a problem.

Additionally, the observation potential uses the set of hidden variables as well, corresponding to the observation clique consisting of one observed variable and all hidden variables in a site.

These changes do not affect the pseudocode to a great extent. Once the score for each set of segment values has been calculated, the highest set is chosen, and the hidden variables of the site are set according to this. The final pseudocode can be seen in Listing 5.1.

```
Randomly assign all sites to a combination of segments $\mathbb{L}$
Estimate distributions of observed values
for k $= 1$ to $n_it$ ; Iterate iterations
  for i $= 1$ to $n_si$ ; Iterate sites
    Randomly select site
    for s $= \{1, 1 \dots 1\}$ to $\{n_{se_1}, n_{se_2} \dots n_{se_l}\}$ ; Iterate segment combinations
      Calculate probabilities based on Equation $5.2$
  Set segments of sites according to probabilities
  Re$-$estimate distributions
```

Listing 5.1: Pseudocode for multinomial segmentation.

6 Logistic Regression

In this section we will examine logistic regression and how we can use it to define a model for observed variables that properly takes the configuration of which segments a site belongs to in each layer into account.

Logistic regression is a method used in statistics often used to model the probability of an event occurring given other data points. Logistic regression requires a data set consisting of different cases with all relevant data, e.g. for a heart disease study, the cases could contain data such as blood pressure, age, etc, as well as whether or not the patient died within 10 years of the other variables being collected.

Given this data we can select an attribute as being the dependent variable, and a subset of the remaining variables as independent variables. Using logistic regression we can use the data set and these categorisations to fit a set of parameters to a logistic regression model.

Inserting these parameters into the logit equation seen in Equation 6.1, where β_0 is the intercept, a parameter used to weight a logit where all variables are 0, x_i is the value of the variable i, and β_i is the parameter for the variable i.

$$logit = \beta_0 + x_1 \cdot \beta_1 + x_2 \cdot \beta_2 + ... + x_n \cdot \beta_n \tag{6.1}$$

Given the logit, we can define an equation, as seen in Equation 6.2, to find the probability of the dependent variable being present given the independent variables, e.g. in the heart disease study, if patient death was selected as the dependent variable, inserting blood pressure, age and all other variables for a patient, this equation models the probability of that patient dying within the next 10 years.

$$P = \frac{e^{logit}}{1 + e^{logit}} \tag{6.2}$$

Logistic regression is able to handle binary variables, e.g. gender. This is simply modelled as 0 for one gender, 1 for the other, and is otherwise handled as a numerical variable. Logistic regression is also capable of handling multinomial categorical variables, e.g. race or civil status. This is handled by splitting these up into a binary indicator variable for each category, and these are then handled as any binary variable. For an extension of the logit function to handle these types of data, see Equation 6.3, where $I_{[x_i=j]}$ is an indicator variable that is 1, when x_i is j, and 0 when it is not.

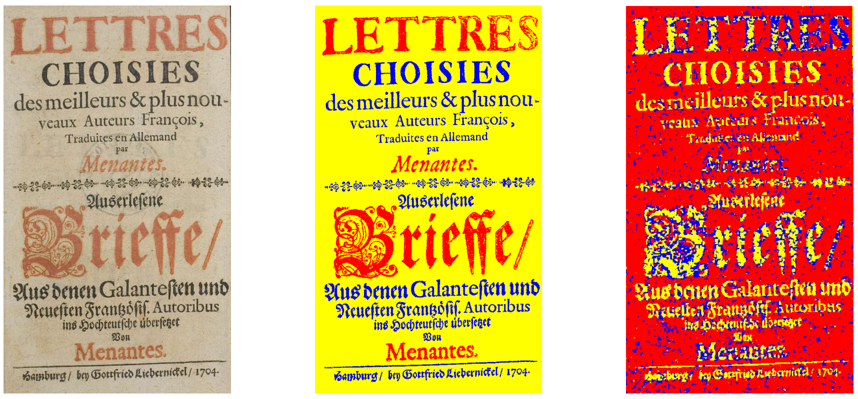
$$logit = \beta_0 + I_{[x_1=1]} \cdot \beta_{1,1} + I_{[x_1=2]} \cdot \beta_{1,2} + ... + I_{[x_n=m]} \cdot \beta_{n,m} \tag{6.3}$$

We will establish a function by estimating the parameters that describes the logistic regression for a plant, given the segment configuration of the site that the plant belongs to. The segments can be given as both numerical and categorical variables. Once the logistic regression model has been fitted, we can calculate the probability of a plant, given a segment configuration. This probability is then used in place of learned model as defined in Section 4.3.

7 Histogram Initialisation

In this chapter we will propose several initialisation methods based on histograms and selection heuristics. We will also illustrate the motivation for this. Additionally, the selection heuristics for each method will be explained along with the possible downsides to them.

A naive approach to initialisation is to set the configuration of segments for each site randomly. When using our clustering algorithm for images with an entirely random initial configuration, segmentations are likely to be unfocused and wrong. This is mainly due to the fact that the clustering algorithm yields a segmentation where the likelihood value has reached a local maximum, e.g. when attempting to cluster the poster in Figure 7.1(a). If we configure our clustering algorithm to use three segments, we expect that one segment will contain the background and the differently coloured text will be contained in a segment each, as illustrated in Figure 7.1(b). However, due to the way the computer interprets the colour data, the noise in the background and letters end up taking up the third segment, leaving only two segments for the two types of text and the background. This effect is shown in Figure 7.1(c).


LETTRES
CHOISIES
des meilleurs & plus nou-
veaux Auteurs François,
Traduites en Allemand
par
Menantes.
Auſerleſene
Brieffe/
Aus denen Galanteſten und
Neueſten Frantzöſiſ. Autoribus
ins Hochteutſche überſetzet
Von
Menantes.
Hamburg / bey Gottfried Liebernickel / 1704.

(a) The original Menantes poster

(b) The expected clustering of the Menantes poster

(c) The results using random initialisation

Figure 7.1: The Menantes poster problem illustrated

One solution to this is to make a heuristic for initialisation. For pictures, a natural choice is a histogram solution. A histogram is a way to roughly divide the pixels of an image into groups based on their colours. This is commonly achieved by using the RGB values of each pixel, using a 3-dimensional matrix, where each axis of the matrix corresponds to a colour, see Figure 7.2 for a visual representation of this.

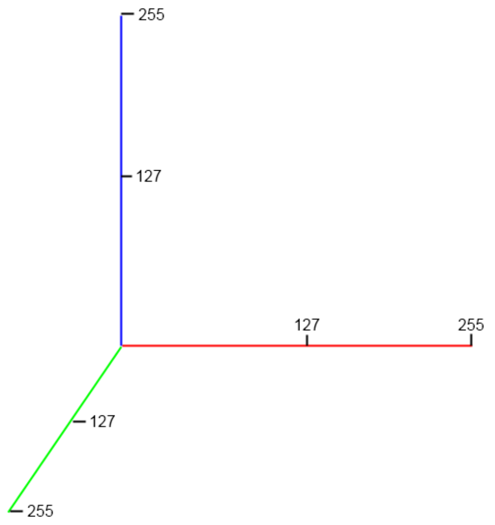


Figure 7.2: An illustration of how the histogram matrix corresponds to colours

Each axis is subdivided into a number of groups, for example for colour values for red of 0 to 255, these may be 0 to 127 and 128 to 255, as is the case in the visual representation. The amount of subdivisions along each axis is the granularity factor.

Each pixel is then placed in the cell that corresponds to the colour values it has. This is essentially a form of crude clustering, however there is no way to directly control how many segments are produced. The image will be divided into a number of groups dependent on the differences in colour. An image consisting entirely of shades of the same colour could potentially all end up in the same bucket, if the red axis is not divided into sufficiently small buckets. This problem is illustrated in Figure 7.3. For this image, two histograms have been created, with different bracket sizes. One has a granularity factor of 2. This leaves every pixel in the image in the same bucket, as shown in Figure 7.4 and Table 7.1. With a higher granularity factor of 4, the image is subdivided into two buckets, as shown in Figure 7.5 and Table 7.2. In both of the tables, average colour is represented as ([red], [green], [blue]), with each colour ranging from 0 to 255.

Assuming we can control the amount of divisions along each axis, we can assume that for all images with at least n distinct colours, we can find n distinct

Figure 7.3: An image where colour values are very close to each other.

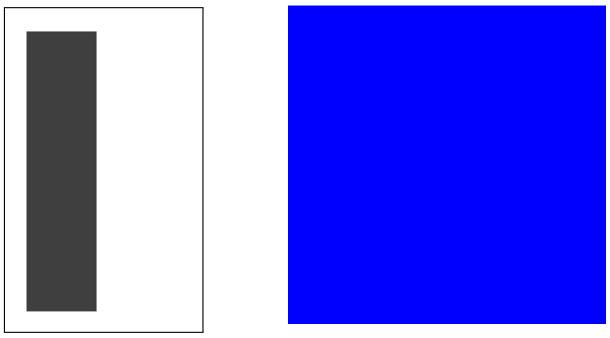


Figure 7.4: The chosen buckets are shown visually on the left for a granularity factor of 2. On the right, the segmentation corresponding to this initialisation is shown.

groups of pixels, where n is the number of segments we want. We can then use these pixels as a basis for the learned colour model for the gaussian distribution.

However, the problem now becomes that for most images, we will have more than n groups of pixels. This can be solved in a number of different ways, as we will detail in the next sections.

7.0.1 Base Histogram

The naive approach is to simply select the n highest buckets, however this can be a problem, if the two largest groups of pixels are similar in colour. An example of this is using the base histogram method on the Menantes poster shown in Figure 7.1(a). Here, a granularity factor of 3 means that the background is divided into two buckets, as shown in Figure 7.6 and Table 7.3.

Average colour	(R=63, G=63, B=63)	Not applicable
Pixel count	16384	0

Table 7.1: A histogram with 2 brackets of 128.

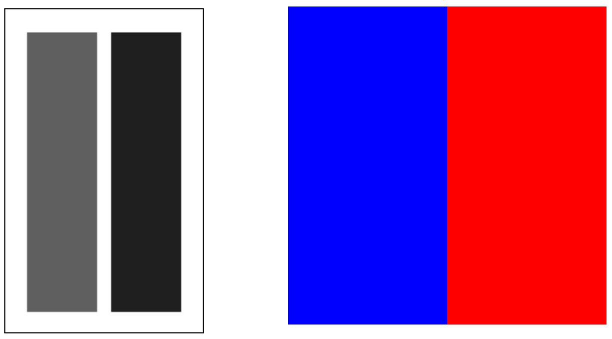


Figure 7.5: The chosen buckets are shown visually on the left for a granularity factor of 4. On the right, the segmentation corresponding to this initialisation is shown.

Average colour	(31, 31, 31)	(95, 95, 95)
Pixel count	8192	8192

Table 7.2: A histogram with 4 brackets of 64.

7.0.2 Distance Rating

One method to prevent picking similar groups is to rate buckets based on their difference from already selected buckets. The simple way of measuring the distance is to consider the histogram as a 3D grid. The distance between two buckets is then the Manhattan distance, $|\begin{bmatrix} c_r \\ c_g \\ c_b \end{bmatrix} - \begin{bmatrix} c_r \\ c_g \\ c_b \end{bmatrix}'|$, where c_r, c_g and c_b is the cell coordinates for the red, green and blue axes.

The first bucket is then chosen as normal, but each following bucket has a rated histogram calculated, for which each cell is recalculated using Equation 7.1, where c and c' is the value in a given cell and the adjusted value based on distance respectively, w is a weight to alter the importance of the distance and d is the distance to the nearest already chosen bucket. An alternative to this is Equation 7.2, where m is the highest of the d values for all buckets.

$$c' = c + w * d \tag{7.1}$$

$$c' = c * (d/m) \tag{7.2}$$

Average colour	(198, 184, 157)	(186, 150, 120)	(208, 197, 175)
Pixel count	421489	92482	78241

Table 7.3: The brackets shown in Figure 7.6.

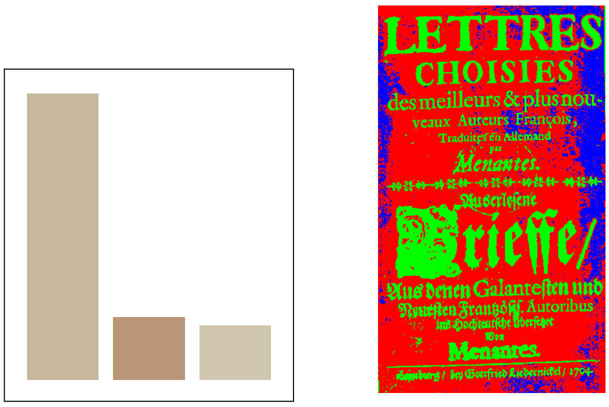


Figure 7.6: The base histogram initialisation is shown on the left. On the right, the segmentation corresponding to this initialisation is shown.

The benefit of this method is that for cases where the primary segments are expected to be close to each other in terms of colours, it will not make this impossible. However, for the first case, the weight w needs to be set carefully, or buckets with an original value of 0 may become the highest rated bucket. This is eliminated in the second case.

7.0.3 Distance Lockout

Another method is to simply not allow buckets within a minimum distance of any selected bucket. This makes it impossible for adjacent buckets to be chosen, eliminating the problem of slight colour differences taking up two separate segments.

The downside to this method is that in images where the buckets are grouped together, one or more segments may contain no pixels. Looking back to Figure 7.5 and Table 7.2, whichever bucket is chosen first, if the lockout distance is even 1, this heuristic would force the system to set the second segment to a bucket containing no pixels.

8 Likelihood Value

In this chapter we will propose a measure that describes the likelihood of a given segmentation. The motivation for this is that we need to estimate the quality of a segmentation.

A given segmentation, created using an MRF clustering model, has a likelihood value, given by the Gibbs distribution for the MRF, as described in Section 4.1. This equation, however, has some problems before it is useful for comparison of segmentations.

The biggest issue is the Z variable, seen in Equation 8.1. This is the sum of the likelihood values over all configurations of segment IDs, which makes it exponential with regards to both the sites, layers and segments. This makes it computationally unfeasible. Additionally, since we base this variable on the currently learned logistic regression model, we cannot simply eliminate it, since for two compared values, the Z variable will not be the same.

$$\begin{aligned} Z &= \sum_{f \in \mathbb{F}} e^{U(f)} \\ U(f) &= \sum_i \sum_j V_c(x_i, x_j) + \sum_i \sum_k V_c(x_i, y_{i,k}) \end{aligned} \tag{8.1}$$

However, given that our observation probabilities are not potentials, we needed to divorce them from the Gibbs distribution, as shown in Equation 8.2.

$$P(f) = P(X, Y) = P(X)P(Y|X) \tag{8.2}$$

Since $P(X)$ is the Gibbs distribution for an MRF ignoring the observed variables, if we set β to 0, this becomes a constant, and can thus be eliminated. This also eliminates the Z variable, leaving us with $P(Y|X)$, which is our observation probability. $P(X, Y)$ given a β of 0, is then the sum of the observation potential over all sites and plants. To prevent overflows, we take the natural logarithm of this, leaving us with Equation 8.3. This value is proportional with the likelihood value, given a β of 0, and can thus be used for comparisons.

$$ln(P(f)) \approx \sum_i \sum_k ln(V_c(x_i, y_{i,k})) \tag{8.3}$$

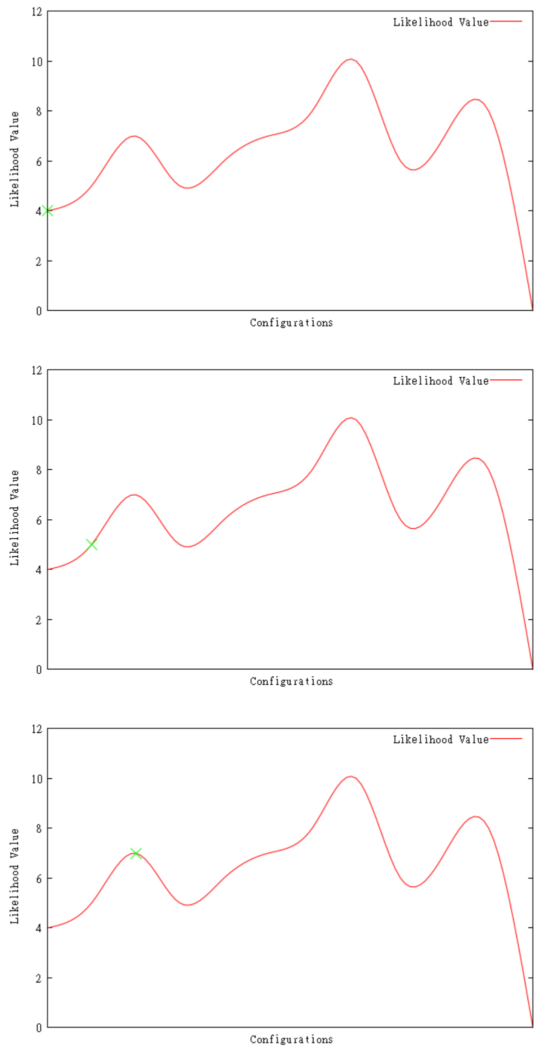


Figure 8.1: A representation of likelihood values over different segment configurations, and how the algorithm moves from configuration to configuration towards a local maximum.

When clustering, we expect this value to go up for each iteration, signifying that the segmentation is getting better for each iteration. As we progress, the changes in the value will become smaller, and finally the segmentation will reach a local maximum. This behaviour is illustrated in Figure 8.1. Note that the X-axis here represents different configurations of segments over all sites.

The likelihood value is the strength of modelling observation potentials with a logistic regression model. For the multinomial model, we expect this value to form a number of maxima of the same value, making it impossible to distinguish which segmentations are better, as shown in Figure 8.2.

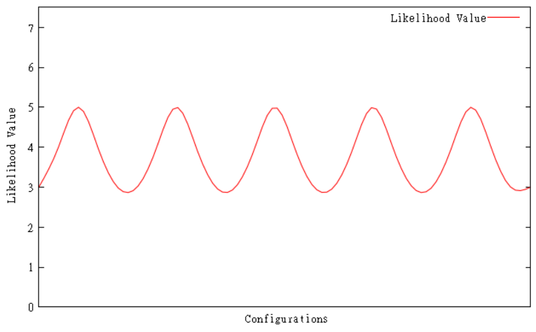


Figure 8.2: A representation of likelihood values over different segment configurations for a multinomial clustering model.

