# ESTIMATION OF TIME-VARYING MIXTURE MODELS: AN APPLICATION TO TRAFFIC ESTIMATION

*Sean Lawlor and Michael G. Rabbat*

McGill University
Department of Electrical and Computer Engineering
Montréal, Québec, Canada

## ABSTRACT

Time varying mixture models can be a useful tool for modelling complex data collections. However the additional complexity of letting the number of mixture components vary over time adds even more difficulty in inference of the distribution parameters. We propose the automatic $k$-means algorithm to infer the parameters of these complex, time-varying mixture models. We demonstrate its performance using simulated and real data in a traffic estimation scenario.

## 1. INTRODUCTION

Time varying mixture models are a useful model for many complex distributions. However inference of mixture models with a time-varying number of components is typically difficult. Research into distributions other than a mixture of Gaussian distributions is limited in the literature. We explore a relatively little-studied mixture model, one of discrete-time Markov chains, and look at its application to road traffic estimation using *license plate recognition* (LPR) sensors. These sensors, also referred to *automatic number plate recognition* (ANPR), have opened many new opportunities of traffic pattern research. One of these research fields is to estimate the traffic flow in an online fashion. This paper provides a new and innovative method for modelling traffic on the road network using this kind of data as well as an inference method for estimating the model's parameters. Thorough performance analysis of the proposed inference algorithm is performed using both simulated and real data.

### 1.1. License Plate Recognition

Traffic today can be measured and estimated using the data retrieved from LPR sensors. These sensors provide discrete data about which vehicles pass by them. When a sensor detects a vehicle passing in its field of view, it records and reports meta information about the event (e.g., time, location, license plate number, etc).

From this data one would like to estimate the current road network traffic patterns. For all intents and purposes, the data is sampled from irregular locations on the road network and at irregular times, when a vehicle passes in the field of view. We assume that the meta-data recorded at each sensor is error free (e.g., there are no errors in license plate characters), so the only source of randomness is intrinsic to the traffic itself.

This work was funded in part by Genetec Inc., and by the Natural Sciences and Engineering Research Council of Canada

### 1.2. Literature Review

Previous work on time-varying mixture models typically adopts a Bayesian non-parametric perspective and focuses on Dirichlet process mixtures [1, 2]. Work on deterministic inference methods of these types of models is limited, and therefore time constraints in real-time, streaming applications is difficult. Also much of the previous work on time-varying mixture models focuses on mixtures of Gaussian distributions [3] which are difficult to apply to LPR data because of the discrete observations LPR data consists of. We propose a method using a more natural representation of time-varying mixture models for LPR and an inference algorithm which is not dependent on the distribution of the mixands.

Other work on modelling road traffic networks is available [4–6]. However these works typically use other sensors such as road counters, induction loops, cell phone data, and manual counting to estimate traffic patterns. Estimation of traffic network patterns using LPR has received limited study.

The algorithm proposed in this work includes a novel automatic selection of the mixture model order. Other work in selecting the model order has also been examined in the literature [7].

## 2. PROBLEM FORMULATION

We extend the static Markov chain mixture model introduced in [8] to allow the model parameters to vary over time. Let $\phi^{(m)}(t) = \{\pi^{(m)}(t), P^{(m)}(t)\}$ denote the parameters of the $m$th discrete-time Markov chain mixture component at time step $t$, where $\pi^{(m)}(t)$ is the initial state distribution and $P^{(m)}(t)$ is the transition matrix. The state space $\Omega$ is finite and contains one index for each LPR sensor. We describe a mixture model composed of $M(t)$ such mixture components at time $t$.

Observations of vehicle $i$ can be summarized by a starting state $x_0^i \in \Omega$ and a transition count matrix $X^i \in \mathbb{N}^{|\Omega|\times|\Omega|}$, where $X^i_{j,k}$ counts the number of times where two consecutive observations the vehicle were at sensor $j$ and then sensor $k$. We assume that the traffic model parameters remain constant over a sufficiently long time so that a vehicle may be observed multiple times before the vehicle changes. For example, one time step may correspond to one hour. The likelihood of the observations $x^i$ of vehicle $i$ in time window $t$ under component $m$ is given by

$$p\left(x^i \middle| \phi^{(m)}(t)\right) = \pi^{(m)}_{x_0^i}(t) \prod_{j=1}^{|\Omega|} \prod_{k=1}^{|\Omega|} \left(P^{(m)}_{j,k}(t)\right)^{X^i_{j,k}}. \quad (1)$$

The trajectory of each vehicle (i.e., the sequence of cameras where it is observed) is governed by one of the mixture components.

