Reducing the Cost of Fitting Mixture Models via Stochastic Sampling

Milan Papež[1] **Tomáš Pevný**[1] **Václav Šmídl**[1]

[1]Artifcial Intelligence Center, Czech Technical University, Prague, Czech Republic

Abstract

Traditional methods for unsupervised learning of finite mixture models require to evaluate the likelihood of all components of the mixture. This quickly becomes prohibitive when the components are abundant or expensive to compute. Therefore, we propose to apply a combination of the expectation maximization and the Metropolis-Hastings algorithm to evaluate only a small number of, stochastically sampled, components, thus substantially reducing the computational cost. The Markov chain of component assignments is sequentially generated across the algorithm's iterations, having a non-stationary target distribution whose parameters vary via a gradient-ascent scheme. We put emphasis on generality of our method, equipping it with the ability to train mixture models which involve complex, and possibly nonlinear, transformations. The performance of our method is illustrated on mixtures of normalizing flows.

1 INTRODUCTION

Finite mixture models [McLachlan et al., 2019] constitute a fundamental class of density estimation models. They are formed by a weighted sum of probability distributions—here referred to as *components*—and their training is commonly undertaken via the maximum likelihood estimation. This approach maximizes either (i) the marginal likelihood via gradient-ascent [Redner and Walker, 1984] or (ii) the evidence lower bound via variational methods [Blei et al., 2017, Humphreys and Titterington, 2000, Kucukelbir et al., 2017], including the expectation-maximization (EM) [Dempster et al., 1977]. The computational cost of such methods typically scales with $\mathcal{O}(TDNK)$ operations, where T is the number of iterations, D is the dimension of data, N is the number of data, and K is the number of components.

However, deploying these methods is challenging in applications involving large K, such as in face recognition [Otto et al., 2017], astronomical imaging [Welton et al., 2013], natural language processing [Nayak et al., 2014] and DNA data storage [Rashtchian et al., 2017]. The problem is even more severe for mixtures with components given by intricate models, including neural networks [Greff et al., 2017, Monnier et al., 2020], Gaussian processes [Wu and Ma, 2019], normalizing flows [Pires and Figueiredo, 2020]; or deep mixtures, such as sum-product (transform) networks [Peharz et al., 2020, Pevný et al., 2020]. In spite of this, a little attention has been paid to the design of algorithms that do not evaluate all K components. The notable exceptions are the sparse EM algorithm [Hughes and Sudderth, 2016] and the truncated EM algorithm [Forster and Lücke, 2018]. Moreover, the methods are mostly tailored for a fixed class of mixture models, e.g. Gaussian mixture models (GMMs).

In this paper, we make the following contributions:

- We instantiate the MCMC stochastic approximation EM (MCMCSAEM) framework [Kuhn and Lavielle, 2004] in the context of finite mixture models. This may seem wasteful, since the expectation over the component assignments in the EM objective function is analytically tractable, whereas the MCMCSAEM framework is notoriously applied when the expectation is intractable. However, notwithstanding this standard practice allows us to evaluate less components in mixture models and thus substantially reduce the computational cost of their training.
- We design our method to enable the maximization of generic EM objectives via a gradient-based form of stochastic approximation, making it suitable for components containing nonlinear transformations. This enhances the MCMCSAEM framework which has been almost exclusively used in cases admitting the maximization under a closed-form solution.
- We apply our method to mixtures of real-valued non-volume preserving (real NVP) flows [Dinh et al., 2017], reaching up-to 350$\times$ speed-up compared to the baseline EM algorithm.

Accepted for the 5th Workshop on Tractable Probabilistic Modeling at UAI (TPM 2022).

2 EXPECTATION MAXIMIZATION

2.1 THE EM ALGORITHM

The EM algorithm [Dempster et al., 1977] seeks the unknown parameters, $\theta \in \Theta$, maximizing the marginal (incomplete-data) log-likelihood in latent data models,

$$\mathcal{L}(\theta) \coloneqq \log p_\theta(\mathbf{x}) = \log \int p_\theta(\mathbf{x}, \mathbf{z}) d\mathbf{z}, \tag{1}$$

where $\mathbf{x} \coloneqq (x_i)_{i=1}^N$ are observed (known) variables, $x \in \mathsf{X}$; $\mathbf{z} \coloneqq (z_i)_{i=1}^N$ are latent (unknown) variables, $z \in \mathsf{Z}$; and $p_\theta(\mathbf{x}, \mathbf{z})$ is the joint (complete-data) likelihood.

