HW #6 Solns.
EE 556

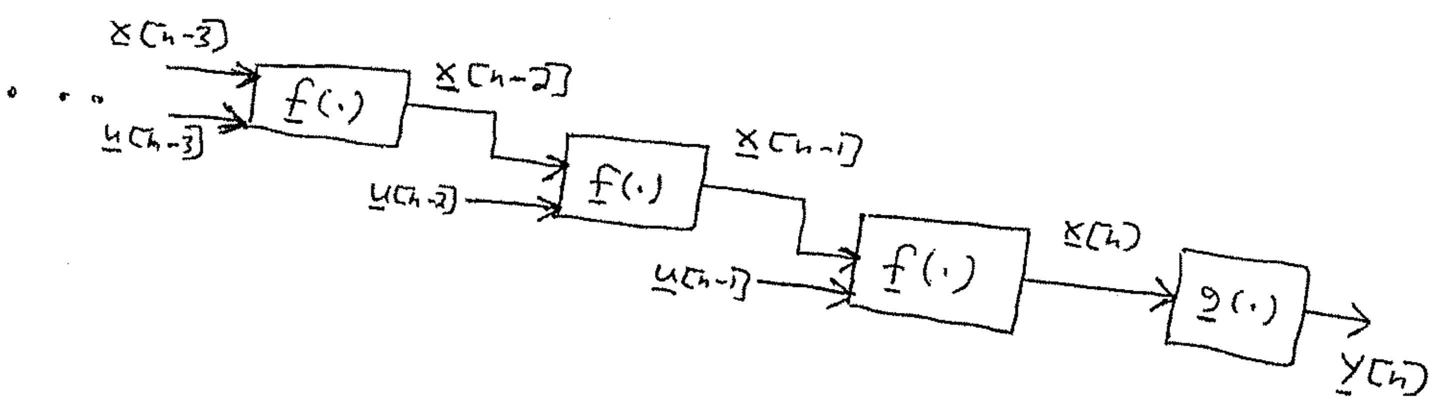
#1) consider the state-space model:

(15.12)
$$\underline{x}[n+1] = \underline{f}(\underline{x}[n], \underline{u}[n]) \qquad (1)$$
$$\underline{y}[n+1] = \underline{g}(\underline{x}[n+1])$$

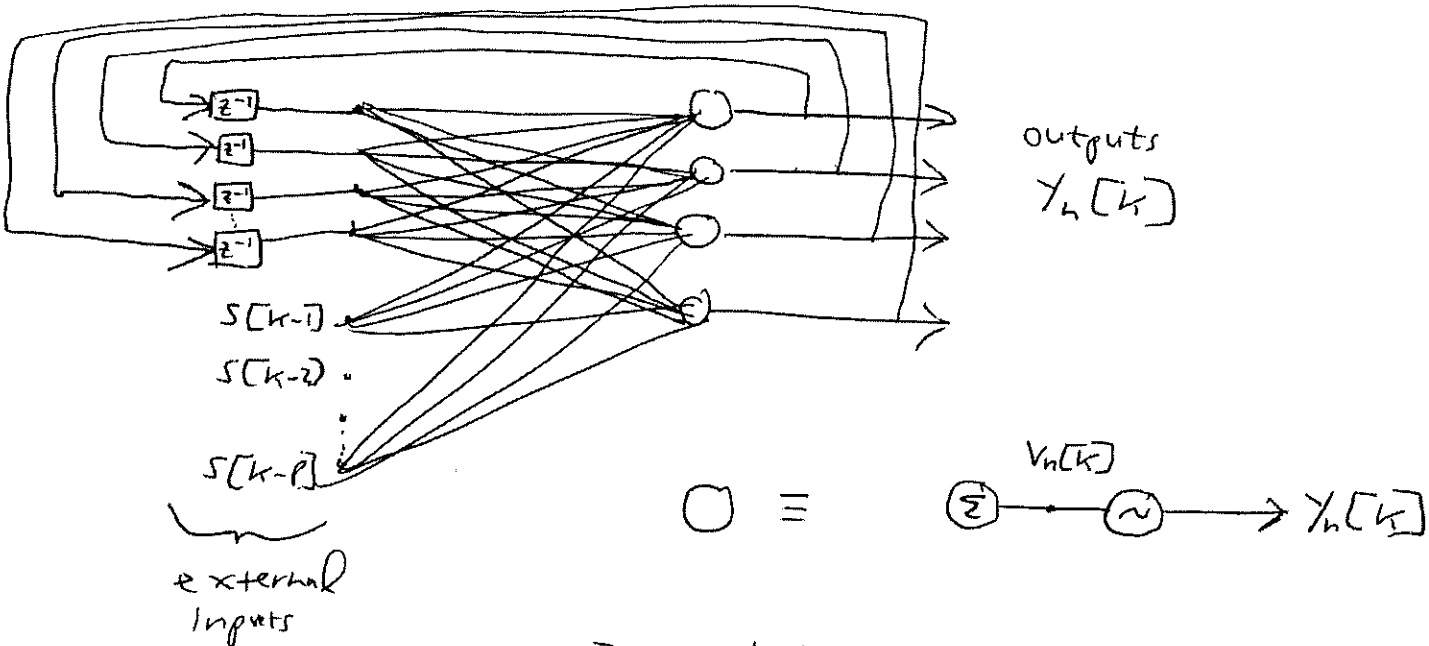
Using (1), we can "walk" backwards in time:

$$\underline{x}[n] = \underline{f}(\underline{x}[n-1], \underline{u}[n-1])$$
$$\underline{x}[n-1] = \underline{f}(\underline{x}[n-2], \underline{u}[n-2])$$
$$\underline{x}[n-2] = \underline{f}(\underline{x}[n-3], \underline{u}[n-3])$$
$$\vdots$$

Accordingly, the recurrent neural net in Fig. 15.3 is unfolded as:



#2) Consider the following recurrent structure:



$$y_n[k] = \phi(v_n(k)), \quad n = 1, 2, \ldots N$$

$$v_n[k] = \sum_{l=1}^{P+N} w_{n,l}[k]\, u_l[k],$$

where $\underline{u_n}^T[k] = [s[k-1], s[k-2], \ldots s[k-p], y_1[k-1], y_2[k-1], \ldots y_N[k-1]]$

Let $e^2[k] = (s[k] - y_1[k])^2$, and think of $y_n[k]$, $n > 1$ as "state variables".

$$\Delta w_{n,l}[k] = -\mu \frac{\partial e^2[k]}{\partial w_{n,l}[k]}$$

$$= -\tilde{\mu}\, e[k] \frac{\partial e[k]}{\partial w_{n,l}[k]}$$

$$\frac{\partial e[k]}{\partial w_{n,l}[k]} = -\frac{\partial y_1[k]}{\partial w_{n,l}[k]} = -\phi'(v_1[k]) \frac{\partial v_1[k]}{\partial w_{n,l}[k]}$$

Continuing,

$$\frac{\partial e[k]}{\partial W_{n,l}[k]} = -\phi'(v_1[k])\cdot\left(\underbrace{\sum_{m=1}^{N} \frac{\partial y_m(k-1]}{\partial W_{n,l}(k]}}\cdot W_{1,p+m}[k]\right)$$

Note: $\frac{\partial s[k-j]}{\partial W_{n,l}[k]} = 0$

Again, we see we need a noncausal term $\Longrightarrow$ approximate by:

$$\frac{\partial y_m[k-1]}{\partial W_{nl}[k]} \approx \frac{\partial y_m[k-1]}{\partial W_{nl}[k-1]}$$

#3) Given: $f_{x_1,x_2}(x_1, x_2 / M=k) = f_{x_1}(x_1 / M=k) \cdot f_{x_2}(x_2 / M=k)$

Does this imply $f_{x_1,x_2}(x_1,x_2) = f_{x_1}(x_1) \cdot f_{x_2}(x_2)$?

A: No!

Note: $$f_{x_1,x_2}(x_1,x_2) = \sum_{k=1}^{\#\text{ components}} \alpha_k f_{x_1,x_2}(x_1, x_2/M=k)$$

$$= \sum_{k=1}^{\#\text{ comps.}} \alpha_k f_{x_1}(x_1/M=k) \cdot f_{x_2}(x_2/M=k)$$

But $$f_{x_1}(x_1) \cdot f_{x_2}(x_2) = \left(\sum_{k=1}^{\#\text{ comps}} \alpha_k f_{x_1}(x_1/M=k)\right) \cdot \left(\sum_{k=1}^{\#\text{ comps}} \alpha_k f_{x_2}(x_2/M=k)\right)$$

$$\neq f_{x_1,x_2}(x_1,x_2)$$

#4) In this case, the log-likelihood function is:

$$\log \mathcal{L} = \sum_{i=1}^{T} \log\left(\sum_{j=1}^{M} \alpha_j \prod_{l=1}^{d} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mu_{jl} - x_{il})^2}{2\sigma^2}\right)\right)$$

(T = # data, d = # dimensions)

Let's invoke the "complete data" strategy discussed in lecture.

