Approximate Inference for Generic Likelihoods via Density-Preserving GMM Simplification

Lei Yu **Tianyu Yang** **Antoni B. Chan**
Department of Computer Science
City University of Hong Kong
{leiyu6-c, tianyyang8-c}@my.cityu.edu.hk, abchan@cityu.edu.hk

Abstract

We consider recursive Bayesian filtering where the posterior is represented as a Gaussian mixture model (GMM), and the likelihood function as a sum of scaled Gaussians (SSG). In each iteration of filtering, the number of components increases. We propose an algorithm for simplifying a GMM into a reduced mixture model with fewer components, which is based on maximizing a variational lower bound of the expected log-likelihood of a set of virtual samples. We also propose an efficient algorithm for approximating an arbitrary likelihood function as an SSG. Experiments on synthetic 2D GMMs, simulated belief propagation and visual tracking show that our algorithm can be widely used for approximate inference.

1 Introduction

Recursive Bayesian filtering estimates the current state of a system using noisy measurements from the past and present. Common applications in computer vision include object tracking [1–10] and robot/vehicle localization [11–14]. A typical framework for a first-order Markov model is given by the predict-update equations:

$$\text{prediction:} \quad p(x_t|y_{1:t-1}) = \int p(x_t|x_{t-1})p(x_{t-1}|y_{1:t-1})dx_{t-1}, \tag{1a}$$

$$\text{update:} \quad p(x_t|y_{1:t}) = \frac{p(y_t|x_t)p(x_t|y_{1:t-1})}{\int p(y_t|x_t)p(x_t|y_{1:t-1})dx_t}, \tag{1b}$$

where x_t is the state at time t (e.g., object location) and y_t is the observation at time t (e.g., video frame). The posterior distribution of the current state $p(x_t|y_{1:t})$, conditioned on the observations so far $y_{1:t} = (y_1, \ldots, y_t)$, is obtained by first predicting the current state x_t using the previous posterior distribution $p(x_{t-1}|y_{1:t-1})$ and the transition model $p(x_t|x_{t-1})$ (Eq. 1a), and then factoring in the current observation y_t using the observation model $p(y_t|x_t)$ (Eq. 1b). Assuming the transition and observation models to be Gaussians yields the Kalman filter, which is tractable to compute. However, more complex models cannot be well represented with Gaussians, and non-conjugate likelihood functions make inference intractable. One solution is to use a particle filter [1–8, 10, 15, 16], where the posterior is approximated as a set of weighted particles. However, the limited set of samples may not well characterize the true posterior especially when the distribution is heavy-tailed, and errors may accumulate quickly during inference. Increasing the number of particles increases the accuracy of the posterior, but also increases the variance [17] and the computational load.

Considering that Gaussian mixture models (GMMs) are universal approximators for any continuous probability density [18–20], a generic approximate inference algorithm could represent the posterior as a GMM, and the likelihood function as a sum of scaled Gaussians (SSG; not necessarily integrating to 1). The prediction and update procedure shown in (1) would only involve the product of Gaussian mixtures and the integral is also tractable. However, the problem would be the exponential increase in the number of components in the posterior GMM.

30th Conference on Neural Information Processing Systems (NIPS 2016), Barcelona, Spain.

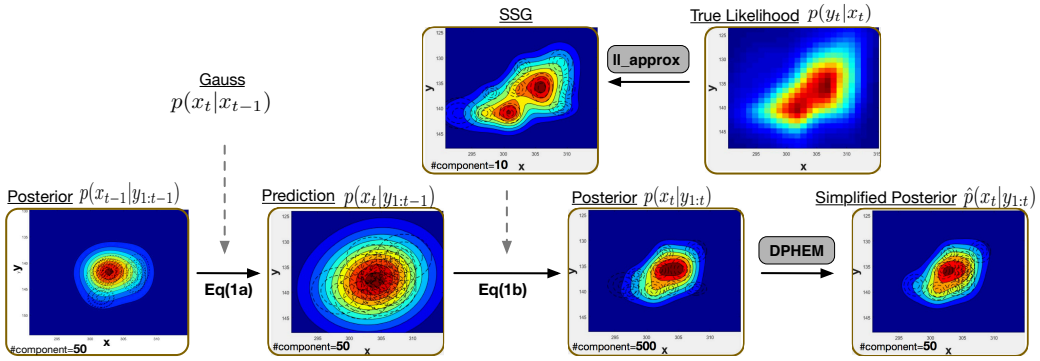


Figure 1: Framework for approximate recursive Bayesian filtering with Gaussian mixture model (GMM). The posterior and prediction are represented as a GMM, and the likelihood is approximated as a sum of scaled Gaussians (SSG). The new posterior is the product of the prediction and the likelihood, which results in an increase in the number of components. A tractable simplified posterior is obtained by applying our proposed GMM simplification algorithm, DPHEM. This example is from the visual tracking experiment (see Section 4.3).

