Statistical Machine Learning

https://cvml.ist.ac.at/courses/SML_W18

Christoph Lampert

IST AUSTRIA

Institute of Science and Technology

Winter Semester 2018/2019

Lecture 12

(lots of material courtesy of S. Nowozin, http://www.nowozin.net)

Overview (tentative)

Date		no.	Topic
Oct 08	Mon	1	A Hands-On Introduction
Oct 10	Wed	–	self-study (Christoph traveling)
Oct 15	Mon	2	Bayesian Decision Theory
			Generative Probabilistic Models
Oct 17	Wed	3	Discriminative Probabilistic Models
			Maximum Margin Classifiers
Oct 22	Mon	4	Generalized Linear Classifiers, Optimization
Oct 24	Wed	5	Evaluating Predictors; Model Selection
Oct 29	Mon	–	self-study (Christoph traveling)
Oct 31	Wed	6	Overfitting/Underfitting, Regularization
Nov 05	Mon	7	Learning Theory I: classical/Rademacher bounds
Nov 07	Wed	8	Learning Theory II: miscellaneous
Nov 12	Mon	9	Probabilistic Graphical Models I
Nov 14	Wed	10	Probabilistic Graphical Models II
Nov 19	Mon	11	Probabilistic Graphical Models III
Nov 21	Wed	12	Probabilistic Graphical Models IV
until Nov 25			final project

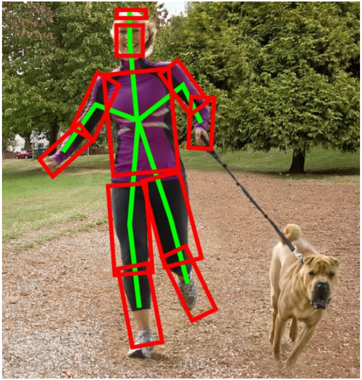
MAP Prediction / Energy Minimization

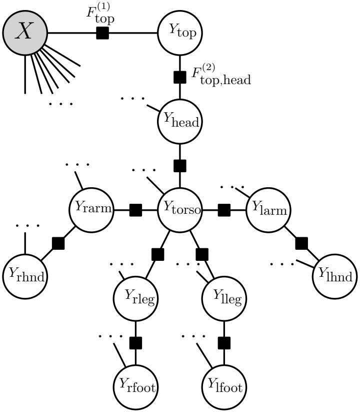
$$\mathbf{argmax}_y \, p(y|x) \ / \ \mathbf{argmin}_y \, E(x,y) \ /$$
$$\mathbf{argmax}_y \, \Delta(y_i, y) + \langle w, \psi(x,y) \rangle$$

MAP Prediction / Energy Minimization

Task: Minimize $E(x, y)$ or $\Delta(y_i, y) + E(x, y)$ for $E(x, y) = \langle w, \psi(x, y) \rangle$

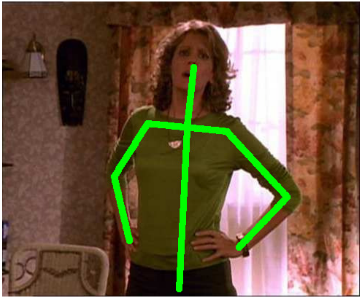
- Exact Energy Minimization
  - Belief Propagation on chains/trees
  - Graph-Cuts for submodular energies
  - Integer Linear Programming
- Approximate Energy Minimization
  - Linear Programming Relaxations
  - Local Search Methods
    - Iterative Conditional Modes
    - Multi-label Graph Cuts
  - Simulated Annealing



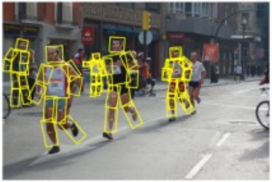


- **Tree-structured model** for articulated pose
(Felzenszwalb and Huttenlocher, 2000), (Fischler and Elschlager, 1973), (Yang and Ramanan, 2013), (Pishchulin *et al.*, 2012)

Example: Pictorial Structures / Deformable Parts Model

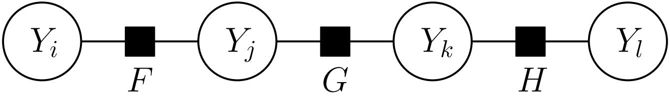


- most likely configuration $y^* = \operatorname*{\mathbf{argmax}}_{y \in \mathcal{Y}} p(y|x) = \operatorname*{\mathbf{argmin}}_{y} E(y, x)$



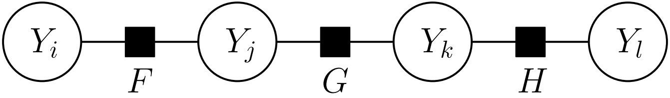
Energy Minimization – Belief Propagation

Chain model: same trick as for *inference*: **belief propagation**



$$\min_y E(y) = \min_{y_i,y_j,y_k,y_l} \quad E_F(y_i,y_j) + E_G(y_j,y_k) + E_H(y_k,y_l)$$

Chain model: same trick as for *inference*: **belief propagation**

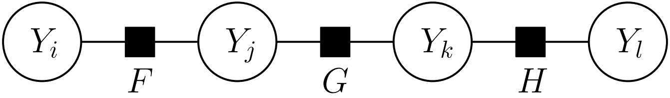


$$\min_y E(y) = \min_{y_i,y_j,y_k,y_l} \quad E_F(y_i,y_j) + E_G(y_j,y_k) + E_H(y_k,y_l)$$

$$= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \min_{y_k} \left[E_G(y_j,y_k) + \min_{y_l} E_H(y_k,y_l) \right] \right]$$