Introduce the variables $V_{ij} = \begin{cases} 1, & \underline{x_i} \in \text{component } j \\ 0, & \text{else} \end{cases}$

We can then, supposing the V_{ij} are known, write: the complete data log-likelihood:

$$\log \mathcal{L}_c = \sum_{i=1}^{T} \log \sum_{j=1}^{M} V_{ij} \alpha_j \prod_{l=1}^{d} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mu_{jl} - x_{il})^2}{2\sigma^2}\right)$$

$$= \sum_{i=1}^{T} \sum_{j=1}^{M} V_{ij} \log\left(\alpha_j \prod_{l=1}^{d} \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(\mu_{jl} - x_{il})^2}{2\sigma^2}\right)\right)$$

Then,

$$E[\log \mathcal{L}_c] = \sum_{i=1}^{T} \sum_{j=1}^{M} E[v_{ij} / \mathcal{X}; \theta] \cdot \log\left(\alpha_j \prod_{l=1}^{d} \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(\frac{-(\mu_{jl} - x_{il})^2}{2\sigma^2}\right)\right)$$

Now,

$$E[v_{ij} / \mathcal{X}; \theta] = 1 \cdot \text{Prob}[M = j / \underline{x}_i; \theta] + 0 \cdot \text{Prob}[M \neq j / \underline{x}_i, \theta]$$

$$= \text{Prob}[M = j / \underline{x}_i; \theta]$$

Based on the current parameters $\theta = \{\{\alpha_j\}, \{\underline{\mu}_j\}, \sigma^2\}$, this posterior probability is:

$$\text{Prob}[M = j / \underline{x}_i; \theta] = \frac{f(M = j, \underline{x}_i; \theta)}{f(\underline{x}_i; \theta)}$$

$$= \frac{f_{\underline{x}}(\underline{x}_i / M = j; \theta) \cdot \alpha_j}{\sum_{j'=1}^{M} f_{\underline{x}}(\underline{x}_i / M = j'; \theta) \cdot \alpha_{j'}},$$

where $f_{\underline{x}}(\underline{x}_i / M = j; \theta) = \prod_{l=1}^{d} \frac{1}{\sqrt{2\pi}\,\sigma} \exp\left(-\frac{(-\mu_{jl} - x_{il})^2}{2\sigma^2}\right)$

computing these posterior probabilities for all data points is the <u>E - step</u>.

In the M-step, given E-step quantities held fixed, we maximize $E[\log \mathcal{L}_c]$ w.r.t. Θ while satisfying all constraints on parameters.

Define the Lagrangian

$$L = E[\log \mathcal{L}_c] + \lambda\left(\sum_{j=1}^{m} \alpha_j - 1\right)$$

$$= \sum_{i=1}^{T} \sum_{j=1}^{m} \mathrm{Prob}[M_i = j / \underline{x}_i;\, \Theta^{(t)}] \cdot \left(\log \alpha_j + \sum_{l=1}^{d}\left[-\frac{(\mu_{jl} - x_{il})^2}{2\sigma^2} - \frac{1}{2}\log\sigma^2 \right] + \text{constant term} \right)$$

$$+ \lambda\left(\sum_{j=1}^{m} \alpha_j - 1\right)$$

To find $\alpha_j^{(t+1)}$, $j = 1, \ldots m$, set

$$\frac{\partial L}{\partial \alpha_j} = 0 \implies \sum_{i=1}^{T} \frac{\mathrm{Prob}[M_i = j / \underline{x}_i;\, \Theta^{(t)}]}{\alpha_j} + \lambda = 0$$

Choose λ to satisfy constraint $\sum_{j=1}^{m} \alpha_j = 1 \implies$ $\lambda = -T$, i.e.

$$\alpha_j^{(t+1)} = \frac{1}{T} \sum_{i=1}^{T} \mathrm{Prob}[M_i = j / \underline{x}_i;\, \Theta^{(t)}], \quad j = 1, \ldots m$$