In this paper, we propose a simplifying algorithm for GMMs to keep the number of components at a tractable level, while also preserving density structure. To facilitate the use of this general recursive Bayesian filtering framework for any likelihood function, we also propose an efficient algorithm for approximating an arbitrary likelihood function as an SSG. The framework for approximate recursive Bayesian filtering is shown in Figure 1.

2 Density-preserving hierarchical EM algorithm (DPHEM)

Suppose that the density of $y \sim \Theta^{(b)}$ is given by a mixture model $p(y|\Theta^{(b)}) = \sum_{i=1}^{K_b} \pi_i^{(b)} p(y|\theta_i^{(b)})$. Our goal is to simplify the base model $\Theta^{(b)}$ to a "reduced" mixture $\Theta^{(r)}$ with much fewer components $K_r \ll K_b$, namely, $p(y|\Theta^{(r)}) = \sum_{j=1}^{K_r} \pi_j^{(r)} p(y|\theta_j^{(r)})$. Note that we will always use i and j to index the base and reduced mixture components, respectively.

We take our inspiration from HEM [21], which is a hierarchical EM algorithm for clustering mixture components directly from the parameters of the base mixture components using a set of *virtual samples*. However, [21] does not preserve the density structure of the base mixture.

We define a set of i.i.d. virtual samples $Y = \{y_1, y_2, \cdots, y_N\}$ with each $y_n \sim \Theta^{(b)}$. The reduced model $\Theta^{(r)}$ is obtained by maximizing the *expected* log-likelihood of the reduced model $\Theta^{(r)}$ with respect to the virtual samples,

$$\mathcal{J}(\Theta^{(r)}) = \mathbb{E}_{Y|\Theta^{(b)}}[\log p(Y|\Theta^{(r)})] = \sum_i \pi_i^{(b)} \mathbb{E}_{Y|\theta_i^{(b)}}[\log p(Y|\Theta^{(r)})]. \tag{2}$$

Since the maximization of the expected log-likelihood of a mixture model $\mathbb{E}_{Y|\theta_i^{(b)}}[\log p(Y|\Theta^{(r)})]$ is intractable. We use a variational perspective of the EM algorithm [22–24], which treats the E and M-step both as maximization problems, and take the expectation of the variational log-likelihood to obtain a variational lower bound of the expected log-likelihood (see Supplemental),

$$\mathcal{J}_{DP}(\Theta^{(r)}) = \max_{z_{ij}} \sum_i \sum_j \pi_i^{(b)} z_{ij} \left\{ \log \frac{\pi_j^{(r)}}{z_{ij}} + N \mathbb{E}_{y|\theta_i^{(b)}}[\log p(y|\theta_j^{(r)})] \right\} \leq \mathcal{J}(\Theta^{(r)}). \tag{3}$$

The variational parameters z_{ij} can be interpreted as an assignment of virtual samples generated from the ith base component to the jth reduced component. The variational lower bound (3) is maximized by iterating between maximizing w.r.t. the assignments z_{ij} and the reduced mixture parameters $\Theta^{(r)}$.

For GMMs, the variational parameter update is $\hat{z}_{ij} = \frac{\pi_j^{(r)} \exp(N \mathbb{E}_{y|\theta_i^{(b)}}[\log p(y|\theta_j^{(r)})])}{\sum_{j'=1}^{K_r} \pi_{j'}^{(r)} \exp(N \mathbb{E}_{y|\theta_i^{(b)}}[\log p(y|\theta_{j'}^{(r)})])}$, where the expected log-Gaussian between $\theta_i^{(b)}$ and $\theta_j^{(b)}$ is $\mathbb{E}_{y|\theta_i^{(b)}}[\log p(y|\theta_j^{(r)})] = \log \mathcal{N}(\mu_i^{(b)}|\mu_j^{(r)}, \Sigma_j^{(r)}) -$

