Graphical models, belief propagation, and Markov random fields

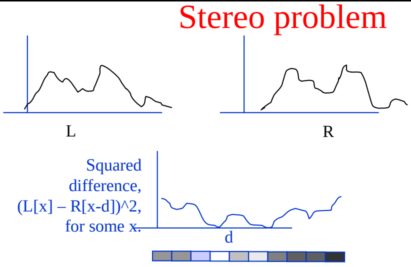
Bill Freeman, MIT
Fredo Durand, MIT

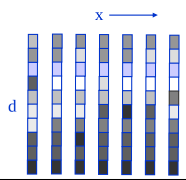
6.882 March 21, 2005

Color selection problem

- (see Photoshop demonstration)

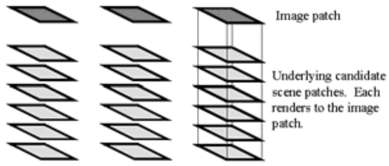
Stereo problem



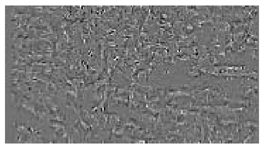


Showing local disparity evidence vectors for a set of neighboring positions, x.

Super-resolution image synthesis



How select which selection of high resolution patches best fits together? Ignoring which patch fits well with which gives this result for the high frequency components of an image:

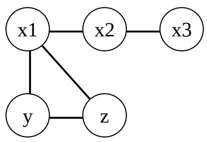


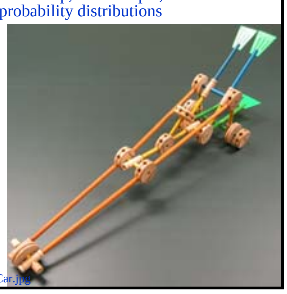
Things we want to be able to articulate in a spatial prior

- Favor neighboring pixels having the same state (state, meaning: estimated depth, or group segment membership)
- Favor neighboring nodes have compatible states (a patch at node i should fit well with selected patch at node j).
- But encourage state changes to occur at certain places (like regions of high image gradient).

Graphical models: tinker toys to build complex probability distributions

- Circles represent random variables.
- Lines represent statistical dependencies.
- There is a corresponding equation that gives P(x1, x2, x3, y, z), but often it's easier to understand things from the picture.
- These tinker toys for probabilities let you build up, from simple, easy-to-understand pieces, complicated probability distributions involving many variables.





http://mark.michaelis.net/weblog/2002/12/29/Tinker%20Toys%20Car.jpg

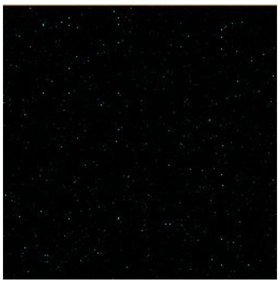
Steps in building and using graphical models

- First, define the function you want to optimize. Note the two common ways of framing the problem
  - In terms of probabilities. Multiply together component terms, which typically involve exponentials.
  - In terms of energies. The log of the probabilities. Typically add together the exponentiated terms from above.
- The second step: optimize that function. For probabilities, take the mean or the max (or use some other “loss function”). For energies, take the min.
- 3rd step: in many cases, you want to learn the function from the 1st step.

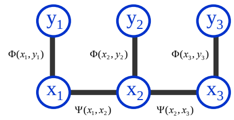
Define model parameters

$$\begin{pmatrix} 1 & \alpha & \alpha \\ \alpha & 1 & \alpha \\ \alpha & \alpha & 1 \end{pmatrix}$$

A more general compatibility matrix (values shown as grey scale)



Derivation of belief propagation

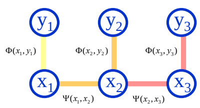


$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \underset{x_2}{\text{sum}}\ \underset{x_3}{\text{sum}}\ P(x_1, x_2, x_3, y_1, y_2, y_3)$$

The posterior factorizes

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \underset{x_2}{\text{sum}}\ \underset{x_3}{\text{sum}}\ P(x_1, x_2, x_3, y_1, y_2, y_3)$$

$$= \underset{x_1}{\text{mean}}\ \underset{x_2}{\text{sum}}\ \underset{x_3}{\text{sum}}\ \Phi(x_1, y_1)\ \Phi(x_2, y_2)\ \Psi(x_1, x_2)\ \Phi(x_3, y_3)\ \Psi(x_2, x_3)$$

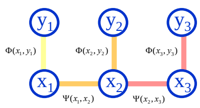


Propagation rules

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \underset{x_2}{\text{sum}}\ \underset{x_3}{\text{sum}}\ P(x_1, x_2, x_3, y_1, y_2, y_3)$$

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \underset{x_2}{\text{sum}}\ \underset{x_3}{\text{sum}}\ \Phi(x_1, y_1)\ \Phi(x_2, y_2)\ \Psi(x_1, x_2)\ \Phi(x_3, y_3)\ \Psi(x_2, x_3)$$

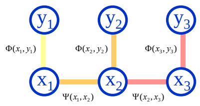
$$x_{1MMSE} = \underset{x_1}{\text{mean}}\ \Phi(x_1, y_1)\ \underset{x_2}{\text{sum}}\ \Phi(x_2, y_2)\ \Psi(x_1, x_2)\ \underset{x_3}{\text{sum}}\ \Phi(x_3, y_3)\ \Psi(x_2, x_3)$$



Propagation rules

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\, \Phi(x_1, y_1)\ \underset{x_2}{\text{sum}}\, \Phi(x_2, y_2)\, \Psi(x_1, x_2)\ \underset{x_3}{\text{sum}}\, \Phi(x_3, y_3)\, \Psi(x_2, x_3)$$