To find μ_{jl}:

$$\frac{\partial L}{\partial \mu_{jl}} = 0 \Rightarrow \mu_{jl}^{(t+1)} = \frac{\sum_{i=1}^{T} x_{il} \, Prob[M_i = j / \underline{x}_i; \theta^{(t)}]}{\sum_{i=1}^{T} Prob[M_i = j / \underline{x}_i; \theta^{(t)}]},$$

$l = 1, .. d,$
$j = 1, ... m$

To find σ^2:

$$\left.\frac{\partial L}{\partial \sigma^2}\right|_{\{\mu_{jl} = \mu_{jl}^{(t+1)}\}} = 0 \Rightarrow \sum_{i=1}^{T} \sum_{j=1}^{M} Prob[M_i = j / \underline{x}_i; \theta^{(t)}] \cdot \left(\sum_{l=1}^{d} + \frac{(\mu_{jl}^{(t+1)} - x_{il})^2}{2\sigma^4} - \frac{1}{2\sigma^2} \right) = 0$$

Multiplying through by $\sigma^4 \Rightarrow$

$$\sigma^{2\,(t+1)} = \frac{\sum_{i=1}^{T} \sum_{j=1}^{M} \sum_{l=1}^{d} (\mu_{jl}^{(t+1)} - x_{il})^2 \cdot Prob(M_i = j / \underline{x}_i; \theta^{(t)}]}{T \cdot d}$$

<u>These updates form the M-step</u>.

successive E + M steps are nondecreasing in $\log \mathcal{L}$ $\Rightarrow$ EM converges to a locally optimal solution ...

#5) The K-means algorithm performs centroid and nearest neighbor updates of each cluster, either for a specified number of iterations or until convergence.

Spse. T_{max} iterations are performed.

Let N denote # of data points, K the # of clusters, d the feature dimensionality.

The centroid rule is:

$$\underline{Y}_j = \frac{\sum_{i=1}^{N} V_{ij} \underline{X}_i}{\sum_{i=1}^{N} V_{ij}}, \quad j=1,..K$$

This requires $K \cdot ((N-1)d + N-1)$ additions

The nearest neighbor rule is:

$$V_{ij} = 1 \text{ iff } \|\underline{X}_i - \underline{Y}_j\|^2 \leq \|\underline{X}_i - \underline{Y}_k\|^2, \forall k$$

each vector sqd. distance computation requires ~ $2d$ scalar multiplies + $2d$ scalar additions

$\Rightarrow \sim 2N \cdot Kd$ multiplies + adds for nearest neighbor assignment of all the data.

The overall complexity is thus $O(T_{max} NKd)$.

$$R = \text{rate} = \left\lceil \frac{\log_2 K}{d} \right\rceil \simeq \frac{\log_2 K}{d} \quad (*)$$

This definition is motivated by vector quantization (VQ) compression application of the clustering solution, where it indicates the number of bits per sample needed to specify a cluster index to the decoder (source encoder implements the nearest neighbor rule, source decoder reconstructs cluster approximation to the source vector, $\underline{x}$ -- we discussed this application in lecture this week, in the context of "noisy channel VQ"...)

Eq. $(*)$ $\Rightarrow$ $K \cong 2^{dR}$ $\Rightarrow$ to keep the rate constant while increasing d, K grows <u>exponentially</u>. This means both the storage complexity and the complexity of nearest neighbor encoder search grows exponentially with d for fixed R.

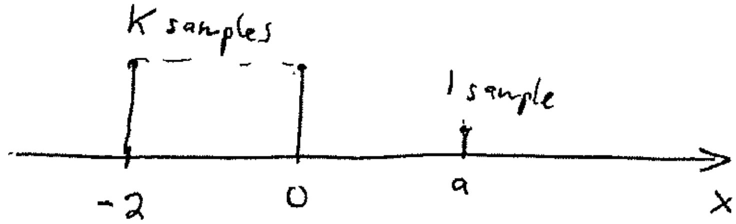
For image coding applications, to achieve a system with acceptable image reconstruction quality and reasonable (not too high) bit rate, we may, e.g., have $d = 64$ (8×8 block) and $R = 0.25$ bits per pixel.

This means: $K = 2^{16} = 64K$ clusters/codevectors.

The complexity of "nearest neighbor" search is very high (!) in this case...