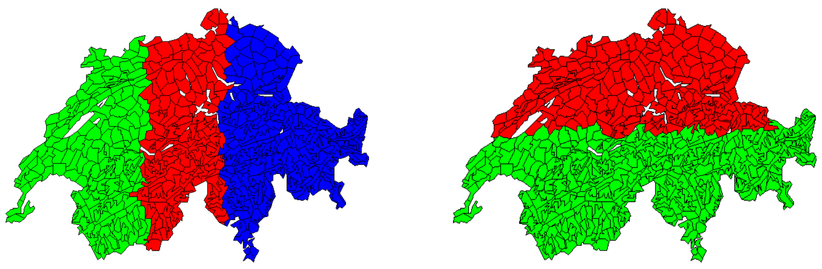


Figure 8.3: A segmentation showing three vertical bars in one layer, and two horizontal bars in the other.

This is due to how the multinomial model models the plants. If a given plant is in all of the sites with a given segment combination, this plant contributes a value of 1 for each site. That means that for an intended segmentation as seen in Figure 8.3, any segmentation that, for each layer, splits the sites up in such a way that the combined segmentation, as seen in Figure 8.4, is still modelled, will have the same likelihood value, e.g. the segmentation seen in Figure 8.5 has the same likelihood value as the intended segmentation.

For the logistic regression model, we do not expect this behaviour to be present, instead having one global maximum. We still expect there to be a number of local maxima, however, it is possible to distinguish a good segmen-

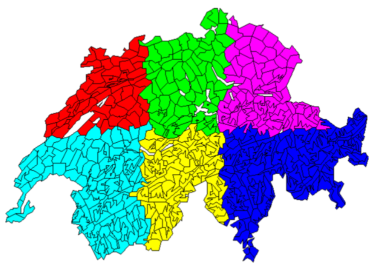


Figure 8.4: The combined segmentation of the two layers of Figure 8.3.

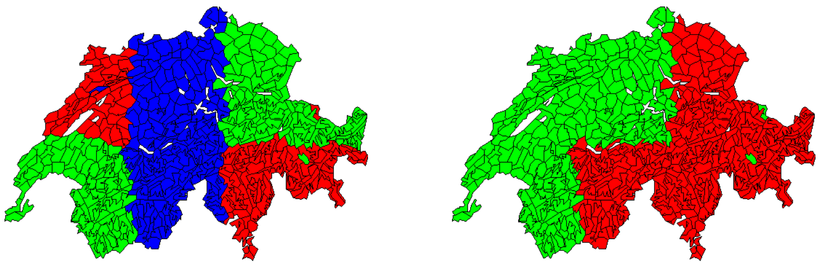


Figure 8.5: An example of a wrong multinomial segmentation. Note that the combined segmentation of the layers is still that of Figure 8.4.

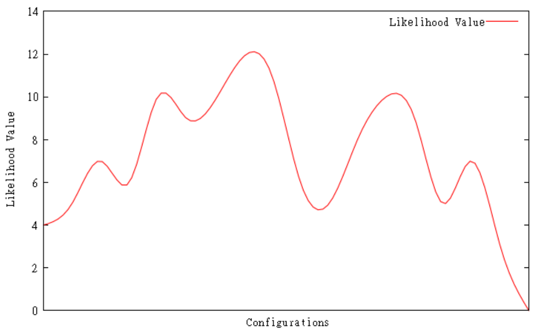


Figure 8.6: A representation of likelihood values over different segment configurations for a logistic clustering model.

tation from a bad one using the value. An example of the values we expect are shown in Figure 8.6.

Part II

Implementation

9 Architecture

The system is designed as a class library, with one main class, `Data`, containing all other elements of the clustering. This is then inherited by two specialised classes, `ContinuousData` and `DiscreteData`, for images and plant data respectively, as seen in Figure 9.1.

The individual sites are contained in instances of the `Site` class in a list in the `Data` class instance, each of which contain the observed variables for that site.

Additionally, the neighbourhood potential and the learned model for the observed variables are contained in a list of instances of specialised classes inheriting from the `Potential` class in the `Data` instance, one for the neighbourhood and one for each observed variable.

9.1 Data

The `Data` class is both the container for the data necessary to perform a segmentation, and the starting point for the clustering method. Specialised types are `ContinuousData` and `DiscreteData`. This specialisation is due to the unique conditions for the two data types. For continuous data, multiple layer segmentation is not of much interest within the scope of this project. Additionally, for continuous data, the initialisation is handled by histogram methods. For discrete data, categorical logistic regression is available. Additional information for the two variable types is contained in `continousVariables` and `discreteVariables`.

Cluster is used by the program using the library. It performs a clustering on the given filename, using the settings given. This method simply initialises the `ClusterSingle` method a number of times, choosing the result with the highest likelihood value.

ClusterSingle performs a single clustering using the given file and settings.

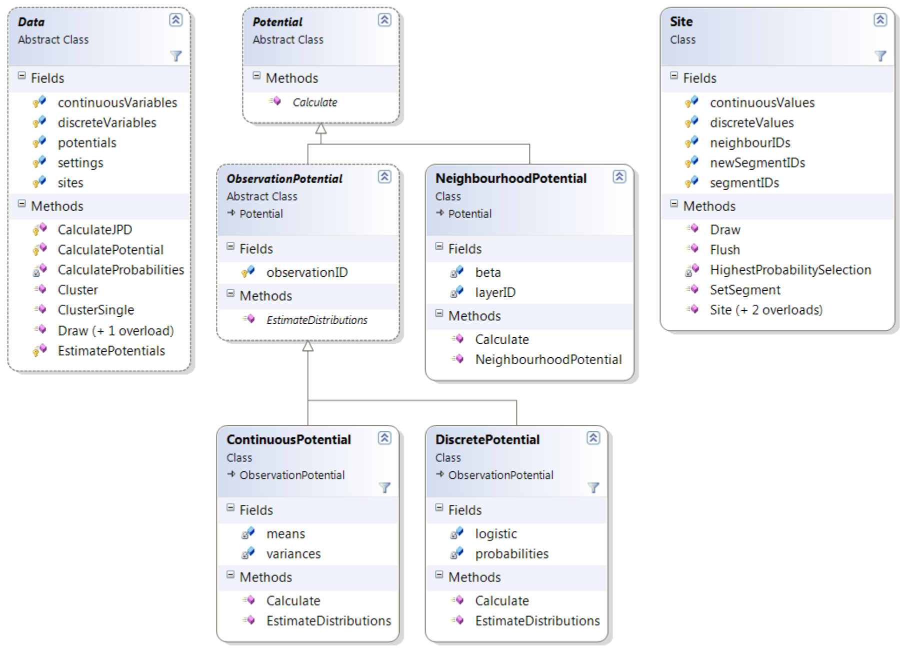


Figure 9.1: Class diagram of our clustering solution.

9.2 Site

The `Site` class contains all information about a single site. Observed variables for each site are contained in a list indexed by the ID of the observed variable, in `continuousValues` and `discreteValues` respectively.

The `Site` class has two constructors, one with an XML node as a parameter, basing the construction of the instance on the MRF data format, and one taking a pixel as a parameter, for use with image data.

9.3 Potential

The `Potential` class contains the information necessary to calculate the neighbourhood potential or the probability for an observed value. It is specialised into two classes, `NeighbourhoodPotential` and `ObservedPotential`, the latter of which is specialised further into the `DiscretePotential` and `ContinuousPotential` classes. These specialised classes simply handle the appropriate types of data.

Calculate returns the value of the given potential or probability based on the given site and segment IDs.

EstimateDistributions learns the model of the observed values.

10 SPSS

We use the IBM SPSS Statistics Programmability Extension to access the backend of SPSS in our code. The extension contains a class library that we import into our project. This allows us to use mediator component that contains functionality to submit SPSS syntax to the backend. This chapter will describe how we have implemented SPSS and any troubles we have had.

10.1 Implementation

When designing our statistics module, we want to keep modularity and flexibility in mind. We have decided to create an abstract class, `StatisticsBackend`, describing a generic backend for a statistics module which can be seen in Figure 10.1. Our SPSS module and any future statistics modules will implement this interface. Below is a short description of each method in the `SPSSBackend` class which implements the `StatisticsBackend` abstract class.

CalculateProbabilities will calculate the probabilities of each plant being present in a site given a segment configuration, for all sites. Results will be stored for later reference.

GetIntercept will retrieve the intercept of a plant, p.

GetProbability will retrieve the probability of a plant, p, being present in a site s, given a specific configuration of segments in that site, i.e. the probability $P(p|L_0 = 2, L_1 = 3)$.

GetWeights will return all parameter weights for the logistic regression model of a plant, p.

SPSSBackend initialises the internal components of the module.

Start will start the SPSS backend and set the desired encoding of files.

Stop will stop the SPSS backend.

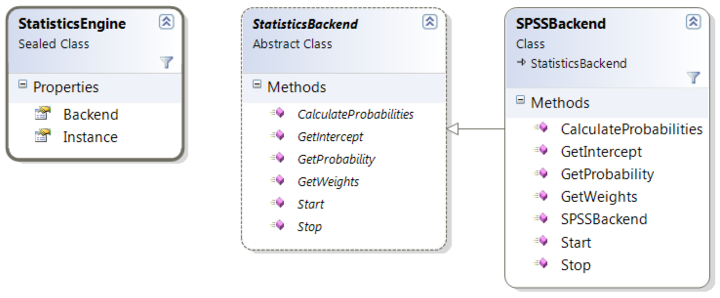


Figure 10.1: A class diagram of the SPSS statistics backend.

10.2 Submitting Data to SPSS

It is possible to submit data to SPSS in a variety of formats. Among these are Excel spreadsheets, various formats of raw text and SPSS SAV data set files. We will use tab delimited text files as these are the easiest to read and write. Before we calculate the probabilities in each iteration, we write a tab delimited text file containing all the necessary information for SPSS. The tabulator character will be used to separate columns. The format of the text file can be seen in Listing 10.1, where \t indicates the tabulator character and m and n are the number of layers and plants, respectively. Each line in the text file represents a site in the dataset. A site's identifier is represented by the ID column and is only included to simplify debugging as it is not needed for the actual logistic regression. The L1 to Lm columns represents the current segment that each layer in the site is currently assigned to. The P1 to Pn columns are categorical values that represent the presence of plants, where 1 means present and 0 means absent. An example of a format containing sites with ID 101 through 105, two layers and five plants can be seen in Listing 10.2.

```
[ID]\t[L1]\t ...\ t[Lm]\t[P1]\t ...\ t[Pn]
```

Listing 10.1: Format of the tab delimited text file.

```
ID\tL1\tL2\tP1\tP2\tP3\tP4\tP5
101\t2\t3\t1\t0\t1\t0
102\t3\t1\t0\t1\t0\t0
103\t2\t1\t1\t0\t0\t0
104\t3\t2\t1\t0\t1\t1
105\t3\t1\t1\t0\t0\t1
```

Listing 10.2: An example tab delimited text file.

Once we have created our data file, we need to instruct SPSS to load it as a dataset. In Listing 10.3 the syntax we use to load a tab delimited text file into SPSS. `FILENAME` indicates the desired file name to load and `VARIABLE`

DEFINITION indicates a defitinion of the variables expected to be found in the specified file. The variable definition defines the names of the columns and the order in which they occur. After each variable name, we declare the maximum number of digits that can occur in the dataset before and after a decimal point. A variable definition of the variables from the tab delimited text file example from Listing 10.1 can be seen in Listing 10.4.

```
DATA LIST FREE FILE='[FILENAME]' /[VARIABLE DEFINITION].
```

Listing 10.3: SPSS Load Syntax

```
    /ID (F5.0) L1 (F1.0) ... Lm (F1.0) P1 (F1.0) ... Pn (F1.0).
```

Listing 10.4: An example variable definition.

10.3 Executing the Logistic Regression

Once data has been loaded into SPSS, we want to fit each plant to a logistic regression model. In order to extract results from SPSS when a calculation has been completed, we need to use the Output Management System (OMS) of SPSS. As when loading data into SPSS, we have several formats for outputting. We will again choose tab delimited text files as they are straight-forward to parse. The syntax for activating OMS and instructing SPSS to direct the results of any subsequent commands to a text file can be seen in Listing 10.5, where FILENAME indicates the output destination and SUBSEQUENT COMMANDS indicates any result producing commands such as logistic regression. One thing to note is that we instruct SPSS to only include the Parameter Estimates result tables, which contain the only information we need.

```
OMS /SELECT TABLES
    /IF SUBTYPES=['Parameter Estimates']
    /TAG = 'logistic'
    /DESTINATION FORMAT = TABTEXT OUTFILE = '[FILENAME]'.
[SUBSEQUENT COMMANDS]
OMSEND TAG = ['logistic'].
```

Listing 10.5: Syntax for activating the OMS.

Now, for each plant we will instruct SPSS to calculate parameter estimates that fit a multinomial logistic regression model. The syntax for this can be seen in Listing 10.6, where PLANT ID is the plant we currently want to estimate parameters for and LAYERS are the names of the layer columns in the dataset.

An example of an output file can be seen in Listing 10.7.

```
NOMREG [PLANT ID] (BASE=LAST ORDER=ASCENDING) BY [LAYERS]
    /CRITERIA CIN(95) DELTA(0) MXITER(100) MXSTEP(5) CHKSEP(20) LCONVERGE(0)
        PCONVERGE(0.000001) SINGULAR(0.00000001)
    /MODEL
    /STEPWISE=PIN(.05) POUT(0.1) MINEFFECT(0) RULE(SINGLE) ENTRYMETHOD(LR)
        REMOVALMETHOD(LR)
    /INTERCEPT=INCLUDE
    /PRINT=PARAMETER SUMMARY LRT CPS STEP MFI.
```

Listing 10.6: Syntax for a multinomial logistic regression.

```
Parameter Estimates
      B   Std. Error  Wald   df  Sig.    Exp(B)  95% Confidence Interval
           for Exp(B)
P174(a)                            Lower Bound Upper Bound
0   Intercept  -5,032  ,661     58,003  1   ,000
    [L0=0]  6,848   ,691     98,240  1   ,000     942,350 243,265 3650,438
    [L0=1]  ,569    ,759     ,562    1   ,453     1,767   ,399    7,823
    [L0=2]  0(b)    .   .    0   .   .   .   .
    [L1=0]  1,778   ,588     9,150   1   ,002     5,921   1,870   18,744
    [L1=1]  0(b)    .   .    0   .   .   .   .
a   The reference category is: 1.
b   This parameter is set to zero because it is redundant.
```

Listing 10.7: An output file showing an example Parameter Estimates table.

Once results have been written to a text file, we load them into our system. We do this by simply opening the text and parsing the results with regular expressions.

10.4 Retrieving Warnings and Errors

We want to be aware any warnings or errors that occur in SPSS while executing our commands. For this we can use the OMS in a similar manner as in Section 10.3, except instead of selecting tables, we will be selecting logs and warnings as seen in Listing 10.8. We will execute this command as soon as we have initialised SPSS, so that any subsequent commands can be captured by the OMS.

```
OMS SELECT LOGS WARNINGS
    /TAG = 'errorlog'
    /DESTINATION FORMAT=TEXT OUTFILE='[FILENAME]'.
[SUBSEQUENT COMMANDS]
OMSEND TAG = ['errorlog'].
```

Listing 10.8: Syntax for activating the OMS.

Since we are using the .NET plugin instead of executing commands directly in SPSS, we found that there is a problem with the above approach. When submitting commands to SPSS from our application, any error that occurs results in an exception being thrown with error code 3 (meaning fatal error). No additional information is supplied in the thrown exception and it can thus be troublesome to debug the exact cause of an error. Since normally OMS is able to capture and report errors, we need to find a way to retrieve detailed error descriptions. When our application is executed in Visual Studio 2008, the hosting process feature of Visual Studio enables us to catch exceptions at run time and allows us to inspect the state of objects at the time the exception was thrown. When SPSS throws an exception due to an error occurring somewhere in our syntax, executions of our application is paused and we are able to inspect the the `Processor` class, that mediates all communication with SPSS. We have found that it has a static, non-public `FileStream` object, `resultFile`, where any communication with SPSS is written. By examining the `Name` accessor of `resultFile`, we can determine the location and name of that file, which is a nonsensically named file in the temporary files folder of the current user. This file contains all output produced by SPSS in an unfiltered format, including verbose error descriptions of any errors encountered while executing syntax.

10.5 Performance Issues

We observed a performance issue when computing on the real world data set, where the first iteration would complete within a matter of minutes and the computation time for each subsequent iteration would increase significantly, taking over a day within a few iterations. At first we wondered if it could be caused by previous data remaining in memory and somehow affecting the calculations on the new data. However, examining the memory footprint of SPSS indicated that previous data was released as each iteration ended. We then considered, if it could be a problem with our code, but profiling the code showed that over 99 % of CPU time was spent waiting for the SPSS process. Examining the performance of the code, we noticed that SPSS was able to fit a logistic regression model for several plants within a second but this gradually slowed down until at around 75 plants, each plant took a second to calculate. We experimented with restarting the SPSS mediator component and we noticed that the restart resulted in an immediate increase in performance. We were not able to determine the exact reason for the performance decrease but we were able to circumvent the problem by restarting the mediator process after a fixed amount of plants had been processed.

