ERASMUS UNIVERSITEIT ROTTERDAM
ERASMUS SCHOOL OF ECONOMICS

Master Thesis
Business Analytics and Quantitative Marketing
FEM61008

Generalised Convex Clustering: A Parametric Technique to Perform Market Segmentation

Author:
Tyko Kieskamp (409172)

Supervisor:
Dr. A.J. Koning

Second assessor:
Prof. dr. P.J.F. Groenen

Abstract

Market segmentation is a valuable analysis for businesses, especially when they have a diverse clientele. Clustering is the unsupervised learning method that can deal with market segmentation. While many clustering techniques exist, they generally suffer from instabilities and thus do not necessarily generate the best solutions to a problem. A novel method that does not suffer from these instabilities is convex clustering. Although convex clustering is already established for continuous and binary data, this paper extends the literature by generalising convex clustering for different data types. Two practical cases show that the generalised model has a practical use and can provide businesses with vital information. A simulation study furthermore reveals promising results: for categorical data this model outperforms the benchmark of an established method, especially when the relative difference between clusters is small. Finally, it is the first clustering method that is able to analyse Poisson distributed data and its performance in the simulation study is decent. Thus, this paper presents the generalisation for convex clustering for different data types and supports that this extension to the literature is logical and beneficial.

December 27, 2019

Contents

1 Introduction

In 2001, the German car company BMW released "The Hire", a series of short films. The protagonists of these action-packed stories are "The Driver" and his trusted BMW. The pictures were professionally produced by A-list actors and directors and were staggeringly successful; the number of views in the first four months surpassed 11 million. The marketing success also translated to the business. BMW recorded a 13% increase in sales from 2000 to 2001 and 17% in the consecutive year (Boeriu, 2009).

This astonishing accomplishment is due to the implementation of the Segmentation-Targeting-Positioning model. BMW used this model as follows. They first analysed their (potential) customers and segmented them into groups. This included a group of work-hard, play-hard customers: they were married men without children, 46 years old, had an income of around $150,000, and 85% had access to and used the internet – which was uncommon in 2001. This is the group BMW decided to target, since they identified an opportunity for business expansion. To position the BMW brand accordingly, they utilised the knowledge gained from the segmentation and produced high action films which these men would enjoy. The result is that the group started to associate driving a BMW with feelings of excitement. Lastly, BMW distributed the films through their website to effectively target this group, knowing most of them had access to internet (Hespos, 2002).

The example about BMW's "The Hire" shows that segmentation of consumers is a valuable asset for companies. The knowledge gained from discovering types of customers opens up the possibility for effective targeted marketing where the business can position themselves optimally for each type. It furthermore identifies other indispensable information, such as which type of customer is likely to churn, which groups are prone to up-selling and/or cross-selling, and how the company can differentiate their current and future products to fit the wishes of each segment (Kotler et al., 2000).

In practice, businesses have little to no prior knowledge concerning the categories of customers they serve or the distribution of their clientele among those classes. They need a market segmentation analysis to uncover this information. A common data analysis method to achieve such unsupervised learning task is cluster analysis. Clustering takes observations and their corresponding variables as input and provides a hypothesis for a segmentation as output. The general objective of clustering is to put similar observations in the same cluster and dissimilar ones into different clusters. Most methods divide this problem into two aspects. First, the (dis)similarities between the observations are determined and stored. Second, the observations are separated based on the (dis)similarity.

The literature presents many different (dis)similarity metrics and clustering techniques with each their own strategy for defining (dis)similarities and clusters. For surveys about the (dis)similarity metrics and/or clustering methods the reader is referred to Fahad et al. (2014), Xu and Wunsch (2005), Choi et al. (2010), and Boriah et al. (2008). The most common metrics, like the Euclidean norm and Jaccard coefficient, and methods, like K-means and hierarchical clustering, perform well, but they have drawbacks. The metrics are often nonparametric and might not represent the true parametric structure behind the data. The methods themselves are unstable; they are either greedy, implement hard thresholding, solve a non-convex optimisations problem, or a combination of these three. This means that the final result of such methods is not necessarily the best solution possible and can vary due to different initialisation.

A recent development within the field of clustering that does not suffer from these problems is convex clustering. This technique is proposed independently by Hocking et al. (2011) and Lindsten et al. (2011). It shows competitive results compared to older, well established methods. The main idea of convex clustering is to simultaneously define the similarity and separate observations into clusters. This results in a parametric and convex optimisation problem consisting of two parts. The first part determines the similarity between observations by means of the likelihood function of the distribution corresponding to the data. The second part ensures that clusters exist by introducing a penalty term on the parameters from the likelihood function. This penalty function forces pairs of parameters to move to each other and clusters are formed when the parameters merge together.

Until now the convex clustering method is developed for continuous (Hocking et al., 2011) and binary data (Choi and Lee, 2018) by respectively implementing the Gaussian and Bernoulli likelihood function, both members of the exponential family. This leaves room to extend convex clustering to other distributions. Accordingly, the goal of this paper is to determine whether convex clustering is possible for the entire exponential family and devise a general model for convex clustering. This method is called Generalised Convex Clustering (GCC). This extension adds to the literature as it enables convex clustering for different types of data and it allows for the vast amount of knowledge about the exponential family to be implemented on convex clustering. It furthermore shows the added practical benefit of GCC by implementing this method on two real-life cases

The outline of this paper is as follows: section 2 explains the methodology of the technique. It starts with an overview of GCC and how it comprises from the likelihood of the exponential family and a pairwise fused penalty. It furthermore includes the general algorithm to optimise the GCC. Hereafter, section 3 demonstrates applicability of the technique by applying it on two cases with different data types. Section 4 supports the findings with a simulation study and shows the strengths and weakness of the method. Lastly, section 5 concludes the paper.

2 Methodology

This paper proposes a generalisation of convex clustering to the exponential family. Suppose the goal is to cluster the observations of a $n \times d$ data matrix $\boldsymbol{X}$, with $\boldsymbol{X} = [\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n]'$. Here n denotes the number of observations, $\boldsymbol{x}_i$ is a $d \times 1$ column-vector of variables for observation i, and d is the number of variables. Convex clustering assumes that each observation $\boldsymbol{x}_i$ is approximated by the centroid of a cluster, each represented by its parameter $\boldsymbol{\theta}_i$, a $d \times 1$ column-vector. Hence, the target is to find the most optimal $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_n]'$. These centroids are estimated according to this optimisation problem:

$$\min_{\boldsymbol{\Theta}} \; -\ell(\boldsymbol{\Theta}|\boldsymbol{X}) + P_\lambda(\boldsymbol{\Theta}), \tag{1}$$

Here, the goodness-of-fit function $\ell(\boldsymbol{\Theta}|\boldsymbol{X})$ measures the log-likelihood of the centroid describing the observations. The penalty function $P_\lambda(\boldsymbol{\Theta})$ is a Lagrange function, with λ being the Lagrange multiplier. This function penalises the number of unique parameters $\boldsymbol{\theta}_i$ to ensure that centroids merge. Consequently the parameters of the centroids are called fusion parameters, since the fusion of these parameters causes clusters to form.

The idea of convex clustering presented in Hocking et al. (2011) is given by the following constrained optimisation problem:

$$\max_{\boldsymbol{\Theta}} \; \ell(\boldsymbol{\Theta}|\boldsymbol{X}) \quad \text{subject to} \sum_{1 \leq u < v \leq n} 1_{\boldsymbol{\theta}_u \neq \boldsymbol{\theta}_v} \leq c, \tag{2}$$

where 1 is an indicator function, $\sum_{1 \leq u < v \leq n} = \sum_{u=1}^{n-1} \sum_{v=u+1}^{n}$, and c determines the number of unique fusion parameters and therefore the number of clusters.

If c is chosen larger than the amount of pairs of observations, i.e. $c > \frac{n(n-1)}{2}$, the optimisation problem is unconstrained. For such values of c, each $\boldsymbol{\theta}_i$ is set optimally towards $\boldsymbol{x}_i$ given the log-likelihood function. Since the log transformation ensures that the function is monotonic, there is an unique estimation of $\boldsymbol{\theta}_i$ for each unique observation $\boldsymbol{x}_i$. Thus only arbitrary clusters – when data points are equal – are formed. Contrarily if $c = 0$, all $\boldsymbol{\theta}_i$'s are equal to each other and only one cluster exists. The results between those two values of c is of interest, since the clustering occurs then. Suppose c starts at $\frac{n(n-1)}{2}$ and decreases. The allowed number of unique parameters is then reduced. The stronger penalty on the fusion parameters forces certain observations to fuse based on the effect on the goodness-of-fit. Concretely, the clusters for which the loss of goodness-of-fit is lowest after merging are combined. This continues until one cluster remains at $c = 0$.

This seems similar to the popular cluster method Hierarchical Agglomerative Clustering (HAC). In this bottom-up clustering technique, each observations starts in its own cluster and the pair of clusters with the smallest distance for a certain metric are combined. This continues until all observations are merged into one cluster. The first major difference is that HAC is greedy and does not allow for clusters to break-up once they have formed. Convex clustering does allow for this due to its convexness and finds the optimal solution given c. The second difference is that HAC builds upon distance metrics, whereas convex clustering uses the distribution of the data to compare observations.

Following the strategy of Hocking et al. (2011) the indicator constraint is relaxed to a convex penalty function, since indicator functions are difficult to optimise. Moreover, it is easier to minimise instead of maximise and to rewrite the penalty as a Lagrange function. These three adjustments lead to equation (1).

The rest of the methodology is divided into four parts. It starts by delving into the penalty function, after which the likelihood function of the exponential family is discussed. The next section combines these into the GCC and indicates how clusters are specifically determined. Lastly, a general algorithm for optimisation is derived.

2.1 Penalty Function

The penalty function ensures that the fusion parameters move towards each other and form clusters between the observations. The penalty term is implemented as a Lagrange function, which introduces the Lagrange multiplier λ. The size of λ determines the influence of this penalty function and is negatively correlated with the number of clusters found.

Originally however, the indicator function imposes the formation of clusters. The indicator function is difficult to optimise, partly due to not being convex. Although it theoretically ensures the clear definition of clusters, as the name convex clustering suggests, this function is clearly not appropriate.

A solution lies in the 0-norm; the indicator function of two vectors being dissimilar is the same as the 0-norm of the difference between those two vectors (Donoho, 2001):

$$1_{\boldsymbol{\theta}_u \neq \boldsymbol{\theta}_v} = \|\boldsymbol{\theta}_u - \boldsymbol{\theta}_v\|_0. \tag{3}$$

It is common practice in the literature to relax a norm into a higher norm. This leads to the penalty function:

$$P_\lambda^q(\boldsymbol{\Theta}) = \lambda \sum_{1\leq u<v\leq n} \|\boldsymbol{\theta}_u - \boldsymbol{\theta}_v\|_q, \tag{4}$$

where $\| * \|_q$ is the q-norm, with $q \geq 1$. For these values of q the norm is always convex and easier to optimise then the 0-norm. Popular values for q are 1 and 2, respectively called the Manhattan and the Euclidean norm.

This penalty term is related to the pairwise fused Lasso restriction. This shrinkage method is proposed in Petry et al. (2011) and generalises fused Lasso (Tibshirani et al., 2005). Pairwise fused Lasso shrinks all pairs of parameters as well as each parameter individually with a 1-norm:

$$\lambda_1 \sum_{j=1}^{p} |\theta_j| + \lambda_2 \sum_{1\leq j<l\leq p} |\theta_j - \theta_l|, \tag{5}$$

where p the number of variables. Depending on the size of λ_1 and λ_2 respectively, the shrinkage forces parameters to both move towards 0 as well as towards each other. The last part forces a clustering in the parameters. Price et al. (2015) proposes a fused Ridge restriction. The generalisation to pairwise fused Ridge is straightforward and follows that of pairwise fused Lasso.

The pairwise fused Lasso and Ridge originally compare all the pairs of variables. For convex clustering this is not desirable, but rather the pairs of observations. With this adjustment the general form for the penalty function of convex clustering is:

$$P_{\lambda_1,\lambda_2}^q(\boldsymbol{\Theta}) = \lambda_1 \sum_{i=1}^{n} \|\theta_i\|_q + \lambda_2 \sum_{1\leq u<v\leq n} \|\theta_u - \theta_v\|_q. \tag{6}$$

This shows the relation of the penalty term of convex clustering to other fields within statistics.

Setting $\lambda_1 > 0$ offers little in terms of convex clustering. Intuitively the sparsity within the fusion parameters this introduces might increase the interpretability of the results by indicating which variables are important for clustering. This is however not true. The Euclidean norm is inseparable variable-wise and forces the entire fusion parameter for an observation towards 0 simultaneously. The impact of individual parameters is thus indistinguishable from others. Even though the Manhattan norm is separable variable-wise, it does not force an entire column of $\boldsymbol{\Theta}$ towards 0, but instead individual parameters θ_{ij}. Sparsity in specific columns indicates that the variables corresponding to these columns are less important for the clustering, while sparsity in the entire matrix simultaneously does not provide clear proof concerning the importance of variables.

The most appropriate penalty function for convex clustering is thus equation (4). Similar to the indicator function it captures the effect of clustering the observations. The advantage it has is that this function is convex, easing the computational effort. The choice between $q = 1$ and $q = 2$ is important and has substantial consequences. This, together with the choice of λ and the evaluation method, is discussed in the next section.

