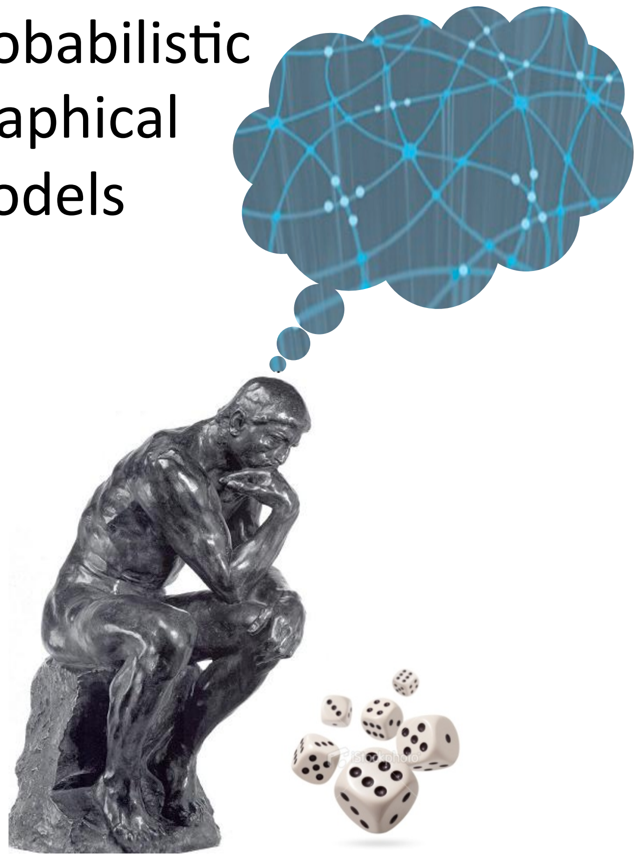
Probabilistic Graphical Models



Learning

Parameter Estimation

Max Likelihood for Log-Linear Models

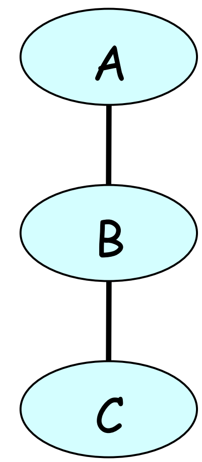
Log-Likelihood for Markov Nets

$P_\phi(a, b, c) = \frac{1}{Z}\, \phi_1(a,b) \cdot \phi_2(b,c)$

$$\ell(\boldsymbol{\theta} : \mathcal{D}) = \sum_m \left(\ln \phi_1(a[m], b[m]) + \ln \phi_2(b[m], c[m]) - \ln Z(\boldsymbol{\theta}) \right)$$

$$= \sum_{a,b} \boxed{M[a,b]} \ln \phi_1(a,b) + \sum_{b,c} \boxed{M[b,c]} \ln \phi_2(b,c) - M \ln Z(\boldsymbol{\theta})$$