11 Histogram Initialisation

As described in Chapter 7, random initialisation often produced unsatisfactory results in image segmentation. To combat this, a histogram heuristic was implemented. The implementation of this is very simple, as shown in Listing 11.1. The pseudocode indicates that multiple histogram methods can be used simultaneously. This is not the case, though this is merely a constraint applied by us. The methods available are the base histogram method, additive distance rating, multiplicative distance rating and distance lockout, described in Chapter 7.

```
Create histogram as a three-dimensional array of $(b)$x$(b)$x$(b)$, where $b$ is the
    granularity factor
  For each site
    Find the colour bracket it fits into for each color
    Add 1 to histogram cell with coordinates matching the three brackets

Select $n$ best buckets, where $n$ is the wanted number of segments
  For $i = 1$ to $n$
    If using Additive Distance Ranked Histogram
      Rank buckets using Equation 7.1
        For each bucket
          Find distance, $d$, from nearest previously chosen bucket (or 0, if no
              bucket has been chosen)
          Find adjusted cell value using Equation 7.1
    If using Multiplicative Distance Ranked Histogram
      Rank buckets using Equation 7.2
        Find maximum value of $d$, $m$, over all buckets
        For each bucket
          Find distance, $d$, from nearest previously chosen bucket (or 0, if no
              bucket has been chosen)
          Find adjusted cell value using Equation 7.2


    If using Distance Lockout Histogram
      Select highest bucket a minimum of $l$ away from any chosen bucket, where
          $l$ is the lockout distance
    Else
      Select highest bucket

For each site
  Find the colour bracket it fits into for each color
  If the coordinates of the three brackets match chosen bucket, set to
      segment based on this

Cluster as normal, ignoring any pixels which has not been set to a segment
```

Listing 11.1: Pseudocode for histogram initialisation.

Part III

Experimentation

12 Histogram Initialisation

This chapter describes experiments performed with the histogram initialisation methods. They will be separated into synthetic test cases, showcasing the shortcomings of each different method, and real world image tests, proving the effectiveness of the methods on real images.

Our motivation for performing these tests is based on results when attempting to cluster real world images using random initialisation, e.g. those in Section 12.2. In these pictures, the background took up a far higher amount of pixels than either type of text. This meant that for all segments, the colour model was heavily skewed towards the background. Due to the colour of the background being heterogeneous, this meant that all segments had slightly different versions of the background. Therefore, the text, despite intuitively being the most important, did not affect the segmentation as much as desired. In order to remedy this, we have devised various initialisations based on histograms, which should be able to mitigate these issues.

In all these tests, initialisations using histogram methods will be described using a bar graph of the chosen buckets, showing the relative amount of pixels, as well as the average colour of the pixels. Additionally, a table is constructed, showing the average colour of each bucket in the format ([red], [green], [blue]), with each colour between 0 and 255, and the total amount of pixels in each bucket.

12.1 Synthetic Images

These test cases use constructed images to test the histogram initialisation methods, to prove that they work as intended, as well as showcasing how each method is not always ideal for different scenarios.

12.1.1 Gradient Image Tests

For the first set of tests, an image consisting of a gradient from a colour value of (0, 0, 0) to (127, 127, 127), seen in Figure 12.1.

Figure 12.1: Gradient spanning the RGB values (128, 128, 128) to (255, 255, 255)

Base Histogram, Granularity Factor 2,4,8

This test showcases the problems associated with selecting different granularity factors, GF. We will be using granularity factors of 2, 4 and 8. With $GF = 2$, the axes will be separated into 0-127 and 128-255. With the gradient ranging from 128 to 255, we expect all pixels will end up in the same bucket. For a $GF = 4$, two buckets will be available in the range 128 to 255 and we expect that pixels will be distributed evenly among the buckets. Lastly, for $GF = 8$ we expect the results to be similar to those of $GF = 4$, but since there are more valid buckets, we expect the algorithm will run more iterations before reaching a final result.

Results As expected, for $GF = 2$, the image initialised with all pixels in the first segment, and all other segments were eliminated, as shown in Figure 12.2 and Table 12.1. For $GF = 4$, we observed an even split of the picture into two segmentations, as shown in Figure 12.4 and Table 12.2. Additionally, over multiple runs, the choice of which bucket is selected first was random. For $GF = 8$, the histogram showed the image separated into 4 buckets, 2 of which were chosen at random. Over multiple runs, 6 distinct initialisations became apparent. An assortment of these are shown in Figure 12.6 and Table 12.3. Full segmentations have been run, and all resulted in the same segmentation as in the previous test, though it took more iterations of segmentation to get to the final result for these initialisations.

In summary, these tests have highlighted the various problems that can arise when using different granularity factors. If the granularity factor is set too low, such as $GF = 2$ in our test, then the algorithm tends to place most pixels into the same bucket. This is easily solved by increasing the granularity factor, as seen in our tests with $GF = 4$ and $GF = 8$. However, as $GF = 8$ highlights, choosing a large granularity factor can introduce quirks as well. In our case, the consequences of this were not severe, as it merely took more iterations of segmentation to get to the final result.

Average colour	(63, 63, 63)	Not applicable
Pixel count	16384	0

Table 12.1: Base histogram for the gradient image with a granularity factor of 2.

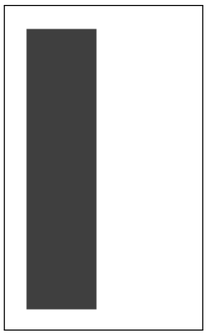


Figure 12.2: Base histogram for the gradient image with a granularity factor of 2.

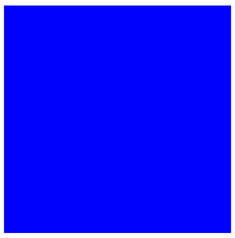


Figure 12.3: Result of the segmentation.

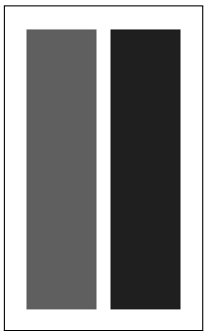


Figure 12.4: Base histogram for the gradient image with a granularity factor of 4.

Average colour	(31, 31, 31)	(95, 95, 95)
Pixel count	8192	8192

Table 12.2: Base histogram for the gradient image with a granularity factor of 4.

12.1.2 Colour Blocks Image Tests

The second set of tests is performed with an image composed of three blocks of colours, of which two are similar in terms of RGB values, and the last one is very different, seen in Figure 12.7.

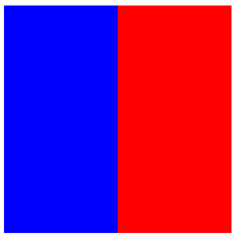


Figure 12.5: Result of the segmentation.

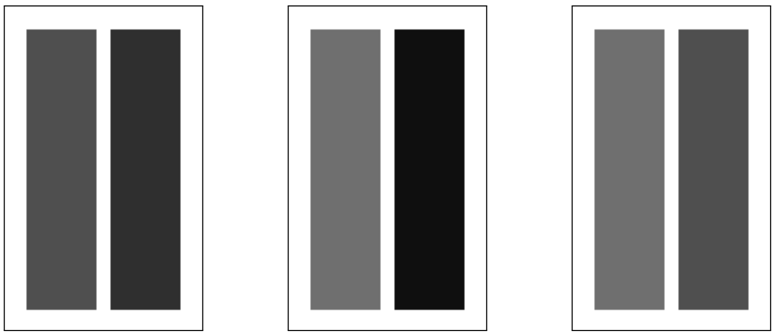


Figure 12.6: Base histograms for the gradient image with a granularity factor of 8.

Average colour	(47, 47, 47)	(79, 79, 79)
Pixel count	4096	4096
Average colour	(111, 111, 111)	(15, 15, 15)
Pixel count	4096	4096
Average colour	(111, 111, 111)	(79, 79, 79)
Pixel count	4096	4096

Table 12.3: Base histograms for the gradient image with a granularity factor of 8.

Base Histogram, Granularity Factor 4

This method shows the issue with the base histogram. The image used is composed of three blocks of colour, two similar to each other, purple and blue, and one very different, yellow. However, when simply selecting the biggest buckets, the purple and blue blocks should be chosen, leading to a worse segmentation.

Results As expected, the initialisation chose the two biggest blocks. The initialisation and resulting segmentation is shown in Figure 12.8. The final segmentation does choose the yellow block, and segments the blue and purple blocks together. This takes 9 iterations.

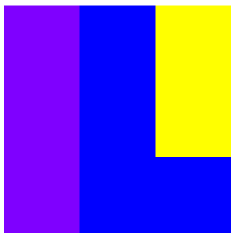


Figure 12.7: An image composed of blocks of colour. RGB values are, from left to right, (127, 0, 255), (0, 0, 255) and (255, 255, 0).

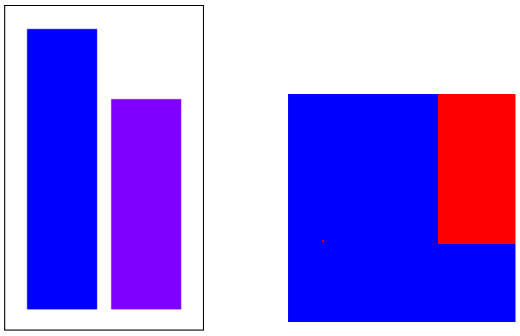


Figure 12.8: The result of the base histogram test of the colour blocks image. The initialisation bar graph is shown to the left, the resulting segmentation to the right.

Additive Distance Ranked Histogram, Distance Weight 1600, Granularity Factor 4

This test shows how weighting the distance makes the more different colour more important, resulting in a better segmentation.

Results As expected, the yellow colour block is chosen, as shown in Figure 12.9. While the final segmentation is the same, with this method, it is achieved in a single iteration.

Multiplicative Distance Ranked Histogram, Granularity Factor 4

This test is expected to behave similar to the test above. The blue block will still be chosen first. The bucket containing the purple pixels will then in the rated histogram have a ninth of its original value, while the bucket containing the yellow pixels will have the same value as before, resulting in the yellow bucket becoming much higher.

Average colour	(0, 0, 255)	(127, 0, 255)
Pixel count	6400	4800

Table 12.4: Base histogram for the colour blocks image with a granularity factor of 4.

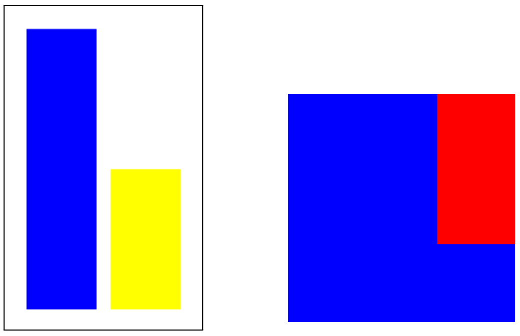


Figure 12.9: The result of the additive distance ranked histogram test on the colour blocks images. The initialisation bar graph is shown on the left, and the resulting segmentation on the right.

Average colour	(0, 0, 255)	(255, 255, 0)
Pixel count	6400	3200

Table 12.5: Additive distance ranked histogram for the colour blocks image with a granularity factor of 4 and a distance weight of 1600.

Results As expected, the segmentation is identical to that of the additive distance ranked histogram test.

Distance Lockout Histogram, Lockout Distance 3, Granularity Factor 4

As the previous test, this test should yield the same result as the additive distance ranked histogram test. With a lockout distance of 3, the bucket containing the purple pixels should not be possible to choose, resulting in the algorithm choosing the bucket with the yellow pixels.

Results As expected, the segmentation is identical to that of the additive distance ranked histogram test.

12.1.3 Conclusion

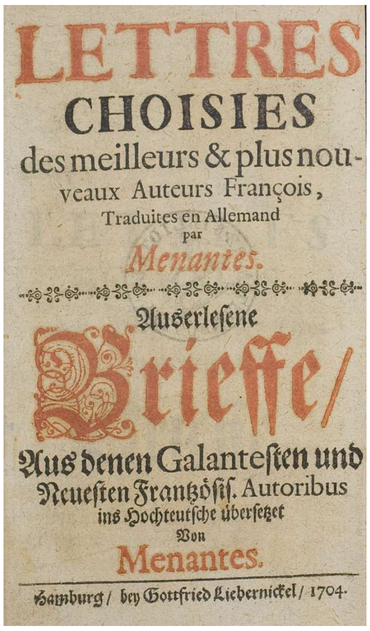
We have verified that in our synthetic data experiment, our histogram algorithms worked as expected. We have also illustrated the pitfalls of using an inappropriate method or parameters on an image.

12.2 Real World Images

These tests show how the different methods handle real world images with different settings. The intention is to show that each method can perform better or worse than the other methods, given the right image.

12.2.1 Menantes Poster

For the first set of tests, the image used is the Menantes poster, shown in Figure 12.10.



LETTRES
CHOISIES
des meilleurs & plus nou-
veaux Auteurs François,
Traduites en Allemand
par
Menantes.

Auserleſene
Brieffe/
Aus denen Galanteſten und
Neueſten Frantzöſiſ. Autoribus
ins Hochteutſche überſetzet
Von
Menantes.

Hamburg / bey Gottfried Liebernickel / 1704.

Figure 12.10: Menantes poster, an aged poster with faded text.

Random Initialisation

This is included as a baseline test to compare the histogram initialisation methods against.

Results The initialisation bears no resemblance to the poster. The final segmentation has not identified the two text types, instead separating the background into two segments. See Figure 12.11 for the results.

Base Histogram

This is used to show the effectiveness of the histogram test as opposed to the random initialisation test, as well as to provide a baseline for the other histogram methods.

Parameters Runs will be performed with a granularity factor of 2, 3 and 4.

Results The results are shown in Figure 12.12. Surprisingly, for $GF = 2$, the initialisation is nearly perfect. With $GF = 3$, the background is separated into two segments, and with $GF = 4$, parts of the background takes up all three segments. In both of these cases, the resulting segmentation adopts this separation of the background, leading to a worse result.

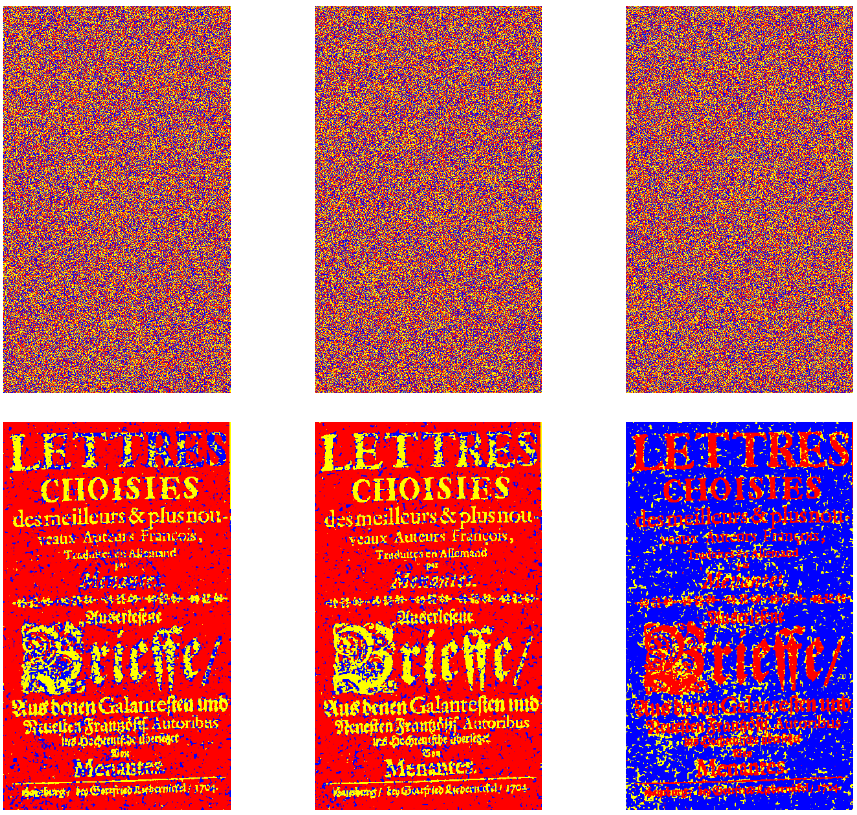


Figure 12.11: Different results of the Menantes random initialisation test. The top row shows each initialisation, the bottom row shows the respective final result after segmentation.

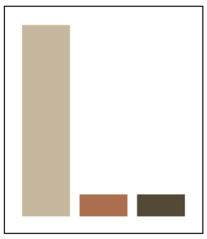
Average colour	(198, 184, 158)	(171, 111, 80)	(83, 73, 55)
Pixel count	548466	62910	60915
Average colour	(198, 184, 157)	(186, 150, 120)	(208, 197, 175)
Pixel count	421489	92482	78241
Average colour	(199, 183, 157)	(208, 196, 172)	(184, 171, 144)
Pixel count	323799	120918	103376

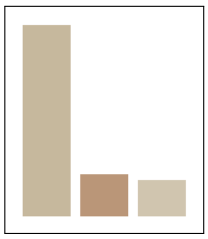
Table 12.6: Different results of the Menantes base histogram test.

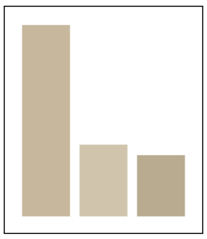
Additive Distance Ranked Histogram

This test demonstrates the additive distance ranked histogram using different weights.

Parameters The granularity factor will be set to 3 and 4. $GF = 2$ is excluded due to the precision of the base histogram method, and as such, the heuristics hold little interest for that case. Distance weights will be 10000 and 15000 for


(a) $GF = 2$


(b) $GF = 3$


(c) $GF = 4$

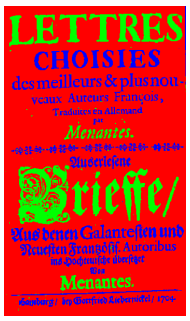
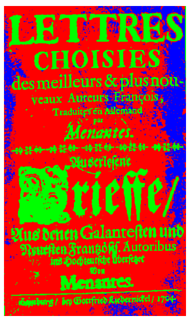
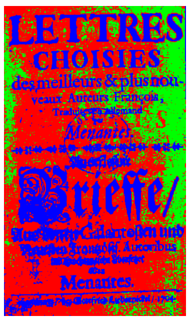




Figure 12.12: Different results of the Menantes base histogram test. The top row shows each initialisation, the bottom row shows the respective final result after segmentation.

$GF = 3$, and 15000 and 20000 for $GF = 4$. These numbers were chosen as representative based on a wider selection.

