A rapid algorithm for inferring latent mixture structure in replicate social science data

An Honors Paper for the Department of Mathematics

By Nathan Joseph Kellerman

Project Director Jack O'Brien

Bowdoin College, 2026

Contents

Abstract

We utilize the finite mixture model, a common parametric distributional approach in the natural sciences, to infer latent populations in high-dimensional longitudinal referendum voting data. Building upon a previous Bayesian model which uses Dirichlet-multinominal components to infer voting blocs, we propose a custom Expectation-Maximization algorithm to quickly approximate similar results. Through a series of simulation studies, we demonstrate the empirical speed of the algorithm, as well as its inferential quality across common parameter sets. A first case study of the US state of Maine shows that the algorithm accurately recovers similar voting blocs as the Bayesian analysis in a fraction of the time. Our second case study of Swiss municipalities—previously computationally inaccessible to the Bayesian method—reveals a distinct latent clustering which aligns with historical divisions of language and religion across time and space. We conclude by discussing how this computationally quick approximation can be used more generally by social scientists to explore high-dimensional count data.

Acknowledgments

First and foremost, I would like to thank Professor Jack O'Brien for his thoughtful guidance and support through this research and writing process, and for cultivating my excitement for statistical thought across disciplines.

It has been a privilege to learn alongside so many remarkable peers and mentors in the Bowdoin Mathematics Department, and I am grateful to this academic family for the role it has played in my life over these past four years.

I am also sincerely grateful to my friends and family for their unwavering support throughout my time at Bowdoin, and for their continued faith in the work still ahead.

Chapter 1

Introduction

This chapter serves to sharpen our definitions of voting ideology, introduce the relevant statistical literature to our political science analysis, and detail the mathematics of finite mixture models. When stating a definition to be referred back to later in the manuscript, we use **bolded** text.

1.1 Statistically modeling latent voting ideologies

1.1.1 Assumptions about ideology

We first assume that when polling groups of individuals in a stable government system, the electorate's[1] voting results manifest from a finite number of different political cultures. As such, we choose to formally call these voting blocs **ideologies**, which we define as latent, consistent sets of principles and beliefs that guide political engagement. We construct this empirical assumption very carefully, as it is not clear at this juncture if the basis for these principles and beliefs is strictly cultural, linguistic, religious, regional, political, economic, or most likely some combination of these six. Once accepting this assumption, we think to pose the natural questions:

1. In a given voting population, how many distinct ideologies exist?
2. How is each locale (i.e. town, municipality) in this population mixed across ideologies?
3. What parameters define these ideologies?

One might imagine that quick answers to these questions would greatly benefit a social scientist deciding whether or not "interesting" hidden structures exist in their voting data. We therefore propose in this manuscript a modeling approach and rapid algorithm for social scientists to quickly examine referendum voting data and understand how relevant populations are partitioned across time and space by ideology.

At a high level, we can think of our model and subsequent inference as a function from **voting feature space** to **political latent space**. This function takes referendum voting data as an input and outputs a set of parameters to describe ideologies as defined above.

1.1.2 The machine of referendum voting

The nature of referendum voting is itself very direct. While votes cast at elections are usually made in relation to a particular political party or candidate, referendum votes target specific issues within communities and only require a "yes" or "no" answer. In comparing the two, we might expect the mixture of ideologies informing referendum votes to transcend party

[1] An electorate is defined as the total body of people entitled to vote.

lines and perhaps reveal an interesting pattern within political latent space. To emphasize the direct, specific nature of these questions, we provide the following referendum ballot measures from the US State of Maine in 2016.

Referendum ballot measure
Q_1: Legalize marijuana for personal use.
Q_2: 3% tax on household income over $\$200,000$.
Q_3: Background checks for gun sales and transfers.
Q_4: Increase minimum wage to \$12 per hour by 2020.
Q_5: Establish ranked-choice voting.
Q_6: \$100 million in bonds for transportation projects.

Table 1.1: Polled referendum questions posed to Maine residents in 2016.

The processes which initiate and facilitate referendum voting vary from country to country, but we choose to detail in brief the particularities of referendum in the US state of Maine, as well as the country Switzerland.

In Maine, there exist two common forms of referendum: citizen-initiated ballot measures (in which citizens have the power to initiate state statutes and/or veto an enacted law) and legislative referrals (in which citizens vote their approval of measures passed by the Legislature). In our case study, we will treat these forms of referendum as the same, although the former is much more common than the latter in our data [3].

The Switzerland system of direct democracy has a similar structure, with both citizen-initiated and government-initiated referendum voting taking place to both propose and repeal existing laws. To be explored more in-depth in another case study, we note a varied system of how these votes come to fruition at different scales of government and country territories [15].

1.1.3 Relevant statistical literature

The practice of modeling voting patterns (sometimes referred to as blocs) is common in the political science literature, with traditional statistical methods being generalized regression, ecological regression, and latent class analysis [4] [13] [17].

Oftentimes, however, these modeling approaches do not solely concern themselves with aggregated voting data, and instead utilize some demographic or reported information about individuals to make inference about groups of similar voters. As such, the work in this manuscript is largely motivated by [9], in which the authors built a series of mixture models to identify voting blocs within the 1997 Irish presidential electorate using ranked-choice voting (RCV) data.

Since RCV data is not readily available in many government systems, we seek to generalize this approach to aggregated count data (as observed in voting referendum). A proposed solution to this problem is detailed in [18], in which the author developed a birth/death Monte-Carlo Markov Chain approach to infer voting blocs through a Bayesian mixture model with Dirichlet-multinominal components. When applied to referendum data in the US state

of Maine, this model recovered interpretable voting blocs and raised pertinent questions about polarization of ideology across time. Unfortunately, this Bayesian model has computational limitations and cannot be realistically leveraged by political scientists for a timely analysis of latent groups in large datasets. As such, we imagine the following gradient of statistical techniques that a social scientist might choose from when estimating latent voting blocs. The work in this manuscript thus serves as a complement to the Bayesian method, in that it procures a quick yet reliable representation of mixture structure in these data contexts.

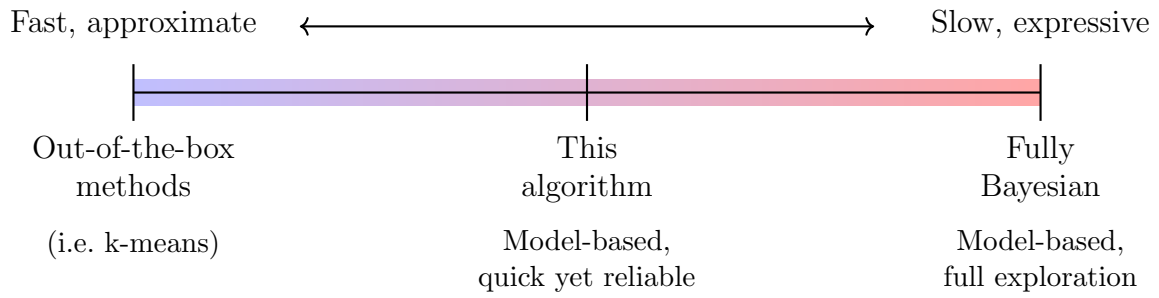


Figure 1.1: Gradient foregrounding the tradeoff between computational speed and statistical richness, in the context of finding latent populations in social science data.

1.2 Mathematical background

In this section we detail the mathematical background needed to understand how mixture models are specified. We conclude with a scaffolded outline of the manuscript.

1.2.1 Mixture models

Mixture models are commonly used to infer the distributions of hidden populations in data, and in the literature have been used by researchers in empirical fields to answer questions such as:

1. How can one probabilistically model taxa frequencies in microbial metagenomics data? [12]
2. To what extent can the DNA of admixed populations explain the rise and fall of human populations? [11]
3. How can one uncover and understand an underlying set of topics from a series of text documents? [5]

In simple terms: if a population P is divided into K heterogeneous subgroups yet we only observe the random variable Y, a mixture model allows us to model the distribution of Y without knowing how P is partitioned into its K groups.

To describe this process with better notation, suppose that observed counts Y arise from a **finite mixture distribution**. We will define each $p_k(\mathbf{y})$ as a **component density**

which arises from some well-behaved distribution family $\mathcal{T}(\vec{\theta})$ with component parameters $\vec{\theta}_1, \ldots, \vec{\theta}_K$. Then

$$Y \sim \lambda_1 \mathcal{T}(\vec{\theta}_1) + \cdots + \lambda_K \mathcal{T}(\vec{\theta}_K)$$

with probability density

$$p(\mathbf{y}) = \lambda_1 p_1(\mathbf{y}) + \cdots + \lambda_K p_K(\mathbf{y}),$$

for mixture weights $\lambda_k \geq 0$ and $\sum_{k=1}^{K} \lambda_k = 1$. This construction fits in nicely with our definition of ideology. With this in mind, we note the following parameters required to explicitly define this model:

- K, the number of heterogenous subgroups,
- λ_k, the mixture weights of each component density,
- $p_k(\cdot)$, the component densities which make up the mixture, and
- $z_i \in \{1, \ldots, K\}$, the latent assignment that describes the membership of data point i to a unique mixture component.

Remark 1.1. For our purposes, we will mostly use the Beta-binomial distributional family.

Remark 1.2. Such mixture distributions are often multi-modal, depending on the mixture weights λ_k and how similar the parameters $\vec{\theta}_k$ are across component densities.

In a nutshell, the finite mixture model in this context will allow us to capture unobserved heterogeneity caused by an unknown categorical variable, namely ideology [8].

1.2.2 Likelihoods and frequentist vs Bayesian frames

Given this work places a frequentist spin on a previously established Bayesian method, we briefly discuss the conceptual differences between the inference schemes. A clear way to understand frequentist statistics lies in the name itself: frequency. In this more traditional statistical frame, probability is defined from the perspective of repeating the same experiment many times under ideal circumstances, and then examining proportions of occurrence. This outlook prompts the core question of "how likely is my data $\mathcal{D}$ given my parameters θ," leading to an examination of $\mathbb{P}(\mathcal{D}|\theta)$. This construction gives rise to the likelihood function $\mathcal{L}$, which describes the probabilistic parameters θ which best explain the empirical data $\mathcal{D}$ at its maximum value. Naturally, this estimated set of parameters is a point estimate of a complex probabilistic structure and does not invoke any notions of the experiment designer's belief within the system.

Bayesian statistics on the other hand inverts this supposition, and examines the probability of specified parameters θ, conditional upon the data $\mathcal{D}$. This is done through the specification of prior distributions of the model's parameters, which are then "morphed"

curvature-wise across some inference process by the likelihood $\mathbb{P}(\mathcal{D}|\theta)$ into some posterior distribution $\mathbb{P}(\theta|\mathcal{D})$. This process is beautifully described by Bayes' formula

$$\mathbb{P}(\theta|\mathcal{D}) = \frac{\mathbb{P}(\mathcal{D}|\theta) \cdot \mathbb{P}(\theta)}{\mathbb{P}(\mathcal{D})}.$$

This naturally gives the modeler a choice to specify such prior distributions on their parameters, the choice of which introduces subjectivity into the scientific process of inference.

Notably, the computational challenge of estimating posterior distributions is a limiting factor, since the inference process explores the entire parameter space. This challenge is especially pertinent when working with complex hierarchical models and high-dimensional data.

As described in [10], reconciling between aleatory probability (frequencies) and epistemic probability (beliefs) in this way is a difficult experiment design task. Of course, these ontological differences carry with them computational implications, which is a large motivation for this work. As foregrounded in Figure 1.1, since our intent is computational speed, we carry forth with a maximum likelihood procedure on our mixture model parameters as specified. While this approach does not have a Bayesian-level quantification of uncertainty—i.e. point estimates of optimal parameter choices rather than posterior distributions of said parameters—we show in this manuscript its inferential power and ability to detect meaningful patterns in real political science data.

1.3 Outline of paper

We begin the manuscript with a thorough description of our data notation in the context of a Beta-binomial mixture model. We introduce in this section a toy example describing referendum among college professors, which does most of the conceptual heavy lifting needed to understand our procedure. We then scaffold this example into the next section, which describes a novel Expectation-Maximization (E-M) inference scheme. After explaining the E-M algorithm in the context of our experiment design, we describe the mathematical analysis of its convergence and explain the model thoroughly in pseudo-code. We then discuss the nuances of interpreting this model before detailing simulation study results which test the algorithm's inferential ability and computational speed with two data-generating-processes (DGPs) across many common parameter sets. We finally showcase the model's performance through an analysis of voting blocs in two case studies, of Maine and Switzerland, the latter of which was previously inaccessible to Bayesian methods due to its high dimension. We finish with a discussion about the tradeoffs of writing a rapid algorithm, as well as future directions concerning the application of this model to other social science contexts.

Chapter 2

Data and Model

In the following three subsections, we describe the particularities of referendum voting data, as well as how we choose to model said data. The majority of this manuscript's notation is summarized in Table 2.1, which will be referenced in all remaining chapters, including the Appendix.

2.1 Aggregate count data

Our data is comprised entirely of voter responses to referendum ballot measures, which conveniently manifest as binary "yes" or "no" choices at the level of the individual. An important feature of this data is that it is aggregated at the level of locales (i.e. county, municipality); specifically, we do not have access to individual level voting behavior. To motivate the connection between the voting process and our data, we initialize the following toy example to build upon later in the manuscript.

Example 2.1. Bowdoin College professor pedagogy

Consider Bowdoin College, which we suppose has 200 faculty, each belonging to one of 33 unique academic departments. To each faculty member in each department, we propose the following statements and record each professor's yes or no answer.

1. I train my students for a professional workplace.
2. Learning should always be difficult.
3. All students should pursue undergraduate research.

After collecting individual responses within each department for Q_1 through Q_3, we sum our yes and no counts per department per question and are yielded the following data structure.

	Q_1, Yes	Q_1, No	Q_2, Yes	Q_2, No	Q_3, Yes	Q_3, No
Department 1	$y_{1,1}$	$n_{1,1}$	$y_{1,2}$	$n_{1,2}$	$y_{1,3}$	$n_{1,3}$
Department 2	$y_{2,1}$	$n_{2,1}$	$y_{2,2}$	$n_{2,2}$	$y_{2,3}$	$n_{2,3}$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\vdots$
Department 33	$y_{33,1}$	$n_{33,1}$	$y_{33,2}$	$n_{33,2}$	$y_{33,3}$	$n_{33,3}$

Thinking in a probabilistic frame, we can easily turn these yes/no counts into number of successes and number of trials, per department per question. In fact, in this toy example we can plot out the distributional shapes of each question's observed success counts across all polled faculty.

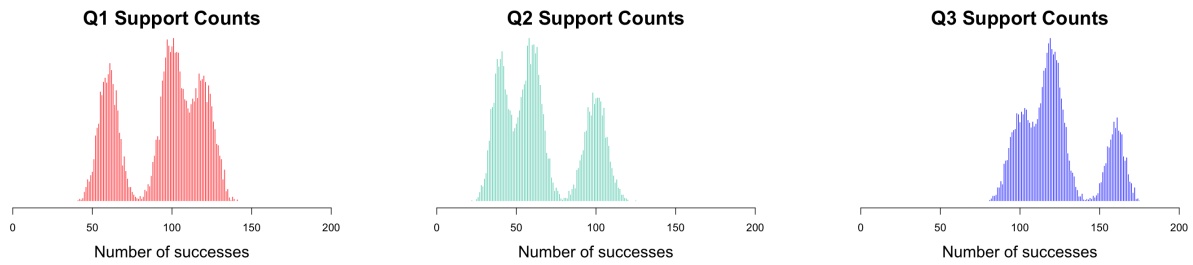


Figure 2.1: Mixture distributions of success probabilities for pedagogy toy example questions.

We notice these distributions to appear somewhat Binomial in shape, but perhaps with an element of over-dispersion in different questions. This is how real referendum voting data often reads, namely not symmetric around a single mode. Thinking forth to how we will model this data probabilistically within our mixture model, we will use the Beta-binomial distribution, that can be thought of as a Binomial distribution with an additional dispersion parameter. The upshot is that this distributional flexibility should allow our model to explain increased variance within answers to certain questions, while still remaining true to the "number of successes" and "number of trials" lens.

With this explicit representation of our data written out, we can begin to imagine how a consistent set of latent pedagogical ideologies inform each faculty's set of answers. An experienced Bowdoin academician might for example detect the latent groups of professors who are "tradition-oriented" or "outcome-focused," but we aim to mathematically define such archetypes.

Remark 2.1. While this toy example assumes full participation of all Bowdoin faculty in each referendum response, real-life aggregate count data does not usually possess this quality.

Remark 2.2. In addition to our 33 departments, 3 polled questions, and the counts themselves, the total number of voters per department per question is a relevant parameter to how we interpret this data.

To motivate our later case studies in light of this toy example, we note the following longitudinal nature and size of our analyzed data.

Case Study	**# of Locales**	**# of Questions**	**Polling Year Range**
Pedagogy Toy Example	33	3	`NA`
Maine Referendum Data	423	70	$2008 - 2024$
Swiss Referendum Data	2048	390	$1981 - 2026$

While our later case studies deal with country or state-scale political referendum and not the pedagogical perspectives of college professors, the modeling implications of working with aggregated count data across time and space remain largely the same. In its simplest form, data of this type serves as a temperature of opinion across different partitions of space, across time.

2.2 Data notation

For the remainder of this manuscript, we will refer to the following notation table for our aggregated count data and resulting Beta-binomial mixture model.

Notation	Interpretation
Q	Number of referendum questions
$q = 1, \ldots, Q$	Index over referendum questions
M	Number of locales
$i = 1, \ldots, M$	Index over locales
K	Number of latent ideologies
$k = 1, \ldots, K$	Index over latent ideologies
N_{iq}	Number of voters per town, per question
$\mathbf{x}_{iq} = (y_{iq}, n_{iq})$	Aggregated yes/no vote totals, per locale per question
μ_{kq}	Beta-binomial mean parameter per ideology, per question
ν_{kq}	Beta-binomial dispersion parameter per ideology, per question
$\vec{\theta}_k$	Collection of Beta-binomial parameters across ideology k
θ	Collection of all Beta-binomial parameters
λ_k	Global mixture weight across M locales per ideology k
$z_i \in \{1, \ldots, K\}$	Latent assignment z per locale i

Table 2.1: Parameters of Beta-binomial mixture model.

Remark 2.3. When considering the raw input of count data as previously described, Q, M, N_{iq}, and $\mathbf{x}_{iq}$ are observed, while $K, \mu_{kq}, \nu_{kq}, \lambda_k$, and z_i are not observed and serve as parameters to help describe the system of referendum voting. We will soon learn the most likely of these parameters conditional upon our count data, through an iterative optimization scheme.

Remark 2.4. It should be clear upon inspection that $\sum_{k=1}^{K} \lambda_k = 1$.

2.3 Beta-binomial mixture model

As originally described in [18] and further motivated in our toy example, we choose to use a Beta-binomial data likelihood to model our over-dispersed counts.