$\frac{1}{2}\text{tr}\{(\Sigma_j^{(r)})^{-1}\Sigma_i^{(b)}\}$. Given the variational parameters, the reduced model parameters $\Theta^{(r)} = \{\pi_j^{(r)}, \mu_j^{(r)}, \Sigma_j^{(r)}\}$ are updated using the base model parameters: $\hat{N}_j = \sum_{i=1}^{K_b} \hat{z}_{ij}\pi_i^{(b)}$, $\hat{\pi}_j^{(r)} = \sum_{i=1}^{K_b} \pi_i^{(b)}\hat{z}_{ij}$, $\hat{\mu}_j^{(r)} = \frac{1}{\hat{N}_j}\sum_{i=1}^{K_b} \hat{z}_{ij}\pi_i^{(b)}\mu_i^{(b)}$, $\hat{\Sigma}_j^{(r)} = \frac{1}{\hat{N}_j}\sum_{i=1}^{K_b} \hat{z}_{ij}\pi_i^{(b)}[\Sigma_i^{(b)} + (\mu_i^{(b)} - \hat{\mu}_j^{(r)})(\mu_i^{(b)} - \hat{\mu}_j^{(r)})^T]$.

3 Likelihood approximation

In this section, we propose an algorithm that iteratively computes a lower bound of a likelihood function, $f(x) = p(y|x)$, using a set of state-likelihood pairs, $\mathcal{D} = \{(x_i, p_i)\}_{i=1}^N$, where $p_i = p(y_i|x_i)$. In each iteration k, a scaled Gaussian $f^{(k)}(x)$ is found that lower bounds the residuals between the current SSG and the points in $\mathcal{D}$, denoted as $\mathcal{D}^{(k)}$. Then $f^{(k)}(x)$ is added as a component to the SSG, and the next iteration proceeds. The algorithm is initialized with $\mathcal{D}^{(1)} = \mathcal{D}$.

More specifically, in the k-th iteration, first the highest point in $\mathcal{D}^{(k)}$ is found, $m = \text{argmax}_i \log p_i$. Next, the peak of a scaled Gaussian $f^{(k)}$ is anchored on the maximum point,

$$h^{(k)}(x) = -(x - x_m)^T W_k (x - x_m) + \ell_m, \quad f^{(k)}(x) = \exp(h^{(k)}(x)), \tag{4}$$

where $\ell_m = \log p_m$ and W_k is the precision matrix of the Gaussian. The precision matrix W_k is found by minimizing the square-error with $\mathcal{D}^{(k)}$ in the log-likelihood space, with the constraint that the log-Gaussian $h^{(k)}(x)$ is a lower bound of the points in $\mathcal{D}^{(k)}$,

$$W_k^* = \underset{W_k}{\text{argmin}} \ \frac{1}{2}\sum_{i=1}^{N}(\ell_i - h^{(k)}(x_i))^2 \quad \text{s.t. } \ell_i - h^{(k)}(x_i) \geq 0, \forall i, \tag{5}$$

where $\ell_i = \log p_i$. When $W_k = \text{diag}(w_k)$ is a diagonal matrix, then (5) is a quadratic program (see Supplemental). The constraints in (5) ensure that $f^{(k)}$ is a lower bound of $\mathcal{D}^{(k)}$. Finally, the residual points are calculated $r_i = p_i - f^{(k)}(x_i), \forall i$ and the next iteration is run on the residual data $\mathcal{D}^{(k+1)} = \{(x_i, r_i)\}_{i=1}^N$. After sufficient iterations to reduce all residuals to under a threshold, the approximate likelihood is $f(x) = \sum_k f^{(k)}(x)$. Because each iteration forms a lower bound to the residuals, $f(x)$ is a lower-bound of the original data $\mathcal{D}$.

4 Experiments

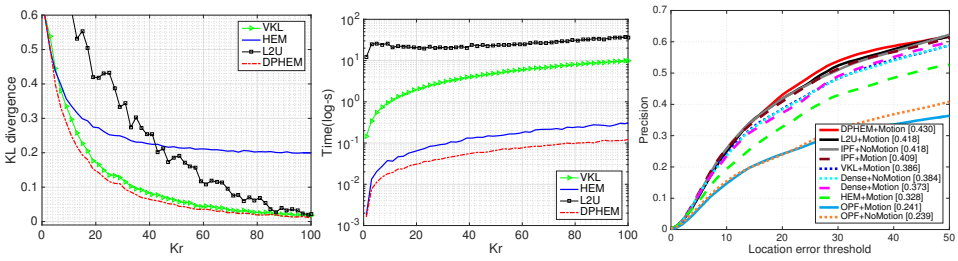
To show the applicability of DPHEM and the Bayesian filtering framework, we present 3 experiments: 1) simplifying synthetic 2D GMMs; 2) synthetic experiments on belief propagation; 3) visual tracking with Bayesian filtering and GMM posteriors. Comparisons are made with 3 other density simplifying algorithms: 1) the original HEM [21]; 2) variational KL minimization (VKL) [14]; 3) L2-norm upper-bound minimization (L2U) [25]. Experiments were implemented with Matlab on a desktop PC.