Results In both cases, only the results on either side of the distance ranking taking effect are shown. Before the switch, both initialisations were identical to the base histogram samples. After the switch, the initialisation for $GF = 3$ and a weight of 15000 separated the poster into the two colours of text and the background, as intended. This is shown in Figure 12.13 and Table 12.7. For $GF = 4$, no such segmentation was found, though one of the segments previously assigned to a part of the background is assigned to the text. This is shown in Figure 12.14 and Table 12.8.

Average colour	(198, 184, 157)	(186, 150, 120)	(208, 197, 175)
Pixel count	421489	92482	78241
Average colour	(198, 184, 157)	(70, 61, 44)	(186, 150, 120)
Pixel count	421489	36623	92482

Table 12.7: Different results of the Menantes additive distance ranked histogram test for $GF = 3$. The initialisations for distance weights 10000 and 15000 are the top and bottom, respectively.

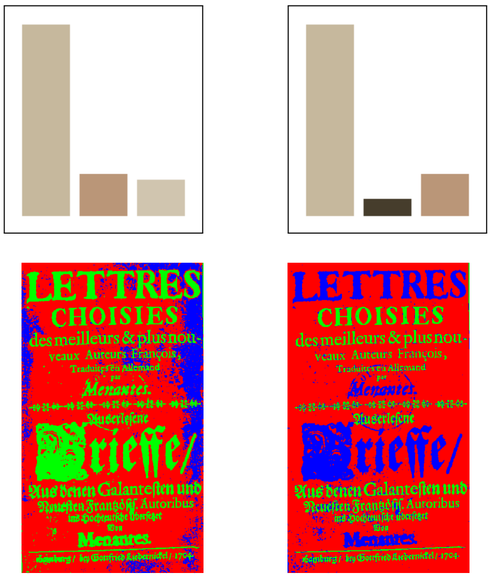


Figure 12.13: Different results of the Menantes additive distance ranked histogram test for $GF = 3$. The top row shows the initialisations for distance weights 10000 and 15000 from left to right, the bottom row shows the respective final result after segmentation.

Average colour	(199, 183, 157)	(208, 196, 172)	(184, 171, 144)
Pixel count	323799	120918	103376

Average colour	(198, 184, 157)	(56, 48, 31)	(208, 196, 172)
Pixel count	323799	36623	120918

Table 12.8: Different results of the Menantes additive distance ranked histogram test for $GF = 4$. The initialisations for distance weights 15000 and 20000 are the top and bottom, respectively.

Multiplicative Distance Ranked Histogram

This test shows the results of the multiplicative distance ranked histogram on the Menantes poster. As this method was developed for this image, this is expected to result in an equal or better segmentation than the additive distance ranked histogram.

Parameters The granularity factor will be set to 3 and 4.

Results For $GF = 3$, the result was identical to the additive distance ranked histogram initialisation for the same granularity factor and a weight of 15000. For $GF = 4$, the heuristic chose to assign the second segment to the red text instead of the black. These results are shown in Figure 12.15 and Table 12.9, along with the resulting segmentations.

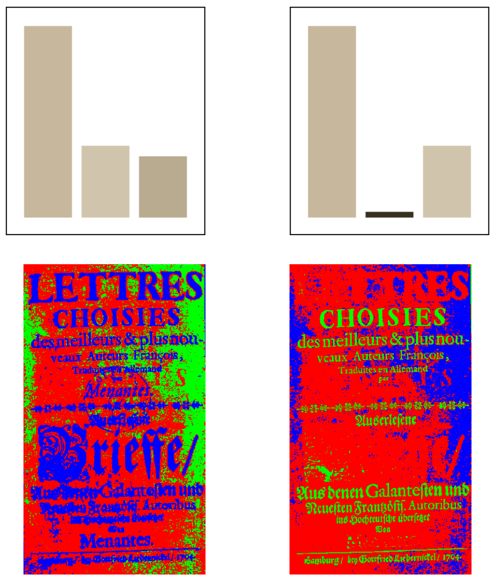


Figure 12.14: Different results of the Menantes additive distance ranked histogram test for $GF = 4$. The top row shows the initialisations for distance weights 15000 and 20000 from left to right, the bottom row shows the respective final result after segmentation.

For $GF = 3$, the segmentation was identical to that of the additive distance ranked histogram test. For $GF = 4$, due to the inferior initialisation, the resulting segmentation separated the background into two segments.

Average colour	(198, 184, 157)	(70, 61, 44)	(186, 150, 120)
Pixel count	421489	36623	92482

Average colour	(198, 184, 157)	(171, 112, 81)	(208, 196, 172)
Pixel count	323799	53856	120918

Table 12.9: Different results of the Menantes multiplicative distance ranked histogram test. The initialisations for $GF = 3$ and $GF = 4$ are the top and bottom, respectively.

Distance Lockout Histogram

This shows the distance lockout histogram method on the Menantes poster. This method is intended for use when two colours are very fairly close, and so abundant in the image as to overshadow all other colours. The Menantes poster does fit this somewhat, due to the background being split for most granularity factors.

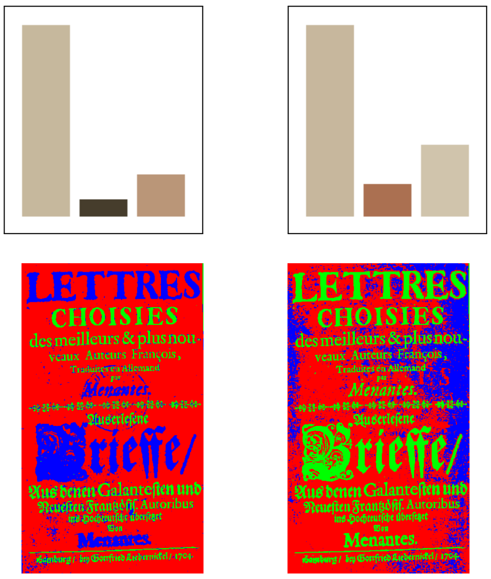


Figure 12.15: Different results of the Menantes multiplicative distance ranked histogram test. The top row shows the initialisations for $GF = 3$ and $GF = 4$ from left to right, the bottom row shows the respective final result after segmentation.

Parameters The granularity factor will be set to 3 and 4. Distance lockout will be 1, as all other distance lockouts for both granularity factors simply eliminated the third segment.

Results For both cases, both colours of text were identified along with the background. Due to the good initialisation, for both $GF = 3$ and $GF = 4$, the resulting segmentation showed a separation of background and both colours of text. These results are shown in Figure 12.16 and Table 12.10.

Average colour	(198, 184, 157)	(70, 61, 44)	(139, 101, 71)
Pixel count	421489	36623	29827

Average colour	(198, 184, 157)	(171, 112, 81)	(82, 73, 52)
Pixel count	323799	53856	22308

Table 12.10: Different results of the Menantes distance lockout histogram test. The initialisations for $GF = 3$ and $GF = 4$ are the top and bottom, respectively.

12.2.2 Conclusion

Unfortunately, the image we used had a very simple solution to the best method, namely the base histogram with $GF = 2$. However, disregarding this, we found that the different methods gave good results. For most methods, $GF = 4$ was unsalvageable, but with the distance lockout, even at this GF, the segmentation was good. Generally, the distance ranked histogram methods

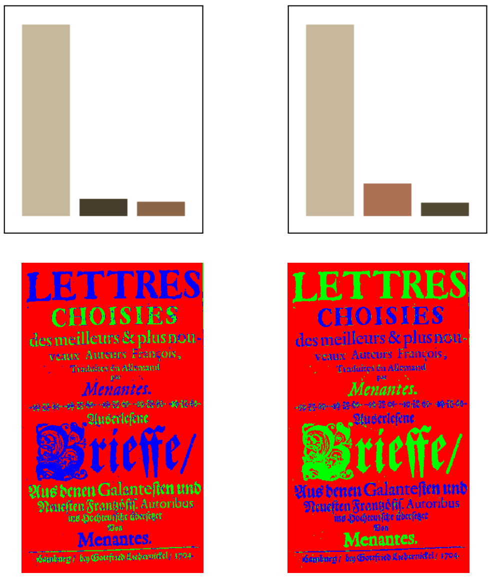


Figure 12.16: Different results of the Menantes distance lockout histogram test. The top row shows the initialisations for $GF = 3$ and $GF = 4$ from left to right, the bottom row shows the respective final result after segmentation.

should be utilised in scenarios where the distance is large between the buckets that would intuitively be deemed as good choices. Additionally, the distance lockout histogram method should be used in scenarios, where two adjacent buckets would be allocated most pixels using the other methods proposed here, leaving a more appropriate bucket unchosen. Changing the parameters of the distance ranked histogram methods would not always be able to solve this problem but the distance lockout histogram method would.

13
Multi-Layer Geofloral Segmentation

In this chapter, experiments will be run with two multi-layer segmentation methods, the multinomial method described in the previous report[1] and the logistic methods described in Chapter 6, on both synthetic and real world data to show the improvement of the new methods over the old method.

13.1 Synthetic Data Experiments

In this section we will describe the experiments we will be performing on the synthetic data sets from Section 13.1.1. Each test will be run with three data sets, the probability matrix model, the logistic model and a numerical model.

For each of these tests, 20 clusterings will be performed. These will be separated into two categories, reflecting the original segmentation, shown in Figure 13.1, and not reflecting this. This is referred to in the chapter as "correct" and "otherwise". Two criteria for a segmentation being correct are established. The first criterion is a visual confirmation, the resulting segmentation should in the 3-segment layer show three vertical bars and in the 2-segment layer show two horizontal bars. The second criterion is a stability measure, a measure of how many site configurations match those of another segmentation, in this case the original segmentation. We calculate this using the stability measure algorithm defined in Appendix A. A result must have a stability measure above 0.9 to be considered correct.

When likelihood value or value is noted in result analysis, this is to the log of the likelihood value, computed using Equation 13.1. We take this value instead of the likelihood value itself due to underflow problems. For more information on how we reached this equation, see Chapter 8.

$$ln(P(f) = \sum_i \sum_k ln(V_c(x_i, y_{i,k})) \tag{13.1}$$

13.1.1 Synthetic Data Sets

In this section we will describe the synthetic data sets that our experiments will rely on. Synthetic data will be used to illustrate how our algorithms behave when introduced to data sets that have specific properties, i.e. a data set that describes two layers with different segmentations. The purpose of these experiments on synthetic data is to provide empirical proof that our algorithms work as intended before moving on to data from the real world.

Generating Synthetic Geofloral Data

When we generate the synthetic data sets, we assign the presence of plants to regions according to distribution types. We have defined a distribution type as a matrix consisting of probabilities that will be used when assigning plants. The matrix will contain all possible segment combinations. For example, consider a data set with two layers containing three and two segments, respectively. A distribution type for this data set could look as in Table 13.1.

		L0		
		0	1	2
L1	0	$p_{0,0}$	$p_{1,0}$	$p_{2,0}$
	1	$p_{0,1}$	$p_{1,1}$	$p_{2,1}$

Table 13.1: Probability matrix example.

We determine the presence of a plant by using the probabilities of the distribution type that describe this plant. As we iterate over all sites, we look up the probability, p, in the distribution type and use this to determine whether or not the plant exists in this site. Pseudocode for our data set generation can be seen in Figure 13.1.

```
For each distribution type, $dt$
  For each site, $s$
    For $i = 1$ to $15$
      Look up probability, $p$, in $dt$ based on which segments $s$ belongs to
      Generate random number, $rand$, between $0$ and $1$
      if $rand < p$
        Assign $plant_{i,dt}$ as present in $s$
```

Listing 13.1: Pseudocode for generating a data set.

The generated data sets will all consist of two layers, where the sites in one layer will be distributed between three segments vertically and the sites in the other layer will be distributed horizontally, as illustrated in Figure 13.1. The purpose of the two layered data sets is to examine the quality of the clusterings produced by using the different layering models on them.

Direct Probability Matrix Data

This data set will consist of five distribution types that each prefers one segment in one layer. Three of these will prefer one of the vertical segments illustrated

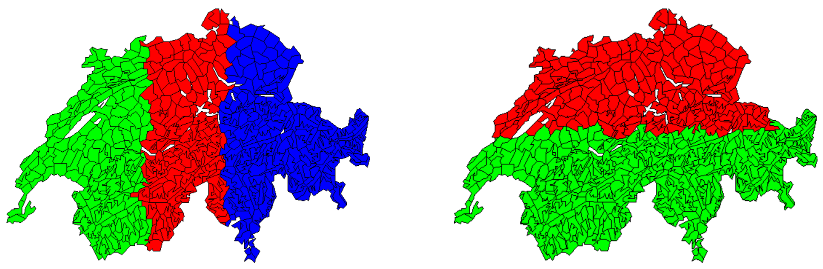


Figure 13.1: The original segmentation used to generate the data sets.

in Figure 13.1. The remaining two distribution types will prefer one of the horizontal segments illustrated in Figure 13.1. We will create variations of this data set as we want to gradually introduce more noise. An example of a distribution type preferring segment 0 in layer 0 is shown in Figure 13.2, where p is the given accuracy level, one of 1.0, 0.9, 0.8, 0.7 and 0.6.

		L0		
		0	1	2
L1	0	p	$1-p$	$1-p$
	1	p	$1-p$	$1-p$

Table 13.2: Probability matrix example.

The different values for p introduces noise in the data sets. This will be done for these data sets in order to evaluate how much noise our layered models are capable of handling before producing low quality segmentations.

Since randomness is involved, there is a possibility that a generated data set does not correspond well with the probabilities for each distribution type. In order to decrease the likelihood of that occurring, we add 15 plants to the data set for each distribution type. Thus, sites in a data set that has 5 distribution types will in actuality contain 75 plants. Note that the 75 plants still only describe 5 distribution types.

For each synthetic data set, we will create a reference file that details the original segmentations of the layers. This is useful when examining how close a given clustering result is to the original segmentation.

Logistic Distribution Data

Each distribution type in this data set will depend on both layers and the probabilities for each will be based on a logistic regression model. We want a different model for each distribution type, and we want them to describe distribution types that prefer different segment configurations. Each model will be described by a set of parameters, an intercept, and one parameter for each segment in each layer. One such distribution type can be seen in Table 13.3. These values are plotted into Equation 13.2 for all combinations of segments.

In Table 13.4 we have calculated the probabilities for the example distribution type.

β_0	β_{L_0,S_0}	β_{L_0,S_1}	β_{L_0,S_2}	β_{L_1,S_0}	β_{L_1,S_1}
0	2	-2	-2	2	-2

Table 13.3: Parameters for a logistic distribution model preferring segment 0 in layer 0 and segment 0 in layer 1.

		L0		
		0	1	2
L1	0	0.982	0.5	0.5
	1	0.5	0.018	0.018

Table 13.4: Probability matrix example for logistic distribution.

$$\text{logit} = \beta_0 + \beta_{L_0,S_x} + \beta_{L_1,S_y} \tag{13.2}$$

Numerical Distribution Data

The numerical distribution model works much like the logistic distribution type, with the key difference being that the model is numerical instead of categorical. Due to this, each layer only has one parameter, as shown in Table 13.5. Two distribution types are created, one spanning over layer 0, and one spanning over layer 1. The calculations for turning these into probabilities for use is slightly different, as instead of having a parameter for each, the parameter for the layer is multiplied by the segment ID, as shown in Equation 13.3. This results in the probabilities shown in Table 13.6.

β_0	β_{L_0}	β_{L_1}
-2	2	0

Table 13.5: Parameters for a numerical distribution model depending on layer 0.

		L0		
		0	1	2
L1	0	0.119	0.5	0.881
	1	0.119	0.5	0.881

Table 13.6: Probability matrix example for numerical distribution.

$$\text{logit} = \beta_0 + \beta_{L_0} \cdot S_x + \beta_{L_1} \cdot S_y \tag{13.3}$$

13.1.2 Multinomial Two Layer (3, 2) Segmentation

The purpose of this test is to examine how well our multinomial layering model works on our synthetic data sets. This is provided as a baseline to compare against, and is not expected to perform well based on the experimentation done in the previous report[1].

Results None of the results were correct, across all data sets. This, however, does not mean that the multinomial method cannot return the original segmentation, only that the probability of this happening is slim at best. Additionally, we can see that for the matrix model with $p = 1$, the incorrect segmentations have a likelihood value of 0, this indicates a perfect segmentation. This means that even if it returned a correct segmentation, this could not have a higher likelihood value. An example of a segmentation with a high likelihood value is shown in Figure 13.2.

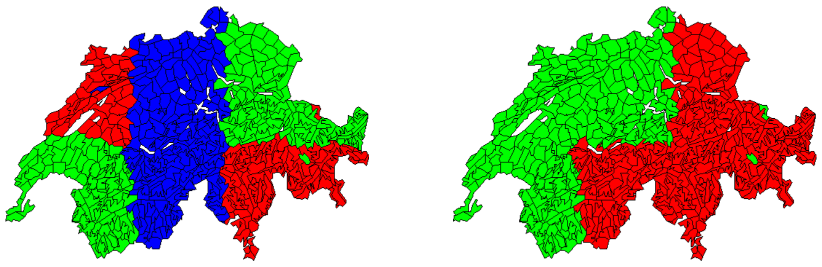


Figure 13.2: An example of a result of the multinomial model on the logistic dat set.

	$p = 1$	Logistic	Numerical
# Correct (of 20)	0	0	0
Highest value, correct	N/A	N/A	N/A
Lowest value, correct	N/A	N/A	N/A
Highest value, otherwise	0	-19660.296	-30142.9
Lowest value, otherwise	-8811.076	-23801.189	-30485.284

Table 13.7: Results for the multinomial two layer test.

13.1.3 Logistic Two Layer (3, 2) Categorical Segmentation

In this test, we will run a logistic regression with two layers on the data sets. Since the original data sets were all based on a two layer model, we expect that we will be able to approximately reproduce the original segmentation. For the data sets with noise, we expect that the results produced will gradually become worse as the noise increases.

Results For the probability matrix data sets, from a p of 1 to 0.8, the original segmentation as seen in Figure 13.1 is reproduced perfectly. At a p of 0.7 and below, the segmentation starts to be affected by noise, as seen in Figure 13.3. As such, we will focus on the results for the p values 0.8 and 0.7.

