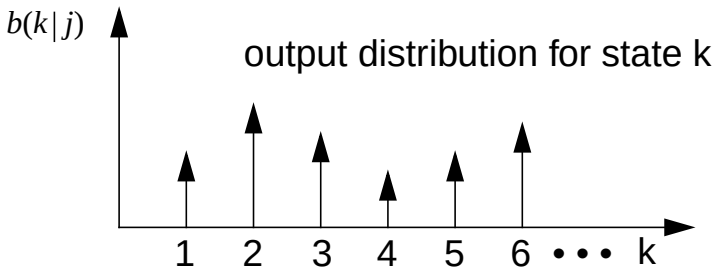
Continuous Density HMMs

The discrete HMM incorporates a discrete probability density function, captured in the matrix $\boldsymbol{B}$, to describe the probability of outputting a symbol:



Signal measurements, or feature vectors, are continuous-valued N-dimensional vectors. In order to use our discrete HMM technology, we must vector quantize (VQ) this data — reduce the continuous-valued vectors to discrete values chosen from a set of M codebook vectors. Initially, most HMMs were based on VQ front-ends. However, recently, the continuous density model has become widely accepted.
Let us assume a parametric model of the observation pdf:

$$M = \left\{ S, \pi(1), \boldsymbol{A}, \left\{ f_{\underline{y}|\underline{x}}(\xi|i), 1 \le i \le S \right\} \right\}$$

The likelihood of generating observation $\boldsymbol{y}(t)$ in state j is defined as:

$$b(\boldsymbol{y}(t)|j) \equiv f_{\underline{y}|\underline{x}}(\boldsymbol{y}(t)|j)$$

Note that taking the negative logarithm of $b(\ )$ will produce a log-likelihood, or a Mahalanobis-like distance. But what form should we choose for $f(\ )$? Let's assume a Gaussian model, of course:

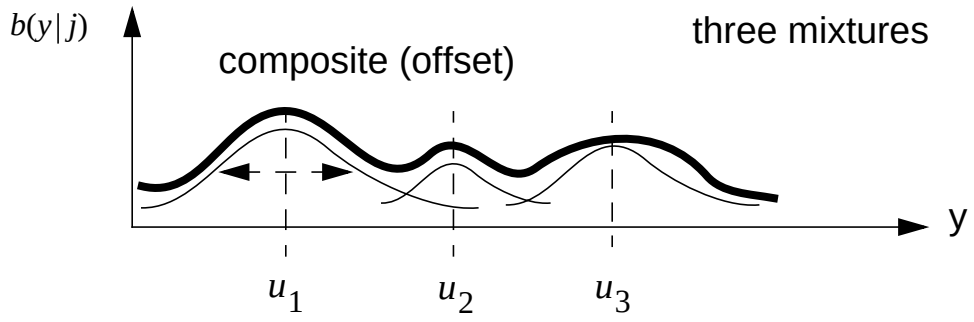
$$f_{\underline{y}|\underline{x}}(\boldsymbol{y}|i) = \frac{1}{\sqrt{2\pi|\boldsymbol{C}_i|}} \exp\left\{ -\frac{1}{2}(\boldsymbol{y}-\mu_i)^T \boldsymbol{C}_i^{-1}(\boldsymbol{y}-\mu_i) \right\}$$

Note that this amounts to assigning a mean and covariance matrix to each state — a significant increase in complexity. However, shortcuts such as variance-weighting can help reduce complexity.

Also, note that the log of the output probability at each state becomes precisely the Mahalanobis distance (principal components) we studied at the beginning of the course.

Mixture Distributions

Of course, the output distribution need not be Gaussian, or can be multimodal to reflect the fact that several contexts are being encoded into a single state (male/female, allophonic variations of a phoneme, etc.). Much like a VQ approach can model any discrete distribution, we can use a weighted linear combination of Gaussians, or a mixture distribution, to achieve a more complex statistical model.



Mathematically, this is expressed as:

$$f_{\underline{y}|\underline{x}}(\boldsymbol{y}|i) = \sum_{m=1}^{M} c_{im}\,\aleph(\boldsymbol{y};\mu_{im},\boldsymbol{C}_{im})$$

In order for this to be a valid pdf, the mixture coefficients must be nonnegative and satisfy the constraint:

$$\sum_{m=1}^{M} c_{im} = 1, \qquad 1 \le i \le S$$

Note that mixture distributions add significant complexity to the system: m means and covariances at each state.