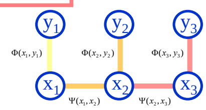
$$M_1^2(x_1) = \underset{x_2}{\text{sum}}\ \Psi(x_1, x_2)\ \Phi(x_2, y_2)\ M_2^3(x_2)$$



Propagation rules

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\, \Phi(x_1, y_1)\ \boxed{\underset{x_2}{\text{sum}}\, \Phi(x_2, y_2)\, \Psi(x_1, x_2)}\ \boxed{\underset{x_3}{\text{sum}}\, \Phi(x_3, y_3)\, \Psi(x_2, x_3)}$$

$$\boxed{M_1^2(x_1)} = \underset{x_2}{\text{sum}}\ \Psi(x_1, x_2)\ \Phi(x_2, y_2)\ \boxed{M_2^3(x_2)}$$



Belief propagation: the nosey neighbor rule

"Given everything that I know, here's what I think you should think"

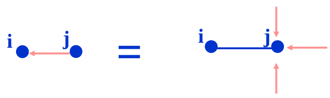
(Given the probabilities of my being in different states, and how my states relate to your states, here's what I think the probabilities of your states should be)

Belief propagation messages

A message: can be thought of as a set of weights on each of your possible states

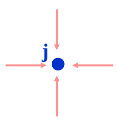
To send a message: Multiply together all the incoming messages, except from the node you're sending to, then multiply by the compatibility matrix and marginalize over the sender's states.

$$M_i^j(x_i) = \sum_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in N(j) \setminus i} M_j^k(x_j)$$



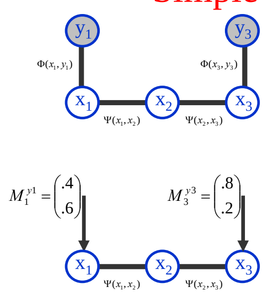
Beliefs

To find a node's beliefs: Multiply together all the messages coming in to that node.



$$b_j(x_j) = \prod_{k \in N(j)} M_j^k(x_j)$$

Simple BP example

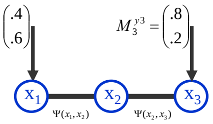


$$\Psi(x_1, x_2) = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}$$

$$\Psi(x_2, x_3) = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}$$

Simple BP example

$M_1^{y1} = \begin{pmatrix} .4 \\ .6 \end{pmatrix}$ $\quad M_3^{y3} = \begin{pmatrix} .8 \\ .2 \end{pmatrix}$



$$\Psi(x_1, x_2) = \Psi(x_2, x_3) = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}$$

To find the marginal probability for each variable, you can

(a) Marginalize out the other variables of:

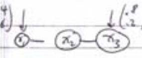
$$P(x_1, x_2, x_3) = \Psi(x_1, x_2)\Psi(x_2, x_3)M_1^{y1}(x_1)M_3^{y3}(x_3)$$

(b) Or you can run belief propagation, (BP). BP redistributes the various partial sums, leading to a very efficient calculation.

---

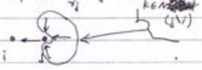
local evidence $\begin{pmatrix} .4 \\ .6 \end{pmatrix}$ $\quad \begin{pmatrix} .8 \\ .2 \end{pmatrix}$ $\quad \psi_{12} = \psi_{23} = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}$

Bill Freeman
March 22, 2006



$$M_i^j(x_i) = \sum_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in N(j \backslash i)} M_j^k(x_j)$$

all incoming messages except that from node i



$$b_j(x_j) = \prod_{k = N(j)} M_j^k(x_j) \quad \Phi(x_j)$$

all incoming messages

any local potential term (none here)

iteration n	$M_1^{y_1}$	M_2^1	M_3^2	$M_3^{y_3}$	M_2^3	M_1^2	b_1	b_2	b_3
	messages						beliefs		
n=0	.4 .6	1 1	1 1	.8 .2	1 1	1 1	.4 .6	.5 .5	.8 .2
n=1	.4 .6	.42 .58	1 1	.8 .2	.74 .26	1 1	.4 .6	.67 .33	.8 .2
n=2	.4 .6	.42 .58	.44 .56	.8 .2	.74 .26	.69 .31	.6 .4	.67 .33	.76 .24

---

calculations

n=1
$$M_2^1 = \sum_{x_1} \psi_{12}(x_1, x_2) M_1^{y_1}(x_1) = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}\begin{pmatrix} .4 \\ .6 \end{pmatrix} = \begin{pmatrix} .42 \\ .58 \end{pmatrix}$$
(the matrix multiply does the sum)

$$M_2^3 = \sum_{x_3} \psi_{32}(x_3, x_2) M_3^{y_3}(x_3) = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}\begin{pmatrix} .8 \\ .2 \end{pmatrix} = \begin{matrix} .72 + .02 \\ .08 + .18 \end{matrix} = \begin{pmatrix} .74 \\ .26 \end{pmatrix}$$

n=1
$$b_2 = \begin{pmatrix} .42 \\ .58 \end{pmatrix} .* \begin{pmatrix} .74 \\ .26 \end{pmatrix} = \begin{pmatrix} .67 \\ .33 \end{pmatrix}$$
(normalized to sum to 1)

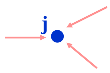
n=2
$$M_3^2 = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}\begin{pmatrix} .42 \\ .58 \end{pmatrix} = \begin{pmatrix} .436 \\ .564 \end{pmatrix}$$

n=2
$$M_1^2 = \begin{pmatrix} .9 & .1 \\ .1 & .9 \end{pmatrix}\begin{pmatrix} .74 \\ .26 \end{pmatrix} = \begin{pmatrix} .692 \\ .308 \end{pmatrix}$$

n=2
$$b_1 = \begin{pmatrix} .4 \\ .6 \end{pmatrix} .* \begin{pmatrix} .69 \\ .31 \end{pmatrix} = \begin{pmatrix} .6 \\ .4 \end{pmatrix}$$
(normalized)