Not all results produced the original model, however this was reflected in a lower value. This is shown in Table 13.9. Examples of segmentations not reflecting the original model is shown in Figure 13.4. The reason for these results is that the algorithm works by looking for the local optimum based on the initialisation. However, there are several local optima, so the algorithm is not guaranteed to find the global optimum. The lower likelihood values allow us to perform several runs, and then pick the highest scoring result as the correct one.

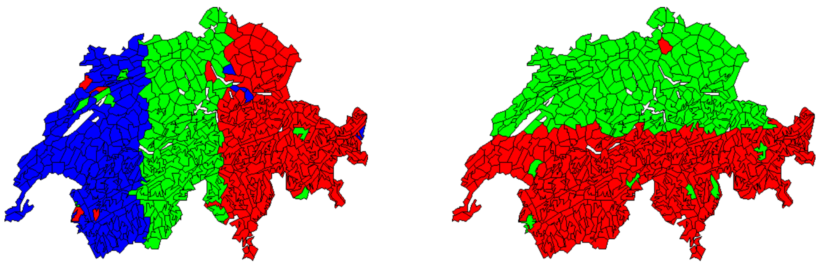


Figure 13.3: Examples of the results degrading for the $p = 0.7$ model.

	$p = 0.8$	$p = 0.7$	Logistic	Numerical
# Correct (of 20)	4	6	7	5
Highest value, correct	-21204.462	-25687.584	-19783.617	-30213.728
Lowest value, correct	-21204.462	-25700.7	-19912.104	-30228.431
Highest value, otherwise	-22124.804	-25996.11	-21677.922	-30320.508
Lowest value, otherwise	-23014.636	-26203.037	-23474.905	-30438.311

Table 13.8: Results for the logistic two layer categorical test.

13.1.4 Logistic Two Layer (3, 2) Numerical Segmentation

In this test, we run the logistic regression as a numerical segmentation. We do not expect this to do well on the categorical sets, but the numerical sets should reproduce the three tiers of plants as a segmentation.

Results In this test, for the probability matrix model, the segmentation started to degrade earlier, at $p = 0.8$. Due to this, we will focus on the p values of 0.9 and 0.8 for this test. For the $p = 0.9$ data set, the only correct result did not find a three tiered model. For the $p = 0.8$ data set, several results had a higher value than the correct results despite not being a correct segmentation. This is shown in Figure 13.6. All such results were found to be close to the original segmentation, but having eliminated the middle segment.

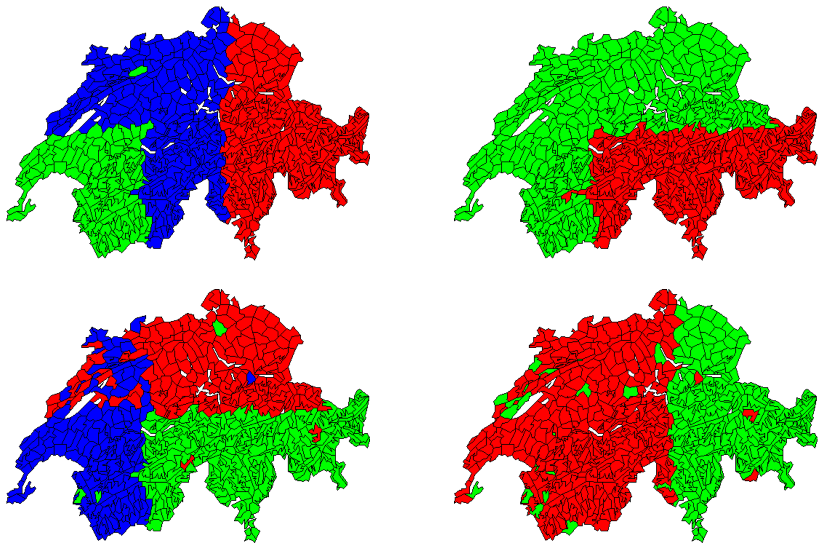


Figure 13.4: Examples of the algorithm not reflecting the original model. Shown are the two layers of the incorrect segmentation with the best value for the $p = 0.8$ model above and the same for the $p = 0.7$ model below.

For the numerical data set, this behaviour was not present, and in all segmentations representing the original segmentation, the three tiers were also reproduced. Slight noise was present in the second segment, which may be attributed to no plants being close to a probability of 1 for that segment.

	$p = 0.9$	$p = 0.8$	Logistic	Numerical
# Correct (of 20)	1	2	3	10
Highest value, correct	-17410.121	-23326.62	-23714.789	-30268.719
Lowest value, correct	-17410.121	-23361.778	-24258.447	-30270.395
Highest value, otherwise	-17620.916	-23258.869	-23787.384	-30615.567
Lowest value, otherwise	-18818.017	-23929.631	-25273.17	-30621.211

Table 13.9: Results for the numerical logistic two layer categorical test.

13.1.5 Logistic Single Layer (6) Categorical Segmentation

In this test, we show that it is possible to show a multi-layer segmentation in a single layer with a segment count equal to the product of the segment counts for each layer in the original segmentation.

Results For the probability matrix data set, we again found that the algorithm found the correct segmentation up to $p = 0.8$, and began to deteriorate at higher noise levels. For all data sets, the combined segmentation of the 3 segment and 2 segment layers was found, and had a similar likelihood value

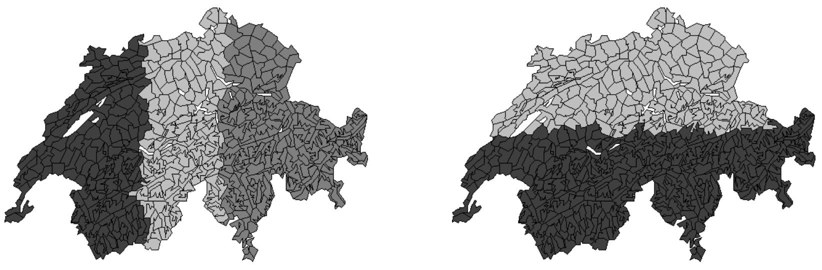


Figure 13.5: A result, where the first layer does not represent a three-tiered structure. The darkness of the colour represents the ordering of the segments.

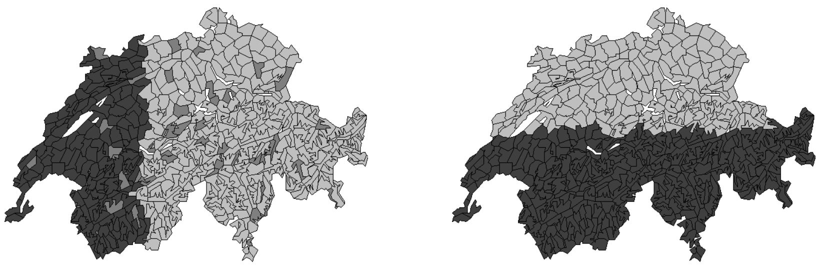


Figure 13.6: A result, where the likelihood value of an incorrect segmentation is higher than any correct segmentation.

to that of the multi-layer segmentation, as expected. Key data for the results are shown in Table 13.10, and an example of the segmentation is shown in Figure 13.7.

	$p = 0.8$	$p = 0.7$	Logistic	Numerical
# Correct (of 20)	8	10	4	5
Highest value, correct	-21122.732	-25587.077	-19660,296	-30134.102
Lowest value, correct	-21310.436	-25595.557	-19776,37	-30167.078
Highest value, otherwise	-21852.727	-25803.17	-21285,945	-30186.969
Lowest value, otherwise	-22724.238	-26080.505	-25162,264	-30340.318

Table 13.10: Results for the logistic single layer categorical test.

13.1.6 Conclusion

As expected, the multinomial method did not provide any correct results, and there were indications that any correct results found by chance would not be distinguishable using the likelihood value. For the logistic regression model, either categorical or numerical, with the right data, the results were clear, the methods work. An unexpected error was found in that the numerical method, using non-numerical data, tended to eliminate one segment in the first layer

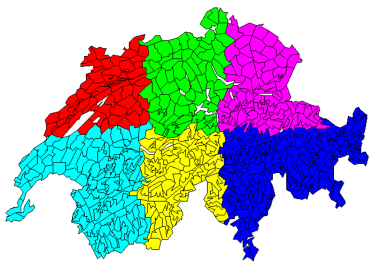


Figure 13.7: A result of the single layer 6 segment segmentation.

to fit the data found, leading to incorrect segmentations with higher likelihood values. With the numerical data set, this behaviour was not present.

13.2 Real World Experiments

In this section, we perform multi-layer segmentations with 2 and 3 layers on the Swiss plant data set. For both these, the segmentation will be done with a multinomial model as described in Chapter 5 and a logistic categorical and numerical model as described in Chapter 6. For all these, 20 runs will be performed, and the likelihood value, as defined in Chapter 8, will be used to determine which results will be focused on. The results of all the tests will be compared visually to maps of known climatological factors, shown in Appendix B or cited directly when relevant.

13.2.1 Real World Data Set Modifications

The data set used consists of 2398 plant species and is a subset of the data set in Section 2.2. Certain plants in the original data set were only classified as rare or frequent before a certain year in regions. Since we only use plants that are classified as frequent in the regions, these were eliminated completely. We cannot fit parameters for a logistic regression model with an intercept based on a plant that is not present in any regions, because it will have only one valid value (absent). We have thus excluded any plants that are not classified as frequent in one or more regions.

13.2.2 Two Layer (3, 3) Segmentation

In this test we will be using a configuration consisting of two layers, both containing three segments.

Results Table 13.11 contains the highest and lowest likelihood values for each model.

The multinomial method had the lowest likelihood compared to the other models. In Figure 13.8 and Figure 13.9, the best and worst result for the multinomial tests are displayed. Both results failed to present any relevant data

	Multinomial	Categorical	Numerical
Highest value	-405937.186	-278341.644	-296552.998
Lowest value	-433973.865	-288413.695	-308135.866

Table 13.11: Results for the two layer test.

and are mostly meaningless. There is no discernible difference between the best and worst result and we expect the higher likelihood value is due to random chance. While there are faint indications of mountains, the noise overpowers any meaning. As with the synthetic testing, the multinomial method is not able to separate the different factors of the data, and instead each layer becomes a mixture of different parameters, resulting in what seems like random noise.

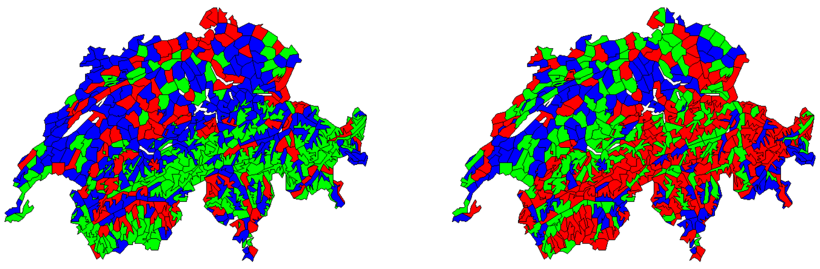

(a) Layer 0 (b) Layer 1

Figure 13.8: The best result for the (3, 3) multinomial segmentation.

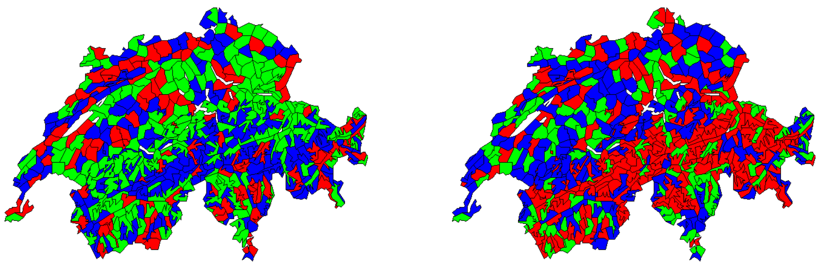

(a) Layer 0 (b) Layer 1

Figure 13.9: The worst result for the (3, 3) multinomial segmentation.

The result for the categorical model with the highest likelihood value can be seen in Figure 13.10. The two layers appear to be describing different factors of the data. In Figure B.1(b), an above/below tree line separation is depicted, which resembles layer 0. The red segment appears to describe the regions above the tree line, the green and blue segment seem to describe the regions below the tree line. We need to understand which aspects of the regions that the green and blue segment describe. In Figure B.1(a), a topographical map of Switzerland can be seen. It appears that the green segment describes regions below the tree line with nearly no elevation. Blue seems to describe regions below the tree line with an elevation higher than those regions in the green segment and less than those regions in the red segment. In regards to layer 1, we are unsure which factors are described. It is possible that more factors than one are being depicted in the segmentation. Looking at the average annual temperature of Switzerland[8], it appears the blue segment describes regions where the average annual temperature is around 8-10 °C. The green segment appears to be describing regions with an average annual temperature of around 2-4 °C.

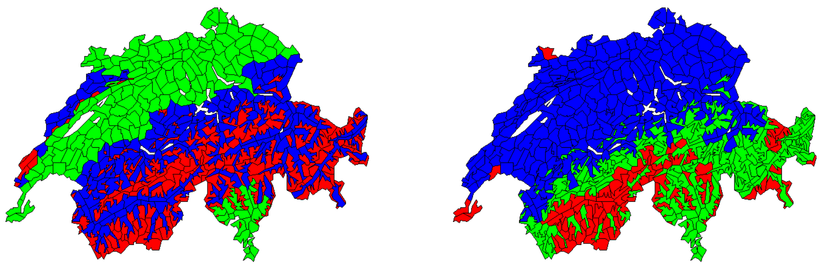

(a) Layer 0 (b) Layer 1

Figure 13.10: The best result for the (3, 3) categorical segmentation.

Finally, the results of the numerical model can be seen in Figure 13.11. Looking at the average precipitation[9], it appears that layer 0 approximately fits this although it might also be describing other factors that we are unable to identify. Referring again to the topographical map in Figure B.1(a), layer 1 appears to be descriptive of this.

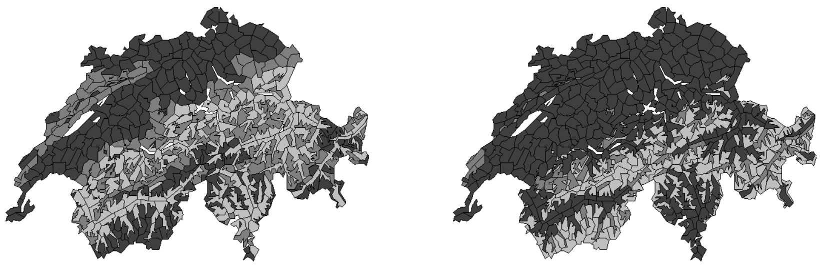

(a) Layer 0 (b) Layer 1

Figure 13.11: The best result for the (3, 3) numerical segmentation.

13.2.3 Three Layer (3, 3, 3) Segmentation

This test is a segmentation with three layers of three segments each.

Results The likelihood values for all three models are shown in Table 13.12. The first thing we notice is that for both logistic models, the likelihood values have gone up. This is due to having more layers, and as such, more parameters to fit, resulting in a closer fit than with only 2 layers. This behaviour is not present for the multinomial model.

	Multinomial	Categorical	Numerical
Highest value	-425282.323	-257595.446	-267004.937
Lowest value	-440968.815	-269318.291	-277660.525

Table 13.12: Results for the three layer test.

For the multinomial model, as with the previous test, no meaningful segmentation was found. The highest likelihood result is seen in Figure 13.12.

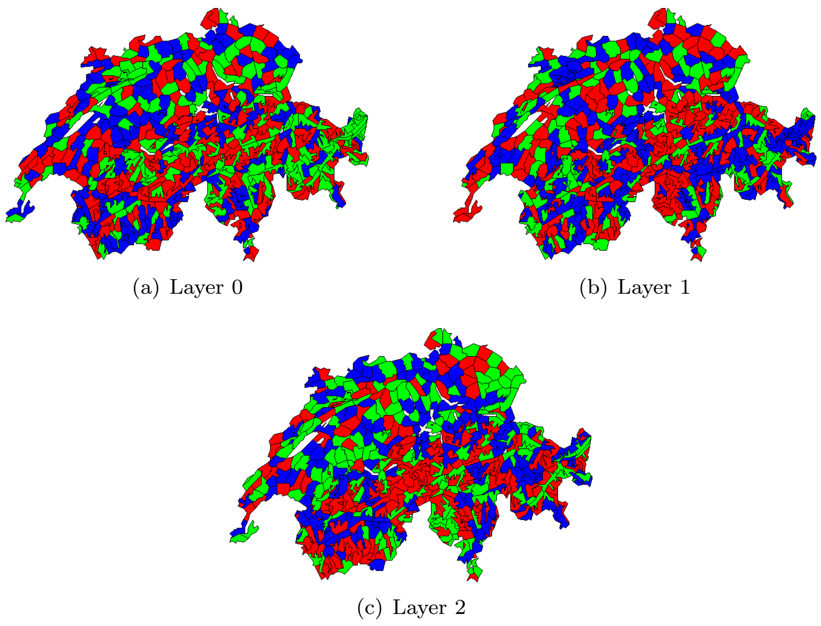
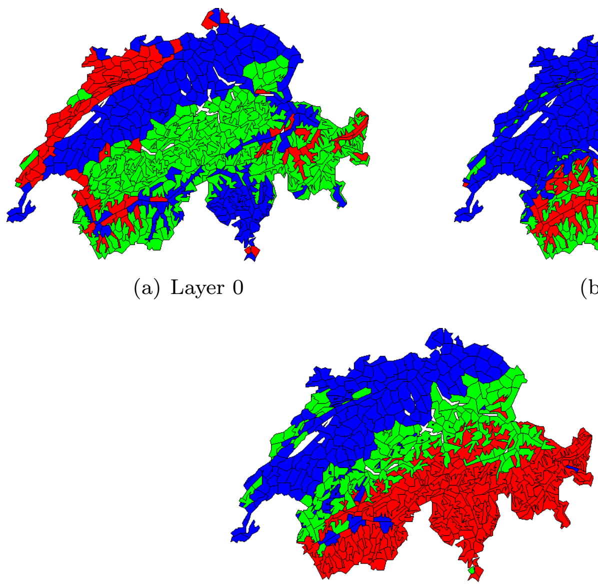

(a) Layer 0
(b) Layer 1
(c) Layer 2

Figure 13.12: The best result for the (3, 3, 3) multinomial segmentation.

