Fast Methods and Nonparametric Belief Propagation

Alexander Ihler

Massachusetts Institute of Technology

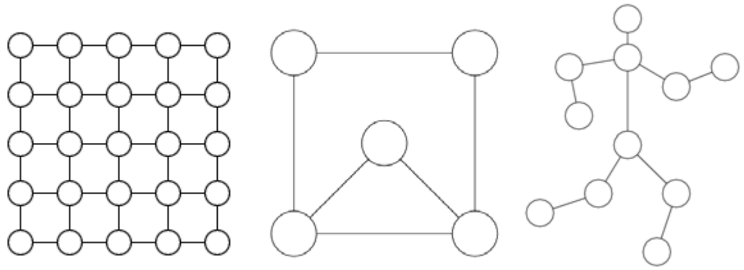
ihler@mit.edu

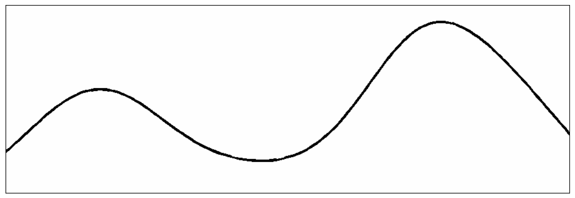
Joint work with Erik Sudderth
William Freeman
Alan Willsky

Introduction

Nonparametric BP

- Perform inference on graphical models with variables which are
  - Continuous
  - High-dimensional
  - Non-Gaussian
- Sampling-based extension to BP
  - Applicable to general graphs
  - Nonparametric representation of uncertainty
- Efficient implementation requires fast methods





Outline

Background

- Graphical Models & Belief Propagation
- Nonparametric Density Estimation

Nonparametric BP Algorithm

- Propagation of nonparametric messages
- *Efficient multiscale sampling from products of mixtures*

Some Applications

- Sensor network self-calibration
- Tracking multiple indistinguishable targets
- Visual tracking of a 3D kinematic hand model

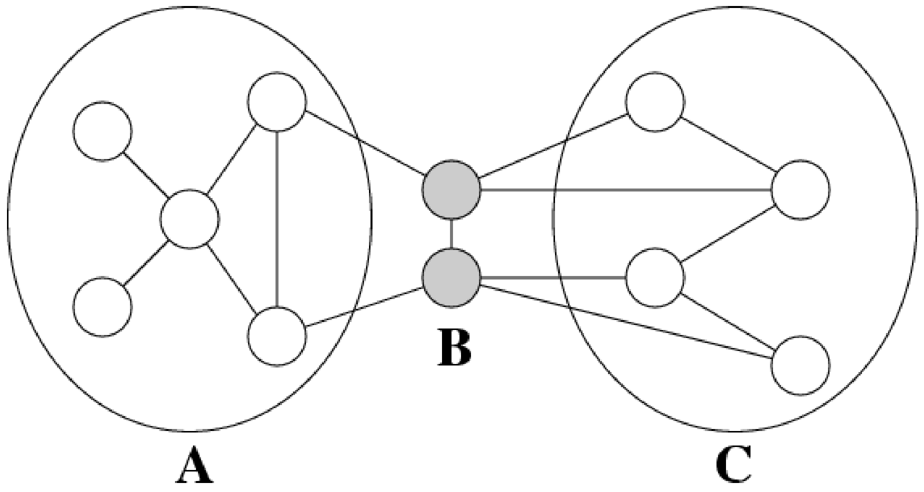
Graphical Models

An undirected graph $\mathcal{G}$ is defined by

$\mathcal{V} \longrightarrow$ set of N nodes $\{1, 2, \ldots, N\}$

$\mathcal{E} \longrightarrow$ set of edges (s, t) connecting nodes $s, t \in \mathcal{V}$

Nodes $s \in \mathcal{V}$ are associated with random variables x_s



Graph Separation

$\Updownarrow$

Conditional Independence

$$p(x_A, x_C | x_B) = p(x_A | x_B) p(x_C | x_B)$$

Pairwise Markov Random Fields

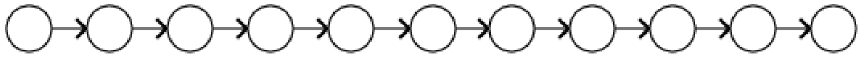
$$p(x,y) = \frac{1}{Z} \prod_{(s,t)\in\mathcal{E}} \psi_{st}(x_s, x_t) \prod_{s\in\mathcal{V}} \psi_s(x_s, y_s)$$

$x_s \longrightarrow$ hidden random variable at node s

$y_s \longrightarrow$ noisy *local* observation of x_s

Special Case:
Temporal Markov
Chain Model (HMM)

$$p(x,y) = p(x_0) \prod_{t=1}^{T} p(x_t | x_{t-1}) p(y_t | x_t)$$



GOAL: Determine the conditional *marginal* distributions

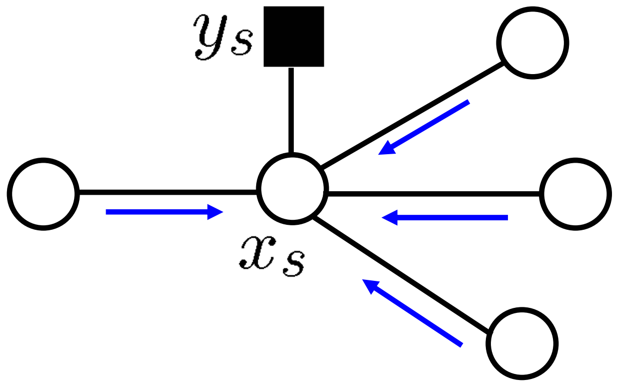
$$p(x_s|y) = \alpha \int_{x_{\mathcal{V}\setminus s}} p(x,y)\, dx_{\mathcal{V}\setminus s}$$

- Estimates: Bayes' least squares, max marginals, …
- Degree of confidence in those estimates

Belief Propagation

Beliefs: *Approximate posterior distributions summarizing information provided by all given observations*

- Combine the observations from all nodes in the graph through a series of local *message-passing* operations



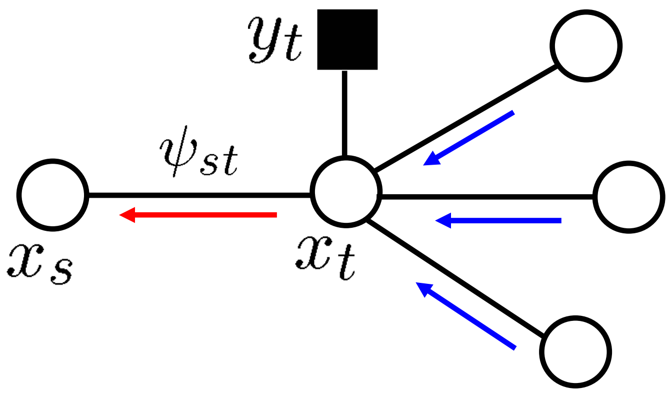
$$\widehat{p}(x_s|y) = \alpha\psi_s(x_s, y_s) \prod_{t\in\Gamma(s)} m_{ts}(x_s)$$

$\Gamma(s) \longrightarrow$ *neighborhood* of node *s* (adjacent nodes)

$m_{ts}(x_s) \longrightarrow$ *message* sent from node *t* to node *s*

(“sufficient statistic” of *t*’s knowledge about *s*)

BP Message Updates



$$m_{ts}(x_s) = \alpha \int_{x_t} \underbrace{\psi_{s,t}(x_s,x_t)}_{\text{II}} \underbrace{\psi_t(x_t,y_t) \prod_{u\in\Gamma(t)\setminus s} m_{ut}(x_t)}_{\text{I}} \, dx_t$$

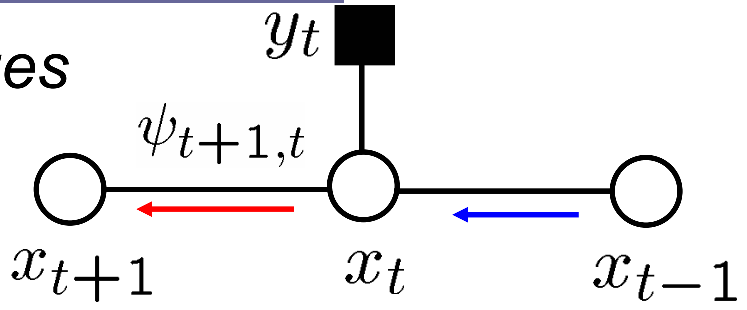
I. Message Product: Multiply incoming messages (from all nodes but s) with the local observation to form a distribution over x_t

II. Message Propagation: Transform distribution from node t to node s using the pairwise interaction potential $\psi_{s,t}(x_s,x_t)$

$\longrightarrow$ Integrate over x_t to form distribution summarizing node t's knowledge about x_s

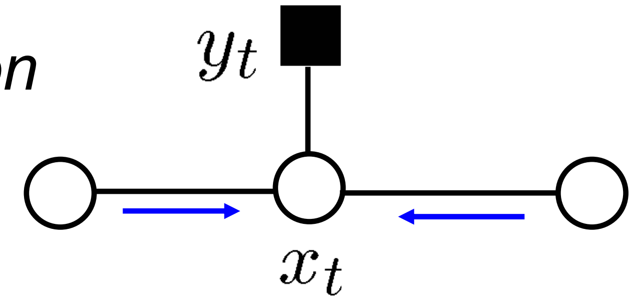
BP for HMMs

Forward Messages



$$m_{t,t+1}(x_{t+1}) = \alpha \int_{x_t} \underbrace{p(x_{t+1}|x_t)}_{\text{Message Propagation}} \underbrace{p(y_t|x_t)m_{t-1,t}(x_t)}_{\text{Message Product}} \, dx_t$$

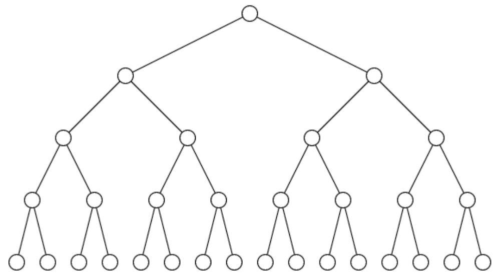
Belief Computation



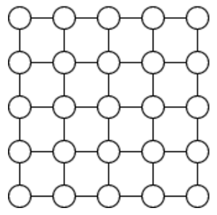
$$p(x_t|y) \propto p(y_t|x_t)m_{t-1,t}(x_t)m_{t+1,t}(x_t)$$

BP Justification

- Produces *exact* conditional marginals for tree-structured graphs (no cycles)



- For general graphs, exhibits excellent empirical performance in many applications (especially coding)



Statistical Physics & Free Energies *(Yedidia, Freeman, and Weiss)*

Variational interpretation, improved region-based approximations

BP as Reparameterization *(Wainwright, Jaakkola, and Willsky)*

Characterization of fixed points, error bounds

Many others…

Representational Issues

$$m_{ts}(x_s) = \alpha \int_{x_t} \psi_{s,t}(x_s, x_t)\psi_t(x_t, y_t) \prod_{u \in \Gamma(t) \setminus s} m_{ut}(x_t)\, dx_t$$

Message representations:

Discrete: Finite vectors

Gaussian: Mean and covariance (Kalman filter)

Continuous Non-Gaussian: No parametric form

$\longrightarrow$ Discretization intractable in as few as 2-3 dimensions

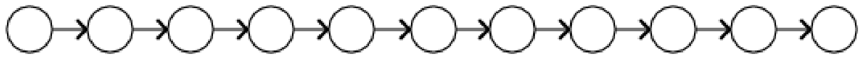
BP Properties:

- May be applied to arbitrarily structured graphs, *but*
- Updates intractable for most continuous potentials

Particle Filters

Condensation, Sequential Monte Carlo, Survival of the Fittest,…

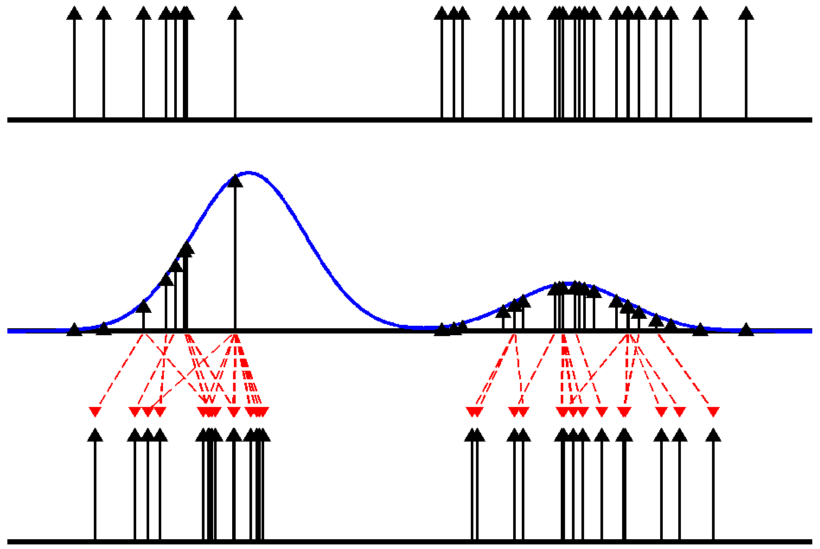
Nonparametric Markov chain inference:



Sample-based density estimate

Weight by observation likelihood

Resample & propagate by dynamics



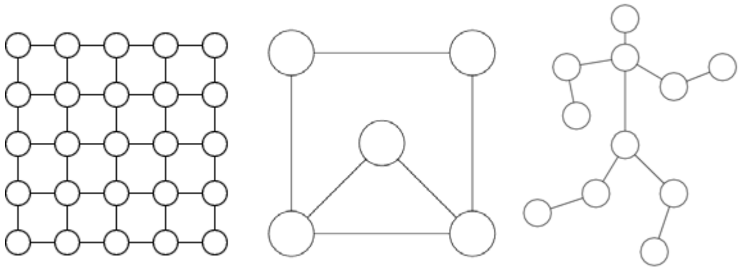
Particle Filter Properties:

- May approximate complex continuous distributions, *but*
- Update rules dependent on Markov chain structure

Nonparametric Inference For General Graphs

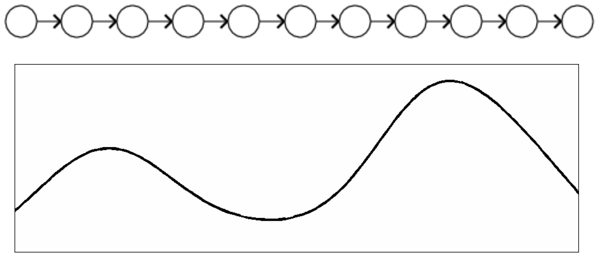
Belief Propagation

- General graphs
- Discrete or Gaussian



Particle Filters

- Markov chains
- General potentials



Nonparametric BP

- General graphs
- General potentials

Problem: What is the product of two collections of particles?