Let $z_i^{(m)}$ denote a binary random variable equal to 1 if and only if vehicle $i$ is governed by mixture component $m$, and let $\boldsymbol{\alpha}(t) = (\alpha^{(1)}(t), \ldots, \alpha^{(m)}(t))$ denote its distribution with $\Pr(z_i^{(m)} = 1) = \alpha^{(m)}(t)$ for a vehicle observed during time window $t$. Let $X(t) = \{x^1, \ldots, x^{N(t)}\}$ denote the set of $N(t)$ vehicles observed during time window $t$. The marginalized complete-data likelihood is

$$L(X(t)|\boldsymbol{\alpha}(t), \boldsymbol{\phi}(t)) = \prod_{i=1}^{N(t)} \sum_{m=1}^{M(t)} \alpha^{(m)}(t) p\left(x^i | \phi^{(m)}(t)\right) \quad (2)$$

where $M(t)$ is the number of mixture components at time step $t$ and $\boldsymbol{\phi}(t)$ is the collection all mixture component parameters. In this model we allow for the number of components and the parameters of each component to change at each timestep. The remainder of this section describes the dynamic model.

### 2.1. Mixture Component Death

Consider the mixture model at $t-1$ with $M(t-1)$ components. This mixture model can be described by the vector of mixture weights $\boldsymbol{\alpha}(t-1)$ and parameters of each component $\phi^{(m)}(t-1)$. Together these are the set of parameters, $\Theta(t-1)$, that describe the overall current model at time $t-1$. Existing components can persist or die, and they die with a probability $p_d$. For each component death, the mixture weight assigned to that component is distributed proportionally amongst the remaining components. This is similar to the model of Stephens [9].

### 2.2. Persistent Mixture Component Evolution

If a component persists from timestep $t-1$ to $t$ its parameters evolve according to the dynamics

$$P^{(m)}(t) \sim \begin{bmatrix} \mathrm{Dir}(P^{(m)}_{(1,:)}(t-1)) \\ \mathrm{Dir}(P^{(m)}_{(2,:)}(t-1)) \\ \vdots \\ \mathrm{Dir}(P^{(m)}_{(|\Omega|,:)}(t-1)) \end{bmatrix} := \mathbf{Dir}\left(P^{(m)}(t-1)\right) \quad (3)$$

and

$$\pi^{(m)}(t) \sim \mathrm{Dir}(\pi^{(m)}(t-1)) \quad (4)$$

where $\mathrm{Dir}(\cdot)$ is the Dirichlet distribution.

### 2.3. Mixture Component Birth

The number of mixture components born at each timestep follows a Poisson distribution with parameter $\lambda(t)$. Each new component born at iteration $t$ has parameters distributed according to the base distribution,

$$P^{(m)}(t) \sim \mathbf{Dir}\left(G_p\right) \quad (5)$$

$$\pi^{(m)}(t) \sim \mathrm{Dir}(G_\pi) \quad (6)$$

where $G_p$ and $G_\pi$ are the base distribution parameters. Each new component also has a mixture weight distributed according to a Beta distribution so that

$$\alpha^{(m)}(t) \sim \mathrm{Beta}(1, M(t)). \quad (7)$$

Following [9], for each new component added, the weights of previously existing components are scaled down by $(1 - \alpha^{(M(t)+1)})$ so that $\boldsymbol{\alpha}(t)$ always sums to 1. This model for component birth can be related to the notion that new components are "born from the prior" so they have some shared knowledge but are independent from existing components. In this model new components also receive progressively smaller weights which helps in that a new overwhelming component cannot appear out of nowhere. New components start off small and grow through their evolution. This notion is also borrowed from [9] and differs from the standard Dirichlet process mixture model in that the parameter to the Beta distribution can vary over time.