For the categorical results, the results were far better, although they require manual interpretation. The highest likelihood result is seen in Figure 13.13. For this, layer 0 seems to be similar to the precipitation map[9], or the groundwater hardness map[10]. Layer 1 is very similar to the topographical map seen in Figure B.1(a), and layer 2 matches the groundwater hardness map again. This repetition of one factor is unlikely with a high likelihood value, but it may be because we misinterpreted one of the layers, or simply because another factor depends on the same underlying cause as the groundwater hardness, leading to two factors making two sets of plants favour the same regions.



(a) Layer 0

(b) Layer 1

(c) Layer 2

Figure 13.13: The best result for the (3, 3, 3) categorical segmentation.

For the numerical results, the results were again better than the multinomial results. The highest likelihood result is seen in Figure 13.14. In this, layer 0 matches the temperature map[8]. Layer 1 does not match any of the climate maps we have seen, however this does not mean the result is meaningless. To better understand what causes this layer, it would be interesting to look into the characteristic plants of this segment, this approach is described in more detail in Appendix C. Given a set of plants, a domain expert could then look at the common characteristics of these plants, to determine what the map is showing, if anything. This, however, is outside the scope of this project.

Finally, layer 2 is a north/south division, something that is in most of the three layer numerical segmentations. Again, this does not match any climate data we know of, however given how often it shows up, it is likely that this represents some climate factor we do not know of. Again, this would require a domain expert looking at the plants to determine the cause of this segmentation.

13.2.4 Conclusion

For multinomial results, the results found in the previous project holds, the method does not produce meaningful segmentations. For categorical and numerical results, it is possible to find approximate matches with climatological maps, indicating that the segmentations are meaningful. However, the approach of trying to match the segmentations to know climatological factors is

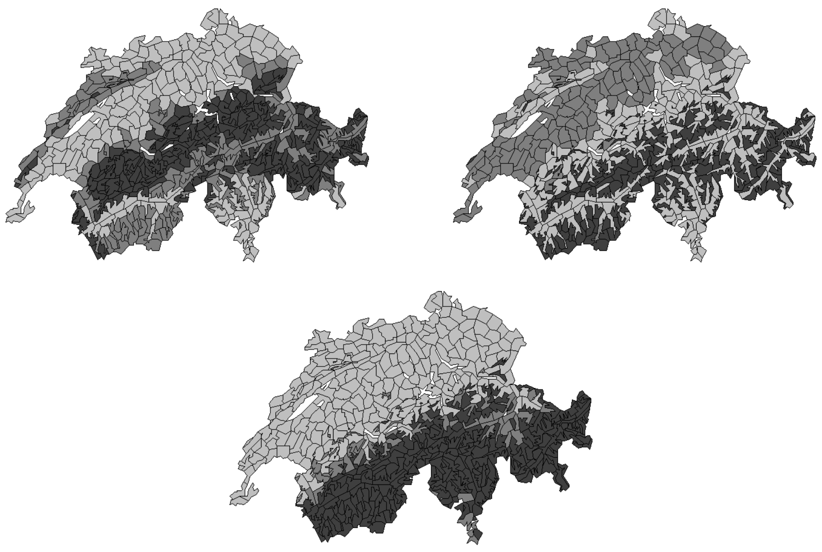


Figure 13.14: The best result for the (3, 3, 3) numerical segmentation.

not the right way to interpret these results. For a more scientific approach, the characteristic plants would be found using the method described in Appendix C, and the common characteristics of what would make these plants thrive can then be found. Based on these findings, the segmentations could then be used to find and map unknown climatological factors.

Part IV

Conclusion

14 Conclusion

The purpose of this report was to propose a method for clustering based on a multi-layer spatial clustering model along with initialisation methods.

Regarding initialisation, the transition from theory to implementation was straight-forward. The initially proposed naive model, base histogram, in Section 7.0.1, did not live up to our expectations and more specialised bucket selection heuristics were proposed.

These proved to be capable of handling the image tests, though a manual selection of the method is still necessary, as each method is highly dependent on the image used.

The multi-layer approach based on a logistic regression model was a vast improvement over the multinomial model implemented in the previous report. We tried multiple statistics libraries before settling on SPSS and found that few libraries provided results that were applicable to our project. We had issues with SPSS in terms of error reporting and performance but once resolved, SPSS provided better than results than any other library we examined.

For the synthetic data, the results were very clear and the logistic regression model provided the expected results even when subjected to a significant noise level. For the real world data, the results were less clear due to the fact that it requires extensive of the domain, i.e. the geofloral aspects of Switzerland and the countless climatological factors that affect these. Certain segmentations however were clearly understandable, e.g. the topographical segmentation, and we are confident that the results, given proper analysis by domain experts, would prove to be useful in understanding certain geofloral aspects.

In conclusion, we believe that we have succeeded in proposing both better initialisation methods for images as well as a multi-layer segmentation model providing meaningful results.

15 Future Development

This chapter outlines some of the possible topics for future development on this topic.

Multithreading The current implementation splits the computation up into two threads, a GUI thread and a worker thread. Optimally, to take advantage of all the cores in a machine, there should be enough worker threads handling part of the segmentation to provide work for all cores. This becomes especially relevant if the method is to be usable on computer clusters.

GPU Support While this was also mentioned in the previous report, it is still a viable research topic. If this is to be useful on a personal computing level, taking advantage of the GPU often present in modern PCs would provide an avenue for major speedups.

Multi-Layer Support for Images At the time of the project completion, the implementation sets a constraint in the code that images can only segment on one layer, as the histogram initialisation was the focus of the work for images. However, SPSS is able to handle numeric data, and the constraint can be removed, to try and see what this could accomplish for image segmentation. The expectation is that for a three-layer segmentation, the method would segment based on one colour, red, green and blue, for each layer.

Analysing Results With Help From Domain Expert The results found in Section 13.2 were analysed by visual comparison with known climatological factors. To make the results of this project more relevant to the plant research domain, it would be interesting to go through the results looking at which plants contribute most to the segmentation using mutual information calculations, as shown in Appendix C, and what these plants have in common in terms of what would make them thrive in a region. However, this is not a computer science project, and would be best left in the hands of experts in the field of flora.

Automatic Learning of Best Parameters At this time, a set of layers and beta values needs to be supplied for the segmentation. Using multiple restarts and the EM algorithm[11], it would be possible to find the best set of parameters for the data set.

Automation of Initialisation Method Selection As the implementation of histogram initialisation stands, the method needs to be selected by hand. Using multiple restarts, a logistic regression model for continuous values and likelihood values, it would be possible to do multiple restarts using different methods and parameters for initialisation to find the best result. Like the learning of segmentation parameters, this could be done with the EM algorithm.

Heuristics for Multiple Restarts While multiple restarts are not used in the report, it is currently implemented in such a way that the number of intended runs is supplied, and the method returns the segmentation with the highest likelihood value. It would be beneficial to develop a heuristic that looks at how different the likelihood values and similarity of the segmentations done so far during the run are, to estimate the probability of seeing a radically different segmentation, and then determining whether or not the run should terminate.

Soft Clustering This report determines a method for hard clustering, i.e. for each iteration, each site is set fully to one segment configuration. In cases where the probabilities of segment configurations are close to each other, this does not fully describe the data. Instead, the segment configurations could simply be set to the probabilities for each one, and these could then be used in the next iteration.

Part V

Appendices

A Similarity of Segmentations

This chapter describes a method for comparing two segmentations, where the segmentations are assumed to have the same layer configuration in terms of number and ordering of layers, and number of segments in each layer. This was constructed for the previous report[1], but is used here for the same purposes, namely synthetic geofloral data testing.

A.1 Implementation

We have designed a greedy algorithm that finds a near-optimal solution. We will not discuss how to count the number of sites, as this is a trivial problem. Therefore, we assume that we have a number of mappings from a segment in one segmentation to a segment in another segmentation, along with the count of how many sites fit this mapping. The mappings for the segmentations seen in Figure A.1 is seen in Table A.1.

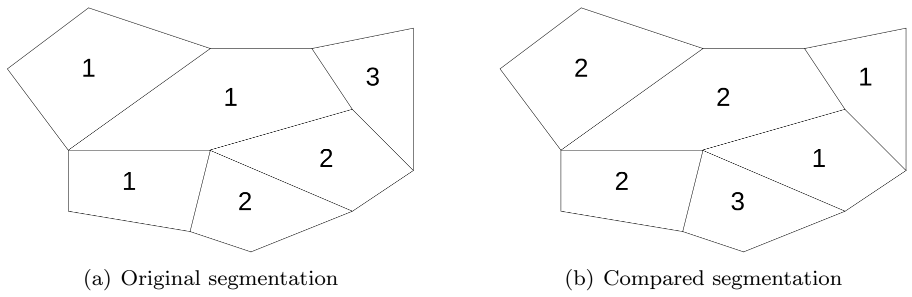


Figure A.1: An example of a mapping between two segmentations.

$1 \to 1$	0	$2 \to 1$	1	$3 \to 1$	1
$1 \to 2$	3	$2 \to 2$	0	$3 \to 2$	0
$1 \to 3$	0	$2 \to 3$	1	$3 \to 3$	0

Table A.1: Mappings corresponding to Figure A.1

1	$1 \rightarrow 2$	3
2	$2 \rightarrow 1$	1
3	$2 \rightarrow 3$	1
4	$3 \rightarrow 1$	1
5	$3 \rightarrow 3$	0

Table A.2: Sorted and abbreviated mappings of Figure A.1.

This list of mappings is then sorted according to the counts, from highest to lowest. Two lists of booleans are maintained, detailing which segments have been assigned in either segmentation.

Beginning from the top, the mappings are iterated, with a mapping being assigned to a segment in either segmentation, if these have not yet been assigned. Once all segments are assigned, the counts are summed up and divided by the number of sites, giving the final result.

One problem exists with this approach. In the example in Figure A.1, two segmentations are being compared. While an optimal selection of mappings is intuitively available, the algorithm will have a problem with these segmentations.

The mappings are sorted resulting in Table A.2. Some results are left out, as they are not relevant to the problem.

In both the optimal selection, and the selection returned by the algorithm, index 1 is chosen first. In the algorithm, index 2 is chosen next, and index 3 is skipped over, as segment 2 on the left side has already been taken. Finally, index 4 is skipped over due to 1 on the right side being taken, and index 5 is taken last, leading to a segmentation rating of 4/6.

Optimally, indices 1, 3 and 5 would be taken instead, for a segmentation rating of 5/6.

This problem is due to the greedy approach of the algorithm, and is only apparent, when the counts of two different mappings are the same. This makes the problem unlikely in the normal operation of the program. The only solution is to search through all possible setups, leading to an exponential complexity. As such, we accept this problem as a necessary cost of effectiveness in the algorithm.

The complete algorithm is expressed in the pseudocode in Listing A.1.

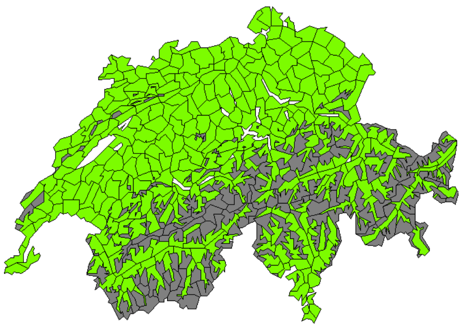
```
Initialise mappings ;Count number of correct sites for each mapping
Sort mappings ;Sort according to count
Initialise leftSegments, rightSegments ;Initialise booleans for segments on
     both sides
Initialise sum

for each (mapping in mappings)
  if leftSegments[mapping.leftSegment] and rightSegments[mapping.
       rightSegment] ;If both segments not assigned
    sum += mapping.count

return sum / numSites
```

Listing A.1: Pseudocode for the calculation of the stability index.

B
Domain Information on Switzerland

In order to determine whether the segmentations produced for the real world geofloral data set are meaningful, we need to have an understanding of the data. In this section we will describe the domain information that is used for explaining certain results in Section 13.2.

(a) A topographical map of Switzerland. Source: [12]



(b) A mountain and valley separation extracted from the geofloral data set. Gray indicates a region above the tree line, i.e. a mountain region. Green indicates a region below the tree line, i.e. a valley region.

C
Characteristic Plants

When clustering on real world geofloral data, the amount of potential factors, e.g. soil pH-value or precipitation, is very high. This means that identifying the factors contributing to the segmentation of each layer is very hard when only looking at the resulting segmentation.

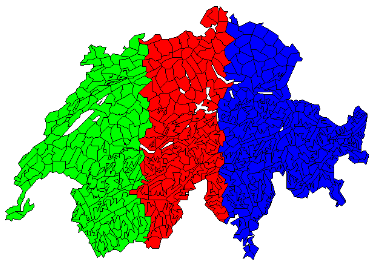
However, given information about where the plants grow, and the final segmentation, the mutual information for each plant can be calculated using the equation seen in Equation C.1, where x is a given segment ID and p is a given plant. Additionally, instead of $\sum\limits_{x,\not{x}}$, it is possible to change this to $\sum\limits_{x \in L_n}$, and find the mutual information of a plant and a given layer.

$$MI(x,p) = -\sum_{x,\neg x}\sum_{p,\neg p} p(x,p) \cdot log(\frac{p(x)p(p)}{p(x,p)} \tag{C.1}$$

With the mutual information for all plants for a given segment or layer, those with the highest value are then the most characteristic plants for that segment or layer. For a segment, this would typically mean a plant that either grows in every region in the segment and no other regions, or a plant that grows in every region except for the regions in the segment.

Given these plants, one would then be able to look into the common characteristics for the highest rated plants, looking for climatological factors that limit where these plants would grow, e.g. soil pH. If one or more such factors are found, the layer or segment for which the plants are characteristic could then be assumed to describe an area with this climatological factor.

Additionally, the mutual information for layers can be used to indicate whether or not a plant is descriptive of more than one layer. If the mutual information for one plant is approximately the same for two or more layers, this plant is equally descriptive of both those layers. A good example of this is the synthetic data sets. For the direct probability data, each plant is only descriptive of one layer by design, as the probabilities are identical across all segments for one layer or the other. For the logistic data, the opposite is true, as for any given plant, the logistic parameters for the distribution have one high parameter for each layer, as described in Section 13.1.1. When looking at the mutual information for a correct result, seen in Figure C.1, for both these data sets, the difference is clear, as is shown in the plots in Figure C.2. For Figure C.2(a), the plot for the direct probability data set result, the plants are separated into two groups, one favouring layer 0 and one favouring layer 1. For Figure C.2(b), the plot for the logistic data result, the plants are all around the line marking $x = y$.


(c) Layer 0

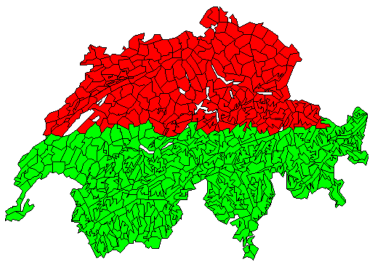
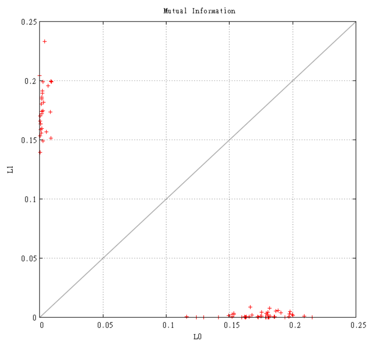
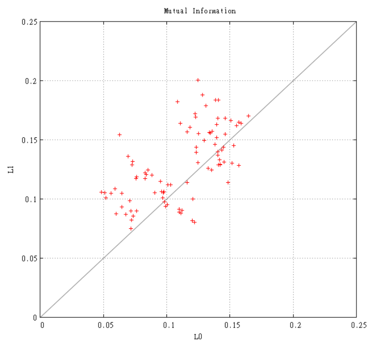

(d) Layer 1

Figure C.1: The resulting segmentation for both data sets used to generate the mutual information plots.



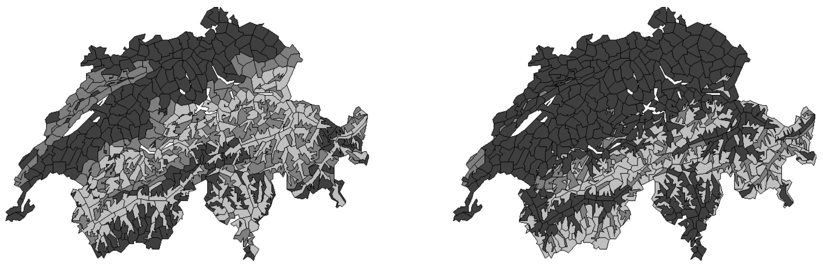
(a) Direct Probability



(b) Logistic

Figure C.2: Plots for all plants by their mutual information values, for layer 0 on the x-axis, and layer 1 on the y-axis.

For real world data, the plots can be used to determine which of the two layers is the most important in the segmentation. For the result of a numerical clustering on the real world data, shown in Figure C.3, the plot, shown in Figure C.4, favours layer 1. Looking at the segmentation, this seems to represent the topography factor, which correlates well with how often we see this segmentation, and what this says about the impact the topography has on the Swiss flora.



(a) Layer 0 (b) Layer 1

Figure C.3: The numerical result on the real world data used to generate the plot.

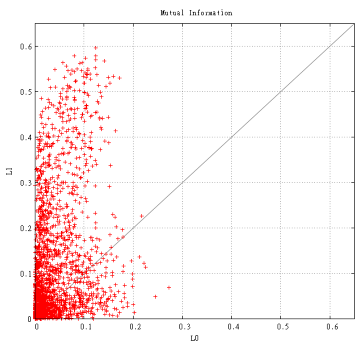


Figure C.4: A plot for a numerical result on real world data, for all plants by their mutual information values, for layer 0 on the x-axis, and layer 1 on the y-axis.

Bibliography

[1] Michael Wodstrup Vandborg Simon Piepgras Lyager. Cluster analysis of spatial data, 2010.

[2] Stan Z. Li. *Markov Random Field Modeling in Image Analysis.* Springer, 3rd edition, 2009.