2.3.1 Choosing a Beta-binomial parameterization

The Beta-binomial (BB) distribution is most commonly used to describe success counts with the α, β parameterization, with probability density function

$$\begin{aligned} f_{\text{BB}}(x|n,\alpha,\beta) &= \int_0^1 \text{Binomial}(x|n,p) \cdot \text{Beta}(p|\alpha,\beta)\,\mathrm{d}p \\ &= \binom{n}{x} \frac{1}{B(\alpha,\beta)} \cdot \int_0^1 p^{x+\alpha-1}(1-p)^{n-x+\beta-1}\,\mathrm{d}p \\ &= \binom{n}{x} \frac{B(x+\alpha, n-x+\beta)}{B(\alpha,\beta)}. \end{aligned}$$

In this construction, x is the number of successes in n trials with success probability p. By integrating the success probability p over a Beta distribution (which is parameterized by α and β), we have effectively made the Binomial distribution more flexible to allow for additional dispersion.

Remark 2.5. The Beta function defined above can be written as

$$B(\alpha,\beta) = \int_0^1 t^{\alpha-1} \cdot (1-t)^{\beta-1}\,\mathrm{d}t.$$

Later in the manuscript, we elect to construct the Beta function $B(\cdot,\cdot)$ in terms of the Gamma function $\Gamma(\cdot)$:

$$B(\alpha,\beta) = \frac{\Gamma(\alpha)\cdot\Gamma(\beta)}{\Gamma(\alpha+\beta)}.$$

As described more completely in the Appendix, we choose to perform a change of variables procedure to have finer-grain control over the mean and dispersion of our modeled counts:

$$\begin{aligned} \mu &= \frac{\alpha}{\alpha+\beta} \\ \nu &= \alpha+\beta \end{aligned} \quad \Longrightarrow \quad \begin{aligned} \alpha &= \mu\cdot\nu \\ \beta &= (1-\mu)\cdot\nu \end{aligned}$$

To provide intuition for why this change of variables will help the interpretability of our model, we plot out the following distributional shapes.

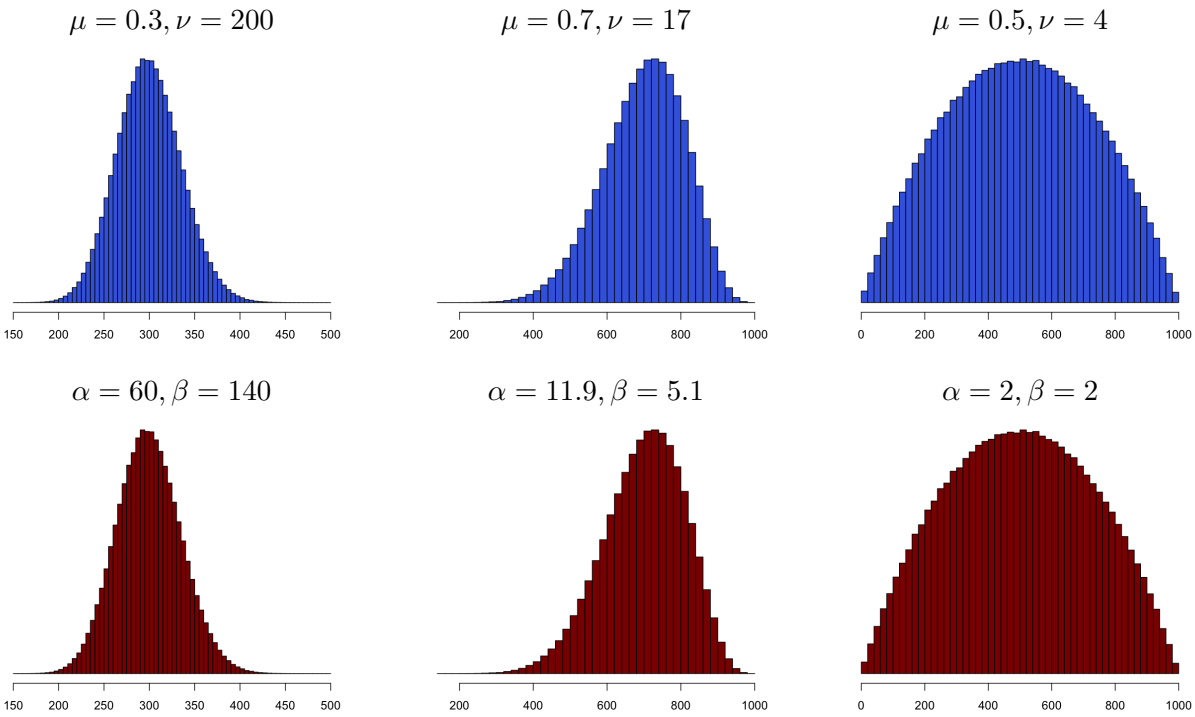


Figure 2.2: Varied Beta-binomial distributional shapes with $n = 1000$.

Formally, we have written the above probability densities as

$$f_{\text{BB}}(x|n, \mu, \nu) = \binom{n}{x} \frac{B(x + \mu \cdot \nu, n - x + (1 - \mu) \cdot \nu)}{B(\mu \cdot \nu, (1 - \mu) \cdot \nu)}.$$

Remark 2.6. An additional upshot of using the Beta-binomial distribution is that it allows for limited distributional bimodality.

2.3.2 Data generating process

At this stage we understand the data structure and distributional shape of aggregated count data, but need a way to reproduce it probabilistically in the language of our Beta-binomial mixture model. Drawing heavily from our assumptions about ideology, we propose the following plate diagram to explain the hierarchy and underlying data generating process (DGP) of our model.

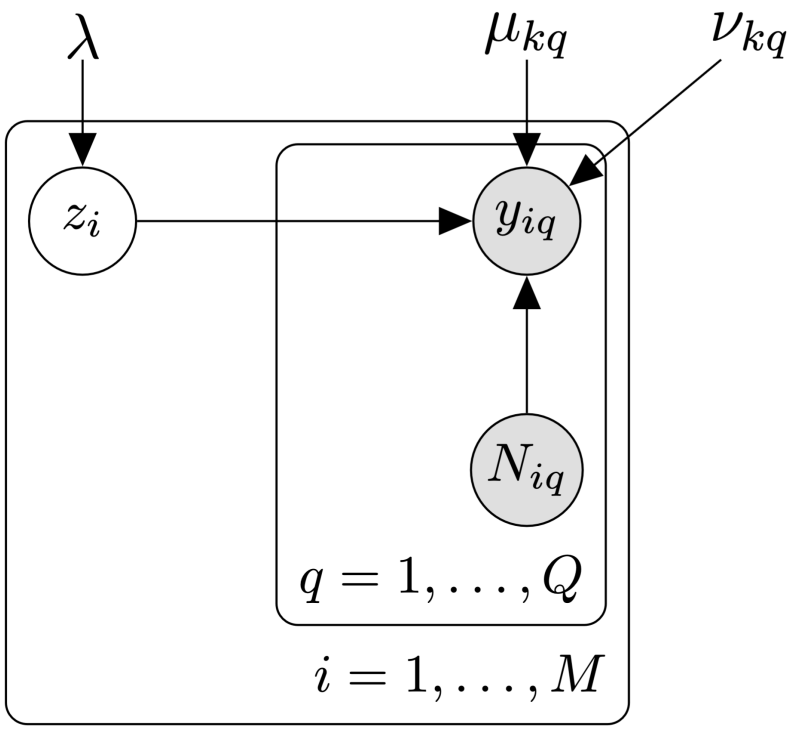


Figure 2.3: Plate diagram describing DGP of Beta-binomial mixture model.

In the above plate diagram, shaded shapes are observed and unshaded ones latent. Recalling our indexing from Table 2.1, we outline the following procedure to describe voting in our system.

1. Each town i is assigned to one of the K latent ideologies from a probability vector λ of length K.

2. According to its assigned latent group $z_i \in \{1, \ldots, K\}$, each town will procure a number of "yes" counts to Q referendum questions.

3. The "yes" counts y_{iq} are modeled as Beta-binomial with size N_{iq}, mean parameter μ_{kq}, and dispersion parameter ν_{kq}.

Remark 2.7. In our later simulation studies, we detail two primary sampling procedures for how z_i is controlled by λ; namely, (1) a mixture model where all M municipalities are allocated by the same global λ to latent ideologies and (2) a mixture model where each municipality i has its own probability vector λ_i controlling how individuals are allocated to latent ideologies. We will also discuss the modeling implications involved with choosing (1) or (2).

We briefly think back to our three motivating questions from the Introduction chapter, but this time in the context of our refined notation.

1. Across our M locales, how many unique ideologies, K, exist?

2. How is each locale i mixed by these K ideologies?

3. What parameters define these K ideologies?

Naturally, we turn to specifying our model in order to construct an algorithm which learns λ, z_i, μ_{kq}, and ν_{kq}.

2.3.3 Specifying the model

We employ a Beta-binomial data likelihood for our count data, and thus let each observation across questions $\mathbf{x}_{iq} = (y_{iq}, n_{iq})$ be generated by a finite mixture with K components. Using our latent assignment notation $z_i \in \{1, \dots, K\}$, we write that each response

$$y_{iq}|z_i = k \sim \text{BetaBinomial}(N_{iq}, \mu_{kq}, \nu_{kq}).$$

We thus write the probability density per ideology per question as

$$f_{kq}(y_{iq}) = \text{BetaBinomial}(y_{iq}|N_{iq}, \mu_{kq}, \nu_{kq}),$$

meaning the probability across all questions per ideology is

$$\pi(\vec{\theta}_k) = \prod_{q=1}^{Q} f_{kq}(y_{iq}).$$

We then simply sum across our K voting ideologies to obtain the mixture distribution

$$(y_{iq}, n_{iq}) \sim \sum_{k=1}^{K} \lambda_k \pi(\vec{\theta}_k),$$

for mixture weights $\lambda_k \geq 0$ and $\sum_{k=1}^{K} \lambda_k = 1$. We then write our marginal as

$$\sum_{k=1}^{K} \Big(\lambda_k \prod_{q=1}^{Q} \text{BetaBinomial}(\mathbf{x}_{iq}|N_{iq}, \mu_{kq}, \nu_{kq}) \Big),$$

and our model log likelihood

$$\log \mathcal{L}(\vec{\theta}; y) = \sum_{i=1}^{M} \log \Big\{ \sum_{k=1}^{K} \lambda_k \Big(\prod_{q=1}^{Q} \text{BetaBinomial}(\mathbf{x}_{iq}|N_{iq}, \mu_{kq}, \nu_{kq}) \Big) \Big\}.$$

Remark 2.8. This should be clear by the nature of Example 2.1, but by assuming a somewhat equivalent hypothesis to our "ideologies of culture" in the context of other count data, this model and subsequent inference may be applied in fields beyond political science.

Chapter 3

Inference

The following two subsections describe our model's inference procedure of the above specified Beta-binomial mixture model.

3.1 Modeling latent structures

As described in the Introduction chapter, we aim to find the most likely set of model parameters θ which describe our data $\mathcal{D}$, which we will break down into observed data X and unobserved latent assignments $\mathcal{Z}$. We recall that our observed data log likelihood $\mathcal{L}(\theta; X)$ is defined as

$$\sum_{i=1}^{M} \log \Big\{ \sum_{k=1}^{K} \lambda_k \Big(\prod_{q=1}^{Q} \text{BetaBinomial}(\mathbf{x}_{iq} | N_{iq}, \mu_{kq}, \nu_{kq}) \Big) \Big\}.$$

In order to find each maximum likelihood estimator $\widehat{\lambda}_k$, $\widehat{\mu}_{kq}$, and $\widehat{\nu}_{kq}$ to our model, we would traditionally take partial derivatives of the observed data log likelihood $\mathcal{L}(\theta; X)$ with respect to each parameter, set this expression equal to zero, and solve for said parameter. If we start to write out this process

$$\frac{\partial}{\partial \mu_{kq}} \left[\sum_{i=1}^{M} \log \Big\{ \sum_{k=1}^{K} \lambda_k \Big(\prod_{q=1}^{Q} \text{BetaBinomial}(\mathbf{x}_{iq} | N_{iq}, \mu_{kq}, \nu_{kq}) \Big) \Big\} \right],$$

we quickly notice that since the summation over K does not allow our log function to "break up" the product over Q, our solvable parameters will be entangled. Stated simply, finding $\arg\max \log \mathcal{L}(\theta; X)$ with respect to the desired parameters is difficult in the context of our problem.

As a solution, we can "pretend" to know latent assignments $\mathcal{Z}$ and thus examine the complete data log likelihood $\mathcal{L}(\theta; X, \mathcal{Z})$, defined as

$$\sum_{i=1}^{M} \sum_{k=1}^{K} z_{ik} \left[\log \lambda_k + \sum_{q=1}^{Q} \log \text{BetaBinomial}\big(X_{iq} \mid N_{iq}, \mu_{kq}, \nu_{kq}\big) \right].$$

Critically, the "log of a sum" problem goes away in this construction, meaning finding $\arg\max \log \mathcal{L}(\theta; X, \mathcal{Z})$ is an easier task. Now, we need an optimization algorithm to deal with the process of "guessing" latent assignments $\mathcal{Z}$.

As such, we seek out a numerical routine which pairs the smooth nature of MLE optimization with the discrete structure of latent variables. This is where we implement a version of the Expectation-Maximization algorithm.

3.2 Expectation-Maximization algorithm

At its core, the Expectation-Maximization (E-M) algorithm is an iterative process which switches between (1) updating our model's latent assignments z_i and (2) updating our desired parameters $\widehat{\lambda}_k$, $\widehat{\mu}_{kq}$, and $\widehat{\nu}_{kq}$, until a convergence criterion is satisfied. The convergence of this process is a consequence of the concavity of the logarithm function, which we utilize through constructing a lower bound $\mathcal{Q}(\theta, \theta_{old})$ of the likelihood function. In section 3.2.2, we walk through the necessary inequalities to show that $\mathcal{L}(\theta; X) \geq \mathcal{Q}(\theta, \theta_{old})$ at each iteration of our algorithm. For now, we will examine the auxiliary function

$$\mathcal{Q}(\theta, \theta_{old}) = \sum_i \sum_k \mathbb{P}(Z_i = k | x_{iq}, \theta_t) \log \frac{\mathbb{P}(X_{iq} = x_{iq}, Z_i = k|\theta)}{\mathbb{P}(Z_i = k | x_{iq}, \theta_t)}.$$

In plain english, the quantity $\mathbb{P}(Z_i = k|x_{iq}, \theta_t)$ is the probability that locale i is in cluster k, given our observed count data per locale per question, x_{iq} and the knowledge of our Beta-binomial parameters at time t, θ_t. In plainer english, for each locale, this quantity is a probability vector of length K. This quantity is quite important, and we thus define the **responsibility** vector for locale i as

$$\gamma_{ik} = \mathbb{P}(Z_i = k|x_{iq}, \theta_t) = \frac{\lambda_k \prod_{q=1}^{Q} \text{BB}(x_{iq}|N_{iq}, \mu_{kq}, \nu_{kq})}{\sum_{k=1}^{K} \lambda_k \prod_{q=1}^{Q} \text{BB}(x_{iq}|N_{iq}, \mu_{kq}, \nu_{kq})}.$$

We now describe one iteration of the E-M algorithm more precisely:

- **E-step:** Compute responsibilities γ_{ik} for each locale i.
- **M-step:** Find the updated parameters which maximize $\mathcal{Q}(\theta, \theta_{old})$ and improve the likelihood $\mathcal{L}(\theta; X)$. Formally, obtain

$$\begin{aligned} &\max_\theta \sum_i \sum_k \gamma_{ik} \log \frac{\mathbb{P}(X_{iq} = x_{iq}, Z_i = k|\theta)}{\gamma_{ik}} \\ &= \max_\theta \sum_i \sum_k \gamma_{ik} \log \mathbb{P}(X_{iq} = x_{iq}, Z_i = k|\theta), \end{aligned}$$

  since the responsibilities term in the denominator does not depend on our parameters due to rules of the logarithm function.

After one E-M iteration, we have an updated set of parameters which can be used to compute γ_{ik}, with which we repeat the whole process until our likelihood improves only epsilonically. This process is detailed in the below flow diagram.

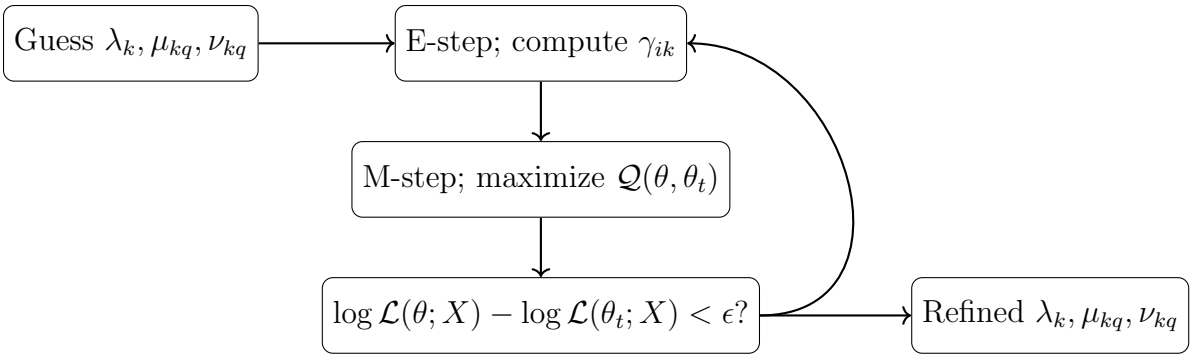


Figure 3.1: E-M algorithm flow diagram.

Remark 3.1. We note that this E-M procedure yields local maxima of the likelihood function $\mathcal{L}(\theta; X)$, not necessarily global maxima.

Through this construction, we have made our MLE task more manageable but still need to optimize the function $\mathcal{Q}$ with respect to μ_{kq}, ν_{kq}, and λ_k, in order to fully specify the M-step for our Beta-binomial mixture model.

To obtain $\widehat{\lambda}_k$, we differentiate $\mathcal{Q}$ with respect to λ_k and notice the cancellations

$$\frac{\partial}{\partial \lambda_k}[\mathcal{Q}] = \sum_{i=1}^{M} \sum_{k=1}^{K} \gamma_{ik} \log \lambda_k.$$

Since this constrained optimization does not depend on our data likelihood, we are free to mimic the common procedure involving Lagrange multipliers detailed in [21] to obtain that

$$\boxed{\widehat{\lambda}_k = \frac{1}{M} \sum_{i=1}^{M} \gamma_{ik}}.$$

This result should make intuitive sense, as each mixture weight is the average responsibility per ideology k.

Thinking towards μ_{kq} and ν_{kq}, we recall that our data likelihood is Beta-binomial, meaning our density function to optimize over is constructed of Beta functions. As such, there does not exist a nice, closed form update rule for these parameters. We thus choose to approximate these values using Newton's method.

Simply put, we use the score function (first derivative) and Hessian (second derivative) to "step around" the function $\mathcal{Q}$ with parameter updates[1]

$$\mu_{kq}^{\text{new}} = \mu_{kq}^{\text{old}} - \frac{\mathcal{Q}'(\mu_{kq}^{\text{old}})}{\mathcal{Q}''(\mu_{kq}^{\text{old}})},$$

$$\nu_{kq}^{\text{new}} = \nu_{kq}^{\text{old}} - \frac{\mathcal{Q}'(\nu_{kq}^{\text{old}})}{\mathcal{Q}''(\nu_{kq}^{\text{old}})}.$$

[1] In practice, we must place certain damping conditions on this procedure to ensure numerical stability in the presence of sparse responsibility vectors.

Remark 3.2. In practice, we bound this procedure to a maximum of 50 iterations, with a convergence criterion $\epsilon = \texttt{1e-8}$. To encourage convergence, we initialize our guess for $\widehat{\mu}_{kq}$ at the Binomial MLE estimate. An initial guess of $\widehat{\nu}_{kq} = 100 \; \forall \, k, q$ also speeds up convergence in practice.