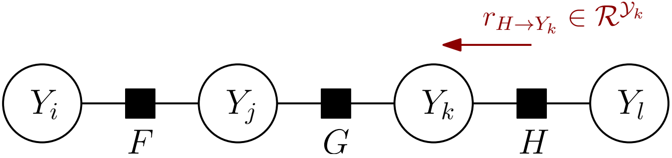
Energy Minimization – Belief Propagation

Chain model: same trick as for *inference*: **belief propagation**



$$\begin{aligned}\min_y E(y) &= \min_{y_i,y_j,y_k,y_l} \quad E_F(y_i,y_j) + E_G(y_j,y_k) + E_H(y_k,y_l) \\ &= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \min_{y_k}\left[E_G(y_j,y_k) + \underbrace{\min_{y_l} E_H(y_k,y_l)}_{r_{H\to Y_k}(y_k)}\right]\right]\end{aligned}$$

Chain model: same trick as for *inference*: **belief propagation**

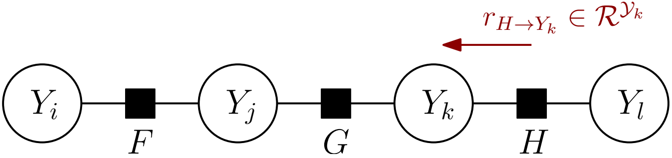


$$\min_{y} E(y) = \min_{y_i, y_j, y_k, y_l} \quad E_F(y_i, y_j) + E_G(y_j, y_k) + E_H(y_k, y_l)$$

$$= \min_{y_i, y_j} \left[E_F(y_i, y_j) + \min_{y_k} \left[E_G(y_j, y_k) + \underbrace{\min_{y_l} E_H(y_k, y_l)}_{r_{H \to Y_k}(y_k)} \right] \right]$$

$$= \min_{y_i, y_j} \left[E_F(y_i, y_j) + \min_{y_k} E_G(y_j, y_k) + r_{H \to Y_k}(y_k) \right]$$

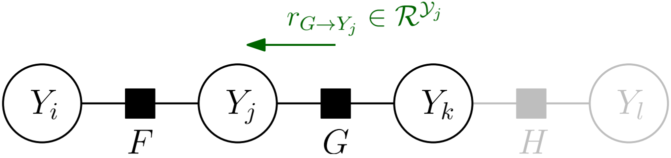
Chain model: same trick as for *inference*: **belief propagation**



$$\begin{aligned}
\min_y E(y) &= \min_{y_i,y_j,y_k,y_l} \quad E_F(y_i,y_j) + E_G(y_j,y_k) + E_H(y_k,y_l) \\
&= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \min_{y_k}\left[E_G(y_j,y_k) + \underbrace{\min_{y_l} E_H(y_k,y_l)}_{r_{H\to Y_k}(y_k)}\right]\right] \\
&= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \underbrace{\min_{y_k} E_G(y_j,y_k) + r_{H\to Y_k}(y_k)}_{r_{G\to Y_j}(y_j)}\right]
\end{aligned}$$

Energy Minimization – Belief Propagation

Chain model: same trick as for *inference*: **belief propagation**



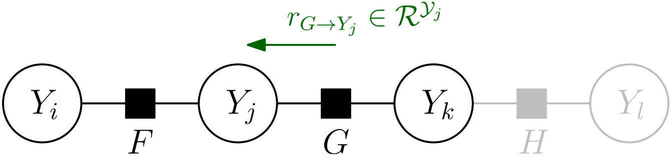
$$\min_y E(y) = \min_{y_i,y_j,y_k,y_l} \quad E_F(y_i,y_j) + E_G(y_j,y_k) + E_H(y_k,y_l)$$

$$= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \min_{y_k} \left[E_G(y_j,y_k) + \underbrace{\min_{y_l} E_H(y_k,y_l)}_{r_{H\to Y_k}(y_k)} \right]\right]$$

$$= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \underbrace{\min_{y_k} E_G(y_j,y_k) + r_{H\to Y_k}(y_k)}_{r_{G\to Y_j}(y_j)} \right]$$

$$= \min_{y_i,y_j} \left[E_F(y_i,y_j) + r_{G\to Y_j}(y_j) \right] \quad \ldots$$

Chain model: same trick as for *inference*: **belief propagation**



$$\min_y E(y) = \min_{y_i,y_j,y_k,y_l} \quad E_F(y_i,y_j) + E_G(y_j,y_k) + E_H(y_k,y_l)$$