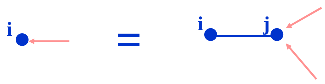
n=2
$$b_3 = \begin{pmatrix} .8 \\ .2 \end{pmatrix} .* \begin{pmatrix} .44 \\ .56 \end{pmatrix} = \begin{pmatrix} .76 \\ .24 \end{pmatrix}$$
normalized

Belief, and message updates



$$b_j(x_j) = \prod_{k \in N(j)} M_j^k(x_j)$$

$$M_i^j(x_i) = \sum_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in N(j) \backslash i} M_j^k(x_j)$$



Optimal solution in a chain or tree: Belief Propagation

- “Do the right thing” Bayesian algorithm.
- For Gaussian random variables over time: Kalman filter.
- For hidden Markov models: forward/backward algorithm (and MAP variant is Viterbi).

Making probability distributions modular, and therefore tractable:

Probabilistic graphical models

Vision is a problem involving the interactions of many variables: things can seem hopelessly complex. Everything is made tractable, or at least, simpler, if we modularize the problem. That’s what probabilistic graphical models do, and let’s examine that.

Readings: Jordan and Weiss intro article—fantastic!
Kevin Murphy web page—comprehensive and with pointers to many advanced topics

A toy example

Suppose we have a system of 5 interacting variables, perhaps some are observed and some are not. There's some probabilistic relationship between the 5 variables, described by their joint probability, P(x1, x2, x3, x4, x5).

If we want to find out what the likely state of variable x1 is (say, the position of the hand of some person we are observing), what can we do?

Two reasonable choices are: (a) find the value of x1 (and of all the other variables) that gives the maximum of P(x1, x2, x3, x4, x5); that's the MAP solution.
Or (b) marginalize over all the other variables and then take the mean or the maximum of the other variables. Marginalizing, then taking the mean, is equivalent to finding the MMSE solution. Marginalizing, then taking the max, is called the max marginal solution and sometimes a useful thing to do.

To find the marginal probability at x1, we have to take this sum:

$$\sum_{x_2,x_3,x_4,x_5} P(x_1,x_2,x_3,x_4,x_5)$$

If the system really is high dimensional, that will quickly become intractable. But if there is some modularity in $P(x_1,x_2,x_3,x_4,x_5)$ then things become tractable again.

Suppose the variables form a Markov chain: x1 causes x2 which causes x3, etc. We might draw out this relationship as follows:

$$x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5$$

P(a,b) = P(b|a) P(a)

By the chain rule, for any probability distribution, we have:

$$P(x_1,x_2,x_3,x_4,x_5) = P(x_1)P(x_2,x_3,x_4,x_5 \mid x_1)$$
$$= P(x_1)P(x_2 \mid x_1)P(x_3,x_4,x_5 \mid x_1,x_2)$$
$$= P(x_1)P(x_2 \mid x_1)P(x_3 \mid x_1,x_2)P(x_4,x_5 \mid x_1,x_2,x_3)$$
$$= P(x_1)P(x_2 \mid x_1)P(x_3 \mid x_1,x_2)P(x_4 \mid x_1,x_2,x_3)P(x_5 \mid x_1,x_2,x_3,x_4)$$

But if we exploit the assumed modularity of the probability distribution over the 5 variables (in this case, the assumed Markov chain structure), then that expression simplifies:

$$= P(x_1)P(x_2 \mid x_1)P(x_3 \mid x_2)P(x_4 \mid x_3)P(x_5 \mid x_4)$$

$$x_1 \rightarrow x_2 \rightarrow x_3 \rightarrow x_4 \rightarrow x_5$$

Now our marginalization summations distribute through those terms:

$$\sum_{x_2,x_3,x_4,x_5} P(x_1,x_2,x_3,x_4,x_5) = P(x_1)\sum_{x_2} P(x_2 \mid x_1)\sum_{x_3} P(x_3 \mid x_2)\sum_{x_4} P(x_4 \mid x_3)\sum_{x_5} P(x_5 \mid x_4)$$

Belief propagation

Performing the marginalization by doing the partial sums is called "belief propagation".

$$\sum_{x_2,x_3,x_4,x_5} P(x_1,x_2,x_3,x_4,x_5) = P(x_1)\sum_{x_2} P(x_2 \mid x_1)\sum_{x_3} P(x_3 \mid x_2)\sum_{x_4} P(x_4 \mid x_3)\sum_{x_5} P(x_5 \mid x_4)$$

In this example, it has saved us a lot of computation. Suppose each variable has 10 discrete states. Then, not knowing the special structure of P, we would have to perform 10000 additions (10^4) to marginalize over the four variables.
But doing the partial sums on the right hand side, we only need 40 additions (10*4) to perform the same marginalization!

Another modular probabilistic structure, more common in vision problems, is an undirected graph:

$$x_1 - x_2 - x_3 - x_4 - x_5$$

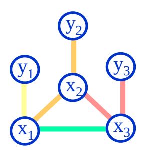
The joint probability for this graph is given by:

$$P(x_1,x_2,x_3,x_4,x_5) = \Phi(x_1,x_2)\Phi(x_2,x_3)\Phi(x_3,x_4)\Phi(x_4,x_5)$$

Where $\Phi(x_1,x_2)$ is called a "compatibility function". We can define compatibility functions we result in the same joint probability as for the directed graph described in the previous slides; for that example, we could use either form.