2.2 Likelihood Function

The exponential family is a group of distributions that follow the same general form of probability distribution. Thus a general method for the entire family covers all individual members. For regressions this lead to the creation of the Generalised Linear Model and this paper proposes an equivalent for convex clustering. The general form of the probability distribution is given by:

$$p(\boldsymbol{x}|\boldsymbol{\theta}) = H(\boldsymbol{x}) \exp\big(\boldsymbol{\theta}' \cdot T(\boldsymbol{x}) - G(\boldsymbol{\theta})\big), \tag{7}$$

where $\boldsymbol{\theta}$ is the natural parameter, also called the canonical parameter. This parameter represents the original parameters, the so-called source parameters, in a different plane. Both the canonical parameter $\boldsymbol{\theta}$ and the data $\boldsymbol{x}$ are column vectors. For some distributions these are the same size, while for others this differs. Since the goal of convex clustering is to form clusters, and not to find the best fit, it suffices to employ $\boldsymbol{\theta}$ and not the source parameters. $T(\boldsymbol{x})$ is the sufficient statistic and $H(\boldsymbol{x})$ is the base measure, which are both only dependent on $\boldsymbol{x}$. Finally, $G(\boldsymbol{\theta})$ is called the log-partition function. This function is a normalisation factor and ensures that the function is in fact a probability and the total probability equals 1: $\int_{-\infty}^{\infty} p(\boldsymbol{x}|\boldsymbol{\theta}) = 1$ (Jordan, 2009).

The likelihood function of the exponential family is the product of the probability distribution over all observations in the corresponding dataset:

$$\begin{aligned} L(\boldsymbol{\Theta}|\boldsymbol{X}) &= \prod_{i=1}^{n} p(\boldsymbol{x}_i|\boldsymbol{\theta}_i) \\ &= \prod_{i=1}^{n} H(\boldsymbol{x}_i) \exp\big(\boldsymbol{\theta}_i' \cdot T(\boldsymbol{x}_i) - G(\boldsymbol{\theta}_i)\big), \end{aligned} \tag{8}$$

where $\boldsymbol{\Theta} = [\boldsymbol{\theta}_1, \boldsymbol{\theta}_2, \ldots, \boldsymbol{\theta}_n]'$. The base measure $H(\boldsymbol{x}_i)$ depends only on $\boldsymbol{x}_i$ and is therefore a constant with respect to $\boldsymbol{\theta}_i$. Thus, this part of the function is ignored for the optimisation of the likelihood function towards the natural parameter:

$$L(\boldsymbol{\Theta}|\boldsymbol{X}) \propto \prod_{i=1}^{n} \exp\big(\boldsymbol{\theta}_i' \cdot T(\boldsymbol{x}_i) - G(\boldsymbol{\theta}_i)\big). \tag{9}$$

The log-likelihood is however simpler to optimise:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{X}) \propto \sum_{i=1}^{n} \boldsymbol{\theta}_i' \cdot T(\boldsymbol{x}_i) - G(\boldsymbol{\theta}_i). \tag{10}$$

It is additionally convex for the entire exponential family and differentiable on a interval corresponding to the datatype and parameters.

Due to the convexness the general log-likelihood function can be inserted into equation (1) without direct problems. All members of the exponential family can thus be implemented for convex clustering. It clearly follows that the natural parameter of the exponential family is the fusion parameter for convex clustering.

The exponential family contains several common distributions, notably the Gaussian, Bernoulli, Poisson, and categorical distribution. For a comprehensive overview of the entire family and the decomposition for each distribution the reader is referred to Nielsen and Garcia (2009). The rest of this section displays the likelihood function of the four distributions mentioned earlier. The derivation of the log-likelihood function of the Gaussian and Bernoulli distribution shows the relation of the GCC to respectively Hocking et al. (2011) and Choi and Lee (2018). The log-likelihoods of the Poisson and categorical distribution are referred to later on in the application and simulation section of this paper to show the capabilities of GCC.

2.2.1 Gaussian distribution: The Gaussian or normal distribution is commonly used for continuous data and is a staple aspect in many models and techniques. The Gaussian distribution has two source parameters: one location and one scale parameter. The scale parameter, the variance-covariance matrix, is set equal to the identity matrix for simplicity. The decomposition of the Gaussian distribution is then given as:

$$\begin{aligned} \boldsymbol{\theta} &= \boldsymbol{\mu}, \\ T(\boldsymbol{x}) &= \boldsymbol{x}, \\ G(\boldsymbol{\theta}) &= \frac{1}{2}\boldsymbol{\theta}'\boldsymbol{\theta} + \frac{d}{2}\log(2\pi), \end{aligned} \tag{11}$$

where $\boldsymbol{\mu}$ is the source parameter for the location. As aforementioned, it suffices to optimise the natural parameter and not the source parameter. In this case the substitution of $\boldsymbol{\theta}$ with $\boldsymbol{\mu}$ does not change the function in any meaningful manner, however for consistency the natural parameter is kept.

The log-likelihood function of the Gaussian distribution is proportional to:

$$\begin{aligned}
\ell(\boldsymbol{\Theta}|\boldsymbol{X}) &\propto \sum_{i=1}^{n} \boldsymbol{\theta}_i'\boldsymbol{x}_i - \frac{1}{2}\boldsymbol{\theta}_i'\boldsymbol{\theta}_i - \frac{d}{2}\log(2\pi) \\
&\propto \sum_{i=1}^{n} \boldsymbol{\theta}_i'\boldsymbol{x}_i - \frac{1}{2}\boldsymbol{\theta}_i'\boldsymbol{\theta}_i \\
&\propto \sum_{i=1}^{n} \boldsymbol{\theta}_i'\boldsymbol{x}_i - \frac{1}{2}\boldsymbol{\theta}_i'\boldsymbol{\theta}_i - \frac{1}{2}\boldsymbol{x}_i'\boldsymbol{x}_i \\
&= \sum_{i=1}^{n} -\frac{1}{2}(\boldsymbol{x}_i - \boldsymbol{\theta}_i)'(\boldsymbol{x}_i - \boldsymbol{\theta}_i) \\
&= -\frac{1}{2}\|\boldsymbol{X} - \boldsymbol{\Theta}\|_F^2,
\end{aligned} \tag{12}$$

where $\|\cdot\|_F$ is the Frobenius norm. This is the same as the function used in the optimisation problem proposed in Hocking et al. (2011), clearly showing that the paper used a Gaussian distribution with an identity variance-covariance matrix as the goodness-of-fit measure for convex clustering.

2.2.2 Bernoulli distribution:

2.2.2 Bernoulli distribution: The next example is the Bernoulli distribution. This distribution is used to represent a situation where there are two options, i.e. binary choices. Choi and Lee (2018) implements it for the convex clustering of binary data. The decomposition of the Bernoulli distribution is:

$$\begin{aligned}
\theta &= \log\left(\frac{p}{1-p}\right), \\
T(x) &= x, \\
G(\theta) &= \log\big(1 + exp(\theta)\big),
\end{aligned} \tag{13}$$

where p is the probability of the value of x being equal to 1. The log-likelihood of the Bernoulli distribution over d variables is given by:

$$\begin{aligned}
\ell(\boldsymbol{\Theta}|\boldsymbol{X}) &\propto \sum_{i=1}^{n}\sum_{j=1}^{d} \theta_{ij}x_{ij} - \log\big(1 + \exp(\theta_{ij})\big) \\
&= \sum_{i=1}^{n} \left(\boldsymbol{\theta}_i'\boldsymbol{x}_i - \sum_{j=1}^{d} \log\big(1 + \exp(\theta_{ij})\big)\right) \\
&= \sum_{i=1}^{n} \Big(\boldsymbol{\theta}_i'\boldsymbol{x}_i - s_i\Big).
\end{aligned} \tag{14}$$

This derivation is different than in Choi and Lee (2018), but the output of the log-likelihood is the same.

2.2.3 Poisson distribution: The Poisson distribution describes data with a high number of trials, where for each trial the chance of the event of interest to happen is small. The input is the count of an event during a specific period and the source parameter ζ indicates the expected number of occurrences during that period.

The decomposition of the Poisson distribution is:

$$\begin{aligned} \theta &= \log \zeta, \\ T(x) &= x, \\ G(\theta) &= \exp(\theta). \end{aligned} \tag{15}$$

Here, x_{it} indicates the discrete number of occurrences of the event for individual i during time period t. The canonical parameter θ_i is the same for each time period t. This is due to the fact that the source parameter describes a series of number of occurrences and not a single instance. The log-likelihood of the Poisson distribution thus is:

$$\begin{aligned} \ell(\boldsymbol{\theta}|\boldsymbol{X}) &\propto \sum_{i=1}^{n}\sum_{t=1}^{\psi}\Big(\theta_i x_{it} - \exp(\theta_i)\Big) \\ &\propto \sum_{i=1}^{n}\Big(\theta_i \boldsymbol{\imath}'\boldsymbol{x}_i - \psi\exp(\theta_i)\Big), \end{aligned} \tag{16}$$

where ψ indicates the total number of time periods, $\boldsymbol{\imath}$ is a $\psi \times 1$ column-vector of ones, and $\boldsymbol{x}_i = [x_{i1}, x_{i2}, \ldots, x_{i\psi}]'$.

2.2.4 Categorical distribution: The categorical distribution represents the distribution of categorical data which take K different classes in a single trial. This is the Bernoulli distribution when $K = 2$ and the multinomial distribution when the trials is more than one.

The decomposition of the categorical distribution for the exponential family is:

$$\begin{aligned} \theta^k &= \log\left(\frac{p^k}{1-\sum_{l=1}^{K-1}p^l}\right), \\ T(x^k) &= x^k, \\ G(\theta^k) &= \log\left(1+\sum_{l=1}^{K-1} exp(\theta^l)\right), \end{aligned} \tag{17}$$

where θ^k is the canonical parameter for class k and p^k is the probability of the random variable being equal to class k. The data point x is represented by the $K \times 1$ column dummy vector $[x^1, x^2, \ldots, x^K]'$, where $x^k = 1$ if $x = k$ and 0 otherwise. Furthermore, $p^K = 1 - \sum_{l=1}^{K-1} p^l$ to ensure identification and $\theta^K = 0$ accordingly (Nielsen and Garcia, 2009). Thus $\boldsymbol{\theta}$ is a $K-1\times 1$ column vector: $[\theta^1, \theta^2, \ldots, \theta^{K-1}]'$

In most cases it is incorrect to assume that each variable has the same number of categories K. To resolve this problem, for each variable j the number of classes is K_j. The log-likelihood of the categorical distribution over d variables is given by:

$$\ell(\boldsymbol{\Theta}_1, \ldots, \boldsymbol{\Theta}_n|\boldsymbol{X}_1, \ldots, \boldsymbol{X}_n) \propto \sum_{i=1}^{n}\sum_{j=1}^{d}\left(\sum_{k=1}^{K_j-1}(\theta_{ij}^k x_{ij}^k) - \log\left(1+\sum_{l=1}^{K_j-1} exp(\theta_{ij}^l)\right)\right), \tag{18}$$

with $\boldsymbol{X}_i = [\boldsymbol{x}_{i1}, \boldsymbol{x}_{i2}, \ldots, \boldsymbol{x}_{id}]'$, $\boldsymbol{x}_{ij} = [x_{ij}^1, x_{ij}^2, \ldots, x_{ij}^{K_j-1}]'$, $\boldsymbol{\Theta}_i = [\boldsymbol{\theta}_{i1}, \boldsymbol{\theta}_{i2}, \ldots, \boldsymbol{\theta}_{id}]'$, and $\boldsymbol{\theta}_{ij} = [\theta_{ij}^1, \theta_{ij}^2, \ldots, \theta_{ij}^{K_j-1}]'$

It is possible to vectorise the log-likelihood of the categorical distribution by implementing the following adjustments. First, the matrix $\boldsymbol{\Theta}_i$ is written as a column vector: $\boldsymbol{\theta}_i = [\theta_{i1}^1, \theta_{i1}^2, \ldots \ldots, \theta_{i1}^{K_1-1}, \theta_{i2}^1, \theta_{i2}^2, \ldots, \theta_{id}^{K_d-1}]'$ and the same for $\boldsymbol{X}_i$ into column vector $\boldsymbol{x}_i$. The size of $\boldsymbol{\theta}_i$ is denoted as $dK \times 1$, with $dK = \sum_{j=1}^{d}(K_j - 1)$. Secondly, the variable z_{ij} is introduced, which captures the log-partition: $z_{ij} = \log\left(1 + \sum_{l=1}^{K_j-1} \exp(\theta_{ij}^l)\right)$. It is important to note that this variable does not change given the value of k, only due to i and j. Substituting these adjustments into the log-likelihood results in:

$$\ell(\boldsymbol{\Theta}|\boldsymbol{X}) \propto \sum_{i=1}^{n} \Bigg(\boldsymbol{\theta}_i' \boldsymbol{x}_i - \boldsymbol{z}_i' \boldsymbol{\imath} \Bigg), \tag{19}$$

where $\boldsymbol{z}_i = [z_{i1}, z_{i2}, \ldots z_{id}]'$ and $\boldsymbol{\imath}$ is a $d \times 1$ column-vector of ones.