The EM algorithm addresses this task indirectly, i.e. by optimizing the evidence lower bound (ELBO),

$$\mathcal{L}(\theta) \geq \mathcal{Q}(\theta, \hat{\theta}) + \mathcal{H}(\hat{\theta}) \coloneqq \text{ELBO}(\hat{\theta}), \tag{2}$$

where $\mathcal{H}(\hat{\theta}) \coloneqq -\mathsf{E}_{p_{\hat{\theta}}(\mathbf{z}|\mathbf{x})}[\log p_{\hat{\theta}}(\mathbf{z}|\mathbf{x})]$ is the differential entropy at an estimate, $\hat{\theta} \in \Theta$, and

$$\mathcal{Q}(\theta, \hat{\theta}) \coloneqq \mathsf{E}_{p_{\hat{\theta}}(\mathbf{z}|\mathbf{x})}[\log p_\theta(\mathbf{z}, \mathbf{x})] \tag{3}$$

is the EM objective function. Here, $p_\theta(\mathbf{z}|\mathbf{x})$ is the posterior distribution over $\mathbf{z}$. Given an initial value, θ_1, the algorithm produces a sequence of estimates, $(\theta_t)_{t=1}^T$, by alternating between the expectation (E) and maximization (M) steps,

$$\text{E-step: } \mathcal{Q}_t(\theta), \tag{4}$$

$$\text{M-step: } \theta_{t+1} \coloneqq \operatorname*{arg\,max}_{\theta \in \Theta} \mathcal{Q}_t(\theta), \tag{5}$$

where $\mathcal{Q}_t(\theta) \coloneqq \mathcal{Q}(\theta, \theta_t)$. The sequence is guaranteed to monotonically tighten (2), arriving at a local optimum of (1) under mild regularity assumptions [Wu, 1983].

2.2 THE MCMCSAEM ALGORITHM

The E-step (4) is analytically intractable in many applications. The MCEM algorithm [Wei and Tanner, 1990] addresses this problem by approximating (3) via the MC average, $\bar{\mathcal{Q}}_t(\theta) \coloneqq \frac{1}{M} \sum_{j=1}^M \log p_\theta(\mathbf{x}, \mathbf{z}_t^j)$, where the samples, $(\mathbf{z}_t^j)_{j=1}^M$, are drawn from $p_{\theta_t}(\mathbf{z}|\mathbf{x})$. However, this algorithm requires high values of M to converge [Fort and Moulines, 2003], and the samples are wastefully discarded at each iteration, t. The stochastic approximation resolves this issue by reusing the samples in $\bar{\mathcal{Q}}_t$ over the iterations as follows:

$$\widehat{\mathcal{Q}}_t(\theta) \coloneqq \widehat{\mathcal{Q}}_{t-1}(\theta) + \gamma_t\big(\bar{\mathcal{Q}}_t - \widehat{\mathcal{Q}}_{t-1}(\theta)\big), \tag{6}$$

where the step-size, γ_t, satisfies the constraints [Robbins and Monro, 1951], $\gamma_t \in [0, 1]$, $\sum_{t \geq 1} \gamma_t = \infty$, $\sum_{t \geq 1} \gamma_t^2 < \infty$.

The normalizing factor of $p_\theta(\mathbf{z}|\mathbf{x})$ is often intractable, preventing direct sampling from this posterior. MCMC obviates this difficulty by simulating a Markov chain, $(\mathbf{z}_t)_{t=1}^T$, from a transition kernel, $\mathbf{z}_t \sim P_\theta(\mathbf{z}_{t-1}, \cdot)$, which leaves $p_\theta(\mathbf{z}|\mathbf{x})$ as its unique stationary distribution, given a fixed θ.