### 2.4. Mixture Weight Dynamics

Once the birth and deaths have occurred, the new set of mixture weights are distributed according to a Dirichlet distribution so $\boldsymbol{\alpha}(t) \sim \mathrm{Dir}(\boldsymbol{\alpha}(t-1))$. The births and deaths need to be established first so that the dimensionality of $\boldsymbol{\alpha}(t-1)$ will be known since with every birth and death it changes.

## 3. AUTOMATIC $K$-MEANS ESTIMATION FOR TIME-VARYING MIXTURE MODELS

The Expectation Maximization algorithm [10] is a popular estimation method for mixture models. It can be viewed as performing a form of clustering where the cluster assignments are soft: for each datapoint we get a likelihood that the datapoint came from each of the possible clusters. Then the cluster centers are moved to the weighted average of the data assigned to the cluster. By modifying the EM algorithm to make hard assignments we obtain the $k$-means algorithm [11].

From the generative perspective, to sample from a mixture model we first draw a mixture component and then sample according to the parameters of that component. Therefore if the data can first be clustered, then the estimation of each component's parameters can simply be computed using a maximum likelihood or maximum a posteriori estimator with the data assigned to that component.

The proposed mixture model inference algorithm is broken into three phases for each timestep $t$. In the first phase, the observed data is clustered according to the previous estimated mixture model using a modified $k$-means algorithm that allows for a possibly unlimited number of clusters to be created at each timestep. In the second phase components are trimmed if the amount of data assigned to the cluster falls below a threshold parameter, $L_\alpha$. In the last phase remaining components are merged if the KL divergence falls below a limit $L_{KL}$. Pseudocode is shown in Alg. 1 and each phase is described in more detail next.

### 3.1. Modified $k$-Means Clustering

The modification to the $k$-Means algorithm we propose allows the clustering algorithm to continually add new components as necessary, so $k$ does not need to be specified. The first phase begins by adding a new component with parameters $\{G_\pi, G_p\}$ (line 3). It then find the component which achieves the maximum likelihood for each datapoint (line 8). From the data assigned to each component, the MAP estimate is computed for each component (line 9). When computing the MAP estimate, $f(S_m|\hat{\phi}_i^{(m)})$, is the likelihood of the Markov chain $m$ and $g(\hat{\phi}_i^{(m)}|\hat{\phi}_0^{(m)})$ is a Dirichlet distribution with parameters $\hat{\phi}_0^{(m)}$. These are the distributions outlined in (3) and (4).

In line 11 of the algorithm, we look to see if any data has been assigned to the last component in the set. This component is a base-distribution component and is a much more vague prior than the

**Algorithm 1** Automatic $k$-Means algorithm for $T$ timesteps

1: $\Theta(0) = \{\}$
**Require:** Data $\boldsymbol{X} = \{X(1), ..., X(T)\}$
2: **for** $t \in 1..T$ **do**
3: Initialize: $\hat{\phi}_0 = \{\phi^{(m)}(t-1)\}^{\forall m} \bigcup \{\{G_\pi, G_p\}\}$
4: $i = 0$
5: **repeat**
6: $i = i + 1$
7: **for** $m \in \{1, ..., M(t)\}$ **do**
8: $S_m = \left\{ x^j : p(x^j | \hat{\phi}_{i-1}^{(m)}) > p(x^j | \hat{\phi}_{i-1}^{(m')}) \forall m' \neq m \right\}$
9: $\hat{\phi}_i^{(m)} = \arg\max_{\hat{\phi}_i^{(m)}} f(S_m | \hat{\phi}_i^{(m)}) g(\hat{\phi}_i^{(m)} | \hat{\phi}_0^{(m)})$
10: **end for**
11: **if** $|S_{M(t)}| > 0$ **then**
12: $\hat{\phi}_i(t) = \{\phi^{(m)}(t)\}^{\forall m} \bigcup \{\{G_\pi, G_p\}\}$
13: **end if**
14: **until** $|\hat{\phi}_i^{(m)} - \hat{\phi}_{i-1}^{(m)}| = 0, \forall m$
15: $\alpha^{(m)}(t) \leftarrow \frac{|S_m|}{|X(t)|}, \forall m$
16: $\Theta(t) \leftarrow \left\{ (\alpha^{(1)}(t), \hat{\phi}_i^{(1)}(t)), ..., (\alpha^{(M(t))}, \hat{\phi}_i^{(M(t))}(t)) \right\}$
17: Trim components in $\Theta(t)$ with $\alpha^{(m)} < L_\alpha$
18: Merge components in $\Theta(t)$ with
19: $D_{KL}(\phi^{(m)}(t) || \phi^{(m')}(t)) < L_{KL}$
20: **end for**