2.3 Convex Clustering

As aforementioned, the optimisation problem for GCC consists of two parts: the likelihood, which serves as a goodness-of-fit to define the similarity between data points, and the penalty function, which ensures that clusters are formed. The general problem is:

$$\min_{\boldsymbol{\Theta}} \; -\ell(\boldsymbol{\Theta}|\boldsymbol{X}) + P_\lambda(\boldsymbol{\Theta}), \tag{20}$$

with $\boldsymbol{X}$ a $n \times d$ data matrix and $\boldsymbol{\Theta}$ a $n \times d$ parameter matrix.

Inserting the functions presented in section 2.2 and section 2.1 results in the optimisation problem for GCC:

$$\min_{\boldsymbol{\Theta}} \; -\sum_{i=1}^{n} \Bigg(\boldsymbol{\theta}_i' \cdot T(\boldsymbol{x}_i) - G(\boldsymbol{\theta}_i) \Bigg) + \lambda \sum_{1 \le u < v \le n} \|\boldsymbol{\theta}_u - \boldsymbol{\theta}_v\|_q. \tag{21}$$

$T(\boldsymbol{x}_i)$ and $G(\boldsymbol{\theta}_i)$ both depend on the specific distribution suited for the data.

In the penalty function of this optimisation problem the absolute influence of each variable j is equal. If there is a difference between the scaling of the variables in the data this scaling might translate to $\boldsymbol{\Theta}$ through the sufficient statistic $T(\boldsymbol{x}_i)$. Then for some variables the corresponding fusion parameters are relatively larger than those of other variables. The penalty function penalises variables with a relative large theta stronger than those with a relatively small theta. Thus the former variables are more important for the clustering than the latter. A solution to this problem is to scale the sufficient statistic, so that the relative size of theta is similar. Some variables can then still be more important for the clustering, however the effect of different size is then eliminated. Another solution is by implementing PCA as done in Choi and Lee (2018), where the difference in scaling is captured in the dimension reduction. Researching whether this problem is relevant and if the two solutions are viable is beyond the scope of this paper.

The previous section explained that there are multiple choices for the norm of the penalty function, most notably $q = 1$ and $q = 2$. This choice is important and has substantial consequences. This, together with the choice of λ and the evaluation method, is discussed in the remainder of this section.

2.3.1 The choice of the penalty norm: The norm of the penalty function has implications for the results of the convex clustering. While any choice of $q \in \mathbb{Z}^+$ shows to result in well-defined clustering for the Gaussian distribution (Hocking et al., 2011), this does not hold for the Bernoulli distribution. Choi and Lee (2018) proves that when the penalty function is separable variable-wise there are only two possible outcomes: either one cluster or only arbitrary clusters. The intuition of this proof is that in case of a variable-wise separable penalty where more than 1 clusters exist, a parameter θ_{ij} is determined independently from the other variables. θ_{ij} then takes only two values for all i, one for $x_{ij} = 1$ and one for $x_{ij} = 0$. This means that the vector $\boldsymbol{\theta}_i$ is unique to each unique $\boldsymbol{x}_i$ and there is no clustering. Generally, a variable-wise separable penalty function creates an all-or-nothing clustering for the Bernoulli distribution.

Needless to say, this result is not desired. This proof is given for a binary case, however for the multiclass case the proof is similar and the same results hold. The multiclass case is part of the exponential family in the form of the categorical and multinomial distribution. The exponential family is therefore split on this. A general proof for which members of the family this holds is beyond the scope of this paper.

A solution to the all-or-nothing clustering is to implement penalty terms which are inseparable per variable. Therefore, the 1-norm is not suitable, but any norm for which $q = 1 + \epsilon$, with $\epsilon > 0$ is. Although other norms are possible, the Euclidean norm is popular, intuitive, and straightforward to optimise. Thus, $q = 2$ for the rest of this paper.

2.3.2 The choice of the penalty parameter: The question of the optimal number of clusters and consequently the optimal value for λ still remains. There are several different ways to answer this question. Obviously, the most desired situation is that the number of clusters is known a priori. Due to the unsupervised nature of clustering this is rarely true. It is however possible to determine the optimal number of clusters not beforehand, but rather on the outcomes of the clustering. For example, the segmentation of the customers of a company into 6 groups does not give enough information to personalise the products towards, but the company also does not have the capabilities to adjust towards 15 segments. In this case 10 clusters might be preferred. While this method is practical, there is no theoretical justification for the chosen number of clusters and incorrect results could be derived from this.

The theoretically supported approach is to analyse the validity of the clustering. There are two types of validity for clustering: internal validity and external validity. Internal validity refers to assessing the clustering results using only the information necessary to create the clusters. External validity on the other hand utilises knowledge about the actual clusters of the data to validate the results. Thus, external validation is not suited as a general method to choose the optimal value of λ. It is however useful to evaluate the performance of the clustering technique when the true clustering is known.

Several internal validity indices exist in the literature, including the Davies-Bouldin, silhoutte, and Dunn indices. These indices support the clustering methods which build upon distance metrics (Rendón et al., 2011). Convex clustering is devised as an alternative to those methods and implements distributions to avoid using distance metrics. Hence, these indices are unsuitable to use as a validation method. The Akaike Information Criterion (AIC) does not use the distances, but exploits the likelihood and the degrees of freedom instead. Originally AIC is designed to assist with overfitting and to estimate the dimensions of a model (Akaike, 1973). The AIC value is defined as:

$$\text{AIC} = -2\ell(\,\boldsymbol{\cdot}\,) + 2 \cdot m, \tag{22}$$

where $\ell(\,\boldsymbol{\cdot}\,)$ is the log-likelihood dependent on certain parameter(s) and m is the number of parameters estimated by the model. This equation takes both the model fit as well as the model complexity into account. Minimising AIC is equivalent to finding the best trade-off between a good fit – a high $\ell(\,\boldsymbol{\cdot}\,)$ – and a low complexity – a low m. The strategy to find the best model is thus to estimate multiple models with varying parameters and to select the model with the smallest AIC value. Several papers however discussed that AIC generally chooses a higher dimension than the "true model".

An related alternative that does find the "true model" is the Bayesian Information Criterion (BIC), proposed in (Schwarz et al., 1978). The BIC value for a model is defined by:

$$\text{BIC} = -2\ell(\,\boldsymbol{\cdot}\,) + \log(n) \cdot m, \tag{23}$$

where n is the number of observations. Similar to AIC minimising BIC is finding the best trade-off between a good fit and a low complexity, and thus the model with the smallest BIC is considered best.

AIC and BIC are the two most commonly used information criterion, however there are many more criteria available. Akogul and Erişoğlu (2016) compared several of these information criterion on their performance for clustering using mixture models. Unsurprisingly, this study showed that BIC outperformed AIC. However their results show that the Kullback information criterion (KIC) outshines BIC. This KIC is first proposed in Cavanaugh (1999) and is specifically designed for large-sample models. In those models it outperforms AIC, but it also suffers from the problem of choosing a higher dimension than the "true model" in small-sample models. The KIC value is given by:

$$\text{KIC} = -2\ell(\,\boldsymbol{\cdot}\,) + 3 \cdot (m - 1). \tag{24}$$

The model with the smallest KIC is the most optimal trade-off between a good fit and a low complexity.

In convex clustering each observation is defined by its own parameter and the goal is to reduce the number of unique parameters, where each unique parameter is a cluster. The number of unique parameters times the number of variables determines the complexity of the model, instead of the number of variables for regressions. Determining the optimal number of unique parameters for convex clustering is thus equivalent to determining which variables to use for regression. It is therefore not far fetched to implement an information criterion to assist with determining the number of clusters. Since BIC is the most commonly used information criterion and seems to perform well in the other papers about convex clustering, the models in the application section use BIC. In the simulation section the estimated number of clusters of the three criteria are compared to the true number of clusters to figure out which criterion is preferred.

With adjustments towards convex clustering the information criteria are:

$$\begin{aligned}
\text{AIC} &= -2\ell(\boldsymbol{\Theta}|\boldsymbol{X}) + 2 \cdot |C| \cdot d, \\
\text{BIC} &= -2\ell(\boldsymbol{\Theta}|\boldsymbol{X}) + \log(n) \cdot |C| \cdot d, \\
\text{KIC} &= -2\ell(\boldsymbol{\Theta}|\boldsymbol{X}) + 3 \cdot (|C| \cdot d - 1),
\end{aligned} \tag{25}$$

where $|C|$ indicates the number of clusters and d the number of variables.

2.3.3 Evaluation of the model: Given a choice for q and λ the output of the model is a clustering of the observations. A problem that is common for unsupervised learning, is that the correct solution for the observations are normally not known and it is not possible to asses the performance the model properly. There are, however, predefined datasets set up for unsupervised learning for which the outcomes are known. Another method to bypass this issue is by simulating the data. Two popular clustering evaluation measures for when external validation is possible are the F_1 score and entropy (Rendón et al., 2011).

The F_1 score combines the precision and recall into a score for each cluster. The F_1 for a specific cluster c is:

$$F_1^c = 2 \cdot \frac{precision_c \cdot recall_c}{precision_c + recall_c}, \tag{26}$$

where $precision_c$ and $recall_c$ are respectively the precision and recall of the model in cluster c versus all other clusters. These scores can then be summarised into a weighted F_1 score:

$$F_1 = \sum_{c \in C} \frac{|c|}{n} \cdot F_1^c, \tag{27}$$

where C is defined as the set of clusters. The cardinality $|c|$ indicates the number of observations in class c. The range of the F_1 score is $[0, 1]$. $F_1 = 1$ is achieved when $precision_c = recall_c = 1$ for all values of c, indicating a perfect clustering; a higher score indicates a better performance.

Entropy has its basis in the information theory, which was first introduced by Shannon (1948). Similar to the F_1 score, the entropy can be calculated as a weighted measure as well as for individual clusters. The entropy for a single cluster c is defined as:

$$\begin{aligned} E(c) &= -\sum_{q \in Q} p_c(q) \log\big(p_c(q)\big), \\ \text{with} \qquad p_c(q) &= \frac{|c_q|}{|c|}, \end{aligned} \tag{28}$$

where q indicates an actual cluster, Q the set of actual clusters, and $|c_q|$ the number of observations belonging to q in cluster c. $\log\big(p_c(q)\big)$ is not defined for the case that $p_c(q) = 0$. In that situation $p_c(q)\log_2\big(p_c(q)\big) = 0$, since

$$\lim_{p_c(q) \to 0^+} p_c(q) \log\big(p_c(q)\big) = 0. \tag{29}$$

The weighted entropy E is deduced similarly from the individual $E(c)$ as the weighted F_1 score

$$E = \sum_{c \in C} \frac{|c|}{n} \cdot E(c). \tag{30}$$

The range of the entropy is $[0, \log(n)]$. If $E = 0$ the clustering is done perfectly, since then $p_c(q) = 1$ or $p_c(q) = 0$ for all c and q. This means that all observations in a cluster c are exclusively from a specific actual cluster q. Therefore, a lower value for the entropy indicates a higher clustering performance.

Both the weighted F_1 score and the weighted entropy give an indication of the clustering quality if the true clusters are known. As mentioned earlier this is rarely true in practice. Hence, this paper includes a simulation study for which these measure can be derived.

2.4 Optimisation

Newton's method is popular estimation tool and is suited to optimise Generalised Convex Clustering. This method comprises of the gradient and the Hessian. Both of these are well known for the log-likelihood function of the exponential family. Newton's method minimises the function:

$$\begin{aligned} F(\boldsymbol{\Theta}|\boldsymbol{X}) &= -\ell(\boldsymbol{\Theta}|\boldsymbol{X}) + P_\lambda(\boldsymbol{\Theta}) \\ &= -\sum_{i=1}^{n}\Big(\boldsymbol{\theta}_i' \cdot T(\boldsymbol{x}_i) - G(\boldsymbol{\theta}_i)\Big) + \lambda \sum_{1\le u<v\le n} \|\boldsymbol{\theta}_u - \boldsymbol{\theta}_v\|_2, \end{aligned} \tag{31}$$

which is equivalent to the optimisation problem given by equation (21). Newton's method is an iterative algorithm which starts with an initial estimate for the parameters $\boldsymbol{\Theta}$ and at each step r of the procedure updates the parameters:

$$\boldsymbol{\theta}_i^r = \boldsymbol{\theta}_i^{r-1} - \mathrm{H}(\boldsymbol{\theta}_i^{r-1})^{-1}\nabla(\boldsymbol{\theta}_i^{r-1}), \tag{32}$$

with

$$\begin{aligned} \nabla(\boldsymbol{\theta}_i) &= \frac{\delta F}{\delta \boldsymbol{\theta}_i} = -\frac{\delta \ell}{\delta \boldsymbol{\theta}_i} + \frac{\delta P_\lambda}{\delta \boldsymbol{\theta}_i}, \\ \mathrm{H}(\boldsymbol{\theta}_i) &= \frac{\delta^2 F}{\delta \boldsymbol{\theta}_i \delta \boldsymbol{\theta}_i'} = -\frac{\delta^2 \ell}{\delta \boldsymbol{\theta}_i \delta \boldsymbol{\theta}_i'} + \frac{\delta^2 P_\lambda}{\delta \boldsymbol{\theta}_i \delta \boldsymbol{\theta}_i'}. \end{aligned} \tag{33}$$

The gradient and the Hessian of the log-likelihood is given by:

$$\begin{aligned} \frac{\delta \ell}{\delta \boldsymbol{\theta}_i} &= T(\boldsymbol{x}_i) - \frac{\delta G}{\delta \boldsymbol{\theta}_i}, \\ \frac{\delta^2 \ell}{\delta \boldsymbol{\theta}_i \delta \boldsymbol{\theta}_i'} &= \frac{\delta^2 G}{\delta \boldsymbol{\theta}_i \delta \boldsymbol{\theta}_i'}, \end{aligned} \tag{34}$$

where gradient and the Hessian of the log-partition function G depends on the distribution. Moreover, the gradient and the Hessian of the 2-norm penalty function is given by:

$$\begin{aligned} \frac{\delta P_\lambda}{\delta \boldsymbol{\theta}_i} &= \lambda \sum_{v\neq i} \frac{\boldsymbol{\theta}_i - \boldsymbol{\theta}_v}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2}, \\ \frac{\delta^2 P_\lambda}{\delta \boldsymbol{\theta}_i \delta \boldsymbol{\theta}_i'} &= \lambda \sum_{v\neq i} \left(\frac{\boldsymbol{I}}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2} - \frac{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)'}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2^3} \right). \end{aligned} \tag{35}$$

The comprehensive derivation of the penalty function is illustrated in appendix A.