Nonparametric Density Estimates

Kernel (Parzen Window)
Density Estimator

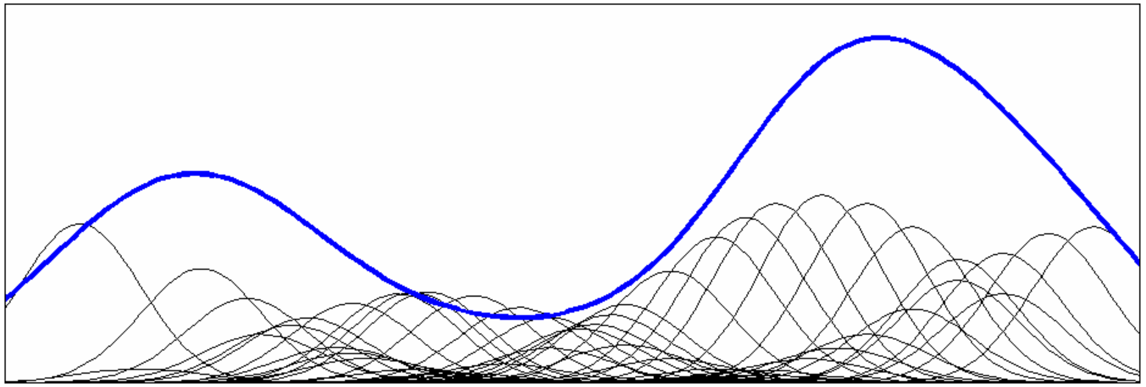
Approximate PDF by a set of smoothed data samples

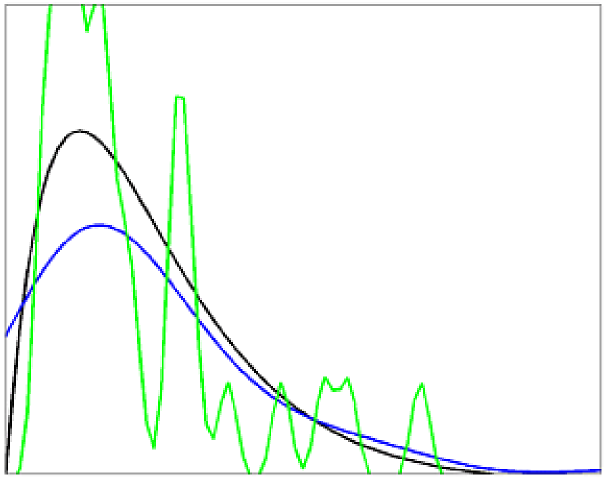
$$\hat{p}(x) = \frac{1}{M} \sum_{i=1}^{M} \frac{1}{\sigma} K\left(\frac{x - X_i}{\sigma}\right)$$

$\{X_i\} \longrightarrow$ *M* independent samples from *p(x)*

$K(\cdot) \longrightarrow$ *Gaussian* kernel function (self-reproducing)

$\sigma \longrightarrow$ Bandwidth (chosen automatically)





Outline

Background

- Graphical Models & Belief Propagation
- Nonparametric Density Estimation

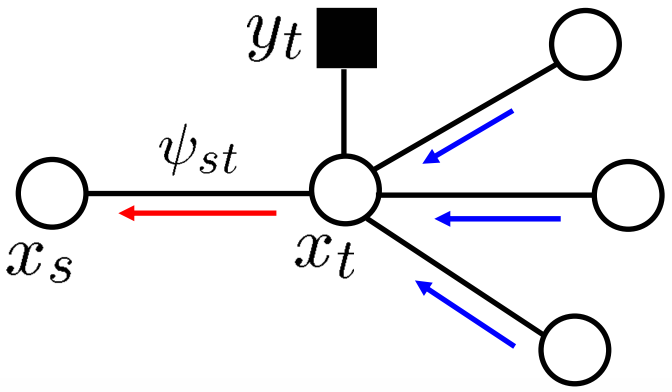
Nonparametric BP Algorithm

- Propagation of nonparametric messages
- Efficient multiscale sampling from products of mixtures

Results

- Sensor network self-calibration
- Tracking multiple indistinguishable targets
- Visual tracking of a 3D kinematic hand model

Nonparametric BP



$$m_{ts}(x_s) = \alpha \int_{x_t} \underbrace{\psi_{s,t}(x_s, x_t)}_{\text{}} \underbrace{\psi_t(x_t, y_t) \prod_{u \in \Gamma(t) \setminus s} m_{ut}(x_t)\, dx_t}_{\text{}}$$

Stochastic update of kernel based messages:

I. Message Product: Draw samples of x_t from the product of all incoming messages and the local observation potential

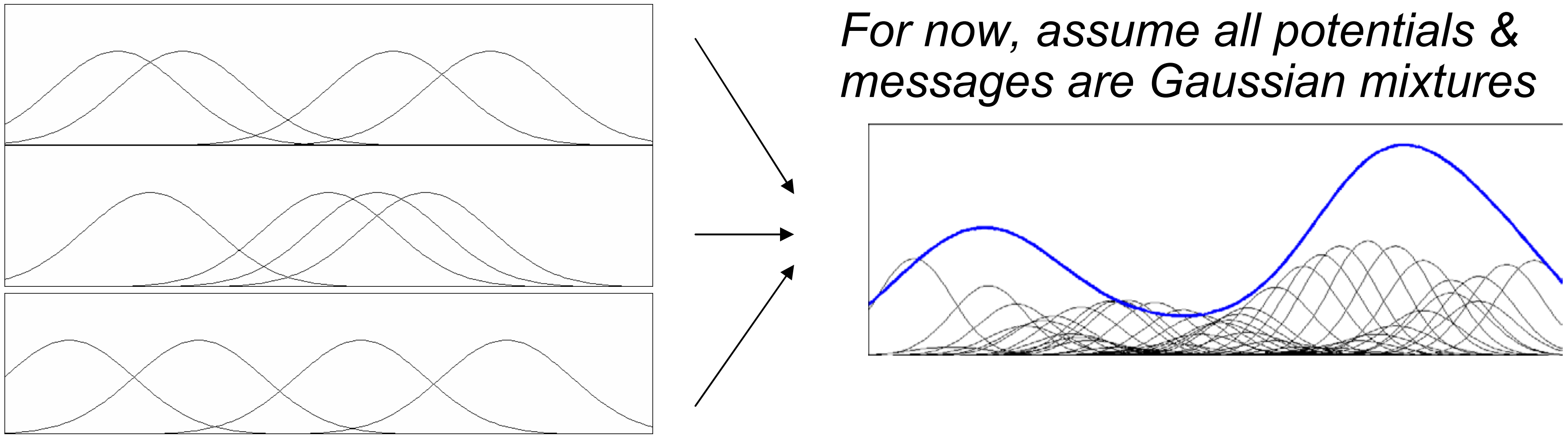
II. Message Propagation: Draw samples of x_s from the compatibility $\psi_{st}(x_s, x_t)$, fixing x_t to the values sampled in step I

$\longrightarrow$ Samples form new kernel density estimate of outgoing message (determine new kernel bandwidths)

I. Message Product

$$m_{ts}(x_s) = \alpha \int_{x_t} \psi_{s,t}(x_s, x_t) \underbrace{\psi_t(x_t, y_t) \prod_{u \in \Gamma(t) \setminus s} m_{ut}(x_t)}_{} \, dx_t$$

For now, assume all potentials & messages are Gaussian mixtures



d messages

M kernels each

$\Longrightarrow$

Product contains M^d kernels

How do we sample from the product distribution without explicitly constructing it?

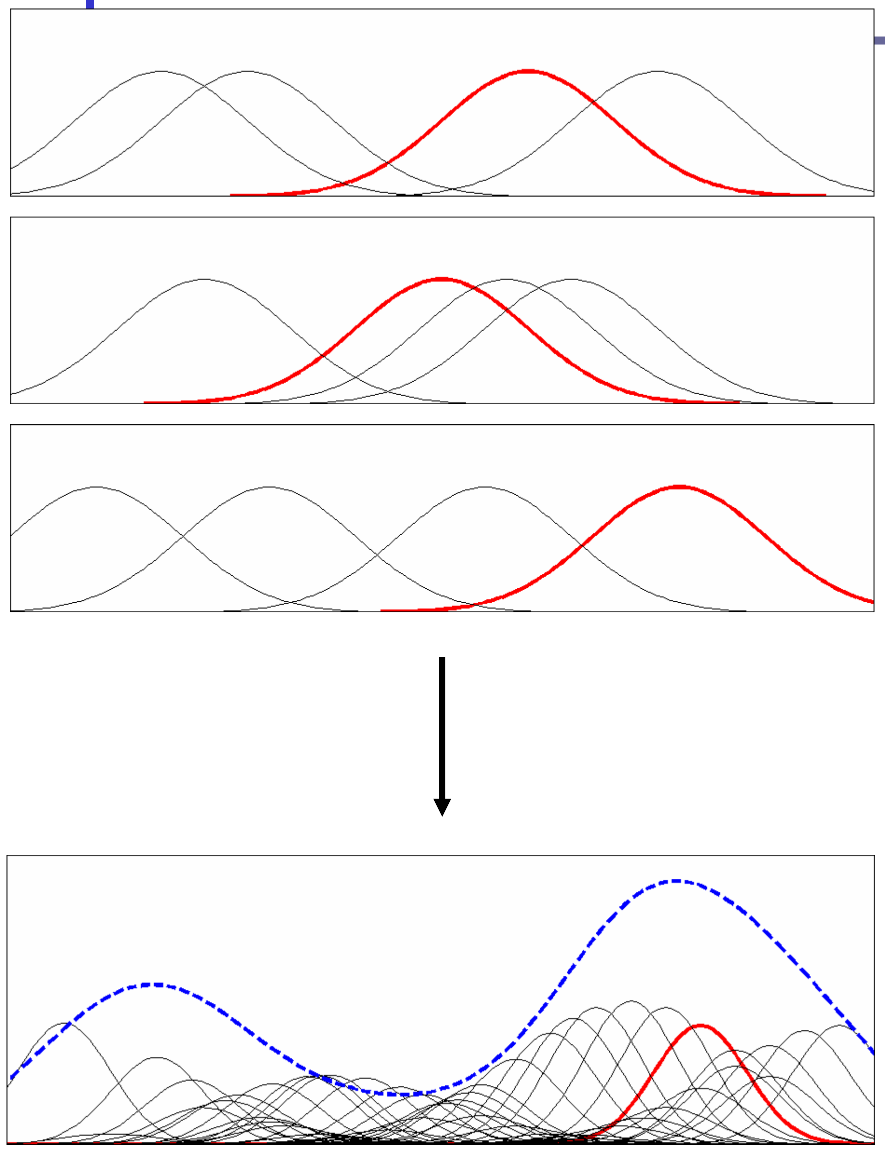
Sampling from Product Densities

d mixtures of M Gaussians → *mixture of M^d Gaussians*

$$p_i(x) = \sum_{l_i} w_{l_i} \mathcal{N}\left(x; \mu_{l_i}, \Lambda_i\right) \longrightarrow p(x) \propto \prod_{i=1}^{d} p_i(x)$$