4.1 Synthetic 2D GMM simplification

In this experiment, a 2D base GMM with 2,500 components is randomly generated. For each target K_r, we use the same randomly picked components as initialization to run all four simplification methods. Ten random initializations are used, and the best reduced mixture model is selected using each method's corresponding objective criteria (i.e. variational expected log-likelihood for HEM and DPHEM, variational KL for VKL, approximate L2 for L2U). The similarity between the base and reduced mixtures is then evaluated using KL-divergence (KLD), which is calculated using Monte Carlo approximation with 100,000 samples. The experiment was repeated using 100 base GMMs, and the average KLD and processing time for $K_r \in [1, 100]$ are shown in Figure 2. DPHEM preserves the base model the best and has the lowest computing time.

4.2 Belief propagation

In this experiment, we run belief propagation (BP) on a 4-node undirected graph (Figure 3a), whose self-potentials are a 1D Gaussian mixture with 2 components and edge-potentials are Gaussian. During BP, the message is simplified using HEM, DPHEM, VKL, or L2U, when its number of components exceeds K_r, We also compare with Gibbs sampling [26]. Exact message passing (i.e., no simplification) for the first 3 iterations is used as the ground-truth.



(a) KLD (b) Processing time (c) Tracking precision

Figure 2: (a) Average KLD between the original and reduced models for different number of reduced components K_r. (b) Processing time vs. number of reduced components. (c) Precision plots for visual tracking. The number in the legend is precision at error threshold 20.

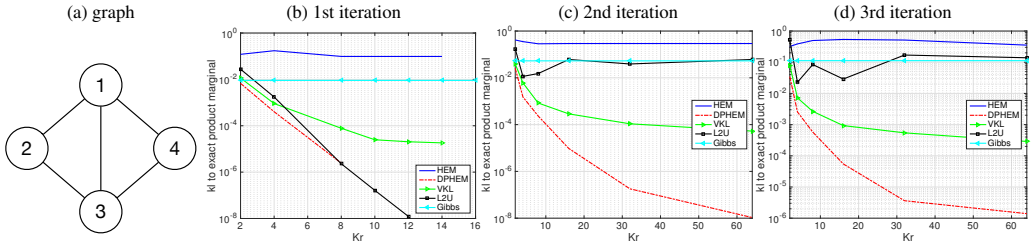


Figure 3: Belief propagation experiment: (a) the graph; (b-d) average KL divergence between different approximate marginals and the exact product marginal at each iteration of belief propagation. 1024 samples are used for Gibbs sampling for all iterations. K_r indicates the number of reduced components for other methods.

The average KLD between the marginals and ground-truth in each BP iteration are shown in Figure 3b-d. As BP iterates, DPHEM yields the best approximation that is most similar to the exact marginals. Examples of marginal distributions at each node after the 3rd iteration can be found in the supplemental.

4.3 Visual object tracking

We apply Bayesian filtering with GMM posteriors to visual object tracking, where the target object location is x_t and the video frame is y_t. A simple Gaussian motion model is used. The observation model is based on compressive tracking (CT) [27], and the SSG approximation of the likelihood function is obtained by applying the lower-bound algorithm (Section 3) on densely sampled locations (see Figure 1). DPHEM is used to reduce the GMM posterior to $K_r = 50$ components. Finally, the tracked position is the location $\hat{x}_t$ with largest posterior probability, and the corresponding image patch is used to update the observation model for the next frame.