From equation (35) it is apparent that the derivative of the 2-norm penalty function is not defined for i when $\boldsymbol{\theta}_i = \boldsymbol{\theta}_v$. This occurs when $i = v$ and when i and v are in the same cluster. The former situation does not occur, but the latter does. To ensure that the derivative exists, it is necessary to merge observations in the same cluster. As aforementioned, every observation is assumed to be its own cluster before any merger occurs. When two clusters c_1 and c_2 are fused they continue from that point onward as a single cluster c with the same parameter $\boldsymbol{\theta}_c$. Since the penalty term is the 2-norm, the fusion parameters are rarely exactly the same. The two clusters are considered fused when $\|\boldsymbol{\theta}_{c_1} - \boldsymbol{\theta}_{c_2}\|_2 < \tau$, where τ is a small threshold. The choice of $\boldsymbol{\theta}_c$ is arbitrary for this new cluster. While a random $\boldsymbol{\theta}_c$ would suffice, the computation time is less by using a $\boldsymbol{\theta}_c$ close to $\boldsymbol{\theta}_{c_1}$ and $\boldsymbol{\theta}_{c_2}$. Since the difference between the two parameters is small, any parameter between the two is a valid initial parameter for the new cluster. In this paper the fusion parameter of the new cluster c is determined by the weighted mean of c_1 and c_2:

$$\boldsymbol{\theta}_c = \frac{|c_1|\boldsymbol{\theta}_{c_1} + |c_2|\boldsymbol{\theta}_{c_2}}{|c_1| + |c_2|}, \tag{36}$$

where $|c|$ is the number of observations in cluster c.

Besides merging the parameters the data points also need to be combined. An intuitive way to do this is to take the mean of $\boldsymbol{x}_i$ for all i in the cluster. $\boldsymbol{X}$ does not directly influence the log-likelihood function though, but rather through the sufficient statistic. Hence, the sufficient statistics of cluster c is the average of the sufficient statistic of the observations within this cluster and is either calculated directly or as a combination of its subclusters c_1 and c_2 which form c together:

$$\begin{aligned} \overline{T}_c &= \frac{1}{|c|}\sum_{i\in c} T(\boldsymbol{x}_i), \\ \overline{T}_c &= \frac{|c_1|\overline{T}_{c_1} + |c_2|\overline{T}_{c_2}}{|c_1| + |c_2|}, \end{aligned} \tag{37}$$

The update of the fusion parameters of cluster c at step r is thus given by:

$$\boldsymbol{\theta}_c^r = \boldsymbol{\theta}_c^{r-1} - \mathrm{H}(\boldsymbol{\theta}_c^{r-1})^{-1}\nabla(\boldsymbol{\theta}_c^{r-1}), \tag{38}$$

with

$$\begin{aligned} \nabla(\boldsymbol{\theta}_c) &= -\overline{T}_c + \frac{\delta G}{\delta \boldsymbol{\theta}_c} + \lambda \sum_{v\neq c} \frac{\boldsymbol{\theta}_c - \boldsymbol{\theta}_v}{\|\boldsymbol{\theta}_c - \boldsymbol{\theta}_v\|_2}, \\ \mathrm{H}(\boldsymbol{\theta}_c) &= \frac{\delta^2 G}{\delta \boldsymbol{\theta}_c \delta \boldsymbol{\theta}_c'} + \lambda \sum_{v\neq c}\left(\frac{\boldsymbol{I}}{\|\boldsymbol{\theta}_c - \boldsymbol{\theta}_v\|_2} - \frac{(\boldsymbol{\theta}_c - \boldsymbol{\theta}_v)(\boldsymbol{\theta}_c - \boldsymbol{\theta}_v)'}{\|\boldsymbol{\theta}_c - \boldsymbol{\theta}_v\|_2^3}\right). \end{aligned} \tag{39}$$

Algorithm 1 displays the pseudo-code for the minimisation of the optimisation problem of general convex clustering given a value of λ, a threshold for clustering τ and the maximum number of iterations Ω.

For parameter tuning algorithm 1 is run for different values of λ and for each the value for the information criterion is determined. The clustering corresponding to the smallest information criterion is considered best and is the final result of GCC of the dataset.

Since the application and the simulation study utilise the GCC optimisation of the Poisson and categorical distribution, the derivation of the average sufficient statistic and the gradient and the Hessian of the log-partition function is shown below for both distributions. The sufficient statistics and log-partition functions are displayed earlier in subsections 2.2.3 and 2.2.4 for respectively the Poisson and categorical distribution.

Algorithm 1 GCC optimisation for given λ

```
1: Initialise all $\boldsymbol{\theta}_i^0$ with random values
2: Set each observation $i$ in cluster $c_i$
3: Define the set of clusters $C = \{\{c_1\}, \{c_2\}, ..., \{c_n\}\}$
4: Set $r = 1$
5: repeat
6:     for all clusters $c \in C$ do
7:         Obtain $\nabla(\boldsymbol{\theta}_c^{r-1})$ and $\mathrm{H}(\boldsymbol{\theta}_c^{r-1})$ using equation (39)
8:         Update $\boldsymbol{\theta}_c^r$ using equation (38)
9:     end for
10:    for all pairs of clusters $c_u, c_v \in C$, with $u \neq v$ do
11:        if $\|\boldsymbol{\theta}_{c_u}^r - \boldsymbol{\theta}_{c_v}^r\|_2 < \tau$ then
12:            Fuse $c_u$ and $c_v$ according to equation (36) and equation (37)
13:        end if
14:    end for
15:    r = r + 1
16: until all values of $\theta_c$ converge, $r > \Omega$ or $|C| = 1$
17: return $C$
```

2.4.1 Poisson distribution: The sufficient statistic for the Poisson distribution of cluster c is:

$$\overline{T}_c = \frac{1}{|c|}\sum_{i\in c} T(\boldsymbol{x}_i) = \frac{1}{|c|}\sum_{i\in c} \boldsymbol{\imath}'\boldsymbol{x}_i. \tag{40}$$

The output of sufficient statistic is now no longer necessarily discrete as it is originally. This does not disrupt the optimisation of the log-likelihood though and does not affect the clustering negatively.

The gradient of the log-partition function is:

$$\frac{\delta G}{\delta \theta_c} = G'(\theta_c) = \psi \exp(\theta_c) \tag{41}$$

and the Hessian is:

$$\frac{\delta^2 G}{\delta\theta_c \delta\theta_c'} = G''(\theta_c) = \psi \exp(\theta_c). \tag{42}$$

2.4.2 Categorical distribution: For the sufficient statistic of the categorical distribution it is clear that:

$$\overline{T}_c = \frac{1}{|c|}\sum_{i\in c} T(\boldsymbol{x}_i) = \frac{1}{|c|}\sum_{i\in c} \boldsymbol{x}_i. \tag{43}$$

Similar to the Poisson distribution, the resulting sufficient statistic is no longer binary. This does not matter for the optimisation problem though and the optimal clustering is still formed.

The gradient of the log-partition function is:

$$\frac{\delta G}{\delta \boldsymbol{\theta}_c} = \boldsymbol{s}_c = [\boldsymbol{s}_{i1}^1, \boldsymbol{s}_{i1}^2, \ldots, \boldsymbol{s}_{i1}^{K_1-1}, \boldsymbol{s}_{i2}^1, \boldsymbol{s}_{i2}^2, \ldots, \boldsymbol{s}_{id}^{K_d-1}]', \tag{44}$$

with $s_{cj}^k = \frac{\exp(\theta_{cj}^k)}{1+\sum_{l=1}^{K_j-1}\exp(\theta_{cj}^l)}$ and the Hessian is a diagonal block matrix:

$$\frac{\delta^2 G}{\delta \boldsymbol{\theta}_c \delta \boldsymbol{\theta}_c'} = \begin{bmatrix} \operatorname{diag}(\boldsymbol{s}_{c1}) - \boldsymbol{s}_{c1}'\boldsymbol{s}_{c1} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{O} & \operatorname{diag}(\boldsymbol{s}_{c2}) - \boldsymbol{s}_{c2}'\boldsymbol{s}_{c2} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O} & \mathbf{O} & \cdots & \operatorname{diag}(\boldsymbol{s}_{cd}) - \boldsymbol{s}_{cd}'\boldsymbol{s}_{cd}, \end{bmatrix}, \tag{45}$$

with $\boldsymbol{s}_{cj} = [s_{cj}^1, s_{cj}^2, \ldots, s_{cj}^{K_j-1}]'$ and $\operatorname{diag}(\boldsymbol{s}_{cj})$ a diagonal matrix with the elements of $\boldsymbol{s}_{cj}$ on the diagonal. The comprehensive derivation of the gradient and Hessian of the log-partition function is given in appendix B.

3 Applications

This section shows the practical applicability of GCC. For this purpose two cases are presented and analysed. The analysis includes the discussion of the practical implementation of the clustering. The first case concerns the deaths of Prussian soldiers by a horse kick in cavalry corps and the clustering of those regiments. This dataset is a famous example of a Poisson distribution and is first used in Bortkewitsch (1898). The second case discusses the clustering of visitors of the Van Gogh Museum (VGM) for the purpose of personalising emails to said visitors. The data type available for VGM visitors is categorical in nature, for this the categorical distribution is suited.

3.1 Prussian horse-kicking

Prussia was a kingdom formed in 1701 on the southern coasts of the Baltic sea. The kingdom grew to become one of the great powers of Europe during the 18th century and it was thé driving force behind the unification of the Germany in the 19th century. The kings of Prussia accomplished this rapid rise to power due to creating a highly militarised country. The proportion of their citizens enlisted in their army was by far the highest of Europe and at its peak the military expenditure was 70% of the total budget of the Prussian state (Koch, 1978). As the Enlightenment writer and philosopher Voltaire put it: “Where some states have an army, the Prussian army has a state”.

From time to time though, this army was at peace. Although undeniably far less than in wartime, deaths did occur during peacetime. The army incurred additional costs due to these deaths, since the army had to pay widows a death gratuity and train a replacement (Koch, 1978). Next to the ethical part of loosing a life, it was thus also financially beneficial for the Prussian kings to reduce the number of deaths in peacetime.

To that end the book “The Law of Small Numbers”, written by the statistician Ladislaus Bortkiewicz in 1898, offered assistance. In this book Bortkiewicz researched the distribution of the number of Prussian soldiers dying due to horse kicks in times of peace. He obtained the annual number of deaths by horse kick from 14 Prussian cavalry corps over 20 years and showed that this data followed the Poisson distribution; there are many soldiers in a unit, where each soldier has a small chance to die by being kicked by a horse. The resulting number of deaths is Poisson distributed. The Prussian state could use this result to predict and anticipate the number of horse-kicking related deaths. The dataset can be obtained from randomservices.org/random/ (Random, 2019).

In this section this famous dataset is revised. GCC with the Poisson distribution clusters the different cavalry regiments and consequently determines which regiments handle their horses similarly. The likely result is that there is one cluster of units whom manage their horses better than the others. If this is indeed true, officers – if the Prussian cavalry core still existed – could apply the doctrine of the regiments in the cluster with the least number of deaths to the other units.

GCC is applied to the Prussian horse-kicking data with BIC and after the optimisation for different values for λ the optimal clustering is four groups of cavalry corps. Table 1 displays the different values of lambda with the corresponding number of clusters and BIC scores. The lowest value for the BIC scores is at $\lambda = 0.5$ and corresponds with four clusters. What is evident from this table is that the number of clusters is not monotonically decreasing, while this is expected. A likely explanation for this is that clusters are combined prematurely. For the optimisation it is necessary to combine clusters at each iteration if the fusion parameters are close to each other. If this is not done the gradient and Hessian of the penalty function are undefined. It is currently however not possible for these clusters to split up later on if this is more optimal. This effect is most notable with few clusters, but as table 1 shows, once λ increases and the number of clusters follows it is irrelevant. There are two possible solutions for this, the first is a different optimisation problem for which parameters do not need to be merged. The other solution is to use multiple random starts and use the most common clustering. This costs additional computational power and while this is doable for this small dataset, if the size increases the viability of this solution decreases.

Furthermore, the BIC scores are not smooth. This is to be expected from the issue described above, since only a decrease in λ or the number of clusters can result in a lower BIC. Specifically, if λ increases the log-likelihood increases and the BIC score can then only go down if the degrees of freedom decreases. This is only the case when the number of clusters decrease, in the other cases the BIC score increases.