The full derivations for these score functions and Hessians are included in the Appendix. For completeness, we list the resulting expressions across each question q and ideology cluster k. As discussed in the Appendix, $\psi(\cdot)$ and $\psi_1(\cdot)$ are the diagamma and trigamma functions, respectively.

$$\frac{\partial}{\partial \mu_k}[\mathcal{Q}_k] = \nu_k \cdot \sum_{i=1}^{M} \gamma_{ik} \Big(\psi(x_i + \mu_k \cdot \nu_k) - \psi(n_i - x_i + (1-\mu_k) \cdot \nu_k) - \psi(\nu_k \cdot \mu_k) + \psi((1-\mu_k)) \cdot \nu_k \Big)$$

$$\frac{\partial^2}{\partial \mu_k^2}[\mathcal{Q}_k] = \nu_k^2 \cdot \sum_{i=1}^{M} \gamma_{ik} \Big(\psi_1(x_i + \mu_k \cdot \nu_k) + \psi_1(n_i - x_i + (1-\mu_k) \cdot \nu_k) - \psi_1(\nu_k \cdot \mu_k) - \psi_1((1-\mu_k) \cdot \nu_k) \Big)$$

$$\begin{aligned} \frac{\partial}{\partial \nu_k}[\mathcal{Q}_k] = \sum_{i=1}^{M} \gamma_{ik} \Big(& \mu_k \cdot \psi(x_i + \mu_k \cdot \nu_k) - \mu_k \cdot \psi(\mu_k \cdot \nu_k) - \psi(n_i + \nu_k) \\ & + (\mu_k - 1) \cdot \psi((1-\mu_k) \cdot \nu_k) \\ & - (\mu_k - 1) \cdot \psi(n_i - x_i + (1-\mu_k) \cdot \nu_k) + \psi(\nu_k) \Big) \end{aligned}$$

$$\begin{aligned} \frac{\partial^2}{\partial \nu_k^2}[\mathcal{Q}_k] = \sum_{i=1}^{M} \gamma_{ik} \Bigg(& \mu_k^2 \cdot \psi_1(x_i + \mu_k \cdot \nu_k) \\ & - \mu_k^2 \psi_1(\mu_k \cdot \nu_k) \\ & - \psi_1(x_i + \mu_k \cdot \nu_k + n_i - x_i + (1-\mu_k) \cdot \nu_k) \\ & + (\mu_k - 1) \cdot (1-\mu_k) \cdot \psi_1((1-\mu_k) \cdot \nu_k) \\ & - (\mu_k - 1) \cdot (1-\mu_k) \cdot \psi_1(n_i - x_i + (1-\mu_k) \cdot \nu_k) \\ & + \psi_1(\nu_k) \Bigg) \end{aligned}.$$

3.2.1 Visualizing the E-M algorithm

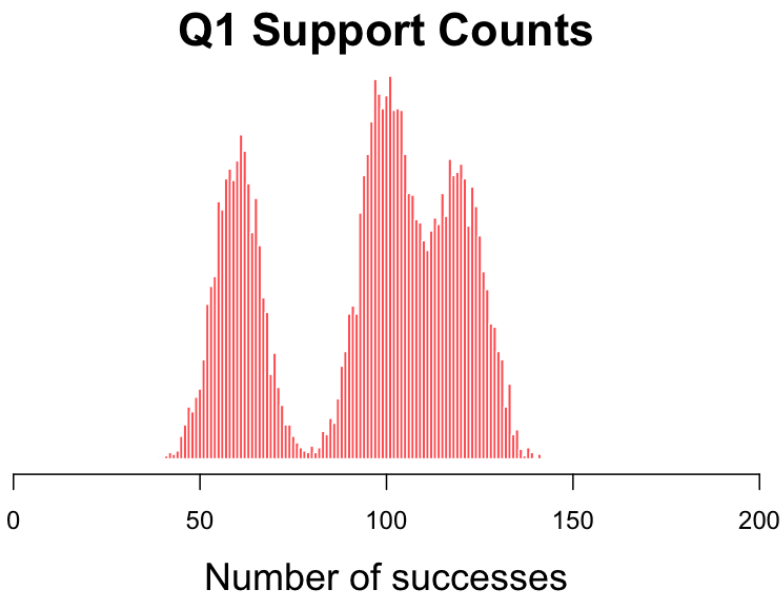
Recall Example 2.1 concerning Bowdoin professor pedagogy. We now seek to recover the parameters used to construct this toy example, using the above specified algorithm.

Example 3.1. Visualizing mixtures of professor pedagogy

We recall that our toy polling procedure involved asking 200 Bowdoin professors across 33 departments 3 "yes" or "no" questions, and then aggregating responses across departments for each question. The questions were as follows:

1. I train my students for a professional workplace.
2. Learning should always be difficult.
3. All students should pursue undergraduate research.

We now intend to visualize how the E-M algorithm infers specified mixture distributions. As such, we draw our attention to Q_1, which has the following distribution of success.



As is relatively clear in this clean example, we will set our model to $K = 3$. As per the above procedure, we now "guess" global mixture weights for these three Beta-binomial component densities, as well as the parameters which define them.[2] In the below pane, the grayed out background is the "true" data distribution, and the dotted lines indicate the shape of each component density at each iteration.

[2]While our full model uses a Beta-binomial likelihood, this toy example was simulated using a Binomial likelihood for ease of visualization.

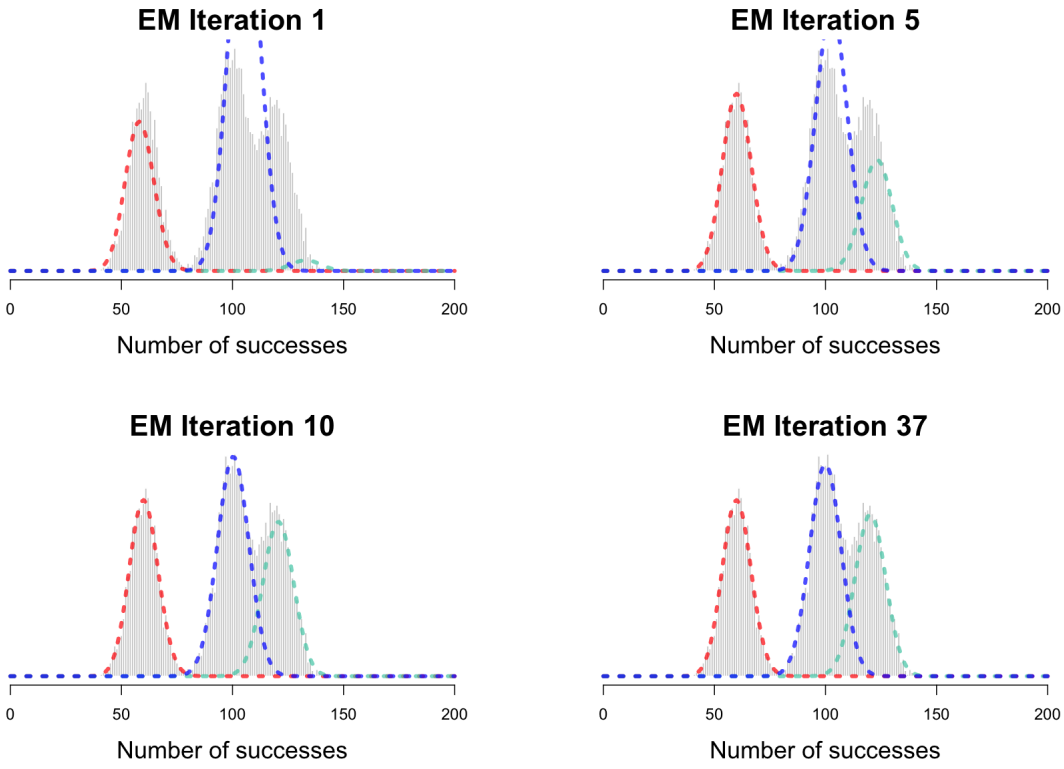


Figure 3.2: A visualization of how the E-M assigns probabilities to each component density, across iterations.

In this example, we see how the E-M algorithm slowly assigns probabilities to the "correct" density, eventually yielding us learned mixture weights λ_k, as well as distribution parameters. This is a common use of the E-M algorithm, namely inferring multi-modal distributional shapes in a singular dimension.

We begin to imagine how this approach extends beyond just one polled question, however. When examining the ideology structure across the independent Q_1 and Q_2, we can imagine the above visualizations at each iteration to be contour diagrams of probability mass, slowly being refined by the E-M to match the true distribution. Recall that our data for these two questions looks like the following[3], meaning their joint density can be represented by the accompanying 2D histogram (discrete contour).

[3]For the sharpness of this toy example, we sample voting responses across $M = 10000$ departments. This is done for the clarity of our contours.

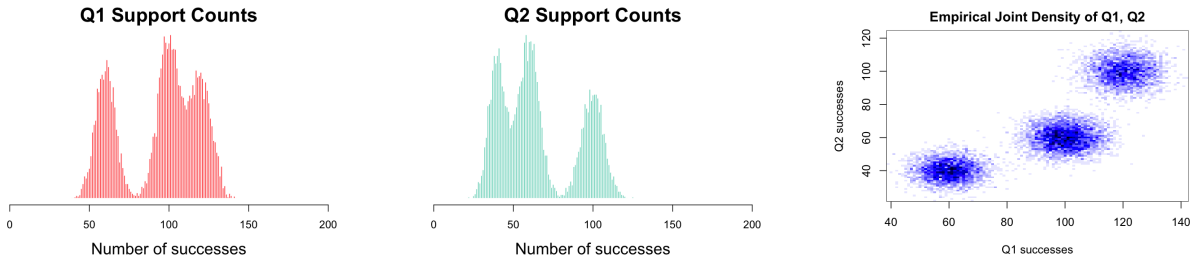


Figure 3.3: Distributions of success probabilities for pedagogy toy example questions.

In the below plots, the grayed contours represent observed probability mass, and the colored contours represent the E-M's guesses at each iteration.

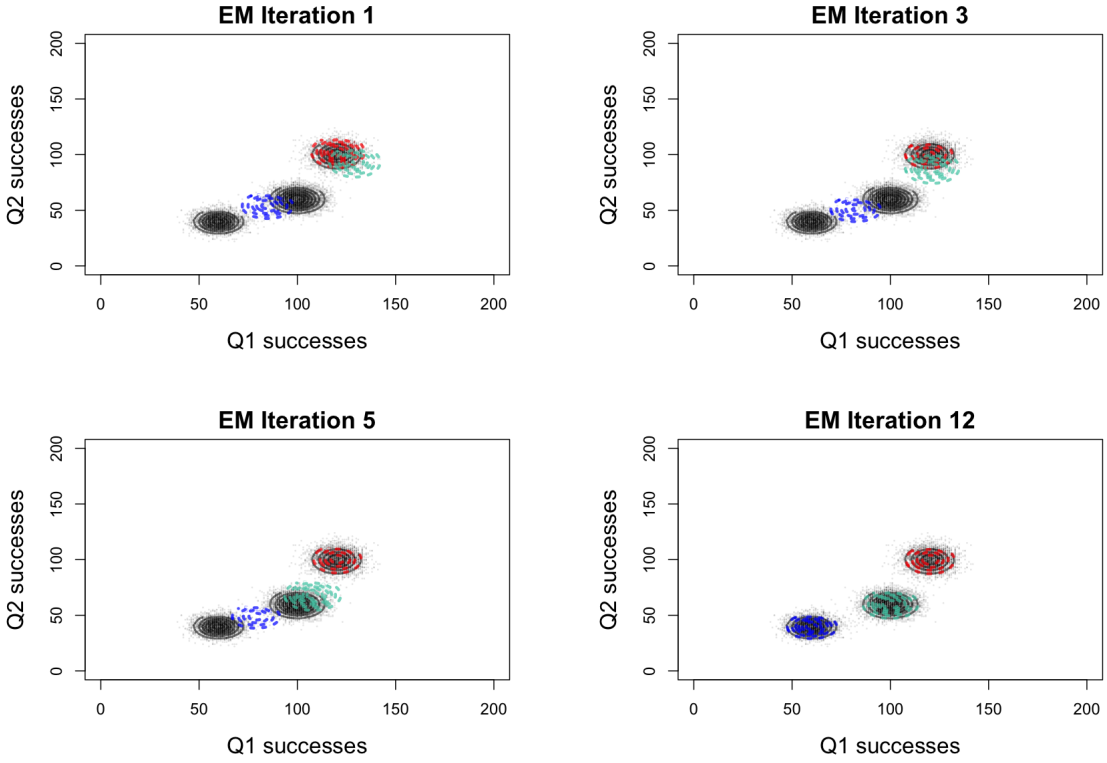


Figure 3.4: An extension of the above, in two dimensions.

While visualization is certainly more difficult, the extension to larger values of Q should be apparent. As such, this novel extension of the E-M algorithm across multiple questions Q should converge to local maxima of $\widehat{\lambda}_k$, $\widehat{\mu}_{kq}$, and $\widehat{\nu}_{kq}$.

Remark 3.3. As formalized in our later simulation study, the number of E-M iterations required for convergence tends to decrease when Q increases.

3.2.2 Mathematical analysis of convergence

We now aim to formally construct the auxiliary function $\mathcal{Q}(\theta, \theta_{old})$ from our observed data log likelihood, as well as examine why each step forward in our algorithm is a better estimate of likelihood. We first recall that our observed data log likelihood $\mathcal{L}(\theta; X)$ is defined as

$$\sum_{i=1}^{M} \log \Big\{ \sum_{k=1}^{K} \lambda_k \Big(\prod_{q=1}^{Q} \text{BetaBinomial}(\mathbf{x}_{iq} | N_{iq}, \mu_{kq}, \nu_{kq}) \Big) \Big\},$$

which we will rewrite for notation's sake as

$$\sum_i \log \sum_k \mathbb{P}(X_{iq} = x_{iq}, Z_i = k | \theta).$$

Letting the variable $t \in \mathbb{N}$ be discrete time steps in our algorithm, we choose to "multiply by 1"

$$\sum_i \log \sum_k \mathbb{P}(Z_i = k | x_{iq}, \theta_t) \cdot \frac{\mathbb{P}(X_{iq} = x_{iq}, Z_i = k | \theta)}{\mathbb{P}(Z_i = k | x_{iq}, \theta_t)}.$$

Looking carefully, we have constructed a weighted average, or expectation

$$\mathbb{E}_z = \sum_k \mathbb{P}(Z_i = k | x_{iq}, \theta_t).$$

So our likelihood expression is equivalent to

$$\sum_i \log \mathbb{E}_z \frac{\mathbb{P}(X_{iq} = x_{iq}, Z_i = k | \theta)}{\mathbb{P}(Z_i = k | x_{iq}, \theta_t)}.$$

Lemma 3.1. Jensen's Inequality [6] For any random variable X, if $g(X)$ is a convex function, then

$$\mathbb{E}[g(X)] \geq g(\mathbb{E}[X]).$$

Thinking back to our problem, if a function g is convex, then $-g$ is concave. Now let $-g(x) = \log(x)$. By Jensen's Inequality, we have that $-g(\mathbb{E}[X]) \geq -\mathbb{E}[g(X)] = \mathbb{E}[-g(x)]$ by linearity of expectation. Thus, we have that $\log(\mathbb{E}[X]) \geq \mathbb{E}[\log(x)]$[21]. Applying this to our expectation above,

$$\begin{aligned}
\mathcal{L}(\theta; X) &= \sum_i \log \mathbb{E}_z \frac{\mathbb{P}(X_{iq} = x_{iq}, Z_i = k | \theta)}{\mathbb{P}(Z_i = k | x_{iq}, \theta_t)} \\
&\geq \sum_i \mathbb{E}_z \log \frac{\mathbb{P}(X_{iq} = x_{iq}, Z_i = k | \theta)}{\mathbb{P}(Z_i = k | x_{iq}, \theta_t)} \\
&= \sum_i \sum_k \mathbb{P}(Z_i = k | x_{iq}, \theta_t) \log \frac{\mathbb{P}(X_{iq} = x_{iq}, Z_i = k | \theta)}{\mathbb{P}(Z_i = k | x_{iq}, \theta_t)} = \mathcal{Q}(\theta, \theta_t).
\end{aligned}$$

We have successfully shown that the auxiliary function $\mathcal{Q}(\theta, \theta_{old})$ as defined is a lower bound for our likelihood. It follows that maximizing this function $\mathcal{Q}$ across iterations will in turn procure a numerical approximation of our maximum likelihood estimators.

3.2.3 Iterative optimization algorithm

Given the above mathematics, we provide the following pseudo code outline of our algorithm.

Algorithm 1 E-M algorithm

Require: Success counts y_{iq}, total trials N_{iq}, number of ideologies K, randomly initialized model parameters θ, LLK tolerance ϵ, maximum E-M iterations T, `BOOL(learn.nu)`, `BOOL(verbose)`.

Outputs: Estimated parameters $\widehat{\theta}$

Initialize parameters $\theta^{(0)}$
Set $t \leftarrow 0$
while $t < T$ **do**
    E-step: Compute responsibilities vector using $\theta^{(t)}$
    M-step: Update parameters with Newton step to obtain $\theta^{(t+1)}$
    Compute convergence criterion
    if convergence criterion $< \epsilon$ **then**
        break
    end if
    $t \leftarrow t + 1$
end while
return $\widehat{\theta} \leftarrow \theta^{(t+1)}$

Remark 3.4. In practice, we often set our LLK tolerance $\epsilon =$ `1e-5` and our maximum E-M iterations $T =$ `200`.

Remark 3.5. The learning of our updated Beta-binomial dispersion parameter $\widehat{\nu}_{kq}$ is empirically slow with large datasets (i.e. large M, Q), so we give our algorithm the choice to accept a hard-coded ν_{kq}. From a modeling perspective, holding this set ν_{kq} still allows the algorithm to procure meaningful clusterings.

3.2.4 Interpreting model results

After running our algorithm, we are yielded the following table of outputs for interpretation.

Output	**Data type**
Learned global mixture weights λ_k	Prob. vector of length K
Learned Beta-binomial mean parameter μ_{kq}	Matrix of size $K \times Q$
Learned Beta-binomial dispersion parameter ν_{kq}	Matrix of size $K \times Q$
Learned responsibilities γ_{ik}	Matrix of size $M \times K$
Number of iterations to convergence, T_C	Positive integer
Likelihoods at each iteration	Vector of length T_C

Table 3.1: Outputs of the proposed E-M algorithm.

The interpretation of these first three model outputs should be clear in the context of our inference scheme, but it is worth taking extra care in how we understand the responsibilities vector. This vector is exactly a soft clustering of our locales $i = 1, \dots M$ into our $k = 1, \dots K$ ideologies. While in practice we often take the arg max of these assignments, it is important to remember this fact. Thinking back to our three motivating questions, we now see more clearly how these E-M outputs are fruitful.

1. Across our M locales, how many unique ideologies, K, exist?
    - Perform a model selection routine (AIC, BIC) across a range of K values.
2. How is each locale i mixed by these K ideologies?
    - Examine the responsibilities vector γ_{ik}.
3. What parameters define these K ideologies?
    - Examine each learned μ_{kq} and ν_{kq} value.