$$Z(\boldsymbol{\theta}) = \sum_{a,b,c} \phi_1(a,b)\phi_2(b,c)$$



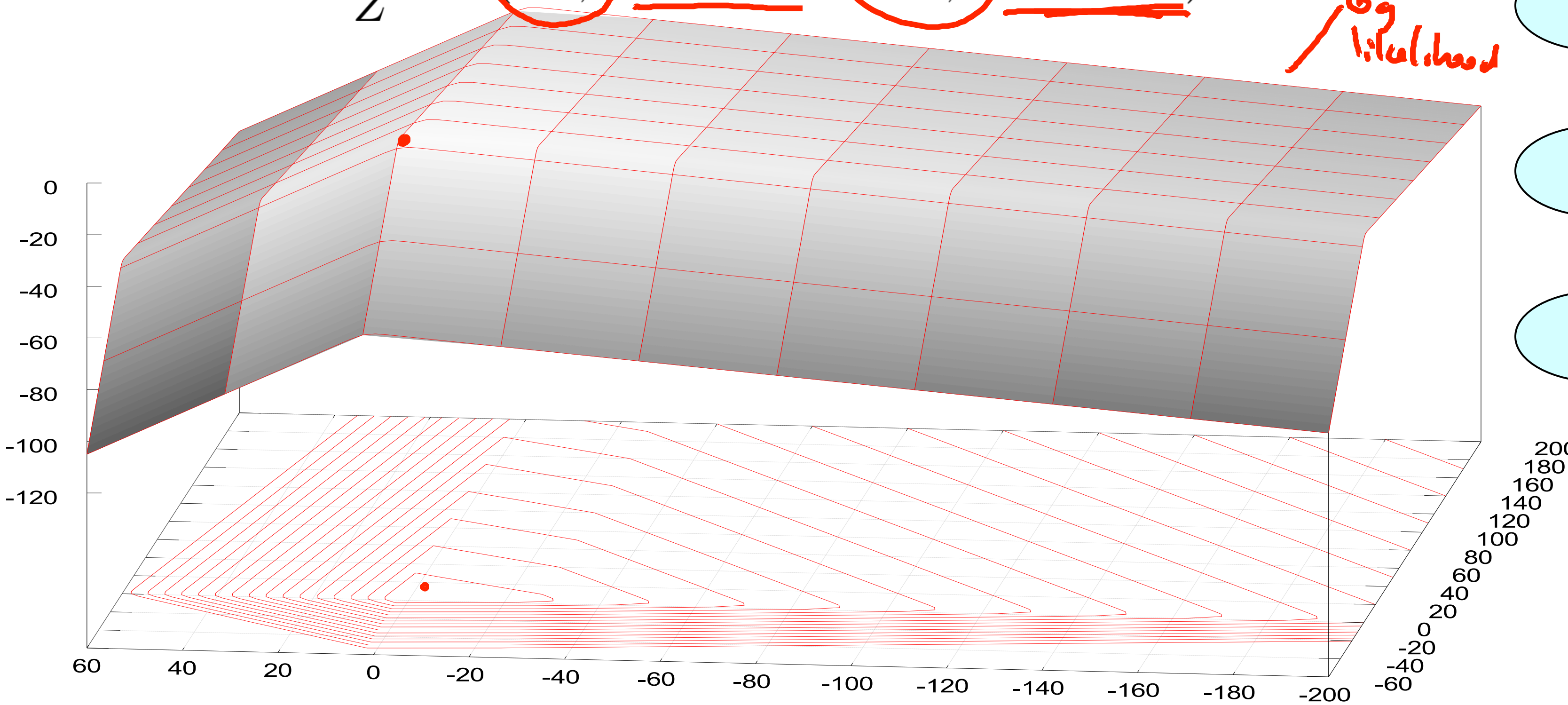
- Partition function couples the parameters
  - No decomposition of likelihood
  - No closed form solution

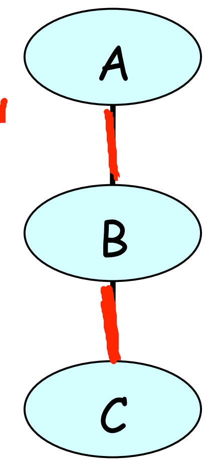
Daphne Koller

Example: Log-Likelihood Function

$$P_\Phi(A,B,C) = \frac{1}{Z}\exp\left(\theta_{a^1,b^1}\,\mathbb{1}\{a^1,b^1\} + \theta_{b^0,c^1}\,\mathbb{1}\{b^0,c^1\}\right)$$

log likelihood





Daphne Koller

Log-Likelihood for Log-Linear Model

parameters

features

$$P(X_1, \ldots, X_n : \boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \exp\left\{ \sum_{i=1}^{k} \theta_i f_i(\boldsymbol{D}_i) \right\}$$

partition function

$$\ell(\boldsymbol{\theta} : \mathcal{D}) = \sum_i \theta_i \left(\sum_m f_i(\boldsymbol{x}[m]) \right) - M \ln Z(\boldsymbol{\theta})$$

feature f_i applied to the m-th instance

$$\ln Z(\boldsymbol{\theta}) = \ln \sum_{\boldsymbol{x}} \exp\left\{ \sum_i \theta_i f_i(\boldsymbol{x}) \right\}$$

log-sum-exp

exponentially large space

Daphne Koller

The Log-Partition Function

vector of derivatives

Theorem: $$\frac{\partial}{\partial \theta_i} \ln Z(\boldsymbol{\theta}) = \boldsymbol{E}_{\boldsymbol{\theta}}[f_i]$$

expectation of f_i relative P_θ $\sum_x P_\theta(x) f_i(x)$

matrix (Hessian)

$$\frac{\partial^2}{\partial \theta_i \partial \theta_j} \ln Z(\boldsymbol{\theta}) = \boldsymbol{C}ov_{\boldsymbol{\theta}}[f_i; f_j]$$

$$\frac{\partial}{\partial \theta_i} \theta_j f_j = \begin{cases} 0 & i \neq j \\ f_i & i = j \end{cases}$$