λ	0.1	0.2	0.3	0.4	**0.5**	0.6	0.7	0.8	0.9	1.0
Number of clusters	8	9	7	8	**4**	4	3	3	3	2
BIC scores	523	527	521	526	**514**	517	521	526	535	531

Table 1: The number of clusters and BIC scores for different values of λ for the Prussian horse-kicking dataset using GCC. The clustering of $\lambda = 0.5$ is the best model according to the BIC.

Table 2 depicts the optimal clustering according to the GCC method of the Prussian horse-kicking dataset. It shows the four clusters ordered by the average annual deaths due to horse kicks and the corresponding cluster sizes, as well as the total of all 14 regiments. Readers familiar with the Poisson distribution would note that the average occurrence of the event of interest is the estimator of the source parameter of the Poisson distribution. Hence, the clustering finds four distinct Poisson distributions for this dataset, each representing a cluster. Cluster B and cluster C are relatively close together with an average annual deaths of respectively 0.58 and 0.72. Merging these clusters however increases the BIC score from 514 to 521 (as depicted in table 1 at $\lambda = 0.5$ and $\lambda = 0.7$) and is thus not an improvement.

Cluster	A	B	C	D	Total
Average annual deaths	0.38	0.58	0.72	1.23	0.70
Cluster size	3	2	7	2	14

Table 2: The average annual number of deaths and the number of cavalry regiments per cluster for the optimal clustering of the Prussian horse-kicking dataset using GCC.

Cluster C is the most average cluster; it has an average annual death nearly equal to the total of all cavalry corps and half of the regiments are in this cluster. Interestingly the Gaurdes du Corps, Prussian's elite cavalry regiment, are in cluster C, while it is expected that the elite would be able to handle their horses better than other regiments. As mentioned earlier, the goal of Prussian officers should be to reduce the number of deaths. They can achieve this by using regiments in cluster A to teach regiments in C and D to reduce the deaths. They should first focus on cluster D, since this has four times more deaths than cluster A.

To summarise, GCC is able to provide a clustering for a Poisson distributed dataset, which has not been possible before. This example did reveal a minor issue with the algorithm. When the number of merges is low, clusters form too quickly and the number of clusters is not negatively correlated with the penalty parameter.

3.2 Van Gogh Museum

The Van Gogh Museum (VGM) is thé museum dedicated to the world renowned painter Vincent van Gogh. Their collection contains 200 paintings, 500 drawings and 700 letters of van Gogh, including the famous "Sunflowers" and "The Potato Eaters". Visitors from all over the world come to see the biggest collection of van Gogh's in the world, supplemented with temporary exhibitions and art of his contemporaries. Last year, in 2018, the total amount of visitors was 217 000. With this feat, the VGM was placed 29th for the most visited museum in the world (AECOM and TEA, 2019).

The mission of the museum is to share the work of Van Gogh with the world and tries to engage and inspire their visitors. To continue competing with the most popular museums in the world VGM intents to innovate while staying true to their mission. To achieve this goal, the latest project they set up is "Van Gogh Personaliseert" (Van Gogh Personalises). This project aims to increase the visitor friendliness and engagement by personalising their visits. One focus of the project is the email contact from the museum to the visitors. Visitors receive emails after they booked their tickets with practical and other information about their visit. This allows for up- and cross-selling of other products and services of the museum. To personalise these emails, information about the types of visitors is necessary, which a market segmentation provides.

For the general marketing strategies of VGM, its marketing employees implement eight segments of people visiting the museum. This is based on a test which incorporates the personal norms and values of people. This information is however expensive to require as it involves individually interviewing visitors. Consequently it is not widely available and not applicable for personalising the emails. The outcomes of this test is a decent reference for a sanity check of the outcomes. The view of the marketing team is that there are four different type of visitors and that there furthermore is a major difference between Dutch and non-Dutch visitors. Thus there are eight segments according to their research.

The available data to create the segmentation is obtained at the moment of purchasing the ticket. This data is predominately categorical; the few non-categorical variables are converted to categorical. The data can be divided into several groups. The first group includes personal information such as the country of origin and preferred language. The second is time related. This includes the date of the purchase, the visit, and the derivatives from those two, such as the month, the day of the week and whether the visit is in the morning, afternoon, or evening. Another time related variable is the temporary exhibition in the museum at the time of the visit. Thirdly, there is information concerning the ticket composition, for which the most important derivatives is the type of ticket, the size of the group, whether there are children, and if an audio guide is bought. The remaining variables are whether the visitor is a repeat visitor, has a subscription for the newsletter, and/or makes a donation. Each visitor is described by one visit, due to the low frequency nature of the data; the mode of the amount of visits is one. This statement is not true for repeat visits, but these are rare and taken into account with a variable for revisit. Taking these directly into account is beyond the scope of this paper.

The GCC with the categorical distribution for the log-likelihood and with BIC is used to obtain results. Table 3 displays the values for λ and the corresponding BIC scores and number of clusters. As expected BIC is the highest at the lowest λ; $\lambda = 0.002$. At this point no merger occurs and there is a trivial clustering; setting $\lambda = 0$ results in the same clustering of the observations. This shows that there are at least 12 pairs of observations that exactly the same. The lowest BIC and therefore the best clustering is found at $\lambda = 0.020$, where there are six clusters. The stepsize is reduced after $\lambda = 0.17$ from 0.0050 to 0.0010 to ensure the best clustering is found. Taking an even smaller stepsize around $\lambda = 0.020$ does not result in a model with a lower BIC score.

Unlike the earlier Prussian horse-kicking example, the number of clusters and therefore the BIC scores are smooth. The number of clusters is monotonically decreasing and the BIC scores are distributed convexly. The issue of clustering observations too quickly together might occur for this dataset, but due to the sample size this issue has little effect. Furthermore at the optimal clustering this is unlikely to have any influence.

λ	0.002	0.007	0.012	0.017	0.018	0.019	**0.020**	0.021	0.022
Number of clusters	988	402	198	74	39	18	**6**	2	1
BIC scores	459987	202739	128102	51813	414746	35788	**33050**	34590	50153

Table 3: The number of clusters and BIC scores for different values of λ for the Van Gogh Museum dataset using GCC. The clustering of $\lambda = 0.020$ is the best model according to the BIC.

Table 4 shows the six clusters at $\lambda = 0.020$, ordered according to the number of observations in each clusters. Surprisingly the model placed almost all observations in one clusters; 994 observations are in cluster A. The remaining six individuals are distributed over the remaining five clusters. In this case the model apparently did not perform a clustering, but rather indicates which observations are outliers. While this is odd, the most likely explanation is that there is in fact only one type of visitor according to this data at the VGM, but this model indicates that some visitors are outliers to this general type.

Cluster	A	B	C	D	E	F
Cluster size	994	2	1	1	1	1

Table 4: The number of individuals per cluster for the optimal clustering of the Van Gogh Museum dataset using GCC.

In the previous example it was not possible to compare the results to an other clustering technique, since those do not exist for a Poisson dataset. While there are methods available that return results for categorical data, they are not developed for this type of data and there is no consensus on the ideal method. To compare with a similar method the VGM dataset is clustered using Hierarchical Agglomerative Clustering (HAC). The average linkage method and the Euclidean distance metric is used. The results of the clustering with HAC where the number of clusters is set to six is displayed in table 5. These results are similar to those of GCC, almost all observations are in a single cluster. In this case however there are 27 observations distributed along the remaining five clusters. Clusters D, E and F do indicate outliers, but clusters B and C could also been seen as small segments.

Cluster	A	B	C	D	E	F
Cluster size	973	12	8	3	3	1

Table 5: The number of individuals per cluster for the optimal clustering of the Van Gogh Museum dataset using HAC.

Given the results of GCC however, the conclusion for the VGM is that a segmentation of this data do not result in multiple clusters, but rather only one. Assuming that this is the case, personalising has very little effect, since the visitors behave the same. The goal for the marketing department of the VGM should be in optimising the content of the emails according to the average visitor. These conclusion do only hold if the results of GCC are in fact true. While it is not possible to determine the external validity of the model for this case study, this is possible in a simulation study.

4 Simulation study

The previous section displayed the practical applicability of GCC and proved that it is a beneficial addition to the clustering methods for business cases. This section follows up on that and supports the results by means of two simulation studies, one for the Poisson distribution and another for the categorical distribution.

For both studies the data is simulated with certain parameters for the corresponding distributions. With the simulated data two aspects are studied. The first is how the estimated number of clusters of AIC, BIC, and KIC relate to the true number of clusters. Secondly, the performance of the GCC model is inspected for different settings of the parameters. Since there is no benchmark clustering method for Poisson generated data available, the performance of GCC can not be compared to another method. For the categorical distribution this is possible by using average linkage HAC with the Euclidean distance metric, as specified in section 3.2.

4.1 Poisson Distribution

This section explains the study of the estimated number of clusters and the performance of GCC for the Poisson distribution in a controlled setting. First, the the parameters of the study and how they are adjusted are explained. Afterwards the study of the three information criteria is reported. Hereafter the performance of GCC for each possible alteration of parameters is shown and discussed. Lastly the implications of the results are explained.

There are many parameters to set for clustering, which can be split up in two groups. The first are the cluster parameters and the second are the distribution parameters.

The former group is defined by the number of clusters and whether the observations are distributed evenly or unevenly across those clusters. The total number of observations is set to 1000. This number is computationally feasible, while it is not so small that variance plays an influential role. The number of clusters considered are 2, 4, and 8. The number of observations remain constant, which ensures that when the number of clusters increases, the number of observations per cluster decreases. This enlarges the impact of randomness on the clustering, but this should not influence the results in a substantial way. An expected result from the simulation study is that for a lower number of clusters it is easier to determine the correct cluster, since there are less incorrect alternatives.

The distribution of the observations across these clusters is either evenly or unevenly distributed. In the evenly distributed setting the number of observations is equal for all clusters. In the unevenly distributed situation there is one clusters which is significantly larger than the others, while the distribution between the remaining clusters is even. For 2 clusters, one cluster contains 750 observations and the other 250. For 4 clusters, the dominant cluster contains 550 observations and the others 150 each. Finally for 8 clusters, the largest cluster has 440 observations and the remaining each 80. In the evenly distributed situation each cluster has an equal impact on the performance score. For the unevenly distributed case this does not hold, the performance of describing the dominant cluster has a substantial effect on the overall performance score. Thus correctly classifying the dominant cluster correctly is essential.

The second group of parameters define the distribution of each cluster. The parameters needed to simulate the Poisson distribution is the number of time periods per observation and the source parameter, which is the number of occurrences of an event during a certain time period, denoted as ζ. For the former 10, 20, and 40 periods are used. The only difference between the different number of periods is due to the variance. The results of 40 periods should have a lower variance in the results than those of 10 periods.

The source parameter ζ has a major effect on the clustering performance. The size of ζ itself should not impact the cluster performance considerably, but the relative difference between the ζ's of different clusters does. If this relative difference is large, there is little to no overlap between the clusters. The observations in different clusters do not look alike and the clusters are clearly defined. It is therefore easy to classify the observations in the corresponding clusters and thus most clustering methods perform well for this situation. The other situation is when the relative differences are large and there is a lot of overlap between the clusters. The observations are then fairly similar across the clusters and the clusters are not clearly defined. This makes it difficult to cluster and the performance of almost all clustering methods are lower in this situation. This study thus includes three situations for ζ: a *small* relative difference, a *medium* relative difference and a *large* relative difference. ζ is computed as follows: first the difference between each cluster is determined. For the *small* setting this is 1, for *medium* 1.5 and for *large* 2. Then this difference is incorporated around a base number, which is 10. For example, for a *small* relative difference with 2 clusters two ζ's are needed with a difference between them of 1 around the base of 10. This yields 9.5 and 10.5. Similarly for a *small* relative difference with 4 clusters the ζ's are 8.5, 9.5, 10.5, and 11.5 and for a *large* relative difference with 2 clusters 9 and 11. Each cluster then gets a ζ assigned randomly.

The data is simulated according to the Poisson distribution with the parameters settings as described as above. These simulations are first used to analyse the estimated number of clusters compared to the true clusters for AIC, BIC, and KIC and secondly to study the clustering performance of GCC.

For the information criteria analysis GCC is executed on all the different parameter alterations with AIC, BIC, and KIC. For each of these alterations the number of clusters estimated by the information criteria are recorded. These can then be compared to the true number of clusters. Figures 1, 2, and 3 show the estimated number of clusters for respectively 2, 4, and 8 true clusters. In these figures GCC is run with each criterion for 18 different datasets, since there are 2 settings for the distribution along the clusters, 3 settings for the time periods, and 3 settings for relative differences of ζ.

Figure 1 shows the estimated number of clusters for AIC, BIC, and KIC where the true number of clusters is 2. In this case BIC seems to preform the best of the three criteria. It estimates the number of clusters correct for 10 datasets, for 7 datasets it has 3 clusters and only one dataset is estimated higher at 4 clusters. KIC performs comparable, albeit it is slightly worse, with two datasets estimated to have 4 clusters. As expected AIC performs the worst of the three and overestimates the number of clusters. This is a problem AIC has for variable selection in regression models and apparently also for GCC on Poisson simulated datasets.