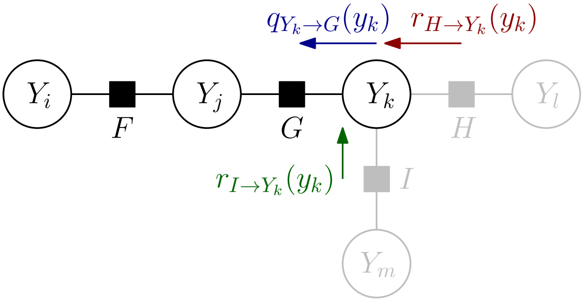
$$= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \min_{y_k}\left[E_G(y_j,y_k) + \underbrace{\min_{y_l} E_H(y_k,y_l)}_{r_{H\to Y_k}(y_k)}\right]\right]$$

$$= \min_{y_i,y_j} \left[E_F(y_i,y_j) + \underbrace{\min_{y_k} E_G(y_j,y_k) + r_{H\to Y_k}(y_k)}_{r_{G\to Y_j}(y_j)}\right]$$

$$= \min_{y_i,y_j} \left[E_F(y_i,y_j) + r_{G\to Y_j}(y_j)\right] \quad \ldots$$

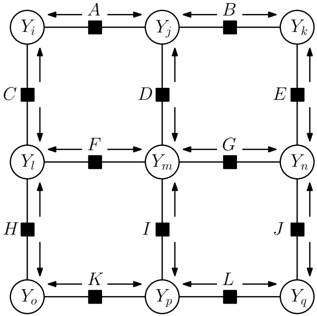
- actual **argmax** by backtracking which choices were maximal

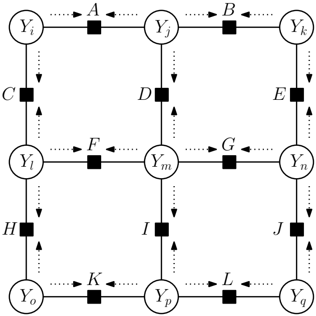
Tree models:



- $q_{H \to Y_k}(y_k) = \mathbf{min}_{y_l} E_H(y_k, y_l)$
- $q_{I \to Y_k}(y_k) = \mathbf{min}_{y_m} E_I(y_k, y_m)$
- $q_{Y_k \to G}(y_k) = q_{H \to Y_k}(y_k) + q_{I \to Y_k}(y_k)$

min-sum (more common **max-sum**) **belief propagation**





Loopy Max-Sum Belief Propagation

Same problem as in probabilistic inference:

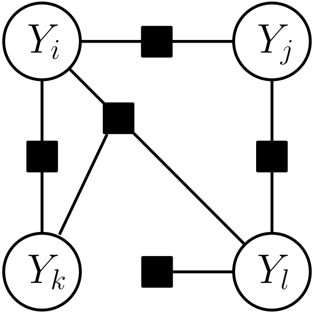
- no guarantee of convergence
- no guarantee of optimality

Convergent variants, e.g. TRW-S [Kolmogorov, PAMI 2006] still approximate

In general, MAP prediction/energy minimization in models with cycles or higher-order terms is **intractable** (NP-hard).

Some important exceptions:

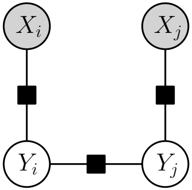
- low tree-width [Lauritzen, Spiegelhalter, 1988]
- **binary states, pairwise submodular interactions [Boykov, Jolly, 2001]**
- binary states, only pairwise interactions, planar graph [Globerson, Jaakkola, 2006]
- special (Potts $\mathcal{P}^n$) higher order factors [Kohli, Kumar, 2007]
- perfect graph structure [Jebara, 2009]



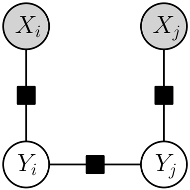
Submodular Energy Functions

- Binary variables: $\mathcal{Y}_i = \{0, 1\}$ for all $i \in \mathcal{V}$
- Energy function: unary and pairwise factors

$$E(y; x, w) = \sum_{i \in \mathcal{V}} E_i(y_i) + \sum_{(i,j) \in \mathcal{E}} E_{ij}(y_i, y_j)$$



Submodular Energy Functions



- Binary variables: $\mathcal{Y}_i = \{0, 1\}$ for all $i \in \mathcal{V}$
- Energy function: unary and pairwise factors

$$E(y; x, w) = \sum_{i \in \mathcal{V}} E_i(y_i) + \sum_{(i,j) \in \mathcal{E}} E_{ij}(y_i, y_j)$$

- Restriction 1 (without loss of generality):

$$E_i(y_i) \geq 0$$

(always achievable by adding a constant to E)

- Restriction 2 (**submodularity**):

$$\begin{aligned} E_{ij}(y_i, y_j) &= 0, && \text{if } y_i = y_j, \\ E_{ij}(y_i, y_j) &= E_{ij}(y_j, y_i) \geq 0, && \text{otherwise.} \end{aligned}$$

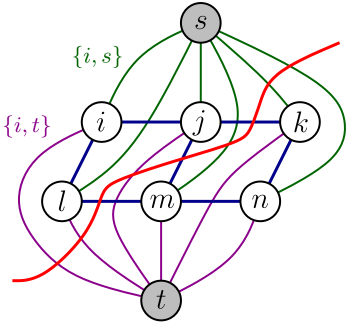
"neighbors prefer to have the same labels"

If conditions are fulfilled, energy minimization can be performed by a solving an s-t-**mincut** problem:

- construct auxiliary undirected graph
- one node $\{i\}_{i \in V}$ per variable
- two extra nodes: source s, sink t
- weighted edges

Edge	weight
$\{i,j\}$	$E_{ij}(y_i = 0, y_j = 1)$
$\{i,s\}$	$E_i(y_i = 1)$
$\{i,t\}$	$E_i(y_i = 0)$