Remark 3.6. When comparing inference of the same model in a Bayesian frame as done in [18], responsibilities are markedly more dispersed across ideologies than with our E-M approach. We attribute this to the point estimate nature of our MLE approach, and keep this lack of uncertainty quantification in mind as we look at each case study.

3.2.5 Label switching problem

An important nuance hinted at in section 3.2.1 is how the colors identifying each cluster across different E-M runs (i.e. component labels) are not necessarily consistent. Formally, this is called the **label switching problem** [8]. It should be clear that while the labeling of locales z_i is hopefully self consistent across individual E-M runs, what is classified as Label 1 in one run is not necessarily guaranteed to be classified as Label 1 in the next.

This poses an issue for example in our simulation studies, specifically when we want to compare ground truth mixture parameters with our model's recovered ones. To step around label switching in this context, we choose to invoke an optimal matching of parameter outputs across runs.[4] Label switching also makes identifying changes in cluster structure across time difficult; namely when looking at two clusterings on a map in subsequent years, the coloring of clusters is not consistent.

[4]This is a version of the linear sum assignment problem, to be explored more completely in Section 4.3.

Chapter 4

Simulation Studies

In this chapter we detail a series of simulation studies[1] to explore the computational and inferential strengths and weaknesses of the algorithm.

4.1 Two data generating processes

At a high level, we seek to learn the empirical time complexity and probabilistic accuracy of our model with simulated count data. First, however, we need to specify feasible simulation routines that emulate the referendum voting process. To simulate aggregate count data which adheres to our modeling assumptions, we propose two distinct simulation routines: (1) a strict mixture model, and (2) a hierarchical strict mixture model with town specific Dirichlet priors. The main difference between these approaches lies in how locales are allocated to latent ideologies.

Regardless of our chosen DGP, we note that the following experiments will most of the time span the parameter sets[2]

$$\begin{aligned} M &\in \{10, 50, 200, 500, 2000\}, \\ Q &\in \{1, 5, 20, 70\}, \\ N_{iq} &\in \{100, 1000, 5000, 30000\}, \\ \nu_{kq} &= 100, \\ \alpha &\in \{0.1, 0.3, 0.5, 0.8\}. \end{aligned}$$

In this context, α is the parameter for a symmetric Dirichlet distribution which acts as a prior over how "mixed" a certain locale is [7]. For example, by choosing $\alpha = 0.1$ with $K = 3$, one draw of a Dirichlet distribution with parameter $\vec{\alpha} = (0.1, 0.1, 0.1)$ would yield a probability vector of length 3 with most of the probability concentrated at a single index. On the contrary, the choice of $\alpha = 0.8$ with $K = 3$ would give us a draw with $\vec{\alpha} = (0.8, 0.8, 0.8)$, yielding another probability vector of length 3 but this time with more evenly distributed probability across indices. In plain english, changing this alpha parameter lets us change the level of mixture within data.

When relevant in specific simulations, we also introduce the K parameter across the discrete range $K = \{3, \ldots, 10\}$.

4.1.1 Strict mixture model

The strict mixture model is best aligned with what our data likelihood assumes in Section 2.3.2, but perhaps not as well-aligned with how individual voting behavior actually manifests.

[1] Computing resources for this chapter provided by courtesy of the Bowdoin HPC Environment.

[2] Each specified N_{iq} value is held constant $\forall\, i, q$.

Simply put, this routine will simulate a single vector of global mixture weights λ and then sample success counts across locales and questions. This is detailed in the below pseudocode.

Algorithm 2 Strict mixture model simulation

Require: number of towns M, voters per town N_{iq}, number of mixture components K, number of questions Q, symmetric Dirichlet parameter vector $\vec{\alpha}$, Beta-binomial dispersion parameter ν.

Outputs: simulated success-count matrix $Y_{M\times Q}$, global mixture weights w, μ_{kq} parameter matrix $P_{K\times Q}$, true locale labels z_i.

Draw global mixture weights $w \sim \text{Dirichlet}(\vec{\alpha})$
Draw ideology-question probabilities $\mu_{kq} \sim \text{Uniform}(0.2, 0.8)\ \forall\ k = 1, \ldots, K,\ q = 1, \ldots, Q$
Initialize success-count matrix Y
Initialize true assignment vector $z_i\ \forall\ i = 1, \ldots, M$
for $i = 1, \ldots, M$ **do**
    Draw town assignment $z_i \sim \text{Categorical}(w)$
    for $q = 1, \ldots, Q$ **do**
        Draw success counts

$$Y_{iq} \sim \text{BetaBinomial}\left(N_{iq}, P_{z_i,q}, \nu\right)$$

    end for
end for
return Y, w, P, z_i

As such, we describe the generative story as:

1. Draw a single global mixture weight over the K ideologies.
2. Draw a support pattern per latent ideology per question.
3. For each locale, draw a latent ideology label.
4. Combine support pattern per ideology with each question, per drawn locale labels.

4.1.2 Hierarchical strict mixture model

Since the E-M as described provides a soft clustering of ideology within each locale i, we think to emulate this in our process of generating count data. This DGP varies from the former, in that we now simulate a vector of mixture weights for each locale i, and then sample our counts from these locale-specific mixture weights. Each locale now has a distribution across ideologies but still chooses just one to vote in accordance with, per question. This process is described in pseudocode below.

Algorithm 3 Hierarchical strict mixture model simulation

Require: number of towns M, voters per town N_{iq}, number of mixture components K, number of questions Q, symmetric Dirichlet parameter vector $\vec{\alpha}$, Beta-binomial dispersion parameter ν.

Outputs: simulated success-count matrix $Y_{M\times Q}$, locale specific mixture weight matrix $W_{M\times K}$, μ_{kq} parameter matrix $P_{K\times Q}$, true locale labels z_i.

1: Initialize locale-specific mixture weight matrix W
2: Draw locale-specific mixture weights $w_i \sim \text{Dirichlet}(\vec{\alpha})\ \forall\ i = 1, \ldots, M$
3: Draw ideology-question probabilities $\mu_{kq} \sim \text{Uniform}(0.2, 0.8)\ \forall\ k = 1, \ldots, K,\ q = 1, \ldots, Q$
4: Initialize success-count matrix Y
5: Initialize true assignment vector $z_i\ \forall\ i = 1, \ldots, M$
6: **for** $i = 1, \ldots, M$ **do**
7: Draw town assignment $z_i \sim \text{Categorical}(w_i)$
8: **for** $q = 1, \ldots, Q$ **do**
9: Draw success count

$$Y_{iq} \sim \text{BetaBinomial}\left(N_{iq}, P_{z_i,q}, \nu\right)$$

10: **end for**
11: **end for**
12: **return** Y, W, P, z_i

Thus, our generative story has changed slightly to:

1. For locale i, draw mixture weights over the K ideologies.
2. Draw a support pattern per latent ideology per question.
3. For each locale, draw a latent ideology label.
4. Combine support pattern per ideology with each question, per drawn locale labels.

Remark 4.1. This process is very similar to the latent Dirichlet allocation described in [5], but varies in that LDA allows for multiple latent draws per locale i. Markedly, our approach has the same setup until line 7 of the above pseudocode.

4.2 Computational efficiency of E-M algorithm

As this work is largely motivated by the computational limitations detailed in [18], we now examine how rapid this so-called rapid algorithm truly is. We choose to hold the parameters $K = 7$ and $\nu = 100$ constant, and then iterate over the M, Q, N_{iq}, and α parameters specified in Section 4.1. This yields us 320 unique parameter combinations, each of which we run through our algorithm 100 times.

When utilizing the strict mixture model DGP (with one global set of mixture weights), we first note the following histogram of runtimes across our $32,000$ separate runs.

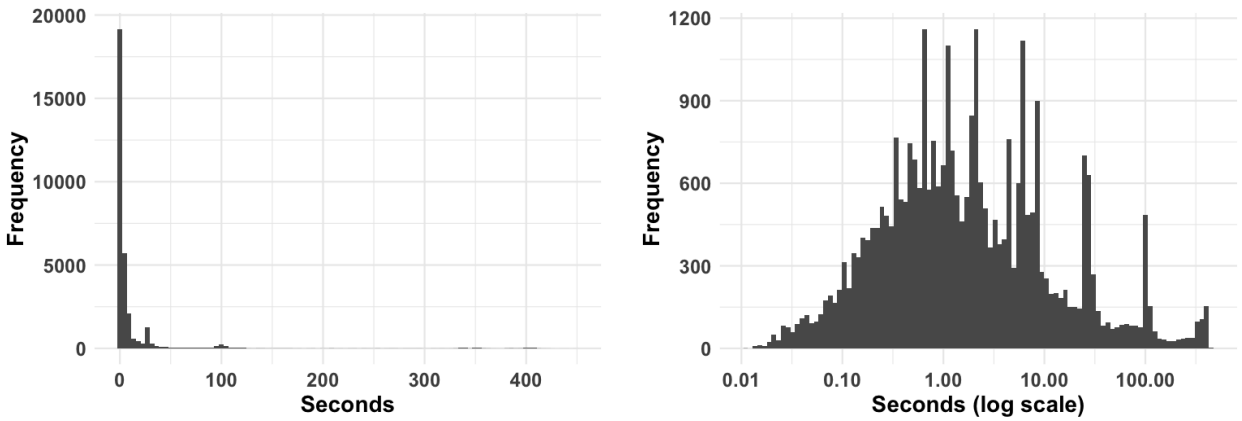


Figure 4.1: Variation in strict mixture model E-M runtimes, in seconds.

We notice when looking at our slowest parameter combinations that larger Q values, larger M values, and sparse mixtures (small symmetric Dirichlet parameter α) are generally the slowest to converge. Interestingly, this slowdown effect does not seem to be conditional upon N_{iq}, the number of voters per locale, as shown in the below heatmap.

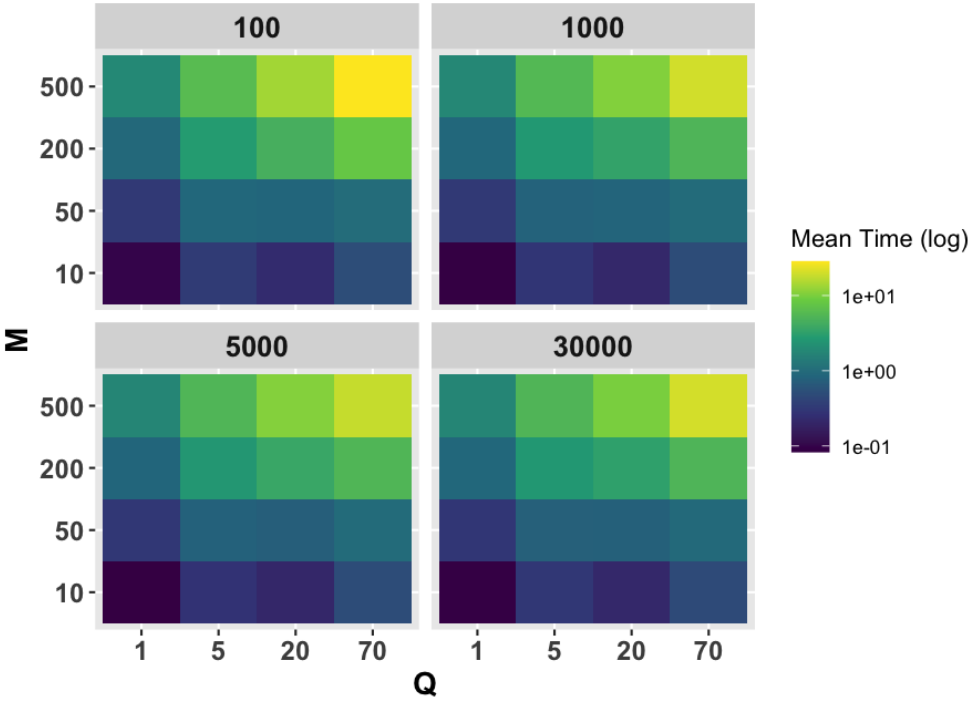


Figure 4.2: Heatmap of log(runtime) across M and Q, faceted by N_{iq}.

We draw extra attention to the fact that our model is not sensitive to N_{iq}, in the event that this quantity holds significant meaning in one's data. It is not surprising that the algorithm struggles to converge quickly when $Q > M$, and we note the interesting gradient across the bottom row of each pane.

We also note that regardless of M, larger Q correlates with requiring fewer E-M iterations to reach the stopping condition. This bodes well for our model—i.e. more information across questions helps our model refine its iterative process, rather than throw it off. A full summary informing this analysis is available in Table 1 in the Appendix, where we also include convergence rates across parameter sets. As expected, when Q increases, convergence rates (i.e. percent of runs which converge in ≤ 200 iterations) also increase within each grouping of M.

From these simulations we can also examine the empirical time complexity of our algorithm, by regressing M, Q, and N against log(time). After running this regression, we obtain that

$$\text{time} \approx C \cdot M^{0.81} \cdot Q^{0.32} \cdot N^{-0.04}.$$

These results line up closely with our above visualizations; namely, the cost of adding an additional locale is almost linear, the cost of increasing Q is relatively low, and the cost of N is negligible.

When using the hierarchical strict model DGP, we notice that our "worst case" parameter combinations follow the same self-consistent pattern, as well as take similar amounts of time to converge. We note that the empirical time complexity regression yields very simular results, and point the interested reader towards Table 2 in the Appendix for a full summary of this DGP's computational timings.

Remark 4.2. We also ran the above experiments when $K = 50$, which is quite large in the context of this data type. Interestingly, we notice similar results in terms of empirical time complexity across both DGPs.

4.3 Probabilistic accuracy of algorithm in simulation

We have shown empirically the computational speed of the algorithm, and now turn to understanding its inferential ability.

4.3.1 Search time and accuracy in recovering K

We recall the first of our three motivating questions: across our M locales, how many unique ideologies, K, exist? To answer this question, we must run our algorithm across a range of K values and then perform some flavor of model selection. As such, we choose to use the Aikake Information Criterion [1] (AIC) and Bayesian Information Criterion [22] (BIC) heuristics.

We recall the following definitions for AIC and BIC:

$$AIC = \underbrace{2\mathcal{K}}_{\text{complexity}} - \underbrace{2\ln(\mathcal{L})}_{\text{fit}},$$
$$BIC = \underbrace{\mathcal{K}\ln(M \cdot Q)}_{\text{complexity}} - \underbrace{2\ln(\mathcal{L})}_{\text{fit}},$$

where $\mathcal{K}$ is the number of parameters estimated by the model and $\mathcal{L}$ is the model's maximized likelihood.

When learning the ν parameter in our algorithm, the number of parameters

$$\mathcal{K} = \underbrace{(K-1)}_{\text{mixture weights}} + \underbrace{(K \cdot Q)}_{\mu \text{ parameters}} + \underbrace{(K \cdot Q)}_{\mu \text{ parameters}} .$$

Similarly, when we choose to specify a fixed ν,

$$\mathcal{K} = \underbrace{(K-1)}_{\text{mixture weights}} + \underbrace{(K \cdot Q)}_{\mu \text{ parameters}} + \underbrace{1}_{\text{fixed } \nu \text{ parameter}} .$$

Remark 4.3. While the origin of our $K \cdot Q$ values should be clear from our indexing in Table 2.1, perhaps the $K - 1$ term is not as clear. This number of parameters arises from the constraint that $\sum_{k=1}^{K} \lambda_k = 1$.

Remark 4.4. We note that the BIC generally over penalizes models, compared to AIC. For this reason, we will choose to utilize BIC for the remainder of the manuscript, unless otherwise specified.

To actually select our model, we choose to plot out our BIC values against our range of K. To find the "best" model, we select the K at which an "elbow" forms. In the example below, this looks to be the blue dot at $K = 7$, even though the $\min(BIC)$ is at $K = 9$.

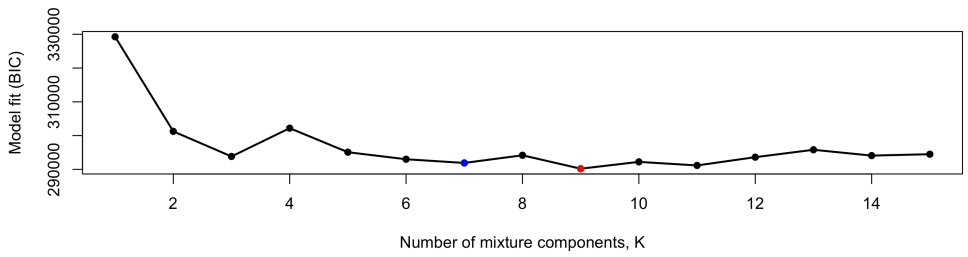


Figure 4.3: Example of an elbow plot, procured by the BIC heuristic.

Remark 4.5. As shown later in our Maine case study, sometimes these elbow plots look more like "W" plots. This is something to be aware of, especially in the context of our soft clustering algorithm (i.e. probabilities might be better aligned with our data with $K = 6$ and $K = 8$, but not with $K = 7$).

This method of model selection is noticeably imprecise, and as of right now relies on a human to look for and identify the elbow themselves. Since we need to run thousands of these simulations, we propose the following heuristics to "find the elbow" without supervision:

1. Take $\arg\min(BIC)$.
2. Choose the K such that $BIC(K) - BIC(K-1)$ is less than some ϵ.
3. Draw a line segment from the BIC value at the first K to the BIC value at the last K. Choose the K index with the maximum distance to an altitude dropped from this line segment.

With this setup established, we can now discuss experiment design. We use the same parameter combinations described in Section 4.1, and choose to additionally simulate over $K \in \{3, \ldots, 9\}$. We then let our BIC search range lie across $K = 1, \ldots, 15$, when testing how well our model recovers the true K.

Remark 4.6. We choose to run each parameter combination 50 times, but since we are scanning over a K range of length 15, the E-M procedure described in Algorithm 1 is fitted 1680000 times.

We first draw our attention to the strict mixture model DGP. When comparing to our previous simulation study results, the main difference is a wider range of K. As such, it makes sense that our slowest parameter combinations are roughly the same in this experiment. We note that our slowest search times involve larger M and Q values. The mean time for the slowest parameter combination $(M, N, Q, \alpha, K) = (2000, 100, 70, 0.5, 3)$ is 77.5 minutes, meaning even large datasets can be inferred reasonably quickly. Table 3 in the Appendix lists the slowest 20 parameter combinations.

Once again, very similar patterns manifest with the hierarchical strict mixture model DGP, but with slightly slower runtimes—i.e. the mean time for slowest combination is 82 minutes. While the 20 slowest cases are summarized in Appendix Table 4, we note that N has a slightly stronger negative effect on search time with this DGP.

In terms of recovering K accurately, we choose to take both the mean squared error (MSE) and mean absolute error (MAE) of each our three heuristics, across each parameter combination. We find the MSE and "epsilonic improvement" heuristic to be the most interpretable. In both of our DGPs, we notice that the model especially struggles to recover K accurately when the number of questions is small and the mixtures are sparse, even when M is large. While it is not likely that a researcher would know the "true K" prior to running the algorithm on their data, we note that the model has a harder time guessing larger K values. Also very important to note is that the algorithm tends to over-predict K in most cases, and will often have responsibility vectors with many zero indices.

4.3.2 Accuracy in recovering θ

We also recall our third motivating question: what parameters define the model's K ideologies? To poke at this question, we assume that an appropriate K has been specified, and thus seek out how Algorithm 1's outputs compare to our simulated DGP parameters. We re-emphasize the following notation, for ease of interpretability in this section.