The next figure shows the situation for 4 true clusters. From figure 2 it appears evident that the general remarks obtained from analysing figure 1 still hold. AIC performs worst and overestimates the number of clusters. BIC and KIC perform comparable, but BIC estimates the true number of cluster more often correctly. A new insight from this graph is that the estimated number of clusters of KIC seems have more variance compared to BIC. The estimated number of cluster of KIC ranges from 3 to 7 clusters, while BIC ranges from 4 to 6.

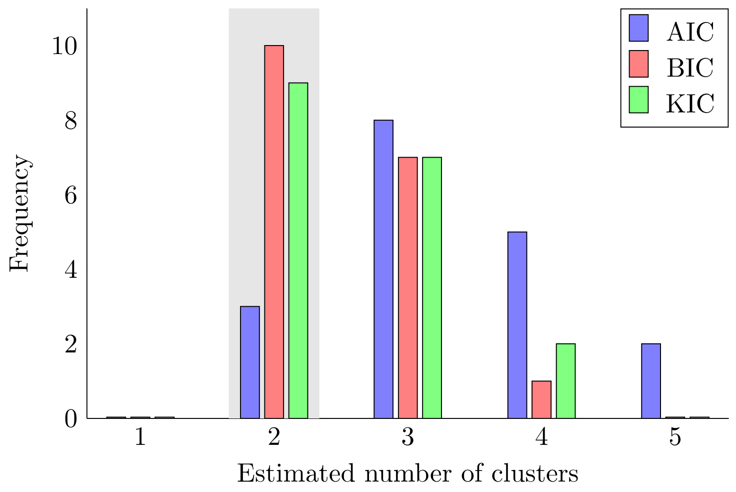

Figure 1: The estimated number of clusters of AIC, BIC, and KIC for 18 Poisson simulated datasets with 2 true clusters. The grey area emphasises the true number of clusters.

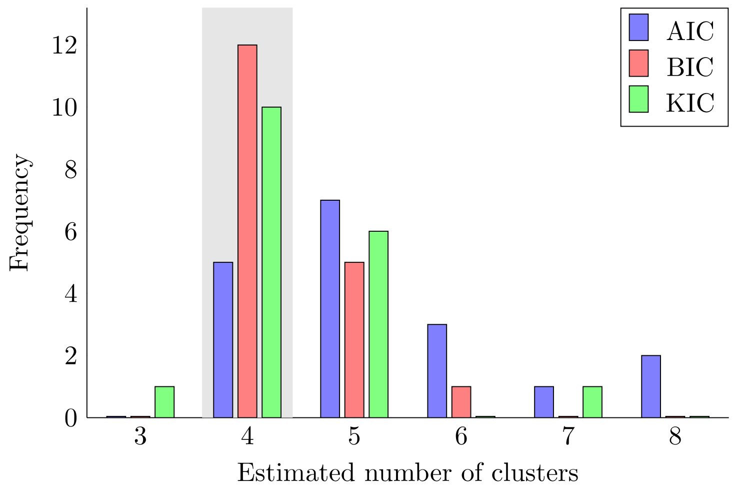

Figure 2: The estimated number of clusters of AIC, BIC, and KIC for 18 Poisson simulated datasets with 4 true clusters. The grey area emphasises the true number of clusters.

Finally, figure 3 compares the information criteria for the 18 datasets with 8 true clusters. This figure reconfirms the previous observations. BIC estimates the number of clusters most often correctly, KIC is slightly worse, and AIC consistently overestimates the number of clusters. Generally the spread of estimated number of clusters is higher than the previous two situations. It appears that for a higher number of clusters, it is more difficult for the information criteria to find the true number of clusters for the GCC. It is however likely that this is due to the model itself; the datasets for which BIC and KIC estimated lower than 8 clusters or higher than 10 clusters are the datasets where there is a lot of overlap between the clusters. As mentioned earlier, these models are more difficult to predict and true clusters are more likely to be merged or split up. Still the variance of KIC seems to be a bit higher than that of BIC.

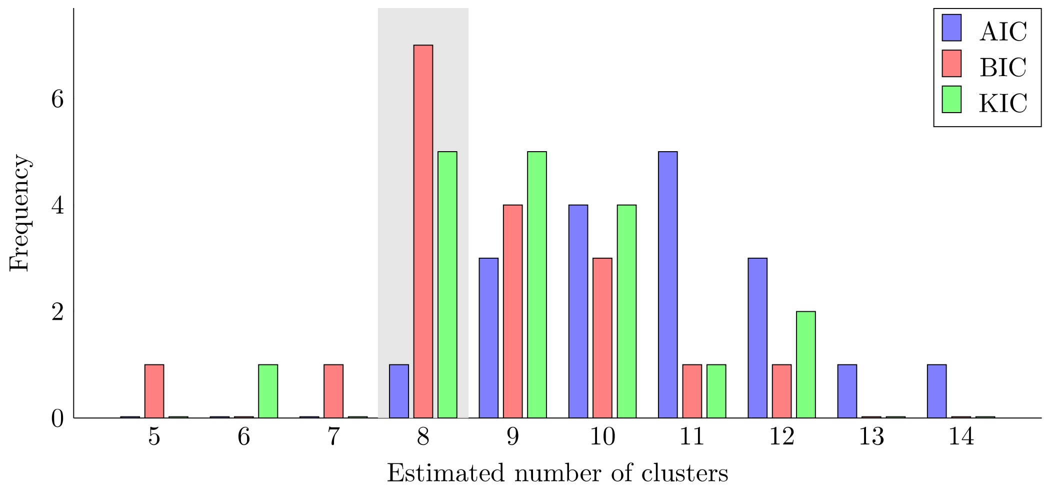

Figure 3: The estimated number of clusters of AIC, BIC, and KIC for 18 Poisson simulated datasets with 8 true clusters. The grey area emphasises the true number of clusters.

As a conclusion of this part of the simulation, BIC is the best information criterion of the three presented here. It predicts the true number of clusters correct most of the time and has the lowest variance in the estimated number of clusters. KIC performs decent, however it has a higher variance than BIC. Lastly, AIC overestimates the true number of clusters consistently, as was expected.

The remaining section of this specific simulation study focuses on the performance of GCC for the Poisson distribution, by inspecting the F_1 and entropy score. Here BIC is used to determine the number of clusters. However as the figures above show, the true number of clusters is not estimated for all models. To calculate the F_1 and entropy score each estimated cluster needs to be linked to a true clusters and thus each estimated clusters is linked to a true clusters according to the highest precision.

After running the data through the GCC and inspecting the performance, it seemed that the number of time periods had minimal influence on the performance scores. As aforementioned this is expected and therefore the time periods are not relevant to show. The number of time periods is thus set to 40. The other parameter changes do provide information about the model and are thus shown in table 6. The table shows the F_1 and entropy scores of the results from GCC for 2, 4, and 8 clusters in an evenly and unevenly distributed setting. This is done for a relatively small, medium and large difference of ζ between the observations. The range of the F_1 score is between 0 and 1, where 1 is a perfect clustering. For the entropy the range is between 0 and 3, where 0 indicates a perfect clustering.

		Evenly distributed Number of clusters			**Unevenly distributed Number of clusters**		
ζ difference		2	4	8	2	4	8
Small	F1	0.665	0.429	0.248	0.856	0.750	0.627
	Entropy	0.673	1.353	1.982	0.531	1.053	1.592
Medium	F1	0.692	0.531	0.363	0.870	0.762	0.654
	Entropy	0.621	1.209	1.638	0.460	0.875	1.321
Large	F1	0.740	0.602	0.485	0.895	0.792	0.712
	Entropy	0.570	1.014	1.208	0.401	0.752	0.837

Table 6: The F_1 and entropy scores of the simulation study of the Poisson distribution using different parameter settings. The ζ difference indicates the difference between the clusters.

What is apparent from table 6 is that an increase in the number of clusters leads to a drop in performance. For all different cases this is evident and there is no exception. This is to be expected, but the difference is quite substantial. Due to more clusters, the relative amount of overlap between the clusters becomes bigger. The increase of the log-likelihood due to merging two different clusters becomes smaller when there is more overlap. Thus the GCC would be more inclined to merge the clusters compared to a low overlap situation. Hence it is obvious why GCC performs better when the relative difference between ζ's increases. As aforementioned, most clustering methods perform better in little overlap situations compared to more overlap. This result is therefore not surprising.

The second result to note is the difference between performance on an evenly and unevenly distributed dataset. As stated earlier in an unevenly distributed dataset, there is one cluster that has a higher influence on the performance measures. The GCC performs better on the unevenly distributed dataset compared to the evenly distributed dataset for all cases, thus the dominant cluster is often described correctly.

The GCC model shows promising results for an unevenly distributed dataset and an evenly distributed dataset with a low number of clusters. It is however not possible to compare this to other clustering models, since there is no benchmark method yet for this type of data. While the clustering of Poisson data is yet to be actively discussed in the literature, there now is a method that can serve as a benchmark and can be improved upon.

4.2 Categorical Distribution

While the previous section followed up on GCC in case of Poisson distributed data this section discusses the simulation study for categorical distributed data. Again the set-up of the parameters is discussed, hereafter comes the comparison of AIC, BIC, and KIC, after which the results are displayed and discussed. Unlike the Poisson case, there is a clustering method that can be used as a benchmark. As argued in section 3.2, Hierarchical Agglomerative Clustering (HAC) with average linkage and the Euclidean distance metric, while not designed specifically for this data, is a possible method to identify clusters within categorical data. The results of GCC are therefore compared to HAC to find out of this new model is an improvement over the established methods.

The set-up of the cluster parameters is the same as the Poisson distribution case. The number of observations is 1000, they are distributed along 2, 4, or 8 clusters and this distribution is either evenly or unevenly.

The difference is the way the source parameters for the distribution are determined. As discussed earlier in section 2.2.4 each variable in the categorical distribution has K classes where each class k has a certain probability p^k that it is observed. In this simulation study there are 10 variables. This is a compromise between computational power and variance. A lower number of variables does speed up the model, but it increases the effect of variance in the data generation. A higher number of variables results in the opposite. For each variable the number of classes is determined as a random number between 2 and 9. Then for each class and each cluster a random weight is set. These weights are later translated into a probabilities, but they are first adjusted to ensure that there is a difference along the clusters. For each class the sum of the weights of all clusters is determined. To ensure that one cluster is dominant per class a proportion of this sum is added to the highest weight. This proportion is either *small* (10%), *medium* (35%), or *large* (60%). For each cluster the resulting weights are then translated into a probability such that the sum of all classes of a variable for each cluster is equal to 1. A larger proportion setting thus results in bigger differences between the clusters in terms of probabilities, which causes the overlap between the clusters to be smaller and thus makes clustering easier. The clustering models should perform best in the *large* p^k difference setting and worst in the *small* p^k difference setting.

With this set-up the simulation study is performed. The comparison of the information criteria is performed in a similar way as the previous section. GCC is executed on the simulated data with AIC, BIC, and KIC. The estimated number of clusters are shown in figures 4, 5, and 6 for respectively 2, 4, and 8 clusters. There are now only 6 different alterations of parameters per number of clusters: even or uneven and a *small*, *medium* or *large* p^k difference. To increase the sample size each alteration is repeated 3 times. Due to the randomness in the process of generating the source parameter the repeated generated datasets differ from each other.

The first situation is where there are 2 true clusters. Figure 4 displays the results. Both BIC and KIC seem to be performing well. For 11 models they estimate the correct number of clusters and most of the remaining models are estimated at 3 clusters. Only for two models for both BIC and KIC is the estimated number of clusters higher. Furthermore, the variance of the estimations of BIC and KIC seems similar. AIC on the other hand does not perform as well. For most datasets it overestimates the number of clusters. This is also seen in the previous simulation study and generally in the literature, the dimensions estimated by AIC is higher than that of the true model.

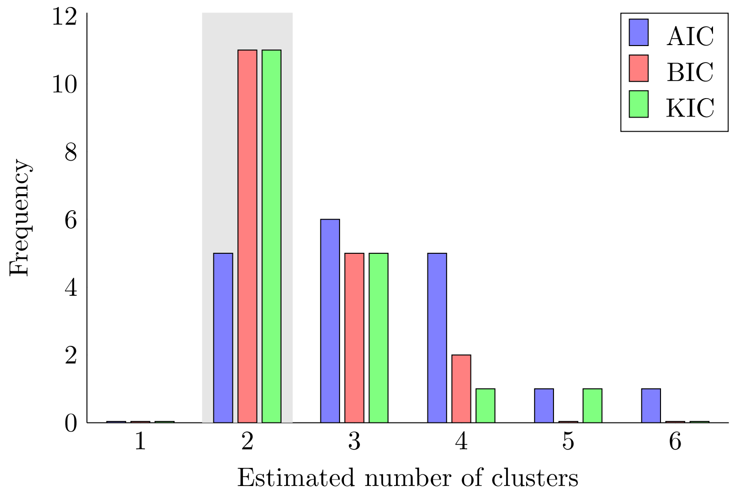

Figure 4: The estimated number of clusters of AIC, BIC, and KIC for 18 categorical simulated datasets with 2 true clusters. The grey area emphasises the true number of clusters.

Figure 5 illustrates a similar story as figure 4. BIC and KIC perform alike, where this time KIC performs slightly better. Again there does not seem to be much difference in the spread of the estimated number of clusters of BIC and KIC. AIC again performs worst, since it overestimates the dimenions.