No factorization with loops!

$$x_{1MMSE} = \underset{x_1}{\text{mean}}\, \Phi(x_1,y_1)$$
$$\underset{x_2}{\text{sum}}\, \Phi(x_2,y_2)\,\Psi(x_1,x_2)$$
$$\underset{x_3}{\text{sum}}\, \Phi(x_3,y_3)\,\Psi(x_2,x_3)\;\Psi(x_1,x_3)$$

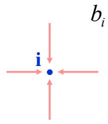


Justification for running belief propagation in networks with loops

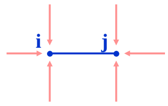
- Experimental results:
  - Error-correcting codes — Kschischang and Frey, 1998; McEliece et al., 1998
  - Vision applications — Freeman and Pasztor, 1999; Frey, 2000
- Theoretical results:
  - For Gaussian processes, means are correct. — Weiss and Freeman, 1999
  - Large neighborhood local maximum for MAP. — Weiss and Freeman, 2000
  - Equivalent to Bethe approx. in statistical physics. — Yedidia, Freeman, and Weiss, 2000
  - Tree-weighted reparameterization — Wainwright, Willsky, Jaakkola, 2001

Region marginal probabilities

$$b_i(x_i) \quad = k\,\Phi(x_i) \prod_{k \in N(i)} M_i^k(x_i)$$

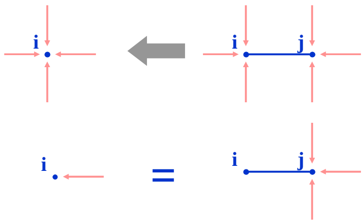


$$b_{ij}(x_i, x_j) = k\,\Psi(x_i, x_j) \prod_{k \in N(i)\backslash j} M_i^k(x_i) \prod_{k \in N(j)\backslash i} M_j^k(x_j)$$



Belief propagation equations

Belief propagation equations come from the marginalization constraints.



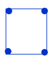
$$M_i^j(x_i) = \sum_{x_j} \psi_{ij}(x_i, x_j) \prod_{k \in N(j)\backslash i} M_j^k(x_j)$$

Results from Bethe free energy analysis

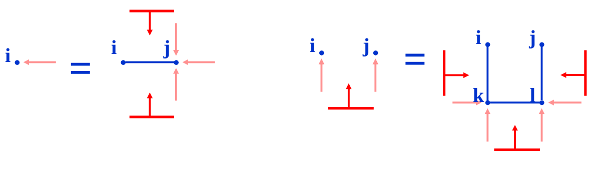
- Fixed point of belief propagation equations iff. Bethe approximation stationary point.
- Belief propagation always has a fixed point.
- Connection with variational methods for inference: both minimize approximations to Free Energy,
  - **variational**: usually use primal variables.
  - **belief propagation**: fixed pt. equs. for dual variables.
- Kikuchi approximations lead to more accurate belief propagation algorithms.
- Other Bethe free energy minimization algorithms—Yuille, Welling, etc.

Kikuchi message-update rules

Groups of nodes send messages to other groups of nodes.



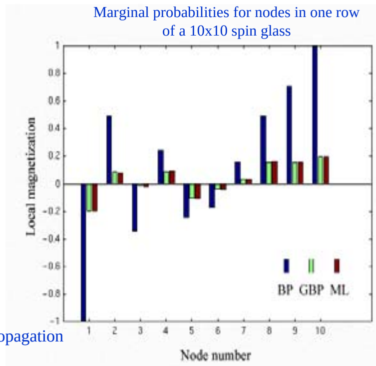
Typical choice for Kikuchi cluster.



Update for messages

Update for messages

Generalized belief propagation



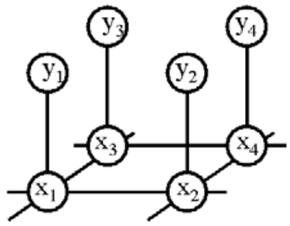
BP: belief propagation
GBP: generalized belief propagation
ML: maximum likelihood

References on BP and GBP

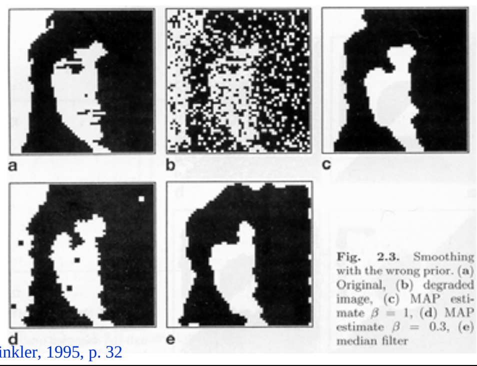
- J. Pearl, 1985
  - classic
- Y. Weiss, NIPS 1998
  - Inspires application of BP to vision
- W. Freeman et al learning low-level vision, IJCV 1999
  - Applications in super-resolution, motion, shading/paint discrimination
- H. Shum et al, ECCV 2002
  - Application to stereo
- M. Wainwright, T. Jaakkola, A. Willsky
  - Reparameterization version
- J. Yedidia, AAAI 2000
  - The clearest place to read about BP and GBP.

Probability models for entire images: Markov Random Fields

- Allows rich probabilistic models for images.
- But built in a local, modular way. Learn local relationships, get global effects out.

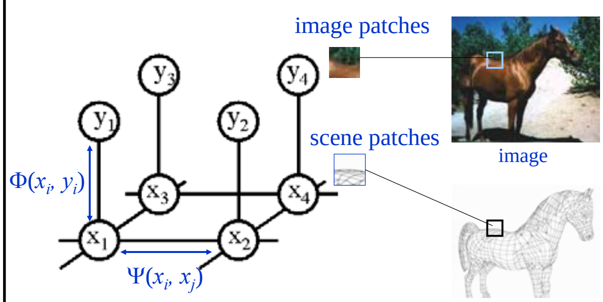


MRF nodes as pixels



Winkler, 1995, p. 32

MRF nodes as patches



Network joint probability

$$P(x,y) = \frac{1}{Z}\prod_{i,j}\Psi(x_i, x_j)\prod_i \Phi(x_i, y_i)$$

scene

image

Scene-scene compatibility function

neighboring scene nodes

Image-scene compatibility function

local observations

In order to use MRFs:

- Given observations y, and the parameters of the MRF, how infer the hidden variables, x?
- How learn the parameters of the MRF?