Notation	Interpretation
$P_{K\times Q}$	Matrix of simulated truth μ_{kq} parameters
$\widehat{P}_{K\times Q}$	Matrix of E-M output μ_{kq} parameters
$\mathbf{w}$	Vector of simulated global mixture weights λ_k
$\widehat{\mathbf{w}}$	Vector of E-M output global mixture weights λ_k
$W_{M\times K}$	Matrix of locale-specific mixture weights (hierarchical DGP simulation)
$\widehat{\gamma}_{ik}$	Matrix of E-M output locale-specific responsibilities
z_i	Simulated locale-specific latent ideology assignments
$\widehat{z}_i$	E-M output locale-specific latent ideology assignments

Table 4.1: Helpful data notation for interpreting inference simulations.

As discussed in Section 3.2.5, the issue of label switching arises when comparing our simulated truth parameters to our E-M outputs. Since we know the "ground truth," however, we can get around this problem by finding the permutation π which minimizes the "cost" of moving from $P_{K\times Q}$ to $\widehat{P}_{K\times Q}$. Once we solve for this reordering of rows, we can apply the same ordering to our parameters $\widehat{\mathbf{w}}$ and $\widehat{z}_i$ to make meaningful inference.[3]

Taking a step back, let us define the cost matrix $C_{K\times K}$ by

$$C_{ab} = \sum_{q=1}^{Q} (\widehat{P}_{aq} - P_{bq})^2$$

for $a, b \in \{1, \ldots, K\}$. As such, each entry in this matrix C measures the distance between the fitted cluster a and the true cluster b. We then define the permutation function

$$\pi : \{1, \ldots, K\} \to \{1, \ldots, K\}$$

such that

$$\pi(a) = \text{index of ground truth cluster optimally matched to cluster } a.$$

As such, we seek out the permutation π which minimizes total cost across rows and columns of C. We can visualize this in the following small example.

Example 4.1. Optimal cost by eye

We pose this small example to provide context for the above matching problem. Suppose that in a strict mixture model simulation, we set $M = 5, Q = 2$, and $K = 3$ and obtain the ground truth results of

$$P_{3\times 2} = \begin{pmatrix} 0.8 & 0.2 \\ 0.3 & 0.7 \\ 0.6 & 0.6 \end{pmatrix},$$

$$\mathbf{w} = \begin{pmatrix} 0.5 \\ 0.3 \\ 0.2 \end{pmatrix},$$

$$z_i = (1, 1, 2, 3, 2) \quad \text{for} \quad i = 1, \ldots, 5.$$

[3] We use the matrix P to find the optimal permutation, since it is the largest in size and thus contains the most information.

We then obtain through running our E-M that

$$\widehat{P}_{3\times 2} = \begin{pmatrix} 0.6 & 0.6 \\ 0.8 & 0.2 \\ 0.3 & 0.7 \end{pmatrix},$$

$$\widehat{\mathbf{w}} = \begin{pmatrix} 0.2 \\ 0.5 \\ 0.3 \end{pmatrix},$$

$$\widehat{\gamma_{ik}} = \begin{pmatrix} 0.05 & 0.90 & 0.05 \\ 0.10 & 0.85 & 0.05 \\ 0.05 & 0.05 & 0.90 \\ 0.90 & 0.05 & 0.05 \\ 0.10 & 0.10 & 0.80 \end{pmatrix}.$$

Constructing the cost matrix C as above, we obtain

$$C_{3\times 3} = \begin{pmatrix} 0.2 & 0.1 & 0 \\ 0 & 0.5 & 0.2 \\ 0.5 & 0.0 & 0.1 \end{pmatrix}.$$

Looking across our rows, the lowest cost of cluster 1 maps to cluster 3. Similarly, cluster 2 maps to cluster 1 and cluster 3 maps to cluster 2. So the optimal permutation $\pi(1,2,3) = (3,1,2)$. We then simply invoke the same ordering on our $\widehat{\mathbf{w}}$ and $\arg\max\, \widehat{\gamma}_{ik} = \widehat{z}_i$ for inference. We note that in this example, all parameters were inferred perfectly. This is not likely to happen with real data, meaning the minimum value of each row of C will not actually be 0. In order to minimize C at scale, we use a linear sum assignment problem solver.[4]

With the issue of label switching out of the way, we detail the following measures of inference accuracy in our analysis.

Notation	Interpretation
$\|\|\Delta\mathbf{w}\|\|_1 = \sum_k \|\mathbf{w}_k - \widehat{\mathbf{w}}_k\|$	$L1$ recovery of global mixture weights
$\|\|\Delta\mathbf{w}\|\|_2 = \sqrt{\sum_k (\mathbf{w}_k - \widehat{\mathbf{w}}_k)^2}$	$L2$ recovery of global mixture weights
$\|\|\Delta P\|\|_F = \sqrt{\sum_k \sum_q (P_{kq} - \widehat{P}_{kq})^2}$	Frobenius norm of matrix P
$\text{MAE}(\Delta P) = \frac{1}{KQ} \sum_k \sum_q \|(P_{kq} - \widehat{P}_{kq})\|$	Mean absolute error of matrix P entries
$\frac{1}{M} \sum_i \widehat{\gamma}_{i,z_i}$	Average recovery of soft clustering

Table 4.2: Measures of accuracy used in inference simulations.

Remark 4.7. We note that the LSAP solver will find the optimal matching, but not necessarily the absolute correct matching. Because this is true, we caution towards the interpretability of the above metrics of inferential quality.

[4]Specifically, `solve_LSAP` from the `clue` `R` package.

We notice that depending on the chosen simulation routine (i.e. strict mixture model versus hierarchical strict mixture model), we have slightly different inference parameters to compare. Regardless, we once again iterate over the parameter sets described in Section 4.1, this time with the true K specified over the range $4, \ldots, 10$. Let us start with the strict mixture model simulation routine.

We recall that the matrix P is made up of our Beta-binomial mean parameters μ_{kq}. When looking across all parameters sets, we see that our algorithm is on average 8% wrong when guessing these parameters.

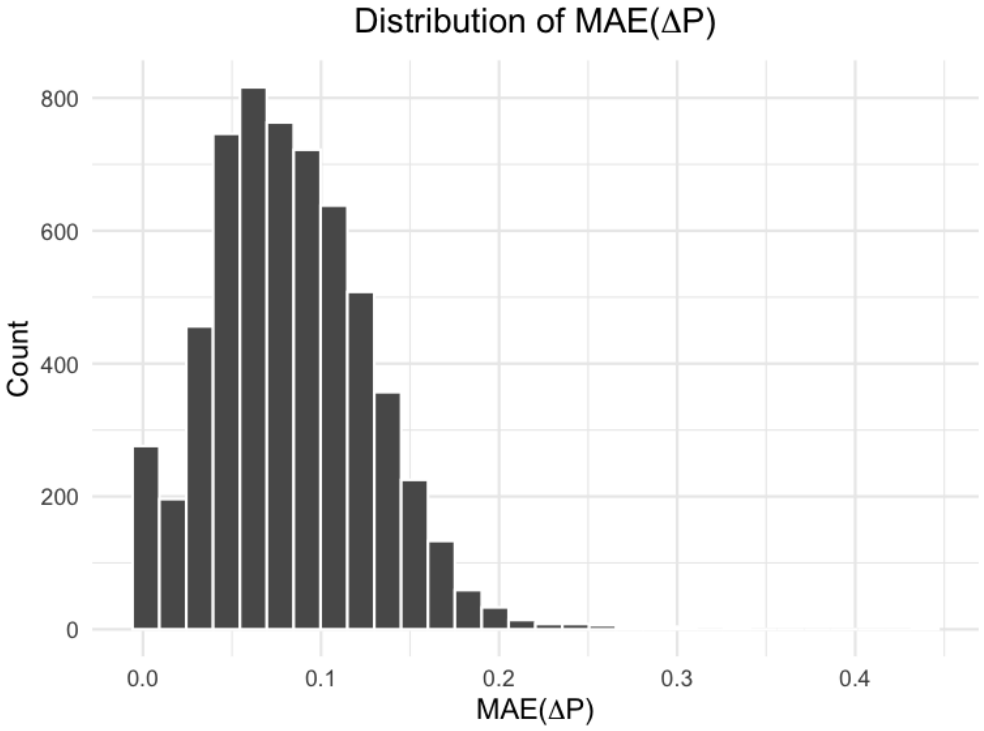


Figure 4.4: Histogram of Beta-binomial mean parameter inference accuracy across all parameter sets.

Interestingly, the number of questions Q does not have a strong effect on this recovery metric, while the number of locales M has a small positive effect. As expected, the number of voters per locale N has almost no effect.

We understand the final entry in our notation Table 4.1 to be the proportion of probability mass assigned to the correct cluster by our algorithm. We thus set a cutoff of 80% correctness to denote a successful E-M run. We notice that evenly mixed locales (symmetric Dirichlet parameter α closer to 1) across the board lead to better inference, and that model performance is not dependent on K in the specified range when mixtures are relatively even across ideologies. Given these above specifications, inferential accuracy improves with larger M and Q. We can also compute the percent accuracy of the optimally matched $\widehat{z}_i$ against z_i, and obtain the following heatmap across M and Q.

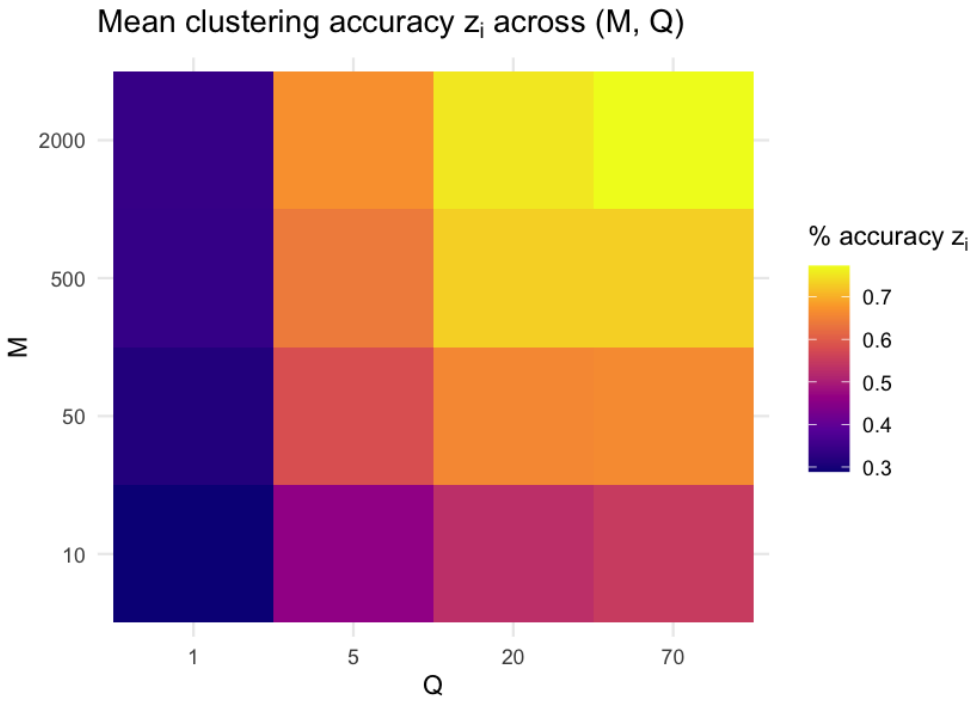


Figure 4.5: Heatmap of latent assignment accuracy across M and Q..

We note that the model improves as Q grows, across each M, albeit at different rates. As is implied by the above, our examinations of $||\Delta \mathbf{w}||_1$ and $||\Delta \mathbf{w}||_2$ reveal a similar story across parameter sets.

Moving onto the second DGP, the hierarchical strict model, we note very similar MAE(ΔP) distributional results, but with a slightly more accurate mean difference of 4.8%. As per the latent Dirichlet allocation-esque simulation routine, we can also compare $W_{M \times K}$ with $\widehat{\gamma}_{ik}$. Unfortunately, our diagnostics reveal that the model does quite poorly in recovering soft assignments $W_{M \times K}$. We can likely attribute this to the hard-clustering aspect of our model specification, as well as poor optimal matchings across permutations π.

In sum, the model does relatively well with recovering model parameters independently of dataset size; on the contrary, the correctness of the recovered mixtures relies more heavily on having enough data for the E-M to work with, as well as the undelying mixture proportions in the data itself.

Chapter 5

Results

In this chapter we apply the algorithm to two real examples of referendum count data, in order to show its exploratory use for social scientists. We especially recall the gradient of statistical approaches detailed in Figure 1.1 when thinking towards these results.

5.1 Case study: Maine referendum data

We recall our initial assumption that political behavior at the referendum level is explained by a finite number of culturally determined political cultures which manifest as latent, consistent sets of principles and beliefs. A large motivation for this work has been to expand upon the Bayesian model described in [18] for computational reasons; as such, we perform inference on the same longitudinal Maine Referendum Voting data from $2008 - 2024$,[1] which contains vote totals from 423 municipalities across 70 questions.[2] We now think back towards our three motivating questions from the manuscript, this time in the context of Maine:

1. How many different cultures of Mainers are there?
2. What cultures of Mainers make up each municipality?
3. How does each culture of Mainer vote in referendum?

We therefore run our model on this longitudinal dataset [2], which has parameter values $M = 423$, $Q = 70$, and $N_{iq} \in [3, 37829]$. After searching over the range $K = 2, \ldots, 12$, we obtain the following BIC model selection heuristic

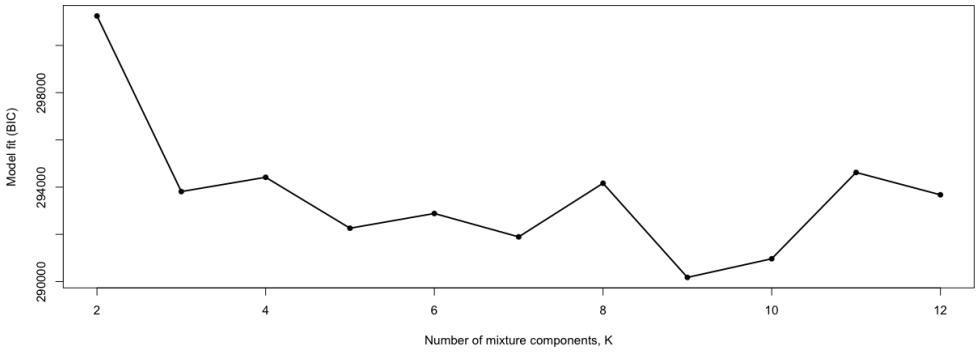


Figure 5.1: Beta-binomial mixture model elbow plot for Maine case study.

[1] The study performed in [18] looked over the years $2008 - 2019$.

[2] For the reader's enjoyment, these 70 questions are detailed in the Appendix.

We note that an elbow seems to appear at $K = 7$, which is almost exactly the posterior mode from the analysis in [18]. So according to our model, there exist 7 distinct cultures of Mainers.

With this K selected, we run our algorithm on the full data and report convergence in 5.31 minutes when learning ν_{kq} and 16.12 seconds when setting $\nu_{kq} = 100 \; \forall \, k, q$.

A natural first step for political scientists may be to examine how accurate the proposed mixture distributions are per town, by question. In the below figure, for Brunswick, Maine, we randomly select a series of questions and plot the simulated mixture distributions in black and the observed success counts in red.

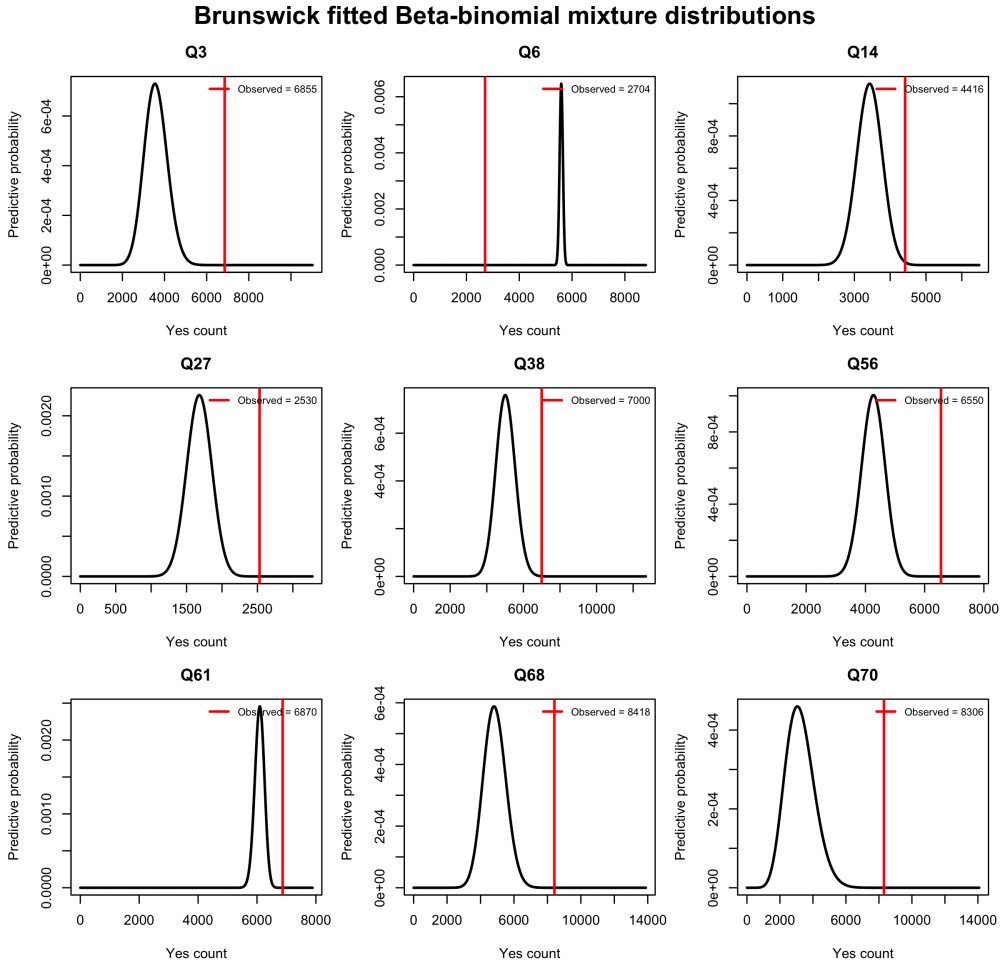


Figure 5.2: Beta-binomial mixture model's proposed mixture distributions for a selection of questions in Brunswick, Maine.

A natural reaction to the above figure is that the so-called mixture distributions do not appear to be very dispersed. As discussed in our simulation study in the context of sparse responsibility vectors, this is a feature of the algorithm itself. Namely, the point-estimate nature of our approach allows the E-M to converge upon a local maxima and not explore the full cluster structure of the mixture. This in turn leads to very "tight" predicted distributions

to describe our system. Interestingly, when the algorithm is not freed to learn ν on its own, this issue is not as prevalent. This sparse mixture problem is one such tradeoff to having a rapid algorithm, which we discuss further in the conclusion.

Another relevant visualization for political scientists may be to see the algorithm's soft clusterings on a map. Drawing directly from [18], we compare the responsibilities per municipality γ_{ik} as inferred through the Bayesian frame (left map) to those inferred by the E-M algorithm (right map).