- Exact sampling
- Importance sampling
  - Proposal distribution?
- Gibbs sampling
  - “parallel” & “sequential” versions
- Multiscale Gibbs sampling
- Epsilon-exact multiscale sampling

Product Mixture Labelings



Kernel in product density

$\updownarrow$

Labeling of a single mixture component in each message

Products of Gaussians are also Gaussian, with easily computed mean, variance, and mixture weight:

$$\prod_{i=1}^{d} \mathcal{N}(x; \mu_i, \Lambda_i) \propto \mathcal{N}(x; \bar{\mu}, \bar{\Lambda})$$

$$\bar{\Lambda}^{-1} = \sum_{i=1}^{d} \Lambda_i^{-1} \qquad \bar{\Lambda}^{-1}\bar{\mu} = \sum_{i=1}^{d} \Lambda_i^{-1}\mu_i$$

$$\bar{w} \propto \frac{\prod_{i=1}^{d} w_i \mathcal{N}(x; \mu_i, \Lambda_i)}{\mathcal{N}(x; \bar{\mu}, \bar{\Lambda})}$$

Exact Sampling

$l_i \longrightarrow$ mixture component label for i^{th} input density

$L = [l_1, \ldots, l_d] \rightarrow$ label of component in product density

$$w_L = \frac{\prod_{i=1}^{d} w_{l_i} \mathcal{N}(x; \mu_{l_i}, \Lambda_i)}{\mathcal{N}(x; \mu_L, \Lambda_L)} \qquad \Lambda_L^{-1} = \sum_{i=1}^{d} \Lambda_i^{-1} \qquad \Lambda_L^{-1}\mu_L = \sum_{i=1}^{d} \Lambda_i^{-1}\mu_{l_i}$$

- Calculate the weight partition function in *O(M^d)* operations: $Z = \sum_L w_L$
- Draw and sort M uniform [0,1] variables
- Compute the cumulative distribution of

$$p(L) = \frac{w_L}{Z}$$

Importance Sampling

$p(x) \longrightarrow$ true distribution (difficult to sample from)
assume may be evaluated *up to normalization Z*

$q(x) \longrightarrow$ proposal distribution (easy to sample from)

- Draw N ¸ M samples from proposal distribution:

$$x_i \sim q(x) \qquad\qquad w_i \propto p(x_i)/q(x_i)$$

- Sample M times (with replacement) from

$$\bar{p}(x_i) = w_i/Z$$

Mixture IS: Randomly select a different mixture $p_i(x)$ for each sample (other mixtures provide weight)

Fast Methods:

Need to repeatedly evaluate pairs of densities (FGT, etc.)

Sampling from Product Densities

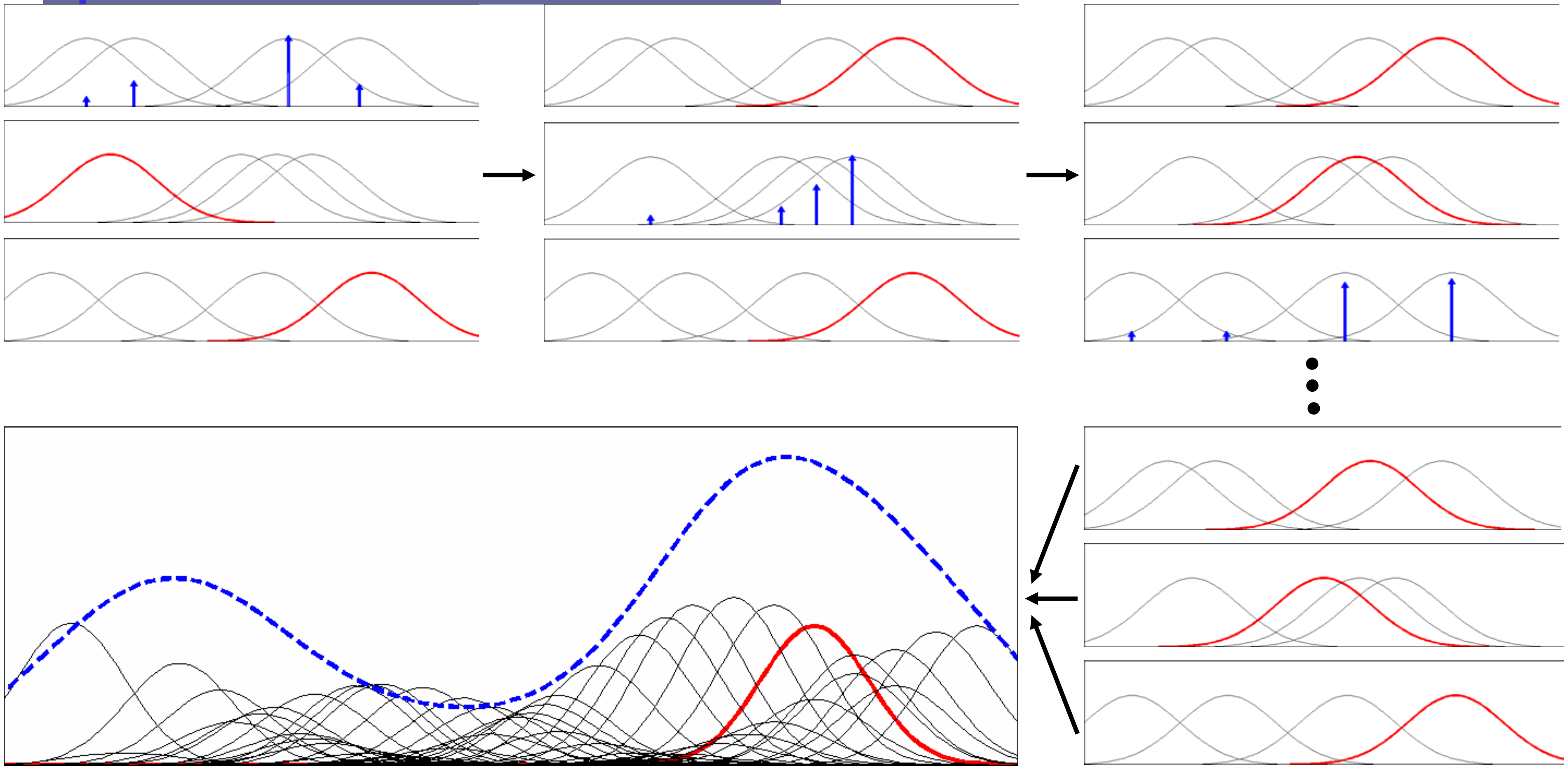
d mixtures of M Gaussians → *mixture of M^d Gaussians*

$$p_i(x) = \sum_{l_i} w_{l_i} \mathcal{N}\left(x; \mu_{l_i}, \Lambda_i\right) \longrightarrow p(x) \propto \prod_{i=1}^{d} p_i(x)$$

- Exact sampling
- Importance sampling
  - Proposal distribution?
- Gibbs sampling
  - “parallel” & “sequential” versions
- Multiscale Gibbs sampling
- Epsilon-exact multiscale sampling