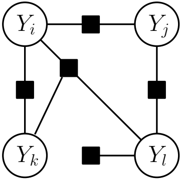
- find s-t-cut of minimal weight (polynomial time using max-flow theorem)



From minimal weight cut we recover labeling of minimal energy:

- $y_i^* = 1$ if edge $\{i,s\}$ is cut. Otherwise $y_i^* = 0$

Integer Linear Programming (ILP)



General energy $E(y) = \sum_F E_F(y_F)$

- variables with more than 2 states
- higher-order factors (more than 2 variables)
- non-submodular factors

Formulate as integer linear program (ILP)

- linear objective function
- linear constraints
- variables to optimize over are integer-valued

ILPs are in general NP-hard, but some individual instances can be solved

- standard toolboxes: e.g. CPLEX, Gurobi, COIN-OR, . . .

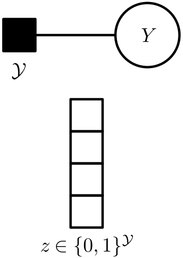
We can write any discrete optimization as an Integer Linear Program:

$$\min_{y \in \mathcal{Y}} \; E(y) \quad \text{for} \quad \mathcal{Y} = \{1, \ldots, K\}$$

$$\min_{z \in \{0,1\}^K} \sum_{k=1}^{K} \theta_k z_k \quad \text{subject to} \quad \sum_{k=1}^{K} z_k = 1$$

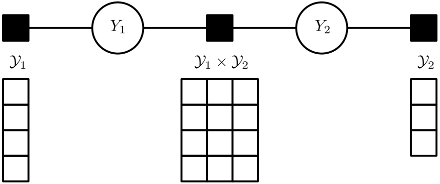
Encode assignment in indicator variables:

- $z \in \{0,1\}^K \qquad z_k = 1 \;\Leftrightarrow\; [\![y = k]\!]$
- coefficient vector: $\theta_k = E(k)$
- constraint: $\sum_k z_k = 1 \;\; \rightarrow$ exactly one 1



Integer Linear Programming (ILP)

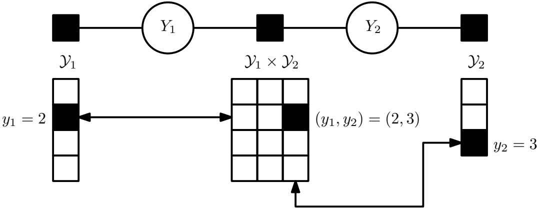
Example:



Encode assignment in indicator variables:

- $z_1 \in \{0,1\}^{\mathcal{Y}_1}$ $\qquad z_{1;k} = 1 \Leftrightarrow [\![y_1 = k]\!]$
- $z_2 \in \{0,1\}^{\mathcal{Y}_2}$ $\qquad z_{2;l} = 1 \Leftrightarrow [\![y_2 = l]\!]$
- $z_{12} \in \{0,1\}^{\mathcal{Y}_1 \times \mathcal{Y}_2}$ $\qquad z_{12;kl} = 1 \Leftrightarrow [\![y_1 = k \wedge y_2 = l]\!]$

Example:



Encode assignment in indicator variables:

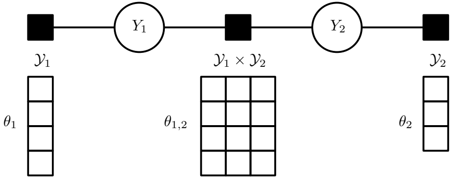
- $z_1 \in \{0,1\}^{\mathcal{Y}_1}$ $\qquad z_{1;k} = 1 \Leftrightarrow [\![y_1 = k]\!]$
- $z_2 \in \{0,1\}^{\mathcal{Y}_2}$ $\qquad z_{2;l} = 1 \Leftrightarrow [\![y_2 = l]\!]$
- $z_{12} \in \{0,1\}^{\mathcal{Y}_1 \times \mathcal{Y}_2}$ $\qquad z_{12;kl} = 1 \Leftrightarrow [\![y_1 = k \wedge y_2 = l]\!]$

Constraints:

$$\sum_{k \in \mathcal{Y}_1} z_{1;k} = 1, \quad \sum_{l \in \mathcal{Y}_2} z_{2;l} = 1, \quad \sum_{k,l \in \mathcal{Y}_1 \times \mathcal{Y}_2} z_{12;kl} = 1 \quad \text{(indicator property)}$$

$$\sum_{k \in \mathcal{Y}_1} z_{12;kl} = z_{2;l} \qquad \sum_{l \in \mathcal{Y}_2} z_{12;kl} = z_{1;k} \qquad \text{(consistency)}$$

Example: $$E(y_1, y_2) = E_1(y_1) + E_{12}(y_1, y_2) + E_2(y_2)$$



Define coefficient vectors:

- $\theta_1 \in \mathbb{R}^{\mathcal{Y}_1}$ $\quad \theta_{1;k} = E_1(k)$
- $\theta_2 \in \mathbb{R}^{\mathcal{Y}_2}$ $\quad \theta_{2;l} = E_2(l)$
- $\theta_{12} \in \mathbb{R}^{\mathcal{Y}_1 \times \mathcal{Y}_2}$ $\quad \theta_{12;kl} = E_{1,2}(k, l)$