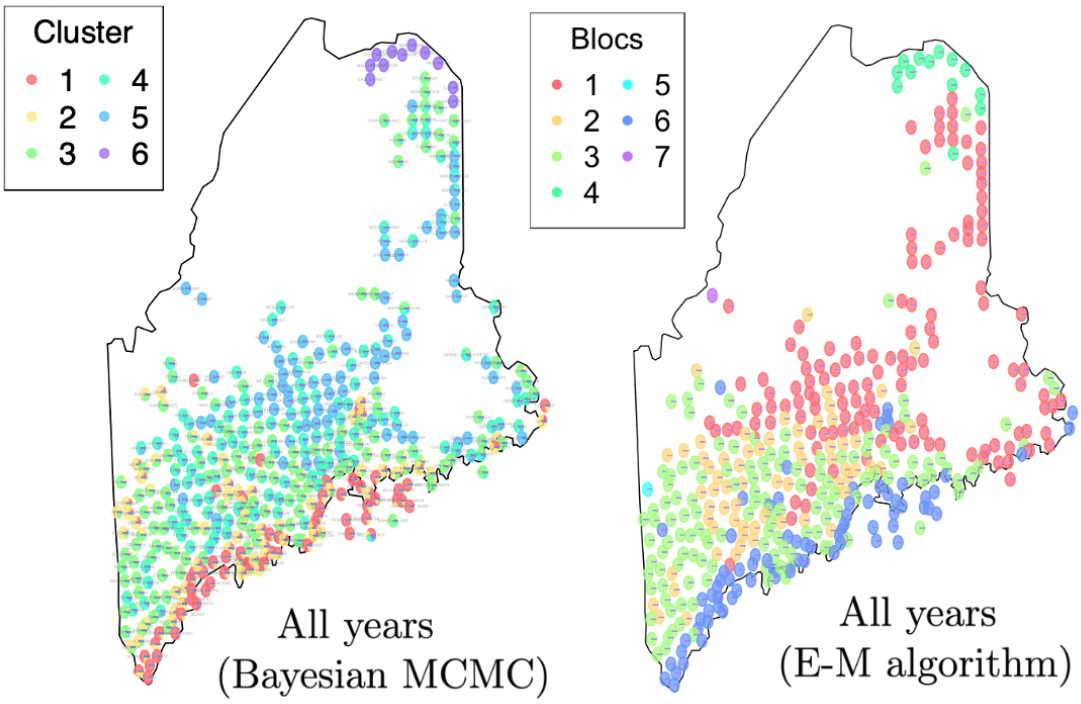


Figure 5.3: Responsibilities per Maine municipality, across longitudinal dataset. Left: Bayesian method (6 weeks of computation). Right: E-M algorithm (16.12 seconds of computation).

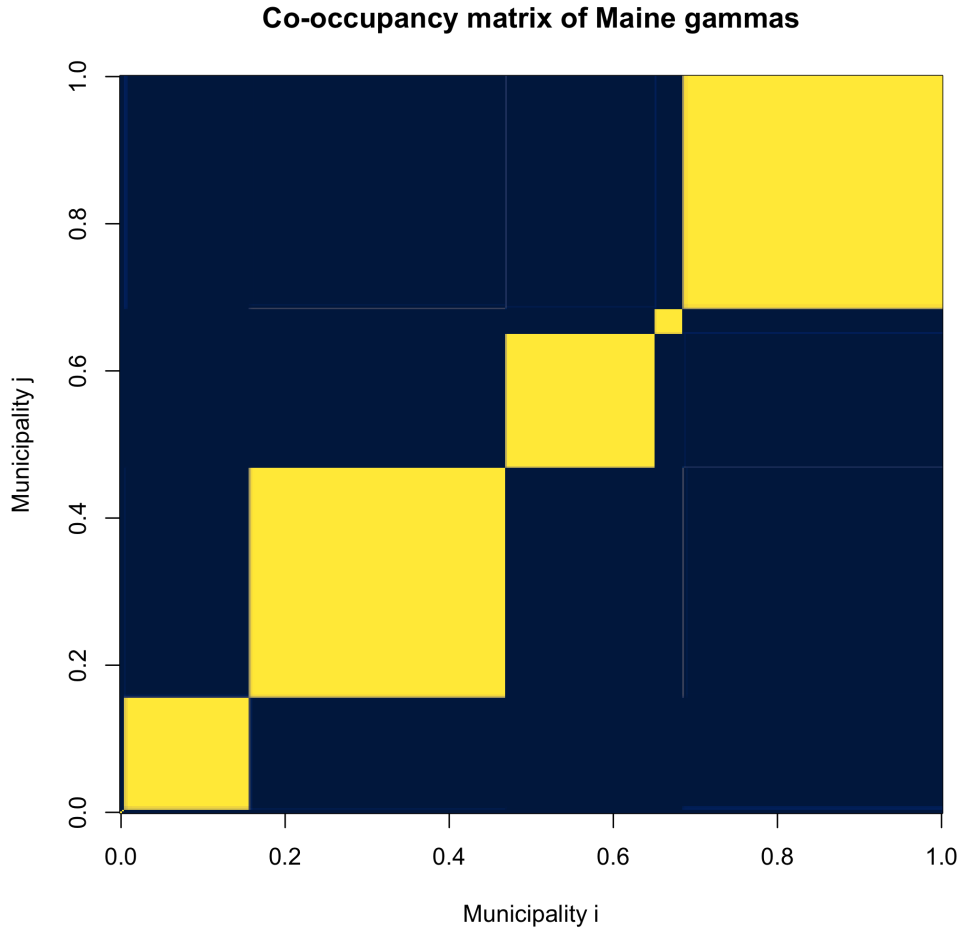
Apart from the different colored labels between these two maps—an artifact of the label switching problem—we notice consistent voting ideologies (or blocs) between the two inference schemes. Critically, the left-hand map took ~ 6 weeks of computation to procure and the right-hand map took 16.12 seconds.

Looking specifically at the E-M algorithm output (right-hand map), we notice the following distinct voting ideologies (or blocs): blocs 1 and 2 appear to encompass rural inland communities, bloc 3 seems to represent inland westerners as well as some of the eastern coast, bloc 4 describes French-speaking communities on the Canadian border, and bloc 6 covers the southern coast and larger inland cities. Thinking back to our initial assumptions, these maps prompt questions about how ideologies are generated. Especially because this strong spatial structure persists across multiple runs of the E-M, it is not unreasonable to postulate that ideologies in this context are heavily spatially informed. We also note that these clusters bear resemblance to settlement patterns of Maine.

Remark 5.1. We notice our problem of sparse responsibilities explicitly in the right-hand map, since most of our pie charts appear to be entirely one color. This issue is less prevalent in the Bayesian frame.

Remark 5.2. Just as discussed in our simulation study, we see the algorithm slightly over-predict K—i.e. blocs 5 and 7 likely need not exist to describe their own unique ideologies.

A final step we can take to visualize our cluster structure is constructing a co-occupancy matrix. Given our soft clusterings γ_{ik}, we simply measure how similar each soft clustering is to one other, order these pairings by similarity, and then plot as a symmetric matrix.



This visualization confirms that our model is very confident in its clustering, since pairings inside the same block have very high co-occupancy. We also note that blocs 5 and 7 seem to have been subsumed into other blocs, as expected from our map.

With these above experiments laid out, we have both illuminated answers to our three motivating questions and shown that this rapid approximation of [18] would be fruitful to a curious political scientist.

5.2 Case study: Swiss referendum data

As mentioned in the introduction, Switzerland has a rich referendum process as a result of the Swiss direct democracy system. Due to the sheer size of this longitudinal data and resulting computational cost, however, this specific case study is previously unexplored in the Bayesian frame.

Foremost, we note that Swiss government processes operate at a range of spatial scales, each with different levels of sovereignty. We will focus in part on the 26 sovereign states called Cantons, but mostly on the communes/municipalities which are more local subdivisions of these Cantons. The `R` package `swissdd` conveniently accesses referendum count data at the municipality level from the Swiss Federal Statistical Office [23].

Our data spans the years $1981 - 2026$ and contains 390 questions across 2048 complete locales. We are at liberty to examine three very similar motivating questions:

1. How many different latent ideologies inform Swiss referendum voting?
2. How is Switzerland mixed across these ideologies?
3. What parameters define these ideologies?

Through the same BIC model selection procedure, we find that $K = 9$ best balances between model fit and complexity. With a fixed $\nu_{kq} = 100$, our algorithm converges in 7.25 minutes.

In light of this algorithm's proposed use as an exploratory tool, we will focus on visualizing its hard clusterings $z_i = \arg\max \gamma_{ik}$, across a map of Switzerland.

Figure 5.4: Swiss municipality hard cluster assignments.

As with the Maine case study, we note the strong spatial structure present in this map. Upon closer inspection, we also note a strong resemblance between our E-M outputs and the following two maps describing Switzerland in terms of language and religion.

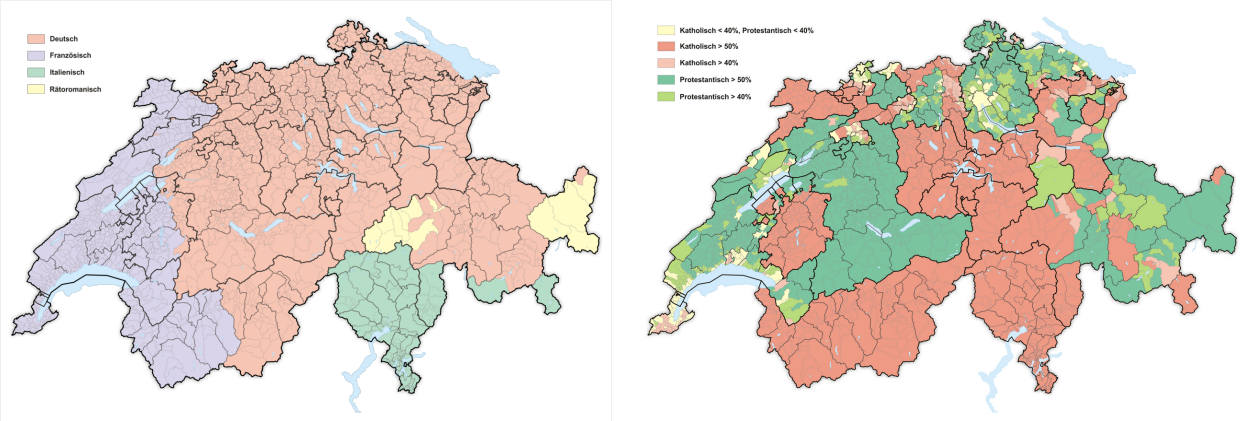


Figure 5.5: Plots detailing how Swiss municipalities are partitioned by most commonly spoken language and most practiced religion, as of January 2026 [14] [20].

Our algorithm almost exactly extracts the linguistic and religious partitions of Switzerland, given referendum count data! We note the distinct clusters of majority Catholic, majority Protestant, and mixed, all across a gradient of linguistic families spoken (German, French, Italian, and Romansh). Using Census data from 2000 accessed through the `BFS R` package [16], we can combine these two above maps into the following.

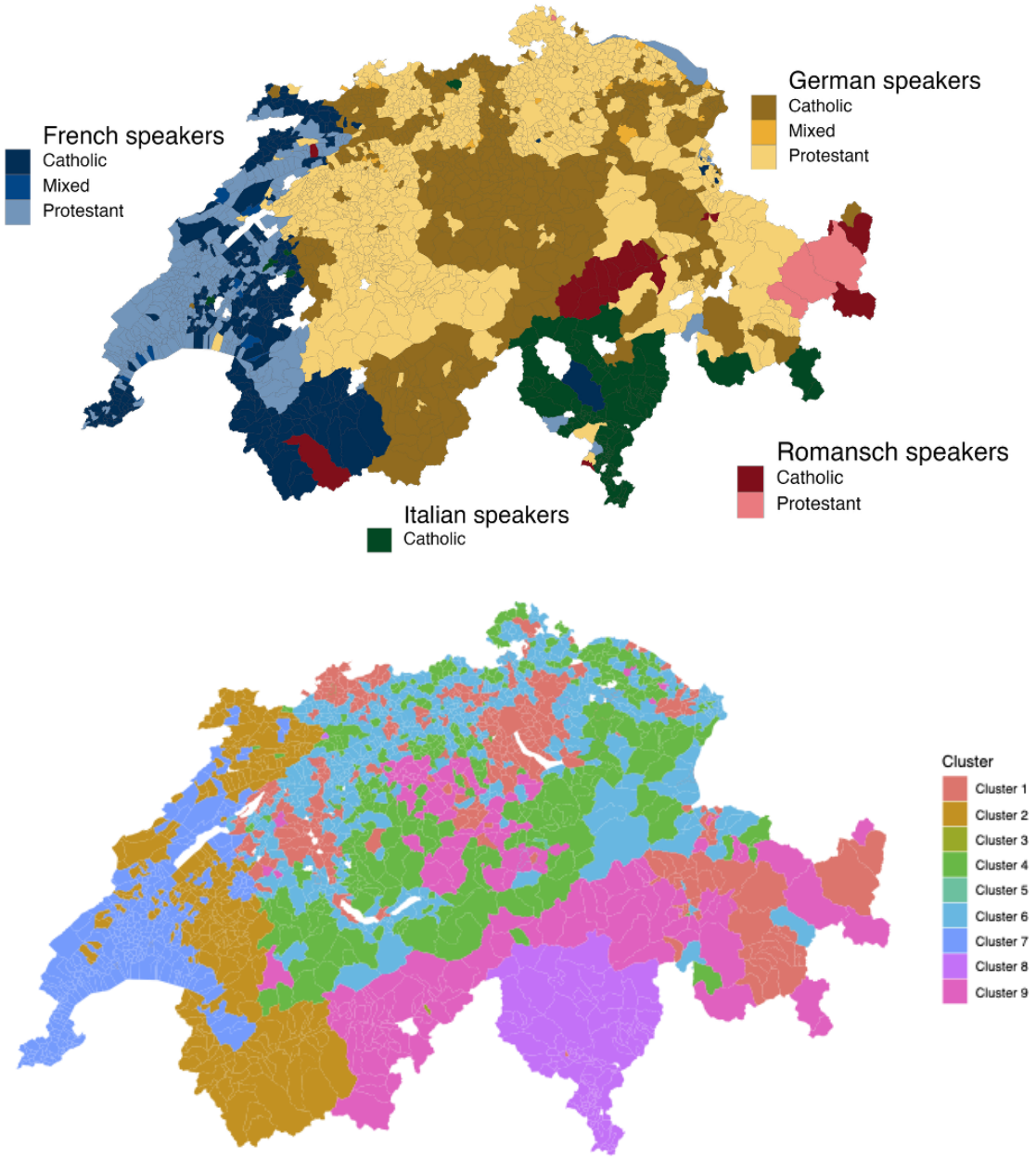


Figure 5.6: Swiss municipality hard cluster assignments $\widehat{z}_i$, stacked underneath a religious-linguistic partitioning of Swiss municipalities from 2000 Census data.

Remark 5.3. Concerning the top map, a threshold of 51% was utilized to denote "mixed" religion. Since the `.gpkg` file controlling spatial boundaries is from 2026, there are a few blank municipalities due to mismatches in spatial boundaries from the 2000 Census data.

Thinking back to our initial assumptions about ideology, there seems to be strong evidence that the basis for ideology in Switzerland is both linguistic and cultural. We also observe consistency across 10-year windows (namely $1981-1991, 1992-2002, 2003-2013$, and $2014-2026$) in these hard clustering maps, which rest in the Appendix. This result both

speaks to the rapid nature of our algorithm, as well as the strength of our religion/language conjecture.

Remark 5.4. In these Appendix plots, we note once again the label switching problem, in how our outputs are colored.

Chapter 6

Discussion

6.1 Tradeoffs to computational speed

As we noticed arise subtly in our simulation studies, as well as more aggressively in our Maine case study, this model has legitimate limitations in how it predicts dispersion of mixtures across a system. While in theory our E-M assignments γ_{ik} are probabilistic assignments of latent membership, in practice they are almost always "pseudo-hard" assignments as a result of strongly concentrated responsibility vectors. Thinking back to how our E-M algorithm worked iteratively in the one and two dimensional visualizations, this issue can be imagined as the E-M thinking that its probability mass allocations are "good enough," thus triggering a stopping condition. Put another way, our algorithm tends to converge upon local maxima before it has a chance to explore the whole mixture structure of the system.

This is to be expected from a rapid algorithm. As evidenced both in simulation and with real data, this is the tradeoff for offering quick, informative approximations of vastly complex systems (as discussed in Figure 1.1). To make our approach pseudo-Bayesian and solve part of this problem, we could utilize a tool like Laplacian approximation, as is done in [12], to better explore parameter space around our MLE point estimate. This is likely to be a fruitful future direction.

Another cost worth mentioning of our rapid algorithm is the lack of uncertainty quantification within our predicted results. This is especially noticeable when our model outputs the aforementioned strongly concentrated responsibility vectors. Given the above, a strong recommendation to those using this algorithm is to leverage it as a first pass on high dimensional count data, and then to progress to more certainty-rich, albeit expensive computationally, approaches such as the one in [18].

6.2 Future direction: invariance to spatial boundaries

We note in the Switzerland Case Study that data is available at the level of Canton and Municipality. Presumably, we can run our algorithm on the same underlying count data but aggregated at these two different scales. Would we uncover two distinct latent ideology structures? If so, what would this indicate about the effect of historically imposed spatial boundaries on ideology? Taking a step back, such a procedure may probe at the larger question: at what scales do voting ideologies manifest, and how well can our mixture model pick out distinct clusters across partitions of space? While this is currently a philosophical inquiry, a future direction would be to explore such pontifications empirically.

6.3 Future direction: extension to other count data in the social sciences

It should be apparent from the above that the ability to quickly look at a dataset and recover latent populations is quite valuable. While this is done strictly with voting data in our manuscript, our modeling approach may be applied to any type of aggregated count data with a rich replicate structure. An interesting future direction for example would be to explore this model's inferential ability on archaeological data as done in a Bayesian frame in [19].

Other than this manuscript, the deliverable of this thesis is an open-source suite of functions, for use by social scientists who hope to explore replicate count data. Such functions are detailed in this GitHub repository.

Appendices

A M-Step parameter derivations

In order to invoke our M-Step parameter update rule as in 3.2, we must derive the quantities $\widehat{\mu}_{kq}$ and $\widehat{\nu}_{kq}$ from the standard Beta-binomial distribution. In this appendix, we carry forth with a change-of-variables procedure between the BB's α, β parameterization and μ, ν parameterization, before computing first and second derivatives of the auxillary function $\mathcal{Q}$ for use in a Newton's step procedure.