Sequential Gibbs Sampler

Product of 3 messages, each containing 4 Gaussian kernels



Labeled Kernels

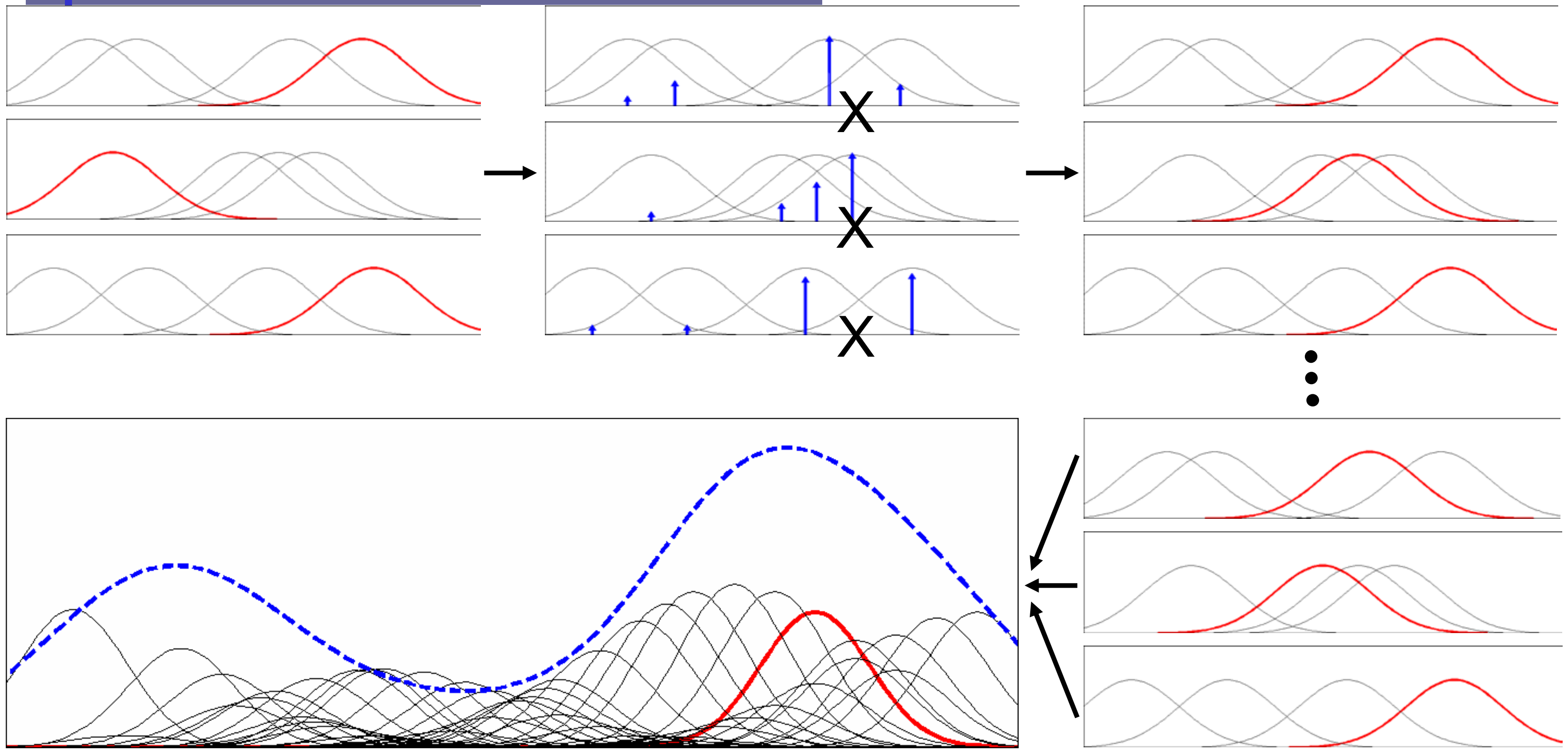
Highlighted Red

Sampling Weights

Blue Arrows

Parallel Gibbs Sampler

Product of 3 messages, each containing 4 Gaussian kernels

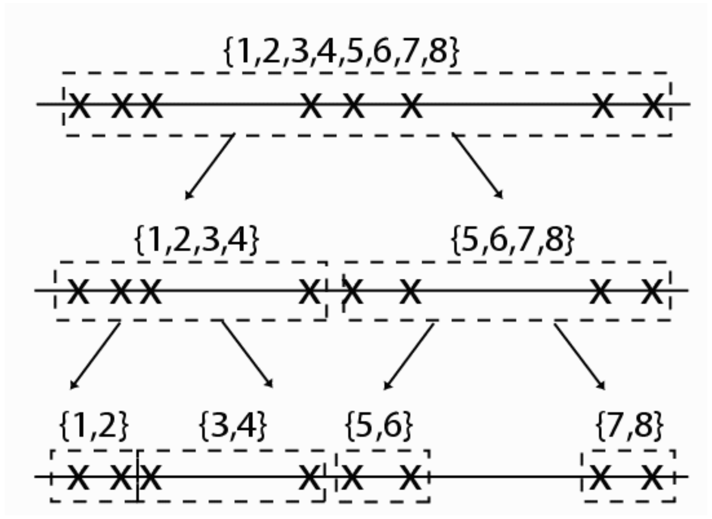


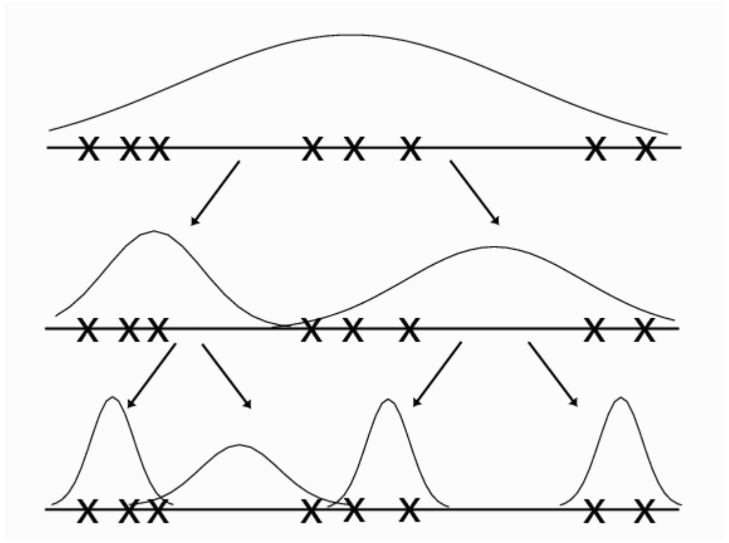
Labeled Kernels *Highlighted Red*

Sampling Weights *Blue Arrows*

Multiscale – KD-trees

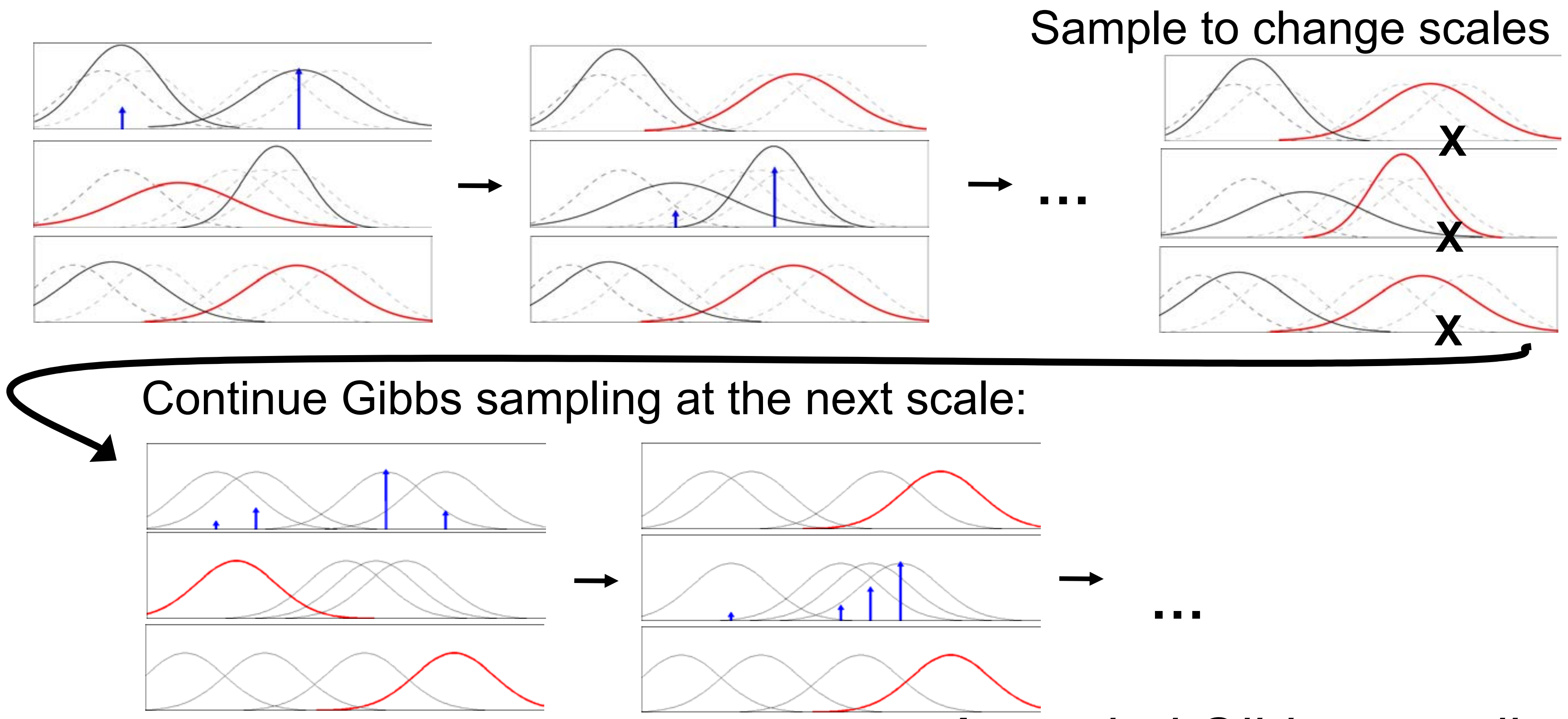
- “K-dimensional Trees”
- Multiscale representation of data set
- Cache statistics of points at each level:
  - Bounding boxes
  - Mean & Covariance
- Original use: efficient search algorithms





Multiscale Gibbs Sampling

- Build KD-tree for each input density
- Perform Gibbs over progressively finer scales:



$\longrightarrow$ *Annealed Gibbs sampling (analogies in MRFs)*

Sampling from Product Densities

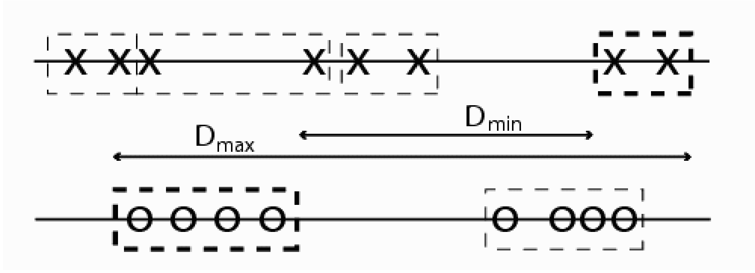
d mixtures of M Gaussians *mixture of M^d Gaussians*

$$p_i(x) = \sum_{l_i} w_{l_i} \mathcal{N}\left(x; \mu_{l_i}, \Lambda_i\right) \longrightarrow p(x) \propto \prod_{i=1}^{d} p_i(x)$$

- Exact sampling
- Importance sampling
  - Proposal distribution?
- Gibbs sampling
  - "parallel" & "sequential" versions
- Multiscale Gibbs sampling
- Epsilon-exact multiscale sampling

ε-Exact Sampling (I)

- Bounding box statistics
  - Bounds on pairwise distances



  - Approximate kernel density evaluation

    KDE: 8 j , evaluate $p(y_j) = \sum_i w_i K(x_i - y_j)$

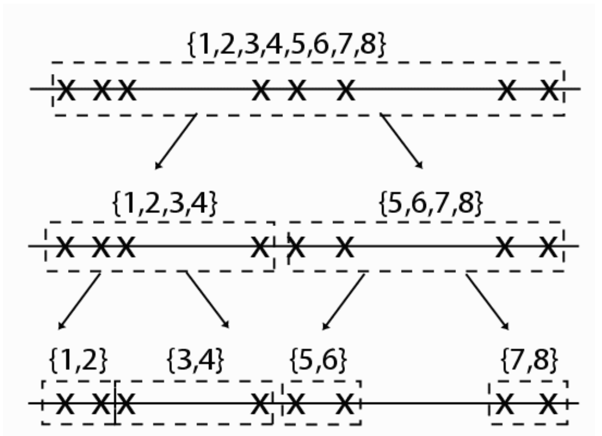
    - FGT – low-rank approximations
    - Gray '03 – rank-**one** approximations
    - Find sets S, T such that

      8 j 2 T , $p(y_j) = \sum_{i \, 2 \, S} K(x_i - y_j)$ ¼ $(\sum_i w_i) C_{ST}$ *(constant)*

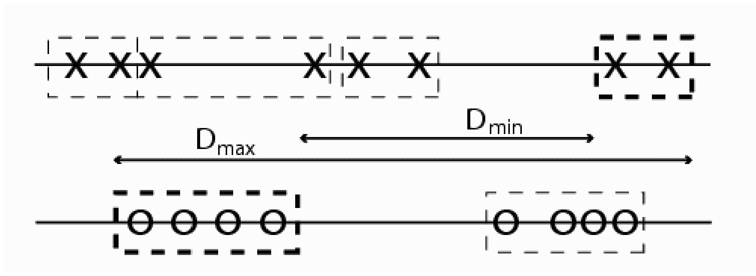
    - Evaluations within fractional error ε:

      If not < ε, refine KD-tree regions

      (= better bounds)



ε-Exact Sampling (II)



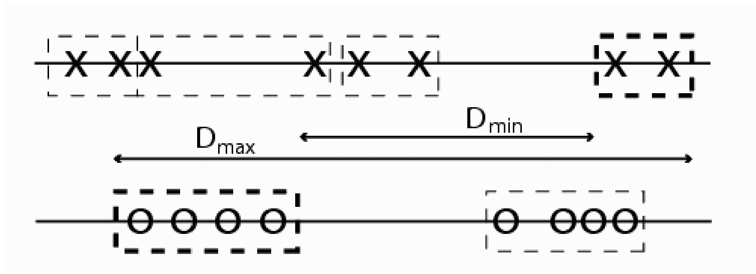
- Use this relationship to bound the weights

$$\bar{w} \propto \frac{\prod_{i=1}^{d} w_i \mathcal{N}(x;\mu_i,\Lambda_i)}{\mathcal{N}(x;\bar{\mu},\bar{\Lambda})} = \left(\prod_{j=1}^{d} w_j\right) \cdot \underbrace{\prod_{(i,j>i)} \mathcal{N}(\mu_i;\mu_j,\Lambda_{(i,j)})}_{\text{(pairwise relationships only)}} \qquad \Lambda_{(i,j)} = \frac{\Lambda_i \Lambda_j}{\Lambda_L}$$

  - Rank-one approximation:
    - Error bounded by product of pairwise bounds
    - Can consider *sets* of weights simultaneously
  - Fractional error tolerance
    - Est'd weights are within a percentage of true value
    - Normalization constant within a percent tolerance

.

ε-Exact Sampling (III)



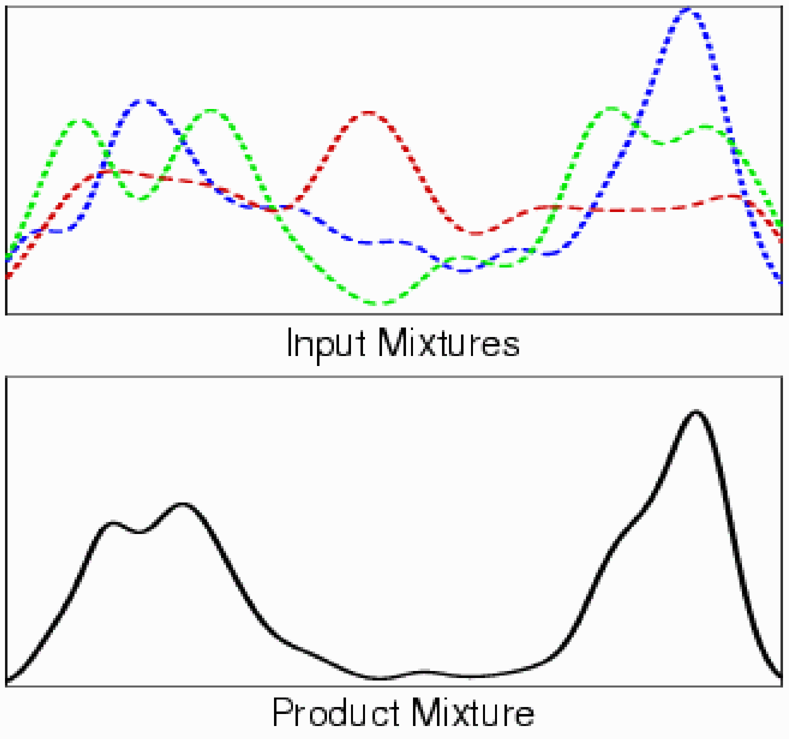
- Each weight has fractional error
- Normalization constant has fractional error
- Normalized weights have *absolute* error:

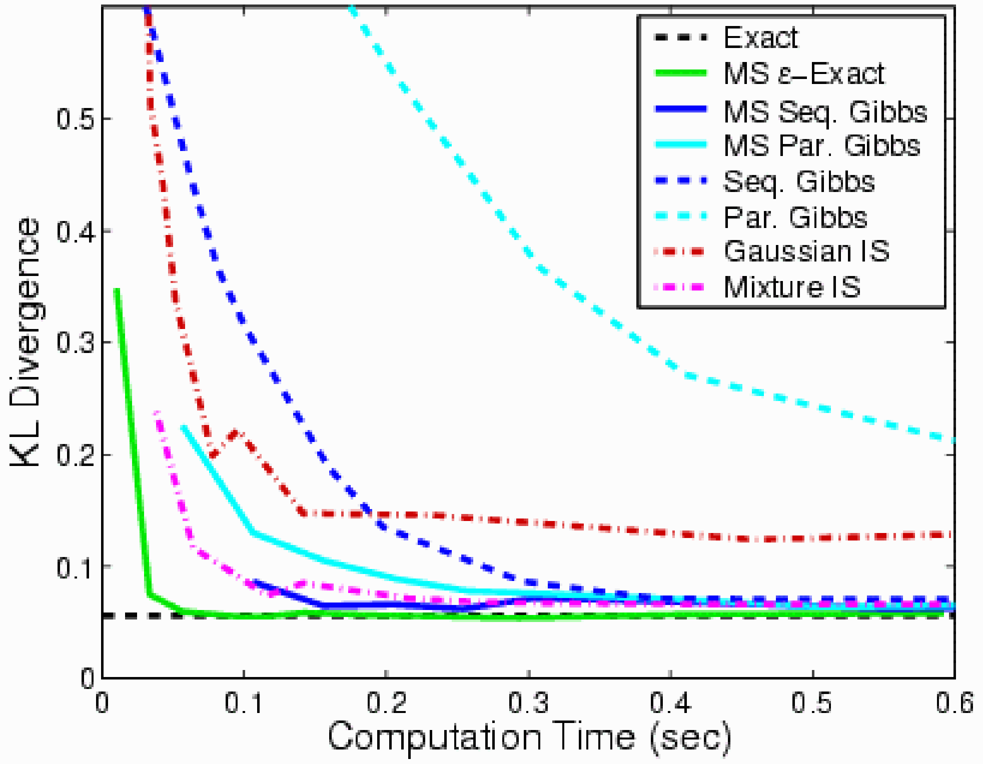
$$|\widehat{p}_L - p_L| = \left| \frac{\widehat{w}_L}{\widehat{Z}} - \frac{w_L}{Z} \right| \le \frac{2\delta}{1-\delta} \equiv \epsilon$$