We test our tracking method on a commonly-used benchmark [28] which contains 50 video sequences. Since the size of the bounding box in our method is fixed, the evaluation is based on the precision plot of OPE (One Pass Evaluation) proposed in [28], which is the percentage of successfully tracked frames whose centre location error is within a certain threshold. We compare our tracker with three particle-filter tracking methods with the same observation model: Dense – dense sampling of candidate locations x_t, where the tracked location is the one with maximum score $p(y_t|x_t)$ (also used in [27]); OPF – original particle filter where resampling is used to propagate particles to the next frame, and the tracked position is the particle with maximum weight; IPF – an improved particle filter where only the particle with maximum weight is propagated to the next frame. For each tracker, we test two versions with and without velocity in the motion model (+Motion, +NoMotion). Since DPHEM+Motion outperforms all other trackers, we also test and show the results by replacing DPHEM with other simplification methods, HEM, VKL and L2U.

Figure 2c presents the precision plots for the various tracking methods. DPHEM significantly outperforms the original particle filter with resampling (OPF), which suggests that using GMMs to represent posteriors has significant advantages over using weighted particles. Comparing simplification methods within the GMM tracking framework, DPHEM also outperforms L2U, VKL, and HEM. IPF has better precision than Dense, mainly because the Gaussian diffusion model prevents selecting outlier points far from the predicted position as the tracked position. However, both method performs slightly worse when using velocity in the motion model (+Motion).

References

[1] P. Pérez, C. Hue, J. Vermaak, and M. Gangnet, "Color-based probabilistic tracking," in *ECCV*, 2002.

[2] D. A. Ross, J. Lim, R.-S. Lin, and M.-H. Yang, "Incremental learning for robust visual tracking," *International Journal of Computer Vision*, vol. 77, no. 1-3, pp. 125–141, 2008.

[3] C. Bao, Y. Wu, H. Ling, and H. Ji, "Real time robust l1 tracker using accelerated proximal gradient approach," in *CVPR*, 2012.

[4] X. Jia, H. Lu, and M.-H. Yang, "Visual tracking via adaptive structural local sparse appearance model," in *CVPR*, 2012.

[5] T. Zhang, B. Ghanem, S. Liu, and N. Ahuja, "Robust visual tracking via multi-task sparse learning," in *CVPR*, 2012.

[6] D. Wang, H. Lu, and M.-H. Yang, "Least soft-threshold squares tracking," in *CVPR*, 2013.

[7] D. Wang and H. Lu, "Visual tracking via probability continuous outlier model," in *CVPR*, 2014.

[8] W. Zhong, H. Lu, and M.-H. Yang, "Robust object tracking via sparse collaborative appearance model," *IEEE Transactions on Image Processing*, vol. 23, no. 5, pp. 2356–2368, 2014.

[9] S. Hong and B. Han, "Visual tracking by sampling tree-structured graphical models," in *ECCV*, 2014, pp. 1–16.

[10] S. Oron, A. Bar-Hillel, D. Levi, and S. Avidan, "Locally orderless tracking," *International Journal of Computer Vision*, vol. 111, no. 2, pp. 213–228, 2015.

[11] A. R. Zamir and M. Shah, "Accurate image localization based on google maps street view," in *ECCV*, 2010, pp. 255–268.

[12] G. Vaca-Castano, A. R. Zamir, and M. Shah, "City scale geo-spatial trajectory estimation of a moving camera," in *CVPR*, 2012.

[13] M. J. Milford and G. F. Wyeth, "Seqslam: Visual route-based navigation for sunny summer days and stormy winter nights," in *IEEE International Conference on Robotics and Automation (ICRA)*, 2012, pp. 1643–1649.

[14] M. A. Brubaker, A. Geiger, and R. Urtasun, "Map-based probabilistic visual self-localization," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 4, pp. 652–665, 2016.

[15] F. Bardet, T. Chateau, and D. Ramadasan, "Illumination aware mcmc particle filter for long-term outdoor multi-object simultaneous tracking and classification," in *ICCV*, 2009, pp. 1623–1630.

[16] D. Varas and F. Marques, "Region-based particle filter for video object segmentation," in *CVPR*, 2014.

[17] Z. Chen, "Bayesian filtering: From kalman filters to particle filters, and beyond," *Statistics*, vol. 182, no. 1, pp. 1–69, 2003.

[18] D. M. Titterington, *Statistical analysis of finite mixture distributions*. John Wiley and Sons, Inc., New York, NY., 1985.

[19] G. McLachlan and D. Peel, *Finite mixture models*. John Wiley & Sons, 2004.

[20] D. W. Scott, *Multivariate density estimation: theory, practice, and visualization*. John Wiley & Sons, 2015.

[21] N. Vasconcelos and A. Lippman, "Learning mixture hierarchies," in *NIPS*, 1998, pp. 606–612.