A.1 Preliminaries

We first denote the standard α, β parameterization of the Beta-binomial distribution:

$$\begin{aligned} f_{\text{BB}}(x|n,\alpha,\beta) &= \int_0^1 \text{Binomial}(x|n,p) \cdot \text{Beta}(p|\alpha,\beta)\,\mathrm{d}p \\ &= \binom{n}{x} \frac{1}{B(\alpha,\beta)} \cdot \int_0^1 p^{x+\alpha-1}(1-p)^{n-x+\beta-1}\,\mathrm{d}p \\ &= \binom{n}{x} \frac{B(x+\alpha, n-x+\beta)}{B(\alpha,\beta)} \end{aligned}$$

where x is the number of successes in n trials, with success probability p. Since our analytical optimization problem is not changed by the $\log(\cdot)$ function, we choose to examine a version of following:

$$\log f_{\text{BB}}(x|n,\alpha,\beta) = \log \binom{n}{x} + \log B(x+\alpha, n-x+\beta) - \log B(\alpha,\beta).$$

To have finer grain control over the mean and dispersion of our modeled counts, we choose to re-parameterize the above in terms of μ and ν.

$$\begin{aligned} \mu &= \frac{\alpha}{\alpha+\beta} \\ \nu &= \alpha+\beta \end{aligned} \quad \Longrightarrow \quad \begin{aligned} \alpha &= \mu \cdot \nu \\ \beta &= (1-\mu)\cdot\nu \end{aligned}$$

Now, we make the appropriate substitutions in order to optimize the auxiliary function $\mathcal{Q}$ with respect to both μ and ν. We reintroduce indexing over our parameters x, n, α, β in accordance with Table 2.1. Note that we choose to exclude mixture weights from this construction, as they are not relevant to these optimization steps—to maintain notation, we subscript the auxiliary function across k.[1] Then

$$\mathcal{Q}_k = \sum_{i=1}^{M} \gamma_{ik} [\log \binom{n_i}{x_i} + \log B(x_i+\alpha_k, n_i - x_i + \beta_k) - \log B(\alpha_k, \beta_k)],$$

[1] We choose to detail our optimization procedure over $Q = 1$ questions, in order to clean up indexing.

where we make the substitutions below for sake of notation:

$$
\begin{aligned}
a_{1i} &= x_i + \alpha_k = x_i + \mu_k \nu_k \\
b_{1i} &= n_i - x_i + \beta_k = n_i - x_i + (1 - \mu_k)\nu_k \\
a_0 &= \alpha_k = \mu_k \nu_k \\
b_0 &= \beta_k = (1 - \mu_k)\nu_k
\end{aligned}
$$

We note the following relationships between the beta function $B(\cdot, \cdot)$, gamma function $\Gamma(\cdot)$, digamma function $\psi(\cdot)$, and trigamma function $\psi_1(\cdot)$ for use in our derivation.

$$
\begin{aligned}
B(x, y) &= \frac{\Gamma(x) \cdot \Gamma(y)}{\Gamma(x + y)}, && (1) \\
\log B(x, y) &= \log \Gamma(x) + \log \Gamma(y) - \log \Gamma(x + y), && (2) \\
\frac{d}{dz}[\log \Gamma(z)] &= \psi(z) \cdot dz, && (3) \\
\frac{d}{dz}[\psi(z)] &= \psi_1(z) \cdot dz. && (4)
\end{aligned}
$$

We thus obtain the following form of $\mathcal{Q}_k$, to take first and second derivatives of during each E-M iteration.

$$
\mathcal{Q}_k = \sum_{i=1}^{M} \gamma_{ik} \left(\log \binom{n_i}{x_i} + \log B(a_{1i}, b_{1i}) - \log B(a_0, b_0) \right)
$$

A.2 Deriving $\widehat{\mu}_k$ parameter update

Taking the above, we obtain that

$$
\frac{\partial}{\partial \mu_k}[\mathcal{Q}_k] = \sum_{i=1}^{M} \gamma_{ik} \left(\underbrace{\frac{\partial}{\partial \mu_k}[\log \binom{n_i}{x_i}]}_{\text{A}} + \underbrace{\frac{\partial}{\partial \mu_k}[\log B(a_{1i}, b_{1i})]}_{\text{B}} - \underbrace{\frac{\partial}{\partial \mu_k}[\log B(a_0, b_0)]}_{\text{C}} \right).
$$

Then

$$
\begin{aligned}
A &= 0, \\
B &= \frac{\partial}{\partial \mu_k}[\log \Gamma(a_{1i}) + \log \Gamma(b_{1i}) - \log \Gamma(a_{1i} + b_{1i})] \\
&= \nu_k \left(\psi(a_{1i}) - \psi(b_{1i}) \right), \\
C &= \frac{\partial}{\partial \mu_k}[\log \Gamma(a_0) + \log \Gamma(b_0) - \log \Gamma(a_0 + b_0)] \\
&= \nu_k \left(\psi(a_0) - \psi(b_0) \right),
\end{aligned}
$$

meaning our expression simplifies cleanly to

$$\sum_{i=1}^{M} \gamma_{ik}(\nu_k \left(\psi(x_i + \mu_k \cdot \nu_k) - \psi(n_i - x_i + \nu_k - \nu_k \cdot \mu_k)\right)) - (\nu_k \left(\psi(\mu_k \cdot \nu_k) - \psi(\nu_k - \nu_k \cdot \mu_k)\right)),$$

giving us the boxed expression

$$\boxed{\begin{aligned}\frac{\partial}{\partial \mu_k}[\mathcal{Q}_k] = \nu_k \cdot \sum_{i=1}^{M} \gamma_{ik}\Big(&\psi(x_i + \mu_k \cdot \nu_k) - \psi(n_i - x_i + (1 - \mu_k) \cdot \nu_k) \\ &- \psi(\nu_k \cdot \mu_k) + \psi((1 - \mu_k)) \cdot \nu_k\Big)\end{aligned}}.$$

To obtain $\frac{\partial^2}{\partial \mu_k^2}[\mathcal{Q}_k]$, we simply take

$$\frac{\partial}{\partial \mu_k}[\nu_k \cdot \sum_{i=1}^{M} \gamma_{ik}(\psi(x_i + \mu_k \cdot \nu_k) - \psi(n_i - x_i + \nu_k - \nu_k \cdot \mu_k) - \psi(\nu_k \cdot \mu_k) + \psi(\nu_k \cdot (1 - \mu_k)))],$$

which simplifies to

$$\begin{aligned}&\sum_{i=1}^{M} \gamma_{ik}\nu_k \left(\frac{\partial}{\partial \mu_k}[\psi(a_{1i})] - \frac{\partial}{\partial \mu_k}[\psi(b_{1i})] - \frac{\partial}{\partial \mu_k}[\psi(a_0)] + \frac{\partial}{\partial \mu_k}[\psi(b_0)]\right) \\ &\qquad = \sum_{i=1}^{M} \gamma_{ik}\nu_k \left(\nu_k\psi_1(a_{1i}) + \nu_k\psi_1(b_{1i}) - \nu_k\psi_1(a_0) - \nu_k\psi_1(b_0)\right),\end{aligned}$$

yielding the boxed expression

$$\boxed{\begin{aligned}\frac{\partial^2}{\partial \mu_k^2}[\mathcal{Q}_k] = \nu_k^2 \cdot \sum_{i=1}^{M} \gamma_{ik}\Big(&\psi_1(x_i + \mu_k \cdot \nu_k) + \psi_1(n_i - x_i + (1 - \mu_k) \cdot \nu_k) \\ &- \psi_1(\nu_k \cdot \mu_k) - \psi_1((1 - \mu_k) \cdot \nu_k)\Big)\end{aligned}}.$$

A.3 Deriving $\widehat{\nu}_k$ parameter update

This derivation is slightly more involved with chain rule simplifications, but we follow roughly the same outline. We thus seek out

$$\frac{\partial}{\partial \nu_k}[\mathcal{Q}_k] = \sum_{i=1}^{M} \gamma_{ik} \left(\underbrace{\frac{\partial}{\partial \nu_k}[\log \binom{n_i}{x_i}]}_{\text{A}} + \underbrace{\frac{\partial}{\partial \nu_k}[\log B(a_{1i}, b_{1i})]}_{\text{B}} - \underbrace{\frac{\partial}{\partial \nu_k}[\log B(a_0, b_0)]}_{\text{C}}\right).$$

Then

$$
\begin{aligned}
A &= 0, \\
B &= \frac{\partial}{\partial \nu_k}[\log\Gamma(a_{1i}) + \log\Gamma(b_{1i}) - \log\Gamma(a_{1i}+b_{1i})] \\
&= \mu_k \cdot \psi(a_{1i}) + (1-\mu_k)\cdot\psi(b_{1i}) - (\mu_k + 1 - \mu_k)\cdot\psi(a_{1i}+b_{1i}) \\
&= \mu_k \cdot \psi(a_{1i}) + \psi(b_{1i}) - \mu_k\cdot\psi(b_{1i}) - \psi(a_{1i}+b_{1i}), \\
C &= \frac{\partial}{\partial \nu_k}[\log\Gamma(a_0) + \log\Gamma(b_0) - \log\Gamma(a_0+b_0)] \\
&= \mu_k \cdot \psi(a_0) + (1-\mu_k)\cdot\psi(b_0) - \psi(\nu_k) \\
&= \mu_k \cdot \psi(a_0) + \psi(b_0) - \mu_k\cdot\psi(b_0) - \psi(\nu_k),
\end{aligned}
$$

meaning our expression simplifies to

$$
\begin{aligned}
&\sum_{i=1}^{M}\gamma_{ik}\Big(\mu_k\cdot\psi(a_{1i}) + \psi(b_{1i}) - \mu_k\cdot\psi(b_{1i}) \\
&\quad - \psi(a_{1i}+b_{1i}) - \mu_k\cdot\psi(a_0) - \psi(b_0) + \mu_k\cdot\psi(b_0) + \psi(\nu_k)\Big) \\
=&\sum_{i=1}^{M}\gamma_{ik}\Big(\mu_k\cdot\psi(a_{1i}) - \mu_k\cdot\psi(a_0) \\
&\quad - \psi(a_{1i}+b_{1i}) + (\mu_k-1)\cdot\psi(b_0) - (\mu_k-1)\cdot\psi(b_{1i}) + \psi(\nu_k)\Big),
\end{aligned}
$$

giving us the boxed expression

$$
\boxed{
\begin{aligned}
\frac{\partial}{\partial\nu_k}[\mathcal{Q}_k] = \sum_{i=1}^{M}\gamma_{ik}\Big(&\mu_k\cdot\psi(x_i+\mu_k\cdot\nu_k) - \mu_k\cdot\psi(\mu_k\cdot\nu_k) - \psi(n_i+\nu_k) \\
&+ (\mu_k-1)\cdot\psi((1-\mu_k)\cdot\nu_k) \\
&- (\mu_k-1)\cdot\psi(n_i - x_i + (1-\mu_k)\cdot\nu_k) + \psi(\nu_k)\Big)
\end{aligned}
}.
$$

As before, to obtain $\frac{\partial^2}{\partial\nu_k^2}[\mathcal{Q}_k]$, we simply take

$$
\begin{aligned}
&\frac{\partial}{\partial\nu_k}[\sum_{i=1}^{M}\gamma_{ik}(\mu_k\cdot\psi(x_i+\mu_k\cdot\nu_k) - \mu_k\cdot\psi(\mu_k\cdot\nu_k) - \psi(n_i+\nu_k) \\
&+(\mu_k-1)\cdot\psi((1-\mu_k)\cdot\nu_k) - (\mu_k-1)\cdot\psi(n_i - x_i + (1-\mu_k)\cdot\nu_k) + \psi(\nu_k))],
\end{aligned}
$$

which simplifies to

$$\sum_{i=1}^{M} \gamma_{ik} \Bigg(\frac{\partial}{\partial \nu_k}[\mu_k \cdot \psi(a_{1i})] - \frac{\partial}{\partial \nu_k}[\mu_k \cdot \psi(a_0)] - \frac{\partial}{\partial \nu_k}[\psi(a_{1i} + b_{1i})] + \frac{\partial}{\partial \nu_k}[(\mu_k - 1) \cdot \psi(b_0)] - \frac{\partial}{\partial \nu_k}[(\mu_k - 1) \cdot \psi(b_{1i})] + \frac{\partial}{\partial \nu_k}[\psi(\nu_k)] \Bigg),$$

giving us the boxed expression

$$\boxed{\begin{aligned} \frac{\partial^2}{\partial \nu_k^2}[\mathcal{Q}_k] = \sum_{i=1}^{M} \gamma_{ik} \Bigg(& \mu_k^2 \cdot \psi_1(x_i + \mu_k \cdot \nu_k) \\ & - \mu_k^2 \psi_1(\mu_k \cdot \nu_k) \\ & - \psi_1(x_i + \mu_k \cdot \nu_k + n_i - x_i + (1 - \mu_k) \cdot \nu_k) \\ & + (\mu_k - 1) \cdot (1 - \mu_k) \cdot \psi_1((1 - \mu_k) \cdot \nu_k) \\ & - (\mu_k - 1) \cdot (1 - \mu_k) \cdot \psi_1(n_i - x_i + (1 - \mu_k) \cdot \nu_k) \\ & + \psi_1(\nu_k) \Bigg) \end{aligned}}.$$

B Selected simulation study tables

B.1 Computational runtimes, per M, Q

Recall that $K = 7, \nu = 100$.

B.1.1 Strict mixture model

M	Q	Mean Time (s)	Mean Iter.	Conv. Rate
10	1	0.09	36.1	0.99
10	5	0.33	38.7	0.98
10	20	0.23	7.3	1.00
10	70	0.53	4.7	1.00
50	1	0.32	99.1	0.82
50	5	0.89	74.9	0.90
50	20	0.82	18.5	1.00
50	70	1.07	6.7	1.00
200	1	0.91	159.0	0.46
200	5	2.57	116.0	0.70
200	20	3.77	44.6	0.97
200	70	5.67	18.6	0.99
500	1	1.84	181.0	0.24
500	5	5.80	141.0	0.51
500	20	11.40	71.2	0.88
500	70	20.40	35.2	0.96
2000	1	5.96	194.0	0.10
2000	5	21.10	161.0	0.32
2000	20	53.70	105.0	0.68
2000	70	133.00	70.8	0.82

Table 1: Computational time results by M and Q.

B.1.2 Hierarchical strict mixture model

M	Q	Mean Time (s)	Mean Iter.	Conv. Rate
10	1	0.10	39.2	0.99
10	5	0.16	18.6	1.00
10	20	0.18	5.4	1.00
10	70	0.47	4.1	1.00
50	1	0.36	110.0	0.80
50	5	0.30	25.5	1.00
50	20	0.31	6.6	1.00
50	70	0.68	4.1	1.00
200	1	0.99	173.0	0.38
200	5	1.07	48.0	0.96
200	20	1.39	16.1	1.00
200	70	1.87	5.9	1.00
500	1	1.95	192.0	0.14
500	5	2.71	65.8	0.90
500	20	4.73	29.4	0.99
500	70	7.47	12.7	1.00
2000	1	6.06	199.0	0.02
2000	5	12.50	95.7	0.67
2000	20	28.80	56.2	0.89
2000	70	60.20	32.0	0.96

Table 2: Computational time results by M and Q.

B.2 Search time to recover K

We list the worst 20 cases of our total search time for BIC-optimized K.

B.2.1 Strict mixture model

M	N	Q	α	K_{true}	Mean Time (s)	SD Time (s)
2000	100	70	0.5	3	4653	895
2000	100	70	0.1	4	4587	916
2000	100	70	0.8	3	4531	707
2000	100	70	0.1	5	4510	860
2000	100	70	0.1	6	4498	1173
2000	100	70	0.3	4	4495	792
2000	100	70	0.8	4	4401	720
2000	100	70	0.1	3	4383	901
2000	100	70	0.1	8	4354	1042
2000	100	70	0.3	3	4319	956
2000	100	70	0.5	4	4289	767
2000	100	70	0.3	5	4220	707
2000	100	70	0.8	5	4206	795
2000	100	70	0.3	6	4191	775
2000	100	70	0.1	7	4137	939
2000	100	70	0.3	7	3951	809
2000	100	70	0.5	5	3937	718
2000	100	70	0.5	6	3913	823
2000	1000	70	0.8	3	3756	785
2000	1000	70	0.5	3	3723	767

Table 3: 20 worst-case search runtimes by parameter combination.

B.2.2 Hierarchical strict mixture model

M	N	Q	α	K_{true}	Mean Time (s)	SD Time (s)
2000	100	70	0.3	3	4972	586
2000	100	70	0.5	3	4917	835
2000	100	70	0.8	3	4805	638
2000	100	70	0.1	3	4730	841
2000	100	70	0.5	4	4611	553
2000	100	70	0.8	4	4603	596
2000	100	70	0.1	4	4534	639
2000	100	70	0.3	4	4372	747
2000	1000	70	0.5	3	4063	689
2000	1000	70	0.3	3	4061	627
2000	100	70	0.1	5	4019	657
2000	100	70	0.3	5	4009	575
2000	1000	70	0.1	3	3999	687
2000	1000	70	0.8	3	3993	654
2000	100	70	0.8	5	3904	618
2000	1000	70	0.3	4	3874	694
2000	5000	70	0.5	3	3854	670
2000	30000	70	0.1	3	3791	764
2000	5000	70	0.1	3	3785	610
2000	100	70	0.5	5	3776	720

Table 4: Top 20 worst-case search runtimes by parameter combination.

C Maine referendum questions

We include the polled Maine referendum questions [2] below, for context of the above analysis.

