Stat 521A
Lecture 17

Outline

- MAP estimation (13.1)
- Exact methods (13.2-13.3)
- Approx method based on clq graph (13.4)
- Linear programming relaxation (13.5)
- Graph cuts (13.6)
- Search (13.7)

Querying a distribution ("inference")

- Suppose we have a joint $p(X_1,\ldots,X_d)$. Partition the variables into E (evidence), Q (query), and H (hidden/ nuisance). We might pose the following queries
- Conditional probability (posterior):

$$p(\mathbf{X}_Q|\mathbf{x}_E) \propto \sum_{\mathbf{x}_H} p(\mathbf{X}_Q, \mathbf{x}_E, \mathbf{x}_H)$$

- MAP estimate (H=∅) (posterior mode)

$$\mathbf{x}_Q^* = \arg\max_{\mathbf{x}_Q} p(\mathbf{x}_Q|\mathbf{x}_E) = \arg\max_{\mathbf{x}_Q} p(\mathbf{x}_Q, \mathbf{x}_E)$$

- Marginal MAP estimate (mode of marginal post):

$$\mathbf{x}_Q^* = \arg\max_{\mathbf{x}_Q} p(\mathbf{x}_Q|\mathbf{x}_E) = \arg\max_{\mathbf{x}_Q} \sum_{\mathbf{x}_H} p(\mathbf{x}_Q, \mathbf{x}_E, \mathbf{x}_H)$$

MAP vs marginal MAP

- Max max ≠ max sum
- Ex 2.1.12. Joint is

$$
\begin{aligned}
a^* &= \arg\max_a \sum_b p(a,b) = 1 \\
b^* &= \arg\max_b \sum_a p(a,b) = 1 \\
(a,b)^* &= \arg\max_{a,b} p(a,b) = (0,1)
\end{aligned}
$$

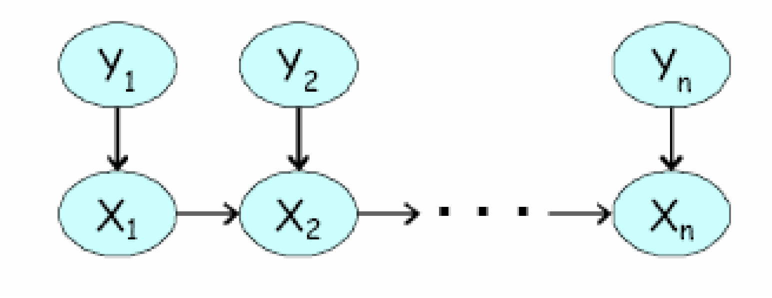
	A=0	A=1	
B=0	0.04	0.3	0.34
B=1	0.36	0.3	0.66
	0.4	0.6	

- Sequence of most probable states <> most probable sequence of states.

MMAP harder than MAP

- Thm 13.1.1. MAP for BNs is NP-hard.
- Thm 13.1.3. MMAP for BNs is complete for NP^{PP}.
- Thm 13.1.4. MMAP for tree structured GMs is NP-hard.
- Pf. Must sum out X before max out Y.

$$y^{p\text{-}map} = \arg\max_{Y_1,\ldots,Y_n} \sum_{X_1,\ldots,X_n} P(Y_1,\ldots,Y_n,X_1,\ldots,X_n).$$



VarElim for MAP

- Since max distributes over products, we can trivially modify the VE algorithm to compute the *scalar* max_x p(x).
- To find the assignment which achieves this MAP probability, we must do a traceback, analogous to the Viterbi traceback algorithm
- For the MMAP case, we can use the same algorithm, but with a constrained elim order (sum before max), which can make the problem harder

Clq Trees for MAP

- VE is inherently sequential: it is hard to imagine how to make a parallel/ distributed version of the traceback operator
- However, we can easily compute the max-marginals in parallel, replacing sum-product messages with max-product

$$\text{MaxMarg}(x_i) = \max_{\mathbf{x}_{-i}} \tilde{p}(\mathbf{x}_{-i}, x_i)$$

- But how do we decode the corresponding assignment? Easy if each MM is unambiguous.

$$\exists \text{unique} \, x_i^* = \arg\max_{x_i} \text{MaxMarg}(x_i)$$

Problems of ambiguity

- Ex 13.3.7

$$
\begin{array}{cc|cc|c}
 & & X_2 & & \\
 & & 0 & 1 & MM_1 \\
\hline
X_1 & 0 & 0.1 & 0.4 & 0.4 \\
 & 1 & 0.4 & 0.1 & 0.4 \\
\hline
MM_2 & & 0.4 & 0.4 &
\end{array}
$$

- If we pick $x_1^*=1$ and $x_2^*=2$, we don't get $(x_1,x_2)^*$
- Must break ties consistently – requires global traceback.