The MCMCSAEM algorithm [Delyon et al., 1999, Kuhn and Lavielle, 2004] approximates the E-step (4) by combining the MCMC simulation (S) and the stochastic approximation (SA) in (6),

$$\text{S-step: } \mathbf{z}_t^j \sim P_{\theta_t}(\mathbf{z}_t^{j-1}, \cdot), \qquad j \in (1, \ldots, M), \tag{7}$$

$$\text{SA-step: } \widehat{\mathcal{Q}}_t(\theta), \tag{8}$$

$$\text{M-step: } \theta_{t+1} \coloneqq \operatorname*{arg\,max}_{\theta \in \Theta} \widehat{\mathcal{Q}}_t(\theta). \tag{9}$$

This algorithm sets $\mathbf{z}_t^0 \coloneqq \mathbf{z}_{t-1}^M$ at each t and produces the chain $(\mathbf{z}_1^1, \ldots, \mathbf{z}_1^M, \ldots, \mathbf{z}_T^1, \ldots, \mathbf{z}_T^M)$ of length MT, initializing (7) with $\mathbf{z}_1^0$. The samples generated during the initial iterations are usually discarded due to their high correlation, which is often referred to as the burn-in [Robert and Casella, 2013]. In the MCMCSAEM algorithm, the initial samples do not have to be discarded, since they are sequentially forgotten via the step-size, γ_t, (forgetting factor) in (6), i.e. a specific form of the sequence, $(\gamma_t)_{t=1}^T$, handles the burn-in.

3 PROBLEM FORMULATION

A finite mixture model characterizes the relation between $x \in \mathsf{X} \subseteq \mathbb{R}^D$ and $z \in \mathsf{Z} \coloneqq \{1, \ldots, K\}$ as follows:

$$p_\theta(x) = \sum_{k=1}^K p_{\eta_k}(x|z=k) p_{\pi_k}(z=k), \tag{10}$$

where $\theta \coloneqq (\pi_1, \eta_1, \ldots, \pi_K, \eta_K)$ are unknown parameters. η_z are the parameters of the conditional likelihood, $p_{\eta_z}(x|z)$, and π_z is the weight parameterizing the prior, $p_{\pi_z}(z) = \pi_z$, such that $0 \leq \pi_k \leq 1$ for each $k \in \mathsf{Z}$ and $\sum_{k=1}^K \pi_k = 1$.

Given independent and identically distributed data, $\mathbf{x}$, our aim is to find the parameters maximizing (1) given by

$$\mathcal{L}(\theta) = \sum_{i=1}^N \log \sum_{k=1}^K p_{\eta_k}(x_i|z_i=k) p_{\pi_k}(z_i=k). \tag{11}$$

For (10), the integration in (1) becomes the summation, which is analytically tractable for all forms of $p_{\eta_z}(x|z)$. However, for high K, this summation in (11) is computationally costly, rendering the optimization objective presumably intractable. We would like to design an algorithm requiring only $M < K$ evaluations of $p_{\eta_z}(x|z)$ at each iteration, t.

4 THE GENERALIZED MHSAEM ALGORITHM

The application of the MCMCSAEM framework is notoriously motivated by analytical intractability of the E-step.

We go against this convention, and use it to reduce the computational cost of the EM algorithm in the context of finite mixture models, where the E-step—the finite sum expected value—is always tractable. Moreover, the MCMCSAEM algorithm involves a closed-form solution of the M-step. We release this assumption by allowing direct, gradient-based optimization of the EM objective.

4.1 E-STEP

The computational cost of the EM algorithm scales with $\mathcal{O}(TDNK)$. This is seen from (3) which, in the context of (10), factorizes as follows:

$$\mathcal{Q}_t(\theta) = \sum_{i=1}^{N} \sum_{k=1}^{K} p_{\theta_t}(z_i = k|x_i) \log p_\theta(z_i = k, x_i), \quad (12)$$

where we need to compute KN, D-dependent, summands at each $t \in (1, \dots, T)$. Indeed, the marginal factor, $p_{\pi_z}(z)$, of $p_\theta(x, z)$ is just the cheap categorical distribution; however, the conditional factor, $p_{\eta_z}(x|z)$, typically involves high-dimensional operations (e.g. the inversion of $D \times D$-dimensional covariance matrices in the GMMs).

To reduce the computational cost, we sample only $M \ll K$ random samples from $p_\theta(z_i|x_i)$, for each $i \in (1, \dots, N)$, enabling us to obtain the Monte Carlo average, $\bar{\mathcal{Q}}_t$, in (6). Note that direct sampling from $p_\theta(z_i|x_i)$ would not lead to any substantial decrease in the number of operations, since we have to first compute the normalizing factor, $p_\theta(x_i)$. This requires K expensive evaluations of $p_\theta(z_i, x_i)$, which is precisely what we want to avoid. The MCMC sampling in (7) facilitates sampling from $p_\theta(z_i|x_i)$ with the cost decreasing to just $M \ll K$ evaluations of $p_\theta(z_i, x_i)$ per iteration.