Proof: $$\frac{\partial}{\partial \theta_i} \ln Z(\boldsymbol{\theta}) = \frac{1}{Z(\boldsymbol{\theta})} \sum_{\boldsymbol{x}} \frac{\partial}{\partial \theta_i} \exp\left\{ \sum_j \theta_j f_j(\boldsymbol{x}) \right\}$$

$$= \frac{1}{Z(\boldsymbol{\theta})} \sum_{\boldsymbol{x}} f_i(\boldsymbol{x}) \exp\left\{ \sum_j \theta_j f_j(\boldsymbol{x}) \right\}$$

$$= \sum_{\boldsymbol{x}} \boxed{\frac{1}{Z(\boldsymbol{\theta})} \exp\left\{ \sum_j \theta_j f_j(\boldsymbol{x}) \right\}} f_i(\boldsymbol{x}) = \sum_{\boldsymbol{x}} P_{\boldsymbol{\theta}}(\boldsymbol{x}) f_i(\boldsymbol{x})$$

Daphne Koller

The Log-Partition Function

Theorem:

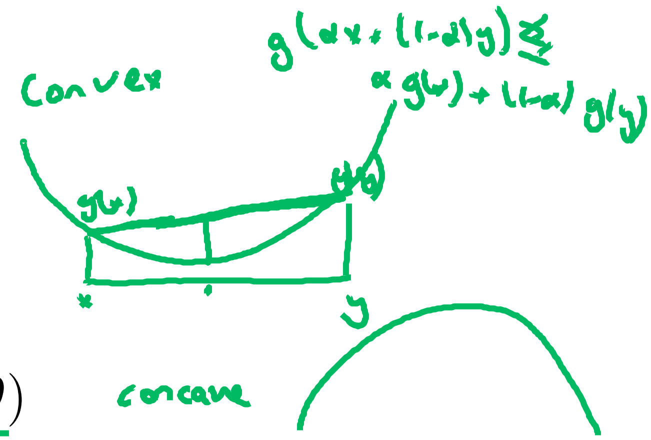
$$\frac{\partial}{\partial\theta_i}\ln Z(\boldsymbol{\theta}) = \boldsymbol{E}_{\boldsymbol{\theta}}[f_i]$$

Hessian

$$\frac{\partial^2}{\partial\theta_i\partial\theta_j}\ln Z(\boldsymbol{\theta}) = \boldsymbol{C}ov_{\boldsymbol{\theta}}[f_i;f_j]$$

$$\ell(\boldsymbol{\theta}:\mathcal{D}) = \underbrace{\sum_i \theta_i\left(\sum_m f_i(\boldsymbol{x}[m])\right)}_{\text{linear in }\theta} - \underline{M\ln Z(\boldsymbol{\theta})}$$

$\ell(\theta:D)$ concave function



- Log likelihood function
  - No local optima
  - Easy to optimize

Maximum Likelihood Estimation

$$\frac{1}{M}\ell(\boldsymbol{\theta}:\mathcal{D}) = \sum_i \theta_i \underbrace{\left(\frac{1}{M}\sum_m f_i(\boldsymbol{x}[m])\right)}_{\text{empirical expectation of } f_i \text{ in } D} - \ln Z(\boldsymbol{\theta})$$

$$\frac{\partial}{\partial\theta_i}\frac{1}{M}\ell(\boldsymbol{\theta}:\mathcal{D}) = \boldsymbol{E}_{\mathcal{D}}[f_i(\boldsymbol{X})] - \boldsymbol{E}_{\boldsymbol{\theta}}[f_i]$$

expectation of f_i in P_θ

Theorem: $\hat{\boldsymbol{\theta}}$ is the MLE if and only if

$\forall i$

$$\boldsymbol{E}_{\mathcal{D}}[f_i(\boldsymbol{X})] = \boldsymbol{E}_{\hat{\boldsymbol{\theta}}}[f_i]$$

expectation in D = expectation relative to $\hat{\theta}$

Daphne Koller

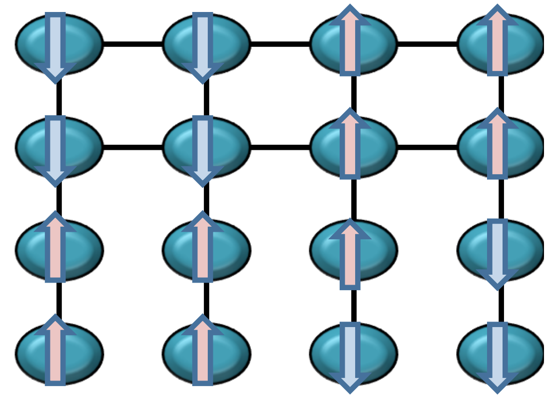
Computation: Gradient Ascent

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell(\boldsymbol{\theta} : \mathcal{D}) = \boxed{\boldsymbol{E}_{\mathcal{D}}[f_i(\boldsymbol{X})]} - \boxed{\boldsymbol{E}_{\boldsymbol{\theta}}[f_i]}$$