Max-product in loopy cluster graphs

- We can change the sum-product algorithm to max product and run it on clique graphs that are not trees. The result is a set of pseudo max marginals which are max-calibrated

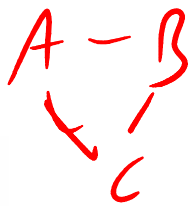
$$\max_{C_i - S_{i,j}} \beta_i = \max_{C_j - S_{i,j}} \beta_j = \mu_{i,j}(S_{i,j}).$$

Decoding pseudo max marginals

- Def 13.3.9. Let β_c be the max marginals in a clique tree/graph. An assignment x^* is locally optimal if

$$\mathbf{x}^*(c) \in \arg\max_{\mathbf{x}_c} \beta_c(\mathbf{x}_c)$$

- We can label each local assignment as equal to the local optimum (1) or not (0). We then need to solve a constraint satisfaction problem (CSP).
- Ex 13.4.2. Consider these “beliefs”:



	a^1	a^0
b^1	1	2
b^0	2	1

	b^1	b^0
c^1	1	2
c^0	2	1

	a^1	a^0
c^1	1	2
c^0	2	1

Max-calibrated but not locally optimal; no solution exists

Quality of approximate solution

- Suppose the solution is locally optimal, so CSP can find a satisfying assignment. This is an exact MAP iff the clique graph is a tree with RIP.
- Suppose it is a general loopy graph. We can show (thm 13.4.6) that the solution is a “strong” local optimum, meaning that any change wrt to a large set of legal moves will decrease the probability.
- The legal moves including flipping states of any embedded subtree or single loops.

Max product TRW

- Suppose we replace "vanilla" max-product with a counting number version

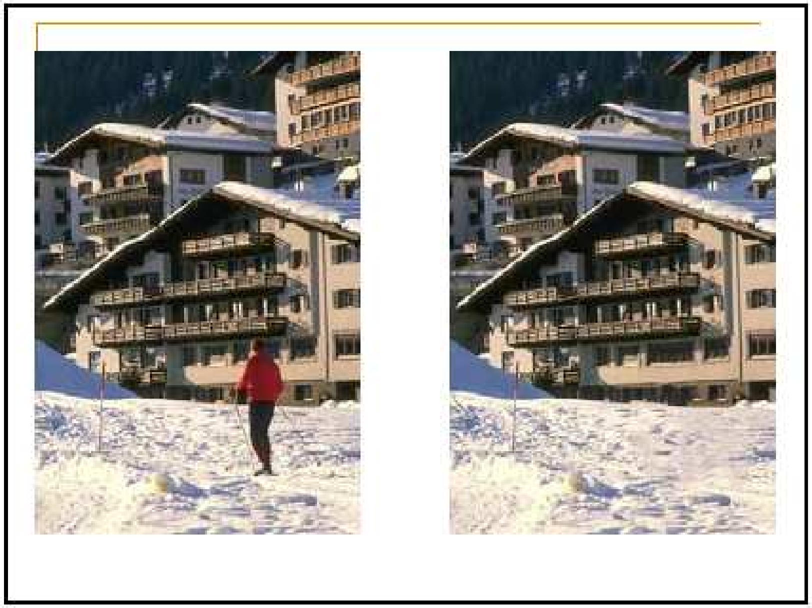
$$\delta_{i \to j} = \max_{x_i} \left[\left(\psi_i[x_i] \prod_{k \in \mathrm{Nb}_i} \delta_{k \to i}(x_i) \right)^{\frac{\mu_{i,j}}{\mu_i}} \frac{1}{\delta_{j \to i}(x_i)} \psi_{i,j}[x_i, x_j] \right].$$

- Tree reweighting algorithm (TRW) uses the following convex counting numbers, given a distribution over trees T st each edge in the pairwise network is present in at least 1 tree

$$\begin{aligned} \mu_i &= -\textstyle\sum_{\mathcal{T} \ni X_i} \rho(\mathcal{T}) \\ \mu_{i,j} &= \textstyle\sum_{\mathcal{T} \ni (X_i, X_j)} \rho(\mathcal{T}) \end{aligned}$$

- Thm 13.4.8. If this algorithm finds a locally optimal solution, it is also globally optimal. (For sum-product, TRW is just convergent.)