[22] R. M. Neal and G. E. Hinton, "A view of the em algorithm that justifies incremental, sparse, and other variants," in *Learning in graphical models*, 1998, pp. 355–368.

[23] M. J. Wainwright and M. I. Jordan, "Graphical models, exponential families, and variational inference," *Foundations and Trends® in Machine Learning*, vol. 1, no. 1-2, pp. 1–305, 2008.

[24] I. Csisz, G. Tusnády *et al.*, "Information geometry and alternating minimization procedures," *Statistics and decisions*, 1984.

[25] K. Zhang and J. T. Kwok, "Simplifying mixture models through function approximation," *IEEE Transactions on Neural Networks*, vol. 21, no. 4, pp. 644–658, 2010.

[26] E. B. Sudderth, A. T. Ihler, M. Isard, W. T. Freeman, and A. S. Willsky, "Nonparametric belief propagation," *Communications of the ACM*, vol. 53, no. 10, pp. 95–103, 2010.

[27] K. Zhang, L. Zhang, and M.-H. Yang, "Real-time compressive tracking," in *ECCV*, 2012, pp. 864–877.

[28] Y. Wu, J. Lim, and M.-H. Yang, "Online Object Tracking: A Benchmark," in *CVPR*, 2013.

[29] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, and L. K. Saul, "An introduction to variational methods for graphical models," *Machine Learning*, vol. 37, no. 2, pp. 183–233, 1999.

[30] T. S. Jaakkola, "10 tutorial on variational approximation methods," *Advanced mean field methods: theory and practice*, p. 129, 2001.

Supplemental

1 Variational approximation for DPHEM

The expected log-likelihood is

$$\mathcal{J}(\Theta^{(r)}) = \mathbb{E}_{Y|\Theta^{(b)}}[\log p(Y|\Theta^{(r)})] = \int p(Y|\Theta^{(b)}) \log p(Y|\Theta^{(r)}) dY \tag{6}$$

$$= \sum_i \pi_i^{(b)} \int p(Y|\theta_i^{(b)}) \log p(Y|\Theta^{(r)}) dY = \sum_i \pi_i^{(b)} \mathbb{E}_{Y|\theta_i^{(b)}}[\log p(Y|\Theta^{(r)})]. \tag{7}$$

The variational lower bound of the log-likelihood of a mixture model is [29, 30]

$$\log p(Y|\Theta^{(r)}) \geq \max_{z_{ij}} \sum_j z_{ij} \log \frac{\pi_j^{(r)} p(Y|\theta_j^{(r)})}{z_{ij}}, \tag{8}$$

where z_{ij} are the variational parameters, with $\sum_{j=1}^{K} z_{ij} = \mathbf{1}$. Taking the expectation of (8) and applying Jensen's inequality, we have

$$\mathbb{E}_{Y|\theta_i^{(b)}}[\log p(Y|\Theta^{(r)})] \geq \max_{z_{ij}} \sum_j z_{ij} \left\{ \log \frac{\pi_j^{(r)}}{z_{ij}} + \mathbb{E}_{Y|\theta_i^{(b)}}[\log p(Y|\theta_j^{(r)})] \right\}. \tag{9}$$

The inner-expectation in (9) is obtained by noting that Y is a set of i.i.d. samples,

$$\mathbb{E}_{Y|\theta_i^{(b)}}[\log p(Y|\theta_j^{(r)})] = \mathbb{E}_{Y|\theta_i^{(b)}} \left[\sum_{m=1}^{N} \log p(y_m|\theta_j^{(r)}) \right] = \sum_{m=1}^{N} \mathbb{E}_{Y|\theta_i^{(b)}}[\log p(y_m|\theta_j^{(r)})] = N \mathbb{E}_{y|\theta_i^{(b)}}[\log p(y|\theta_j^{(r)})]. \tag{10}$$

Finally, substituting (10) and (9) into (7), we obtain the variational lower bound of the expected log-likelihood,

$$\mathcal{J}_{DP}(\Theta^{(r)}) = \max_{z_{ij}} \sum_i \sum_j \pi_i^{(b)} z_{ij} \left\{ \log \frac{\pi_j^{(r)}}{z_{ij}} + N \mathbb{E}_{y|\theta_i^{(b)}}[\log p(y|\theta_j^{(r)})] \right\} \leq \mathbb{E}_{Y|\Theta^{(b)}}[\log p(Y|\Theta^{(r)})]. \tag{11}$$