A concrete form of P_θ in (7) determines a resulting MCMC procedure. We chose the Metropolis-Hastings (MH) sampler (hence MHSAEM) which represents $P_{\theta_t}(z_{i,t}^{j-1}, z_{i,t}^{j})$ as follows: given $\bar{z} \coloneqq z_{i,t}^{j-1}$, draw a sample from the *proposal* distribution, $z \sim q(\cdot|\bar{z})$, compute the acceptance ratio,

$$\alpha(\bar{z}, z) \coloneqq \min\left\{1, \frac{p_{\eta_{z,t}}(x_i|z)\pi_{z,t}q(\bar{z}|z)}{p_{\eta_{\bar{z},t}}(x_i|\bar{z})\pi_{\bar{z},t}q(z|\bar{z})}\right\}, \quad (13)$$

and, if $u < \alpha(\bar{z}, z)$—where u is drawn from the uniform distribution, Uniform$(0, 1)$—accept the sample and set $z_{i,t}^{j} = z$; otherwise, set $z_{i,t}^{j} = \bar{z}$. We repeat this process for each $j \in (1, \dots, M)$, construing a set $\mathbf{z}_{i,t} = (z_{i,t}^{1}, \dots, z_{i,t}^{M})$. Recall that we set $z_{i,t}^{0} \coloneqq z_{i,t-1}^{M}$ at each iteration, i.e. the chain has the length MT (Section 2.2). The samples, $\mathbf{z}_{i,t}$, are then used to obtain the following MC average:

$$\bar{\mathcal{Q}}_t(\theta) = \frac{1}{M} \sum_{i=1}^{N} \sum_{z \in \mathbf{z}_{i,t}} \log p_{\eta_z}(x_i|z)\pi_z. \quad (14)$$

Using (14) to directly approximate (4) is inefficient and impractical (Section 2.2). Therefore, we utilize (14) in a type of stochastic approximation, as detailed in Section 4.2.

Algorithm 1: The generalized MHSAEM algorithm

Input: θ_1, $(z_{i,0}^{M})_{i=1}^{N}$, $(x_i)_{i=1}^{N}$ **Output**: $(\theta_t)_{t=1}^{T}$

```
for t ∈ (1, ..., T) or until convergence do
    for i ∈ (1, ..., N) do
        set z^0_{i,t} := z^M_{i,t-1}
        for j ∈ (1, ..., M) do
            set z̄ := z^{j-1}_{i,t}
            sample z ~ q(z|z̄)
            sample u ~ Uniform(0, 1)
            compute α(z̄, z) in (13)
            if u < α(z̄, z) then
                set z^j_{i,t} := z and z̄ := z
            else
                set z^j_{i,t} := z̄
            end if
        end for
        set z_{i,t} := (z^1_{i,t}, ..., z^M_{i,t})
    end for
    compute (14)
    compute (15) for k ∈ unique(z_{i,t})
    compute π_{k,t} := softmax(ν_t)_k for k ∈ Z
end for
```

4.2 M-STEP

If the M-step (5) cannot be computed under a closed-form solution, one can resort to direct gradient-based optimization of $\mathcal{Q}(\theta)$, where $\arg\max$ is replaced by one (or more) step(s) of a gradient-ascent technique. The EM algorithm is then referred to as the generalized EM algorithm [Wu, 1983]. To the best of our knowledge, this extension has not yet been applied in the MCMCSAEM framework.

We replace (6) by a stochastic gradient-ascent method, $\theta_t = \theta_{t-1} + \gamma_t \nabla_\theta \bar{\mathcal{Q}}_t(\theta)$, where ∇_θ is the gradient w.r.t. θ. This is also a form of the stochastic approximation [Robbins and Monro, 1951], where the computations made in $\nabla_\theta \bar{\mathcal{Q}}$ are accumulated via θ_t and reused over the iterations.