components propagated from the previous timestep. If data is assigned to this component, then at least one new cluster has formed, so we add another component from the base-distribution to see if more than one new component has formed (line 12). This repeats until the component assignments and parameter estimates stop changing (line 14).

This clustering scheme allows us to see if additional components are born and if existing ones move. However if this were the only step of the algorithm, then the number of components would continuously grow without limit between timesteps. Therefore we need a method of annealing the number of components to those which are necessary and not over-fitting the data.

### 3.2. Trimming Weak Components

The most basic way of trimming off components which do not describe the data is to remove components where the proportion of data ($\alpha^{(m)}(t)$) is very small. This is equivalent to stating that components which only help describe a small portion of data are not worth the additional model complexity (line 17). This threshold, $L_\alpha$, could be constant, a timestep-dependent value, or a datasize-dependent value. For example if 10,000,000 datapoints are present, a component explaining 1% of the data may be considered significant, and maybe the component should persist. In other scenarios, one might wish to keep all components which are sufficiently well-separated from the rest, so $L_\alpha$ could be set to 0.

### 3.3. Merging Similar Component

A more complex notion is examining when two components become very close to each other (meaning they are essentially the same distribution). After the $k$-means clustering has converged, we compute the Kullback-Leibler (KL) divergence between all pairs of components and merge pairs if the KL divergence from one to the other is below a threshold, $L_{KL}$, starting from the smallest difference and proceeding sequentially. To merge components $m$ and $m'$ we set

$$\alpha^{\text{merged}}(t) = \alpha^{(m)}(t) + \alpha^{(m')}(t)$$

$$\phi^{\text{merged}}(t) = \frac{\alpha^{(m)}(t)\phi^{(m)}(t)}{\alpha^{(m)}(t) + \alpha^{(m')}(t)} + \frac{\alpha^{(m')}(t)\phi^{(m')}(t)}{\alpha^{(m)}(t) + \alpha^{(m')}(t)}$$

which is the weighted average of the two components based on their respective mixture weight $\alpha$.

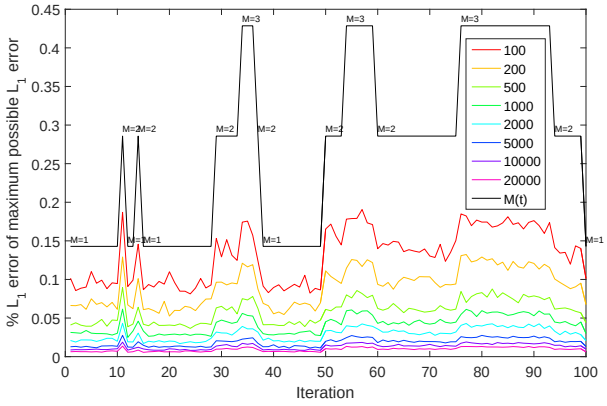



**Fig. 1**. Plot of the percent $L_1$ error for each trial with the number of components super-imposed so the rise in error with increase of the number of components is visible.

