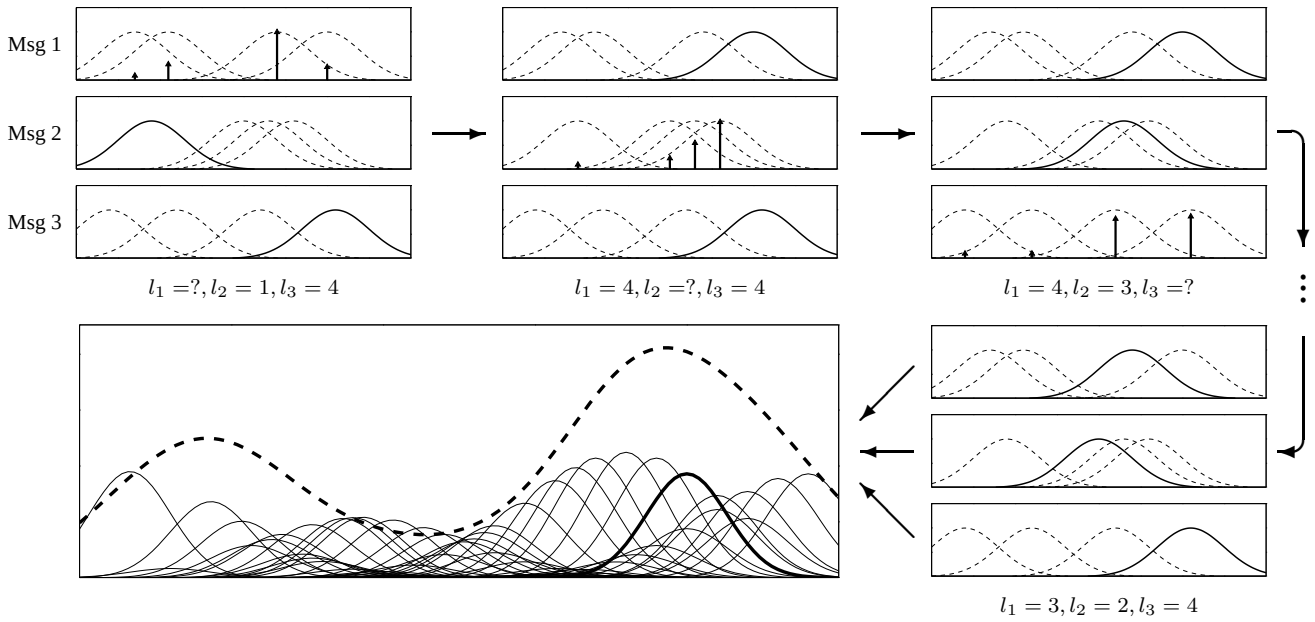

Figure 2. *Top row:* **Gibbs sampler for a product of 3 Gaussian mixtures, with 4 kernels each. New indices are sampled according to weights (arrows) determined by the two fixed components (solid). The Gibbs sampler cycles through the different messages, drawing a new mixture label for one message conditioned on the currently labeled Gaussians in the other messages.** *Bottom row:* **After κ iterations through all the messages, the final labeled Gaussians for each message (right, solid) are multiplied together to identify one (left, solid) of the 4^3 components (left, thin) of the product density (left, dashed).**

to more accurate samples, but require greater computational cost. Following the final iteration, a single sample is drawn from the product mixture component identified by the final labels. To draw M (approximate) samples from the product density, the Gibbs sampler requires $\mathcal{O}(d\kappa M^2)$ operations.

Although formal verification of the Gibbs sampler's convergence is difficult, our empirical results indicate that accurate Gibbs sampling typically requires far fewer computations than direct sampling. Note that NBP uses the Gibbs sampling method differently from classic simulated annealing algorithms [8]. In simulated annealing, the Gibbs sampler updates a single Markov chain whose state dimension is proportional to the graph dimension. In contrast, NBP uses many *local* Gibbs samplers, each involving only a few nodes. Thus, although NBP must run more independent Gibbs samplers, for large graphs the dimensionality of the corresponding Markov chains is dramatically smaller.

In some applications, the observation potentials $\psi_t(x_t, y_t)$ are specified by analytic functions. The Gibbs sampler may be adapted to this case using *importance sampling* [5], as shown in Algorithm 2. At each iteration, the weight of the i^{th} kernel label is rescaled by $\psi_t\left(\bar{\mu}^{(i)}, y_t\right)$, the observation likelihood at that kernel's center. Then, the final sample is assigned an importance weight to account for variations of the analytic potential over the kernel's support. This procedure is most effective when $\psi_t(x_t, y_t)$ varies slowly relative to the typical kernel bandwidth.

3.2. Message Propagation

The NBP algorithm's second stage propagates each sample from the message product by approximating the belief update integral (2). This stochastic integration requires a decomposition of the pairwise potential $\psi_{s,t}(x_s, x_t)$ which separates its *marginal* influence on x_t from the *conditional* relationship it defines between x_s and x_t.

The marginal influence function $\zeta(x_t)$ is determined by the relative weight assigned to *all* x_s values for each x_t:

$$\zeta(x_t) = \int_{x_s} \psi_{s,t}(x_s, x_t)\, dx_s \tag{8}$$

The NBP algorithm accounts for this marginal influence by incorporating $\zeta(x_t)$ into the Gibbs sampler. If $\psi_{s,t}(x_s, x_t)$ is a Gaussian mixture, $\zeta(x_t)$ is easily calculated. Alternately, importance sampling may be used, assuming $\zeta(x_t)$ can be evaluated (or approximated) pointwise. In the common case where pairwise potentials depend only on the difference between their arguments ($\psi_{s,t}(x, \bar{x}) = \psi_{s,t}(x - \bar{x})$), $\zeta(x_t)$ is constant and can be neglected.

To complete the stochastic integration, each sample $x_t^{(i)}$ from the message product is propagated to the neighboring node by sampling $x_s^{(i)} \sim \psi_{s,t}(x_s, x_t^{(i)})$, where equation (5) ensures that $\psi_{s,t}(x_s, x_t^{(i)})$ is normalizable. Finally, having produced a set of independent samples from the output message $m_{ts}(x_s)$, NBP selects a kernel bandwidth to complete the nonparametric density estimate. There are many ways to make this choice; the results in this paper use the compu-