Energy is a linear function of unknown z:

$$E(y_1, y_2) = \sum_{i \in V} \sum_{k \in \mathcal{Y}_i} \theta_{i;k} [\![y_i = k]\!] + \sum_{i,j \in \mathcal{E}} \sum_{k,l \in \mathcal{Y}_i \times \mathcal{Y}_j} \theta_{ij;kl} [\![y_i = k \wedge y_j = l]\!]$$

$$= \sum_{i \in V} \sum_{k \in \mathcal{Y}_i} \theta_{i;k} z_{i;k} + \sum_{(i,j) \in \mathcal{E}} \sum_{(k,l) \in \mathcal{Y}_i \times \mathcal{Y}_j} \theta_{ij;kl} z_{ij;kl}$$

$$\min_{z} \sum_{i \in V} \sum_{k \in \mathcal{Y}_i} \theta_{i;k} z_{i;k} + \sum_{(i,j) \in \mathcal{E}} \sum_{(k,l) \in \mathcal{Y}_i \times \mathcal{Y}_j} \theta_{ij;kl} z_{ij;kl}$$

subject to

$$z_{i;k} \in \{0,1\} \quad \text{for all } i \in V,\ \forall k \in \mathcal{Y}_i,$$

$$z_{ij;kl} \in \{0,1\} \quad \text{for all } (i,j) \in \mathcal{E}, (k,l) \in \mathcal{Y}_i \times \mathcal{Y}_j,$$

$$\sum_{k \in \mathcal{Y}_i} z_{i;k} = 1, \quad \text{for all } i \in V,$$

$$\sum_{k,l \in \mathcal{Y}_i \times \mathcal{Y}_j} z_{ij;kl} = 1, \quad \text{for all } (i,j) \in \mathcal{E},$$

$$\sum_{k \in \mathcal{Y}_i} z_{ij;kl} = z_{j;l} \quad \text{for all } (i,j) \in \mathcal{E},\ l \in \mathcal{Y}_j,$$

$$\sum_{l \in \mathcal{Y}_j} z_{ij;kl} = z_{i;k} \quad \text{for all } (i,j) \in \mathcal{E},\ k \in \mathcal{Y}_i,$$

Integer Linear Programming (ILP)

$$\min_{z} \sum_{i\in V}\sum_{k\in\mathcal{Y}_i} \theta_{i;k} z_{i;k} + \sum_{(i,j)\in\mathcal{E}} \sum_{(k,l)\in\mathcal{Y}_i\times\mathcal{Y}_j} \theta_{ij;kl} z_{ij;kl}$$

subject to

$$z_{i;k} \in \{0,1\} \quad \text{for all } i \in V,\ \forall k \in \mathcal{Y}_i,$$

$$z_{ij;kl} \in \{0,1\} \quad \text{for all } (i,j) \in \mathcal{E}, (k,l) \in \mathcal{Y}_i \times \mathcal{Y}_j,$$

$$\sum_{k\in\mathcal{Y}_i} z_{i;k} = 1, \quad \text{for all } i \in V,$$

$$\sum_{k,l\in\mathcal{Y}_i\times\mathcal{Y}_j} z_{ij;kl} = 1, \quad \text{for all } (i,j) \in \mathcal{E},$$

$$\sum_{k\in\mathcal{Y}_i} z_{ij;kl} = z_{j;l} \quad \text{for all } (i,j) \in \mathcal{E},\ l \in \mathcal{Y}_j,$$

$$\sum_{l\in\mathcal{Y}_j} z_{ij;kl} = z_{i;k} \quad \text{for all } (i,j) \in \mathcal{E},\ k \in \mathcal{Y}_i,$$

NP-hard to solve because of integrality constraints.

Linear Programming (LP) Relaxation

$$\min_{\boldsymbol{z}} \quad \sum_{i \in V} \sum_{k \in \mathcal{Y}_i} \theta_{i;k} z_{i;k} + \sum_{(i,j) \in \mathcal{E}} \sum_{(k,l) \in \mathcal{Y}_i \times \mathcal{Y}_j} \theta_{ij;kl} z_{ij;kl}$$

subject to

$$\cancel{z_{i;k} \in \{0,1\}} \;\; z_{i;k} \in [0,1] \qquad \text{for all } i \in V, \; \forall k \in \mathcal{Y}_i,$$

$$\cancel{z_{ij;kl} \in \{0,1\}} \;\; z_{ij;kl} \in [0,1] \qquad \text{for all } (i,j) \in \mathcal{E}, (k,l) \in \mathcal{Y}_i \times \mathcal{Y}_j,$$

$$\sum_{k \in \mathcal{Y}_i} z_{i;k} = 1, \quad \text{for all } i \in V,$$

$$\sum_{k,l \in \mathcal{Y}_i \times \mathcal{Y}_j} z_{ij;kl} = 1, \quad \text{for all } (i,j) \in \mathcal{E},$$