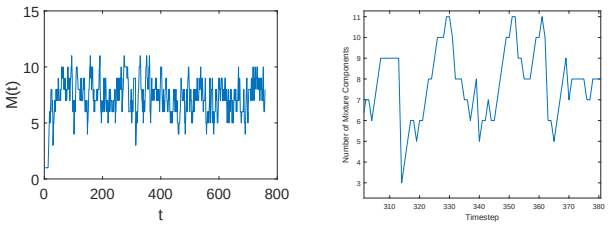
## 4. NUMERICAL EXPERIMENTS

We now continue with a performance evaluation of the automatic $k$-means algorithm using both simulated and real data. The simulated data contains 100 timesteps of a varying number of observations. The model for the dataset was created by generating a random network of 25 nodes. Then a base distribution for the transition matrix, $G_p$, was created by allowing all possible links to have equal transition weight. The base distribution for the initial distributions, $G_\pi$, is a uniform vector. Also the trimming component parameters $L_\alpha$ and $L_{KL}$ are set to 0 and 0.04 respectively. Then the birth-death generative process was run with Poisson births with parameter $\lambda = 0.1$ as well as every component having a probability of death of 0.1 for all timesteps. Once a sequence of 100 mixtures of Markov chains is created, then datasets from each timestep were generated with a varying number of observations $O \in \{100, 200, 500, 1000, 2000, 5000, 10000\}$.

Evaluation of Alg. 1 was then performed by examining both the estimated number of components and the relative $\ell_1$ error: the ratio of the $\ell_1$ error to the maximum possible $\ell_1$ error of the combined parameters $\boldsymbol{\alpha}$ and $\boldsymbol{\phi}$. Recall that the $\ell_1$ error between two multinomial distributions is at most 2. The parameters $\boldsymbol{\alpha}$ comprise one multinomial distribution, and the parameters of one Markov chain mixture component comprise $|\Omega| + 1$ mixture components (the initial state distribution and one for each row of the transition matrix), so the maximum $\ell_1$ error for a model described by $M(t)$ mixture components is

$$2\big(1 + (1 + |\Omega|)M(t)\big).$$

Even with only 100 observations per iteration the algorithm correctly selects the number of components at every timestep. This is




(a) (b)

**Fig. 2**. (a) The estimated number of components in the Markov mixture model ($M(t)$) estimated from the real dataset. Each timestep ($t$) represents 1 hour of data from the real dataset. (b) A portion of the estimated number of components on the real dataset zoomed into a 2-day window to view the cyclic behavior.

a good indicator but we can also investigate the error with a varying number of components. As can be seen in Fig. 1 the error of the parameter estimates steadily decreases with the increase in the amount of data which agrees with what we would expect. The error does peak when the number of components grows, which agrees with what one would expect since the number of observations per iteration is not proportional to the number of components. Therefore each component receives an increasingly smaller share of data with which to estimate its components when $M(t)$ grows.

### 4.1. Real Data Analysis

Next we run Alg. 1 on a real LPR dataset provided by a corporate partner. The dataset is a collection of LPR reads from a network of 20 LPR cameras over a period of 31 days. We define a timestep as one hour so there are a total of 758 timesteps over the 31 days. In every timestep, we select the data collected in that time window and create observation sequences for the vehicles which appear in that window. From the data in each timestep, we estimate the mixture model using Alg. 1. The threshold parameters $L_\alpha$ and $L_{KL}$ are set to 0 and 0.04 respectively, the same as in the simulated data case. The estimated number of components is shown in Fig. 2(a). We can see that the estimated mixture model is quite complex, and can have up to 11 components. However upon closer inspection, we see a cyclic behavior, as present in Fig. 2(b), where the model is most complex at rush-hour in the morning and afternoon and least-complex during the late evening and early morning. This agrees with one's intuition about traffic simply due to the fact that at rush-hours, the volume of traffic traversing the roads will be greatest therefore the largest number of paths are likely to be observed.

We can also get a feel for whether the algorithm is overfitting by looking at the KL divergence between all pairs of components. To illustrate this we take the model fit at timestep $t = 18$ and obtain

$$\begin{pmatrix} 0 & 3.242 & 2.124 & 1.344 & 3.075 & 2.012 \\ 1.1915 & 0 & 2.402 & 1.903 & 1.619 & 2.137 \\ 1.1608 & 0.924 & 0 & 1.018 & 5.075 & 3.638 \\ 0.4282 & 1.844 & 5.538 & 0 & 3.326 & 2.140 \\ 0.8047 & 5.590 & 1.168 & 9.198 & 0 & 2.423 \\ 2.8531 & 3.782 & 2.680 & 2.64244 & 1.884 & 0 \end{pmatrix}$$

where each row and column corresponds to one of the six mixture components fit at this time. We observe that the algorithm appears to learn well-separated components since all of the divergences for different distributions are much larger than $L_{KL} = 0.04$.