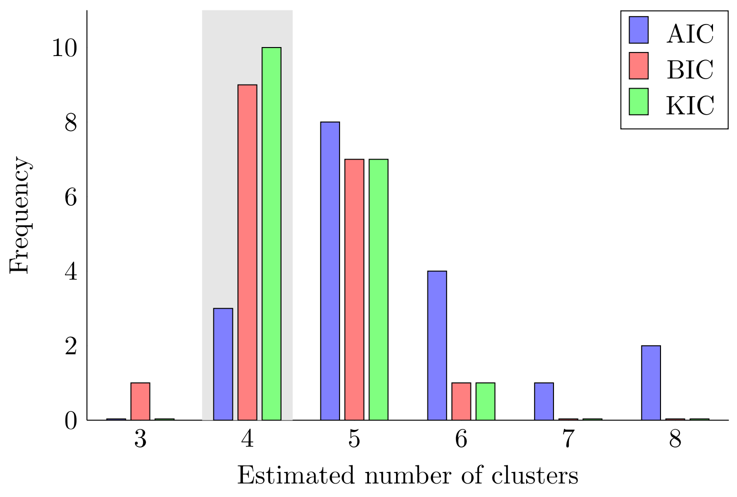

Figure 5: The estimated number of clusters of AIC, BIC, and KIC for 18 categorical simulated datasets with 4 true clusters. The grey area emphasises the true number of clusters.

Lastly is the situation of 8 true clusters. Figure 6 shows the results of the estimated number of clusters for the information criteria. It confirms that BIC and KIC perform comparable. KIC estimates for more datasets the number of clusters correctly, while the spread of the estimations of BIC is smaller, implying a lower variance. AIC again consistently estimates a number too high. As also shown in the Poisson simulation study, the spread of 8 clusters is higher than that of 2 and 4 clusters. For a higher number of clusters, the variance of the estimations seems higher.

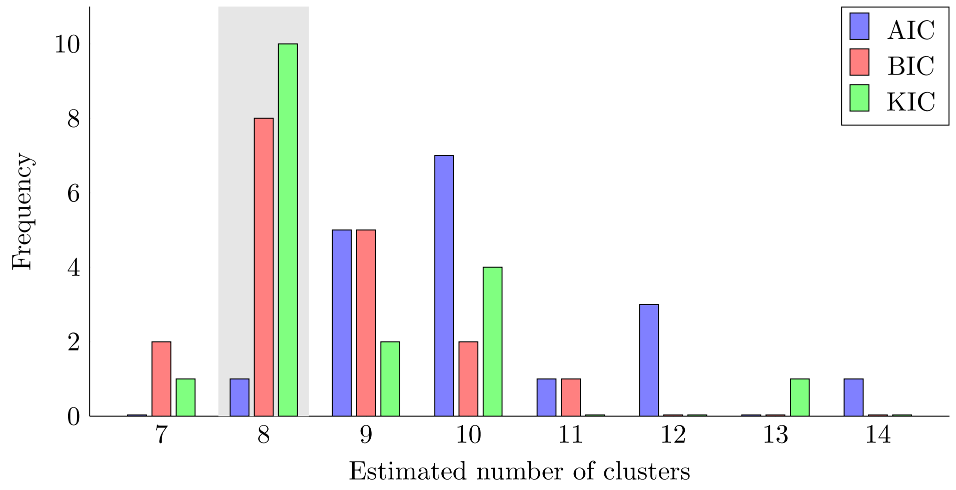

Figure 6: The estimated number of clusters of AIC, BIC, and KIC for 18 categorical simulated datasets with 8 true clusters. The grey area emphasises the true number of clusters.

From the comparison of the information criteria it follows that BIC and KIC perform similarly for categorical data. BIC has a lower variance in the estimations, but KIC is more accurate in predicting the true number of clusters. AIC on the other hand consistently overestimates the number of clusters and is clearly not viable. Both BIC and KIC are viable to use for the rest of the simulation study. BIC is however more consistent, and thus this criterion is chosen.

Therefore the GCC model with BIC is run on each setting of the parameters to obtain the F_1 and the entropy score. Similar as in the Poisson simulation study each estimated clusters is linked to a true clusters on the basis of highest precision. For this part of the simulation study HAC is the benchmark model. HAC however cannot use the BIC score to find the optimal number of clusters. There are other internal validation methods such as the Davies-Bouldin, silhoutte, and Dunn indices. The focus of this paper is the GCC method, hence for the HAC the true number of clusters is assumed to be known. This should enhance the performance of HAC and shows how GCC compares to the ideal situation for the benchmark model.

The performance results are displayed in table 7. For the different parameter settings the F_1 and entropy scores are shown for both the GCC and HAC methods. The F_1 score lies on a range between 0 and 1, where 1 is a perfect clustering, and that of the entropy is between 0 and 3, where 0 indicates a perfect clustering.

			Evenly distributed Number of clusters			**Unevenly distributed Number of clusters**		
p^k difference			2	4	8	2	4	8
Small	F1	GCC	0.713	0.692	0.637	0.871	0.829	0.722
		HAC	0.671	0.636	0.525	0.857	0.740	0.634
	Entropy	GCC	0.586	0.905	1.132	0.533	0.601	0.776
		HAC	0.693	1.085	1.539	0.562	0.761	1.319
Medium	F1	GCC	0.901	0.847	0.777	0.945	0.795	0.783
		HAC	0.862	0.810	0.748	0.936	0.744	0.677
	Entropy	GCC	0.403	0.642	0.830	0.237	0.589	0.788
		HAC	0.541	0.668	0.938	0.301	0.735	1.136
Large	F1	GCC	0.982	0.923	0.899	0.986	0.960	0.913
		HAC	0.931	0.907	0.831	0.990	0.919	0.903
	Entropy	GCC	0.046	0.268	0.368	0.036	0.077	0.192
		HAC	0.091	0.163	0.588	0.048	0.279	0.382

Table 7: The F_1 and entropy scores of the simulation study of the categorical distribution for both Generalised Convec Clustering and Hierarchical Agglomerative Clustering using different parameter settings. The p^k difference indicates the difference between the clusters.

The results in table 7 show that the performance of GCC for categorical distributed data improves the less clusters there are, the more uneven these clusters are distributed, and the bigger the p^k differences. This is similar to what the Poisson simulation study showed. In general though, the F_1 and entropy scores for GCC are better for the categorical distribution than for the Poisson distribution. They are however two different models and it is not possible to compare these as such and state that GCC is more suited for categorical data.

There is an unexpected result of the F_1 score for 4 unevenly distributed clusters with a medium difference. The score drops relatively a lot between 2 and 4 clusters and only slightly between 4 and 8 clusters. The entropy score suggests that this should be more even. The F_1 score is therefore likely an outlier.

The comparison against the benchmark model HAC suggests that GCC outperforms the established models. This is not surprisingly, since GCC incorporates how the data is distributed while HAC does not take this into account. With few, unevenly, and a large difference between the clusters the two models perform comparable however; the F_1 score indicates that HAC outperforms GCC (0.990 and 0.986 respectively), while the entropy scores suggests the opposite (0.048 and 0.036 respectively). The reason both models perform well in this situation, is that there is very little overlap between the clusters. In practice these situations are relatively easy to cluster and a slight difference in performance is irrelevant. The situations were the overlap between the clusters is minimal is were the differences between the models is most noticeable. Then the power of the parametric convex method becomes apparent. For instance when the difference between 8 evenly distributed clusters is small, GCC scores considerably better than HAC. The F_1 score is 0.637 for GCC while it is only 0.525 for HAC and the entropy scores confirm this with 1.132 for GCC and 1.539 for HAC.

In general GCC performs best for categorical data when the differences between the clusters are relatively large and thus the overlap is small. This holds for all clustering methods and hence in this situation GCC performs similar to the established benchmark. The advantage GCC has over the established method is in the situation when the overlap between clusters is large and clustering is more difficult. Naturally the performance of both GCC as the benchmark drop. GCC however becomes better compared to the benchmark if the relative difference between clusters becomes smaller. This is an promising result for GCC.

5 Conclusion

In this paper the Generalised Convex Clustering (GCC) method is presented. While convex clustering was previous only possible for continuous and binary data, GCC extends the convex clustering method to different types of data. The optimisation function of GCC consists of two parts: a log-likelihood function, which models the fit of the data, and a penalty function, which ensures that clusters are formed. If the penalty function is a pairwise fused Ridge restriction, this log-likelihood function can be of any distribution of the exponential family, which enables convex clustering of all data types that those distributions can describe.

To obtain evidence of the viability of the GCC model, four distributions are considered in this paper. Using the Gaussian and Bernoulli distribution allows the the clustering of respectively continuous and binary data. The resulting optimisation problems of GCC is the same as the previous papers on convex clustering for continuous and binary data. This confirms that the generalisation is performed correctly. The Poisson and categorical distribution have not yet been considered in other papers and both show promising practical and simulation results. This paper thus achieves the goal to construct a general method for convex clustering and extends the literature of convex clustering.

GCC with the Poisson distribution is the first clustering method for Poisson distributed data. This opens up new analysis and insights. The simulation shows that this model performs well when the clusters are relatively different, but the performance drops substantially when the overlap between clusters becomes larger.

There are models that deal with cluster analysis for categorical data, however these are not explicitly suited for this type of data. GCC with the categorical distribution is specifically meant for that type and due to this performs well compared to a benchmark model; it scores comparable in a setting were the overlap between the clusters is small, and outperforms the benchmark when the overlap between clusters is large. The case studies demonstrate the benefit for organisations to implement GCC to obtain a market segmentation or for other purposes.

Furthermore this paper studied three information criteria, which estimate the number of clusters of GCC. These criteria are the Akaike Information Criteria (AIC), the Bayesian Information Criteria (BIC), and the Kullback Information Criteria (KIC). Similarly to the conclusions of other literature the AIC overestimates the dimensions of the GCC model. Both BIC and KIC perform comparable, however BIC is preferred for the Poisson and categorical distribution due to the low variance in the estimated number of clusters.

The main shortcoming of the model is the computational effort of the optimisation. Newton's method is easy to implement in programming languages and both the gradient and the Hessian needed to perform this method are available for the entire exponential family. It is however slow and therefore the number of observations this model can cluster is limited. While other papers deal with this problem for the Gaussian distribution, further research should extend this to GCC to speed up the optimisation. Most preferably this solution does not require observations to be merged, as then observations cannot be combined too quickly. The difference between the scaling of the variables is furthermore an interesting topic for further research. If such difference between scaling exists, some variables might have an unsubstantiated stronger influence over the clustering than other variables. Possible solutions are to scale the sufficient statistic or to implement a PCA method. Another aspect this paper addressed is the 1-norm versus 2-norm. While it is proven that the 1-norm results in an arbitrary clustering for some distributions, it is not yet clear for which distributions this exactly holds and for which distributions a choice between the two norms is possible. Lastly, the practical application of GCC is shown for two different distributions in this paper. Case studies for different data types can be beneficial to gain insights in the cases, but also in the way that GCC works. Improvements for the specific cases can be used to enhance the general model.

References

AECOM and TEA (2019). Theme index and museum index 2018. URL: https://www.aecom.com/theme-index.

Akaike, H. (1973). Information theory and an extension of the maximum likelihood principle. In *Proceedings of the second international Symposium on Information Theory (B. N. Petrov and F. Csáki eds.)*, pages 199–213.

Akogul, S. and Erişoğlu, M. (2016). A comparison of information criteria in clustering based on mixture of multivariate normal distributions. *Mathematical and Computational Applications*, 21(3):34.

Boeriu, H. (2009). Video collection: Bmw films – the hire. URL: https://www.bmwblog.com/2009/08/25/video-collection-bmw-films-the-hire/.

Boriah, S., Chandola, V., and Kumar, V. (2008). Similarity measures for categorical data: A comparative evaluation. In *Proceedings of the 8th SIAM International Conference on Data Mining*, pages 243–254.

Bortkewitsch, L. v. (1898). *Das Gesetz der kleinen Zahlen.* B.G. Teubner Verlag, Leipzig, Germany.

Cavanaugh, J. E. (1999). A large-sample model selection criterion based on kullback's symmetric divergence. *Statistics & Probability Letters*, 42(4):333–343.

Choi, H. and Lee, S. (2018). Convex clustering for binary data. *Advances in Data Analysis and Classification*, 13(4):991–1018.

Choi, S.-S., Cha, S.-H., and Tappert, C. C. (2010). A survey of binary similarity and distance measures. *Journal of Systemics, Cybernetics and Informatics*, 8(1):43–48.

Donoho, D. L. (2001). Sparse components of images and optimal atomic decompositions. *Constructive Approximation*, 17(3):353–382.

Fahad, A., Alshatri, N., Tari, Z., Alamri, A., Khalil, I., Zomaya, A. Y., Foufou, S., and Bouras, A. (2014). A survey of clustering algorithms for big data: Taxonomy and empirical analysis. *IEEE transactions on emerging topics in computing*, 2(3):267–279.

Hespos, T. (2002). Bmw films: The ultimate marketing scheme. URL: https://www.imediaconnection.com/articles/ported-articles/red-dot-articles/2002/jul/bmw-films-the-ultimate-marketing-scheme/.

Hocking, T. D., Joulin, A., Bach, F., and Vert, J.-P. (2011). Clusterpath an algorithm for clustering using convex fusion penalties. In *Proceedings of the 28th International Conference on Machine Learning*, pages 745–752.

Jordan, M. I. (2009). The exponential family: Basics. URL: https://people.eecs.berkeley.edu/jordan/courses/260-spring10/other-readings/chapter8.pdf.