Outline of MRF section

- Inference in MRF’s.
  - Iterated conditional modes (ICM)
  - Gibbs sampling, simulated annealing
  - Variational methods
  - Belief propagation
  - Graph cuts
- Vision applications of inference in MRF’s.
- Learning MRF parameters.
  - Iterative proportional fitting (IPF)

Iterated conditional modes

- For each node:
  - Condition on all the neighbors
  - Find the mode
  - Repeat.

Described in: Winkler, 1995. Introduced by Besag in 1986.

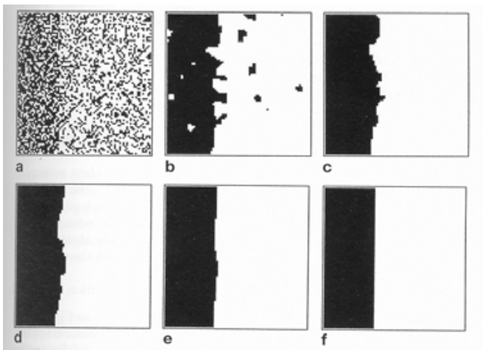


Fig. 6.2. Various steps of ICM

Winkler, 1995

Outline of MRF section

- Inference in MRF’s.
  - Iterated conditional modes (ICM)
  - Gibbs sampling, simulated annealing
  - Variational methods
  - Belief propagation
  - Graph cuts
- Vision applications of inference in MRF’s.
- Learning MRF parameters.
  - Iterative proportional fitting (IPF)

Gibbs Sampling and Simulated Annealing

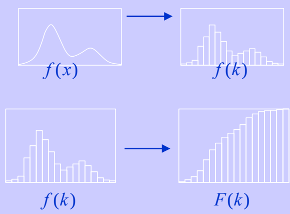
- Gibbs sampling:
  - A way to generate random samples from a (potentially very complicated) probability distribution.
- Simulated annealing:
  - A schedule for modifying the probability distribution so that, at “zero temperature”, you draw samples only from the MAP solution.

$$P(x) = \frac{1}{Z}\exp(-E(x)/kT)$$

Reference: Geman and Geman, IEEE PAMI 1984.

Sampling from a 1-d function

1. Discretize the density function



2. Compute distribution function from density function

3. Sampling

draw $\alpha \sim$ U(0,1);
for k = 1 to n
if $F(k) \geq \alpha$
break;
$x = x_0 + k\tau$;

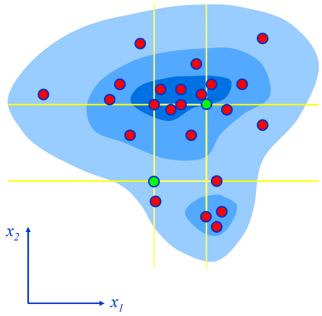
Gibbs Sampling

$$x_1^{(t+1)} \sim \pi(x_1 \mid x_2^{(t)}, x_3^{(t)}, \cdots, x_K^{(t)})$$

$$x_2^{(t+1)} \sim \pi(x_2 \mid x_1^{(t+1)}, x_3^{(t)}, \cdots, x_K^{(t)})$$

$$\vdots$$

$$x_K^{(t+1)} \sim \pi(x_K \mid x_1^{(t+1)}, \cdots, x_{K-1}^{(t+1)})$$



Slide by Ce Liu

Gibbs sampling and simulated annealing

Simulated annealing as you gradually lower the "temperature" of the probability distribution ultimately giving zero probability to all but the MAP estimate.

What's good about it: finds global MAP solution.

What's bad about it: takes forever. Gibbs sampling is in the inner loop…

Gibbs sampling and simulated annealing

So you can find the mean value (MMSE estimate) of a variable by doing Gibbs sampling and averaging over the values that come out of your sampler.

You can find the MAP value of a variable by doing Gibbs sampling and gradually lowering the temperature parameter to zero.

Outline of MRF section

- Inference in MRF's.
  - Iterated conditional modes (ICM)
  - Gibbs sampling, simulated annealing
  - Variational methods
  - Belief propagation
  - Graph cuts
- Vision applications of inference in MRF's.
- Learning MRF parameters.
  - Iterative proportional fitting (IPF)

Variational methods

- Reference: Tommi Jaakkola's tutorial on variational methods,
  http://www.ai.mit.edu/people/tommi/
- Example: mean field
  - For each node
    - Calculate the expected value of the node, conditioned on the mean values of the neighbors.

Outline of MRF section

- Inference in MRF's.
  - Iterated conditional modes (ICM)
  - Gibbs sampling, simulated annealing
  - Variational methods
  - Belief propagation
  - Graph cuts
- Vision applications of inference in MRF's.
- Learning MRF parameters.
  - Iterative proportional fitting (IPF)

Outline of MRF section

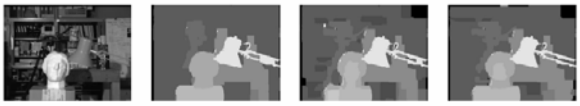
- Inference in MRF’s.
  - Iterated conditional modes (ICM)
  - Gibbs sampling, simulated annealing
  - Variational methods
  - Belief propagation
  - Graph cuts
- Vision applications of inference in MRF’s.
- Learning MRF parameters.
  - Iterative proportional fitting (IPF)

Graph cuts

- Algorithm: uses node label swaps or expansions as moves in the algorithm to reduce the energy. Swaps many labels at once, not just one at a time, as with ICM.
- Find which pixel labels to swap using min cut/max flow algorithms from network theory.
- Can offer bounds on optimality.
- See Boykov, Veksler, Zabih, IEEE PAMI 23 (11) Nov. 2001 (available on web).