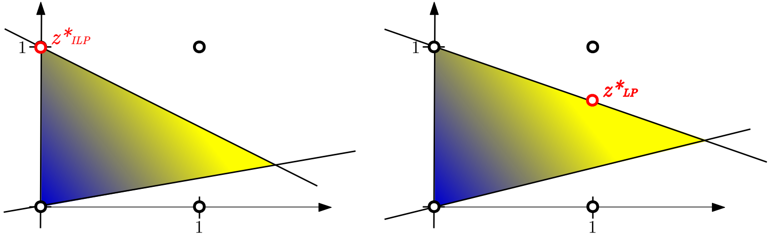
$$\sum_{k \in \mathcal{Y}_i} z_{ij;kl} = z_{j;l} \quad \text{for all } (i,j) \in \mathcal{E}, \; l \in \mathcal{Y}_j,$$

$$\sum_{l \in \mathcal{Y}_j} z_{ij;kl} = z_{i;k} \quad \text{for all } (i,j) \in \mathcal{E}, \; k \in \mathcal{Y}_i,$$

Relax constraints $\rightarrow$ tractable optimization problem

Solution $\boldsymbol{z}^*_{LP}$ might have fractional values

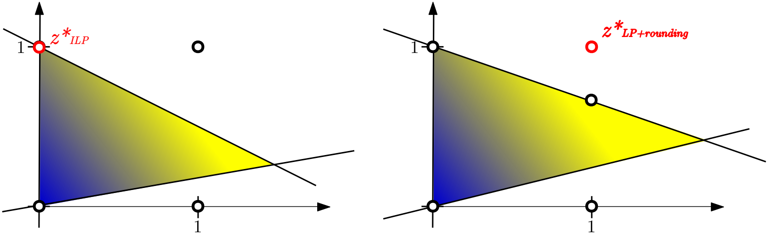
- $\rightarrow$ no corresponding labeling $y \in \mathcal{Y}$
- $\rightarrow$ round LP solution to $\{0, 1\}$ values



Linear Programming (LP) Relaxation

Solution $\boldsymbol{z}^*_{LP}$ might have fractional values

- $\rightarrow$ no corresponding labeling $y \in \mathcal{Y}$
- $\rightarrow$ round LP solution to $\{0, 1\}$ values



Problem:

- rounded solution usually not optimal, i.e. not identical to ILP solution

LP relaxations perform approximate energy minimization

Linear Programming (LP) Relaxation

Example: color quantization

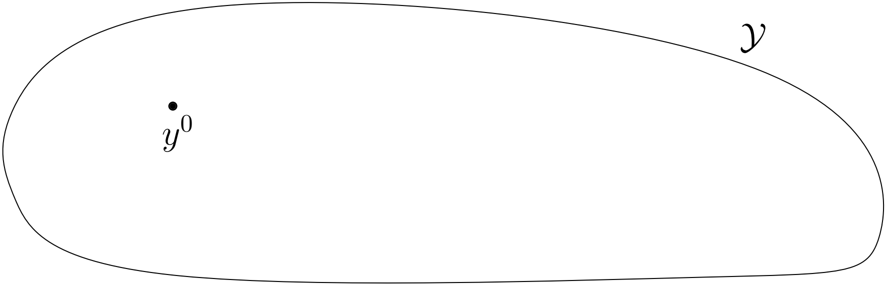
Example: stereo reconstruction

Images: Berkeley Segmentation Dataset

Avoid getting fractional solutions: energy minimization by **local search**

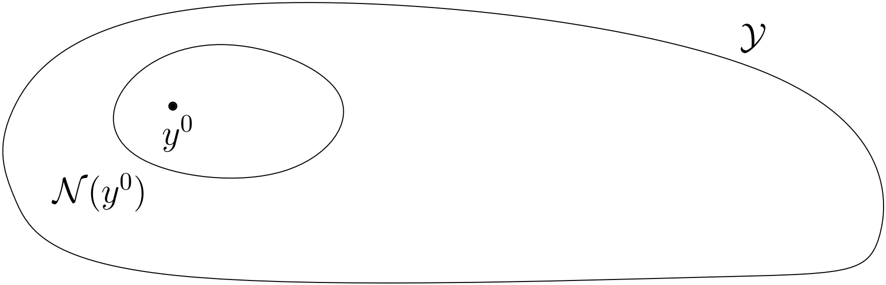
Local Search

Avoid getting fractional solutions: energy minimization by **local search**



- choose starting labeling y^0

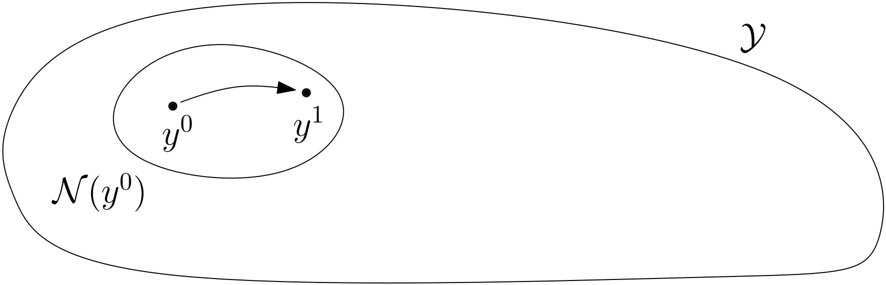
Avoid getting fractional solutions: energy minimization by **local search**



- choose starting labeling y^0
- construct neighborhood $\mathcal{N}(y^0) \subset \mathcal{Y}$ of labelings