Image completion



Priority-BP [Komodakis '06]

- In this case BP has an intolerable computational cost:
  - Just the basic operation of updating messages from node p to node q takes $O(|\mathcal{L}|^2)$ time
  - $|\mathcal{L}|^2$ SSD calculations between patches thus needed **(recall that |L| is huge in our case!)**

- Two extensions over standard-BP to reduce computation cost:
  - **"Dynamic label pruning"** and
  - **"Priority-based message scheduling"**

- **Labels** L = all wxh patches from source region S
- **MRF nodes** = all lattice points whose neighborhood intersects target region T
- **potential** $V_p(x_p)$ = how well source patch x_p agrees with source region around p
- **potential** $V_{pq}(x_p, x_q)$ = how well source patches x_p, x_q agree on their overlapping region

$$\mathcal{F}(\hat{x}) = \sum_{p \in \mathcal{V}} V_p(\hat{x}_p) + \sum_{(p,q) \in \mathcal{E}} V_{pq}(\hat{x}_p, \hat{x}_q)$$

MAP as integer program

- Let $q(x_r^j)=1$ if clique r is in state j.
- Let $\eta_r^j = \log \phi_r(j)$.
- MAP problem:

$$\text{maximize}_{\boldsymbol{q}} \sum_{r \in \mathbf{R}} \sum_{j=1}^{n_r} \eta_r^j q(x_r^j),$$

- Integer constraint:

$$q(x_r^j) \in \{0,1\} \qquad \text{For all } r \in \mathbf{R};\ j \in \{1,\dots,n_r\}$$

- Mutual exclusion constraint:

$$\sum_{j=1}^{n_r} q(x_r^j) = 1 \qquad \text{For all } r \in \mathbf{R}.$$

- Consistency constraint:

$$\sum_{j\ :\ \mathbf{c}_r^j \sim s_{r,r'}} q(x_r^j) = \sum_{l\ :\ \mathbf{c}_{r'}^l \sim s_{r,r'}} q(x_{r'}^l).$$

LP relaxation

- Let $q(x_r^j) \geq 0$ instead of $\{0,1\}$.

$$
\begin{array}{ll}
\text{Find} & \{q(x_{\mathbf{R}}^j) \; : \; r \in \mathbf{R}; \mathbf{j} = \mathbf{1}, \ldots, \mathbf{n_r}\} \\
\text{that maximize} & \eta^{\top} q
\end{array}
$$