- Drawing a sample – two-pass
  - Compute approximate sum of weights Z
  - Draw N samples in [0,1) uniformly, sort.
  - Re-compute Z, find *set* of weights for each sample
  - Find label within each set
    - All weights ¼ equal) independent selection

Taking Products – 3 mixtures

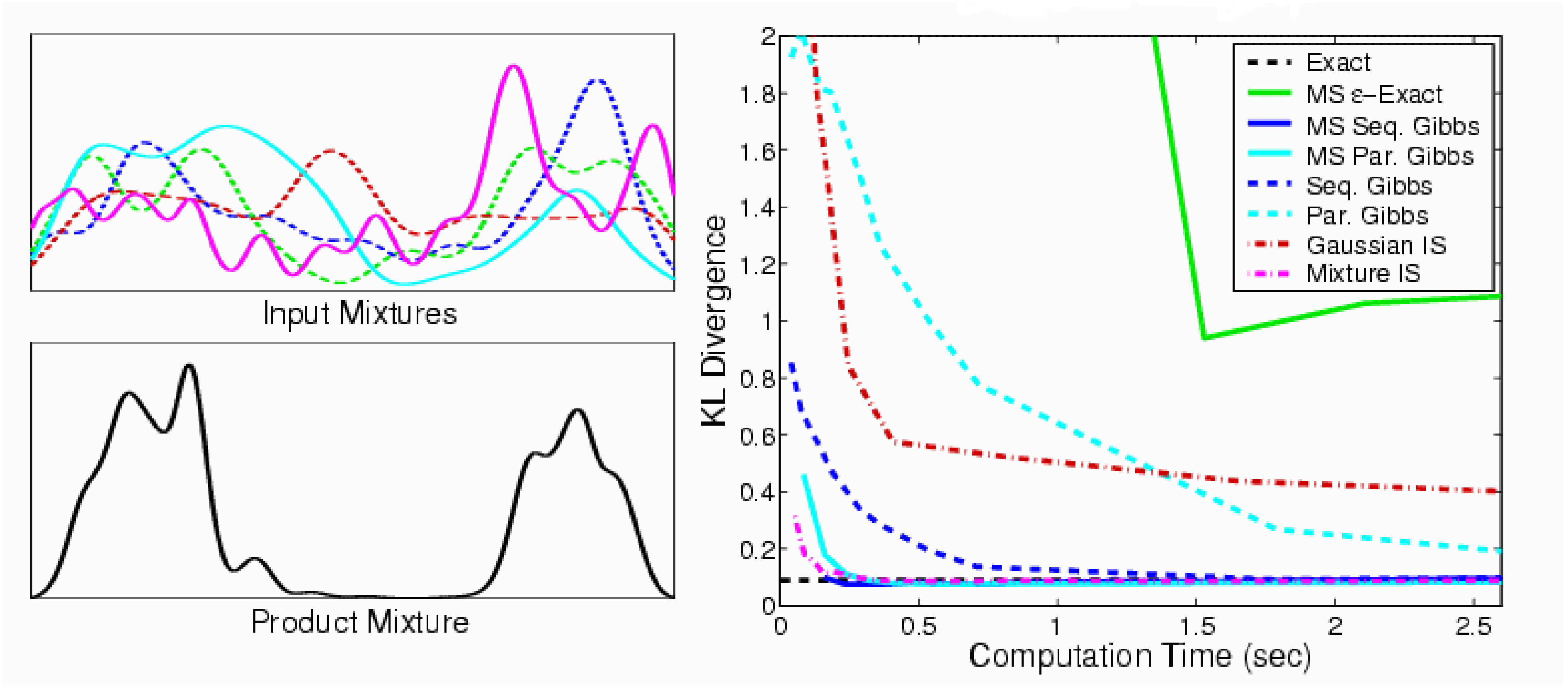
- Epsilon-exact sampling provides the highest accuracy
- Multiscale Gibbs sampling outperforms standard Gibbs
- Sequential Gibbs sampling mixes faster than parallel





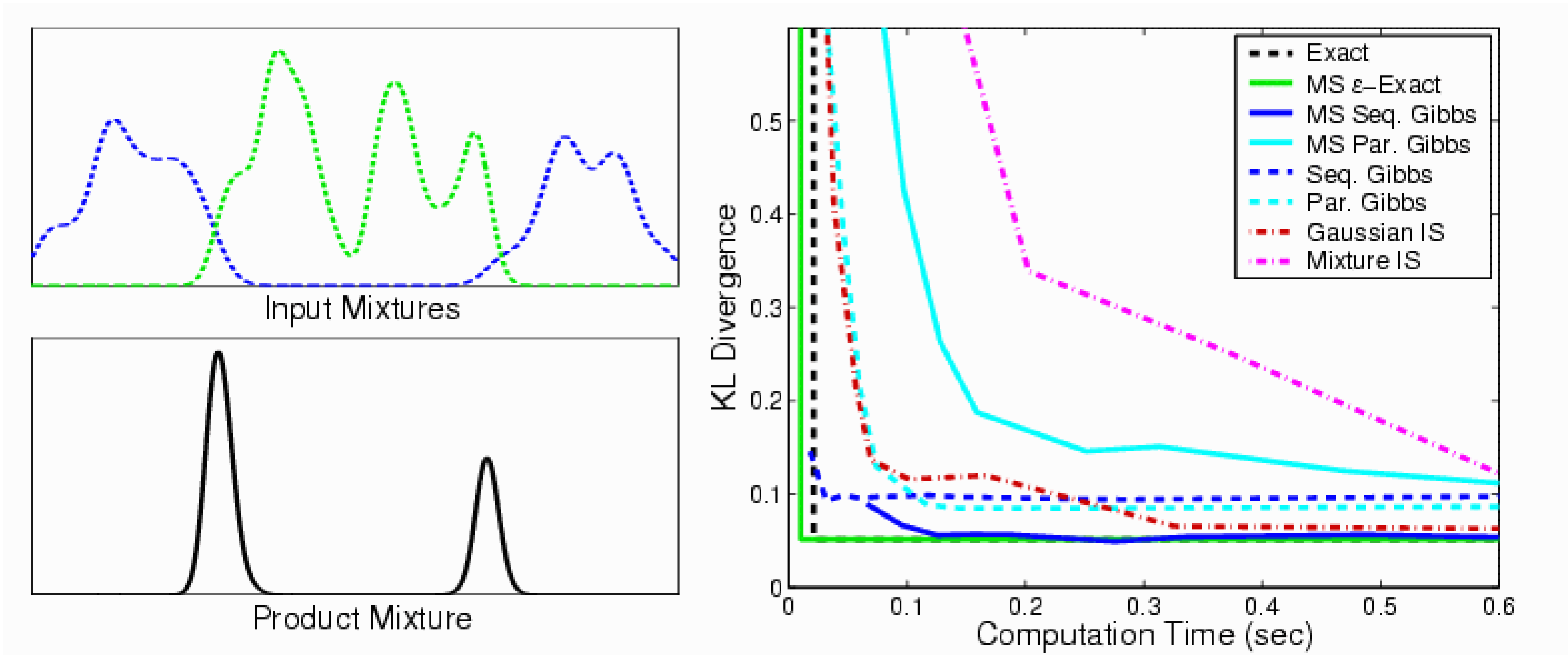
Taking Products – 5 mixtures

- Multiscale Gibbs samplers now outperform epsilon-exact
- Epsilon-exact still beats exact (1 minute vs. 7.6 hours)
- Mixture importance sampling is also very effective



Taking Products – 2 mixtures

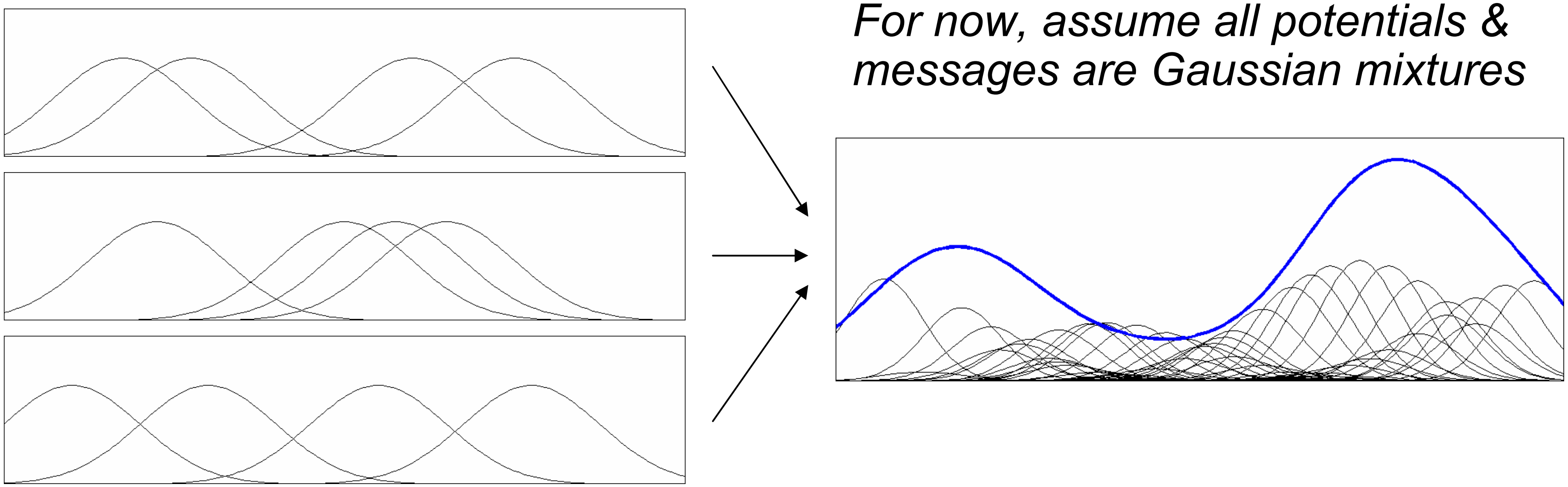
- Importance sampling is sensitive to message alignment
- Multiscale methods show greater consistency & robustness



I. Message Product

$$m_{ts}(x_s) = \alpha \int_{x_t} \psi_{s,t}(x_s, x_t) \underbrace{\psi_t(x_t, y_t) \prod_{u \in \Gamma(t) \setminus s} m_{ut}(x_t)\, dx_t}$$