Analogous reestimation formulae can be derived by defining the intermediate quantity:

$$\nu(i;t,l) \equiv P(\underline{x}(t) = i \,|\, \boldsymbol{y}(t) \text{produced in accordance with mixture } l)$$

$$= \frac{\alpha(\boldsymbol{y}_1^t,i)\beta(\boldsymbol{y}_{t+1}^T \,|\, i)}{\sum_{j=1}^{S} \alpha(\boldsymbol{y}_1^t,j)\beta(\boldsymbol{y}_{t+1}^T \,|\, j)} \times \frac{c_{il}\,\aleph(\boldsymbol{y}_{tl}^t;\mu_{il},\boldsymbol{C}_{il})}{\sum_{m=1}^{M} c_{im}\,\aleph(\boldsymbol{y}_t^t;\mu_{im},\boldsymbol{C}_{im})}$$

The mixture coefficients can now be reestimated using:

$$\bar{c}_{il} = \frac{\nu(i;\bullet,l)}{\sum_{m=1}^{M} \nu(i;\bullet,m)}$$

the mean vectors can be reestimated as:

$$\bar{\mu}_{il} = \frac{\sum_{t=1}^{T} \nu(i;t,l)\boldsymbol{y}(t)}{\nu(i;\bullet,l)}$$

the covariance matrices can be reestimated as:

$$\bar{\boldsymbol{C}}_{il} = \frac{\sum_{t=1}^{T} \nu(i;t,l)[\boldsymbol{y}(t)-\mu_{il}][\boldsymbol{y}(t)-\mu_{il}]^T}{\nu(i;\bullet,l)}$$

and the transition probabilities, and initial probabilities are reestimated as usual.

The Viterbi procedure once again has a simpler interpretation:

$$\mu_{il} = \frac{1}{N_{il}} \sum_{\substack{t=1 \\ y(t)\sim il}}^{T} \boldsymbol{y}(t)$$

and

$$\boldsymbol{C}_{il} = \frac{1}{N_{il}} \sum_{\substack{t=1 \\ y(t)\sim il}}^{T} [\boldsymbol{y}(t)-\mu_{il}][\boldsymbol{y}(t)-\mu_{il}]^T$$

The mixture coefficient is reestimated as the number of vectors associated with a given mixture at a given state:

$$c_{il} = \frac{N_{il}}{N_i}$$

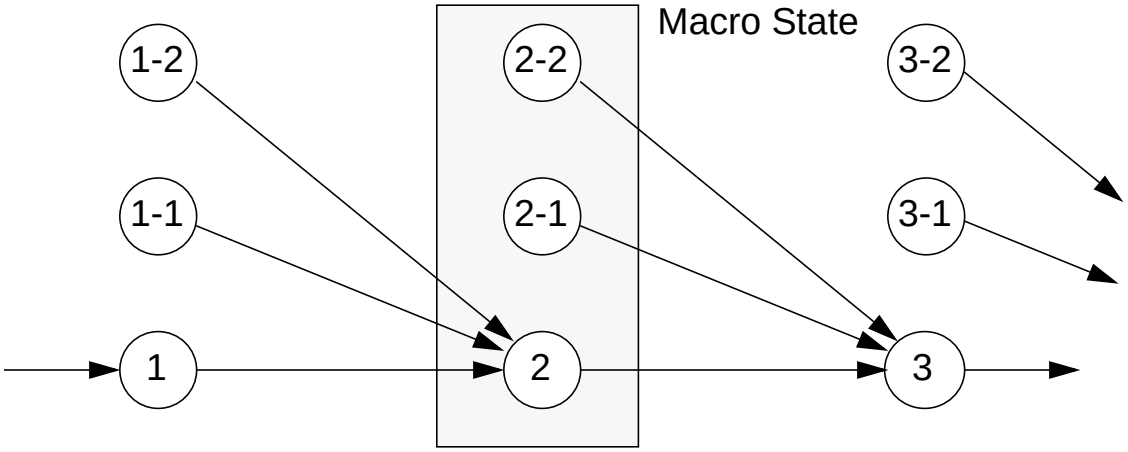
State Duration Probabilities

Recall that the probability of staying in a state was given by an exponentially-decaying distribution:

$$P(\overline{O}|Model, q_1 = i) = P(\overline{O}, q_1 = i|Model)/P(q_1 = i) = a_{ii}^{\,d-1}(1 - a_{ii})$$

This model is not necessarily appropriate for speech. There are three approaches in use today:

• Finite-State Models (encoded in acoustic model topology)



(Note that this model doesn't have skip states; with skip states, it becomes much more complex.)

• Discrete State Duration Models (D parameters per state)

$$P(\underline{d}_i = d) = \tau_d \qquad 1 \le d \le D$$

• Parametric State Duration Models (one to two parameters)

$$f(d_i) = \frac{1}{\sqrt{2\sigma_i^{\,2}}} \exp\left\{\frac{-\sqrt{2}|d|}{\sigma_i}\right\}$$

Reestimation equations exist for all three cases. Duration models are often important for larger models, such as words, where duration variations can be significant, but not as important for smaller units, such as context-dependent phones, where duration variations are much better understood and predicted.

Scaling in HMMs

As difficult as it may seem to believe, standard HMM calculations exceed the precision of 32-bit floating point numbers for the simplest of models. The large numbers of multiplications of numbers less than one leads to underflow. Hence, we must incorporate some form of scaling.

It is possible to scale the forward-backward calculation (see Section 12.2.5) by normalizing the calculations by:

$$c(t) = \frac{1}{\sum_{i=1}^{S} \tilde{\alpha}(y_1^t, i)}$$

at each time-step (time-synchronous normalization).