[3] Stephen Smith Yongyue Zhang, Michael Brady. Segmentation of brain mr images through a hidden markov random field model and the expectation-maximization algorithm. *IEEE transactions on medical imaging*, 20(1):45–57, January 2001.

[4] Peter Longatti Thomas Wohlgemuth, Katia Boschi. Swiss web flora. `http://www.wsl.ch/land/products/webflora/floramodul2-en.html`.

[5] Matthieu Moisse. Clustering biogeographic data using relational model. Master's thesis, Aalborg University, 2010.

[6] Merilee A. Hurn et all. *A Tutorial on Image Analysis.* Springer, 3rd edition, 2009.

[7] Peter Orchard. Markov random field optimisation. `http://homepages.inf.ed.ac.uk/rbf/CVonline/LOCAL_COPIES/AV0809/ORCHARD/`.

[8] Stock Map Agency. Temperature map of switzerland. `http://www.stockmapagency.com/Temperature_Map_Switzerland_C-Swit-2007-Temp.php`.

[9] Stock Map Agency. Precipitation map of switzerland. `http://www.stockmapagency.com/Precipitation_Map_Switzerland_C-Swit-2007-Precip.php`.

[10] Federal Office for the Environment. Natural constituents of groundwater. `http://www.bafu.admin.ch/grundwasser/07496/07518/index.html?lang=en`.

[11] Wikipedia. Expectation-maximization algorithm. `http://en.wikipedia.org/wiki/Expectation-maximization_algorithm`.

[12] Wikipedia. Topographical map of switzerland. `http://en.wikipedia.org/wiki/File:Switzerland_topographic.png`.

Part VI

Supplements

D
Factorial Clustering with an Application to Plant Distribution Data

This supplement is a paper written based on the method defined in this report. This paper is not included to be a part of the report, only to show a different approach to explaining the theory and experimentation performed throughout this report. The paper will be submitted to the 2nd MultiClust Workshop, held as a part of European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases (ECML PKDD) 2011.

Factorial Clustering with an Application to Plant Distribution Data

Manfred Jaeger[1], Simon P. Lyager[1], and Michael W. Vandborg[1] and Thomas Wohlgemuth[2]

[1] Dept. of Computer Science, Aalborg University, Denmark
[2] Swiss Federal Research Institute WSL

Abstract. We propose a latent variable approach for multiple clustering of categorical data. We use logistic regression models for the conditional distribution of observable features given the latent cluster variables. This model supports an interpretation of the different clusterings as representing distinct, independent factors that determine the distribution of the observed features. We apply the model for the analysis of plant distribution data, where multiple clusterings are of interest to determine the major underlying factors that determine the vegetation in a geographical region.

1 Introduction

There exist a variety of different approaches to learning multiple clusterings. They can differ not only with regard to their mathematical models and algorithmic methods, but there can also be widely different intuitions and objectives with regard to the interpretation of the multiple clusterings. On the one hand, in ensemble clustering, the individual clusterings are essentially regarded as different, imperfect versions of a single underlying true clustering (e.g. [8]). In many multiple clustering methods, on the other hand, the different clusterings are intended to represent different views of the data, each providing a different insight into the structure of the data. One objective for clustering methods then is to ensure that different clusterings are in some sense independent, disparate [4], or non-redundant [6].

Several authors have investigated probabilistic latent variable models for multiple clusterings [11, 3, 1]. For the case of discrete observable features, no special assumptions on the distributional form of the features given the latent variables are made in these approaches, i.e. the conditional distribution of the features follows an unconstrained multinomial distribution. Latent variable models are also commonly used for dimensionality reduction of high-dimensional numeric data. An important example is the *factor analysis* model, in which the observed data is interpreted as a noisy linear transformation of a small number of latent dimensions.

In this paper we propose a probabilistic latent variable model for multiple clusterings. As in factor analysis, we interpret the observed (discrete) data as a noisy transformation of underlying, discrete latent dimensions. The linear mappings of factor analysis is replaced by a log-linear logistic regression model. The latent dimensions then define clusterings that can be seen as independent factors that determine the distribution of the observed features.

In contrast with several other multiple clustering methods (e.g., [3, 6]) our method is not based on an association of different clusterings with different feature subsets, even though such associations can emerge.

Our approach is partly motivated by applications to biogeographical data. Specifically, we are investigating plant distribution data. Segmentations of geographic units into floristic regions based on similarity of plant species composition were already undertaken in the 19th century. An early application of formal methods of clustering in this context is [7]. We apply our method to distribution data for 2398 plant species in Switzerland. The goal of factorial clustering for this type of data will be to obtain multiple clusterings, each of which could correspond to one of several underlying environmental, geographical, or historical factors, which jointly influence the vegetation.

2 Latent Variable Models for Clustering

Latent variable models are routinely used for clustering, both for single and multiple clustering. However, they can be used in several, slightly different ways. In order to more clearly explain our approach, we briefly review in this section possible approaches to using latent variable models for clusterings.

Throughout, we assume that the observable data $\boldsymbol{X}$ consists of n observations of k attributes, i.e. $\boldsymbol{X}$ is an $n \times k$ matrix. A latent variable model contains m additional unobserved variables, and we denote with $\boldsymbol{L}$ the $n \times m$ matrix of the latent variables in the n observations. We note that when we assume that in the n observations both the observable and latent variables are identically and independently sampled, it will be simpler and more natural to describe the model in terms of vectors $\boldsymbol{X}$, $\boldsymbol{L}$ of length n and m, respectively. However, in some applications, especially segmentation of time sequences or images, the latent variables are not independent at different data points.

A latent variable model, then, consists of a joint distribution for $\boldsymbol{X}$ and $\boldsymbol{L}$, which can be written as

$$P(\boldsymbol{X} \mid \boldsymbol{L}, \theta_{\boldsymbol{X}|\boldsymbol{L}})P(\boldsymbol{L} \mid \theta_{\boldsymbol{L}}). \tag{1}$$

In hierarchical models, this might be extended by a distribution over $\theta_{\boldsymbol{X}|\boldsymbol{L}}, \theta_{\boldsymbol{L}}$ parameterized by hyperparameters $\boldsymbol{\lambda}$.

The perhaps most common use of model (1) for clustering is to perform two steps [11]: first, fit the parameters $\theta_{\boldsymbol{X}|\boldsymbol{L}}, \theta_{\boldsymbol{L}}$ by maximizing the marginal likelihood of the observed data $\boldsymbol{X} = \boldsymbol{x}$:

$$(\theta^*_{\boldsymbol{X}|\boldsymbol{L}}, \theta^*_{\boldsymbol{L}}) := \underset{\theta_{\boldsymbol{X}|\boldsymbol{L}}, \theta_L}{arg\,max} \sum_{\boldsymbol{l}} P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{L} = \boldsymbol{l}, \theta_{\boldsymbol{X}|\boldsymbol{L}})P(\boldsymbol{L} = \boldsymbol{l} \mid \theta_{\boldsymbol{L}}). \tag{2}$$

This step is usually performed using the EM algorithm. Then, compute the most probable values of $\boldsymbol{L}$ given $\boldsymbol{X} = \boldsymbol{x}$:

$$\boldsymbol{l}^* = \mathop{arg\,max}_{\boldsymbol{l}} P(\boldsymbol{L} = \boldsymbol{l} \mid \boldsymbol{X} = \boldsymbol{x}, \theta^*_{\boldsymbol{X}|\boldsymbol{L}}, \theta^*_{\boldsymbol{L}}) \tag{3}$$

In multiple clustering, a joint configuration of the latent variables defines multiple cluster indices. For simplicity we may assume for now that each latent variable defines its own clustering, and that therefore the membership of the ith data item in the jth clustering is given by $\boldsymbol{l}^*_{i,j}$. However, in the multi-cluster case, the second step can also take a slightly different form, and the most probable latent variable values be computed component-wise. Denoting by $\boldsymbol{l}_j$ the jth column of $\boldsymbol{l}$, this can be written as

$$\boldsymbol{l}^*_j = \mathop{arg\,max}_{\boldsymbol{l}_j} \sum_{\boldsymbol{l}_1,\ldots,\boldsymbol{l}_{j-1},\boldsymbol{l}_{j+1},\boldsymbol{l}_m} P(\boldsymbol{L} = \boldsymbol{l} \mid \boldsymbol{X} = \boldsymbol{x}, \theta^*_{\boldsymbol{X}|\boldsymbol{L}}, \theta^*_{\boldsymbol{L}}). \tag{4}$$

This is the (hard) clustering rule used, e.g., in [11, 12]. The clusterings obtained from (3) and (4) can differ.

If the ultimate goal is only to compute a most probable configuration of $\boldsymbol{L}$, then one may also try to simplify the combination of (2) and (3) into a single optimization:

$$\boldsymbol{l}^* := \mathop{arg\,max}_{\boldsymbol{l}} \max_{\theta_{\boldsymbol{X}|\boldsymbol{L}},\theta_{\boldsymbol{L}}} P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{L} = \boldsymbol{l}, \theta_{\boldsymbol{X}|\boldsymbol{L}}) P(\boldsymbol{L} = \boldsymbol{l} \mid \theta_{\boldsymbol{L}}). \tag{5}$$

This rule can be justified by a Bayesian interpretation, for example: it amounts to finding the jointly most probably values of $\boldsymbol{l}, \theta_{\boldsymbol{X}|\boldsymbol{L}}, \theta_{\boldsymbol{L}}$, given the data $\boldsymbol{X} = \boldsymbol{x}$, and assuming a uniform prior for $\theta_{\boldsymbol{X}|\boldsymbol{L}}, \theta_{\boldsymbol{L}}$. Rule (5) may be still further simplified, if one assumes the model for the latent variables to be fixed, and not subject to optimization, i.e., $P(\boldsymbol{L} = \boldsymbol{l} \mid \theta_{\boldsymbol{L}}) = P(\boldsymbol{L} = \boldsymbol{l} \mid \theta^*_{\boldsymbol{L}})$ for fixed parameters $\theta^*_{\boldsymbol{L}}$, and the parameter optimization is only for $\theta_{\boldsymbol{X}|\boldsymbol{L}}$. If, furthermore, $P(\boldsymbol{L} = \boldsymbol{l} \mid \theta^*_{\boldsymbol{L}})$ is assumed uniform, then (5) reduces to

$$\boldsymbol{l}^* := \mathop{arg\,max}_{\boldsymbol{l}} \max_{\theta_{\boldsymbol{X}|\boldsymbol{L}}} P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{L} = \boldsymbol{l}, \theta_{\boldsymbol{X}|\boldsymbol{L}}). \tag{6}$$

Whether it is justified to assume a fixed distribution $P(\boldsymbol{L} \mid \theta^*_{\boldsymbol{L}})$ can depend on two considerations: first, assuming that (1) actually represents the generative process for the data, one might have sufficient background knowledge to identify the distribution of $\boldsymbol{L}$ a-priori. $\boldsymbol{L}$ being an unobserved variable, whose existence is essentially hypothesized, and for which it is typically even unclear how many states it has, this is a rather unlikely case in practice, however. Second, clustering being an exploratory data-analysis tool, one may also consider what settings of $P(\boldsymbol{L} \mid \theta^*_{\boldsymbol{L}})$ may lead via (5) to interesting insights into the data, regardless of whether the underlying probabilistic model is accurate as a generative model.

For example, in the single clustering case, when the data is generated by a mixture model where one mixture component has a much higher prior probability than the others, then clustering via (3) can easily lead to only obtaining a single cluster. If, on the other hand, one eliminates the influence of the prior distribution by assuming (incorrectly) a uniform distribution over the mixture components, then clustering via (6) can reveal the mixture structure of the data.

3 The Factorial Logistic Model

In the factor analysis model, both $\boldsymbol{X}$ and $\boldsymbol{L}$ are numerical, the rows in $\boldsymbol{X}$ and $\boldsymbol{L}$ are iid, and the model (1) is given by distribution

$$\begin{aligned}&P(\boldsymbol{L}_i) \sim N(\mathbf{0}, \boldsymbol{\Sigma}_{\boldsymbol{L}})\\ &P(\boldsymbol{X}_i \mid \boldsymbol{L}_i) \sim N(\boldsymbol{W}\boldsymbol{L}_i + \boldsymbol{\mu}, \boldsymbol{\Sigma}_{\boldsymbol{X}})\end{aligned}$$

where $\boldsymbol{\Sigma}_{\boldsymbol{L}}$ is an arbitrary covariance matrix, $\boldsymbol{W}$ is a $k \times m$ matrix, $\boldsymbol{\mu}$ a m-dimensional mean vector, and $\boldsymbol{\Sigma}_{\boldsymbol{X}}$ a diagonal covariance matrix. Thus, data is assumed to be generated by sampling from a lower (k) dimensional Gaussian distribution, linearly mapped into the higher (m) dimensional space, and independent Gaussian noise added to each coordinate.

The logistic regression model for the distribution of a binary variable X conditional on numeric latent variables $\boldsymbol{L}$ is given by

$$\log P(X = 1 \mid \boldsymbol{L})/P(X = 0 \mid \boldsymbol{L}) = w_0 + \boldsymbol{w}\boldsymbol{L}, \tag{7}$$

where $\boldsymbol{w} = (w_1, \ldots, w_k)$ is a k-vector of real weights. We write $X \sim LR(w_0, \boldsymbol{w}\boldsymbol{L})$ if X follows (7). This model also applies when the latent variables $\boldsymbol{L}$ are *ordinal*, i.e. each L_j codes by an integer $\{0, \ldots, r_j - 1\}$ one of r_j different, ordered categories. To accommodate *nominal* predictor variables (i.e., unordered categorical variables) in the logistic regression model, one encodes a nominal variable L_j with r states by binary indicator variables $L_{j,1}, \ldots, L_{j,r}$, i.e. $L_{j,h} = 1$, $L_{j,h'} = 0$ ($h' \neq h$) means that L_j is in its hth state.

We will consider both ordinal and nominal latent variables for clustering. An ordinal latent variable defines an ordered clustering, i.e. the cluster indices define an ordering of the clusters. Whether such an ordering is meaningful and interpretable is application dependent. For biogeographical data ordinal latent variables and ordered clusterings are often natural, since data patterns are often determined by underlying continuous variables. We will, thus, assume that $\boldsymbol{L}$ is a vector of m latent variables that define c different clusterings. Furthermore, we assume that one of the following two cases applies: (1) all L_j in $\boldsymbol{L}$ are ordinal; in this case $c = m$, and the jth clustering consists of r_j distinct cluster. (2) $\boldsymbol{L}$ is an encoding by binary indicator variables of c distinct nominal variables with $r_1, \ldots, r_c$ distinct states, respectively. In this case $m = \sum_{i=1}^{c} r_i$. We refer to model (1) as the $(r_1 o, \ldots, r_k o)$ model, and (2) as the $(r_1 n, \ldots, r_c n)$ model. One could also consider models combining ordinal and nominal latent variables, but we will here focus on "pure" models.

As in the factor analysis model, we assume that $P(\boldsymbol{X} \mid \boldsymbol{L}) \sim \prod_{i=1}^{n} P(\boldsymbol{X}_i \mid \boldsymbol{L}_i) \sim \prod_{i=1}^{n} \prod_{j=1}^{k} P(\boldsymbol{X}_{i,j} \mid \boldsymbol{L}_i)$. Assuming that each $X_{i,j}$ follows a logistic regression model (7) with parameters $w_{j,0}, \boldsymbol{w}_j$, one obtains the model for the ith data item:

$$P(\boldsymbol{X}_i \mid \boldsymbol{L}_i) \sim \prod_{j=1}^{k} LR(w_{j,0}, \boldsymbol{w}_j \boldsymbol{L}_i). \tag{8}$$

This conditional model for $\boldsymbol{X}$ may be combined with various models for $P(\boldsymbol{L})$, with or without an iid assumption for the rows of $\boldsymbol{L}$. We refer to multiple clustering based on (8) as *factorial logistic (FL)* clustering.

4 Learning

We apply the simple learning rule (6) for clustering with the logistic regression model. Thus, we assume that $\boldsymbol{L}$ is uniformly distributed, which implies, in particular, independence over rows: $P(\boldsymbol{L}) \sim \prod_i P(\boldsymbol{L}_i)$. In case of $\boldsymbol{L}$ encoding nominal variables, the uniform distribution, of course, is conditional on "legal" states of $\boldsymbol{L}$, i.e. at most one indicator variable for any particular nominal variable being equal to 1.

For the optimization of (6) we then use the obvious iterative procedure, where after a random initialization $\boldsymbol{L} := \boldsymbol{l}_0$ two steps are alternated:

i $\theta_{\boldsymbol{X}|\boldsymbol{L}_t} := \mathit{arg\,max}_{\theta_{\boldsymbol{X}|\boldsymbol{L}}} P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{L} = \boldsymbol{l}_t, \theta_{\boldsymbol{X}|\boldsymbol{L}})$
ii $\boldsymbol{l}_{t+1} := \mathit{arg\,max}_{\boldsymbol{l}} P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{L} = \boldsymbol{l}, \theta_{\boldsymbol{X}|\boldsymbol{L}_t})$

Step **i** is performed in our implementation using the SPSS method of fitting logistic regression models, which supports both ordinal and nominal predictor variables. Due to the factorization (8), the optimization reduces to k independent optimizations for the parameters $(w_{j,0}, \boldsymbol{w}_j)$ $(j = 1, \ldots, k)$. It is thus linear in k. It also is linear in n, since the likelihood only depends on the counts $|\{i \mid \boldsymbol{X}_{i,j} = 1, \boldsymbol{L}_i = \hat{\boldsymbol{l}}\}|$ for fixed configurations $\hat{\boldsymbol{l}}$ of the latent variables.

For step **ii** we have $P(\boldsymbol{X} = \boldsymbol{x} \mid \boldsymbol{L} = \boldsymbol{l}, \theta_{\boldsymbol{X}|\boldsymbol{L}_t}) = \prod_i P(\boldsymbol{X}_i = \boldsymbol{x}_i \mid \boldsymbol{L}_i = \boldsymbol{l}_i, \theta_{\boldsymbol{X}|\boldsymbol{L}_t})$, so that the problem decomposes into n distinct optimizations for the $\boldsymbol{l}_i$. It can be naively performed by computing $P(\boldsymbol{X}_i = \boldsymbol{x}_i \mid \boldsymbol{L}_i = \boldsymbol{l}_i, \theta_{\boldsymbol{X}|\boldsymbol{L}_t})$ for each candidate $\boldsymbol{l}_i$, which gives a procedure that is still linear in n and k, but exponential in c.