Comparison of graph cuts and belief propagation

Comparison of Graph Cuts with Belief Propagation for Stereo, using Identical MRF Parameters, ICCV 2003.
Marshall F. Tappen William T. Freeman



(a) Tsukuba Image (b) Graph Cuts (c) Synchronous BP (d) Accelerated BP

Figure 3. Results produced by the three algorithms on the Tsukuba image. The parameters used to generate this field were $s = 50, T = 4, P = 2$. Again, Graph Cuts produces a much smoother solution. Belief Propagation does maintain some structures that are lost in the Graph Cuts solution, such as the camera and the face in the foreground.

Ground truth, graph cuts, and belief propagation disparity solution energies

Image	Energy of MRF Labelling Returned ($\times 10^3$)			% Energy from Occluded Matching Costs
	Ground-Truth	Graph Cuts	Synchronous Belief Prop	
Map	757	383	442	61%
Sawtooth	6591	1652	1713	79%
Tsukuba	1852	663	775	61%
Venus	5739	1442	1501	76%

Figure 2. Field Energies for the MRF labelled using ground-truth data compared to the energies for the fields labelled using Graph Cuts and Belief Propagation. Notice that the solutions returned by the algorithms consistently have a much lower energy than the labellings produced from the ground-truth, showing a mismatch between the MRF formulation and the ground-truth. The final column contains the percentage of each ground-truth solution's energy that comes from matching costs of occluded pixels.

Graph cuts versus belief propagation

- Graph cuts consistently gave slightly lower energy solutions for that stereo-problem MRF, although BP ran faster, although there is now a faster graph cuts implementation than what we used…
- However, here’s why I still use Belief Propagation:
  - Works for any compatibility functions, not a restricted set like graph cuts.
  - I find it very intuitive.
  - Extensions: sum-product algorithm computes MMSE, and Generalized Belief Propagation gives you very accurate solutions, at a cost of time.

MAP versus MMSE

(a) MAP Estimate (b) MMSE Estimate

Figure 7. Comparison of MAP and MMSE estimates on a different MRF formulation. The MAP estimate chooses the most likely discrete disparity level for each point, resulting in a depth-map with stair-stepping effects. Using the MMSE estimate assigns sub-pixel disparities, resulting in a smooth depth map.

Show program comparing some methods on a simple MRF

testMRF.m

Outline of MRF section

- Inference in MRF's.
  - Gibbs sampling, simulated annealing
  - Iterated condtional modes (ICM)
  - Variational methods
  - Belief propagation
  - Graph cuts
- Applications of inference in MRF's.
- Learning MRF parameters.
  - Iterative proportional fitting (IPF)

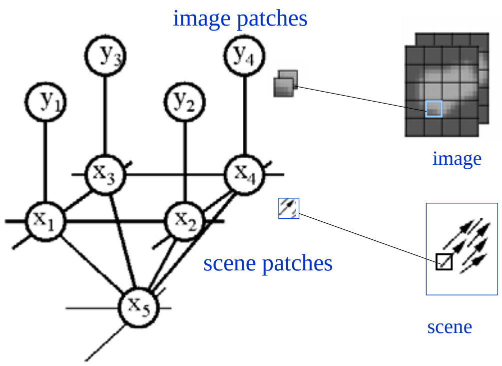
Applications of MRF's

- Stereo
- Motion estimation
- Labelling shading and reflectance
- Many others…

Applications of MRF's

- Stereo
- Motion estimation
- Labelling shading and reflectance
- Many others…

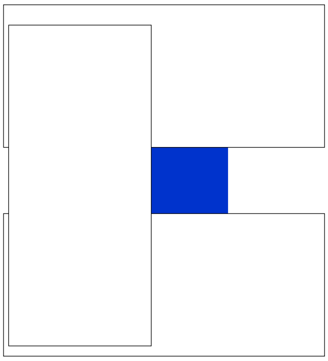
Motion application



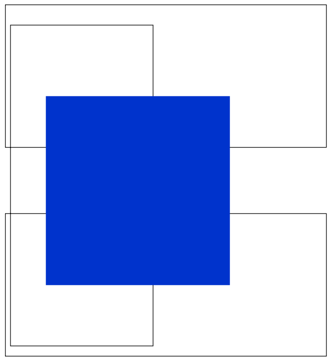
What behavior should we see in a motion algorithm?

- Aperture problem
- Resolution through propagation of information
- Figure/ground discrimination

The aperture problem



The aperture problem



Program demo

Motion analysis: related work

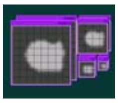
- **Markov network**
  - Luettgen, Karl, Willsky and collaborators.
- **Neural network or learning-based**
  - Nowlan & T. J. Senjowski; Sereno.
- **Optical flow analysis**
  - Weiss & Adelson; Darrell & Pentland; Ju, Black & Jepson; Simoncelli; Grzywacz & Yuille; Hildreth; Horn & Schunk; etc.