The parameters η_z have a different form based on a specific case of $p_{\eta_z}(x|z)$, whereas the parameters π_z of $p_{\pi_z}(z)$ form a fixed structure in (10). Therefore, without loss of generality, we split (9) into a generic part and a fixed part,

$$\eta_{k,t} = \eta_{k,t-1} + \gamma_t \nabla_{\eta_k} \bar{\mathcal{Q}}_t(\theta), \quad (15a)$$

$$\nu_{k,t} = \nu_{k,t-1} + \gamma_t \nabla_{\nu_k} \bar{\mathcal{Q}}_t(\theta), \quad (15b)$$

where—to ensure that the probabilities, $(\pi_{k,t})_{k=1}^{K}$, satisfy the constraints (Section 3)—we transform $\nabla_{\pi_k} \bar{\mathcal{Q}}$ via $\nu_k = \log \pi_k$ and optimize w.r.t. ν_k. Then, to obtain $(\pi_{k,t})_{k=1}^{K}$ from $\boldsymbol{\nu}_t \coloneqq (\nu_{k,t})_{k=1}^{K}$, we use the softmax function, i.e. $\pi_{k,t} \coloneqq \mathrm{softmax}(\boldsymbol{\nu}_t)_k \coloneqq \exp(\nu_{k,t}) / \sum_{l=1}^{K} \exp(\nu_{l,t})$.

The M-step (5) is also computationally costly for large K. This holds even when it can be reduced to closed-form updates of expected sufficient statistics with $p_{\eta_z}(x|z)$ belonging to the exponential family [Nguyen et al., 2020] (again, due to high D). Note that computing the gradients for all pairs in $(\nu_k, \eta_k)_{k=1}^{K}$ would be inefficient, es-

pecially since $\mathbf{z}_{i,t}$ contains only a small number of unique values of Z for $M \ll K$. Therefore, we further reduce the computational cost by computing $\nabla_{\eta_k} \bar{\mathcal{Q}}$ and $\nabla_{\nu_k} \bar{\mathcal{Q}}$ only for $k \in \text{unique}(\mathbf{z}_{i,t})$. By this last step, we achieved the sought decrease in the complexity of the EM algorithm from $\mathcal{O}(TDNK)$ to $\mathcal{O}(TDNM)$, where $M \ll K$. We summarize the proposed approach in Algorithm 1.

5 EXPERIMENTS

To demonstrate the key features of our algorithm—i.e. its low computational cost, competitive learning performance, and generality—we use it to train mixtures of flow models [Pires and Figueiredo, 2020]. Specifically, we transform each $p_{\eta_z}(x|z)$ in (10) via the real NVP flow [Dinh et al., 2017], relying on deep neural networks to flexibly adjust the learning capacity of each component. All experiments have been performed on a Slurm cluster equipped with Intel Xeon Scalable Gold 6146 with 384GB of RAM.

Experiment settings: We use 19 real datasets from the UCI database [Dua and Graff, 2017, Mangasarian and Wolberg, 1990, Little et al., 2007, Siebert, 1987], preprocessed in the same way as in [Pevný, 2016]. For each experiment, we randomly split the data into 64%, 16% and 20% for training, validation and testing, respectively. We calculate the average log-likelihood on the test set and measure the time to reach 95% of the maximal training log-likelihood. We change the number of components as follows: $K \in (2, 4, 8, 16)$. Each real NVP-based component in the mixture model relies on (i) the translation function parameterized via the multi-layer perceptron with a single hidden layer of 10 neurons, choosing the hyperbolic tangent activation function; and (ii) the scale function parameterized via the same network. We use the batch normalization [Dinh et al., 2017], and we stack two layers of the translation-scale transformation. We adopt the real NVP implementation from [Franců, 2020].

Algorithms: We compare the MHSAEM algorithm to the standard EM algorithm. The former computes only one component of the mixture ($M = 1$) per iteration, whereas the latter computes all the components, i.e. we expect a speed-up of the computations. The EM and MHSAEM methods optimize the EM objective functions (12) and (14), respectively. We use the automatic differentiation and the ADAM optimizer with the default settings [Kingma and Ba, 2014], finding $T = 1000$ as a sufficient amount of iterations.

Results: The results are presented in Table 1. Since each dataset may benefit from a different K, we show the test log-likelihood of the mixtures—selected via the best log-likelihood measured on the validation set—and the associated speed-up. It can be seen that the MHSAEM algorithm outperforms the EM algorithm on all but one dataset, and it provides a substantial speed-up on all datasets except one.

Table 1: The speed-up and test log-likelihood, $\mathcal{L}^{\text{test}}$, for the EM and MHSAEM algorithms. The test log-likelihood (higher is better) is computed for the best model, with the corresponding K, which is selected based on the validation log-likelihood. The speed-up is computed as the ratio of EM to MHSAEM, i.e. their time to reach 95% of the training log-likelihood. The results are averaged over five repetitions with different initial conditions. The likelihood is shown with its standard deviation. The higher test log-likelihood is highlighted with bold blue, and no speed-up is highlighted with red. The average rank is computed as the standard competition ("1224") ranking [Demšar, 2006] on each dataset (lower is better).