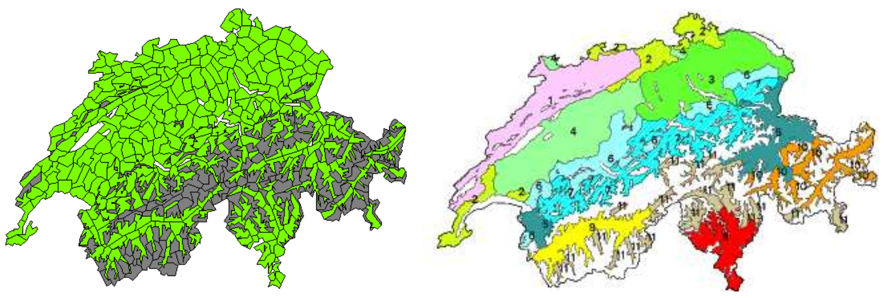
Overall, we obtain a learning method that is linear in the number of data items and the observable attributes, and exponential in the number of clusterings.

5 Experiments

We apply FL-clustering to geobotanical data. In our experiments we use the source data for the "Swiss Web Flora" [3] [9]. The dataset contains information on the distribution of 2697 plant species in Switzerland, which has been divided into 565 mapping areas. We reduced slightly more detailed species abundance information in the original data to simple binary presence/absence data. We also in this process deleted plants with a very sparse and uncertain distribution. This left us with 2398 species in our data.[4] We view each plant species as an

[3] `www.wsl.ch/land/products/webflora/welcome-en.ehtml`
[4] The data will be made publicly available

observable attribute, and the mapping areas as independent observations. Thus, $n = 565$ and $k = 2398$ in the notation of Section 2. Figure 1 shows the division of Switzerland into the mapping areas. Apart from the species occurrence data, only a single additional variable is recorded for each area: a binary variable that indicates whether the area is a mountain area (above timberline), or a valley area (below timberline). The value of this variable is shown if Figure 1 by a green color for valley, and grey color for mountain areas.



(a) (b)

Fig. 1. Mapping areas with mountain - valley division (a), and previous segmentation of valley areas into floristic regions [10] (b)

Conventional (single-) clusterings of the data lead to a segmentation of Switzerland into *floristic regions*. Figure 1 (b) shows a result obtained by agglomerative hierarchical clustering of the valley areas only [10] (thus, the white part of the figure does not correspond to a computed cluster; it comprises areas not included in the clustering).

5.1 Synthetic Data

In order to obtain an initial evaluation of the feasibility of our approach, we first conduct an experiment with synthetic data. For this we constructed two artificial segmentations of Switzerland based on the same mapping regions as in the real data. These segmentations are shown in Figure 2 (top), and henceforth referred to as "vertical" and "horizontal" segmentation, respectively. For each combination of a vertical and a horizontal segment, we defined a species distribution type by a nominal logistic regression model that expresses a preference of the species for the selected vertical and horizontal segment. The logistic regression weights were adjusted so as to obtain conditional probability distributions for the presence of a species of the following form (here showing the case of preference for the first segment in both segmentations):

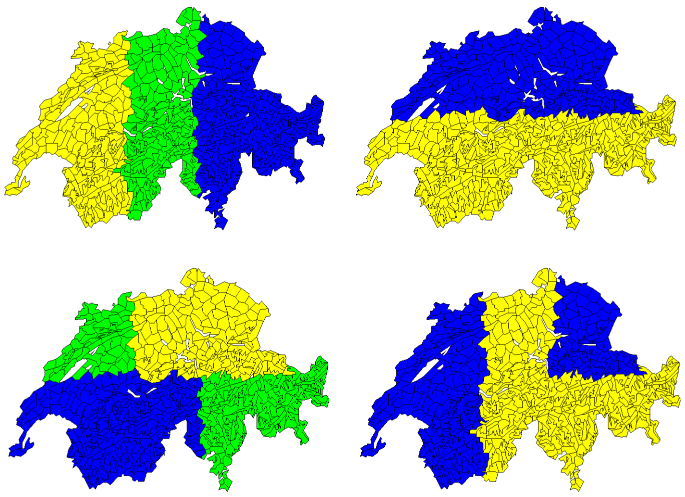


Fig. 2. Artificial segmentations (top); "Wrong" clusterings (bottom)

	Vertical		
Horizontal	yellow	green	blue
blue	0.98	0.5	0.5
yellow	0.5	0.02	0.02

(9)

According to each distribution type we created 15 synthetic plant species, and randomly sampled an occurrence variable for the species at each of the mapping areas.

We then performed FL clustering based on the 90 synthetic species using both the (3o,2o) and the (3n,2n) model (it is not our ambition at this point to detect the "right" number of segmentations and segments per segmentation). In approximately 1 out of 3 random restarts the algorithm terminated with the correct segmentations of Figure 2. In the remaining restarts the algorithm terminated at local optima, a representative example of which is shown in Figure 2 (bottom). However, the correct solutions were identified by a higher likelihood score than that of the wrong solutions.

For comparison, we also performed an experiment where the logistic regression model for $P(\boldsymbol{X} \mid \boldsymbol{L})$ was replaced by a full multinomial model, i.e. for each species we fit a conditional probability table of the form (9) with 6 independent parameters. In this case, almost all restarts terminated with wrong solutions as in Figure 2, and, more importantly, the correct solutions could not be distinguished by a higher likelihood score: in the multinomial model, any pair of segmentations whose combination identifies the 6 different combinations of vertical and horizontal segments achieves the same, optimal, likelihood score.

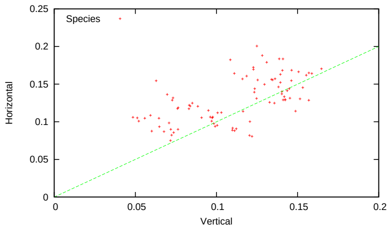
We also use this synthetic data experiment to demonstrate that in FL-clustering there is not necessarily a correlation between clusterings and feature-subsets. Figure 3 (a) shows for each of the 90 synthetic species the mutual information between the species occurrence feature and the two clusterings of Figure 2 (top). The plot shows that there is no strong association of individual species features with one or the other of the two segmentations.

5.2 Real Data

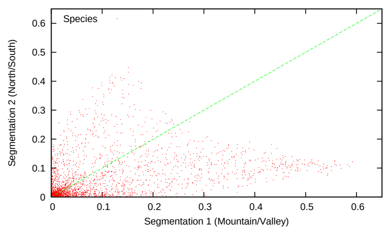
We now perform experiments with the real data consisting of the actual 2398 species. Again, we do not try at this point to automatically detect an appropriate number of segmentations, or segments per segmentation. We run the learning algorithm with a few selected ordinal and nominal logistic models. In all cases we perform 20 runs of the algorithm with different random initializations of the latent variables $\boldsymbol{L}$. The results shown in the following are the segmentations that achieved the highest likelihood score (6) within the 20 restarts.

Figure 4 shows the result of clustering with the (3o,3o) and (3n,3n) logistic models. We use different colors to represents segments computed by nominal logistic models, and greyscale values for ordinal logistic models. The greyscale values then show the ordering of the segments according to their (ordinal) index. One first observes that both models have produced one segmentation in which the mountain areas are identified as one segment: there is an almost perfect correspondence between the mountain attribute illustrated in Figure 1, and the dark grey, respectively yellow, segments in the first segmentations of Figure 4 (note that our method does not entail an ordering of the different segmentations; in particular, in Figure 4 we have just for convenience vertically aligned similar segmentations, and arbitrarily put the ones containing the mountain segment first).

Apart from the mountain/valley attribute our data does not contain "hidden class variables" that could be used for interpreting the segmentations, and therefore one has to look for additional, external data sources, and expert knowledge. As previously mentioned, we expect that the different segmentations to some extent correspond to ecological factors that determine plant growth. A difficulty we now encounter is that many such candidate factors (e.g., average annual tem-



(a)



(b)

Fig. 3. Mutual information: synthetic data (a), real data (b)

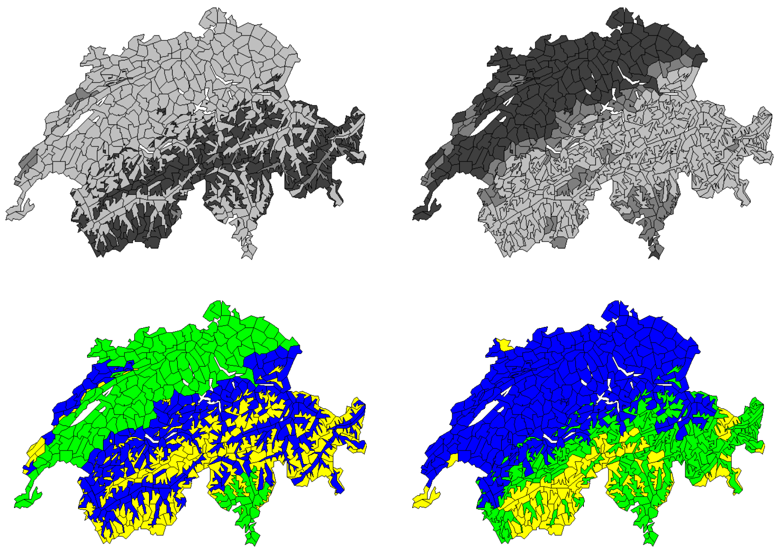


Fig. 4. Clustering result using (3o,3o) (top) and (3n,3n) (bottom) logistic models

perature, average precipitation) are highly correlated with the mountain/valley division, and often show a secondary gradient in north-south direction. The second segmentations of Figure 4 are somewhat dominated by a north-south stratification, and also exhibit some of the patterns visible in Figure 1 (b). However, it seems impossible to identify these north-south segmentations with any particular ecological factor. Instead, it can only be taken as aggregating the north-south dependency of several factors. Moreover, whereas one clustering showing the mountain/valley division was quite consistently produced in the random restarts, there were larger variations observed in the north-south clustering.

Figure 3 shows the mutual information values for the plant features and the (3o,3o) segmentation of Figure 4. This plot shows a relatively strong correlation of some species with the mountain/valley clustering, and a somewhat less pronounced primary correlation of some other species with the north/south segmentation. The large number of species with very small mutual information values for either segmentation is largely made up of species that occur in only a few areas.

In analogy to the experiment with synthetic data, we also perform with the real data an experiment with full multinomial species distribution models instead of the logistic ones. The result is shown in Figure 5. While the mountain/valley pattern is also partly visible in some of the segments, there is no single segment or segmentation corresponding to this division, and the overall segmentation result is clearly less useful than the one obtained with the logistic models.

The poor performance of the multinomial model may in part be due to this model's inability to isolate in its different clusterings several independent explanatory factors, as illustrated by the synthetic data experiments. In addition,

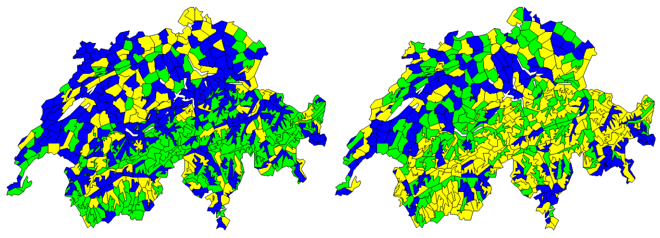


Fig. 5. Clustering result using multinomial $P(\boldsymbol{X} \mid \boldsymbol{L})$

the multinomial model suffered from severe problems of convergence to local optima: even though the global likelihood maximum of the multinomial model must be at least as high as the logistic optimum, the maximum found in 20 restarts was significantly lower for the multinomial than for the logistic models.

The average time consumption of a single run (restart) of (3o,3o) or (3n,3n) clustering was approximately 3 hours, with an average of approximately 15 iterations until convergence. This increased to approximately 6 hours for (3o,3o,3o) or (3n,3n,3n) clusterings. The time is consumed almost entirely in fitting in each iteration the 2398 logistic regression models for all the plants. For comparison, a single run with the multinomial model (taking approximately 8 iterations on average until convergence) takes only about 1 minute, since the multinomial model is easily fit by taking simple counts.

6 Discussion and Future Work

Our experiments have shown that using FL-clustering we can find multiple meaningful clusterings of categorical data. The objective in our approach is explanatory (identify underlying factors that determine the overall data patterns) rather than descriptive (provide the user with multiple views of the data).

For our purpose, it is clearly essential to use a conditional model $P(\boldsymbol{X} \mid \boldsymbol{L})$ of a restricted functional form, rather than an unconstrained multinomial model. Logistic regression models are a canonical choice, and can be seen as a categorical data analogue to the linear mappings between latent and observed dimensions in the factor analysis model.

A common objective in multiple clustering is that different clusterings are in some sense orthogonal or complementary. We are not yet able to say in which sense, or to what extent, FL-clustering satisfies such an objective. Empirically, a bias towards learning complementary clusterings was difficult to verify with our data, since most natural candidate segmentations based on hidden environmental variables would exhibit rather similar patterns (and not at all resemble the segmentations in Figure 2).

Theoretically, one can note that a multi-clustering $\boldsymbol{L} = \boldsymbol{l}$ in which two clusterings are identical can not be a local maximum of the likelihood (6) (except for some degenerate, noise-free, data sets). FL-clustering, thus, is biased away from returning multiple identical clusterings. How this can be strengthened into a formal result linking likelihood gain and complementarity of different clusterings is a subject for future work.

In our experiments we have used data with a spatial structure on the data instances. Within this paper, we have used the spatial structure only for the visualization of the clustering (i.e., segmentation) results. The model can equally be used for other categorical data, and is especially suited for high-dimensional binary data (such as text document data).

On the other hand, our work was also specifically motivated by spatial data, and the relationship in this case of multiple clustering with factorial hidden Markov models [2] and factorial Markov random fields [5]. For spatial data one can impose a Markov random field structure on the latent variables $\boldsymbol{L}$, i.e., the assumption of a uniform distribution for $\boldsymbol{L}$ which we used to derive (6) is replaced, e.g., by the assumption that $P(\boldsymbol{L} = \boldsymbol{l} \mid \theta^*_{\boldsymbol{L}})$ is a Gibbs distribution with fixed parameters $\theta^*_{\boldsymbol{L}}$. Learning in such a setting proceeds in the same way as described in Section 4, only that $P(\boldsymbol{L} = \boldsymbol{l} \mid \theta^*_{\boldsymbol{L}})$ has to be added as a likelihood factor. The optimization in step ii will then usually not be possible precisely, and require an approximate solution. In this paper we did not employ a Markov random field model, since this would usually be used to enforce some smoothness and contiguity properties of the learned segments, which, for our data, seems unwarranted (considering, e.g., the rugged outline of the mountain areas).

In this paper we focused on the core of a probabilistic (multi-) clustering model, i.e., the joint distribution of latent and observable variables. In this model, the number of clusterings, and the number of clusters in each clustering is fixed. We remark, however, that either model selection techniques like BIC or MDL scoring, or a nonparametric Bayesian 'wrapper' around the core model can be used to also learn the model structure.

7 Conclusion

We proposed a latent variable model for multiple clustering of categorical data based on a logistic regression model for the conditional distribution of the observed features. We believe that in analogy to successful techniques for dimensionality reduction, a restricted distributional form for the noisy transformation between the latent and the observed features can be instrumental for revealing relevant patterns in the latent feature space.

For clustering based on a latent variable model we have suggested a simple optimization of the conditional likelihood of the data given the latent variables, with a fixed marginal distribution for the latent variables. This leads to a learning procedure that is linear in the number of observed features, and enables us to experiment with high-dimensional biogeographical data. Our preliminary results from these experiments demonstrate the ability of the method to discover clusterings that represent meaningful explanatory factors for the data. However, further work is needed to consolidate the results returned for this data, and to investigate their potential biological meaning.

References

1. T. Chen, N. Zhang, T. Liu, K. M. Poon, and Y. Wang. Model-based multidimensional clustering of categorical data. *Artificial Intelligence*, 2011. To appear.
2. Z. Ghahramani and M. Jordan. Factorial hidden markov models. *Machine Learning*, 29:245–273, 1997.
3. Y. Guan, J. G. Dy, D. Niu, and Z. Ghahramani. Variational inference for nonparametric multiple clustering. In *KDD10 Workshop on Discovering, Summarizing and Using Multiple Clusterings*, 2010.
4. P. Jain, R. Meka, and I. Dhillon. Simultaneous unsupervised learning of disparate clusterings. In *SIAM Int. Conf. on Data Mining*, pages 858–869, 2008.
5. J. Kim and R. Zabih. Factorial markov random fields. In *Proc. of ECCV 2002*, number 2352 in LNCS, pages 321–334, 2002.
6. D. Niu, J. G. Dy, and M. I. Jordan. Multiple non-redundant spectral clustering views. In *Proc. of the 27th Int. Conf. on Machine Learning (ICML-10)*, 2010.
7. L. Orloci. An agglomerative method for classification of plant communities. *The Journal of Ecology*, 55(1):193–206, 1967.
8. H. Wang, H. Shan, and A. Banerjee. Bayesian cluster ensembles. *Statistical Analysis and Data Mining*, 4(1):54–70, 2011.
9. T. Wohlgemuth. Biogeographical regionalization of switzerland based on floristic data: How many species are needed? *Biodiversity Letters*, 3(6):180–191, 1996.
10. T. Wohlgemuth. Ein floristischer ansatz zur biogeographischen gliederung der schweiz. *Botanica Helvetica*, 106:227–260, 1996.
11. N. L. Zhang. Hierarchical latent class models for cluster analysis. *Journal of Machine Learning Research*, 5:697–723, 2004.
12. N. L. Zhang, Y. Wang, and T. Chen. Discovery of latent structures: Experience with the COIL challenge 2000 data set. *Journal of Systems Science and Complexity*, 21:172–183, 2008.