### 4.2. Comparison to Gibbs Sampling

The method outlined in Stephens [9] was also applied to the generative model in Section 2. This algorithm was run on a singular timestep with a mixture of 3 components and 100 datapoints. It was run for 5,000 burnin iterations and 50,000 sampling iterations. The best candidate model, with the correct number of components, chosen in the sampling had a relative $\ell_1$ error of 0.308779 which is much larger than the relative $\ell_1$ error obtained by the proposed algorithm on the same timestep with the same set of observations of 0.1575. Upon closer inspection, we note that the Gibbs sampler appears to be sampling poor model choices due to the high parameter space, which we know because they are immediately rejected by the sampler from [9]). It simply alternates between 2 and 3 mixture components and does not explore the space of possible models well.

## 5. CONCLUSION

This paper proposes a novel approach to the estimation of a time-varying mixture of Markov chains. The model specified in this paper enables complex, time-varying, network models to be modelled using LPR data. We then go on to define an algorithm to estimate the parameters of this time-varying mixture model and specifically apply it to LPR data. The performance of the algorithm is assessed using simulated data showing that it provides an accurate fit with even a limited amount of data.

We then look at the estimated transition matrices/traffic flows from a real network of LPR sensors provided from an industry partner. We showed that this method finds well-separated Markov chains in the mixture model which we believe demonstrates multiple traffic flows present within the same network of LPR sensors.

Future planned work includes extending the inference algorithm to other types of evolutionary clustering problems. This includes those which are not estimating discrete-time Markov chains.

## 6. REFERENCES

[1] F. Caron, M. Davy, and A. Doucet, "Generalized Polya urn for time-varying Dirichlet process mixtures," in *Proceedings of the 23rd Conference on Uncertainty in Artificial Intelligence, UAI*, July 2007.

[2] K. R. Keane and J. J. Corso, *Pattern Recognition - Applications and Methods: International Conference, ICPRAM 2012 Vilamoura, Algarve, Portugal, February 6-8, 2012 Revised Selected Papers*, chapter Inferring Time Varying Dynamic Linear Model Parameters from a Mixture Model, pp. 37–50, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013.

[3] M.N. Abhijith, P.K. Ghosh, and K. Rajgopal, "Multi-pitch tracking using Gaussian mixture model with time varying parameters and grating compression transform," in *Acoustics, Speech and Signal Processing (ICASSP), 2014 IEEE International Conference on*, May 2014, pp. 1473–1477.

[4] M. G. Singh and H. Tamura, "Modelling and hierarchical optimization for oversaturated urban road traffic networks," *International Journal of Control*, vol. 20, pp. 913–934, 1974.

[5] D. Valerio, A. D'Alconzo, F. Ricciato, and W. Wiedermann, "Exploiting cellular networks for road traffic estimation: A survey and a research roadmap," in *Vehicular Technology Conference, 2009. VTC Spring 2009. IEEE 69th*, 2009.

[6] B. S. Kerner, S. L. Klenov, and D. E. Wolf, "Cellular automata approach to three-phase traffic theory," *Journal of Physics A: Mathematical and General*, vol. 35, no. 47, pp. 9971–10013, November 2002.

[7] O. Shamir, "Stability and model selection in k-means clustering," *Machine Learning*, vol. 80, no. 2, pp. 213–243, September 2010.

[8] S. Lawlor, T. Sider, N. Eluru, M. Hatzopoulou, and M. G. Rabbat, "Detecting convoys using license plate recognition data," *IEEE Transactions on Signal and Information Processing over Networks*, 2016.

[9] M. Stephens, "Bayesian analysis of mixture models with an unknown number of components- an alternative to reversible jump methods," *The Annals of Statistics*, vol. 28, pp. 40–74, 2000.

[10] A.P. Dempster, N.M. Laird, and D.B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *J. of the Royal Statatistics Society Ser. B*, vol. 39, no. 1, pp. 1–38, 1977.

[11] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability, Volume 1: Statistics*, Berkeley, Calif., 1967, pp. 281–297, University of California Press.