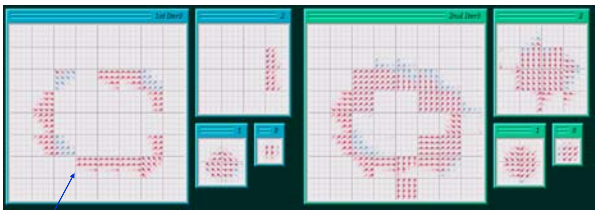
Inference:

Motion estimation results

(maxima of scene probability distributions displayed)

Image data



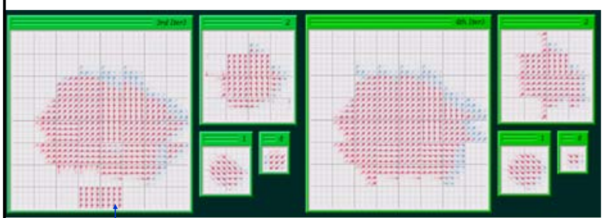


Iterations 0 and 1

Initial guesses only
show motion at edges.

Motion estimation results

(maxima of scene probability distributions displayed)

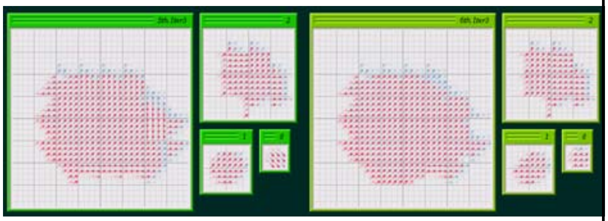


Iterations 2 and 3

Figure/ground still
unresolved here.

Motion estimation results

(maxima of scene probability distributions displayed)



Iterations 4 and 5

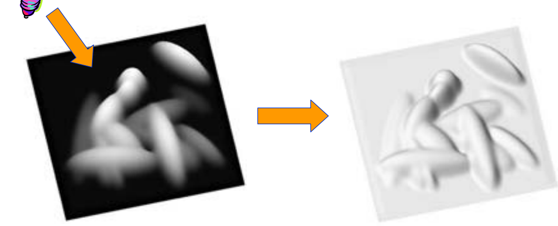
Final result compares well with vector quantized true (uniform) velocities.

Vision applications of MRF's

- Stereo
- Motion estimation
- Labelling shading and reflectance
- Many others…

Forming an Image

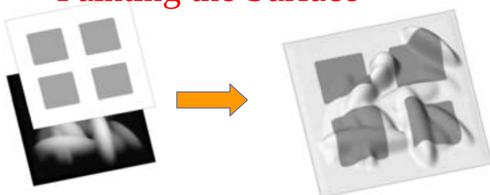
Illuminate the surface to get:



Surface (Height Map) Shading Image

The shading image is the interaction of the shape of the surface and the illumination

Painting the Surface



Scene Image

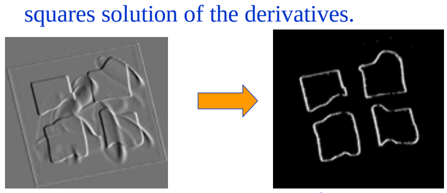
Add a reflectance pattern to the surface. Points inside the squares should reflect less light

Goal

Image Shading Image Reflectance Image

Basic Steps

1. Compute the *x* and *y* image derivatives
2. Classify each derivative as being caused by *either* shading or a reflectance change
3. Set derivatives with the wrong label to zero.
4. Recover the intrinsic images by finding the least-squares solution of the derivatives.



Original *x* derivative image

Classify each derivative (White is reflectance)

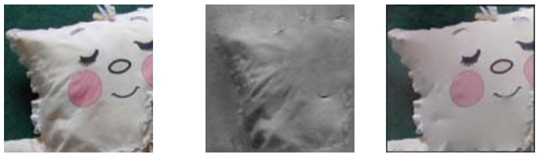
Learning the Classifiers

- Combine multiple classifiers into a strong classifier using AdaBoost (Freund and Schapire)
- Choose weak classifiers greedily similar to (Tieu and Viola 2000)
- Train on synthetic images
- Assume the light direction is from the right

Shading Training Set

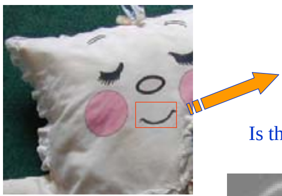
Reflectance Change Training Set

Using Both Color and Gray-Scale Information

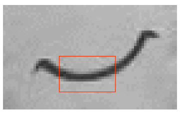


Results without considering gray-scale

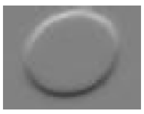
Some Areas of the Image Are Locally Ambiguous



Input



Is the change here better explained as



Shading

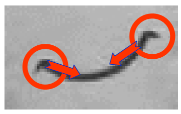
or

Reflectance

?

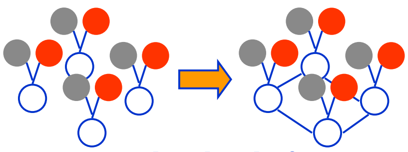
Propagating Information

- Can disambiguate areas by propagating information from reliable areas of the image into ambiguous areas of the image



Propagating Information

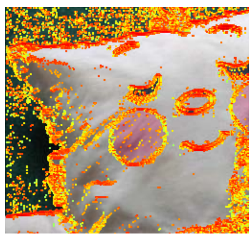
- Consider relationship between neighboring derivatives



- Use Generalized Belief Propagation to infer labels

Setting Compatibilities

- Set compatibilities according to image contours
  - All derivatives along a contour should have the same label
- Derivatives along an image contour strongly influence each other



$\beta=$

0.5 1.0

$$\psi(x_i,x_j)=\begin{bmatrix}1-\beta & \beta\\ \beta & 1-\beta\end{bmatrix}$$

Improvements Using Propagation

Input Image

Reflectance Image Without Propagation

Reflectance Image With Propagation

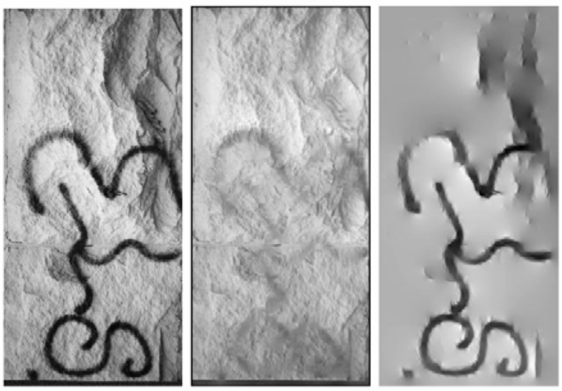
Combining: local evidence from shape and color, and GBP for propagation