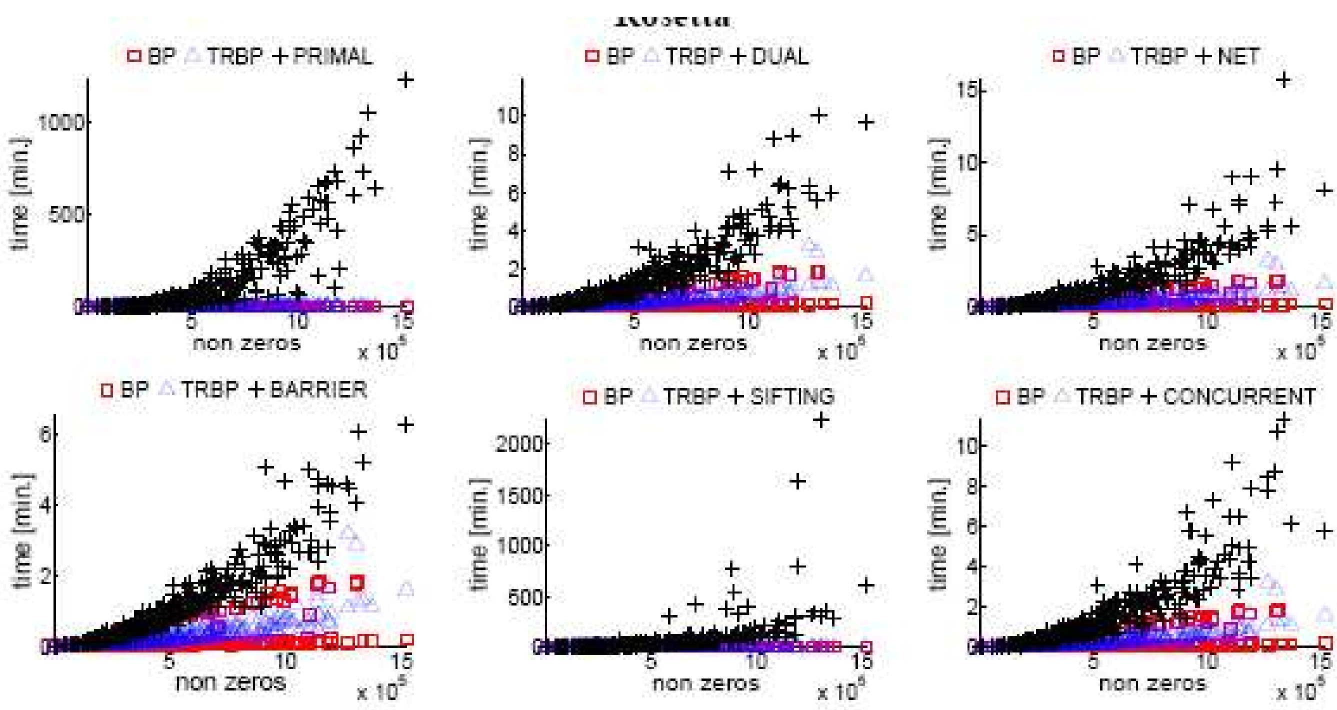
$$
\text{subject to} \quad
\begin{array}{lll}
\displaystyle\sum_{j=1}^{n_r} q(x_r^j) = 1 & r \in \mathbf{R} & (13.25) \\
\displaystyle\sum_{j \,:\, \mathbf{c}_r^j \sim s_{r,r'}} q(x_r^j) = \sum_{l \,:\, \mathbf{c}_{r'}^l \sim s_{r,r'}} q(x_{r'}^l) & \begin{array}{l} r, r' \in \mathbf{R} \\ s_{r,r'} \in Val(C_r \cap C_{r'}) \end{array} & (13.26) \\
q \geq 0 & & (13.27)
\end{array}
$$

Convex BP is solving the dual of this LP.
If the solution is integer, and there are are no ties,
then fixed points of this are exact MAP estimates.

MAP Estimation, Linear Programming and Belief Propagation with Convex Free Energies
Yair Weiss, Chen Yanover, Talya Meltzer, UAI 2007

BP beats CPLEX

- Convex max-product is 100-1000 times faster than CPLEX at finding the exact solution to certain MAP problems in computer vision and protein folding.



Linear Programming Relaxations and Belief Propagation - an Empirical Study
Chen Yanover, Talya Meltzer, Yair Weiss, JMLR 2006

Submodularity

- Let L = {0,1,…,K} be an ordered set.
- Let g: LxL -> R be a function.
- We say g is submodular iff

$$\forall x, y \in \mathcal{L} \;\; g(x \vee y) + g(x \wedge y) \leq g(x) + g(y)$$

$$(x \vee y)_i = \min(x_i, y_i), \;\; (x \wedge y)_i = \max(x_i, y_i)$$

- Submodularity ~ convexity for discrete opt.
- Eg L = {0,1}, g is submodular iff

$$g(0,0) + g(1,1) \leq g(0,1) + g(1,0)$$

$$[(0,1) \vee (1,0)] = [\min(0,1), \min(1,0)] = [0,0],$$

$$[(0,1) \wedge (1,0)] = [\max(0,1), \max(1,0)] = [1,1]$$

Submodular potentials

- Defn 13.6.2. A pairwise energy term on binary nodes is submodular if

$$\epsilon(1,1) + \epsilon(0,0) \leq \epsilon(1,0) + \epsilon(0,1)$$

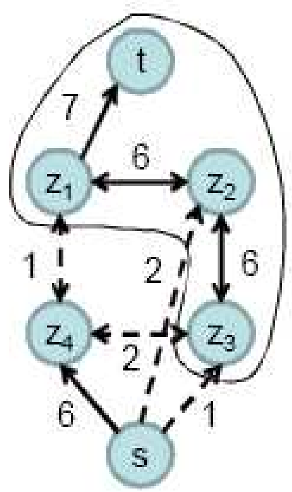
- Example: Ising model with attractive potential