2 Fitting a quadratic lower-bound for likelihood approximation (diagonal case)

Here we show that when the precision matrix is diagonal, then the likelihood lower-bound approximation in (5) is a quadratic program (QP). Assume that $W = \text{diag}(w)$ is a diagonal precision matrix of a multivariate Gaussian distribution $x \sim \mathcal{N}(x|\mu, W^{-1})$, where $x, \mu, w \in \mathbb{R}^d$ and w is a vector of non-negative values. Then the log-Gaussian is

$$h(x) = -\sum_d w_d(x_d - \mu_d)^2 + \ell, \tag{12}$$

where the summation is over the dimensions of the vectors. The precision W can be found by maximizing the square-error with a set of state-likelihood pairs $\mathcal{D} = \{(x_i, p_i)|p_i = p(y_i|x_i)\}_{i=1}^N$ in the log-likelihood space with the constraint that $h(x)$ is a lower bound of the points.

$$W^* = \underset{W}{\text{argmin}} \ \frac{1}{2}\sum_{i=1}^N (\ell_i - h(x_i))^2 \quad \text{s.t. } \ell_i - h(x_i) \geq 0, \ \forall i, \tag{13}$$

Defining $\tilde{x} = (x - \mu)^2$ as the vector of element-wise square differences, and $\tilde{\ell}_i = \ell_i - \ell$, the problem in (13) becomes

$$w^* = \underset{w}{\text{argmin}} \ \frac{1}{2}\sum_{i=1}^N (\tilde{\ell}_i + w^T\tilde{x}_i)^2 \quad \text{s.t. } \tilde{\ell}_i + w^T\tilde{x}_i \geq 0, \ \forall i, \ w \geq 0. \tag{14}$$

Expanding the square term and re-arranging,

$$w^* = \underset{w}{\text{argmin}} \ \frac{1}{2}\sum_{i=1}^N (\tilde{\ell}_i^2 + 2w^T\tilde{x}_i\tilde{\ell}_i + w^T\tilde{x}_i\tilde{x}_i^T w) \quad \text{s.t.} \quad -\tilde{x}_i^T w \leq \tilde{\ell}_i, \forall i, \ w \geq 0. \tag{15}$$

Hence, we have the standard QP form,

$$w^* = \underset{w}{\text{argmin}} \ \frac{1}{2}w^T H w + w^T f \quad \text{s.t. } Aw \leq b, \ w \geq 0. \tag{16}$$

where

$$H = \sum_{i=1}^N \tilde{x}_i\tilde{x}_i^T, \quad f = \sum_{i=1}^N \tilde{\ell}_i\tilde{x}_i, \quad A = -\left[\tilde{x}_1, \cdots, \tilde{x}_N\right]^T, \quad b = \left[\tilde{\ell}_1, \cdots, \tilde{\ell}_N\right]^T. \tag{17}$$

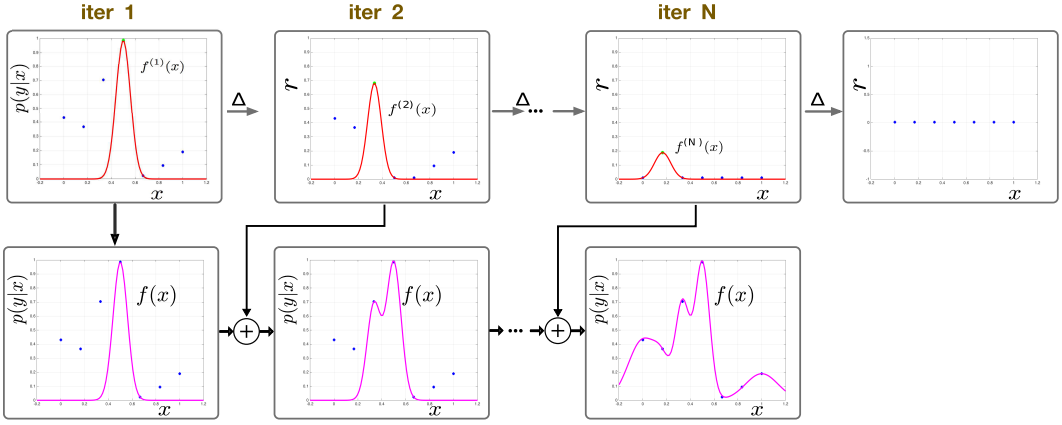


Figure 4: A 1D example of likelihood approximation using sum of scaled Gaussians. Δ indicates calculation of the residuals: $r_i = p(y_i|x_i) - f^{(k)}(x_i)$.