#6) Clearly, if $N \geq K$,
one can reduce the clustering distortion by taking an "unassigned" cluster and placing it coincident with a data point currently assigned to a cluster that owns more than one data point.
The "unassigned" cluster will own the data point it coincides with, which will incur zero distortion, whereas previously there was nonzero distortion for this data point. Thus, this use of the "unassigned cluster" must decrease the distortion, D.

#7)



i) suppose $a^2 < 2(K+1)$

There are 2 candidate solutions:

$\underline{\text{I}}$: $P_1 = \{-2, 0\}$, $P_2 = \{a\}$

$\underline{\text{II}}$: $P_1 = \{-2\}$, $P_2 = \{0, a\}$

$$D_{I} = 2K \quad (y_1 = -1, \; y_2 = a)$$

$$D_{II} = K\left(\frac{a}{K+1}\right)^2 + \left(a - \frac{a}{K+1}\right)^2 \quad \left(y_1 = -2, \; y_2 = \frac{a}{K+1}\right).$$

$$D_{II} < D_{I} \iff \frac{K a^2}{(K+1)^2} + a^2\left(\frac{K}{K+1}\right)^2 < 2K$$

$$\iff \frac{a^2}{(K+1)^2}(K + K^2) < 2K$$

$$\iff \frac{a^2 K}{K+1} < 2K \iff a^2 < 2(K+1),$$

i.e. optimal solution is # II.

ii) Clearly, from part i), solution I is optimal in this case...

#8) $$\min_{\{P_{j/i}\},\{\underline{y}_j\}} \underbrace{\sum_i \sum_j P_{j/i} \log\left(P_{j/i}/q_j\right)}_{\to C}$$

assume $\|\cdot\|^2$

$$\text{s.t.} \quad \langle D \rangle = \sum_i \sum_j P_{j/i}\, d(\underline{x}_i, \underline{y}_j)$$

$$\text{and} \quad \sum_j P_{j/i} = 1 \quad \forall i$$

$$L = \beta \langle D \rangle + C + \sum_i \lambda_i \left(\sum_j P_{j/i} - 1 \right)$$

We Need: $$\frac{\partial L}{\partial P_{j/i}} = 0 \quad \forall i, j$$

$$\nabla_{\underline{y}_j} L = \underline{0} \quad \forall j$$

$$\frac{\partial L}{\partial P_{j/i}} = 0 \implies \beta d(\underline{x}_i, \underline{y}_j) + \log\left(\frac{P_{j/i}}{q_j}\right) + 1 + \lambda_i = 0$$

$$\Downarrow$$

$$P_{j/i} = \frac{q_j e^{-\beta d(\underline{x}_i, \underline{y}_j)}}{\sum_k q_k e^{-\beta d(\underline{x}_i, \underline{y}_k)}}$$

$$\nabla_{\underline{y}_j} L = \underline{0} \implies \underline{y}_j = \frac{\sum_i \underline{x}_i P_{j/i}}{\sum_i P_{j/i}}$$

The new form for $P_{j/i}$ is known, within rate distortion theory (within information theory)) as the "tilted distribution" -- it accounts for the prior knowledge, $\{q_j\}$.

Key aspects of solution to computer assignment:

1) To evaluate unsupervised classification accuracy:

i) Assign each data point to MAP-nearest component:

$$j^*(i) = \arg\max_j P[M=j/\underline{x}_i] = \arg\max_j \frac{\alpha_j f_{\underline{X}/j}(\underline{x}_i/\theta_j)}{\sum_k \alpha_k f_{\underline{X}/k}(\underline{x}_i/\theta_k)}$$

ii) For each cluster/component, find majority (plurality) class:

$$K^*(j) = \arg\max_K \sum_{\substack{i=1:\\ j^*(i)=j,\\ c_i=K}}^{T} 1$$

iii) $$P[\text{error}] = \frac{\sum_j \sum_{\substack{i:\, j^*(i)=j,\\ c_i \neq K^*(j)}} 1}{T}$$

You should find error rates in the range 4-6% (the supervised Bayes classification error rate for the Iris domain is somewhere around 3%).

2) You might observe that some different random initializations lead to different (locally optimal) MLE solutions

3) Log-likelihood must be strictly increasing with EM iterations -- otherwise, your implementation is faulty.