$$\begin{array}{cc} & \begin{array}{cc} 0 & 1 \end{array} \\ \begin{array}{c} 0 \\ 1 \end{array} & \begin{pmatrix} 0 & \lambda \\ \lambda & 0 \end{pmatrix} \end{array} \qquad \lambda \geq 0$$

- For any binary MRF with submodular potentials, we can find the exact MAP in polynomial time

Graph cuts for Ising model

- Create a source and sink node, s, t.
- Add edge X_i->t with weight $\varepsilon_i[0]$.
- Add edge X_i->s with weight $\varepsilon_i[1]$
- Add X_i – X_j with λ_{ij}.
- Find minimal cut. All nodes on t-side of cut are in state 1.



$$\epsilon_1[0] = 7 \quad \epsilon_2[1] = 2 \quad \epsilon_3[1] = 1 \quad \epsilon_4[1] = 6$$
$$\lambda_{1,2} = 6 \quad \lambda_{2,3} = 6 \quad \lambda_{3,4} = 2 \quad \lambda_{1,4} = 1.$$

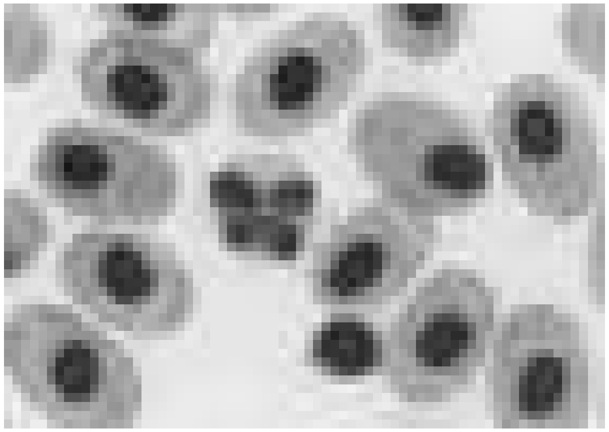
$X_1 - X_2$

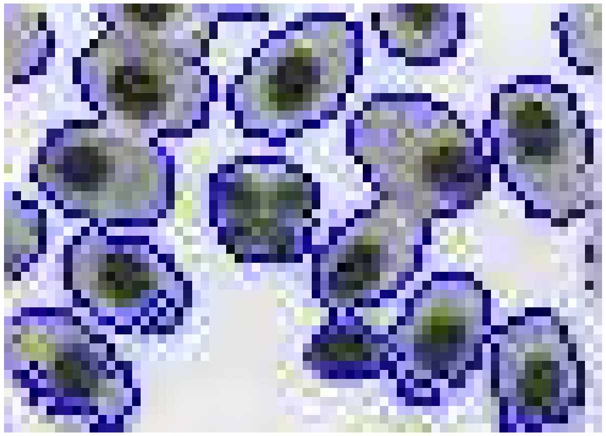
$X_4 - X_3$

$x^* = 1, 1, 1, 0$

$X_{maxmarg} = 1, 0, 0, 0$ but strong constraints

Segmentation using binary MRF





$$P(I(i); p_i = 0) = \frac{1}{\sqrt{2\pi}\sigma_b} \exp\left(-\frac{(I(i)-\mu_b)^2}{2\sigma_b^2}\right),$$

$$P(I(i); p_i = 1) = \frac{0.5}{\sqrt{2\pi}\sigma_{f,1}} \exp\left(-\frac{(I(i)-\mu_{f,1})^2}{2\sigma_{f,1}^2}\right) + \frac{0.5}{\sqrt{2\pi}\sigma_{f,2}} \exp\left(-\frac{(I(i)-\mu_{f,2})^2}{2\sigma_{f,2}^2}\right),$$