3 Experiment results

We show example figures from the experiments in this section. Figure 5 shows an example of 2D reduced GMMs from Section 4.1. Figure 6 shows two examples of computed marginals after 3 iterations of BP from Section 4.2.

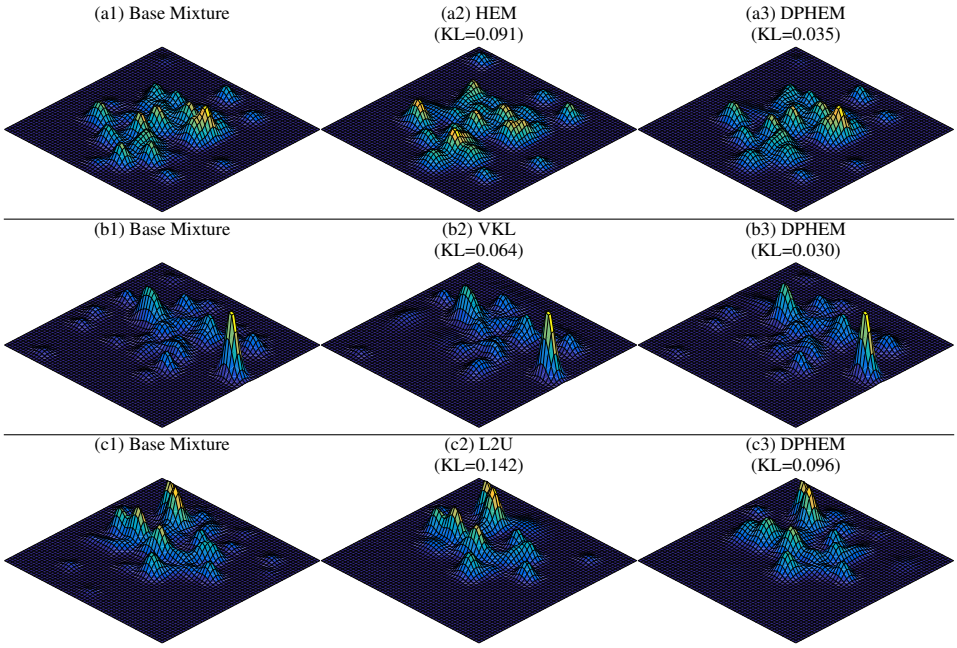


Figure 5: Example of reduced mixture models using different approaches. Each row compares DPHEM with a baseline method, and shows a typical difference. The base mixture model contains $K_b = 2500$ components. The KL divergence is shown between the base mixture and the reduced mixture.

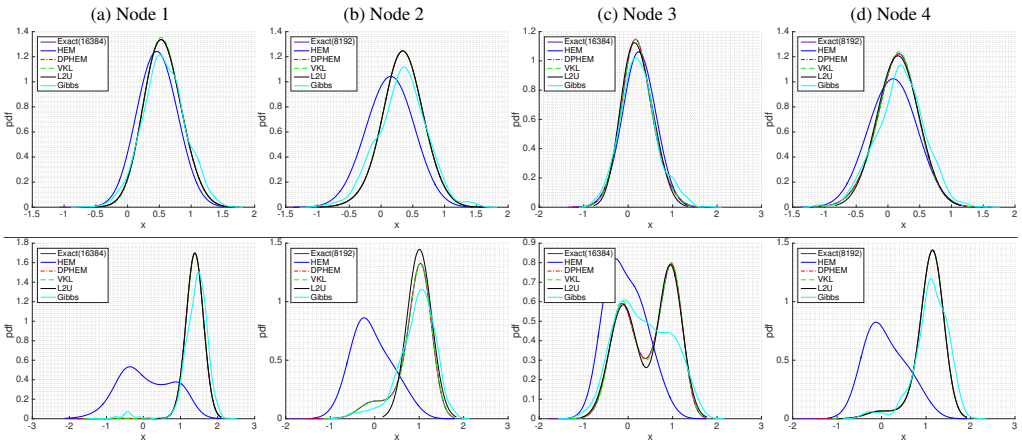


Figure 6: Two examples of the marginal distributions (belief) at each node after the 3rd iteration of belief propagation. The number of components in the exact marginal density at each node is shown in the legend. 1024 samples for Gibbs and $K_r = 64$ for other methods.