	Mixtures of real NVP flows				
	EM			**MHSAEM**	
dataset	**speed-up**	$\mathcal{L}^{\text{test}}$	K	$\mathcal{L}^{\text{test}}$	K
breast-cancer	75.69	17.72±2.59	16	**21.42±1.34**	16
cardio	31.87	-33.57±3.39	4	**-33.42±8.38**	4
ecoli	26.48	9.95±1.06	8	**17.62±0.80**	16
ionosphere	86.19	12.26±3.93	16	**15.91±2.16**	8
iris	340.63	-3.64±1.03	16	**-1.59±0.74**	2
telescope	94.61	-28.62±0.19	16	**-26.14±0.06**	16
blocks	45.68	-38.70±0.85	16	**-31.47±0.36**	16
parkinsons	354.77	30.86±2.17	16	**31.27±1.16**	16
pendigits	58.86	-67.00±0.17	16	**-59.87±0.96**	16
pima-indians	**0.09**	-27.95±0.25	16	**-20.62±1.27**	16
sonar	139.20	**66.85±4.37**	16	66.09±4.74	4
segment	71.96	-45.48±2.88	16	**-35.37±4.82**	16
vehicle	98.01	-59.27±0.41	16	**-56.28±0.44**	8
robot	47.89	-28.80±0.39	16	**-20.61±1.26**	16
waveform-1	143.91	-33.47±0.09	16	**-31.92±0.18**	16
waveform-2	162.95	-33.65±0.10	16	**-31.96±0.09**	8
wine	182.06	-17.99±1.29	8	**-16.37±0.86**	2
yeast	120.85	9.49±1.47	8	**21.92±0.35**	16
rank		1.95		**1.05**	

6 CONCLUSION

This paper has presented a method to decrease computational cost of fitting mixture models. The speed-up is achieved by using the MH sampling to evaluate only a single component per iteration. The experiments confirmed that the method significantly speeds-up the fitting time, and, importantly, it does not undermine the quality of the fit. In fact, the likelihood was better than that of the models fitted by the EM algorithm in more than 90% of cases. We attribute this to the stochastic sampling, which helps to escape from poor local optima. The proposed method has used a uniform proposal distribution. Despite outperforming the baseline EM algorithm, we conjecture that this limits the speed of convergence. Therefore, we believe that there is still a room for improvement. We will address this in future work.

Acknowledgements

The authors acknowledge the support of the GAČR grant no. GA22-32620S and the OP VVV funded project CZ.02.1.01/0.0/0.0/16_019/0000765 "Research Center for Informatics".

References

David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017.

Bernard Delyon, Marc Lavielle, Eric Moulines, et al. Convergence of a stochastic approximation version of the EM algorithm. *The Annals of Statistics*, 27(1):94–128, 1999.

Arthur P Dempster, Nan M Laird, and Donald B Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society: Series B (Methodological)*, 39(1):1–22, 1977.

Janez Demšar. Statistical comparisons of classifiers over multiple data sets. *The Journal of Machine Learning Research*, 7:1–30, 2006.

Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real NVP. In *5th International Conference on Learning Representations, ICLR 2017*, 2017.

Dheeru Dua and Casey Graff. UCI machine learning repository, 2017. URL `http://archive.ics.uci.edu/ml`.

Dennis Forster and Jörg Lücke. Can clustering scale sublinearly with its clusters? A variational EM acceleration of GMMs and k-means. In *International Conference on Artificial Intelligence and Statistics*, pages 124–132. PMLR, 2018.

Gersende Fort and Eric Moulines. Convergence of the Monte Carlo expectation maximization for curved exponential families. *Annals of Statistics*, 31(4):1220–1259, 2003.

Jan Franců. Continuousflows.jl. `https://github.com/janfrancu/ContinuousFlows.jl`, 2020.

Klaus Greff, Sjoerd van Steenkiste, and Jürgen Schmidhuber. Neural expectation maximization. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 6694–6704, 2017.

Michael C Hughes and Erik B Sudderth. Fast learning of clusters and topics via sparse posteriors. *arXiv preprint arXiv:1609.07521*, 2016.