Slide by Nilanjan Ray, from Google search

Metric MRFs

- A metric MRF is one with K states and pairwise potentials of the form

$$\epsilon_{i,j}(v_k, v_l) = \mu(v_k, v_l) \geq 0$$

where \mu is a metric:

$\mu : \mathcal{V} \times \mathcal{V} \mapsto [0, \infty)$ *is a metric if it satisfies:*

- **Reflexivity:** $\mu(v_k, v_l) = 0$ *if and only if* $k = l$;
- **Symmetry:** $\mu(v_k, v_l) = \mu(v_l, v_k)$;
- **Triangle Inequality:** $\mu(v_k, v_l) + \mu(v_l, v_m) \geq \mu(v_k, v_m)$.

- Hence for any v we have submodularity:

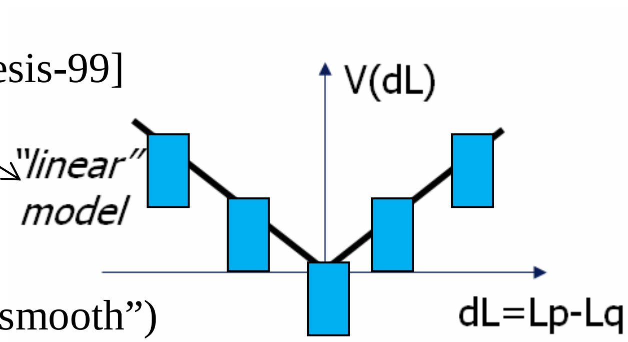
$$\epsilon_{i,j}[x_i, x_j] + \epsilon_{i,j}[v, v] \;\underset{?}{\cancel{\geq}}\; \epsilon_{i,j}[x_i, v] + \epsilon_{i,j}[v, x_j].$$ Δ ineq

Functions of label differences

- V(*p*,*q*) is 2nd order potential of the difference in the labels of pixels *p* and *q*
  - These functions penalize big difference in label values between neighboring data
    - Image restoration: want to maintain similar intensities with neighbors

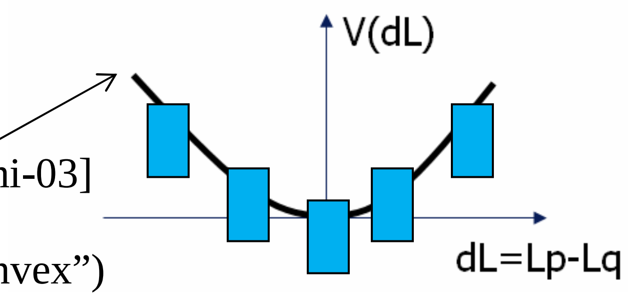
Convex interactions
(minimize is P)

[veksler-phdthesis-99]
(exact)



("everywhere smooth")

[ishikawa-pami-03]
(exact)

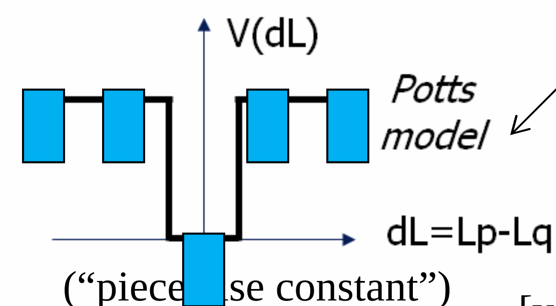


("convex")

$$\epsilon[x_i, x_j] = \min(c\|x_i - x_j\|_p, \mathrm{dist}_{\max}),$$

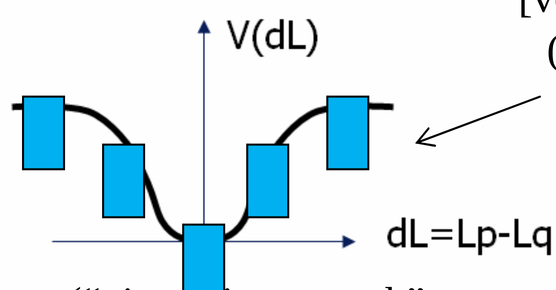
Robust or "discontinuity-preserving" interactions
(minimize is NP-complete)

[boykov-pami-01]
(approximate)



("piecewise constant")

[veksler-cvpr-07]
(approximate)