2008
1. Repeals tax bill to fund the Dirigo Health Plan through beverage taxes
2. Allows the establishment of a casino in Oxford County
3. Issues $3.4 million in bonds for drinking water and wastewater treatment

2009
1. Repeals an act allowing same-sex marriages
2. Reduces excise taxes on vehicles that meet efficiency requirements
3. Repeals the law mandating school district restructuring
4. Limits increases in government spending to inflation and population increases
5. Creates nonprofit medical marijuana dispensaries and identification cards
6. Issues $71.25 million in bonds for transportation programs and improvements
7. Increases the amount of time for municipalities

2010
1. Authorizes a casino at a single site in Oxford County
2. Issues $5 million in bonds to increase access to dental care
3. Issues $9.75 million in bonds to invest in land conservation

2011
1. Would overturn a law dealing with same-day voting registration
2. Amend laws governing deadline for second racino and allow another tribal racino
3. Regarding establishing a slot machine facility
4. Would change the years of redistricting the Maine Legislature

2012
1. Would legalize same-sex marriage in the state
2. $11 million bond for higher education in order to expand the state's community college system
3. One of two measures for a bond for water and sewer projects in the state
4. Allow for a $51 million transportation bond
5. A bond for water and sewer projects in the state

2013
1. $14 million bond for maintenance for Maine Army National Guard readiness centers
2. $15.5 million bond to update and improve University of Maine system facilities
3. $100 million transportation bond to match estimated federal funds
4. $4.5 million bond for a public-private partnership at the Maine Maritime Academy
5. $15.5 million bond to upgrade buildings in the Maine Community College system

2014
1. Puts restrictions on certain bear hunting practices
2. Issues $8 million in bonds to support agriculture and human health monitoring
3. Issues $4 million in bonds to insure loans to small businesses
4. Issues $10 million for a research center for genetic solutions to cancer
5. Issues $3 million in bonds to modernize and expand laboratory in tissue repair
6. Issues $10 million in bonds to ensure clean water and restore wetlands
7. Issues $7 million in bonds to facilitate growth of marine businesses

2015
1. Revision of the Maine Clean Election Act
2. $15 million in bonds for affordable housing
3. $85 million in bonds for transportation

2016
1. Legalize marijuana for personal use
2. 3 percent tax on household income over $200,000
3. Background checks for gun sales and transfers
4. Increase minimum wage to $12 per hour by 2020

5. Establish ranked-choice voting
6. $100 million in bonds for transportation projects

2017
1. Casino or slot machines in York County
2. Expand Medicaid under the ACA
3. $105 million in bonds for highways, bridges, ports, railroads, and trails
4. Time to pay off unfunded liabilities from experience losses

2018
1. Creates a 3.8 percent payroll tax and non-wage income tax for home care program
2. Issues $30 million in bonds for wastewater infrastructure
3. $106 million in bonds for transportation infrastructure
4. $49 million in bonds for the University of Maine System
5. $15 million in bonds for the state's seven community colleges

2019
1. Authorizes $105 million in bonds for transportation infrastructure projects
2. Authorizes legislation allowing persons with physical disabilities to use alternative signatures to sign initiative petitions

2021
1. Prohibits the construction of electric transmission lines in the Upper Kennebec Region and requires a two-thirds legislative vote to approve high-impact electric transmission line projects
2. Issues $100 million in bonds for transportation infrastructure
3. Creates a state constitutional right to produce, harvest, and consume food

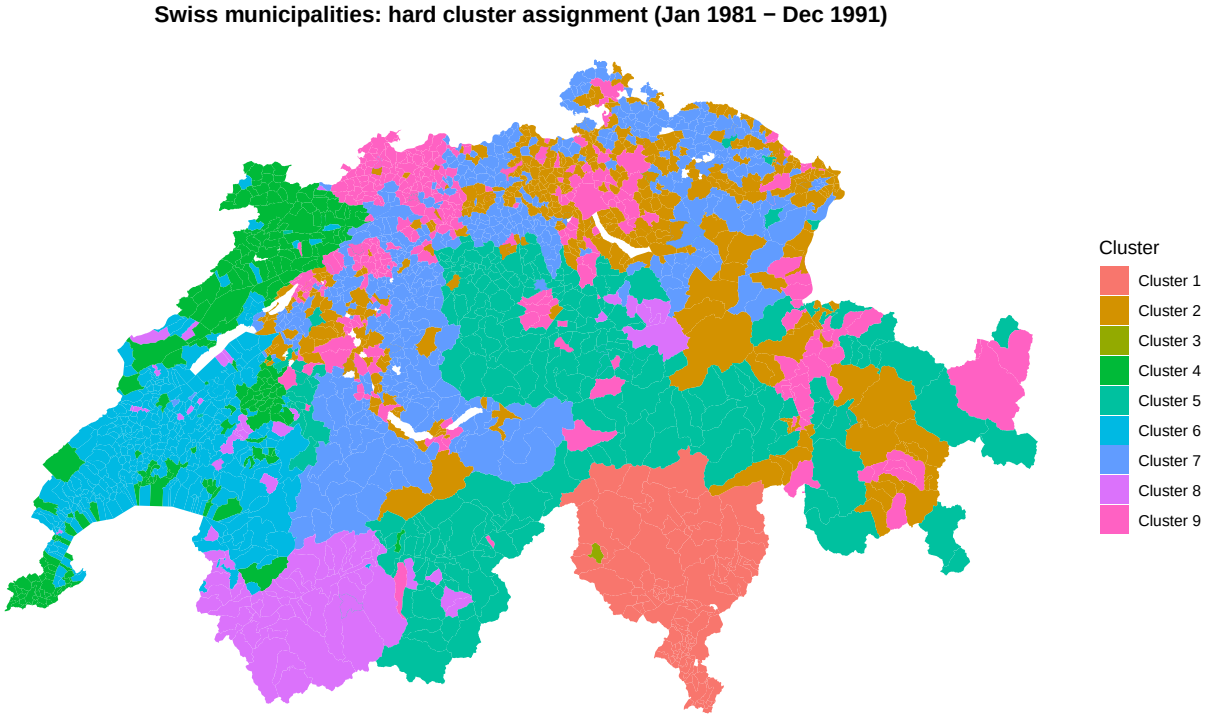
2023
1. Require voter approval for certain state entities or consumer-owned transmission utilities to incur outstanding debt exceeding $1 billion
2. Prohibit election spending by foreign governments, including entities with partial foreign government ownership or control
3. Create the Pine Tree Power Company, an electric transmission and distribution utility governed by an elected board
4. Allow motor vehicle owners and independent repair facilities to access vehicle on-board diagnostic systems
5. Change the judicial review period from within 100 days to within 100 business days from the deadline for filing a petition
6. Repeals Article X, Section 7 of the Maine Constitution concerning American Indian provisions
7. Remove the requirement that an initiative petition signature gatherer must be a resident and registered voter of Maine
8. Removes a constitutional provision, found unconstitutional in 2001, that bars individuals under guardianship for mental illness from voting

2024

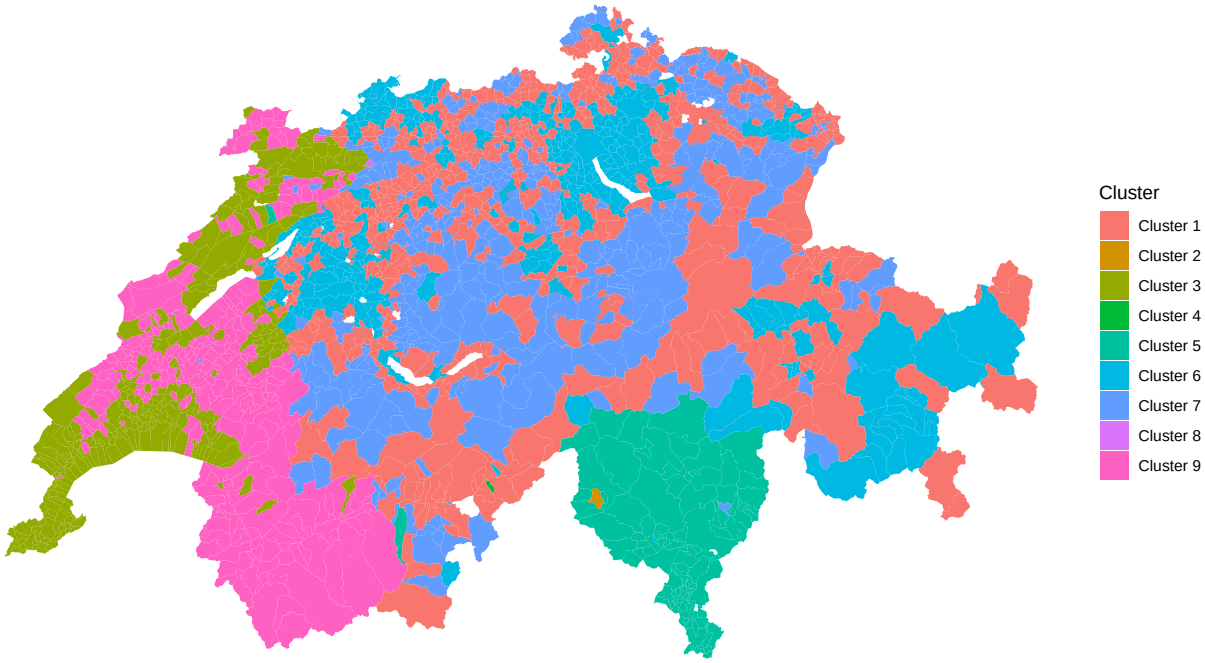
1. Limit campaign contributions to $5,000 from individuals and entities to political action committees making independent expenditures
2. Authorize $25 million in bonds for research, development, and commercialization for Maine-based institutions in support of technological innovation
3. Authorize $10 million in bonds for the restoration of local community buildings
4. Authorize $30 million in bonds for the development and maintenance of outdoor trails
5. Replace the Maine state flag with a flag consisting of a pine tree and the North Star on a buff background

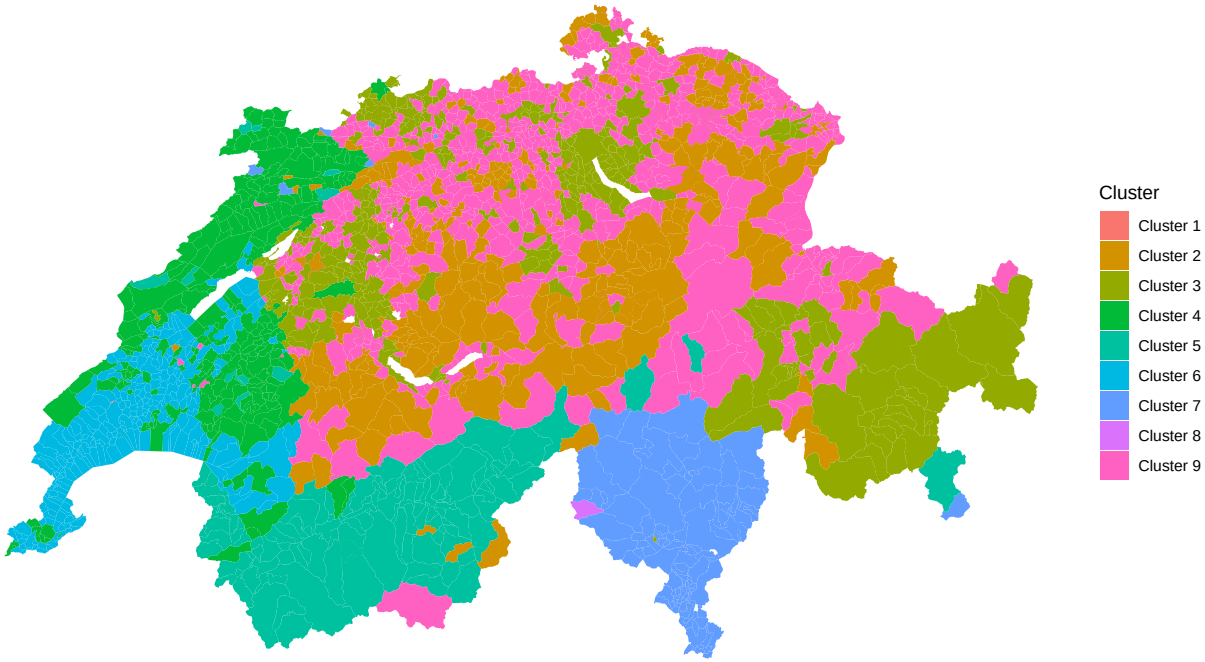
D Swiss hard clustering maps

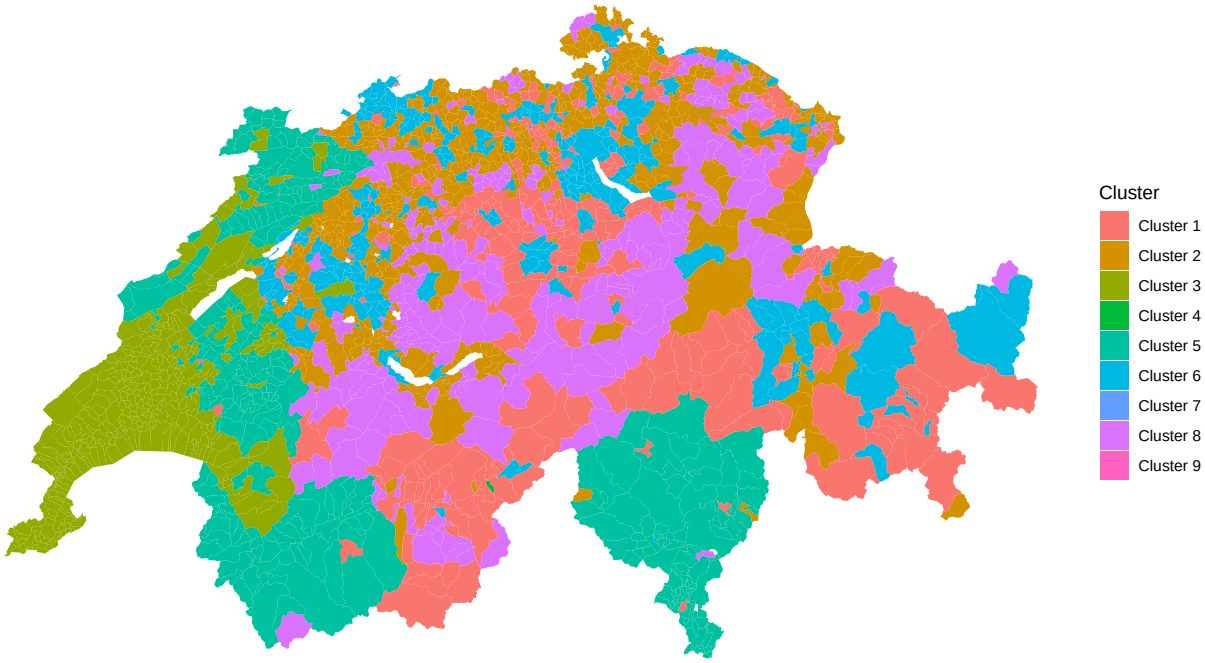
We include the four self consistent Swiss municipality clusterings across 10-year periods, to supplement the above analysis.



Swiss municipalities: hard cluster assignment (Jan 1992 − Dec 2002)




Swiss municipalities: hard cluster assignment (Jan 2003 − Dec 2013)
Cluster
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
Cluster 8
Cluster 9


Swiss municipalities: hard cluster assignment (Jan 2014 − Dec 2026)
Cluster
Cluster 1
Cluster 2
Cluster 3
Cluster 4
Cluster 5
Cluster 6
Cluster 7
Cluster 8
Cluster 9

Bibliography

[1] Hirotogu Akaike. "Information Theory and an Extension of the Maximum Likelihood Principle". In: *Selected Papers of Hirotugu Akaike*. Ed. by Emanuel Parzen, Kunio Tanabe, and Genshiro Kitagawa. Series Title: Springer Series in Statistics. New York, NY: Springer New York, 1998, pp. 199–213. ISBN: 978-1-4612-7248-9 978-1-4612-1694-0. DOI: 10.1007/978-1-4612-1694-0_15. URL: http://link.springer.com/10.1007/978-1-4612-1694-0_15.

[2] Ballotpedia. *List of Maine ballot measures*. en. Encyclopedia. 2026. URL: https://ballotpedia.org/List_of_Maine_ballot_measures.

[3] Ballotpedia. *Types of ballot measures in Maine*. 2026. URL: https://ballotpedia.org/Types_of_ballot_measures_in_Maine.

[4] Nathaniel Beck and Simon Jackman. "Beyond Linearity by Default: Generalized Additive Models". In: *American Journal of Political Science* 42.2 (Apr. 1998), p. 596. ISSN: 00925853. DOI: 10.2307/2991772. URL: https://www.jstor.org/stable/2991772?origin=crossref.

[5] David M. Blei, Andrew Y Ng, and Michael I. Jordan. "Latent dirichlet allocation". en. In: *JMLR.org* 3 (Mar. 2003), pp. 993–1022. ISSN: 1532-4435. URL: https://dl.acm.org/doi/10.5555/944919.944937.

[6] George Casella and Roger L. Berger. *Statistical inference*. eng. 2. ed. Pacific Grove, Calif: Duxbury, 2002. ISBN: 978-0-534-24312-8.

[7] Bela A. Frigyik, Amol Kapila, and Maya R. Gupta. *Introduction to the Dirichlet Distribution and Related Processes*. UWEE Technical Report UWEETR-2010-0006. Dec. 2010. URL: https://mayagupta.org/publications/FrigyikKapilaGuptaIntroToDirichlet.pdf.

[8] Sylvia Frühwirth-Schnatter. *Finite mixture and Markov switching models*. en. Springer series in statistics. Springer, 2006. ISBN: 978-0-387-32909-3.

[9] Isobel Claire Gormley and Thomas Brendan Murphy. "A mixture of experts model for rank data with applications in election studies". In: *The Annals of Applied Statistics* 2.4 (Dec. 2008). ISSN: 1932-6157. DOI: 10.1214/08-AOAS178. URL: https://projecteuclid.org/journals/annals-of-applied-statistics/volume-2/issue-4/A-mixture-of-experts-model-for-rank-data-with-applications/10.1214/08-AOAS178.full.

[10] Ian Hacking. *The emergence of probability: a philosophical study of early ideas about probability, induction and statistical inference*. eng. Second edition, reprinted. Cambridge: Cambridge University Press, 2007. ISBN: 978-0-521-68557-3 978-0-511-81755-7.

[11] Garrett Hellenthal et al. "A Genetic Atlas of Human Admixture History". en. In: *Science* 343.6172 (Feb. 2014), pp. 747–751. ISSN: 0036-8075, 1095-9203. DOI: 10.1126/science.1243518. URL: https://www.science.org/doi/10.1126/science.1243518.

[12] Ian Holmes, Keith Harris, and Christopher Quince. "Dirichlet Multinomial Mixtures: Generative Models for Microbial Metagenomics". en. In: *PLoS ONE* 7.2 (Feb. 2012). Ed. by Jack Anthony Gilbert, e30126. ISSN: 1932-6203. DOI: 10.1371/journal.pone.0030126. URL: https://dx.plos.org/10.1371/journal.pone.0030126.

[13] Gary King, Ori Rosen, and Martin A. Tanner, eds. *Ecological inference: new methodological strategies*. eng. Analytical methods for social research. Cambridge: Cambridge University Press, 2004. ISBN: 978-0-521-83513-8 978-0-511-22759-2.

[14] *Languages of Switzerland*. 2026. URL: https://en.wikipedia.org/wiki/Languages_of_Switzerland.

[15] Vera Leysinger, Benjamin von Wyl, and Pauline Turuban. *How Swiss direct democracy works*. en. Mar. 2025. URL: https://www.swissinfo.ch/eng/swiss-democracy/how-swiss-direct-democracy-works/89073820.

[16] Felix Luginbuhl. *BFS: Get Data from the Swiss Federal Statistical Office*. en. Institution: Comprehensive R Archive Network Pages: 0.7.1. Feb. 2026. DOI: 10.32614/CRAN.package.BFS. URL: https://CRAN.R-project.org/package=BFS.

[17] Allan McCutcheon. *Latent Class Analysis*. 2455 Teller Road, Thousand Oaks California 91320 United States of America: SAGE Publications, Inc., 1987. ISBN: 978-0-8039-2752-0 978-1-4129-8471-3. DOI: 10.4135/9781412984713. URL: https://methods.sagepub.com/book/latent-class-analysis.

[18] John D. O'Brien. *A Bayesian mixture model captures temporal and spatial structure of voting blocs within longitudinal referendum data*. en. arXiv:2301.01677 [stat]. Jan. 2023. DOI: 10.48550/arXiv.2301.01677. URL: http://arxiv.org/abs/2301.01677.

[19] John D. O'Brien, Kathryn Lin, and Scott MacEachern. "Mixture model of pottery decorations from Lake Chad Basin archaeological sites reveals ancient segregation patterns". In: *The Royal Society Publishing* 283.1827 (Mar. 2016). DOI: https://doi.org/10.1098/rspb.2015.2824.

[20] *Religion in Switzerland*. 2026. URL: https://en.wikipedia.org/wiki/Religion_in_Switzerland.

[21] Cynthia Rudin. "Gaussian Mixture Models and Expectation Maximization". en. In: *Intuition for the Algorithms of Machine Learning* (2020). URL: https://users.cs.duke.edu/~cynthia/CourseNotes/GMMEMNotes.pdf.

[22] Gideon Schwarz. "Estimating the Dimension of a Model". In: *The Annals of Statistics* 6.2 (Mar. 1978). ISSN: 0090-5364. DOI: 10.1214/aos/1176344136. URL: https://projecteuclid.org/journals/annals-of-statistics/volume-6/issue-2/Estimating-the-Dimension-of-a-Model/10.1214/aos/1176344136.full.

[23] Switzerland Federal Statistical Office. *opendata.swiss*. URL: https://opendata.swiss/de.