However, a simpler and more effective way to scale is to deal with log probabilities, which work out nicely in the case of continuous distributions. Even so, we need to somehow prevent the best path score from growing with time (increasingly negative in the case of log probs). Fortunately, at each time step, we can normalize all candidate best-path scores, and proceed with the dynamic programming search. More on this later...

Similarly, it is often desirable to trade-off the important of transition probabilities and observation probabilities. Hence, the log-likelihood of an output symbol being observed at a state can be written as:

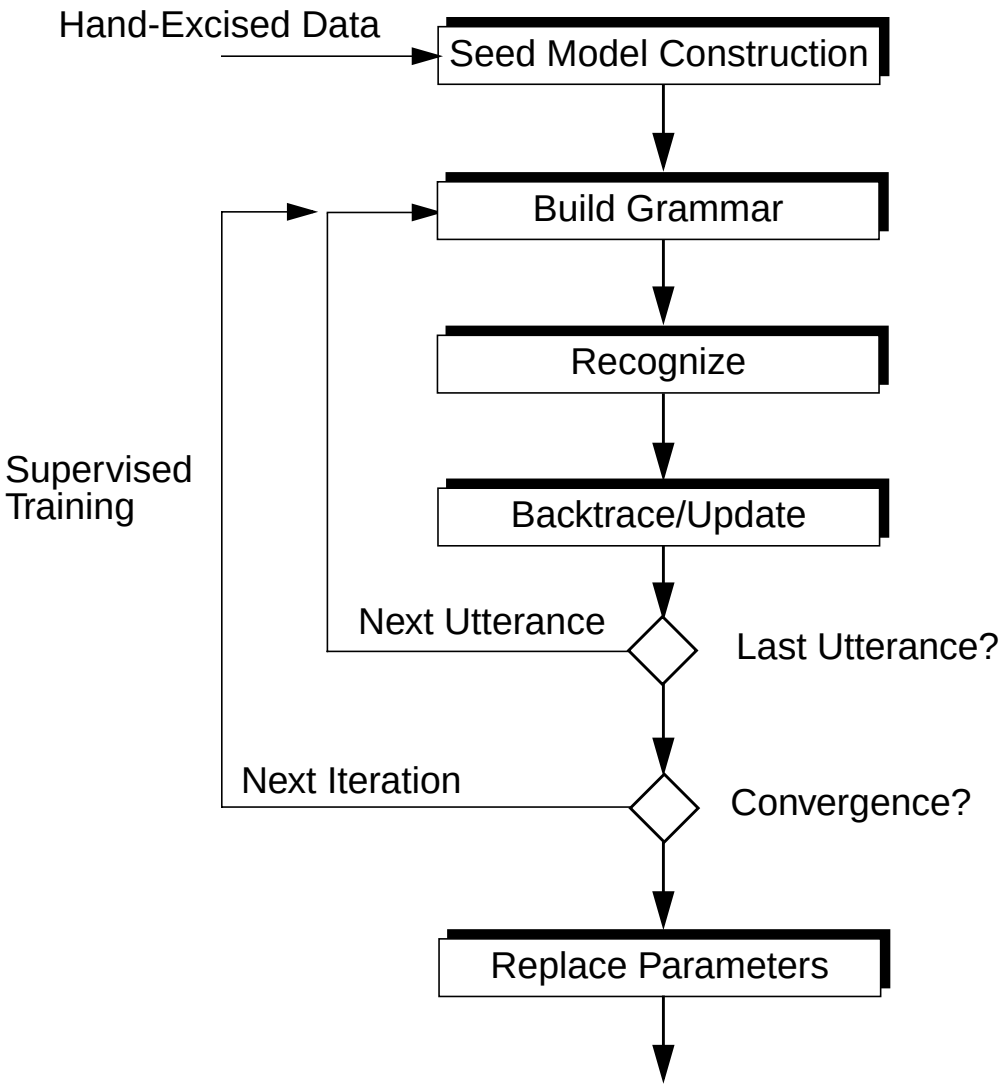
$$P(y_1^t, (\underline{x}(t) = i) \,|\, y, M) = P(\underline{x}(t-1) = j \,|\, y, M)(a_{ij})^{\alpha} + b_i(\boldsymbol{y}(t))^{\beta}$$

or, in the log prob space:

$$\log\left\{P(y_1^t, \ldots)\right\} = \log\{P(\underline{x}(t-1) = j)\ldots\} + \alpha\log(a_{ij}) + \beta\log D_{y,\mu}$$

This result emphasizes the similarities between HMMs and DTW. The weights, α and β can be used to control the importance of the "language model."

An Overview of the Training Schedule



Note that *a priori* segmentation of the utterance is not required, and that the recognizer is forced to recognize the utterance during training (via the build grammar operation). This forces the recognizer to learn contextual variations, provided the seed model construction is done "properly."

What about speaker independence?
Speaker dependence?
Speaker adaptation?
Channel adaptation?