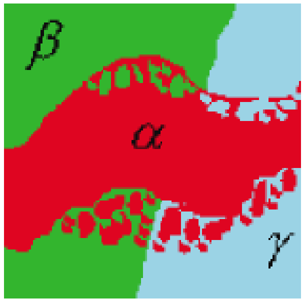


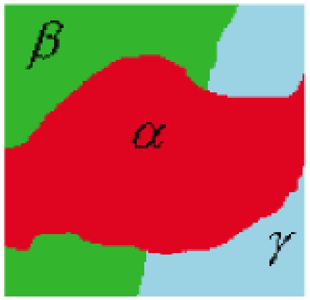
("piecewise smooth", "truncated convex")

Source: Daniel Munoz

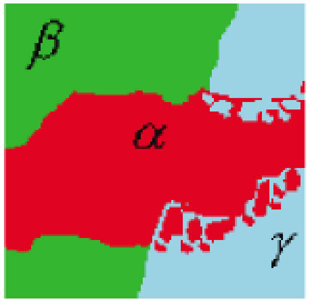
GC for non-binary submodular

- For non-binary models, MAP estimation is NP-hard.
- But if the potential is submodular for any pair of states (eg metric MRF) then we can use a greedy algorithm in which we make large moves
- Alpha expansion: consider setting each node to its current state or to state α (2-optimal).
- Alpha-beta swap: consider swapping any two states; energy function only need be semi-metric (triangle inequality not required).





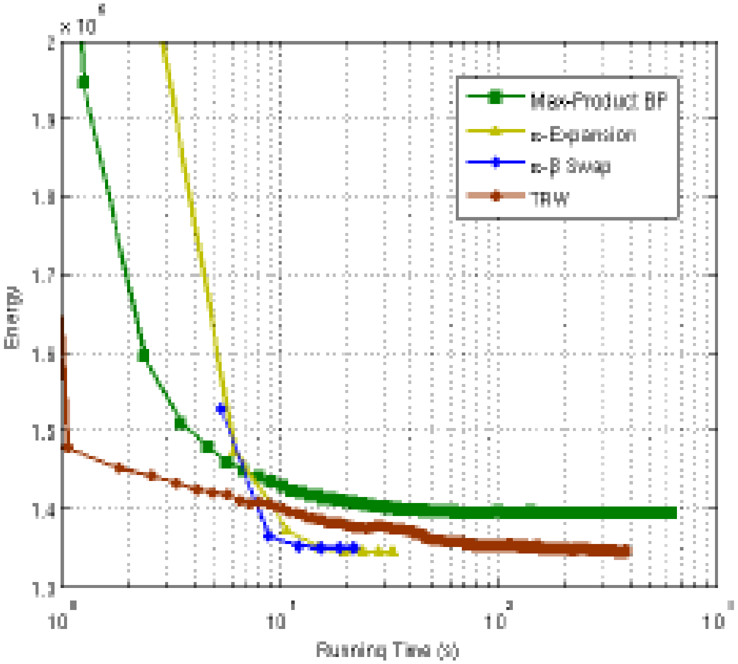
Expansion

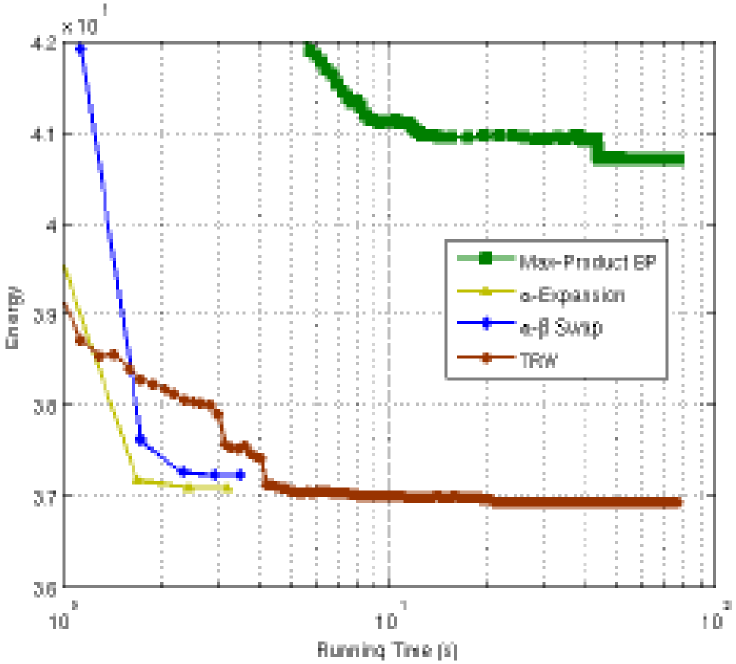


Swap

(Boykov'01)