K Humphreys and DM Titterington. Approximate Bayesian inference for simple mixtures. In *COMPSTAT*, pages 331–336. Springer, 2000.

Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.

Alp Kucukelbir, Dustin Tran, Rajesh Ranganath, Andrew Gelman, and David M Blei. Automatic differentiation variational inference. *The Journal of Machine Learning Research*, 18(1):430–474, 2017.

Estelle Kuhn and Marc Lavielle. Coupling a stochastic approximation version of EM with an MCMC procedure. *ESAIM: Probability and Statistics*, 8:115–131, 2004.

Max Little, Patrick McSharry, Stephen Roberts, Declan Costello, and Irene Moroz. Exploiting nonlinear recurrence and fractal scaling properties for voice disorder detection. *Nature Precedings*, pages 1–1, 2007.

Olvi L Mangasarian and William H Wolberg. Cancer diagnosis via linear programming. Technical report, University of Wisconsin-Madison Department of Computer Sciences, 1990.

Geoffrey J McLachlan, Sharon X Lee, and Suren I Rathnayake. Finite mixture models. *Annual review of statistics and its application*, 6:355–378, 2019.

Tom Monnier, Thibault Groueix, and Mathieu Aubry. Deep transformation-invariant clustering. In *Conference on Neural Information Processing Systems (NeurIPS 2020)*, 2020.

Richi Nayak, Rachel Mills, Christopher De-Vries, and Shlomo Geva. Clustering and labeling a web scale document collection using Wikipedia clusters. In *Proceedings of the 5th International Workshop on Web-scale Knowledge Representation Retrieval & Reasoning*, pages 23–30, 2014.

Hien D Nguyen, Florence Forbes, and Geoffrey J McLachlan. Mini-batch learning of exponential family finite mixture models. *Statistics and Computing*, pages 1–18, 2020.

Charles Otto, Dayong Wang, and Anil K Jain. Clustering millions of faces by identity. *IEEE transactions on pattern analysis and machine intelligence*, 40(2):289–303, 2017.

Robert Peharz, Antonio Vergari, Karl Stelzner, Alejandro Molina, Xiaoting Shao, Martin Trapp, Kristian Kersting, and Zoubin Ghahramani. Random sum-product networks: A simple and effective approach to probabilistic deep learning. In *Uncertainty in Artificial Intelligence*, pages 334–344. PMLR, 2020.

Tomáš Pevný. Loda: Lightweight on-line detector of anomalies. *Machine Learning*, 102(2):275–304, 2016.

Tomáš Pevný, Vašek Šmídl, Martin Trapp, Ondřej Poláček, and Tomáš Oberhuber. Sum-product-transform networks: Exploiting symmetries using invertible transformations. *arXiv preprint arXiv:2005.01297*, 2020.

Guilherme GP Pires and Mário AT Figueiredo. Variational mixture of normalizing flows. *arXiv preprint arXiv:2009.00585*, 2020.

Cyrus Rashtchian, Konstantin Makarychev, Miklós Z Rácz, Siena Ang, Djordje Jevdjic, Sergey Yekhanin, Luis Ceze, and Karin Strauss. Clustering billions of reads for DNA data storage. In *NIPS*, volume 2017, pages 3360–3371, 2017.

Richard A Redner and Homer F Walker. Mixture densities, maximum likelihood and the EM algorithm. *SIAM review*, 26(2):195–239, 1984.

Herbert Robbins and Sutton Monro. A stochastic approximation method. *The annals of mathematical statistics*, pages 400–407, 1951.

Christian Robert and George Casella. *Monte Carlo statistical methods*. Springer Science & Business Media, 2013.

J Paul Siebert. Vehicle recognition using rule based methods. Technical report, Turing Institute, 1987.

Greg CG Wei and Martin A Tanner. A Monte Carlo implementation of the EM algorithm and the poor man's data augmentation algorithms. *Journal of the American statistical Association*, 85(411):699–704, 1990.

Benjamin Welton, Evan Samanas, and Barton P Miller. Mr. scan: Extreme scale density-based clustering using a tree-based network of GPGPU nodes. In *SC'13: Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–11. IEEE, 2013.

C F Jeff Wu. On the convergence properties of the EM algorithm. *The Annals of statistics*, pages 95–103, 1983.

Di Wu and Jinwen Ma. An effective EM algorithm for mixtures of Gaussian processes via the MCMC sampling and approximation. *Neurocomputing*, 331:366–374, 2019.