(a) Original Image (b) Shading Image (c) Reflectance Image

(More Results)

Input Image Shading Image Reflectance Image

(a) Original Image



(a) Original Image (b) Shape Image (c) Reflectance Image

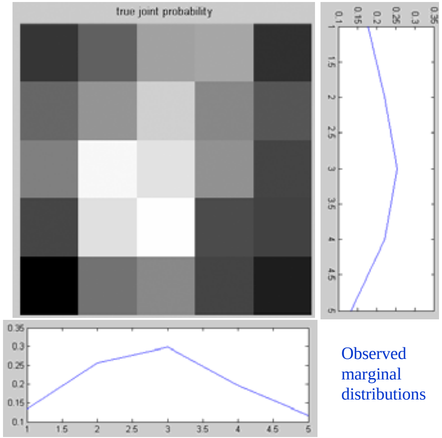
Outline of MRF section

- Inference in MRF’s.
  - Gibbs sampling, simulated annealing
  - Iterated conditional modes (ICM)
  - Variational methods
  - Belief propagation
  - Graph cuts
- Vision applications of inference in MRF’s.
- Learning MRF parameters.
  - Iterative proportional fitting (IPF)

Learning MRF parameters, labeled data

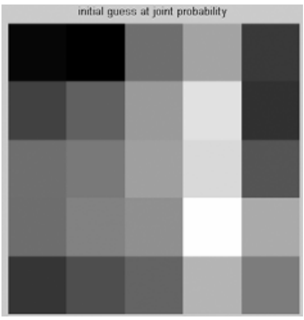
Iterative proportional fitting lets you make a maximum likelihood estimate a joint distribution from observations of various marginal distributions.

True joint probability



Observed marginal distributions

Initial guess at joint probability



IPF update equation

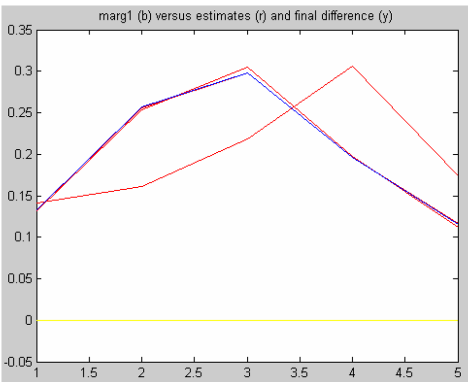
$$P(x_1, x_2, \ldots, x_d)^{(t+1)} = P(x_1, x_2, \ldots, x_d)^{(t)} \frac{P(x_i)^{\text{observed}}}{P(x_i)^{(t)}}$$

Scale the previous iteration's estimate for the joint probability by the ratio of the true to the predicted marginals.

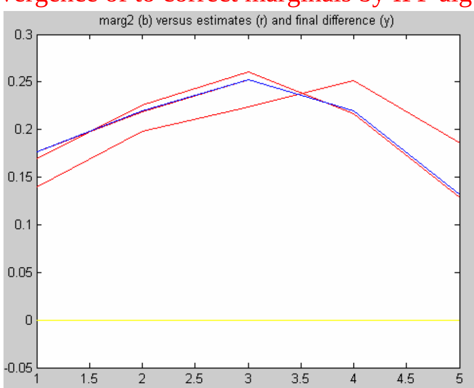
Gives gradient ascent in the likelihood of the joint probability, given the observations of the marginals.

See: Michael Jordan's book on graphical models

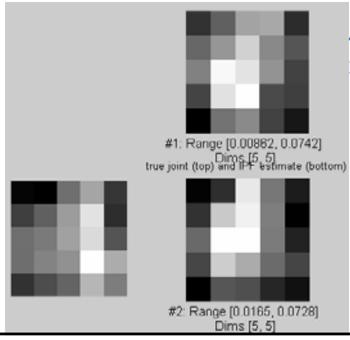
Convergence of to correct marginals by IPF algorithm



Convergence of to correct marginals by IPF algorithm



IPF results for this example: comparison of joint probabilities



Application to MRF parameter estimation

- Can show that for the ML estimate of the clique potentials, $\phi_c(x_c)$, the empirical marginals equal the model marginals,

$$\tilde{p}(x_c) = p(x_c)$$

- This leads to the IPF update rule for $\phi_c(x_c)$

$$\phi_C^{(t+1)}(x_C) = \phi_C^{(t)}(x_C) \frac{\tilde{p}(x_C)}{p^{(t)}(x_C)}$$

- Performs coordinate ascent in the likelihood of the MRF parameters, given the observed data.

Reference: unpublished notes by Michael Jordan

More general graphical models than MRF grids

- In this course, we've studied Markov chains, and Markov random fields, but, of course, many other structures of probabilistic models are possible and useful in computer vision.
- For a nice on-line tutorial about Bayes nets, see Kevin Murphy's tutorial in his web page.

GrabCut

"GrabCut" — Interactive Foreground Extraction using Iterated Graph Cuts

Carsten Rother* Vladimir Kolmogorov† Andrew Blake‡
Microsoft Research Cambridge, UK

Figure 1: **Three examples of GrabCut**. The user drags a rectangle loosely around an object. The object is then extracted automatically.

http://research.microsoft.com/vision/Cambridge/papers/siggraph04.pdf

end