For now, assume all potentials & messages are Gaussian mixtures



d messages

M kernels each

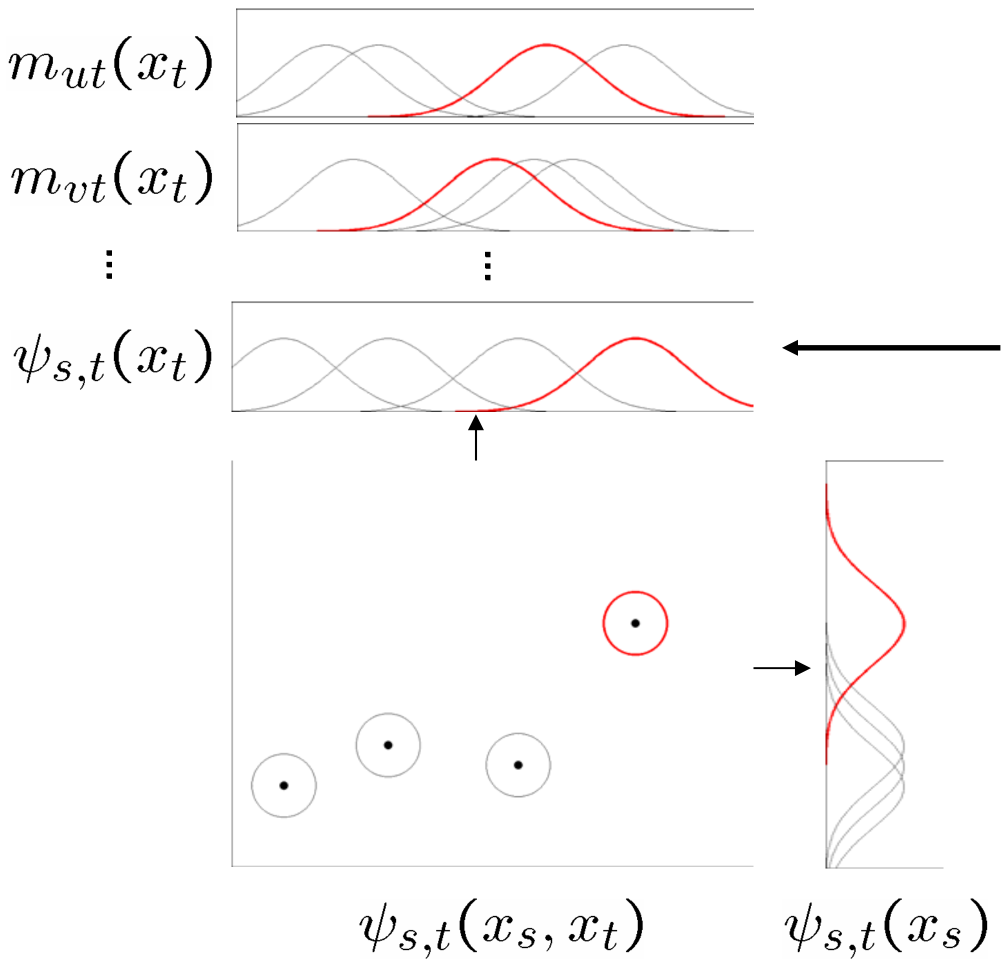
$\Longrightarrow$

Product contains M^d kernels

We can now sample from this message product very efficiently

II. Message Propagation (G.M.)

$$m_{ts}(x_s) = \alpha \int_{x_t} \underbrace{\psi_{s,t}(x_s, x_t)\psi_t(x_t, y_t)}_{} \prod_{u \in \Gamma(t) \setminus s} m_{ut}(x_t)\, dx_t$$



View $\psi_{s,t}(x_s, x_t)$ as a joint distribution

Add marginal $\psi_{s,t}(x_t)$ to the product mix

Label selected by sampler locates kernel center in $\psi_{s,t}(x_s)$

$\longrightarrow$ Draw sample

Extension – Analytic Potentials

$$m_{ts}(x_s) = \alpha \int_{x_t} \psi_{s,t}(x_s, x_t)\psi_t(x_t, y_t) \prod_{u \in \Gamma(t) \setminus s} m_{ut}(x_t)\, dx_t$$

- Assume pointwise evaluation is possible
- Use importance sampling
  - Adjust sampling weights by kernel center value $\psi_t(\bar{\mu}_i, y_t)$
  - Weight final sample by adjustment $w_i = \psi_t(x_t^i, y_t)/\psi_t(\bar{\mu}_i, y_t)$
- Must account for *marginal* influence induced by pairwise potential:

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

$\longrightarrow$ Constant for (common) case $\psi_{s,t}(x_s, x_t) = \psi(x_s - x_t)$

Related Work

Markov Chains

- Regularized particle filters
- Gaussian sum filters
- Monte Carlo HMMs (Thrun & Langford 99)

Approximate Propagation Framework (Koller UAI 99)

- Postulate approximate message representations and updates within junction tree

Particle Message Passing (Isard CVPR 03)

- Avoids bandwidth selection
- Requires pairwise potentials to be small Gaussian mixtures

Outline

Background

- Graphical Models & Belief Propagation
- Nonparametric Density Estimation

Nonparametric BP Algorithm

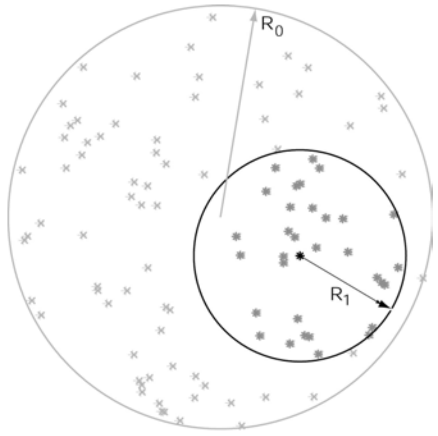
- Propagation of nonparametric messages
- Efficient multiscale sampling from products of mixtures

Results

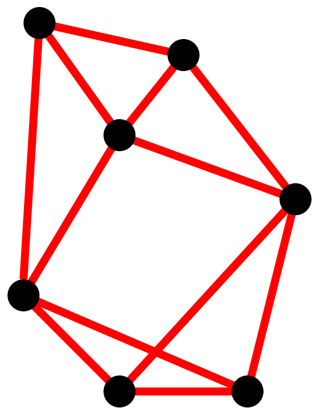
- Sensor network self-calibration
- Tracking multiple indistinguishable targets
- Visual tracking of a 3D kinematic hand model

Sensor Localization

- Limited-range sensors
- Scatter at random
- Each sensor can communicate with other "nearby" sensors
- At most a few sensors have observations of their location



- Measure inter-sensor spacing
  - Time-delay (acoustic)
  - Received signal strength (RF)
- Use relative info to find locations of all other sensors
- Note: MAP estimate *vs.* max-marginal estimate



Uncertainty in Localization

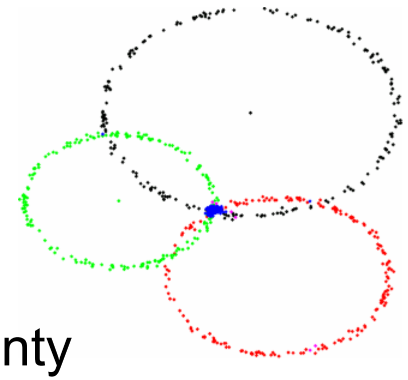
- **Model**

  Location of sensor t is x_t and has prior $p_t(x_t)$

  Observe distance between t and u, $o_{tu} = 1$, with probability

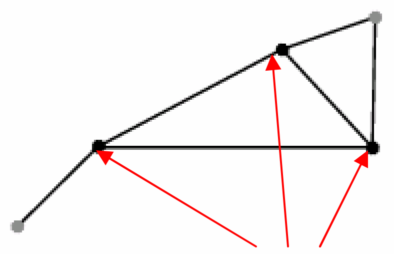
  $$P_o(x_t,x_u) = \exp(- ||x_t-x_u||^\rho / R^\rho) \qquad \text{(e.g. } \rho = 2)$$

  Observe $d_{tu} = ||x_t-x_u|| + \nu$ where $\nu = N(0,\sigma^2)$

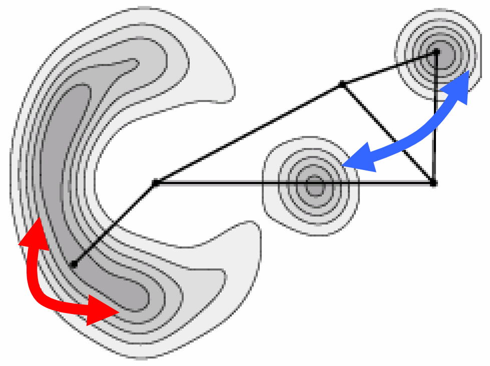


- Nonlinear optimization problem
- Also desirable to have an estimate of posterior uncertainty
- Some sensor locations may be under-determined:

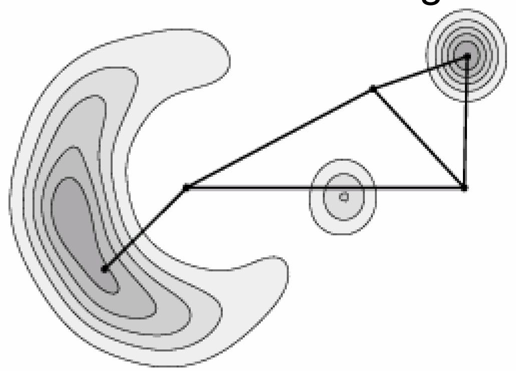
Example Network



True marginal uncertainties

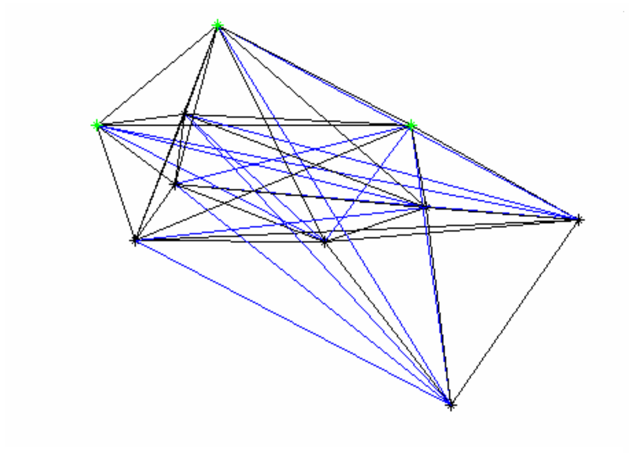


NBP-estimated marginals

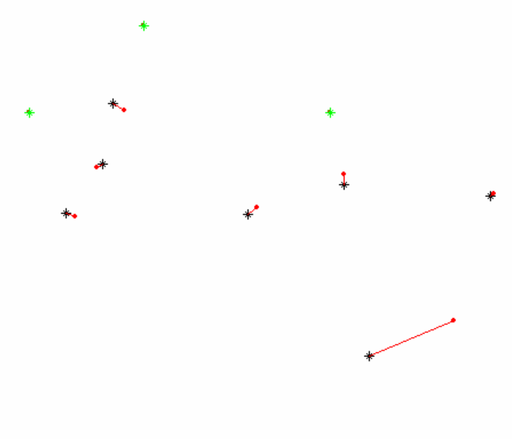


Example Networks : Small

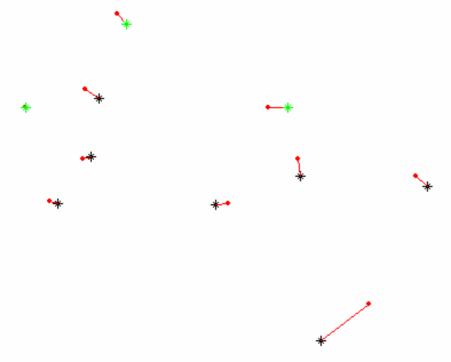
10-Node graph

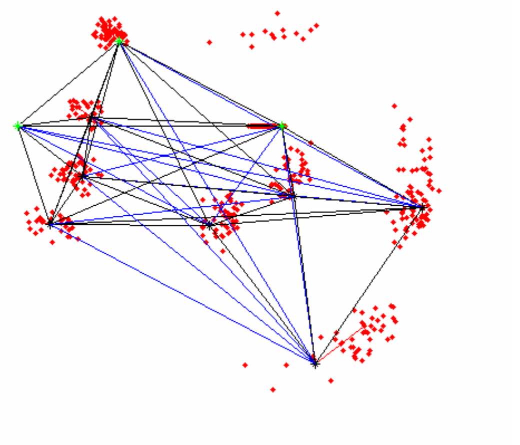


Joint MAP

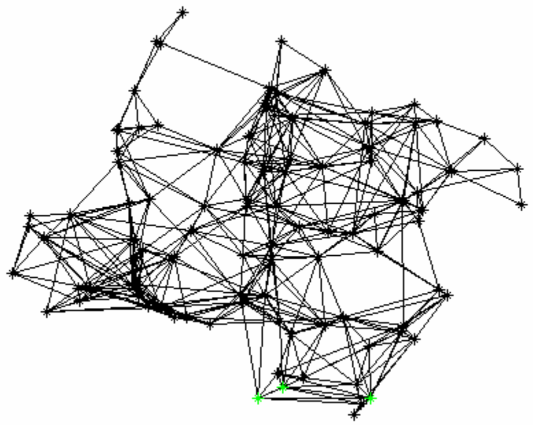


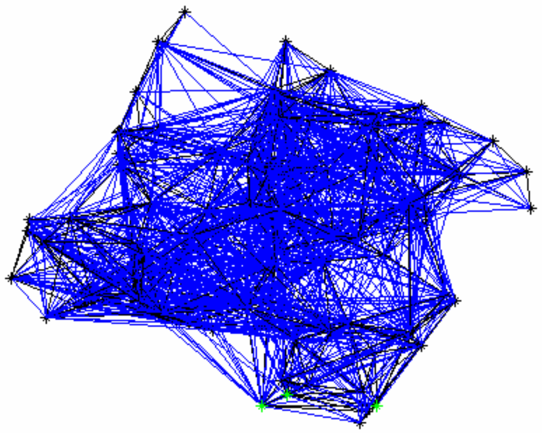
NBP

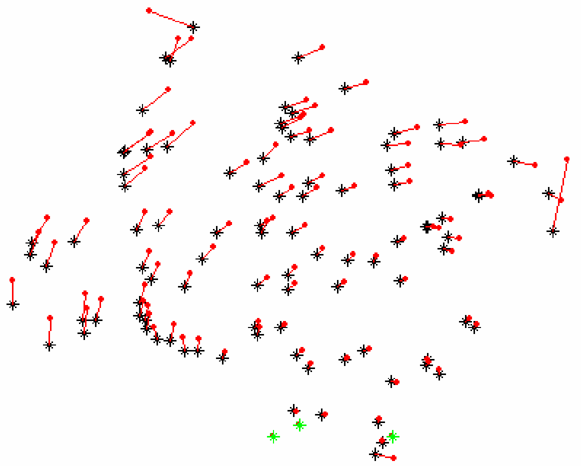


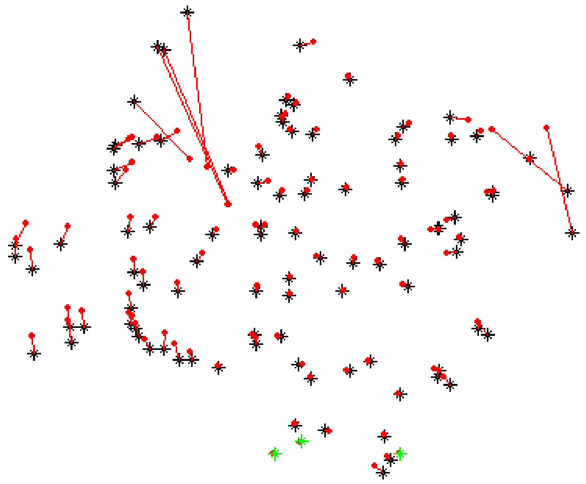


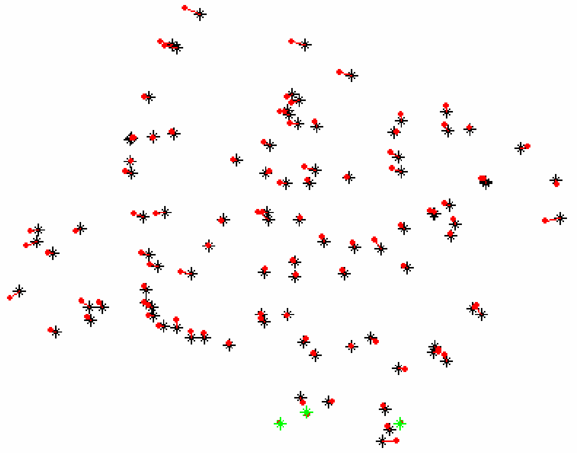
Example Networks : Large


"1-step" Graph

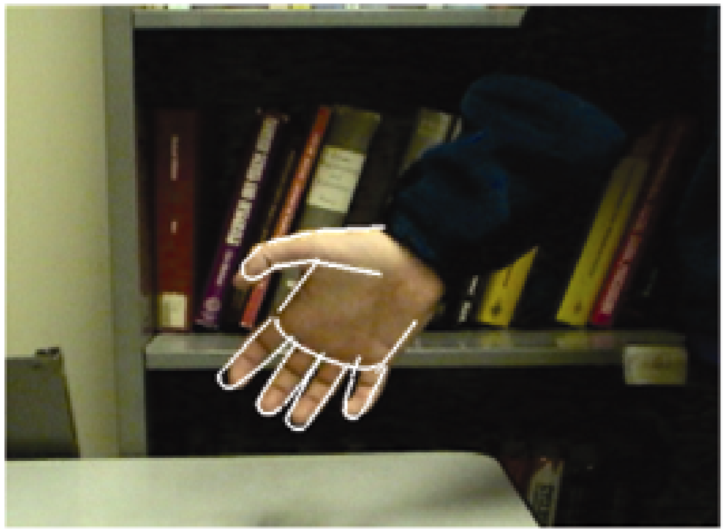

"2-step" Graph

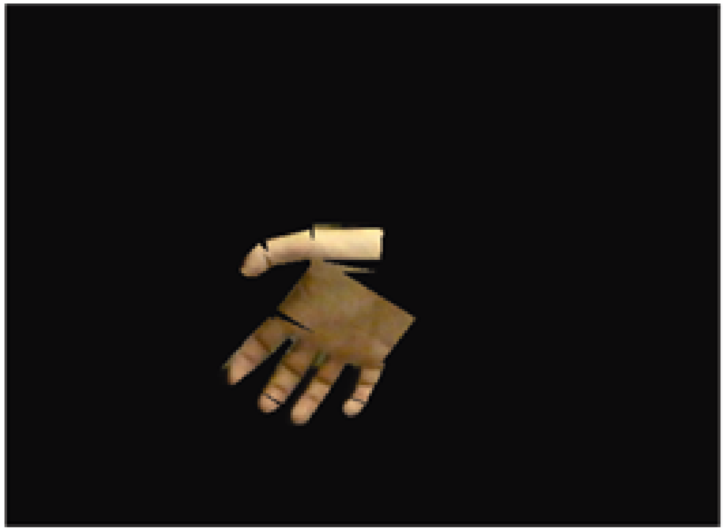

Nonlin Least-Sq

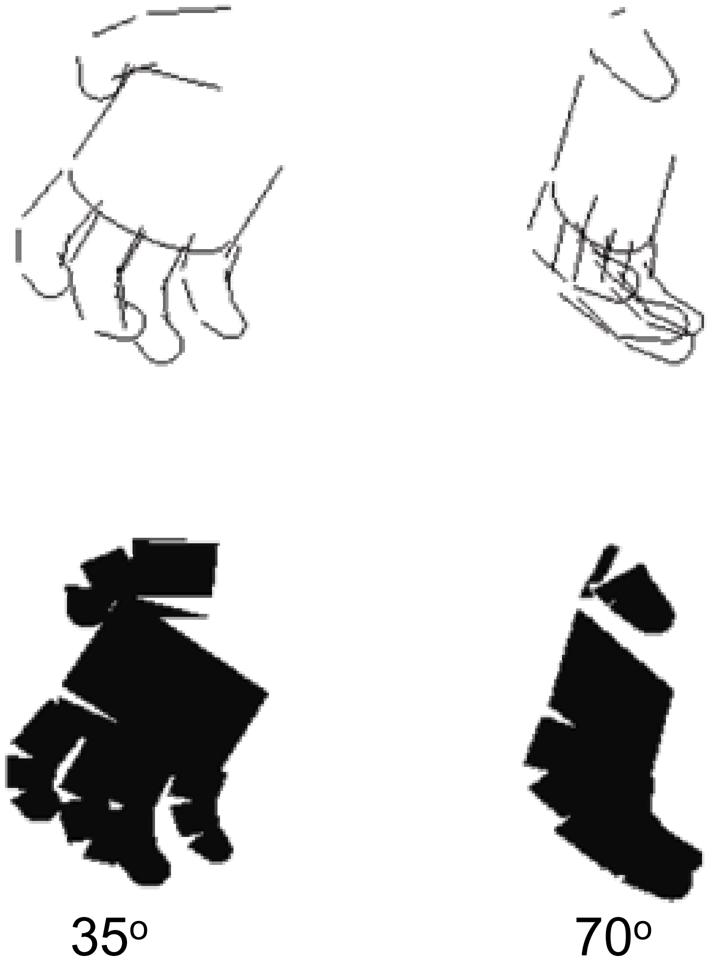

NBP, "1-step"


NBP, "2-step"

Hand model

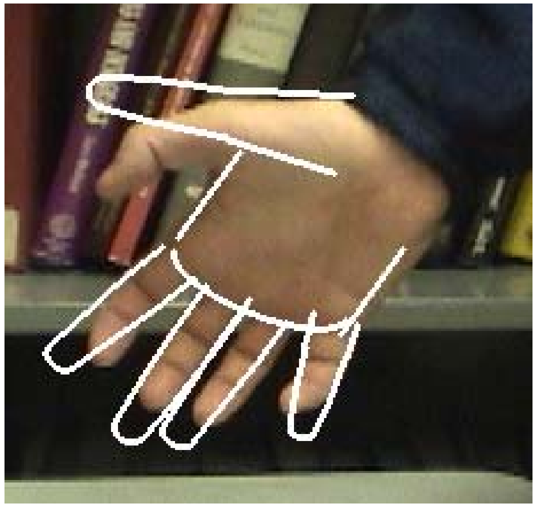




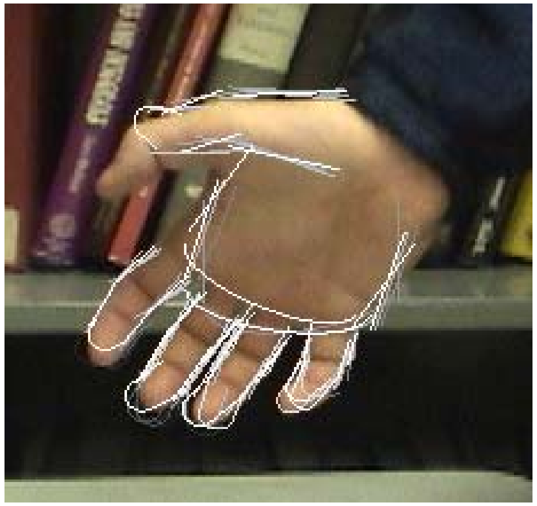


Single-frame Inference

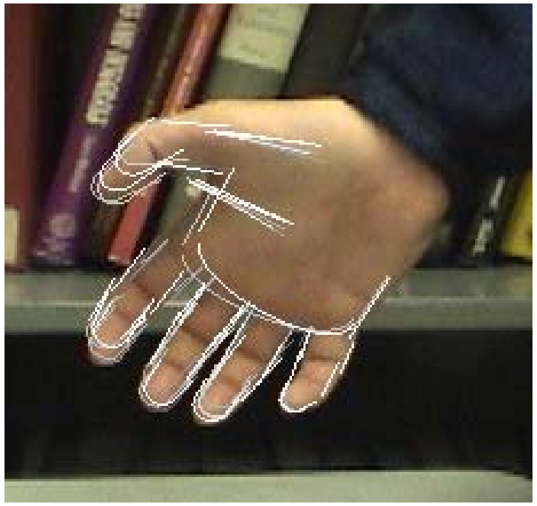
0



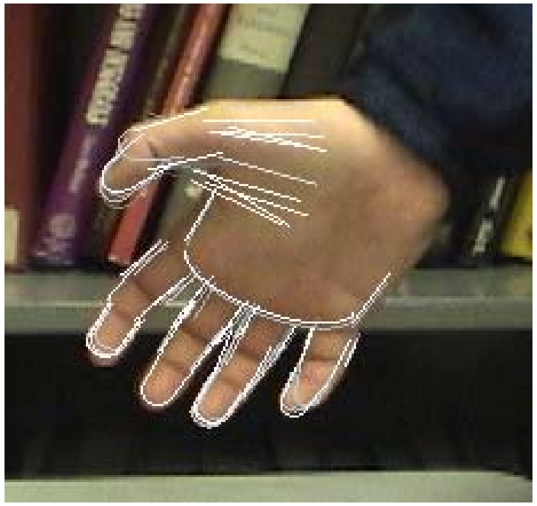
1



2



4



Summary & Ongoing Work

Webpage: *http://ssg.mit.edu/nbp/*

Nonparametric Belief Propagation

- Applicable to general graphs
- Allows highly non-Gaussian interactions
- Multiscale samplers lead to computational efficiency

Applications

- Sensor networks & distributed systems
- Computer vision applications

Code

- Kernel density estimation code (KDE Toolbox)
- More NBP code upcoming…

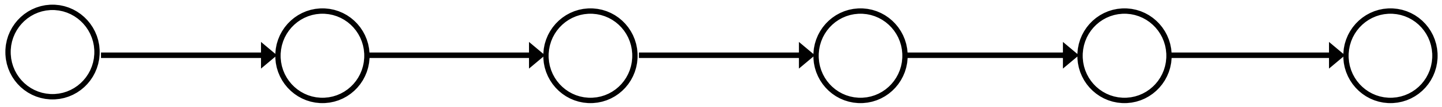
Multi-Target Tracking

Assumptions

- Receive noisy estimates of position of multiple targets
- Also receive spurious observations (outliers)
- Targets indistinguishable based on observations

$\longrightarrow$ *Must use temporal correlations to resolve ambiguities*

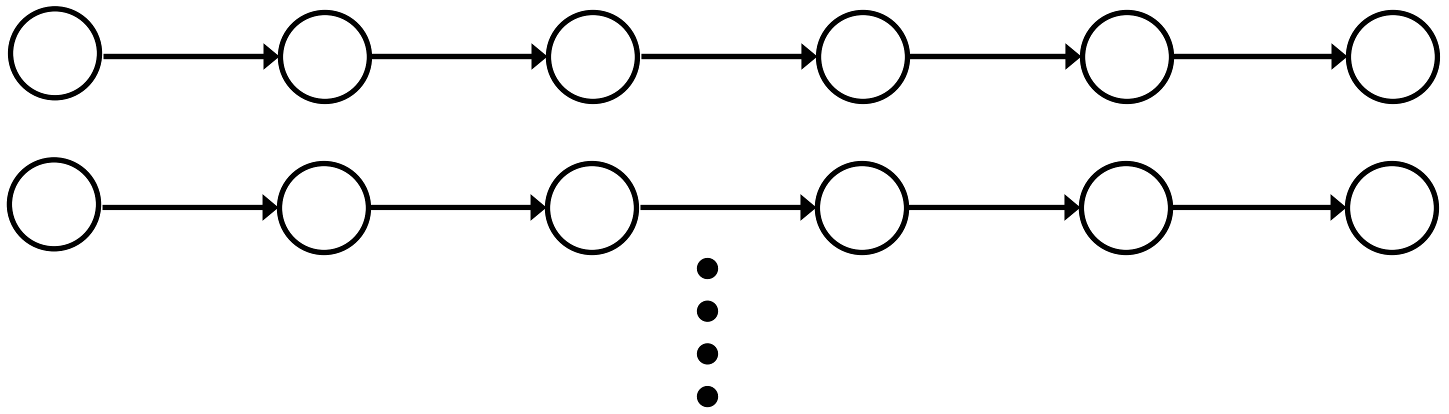
Standard Approach: Particle Filter / Smoother