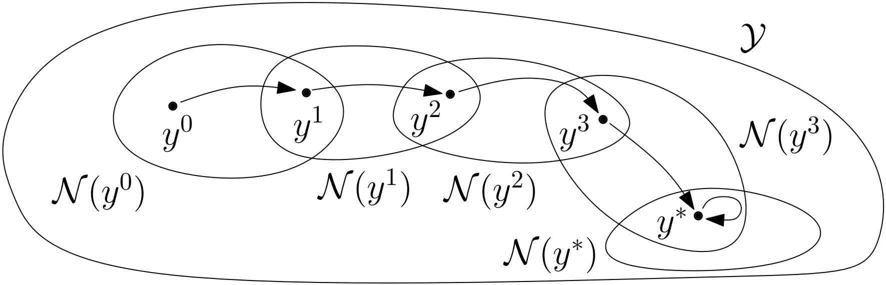
Local Search

Avoid getting fractional solutions: energy minimization by **local search**



- choose starting labeling y^0
- construct neighborhood $\mathcal{N}(y^0) \subset \mathcal{Y}$ of labelings
- find minimizer within neighborhood, $y^1 = \mathbf{argmin}_{y \in \mathcal{N}(y^0)} E(y)$

Avoid getting fractional solutions: energy minimization by **local search**



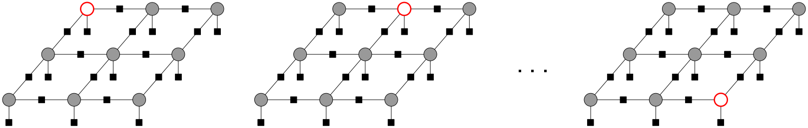
- choose starting labeling y^0
- construct neighborhood $\mathcal{N}(y^0) \subset \mathcal{Y}$ of labelings
- find minimizer within neighborhood, $y^1 = \mathbf{argmin}_{y \in \mathcal{N}(y^0)} E(y)$
- iterate until no more changes

Iterated Conditional Modes (ICM) [Besag, 1986]

Define local neighborhoods:

- $\mathcal{N}_i(y) = \{(y_1, \ldots, y_{i-1}, \bar{y}, y_{i+1}, \ldots, y_n) | \bar{y} \in \mathcal{Y}_i\}$ for $i \in V$.

  all labeling reachable from y by changing value of y_i



Iterated Conditional Modes (ICM) [Besag, 1986]

Define local neighborhoods:

- $\mathcal{N}_i(y) = \{(y_1, \ldots, y_{i-1}, \bar{y}, y_{i+1}, \ldots, y_n) | \bar{y} \in \mathcal{Y}_i\}$ for $i \in V$.

  all labeling reachable from y by changing value of y_i

ICM procedure:

- neighborhood $\mathcal{N}(y) = \bigcup_{i \in V} \mathcal{N}_i(y)$

  all states reachable from y by changing a single variable

- $y^{t+1} = \operatorname*{argmin}_{y \in \mathcal{N}(y^t)} E(y)$ by exhaustive search   ($\sum_i |\mathcal{Y}_i|$ evaluations)

Iterated Conditional Modes (ICM) [Besag, 1986]

Define local neighborhoods:

- $\mathcal{N}_i(y) = \{(y_1, \dots, y_{i-1}, \bar{y}, y_{i+1}, \dots, y_n) | \bar{y} \in \mathcal{Y}_i\}$ for $i \in V$.

  all labeling reachable from y by changing value of y_i

ICM procedure:

- neighborhood $\mathcal{N}(y) = \bigcup_{i \in V} \mathcal{N}_i(y)$

  all states reachable from y by changing a single variable

- $y^{t+1} = \operatorname*{\mathbf{argmin}}_{y \in \mathcal{N}(y^t)} E(y)$ by exhaustive search ($\sum_i |\mathcal{Y}_i|$ evaluations)

Observation: larger neighborhood sizes are better

- ICM: $|\mathcal{N}(y)|$ linear in $|V|$
  $\rightarrow$ many iterations to explore exponentially large $\mathcal{Y}$
- ideal: $|\mathcal{N}(y)|$ exponential in $|V|$,
  $\rightarrow$ but: we must ensure that $\mathbf{argmin}_{y \in \mathcal{N}(y)} E(y)$ remains tractable

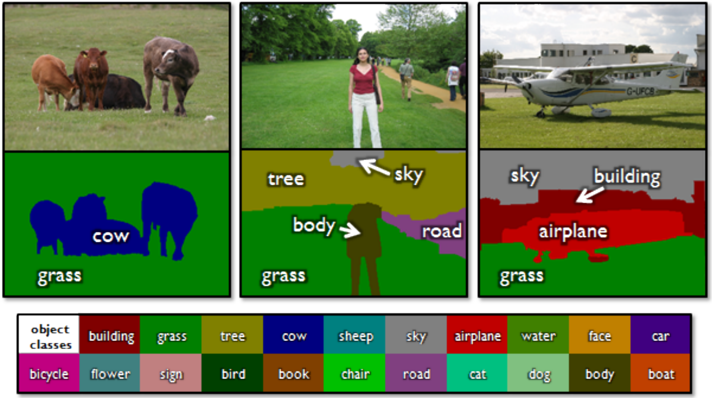
Multilabel Graph-Cut: α-expansion

- $E(y)$ with unary and pairwise terms
- $\mathcal{Y}_i = \mathcal{L} = \{1, \ldots, K\}$ for $i \in V$ (multi-class)