Koch, H. (1978). *A History of Prussia*, chapter 5. Addison-Wesley Longman Limited.

Kotler, P., Armstrong, G., Saunders, J., and Wong, V. (2000). *Principes van marketing. De Europese editie*. Academic Service.

Lindsten, F., Ohlsson, H., and Ljung, L. (2011). Just relax and come clustering! : A convexification of k-means clustering. In *Technical Report, Linköpings Universitet*.

Nielsen, F. and Garcia, V. (2009). Statistical exponential families: A digest with flash cards. *arXiv preprint*. arXiv:0911.4863.

Petry, S., Flexeder, C., and Tutz, G. (2011). Pairwise fused lasso. In *Technical Reports, University of Munich*.

Price, B. S., Geyer, C. J., and Rothman, A. J. (2015). Ridge fusion in statistical learning. *Journal of Computational and Graphical Statistics*, 24(2):439–454.

Random (2019). Prussian horse-kick data. http://www.randomservices.org/random/, last visited at 14/10/2019.

Rendón, E., Abundez, I., Arizmendi, A., and Quiroz, E. M. (2011). Internal versus external cluster validation indexes. *International Journal of computers and communications*, 5(1):27–34.

Schwarz, G. et al. (1978). Estimating the dimension of a model. *The annals of statistics*, 6(2):461–464.

Shannon, C. E. (1948). A mathematical theory of communication. *Bell system technical journal*, 27(3):379–423.

Tibshirani, R., Saunders, M., Rosset, S., Zhu, J., and Knight, K. (2005). Sparsity and smoothness via the fused lasso. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 67(1):91–108.

Xu, R. and Wunsch, D. C. (2005). Survey of clustering algorithms. *IEEE Transactions on Neural Networks and Learning Systems*, 16(3):645–678.

Appendices

A Derivative Penalty Function

The penalty function is given by:

$$P_\lambda(\boldsymbol{\Theta}) = \lambda \sum_{1\le u<v\le n} \|\boldsymbol{\theta}_u - \boldsymbol{\theta}_v\|_2. \tag{46}$$

The gradient of the penalty function towards $\boldsymbol{\theta}_i$ is:

$$\begin{aligned}
\frac{\delta P_\lambda(\boldsymbol{\Theta})}{\delta \boldsymbol{\theta}_i} &= \frac{\delta \lambda \sum_{1\le u<v\le n} \|\boldsymbol{\theta}_u - \boldsymbol{\theta}_v\|_2}{\delta \boldsymbol{\theta}_i} \\
&= \lambda \frac{\delta \sum_{v=1}^{i-1} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2 + \sum_{v=i+1}^{n} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2}{\delta \boldsymbol{\theta}_i} \\
&= \lambda \frac{\delta \sum_{v\neq i} \|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2}{\delta \boldsymbol{\theta}_i} \\
&= \lambda \frac{\delta \sum_{v\neq i} \sqrt{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)'(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)}}{\delta \boldsymbol{\theta}_i} \\
&= \lambda \sum_{v\neq i} \frac{1}{2} \frac{1}{\sqrt{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)'(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)}} \cdot 2(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v) \\
&= \lambda \sum_{v\neq i} \frac{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)}{\sqrt{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)'(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)}} \\
&= \lambda \sum_{v\neq i} \frac{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2}.
\end{aligned} \tag{47}$$

The Hessian is derived as the following:

$$\begin{aligned}
\frac{\delta^2 P_\lambda(\boldsymbol{\Theta})}{\delta \boldsymbol{\theta}_i \delta \boldsymbol{\theta}_i'} &= \lambda \frac{\delta \sum_{v\neq i} (\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)/\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2}{\delta \boldsymbol{\theta}_i'} \\
&= \lambda \frac{\delta \sum_{v\neq i} (\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)\left(\sqrt{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)'(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)}\right)^{-1}}{\delta \boldsymbol{\theta}_i'} \\
&= \lambda \sum_{v\neq i} \left(\frac{\boldsymbol{I}}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2} - \frac{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2^2} \cdot \frac{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)'}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2} \right) \\
&= \lambda \sum_{v\neq i} \left(\frac{\boldsymbol{I}}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2} - \frac{(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)(\boldsymbol{\theta}_i - \boldsymbol{\theta}_v)'}{\|\boldsymbol{\theta}_i - \boldsymbol{\theta}_v\|_2^3} \right).
\end{aligned} \tag{48}$$

B Derivative Categorical Log-partition Function

This appendix shows the gradient and Hessian of log-partition function of the categorical distribution.
The log-partition function is described by z^k_{cj}:

$$G(\theta^k_{cj}) = z^k_{cj} = \log\left(1 + \sum_{l=1}^{K_j-1} \exp(\theta^l_{cj})\right), \tag{49}$$

where θ^k_{cj} and z^k_{cj} can be vectorised as explained in section 3.2. To simplify the derivation of the gradient and Hessian it is broken down into single derivatives:

$$\begin{gathered}\frac{\delta G}{\delta \boldsymbol{\theta}_c} = \left[\frac{\delta G}{\delta \theta^1_{c1}}, \frac{\delta G}{\delta \theta^2_{c1}}, \dots, \frac{\delta G}{\delta \theta^{K_1-1}_{c1}}, \frac{\delta G}{\delta \theta^1_{c2}}, \dots, \frac{\delta G}{\delta \theta^{K_d-1}_{cd}}\right]', \\ \frac{\delta^2 G}{\delta \boldsymbol{\theta}_c \delta \boldsymbol{\theta}'_c} = \begin{bmatrix} \frac{\delta^2 G}{(\delta\theta^1_{c1})^2} & \frac{\delta^2 G}{\delta\theta^1_{c1}\delta\theta^2_{c1}} & \cdots & \frac{\delta^2 G}{\delta\theta^1_{c1}\delta\theta^{K_1-1}_{c1}} & \frac{\delta^2 G}{\delta\theta^1_{c1}\delta\theta^1_{c2}} & \cdots & \frac{\delta^2 G}{\delta\theta^1_{c1}\delta\theta^{K_d-1}_{cd}} \\ \frac{\delta^2 G}{\delta\theta^2_{c1}\delta\theta^1_{c1}} & \frac{\delta^2 G}{(\delta\theta^2_{c1})^2} & \cdots & \frac{\delta^2 G}{\delta\theta^2_{c1}\delta\theta^{K_1-1}_{c1}} & \frac{\delta^2 G}{\delta\theta^2_{c1}\delta\theta^1_{c2}} & \cdots & \frac{\delta^2 G}{\delta\theta^2_{c1}\delta\theta^{K_d-1}_{cd}} \\ \vdots & \vdots & \ddots & \vdots & \vdots & \ddots & \vdots \\ \frac{\delta^2 G}{\delta\theta^{K_d-1}_{cd}\delta\theta^1_{c1}} & \frac{\delta^2 G}{\delta\theta^{K_d-1}_{cd}\delta\theta^2_{c1}} & \cdots & \frac{\delta^2 G}{\delta\theta^{K_d-1}_{cd}\delta\theta^{K_1-1}_{c1}} & \frac{\delta^2 G}{\delta\theta^{K_d-1}_{cd}\delta\theta^1_{c2}} & \cdots & \frac{\delta^2 G}{(\delta\theta^{K_d-1}_{cd})^2} \end{bmatrix}.\end{gathered} \tag{50}$$

For the gradient the one-dimensional derivative is given by:

$$\begin{aligned}\frac{\delta G}{\delta \theta^k_{cj}} &= \sum_{p=1}^{d} \frac{\delta z^k_{cp}}{\delta \theta^k_{cj}} = \frac{\delta z^k_{cj}}{\delta \theta^k_{cj}} \\ &= \frac{\delta \log\left(1 + \sum_{l=1}^{K_j-1} \exp(\theta^l_{cj})\right)}{\delta \theta^k_{cj}} \\ &= \frac{1}{1 + \sum_{l=1}^{K_j-1} \exp(\theta^l_{cj})} \exp(\theta^k_{cj}) \\ &= \frac{\exp(\theta^k_{cj})}{1 + \sum_{l=1}^{K_j-1} \exp(\theta^l_{cj})} \\ &= s^k_{cj},\end{aligned} \tag{51}$$

and thus:

$$\frac{\delta G}{\delta \boldsymbol{\theta}_c} = \left[s^1_{c1}, s^2_{c1}, \dots, s^{K_1-1}_{c1}, s^1_{c2}, \dots, s^{K_d-1}_{cd}\right]' = \boldsymbol{s}_c. \tag{52}$$

For the Hessian there are three different cases for the derivative $\frac{\delta^2 G}{\delta\theta^k_{cj}\delta\theta^q_{cp}}$. The first is $p = j$ and $q = k$, the second is $p = j$ and $q \neq k$, and third is $p \neq j$.0

For $p = j$ and $q = k$:

$$\begin{aligned}
\frac{\delta^2 G}{(\delta \theta_{cj}^k)^2} = \frac{\delta s_{cj}^k}{\delta \theta_{cj}^k} &= \frac{\delta \Big[\exp(\theta_{cj}^k)\big(1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)\big)^{-1} \Big]}{\delta \theta_{cj}^k} \\
&= \frac{\exp(\theta_{cj}^k)}{1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)} - \frac{\exp(\theta_{cj}^k)}{\big(1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)\big)^2} \exp(\theta_{cj}^k) \\
&= \frac{\exp(\theta_{cj}^k)}{1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)} - \left(\frac{\exp(\theta_{cj}^k)}{1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)} \right)^2 \\
&= s_{cj}^k - {s_{cj}^k}^2
\end{aligned} \tag{53}$$

For $p = j$ and $q \neq k$:

$$\begin{aligned}
\frac{\delta^2 G}{\delta \theta_{cj}^k \delta \theta_{cj}^q} = \frac{\delta s_{cj}^k}{\delta \theta_{cj}^q} &= \frac{\delta \Big[\exp(\theta_{cj}^k)\big(1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)\big)^{-1} \Big]}{\delta \theta_{cj}^q} \\
&= -\frac{\exp(\theta_{cj}^k)}{\big(1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)\big)^2} \exp(\theta_{cj}^q) \\
&= -\frac{\exp(\theta_{cj}^k)}{1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)} \cdot \frac{\exp(\theta_{cj}^q)}{1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)} \\
&= -s_{cj}^k \cdot s_{cj}^q
\end{aligned} \tag{54}$$

For $p \neq j$:

$$\begin{aligned}
\frac{\delta^2 G}{\delta \theta_{cj}^k \delta \theta_{cp}^q} = \frac{\delta s_{cj}^k}{\delta \theta_{cp}^q} &= \frac{\delta \Big[\exp(\theta_{cj}^k)\big(1 + \sum_{l=1}^{K_j-1} \exp(\theta_{cj}^l)\big)^{-1} \Big]}{\delta \theta_{cp}^q} \\
&= 0
\end{aligned} \tag{55}$$

Using $\boldsymbol{\theta}_{cj} = [\theta_{cj}^1, \theta_{cj}^2, \ldots, \theta_{cj}^{K_j-1}]'$, $\boldsymbol{s}_{cj} = [s_{cj}^1, s_{cj}^2, \ldots, s_{cj}^{K_j-1}]'$ to simplify the Hessian into a block matrix results in:

$$\frac{\delta^2 G}{\delta \boldsymbol{\theta}_c \delta \boldsymbol{\theta}_c'} = \begin{bmatrix} \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{c1}' \delta \boldsymbol{\theta}_{c1}} & \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{c1}' \delta \boldsymbol{\theta}_{c2}} & \cdots & \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{c1}' \delta \boldsymbol{\theta}_{cd}} \\ \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{c2}' \delta \boldsymbol{\theta}_{c1}} & \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{c2}' \delta \boldsymbol{\theta}_{c2}} & \cdots & \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{c2}' \delta \boldsymbol{\theta}_{cd}} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{cd}' \delta \boldsymbol{\theta}_{c1}} & \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{cd}' \delta \boldsymbol{\theta}_{c2}} & \cdots & \frac{\delta^2 G}{\delta \boldsymbol{\theta}_{cd}' \delta \boldsymbol{\theta}_{cd}} \end{bmatrix}. \tag{56}$$

From the cases above it is clear that:

$$\frac{\delta^2 G}{\delta \boldsymbol{\theta}_{cj} \delta \boldsymbol{\theta}_{cp}'} = \begin{cases} \operatorname{diag}(\boldsymbol{s}_{cj}) - \boldsymbol{s}_{cj}' \boldsymbol{s}_{cj} & \text{if } j = p \\ \mathbf{O} & \text{otherwise.} \end{cases} \tag{57}$$

Thus:

$$\frac{\delta^2 G}{\delta \boldsymbol{\theta}_c \delta \boldsymbol{\theta}_c'} = \begin{bmatrix} \operatorname{diag}(\boldsymbol{s}_{c1}) - \boldsymbol{s}_{c1}'\boldsymbol{s}_{c1} & \mathbf{O} & \cdots & \mathbf{O} \\ \mathbf{O} & \operatorname{diag}(\boldsymbol{s}_{c2}) - \boldsymbol{s}_{c2}'\boldsymbol{s}_{c2} & \cdots & \mathbf{O} \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{O} & \mathbf{O} & \cdots & \operatorname{diag}(\boldsymbol{s}_{cd}) - \boldsymbol{s}_{cd}'\boldsymbol{s}_{cd} \end{bmatrix}. \qquad (58)$$