Stereo reconstruction





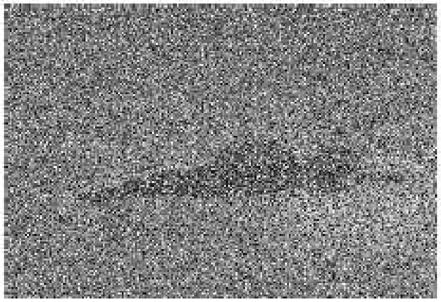
Total variation norm

$$TV(u) = \int |\nabla u| du$$

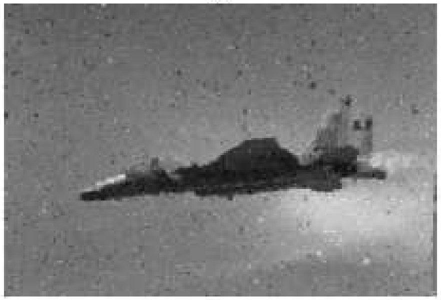
$$g_{pq}(u_p, u_q) = \beta |u_p - u_q|$$

Graph cuts on the level sets

(a)



(b)



(c)

Global Optimization for First Order Markov Random Fields with Submodular Priors
Jerome Darbon,
Discrete Applied Mathematics, 2009

Additional info

Good tutorial at ECCV'08: "**MAP Estimation in Computer Vision**"
Kumar, Kolhi, Zisserman, Torr
http://www.robots.ox.ac.uk/~pawan/eccv08_tutorial/index.html

"A Linear Programming Approach to Max-sum Problem: A Review",
Tomas Werner, PAMI 2007

Search (A.4)

- Systematic tree search – partial assignments
  - Branch and bound: prune off trajectory if lower bound of extension higher than current best
  - Particle filtering: stochastically grow partial solutions
- Local search – complete assignments
  - Hill climbing, Tabu search, Beam search, simulated annealing
  - See Holger Hoos's class in CS
- Search methods for Marginal MAP
  - Search over max, compute sum using VE (cf Rao-Blackwellize). Use unconstrained elim order to get upper bound.

Greedy hill climbing

Algorithm A.5 Greedy local search algorithm with search operators.

```
Procedure Greedy-Local-Search (
   σ_0,     // initial candidate solution
   score,   // Score function
   O,       // Set of search operators
 )
   σ_best ← σ_0
   do
     σ ← σ_best
     Progress ← false
     for each operator o ∈ O
       σ_o ← o(σ)    // Result of applying o on σ
       if σ_o is legal solution then
         if score(σ_o) > score(σ_best) then
           σ_best ← σ_o
           Progress ← true
   while Progress

   return σ_best
```

Instead of looking amongst all neighbors O, we can pick the first improving one (first-ascent or best first search).
Converges to local maximum or plateau.

Tabu search

- Once we get to a plateau, allow selection of ‘neutral’ move to a state that hasn’t been visited before .
- Requires lots of memory. Instead, prevent picking a move that would undo a recently applied operator.

```
σ_best ← σ_0
σ ← σ_best
t ← 1
LastImprovement ← 0
while LastImprovement < N
  o^(t) ← ε      // Set current operator to be uninitialized
  for each operator o ∈ 𝒪      // Search for best allowed operator
    if LegalOp(o, {o^(t−L), ..., o^(t−1)}) then
      σ_o ← o(σ)
      if σ_o is legal solution then
        if o^(t) = ε or score(σ_o) > score(σ_{o^t}) then
          o^(t) ← o
  σ ← σ_{o^t}
  if score(σ) > score(σ_best) then
    σ_best ← σ_o
    LastImprovement ← 0
  else
    LastImprovement ← LastImprovement + 1
  t ← t + 1

return σ_best
```