Multilabel Graph-Cut: α-expansion

- $E(y)$ with unary and pairwise terms
- $\mathcal{Y}_i = \mathcal{L} = \{1, \ldots, K\}$ for $i \in V$ (multi-class)

Example: semantic segmentation



- $E(y)$ with unary and pairwise terms
- $\mathcal{Y}_i = \mathcal{L} = \{1, \dots, K\}$ for $i \in V$ (multi-class)

Algorithm

- initialize y^0 arbitrarily (e.g. everything label 0)
- repeat
  - for any $\alpha \in \mathcal{L}$
    - construct neighborhood:

$$\mathcal{N}(y) = \{(\bar{y}_1, \dots, \bar{y}_{|V|}) \ : \ \bar{y}_i \in \{y_i, \alpha\}\}$$

"each variable can keep its value or switch to α"

    - solve $y \leftarrow \mathbf{argmin}_{y \in \mathcal{N}(y)} E(y)$
- until y has not changed for a whole iteration

Multilabel Graph-Cut: α-expansion

Theorem **[Boykov *et al*. 2001]**
If all pairwise terms are *metric*, i.e. for all $(i, j) \in \mathcal{E}$

$$E_{ij}(k, l) \geq 0 \quad \text{with} \quad E_{ij}(k, l) = 0 \Leftrightarrow k = l$$

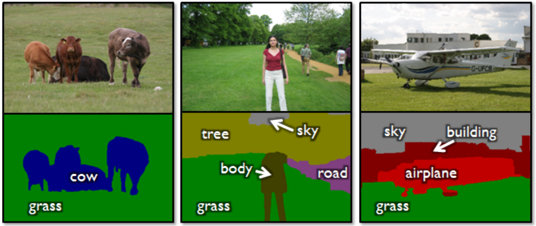
$$E_{ij}(k, l) = E_{ij}(l, k)$$

$$E_{ij}(k, l) \leq E_{ij}(k, m) + E_{ij}(m, l) \quad \text{for all } k, l, m$$

Then $\mathbf{argmin}_{y \in \mathcal{N}(y)} E(y)$ can be solved optimally using GraphCut.

Theorem **[Veksler 2001].** The solution, y_α, returned by α-expansion fulfills

$$E(y_\alpha) \;\leq\; 2c \cdot \min_{y \in \mathcal{Y}} E(y) \quad \text{for } c = \max_{(i,j) \in \mathcal{E}} \frac{\max_{k \neq l} E_{ij}(k, l)}{\min_{k \neq l} E_{ij}(k, l)}$$



object classes	building	grass	tree	cow	sheep	sky	airplane	water	face	car
bicycle	flower	sign	bird	book	chair	road	cat	dog	body	boat

$$E(y) = \sum_{i \in V} E_i(y_i) + \lambda \sum_{(i,j) \in \mathcal{E}} [\![y_i \neq y_j]\!]$$ "Potts model"

- $E_{ij}(k,l) \geq 0 \qquad E_{ij}(k,l) = 0 \;\Leftrightarrow\; k = l \qquad E_{ij}(k,l) = E_{ij}(l,k) \quad \checkmark$
- $E_{ij}(k,l) \leq E_{ij}(k,m) + E_{ij}(m,l) \qquad \checkmark$
- $c = \mathbf{max}_{(i,j) \in \mathcal{E}} \frac{\mathbf{max}_{k \neq l} E_{ij}(k,l)}{\mathbf{min}_{k \neq l} E_{ij}(k,l)} = 1$
- factor-2 approximation guarantee: $E(y_\alpha) \leq 2\, \mathbf{min}_{y \in \mathcal{Y}} E(y)$

Example: Stereo Estimation

$$E(y) = \sum_{i \in V} E_i(y_i) + \lambda \sum_{(i,j) \in \mathcal{E}} |y_i - y_j|$$

- $|y_i - y_j|$ is metric ✓
- $c = \mathbf{max}_{(i,j)\in\mathcal{E}} \frac{\mathbf{max}_{k \neq l} E_{ij}(k,l)}{\mathbf{min}_{k \neq l} E_{ij}(k,l)} = |\mathcal{L} - 1|$
- weak guarantees, but often close to optimal labelings in practice

Images: Middlebury stereo vision dataset

MAP Prediction / Energy Minimization – Summary

Task: compute $\mathbf{argmin}_{y \in \mathcal{Y}} E(x, y)$

Exact Energy Minimization

Only possible for certain models:

- trees/forests: max-sum belief propagation
- general graphs: junction chain algorithm (if tractable)
- submodular energies: GraphCut
- general graphs: integer linear programming (if tractable)

Approximate Energy Minimization

Many techniques with different properties and guarantees:

- linear programs relaxations
- ICM
- α-expansion

Best choice depends on model and requirements.

Summary

Graphical Models

Model probability distributions with explicit independencies

Conditional Random Fields

Log-linear models of conditional probability $p(y|x; w)$, ML training

Structured Support Vector Machine

Non-probabilistic structured prediction models, maximum margin training

Probabilistic Inference

Compute $p(y_F|x)$ for a subset F of variables: exactly or approximately

MAP Prediction / Energy Minimization

Compute $\mathbf{argmax}_y\, p(y|x)$: exactly or approximately

Structured Loss Function

Measure difference between two structured outputs (task-specific)