- *State:* joint configuration of all targets
- *Advantages:* allows complex data association rules
- *Problems:* grows exponentially with number of targets

Graphical Models for Tracking

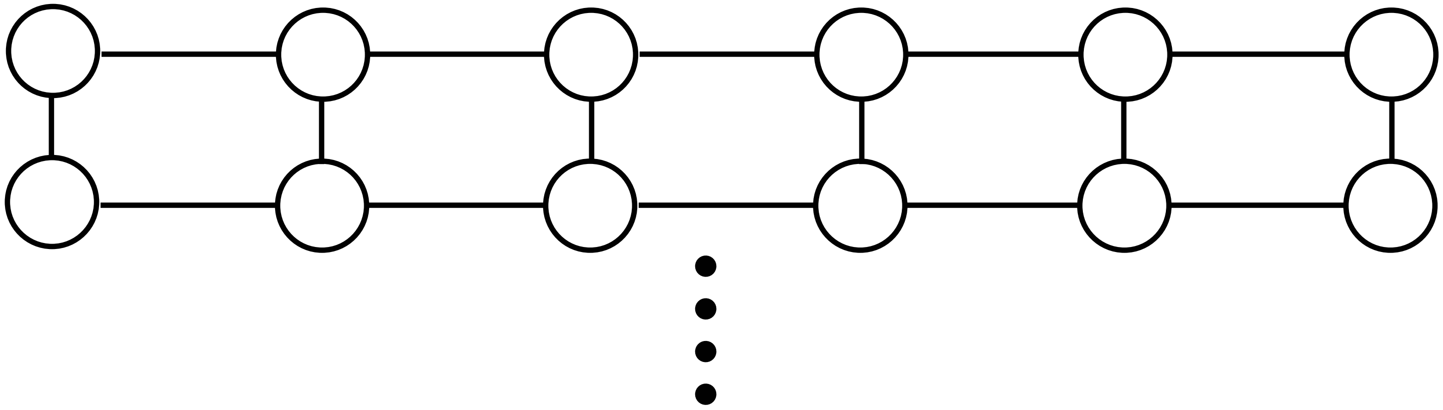
Multiple Independent Smoothers



- *State:* independent Markov chain for each target
- *Advantages:* grows linearly with number of targets
- *Problems:* solutions degenerate to follow best target

Graphical Models for Tracking

Multiple Dependent Smoothers



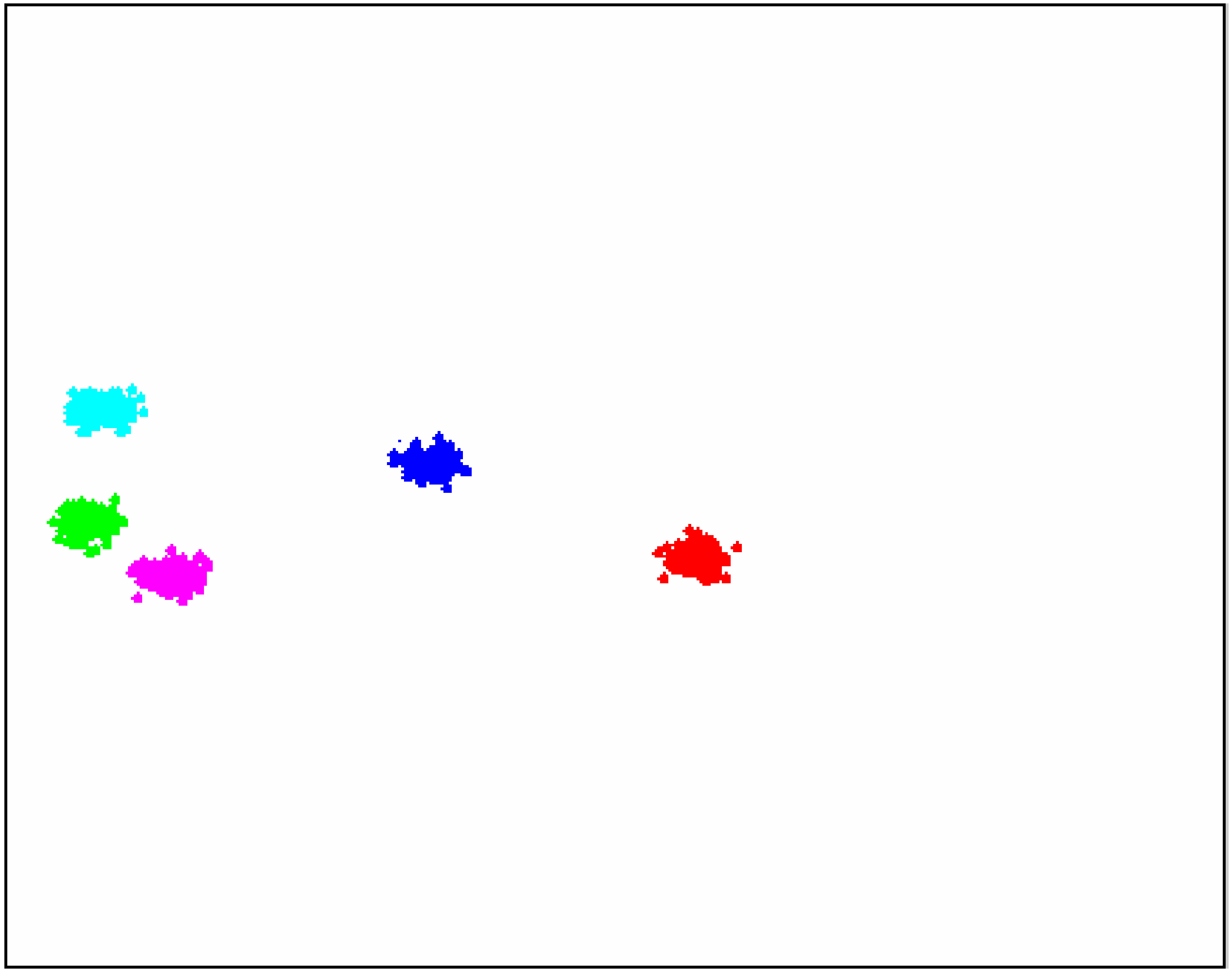
- *State:* Markov chain for each target, where states of different chains are coupled by a repulsive constraint:

  $\longrightarrow$ *Analogous to sensor network potentials for missing distance measurements*

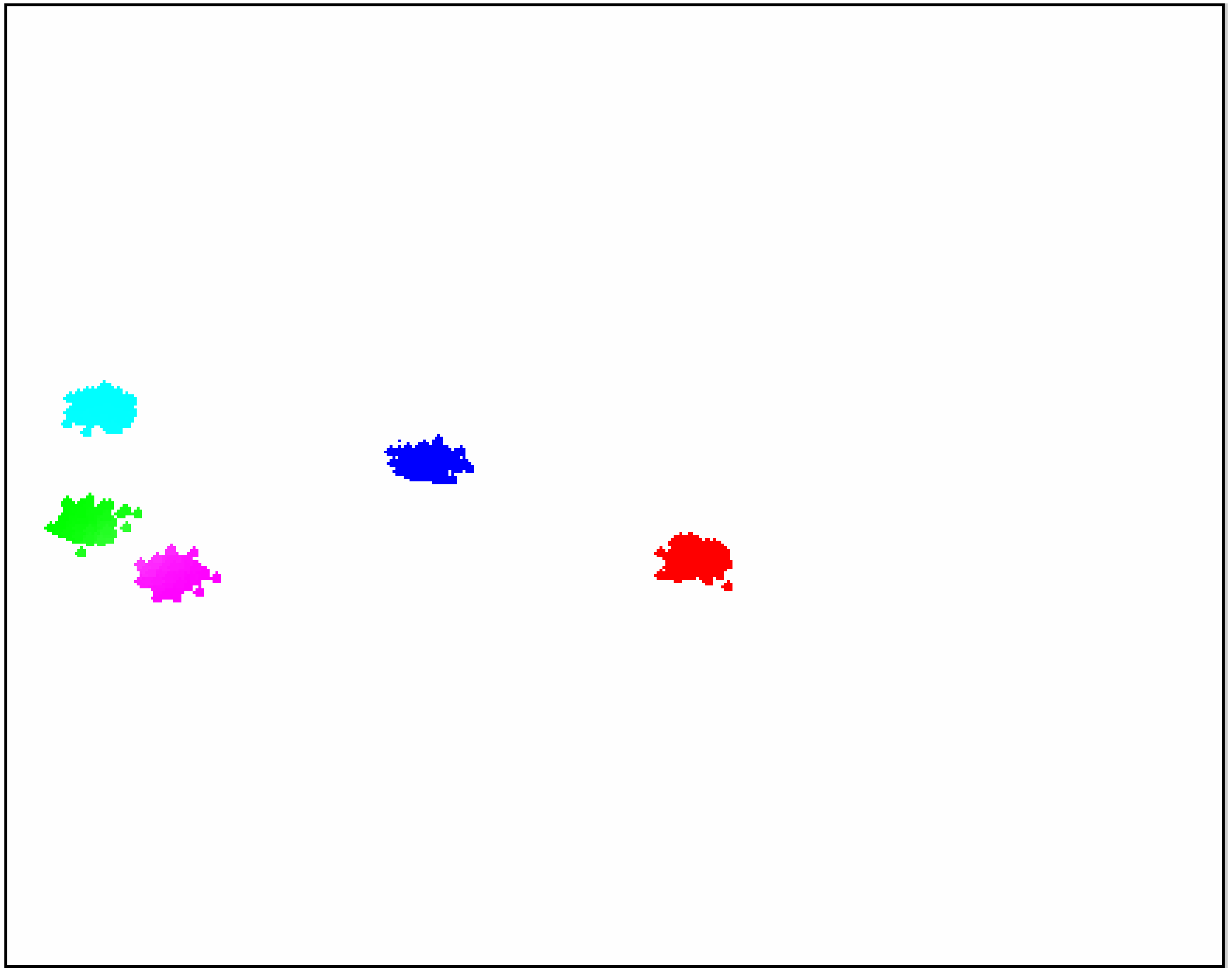
- *Advantages:* storage & computation (NBP) grow linearly

- *Problems (??):* replaces strict data association rule by a prior model on the state space (objects do not overlap)

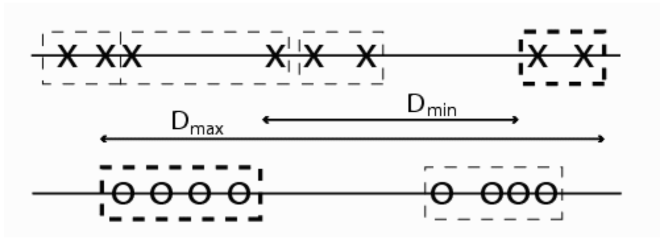
Independent Trackers



Dependent (NBP) Trackers



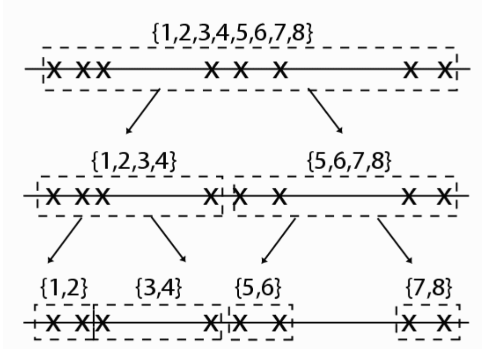
ε-Exact Sampling



- Use bounding box statistics
  - Bounds on pairwise distances
  - Approximate kernel density evaluation [Gray03]:
    - Intuition: find sets of points which have nearly equal contributions
    - Provides evaluations within fractional error ε:

      If not within ε, move down the KD-tree (smaller regions = better bounds)
- Apply to exact sampling algorithm:
  - Can write weight equation in terms of density pairs
    - Estimate normalization (sum of all weights) Z
    - Draw & sort uniform random variables
    - Find their corresponding labels
  - Tunable accuracy level:

$$|\widehat{p}_L - p_L| = \left| \frac{\widehat{w}_L}{\widehat{Z}} - \frac{w_L}{Z} \right| \le \epsilon$$



{1,2,3,4,5,6,7,8}
{1,2,3,4}
{5,6,7,8}
{1,2}
{3,4}
{5,6}
{7,8}