4) If parameters are initialized s.t. every component is the same, EM iterations will not change the solution -- this solution is a fixed point, but NOT locally optimal -- components' parameters must be initialized at least a little differently, to initiate "symmetry-breaking".

5) Hopefully, BIC gave you a fairly convincing minimum at 3 (or possibly 4 or 5) mixture components. (BIC/MDL).

$$\mathrm{BIC}/\mathrm{MDL}(K) =$$

$$\frac{1}{2} C(K) \cdot \log(T) - \log P(\mathcal{X} / \Theta(K)),$$

where $C(K)$ = # free parameters in mixture model with K components =

$$\underset{\substack{\downarrow \\ \#\ \text{comps.}}}{K} \cdot \left(\underbrace{(K-1)}_{\substack{\#\ \text{free mass} \\ \text{parameters} \\ (\alpha)}} + \underbrace{d}_{\#\ \text{mean params.}} + \underbrace{\frac{d(d-1)}{2}}_{\substack{\#\ \text{covariance} \\ \text{matrix params.}}} \right).$$

in general case, if you use full covariance matrix, rather than a diagonal covariance matrix...

Computer assignment:

Example solution:

Error rate = 0.033

5 components

0 = setosa   1 = versicolor   2 = virginica

```
likelihood = -246.114636
mdl = 358.853930
0 2 2 1 1
Error rate = 0.033333
New solution:
0.333333 5.006000 3.418000 1.464000 0.244000 0.121764 0.142276 0.029504 0.011264
0.158604 6.150444 2.815574 5.108080 1.825220 0.089022 0.055468 0.065305 0.049498
0.193202 6.961597 3.116335 5.870547 2.153397 0.245651 0.087761 0.214634 0.054010
0.146240 5.534954 2.571003 3.846288 1.164787 0.099057 0.061967 0.119454 0.018193
0.168621 6.195891 2.906167 4.529823 1.432012 0.233260 0.075775 0.036997 0.013739
```

0 2 2 1 1 → class label for each component

priors   mean vectors   Variances in each dimension

- If all parameters are made common across all components, then the iterations will not "undo" this -- this solution is a "symmetry point", a stationary point but not a maximum -- it's a saddle point solution...
- Each iteration must increase the log-likelihood...
- Different random inits. may lead to different solutions (some quite good, some quite poor)
- poor solutions ⟹ a class might not "own" any components
- I found that best MDL scores typically occurred for model sizes 4, 5, + 6. Since lowest error rate I observed (0.033) also occurred for model size of ≥ 5, this suggests that the "true" model size may be at least 4...

FPAs :

$$\mu_{jl}^{(t+1)} = \frac{\sum_{i=1}^{T} x_{il} \, \mathrm{Prob}[\text{component } j / X_i]^{(t)}}{\sum_{i=1}^{T} \mathrm{Prob}[\text{component } j / X_i]^{(t)}}, \qquad \begin{array}{l} l = 1, \dots d \\ j = 1, \dots N_{comp} \end{array}$$

$$\sigma_{jl}^{2\,(t+1)} = \frac{\sum_{i=1}^{T} (x_{il} - \mu_{jl}^{(t+1)})^2 \, \mathrm{Prob}[\text{component } j / X_i]^{(t)}}{\sum_{i=1}^{T} \mathrm{Prob}[\text{component } j / X_i]^{(t)}}$$

$$\alpha_j^{(t+1)} = \frac{1}{T} \sum_{i=1}^{T} \mathrm{Prob}[\text{component } j / X_i]^{(t)},$$

where

$$\mathrm{Prob}[\text{comp. } j / X_i]^{(t)} = \frac{\alpha_j^{(t)} \cdot \dfrac{e^{-\left(\sum_{l=1}^{d} \frac{(x_{il} - \mu_{jl}^{(t)})^2}{2\sigma_{jl}^{(t)2}}\right)}}{(2\pi)^{d/2} \cdot \left(\prod_{l=1}^{d} \sigma_{jl}^{(t)2}\right)^{1/2}}}{\sum_{k=1}^{N_{comp}} \alpha_k^{(t)} \cdot \dfrac{e^{-\left(\sum_{l=1}^{d} \frac{(x_{il} - \mu_{kl}^{(t)})^2}{2\sigma_{kl}^{(t)2}}\right)}}{(2\pi)^{d/2} \left(\prod_{l=1}^{d} \sigma_{kl}^{(t)2}\right)^{1/2}}}$$