- Use gradient ascent:
  - typically L-BFGS - a quasi-Newton method
- For gradient, need expected feature counts:
  - in data
  - relative to current model
- Requires inference at each gradient step

Example: Ising Model

$$E(x_1, \ldots, x_n) = -\sum_{i<j} w_{i,j} \boxed{x_i x_j} - \sum_i u_i x_i$$



$x_i \in \{-1, +1\}$

$$\boxed{\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell(\boldsymbol{\theta} : \mathcal{D}) = \boldsymbol{E}_{\mathcal{D}}[f_i(\boldsymbol{X})] - \boldsymbol{E}_{\boldsymbol{\theta}}[f_i]}$$

$$\frac{\partial}{\partial u_i} = \frac{1}{M} \sum_m x_i[m] - (P_{\boldsymbol{\theta}}(\underbrace{X_i = 1}_{+1}) - P_{\boldsymbol{\theta}}(\underbrace{X_i = -1}_{-1}))$$

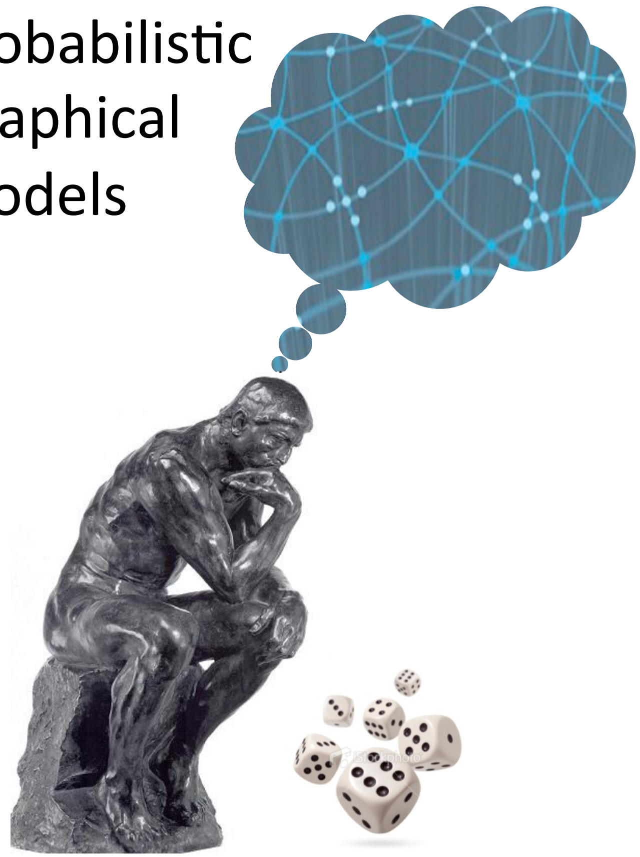
$$\frac{\partial}{\partial w_{ij}} = \frac{1}{M} \sum_m x_i[m] x_j[m] - \begin{pmatrix} P_{\boldsymbol{\theta}}(X_i = 1, X_j = 1) + P_{\boldsymbol{\theta}}(X_i = -1, X_j = -1) \\ -P_{\boldsymbol{\theta}}(X_i = 1, X_j = -1) - P_{\boldsymbol{\theta}}(X_i = 1, X_j = -1) \end{pmatrix}$$

Daphne Koller

Summary

- Partition function couples parameters in likelihood
- No closed form solution, but convex optimization
  - Solved using gradient ascent (usually L-BFGS) global opt.
- Gradient computation requires inference at each gradient step to compute expected feature counts
- Features are always within clusters in cluster-graph or clique tree due to family preservation
  - One calibration suffices for all feature expectations

Probabilistic
Graphical
Models



Learning

Parameter Estimation

Max Likelihood for CRFs

Estimation for CRFs

$$P_{\boldsymbol{\theta}}(\boldsymbol{Y} \mid \boldsymbol{x}) = \frac{1}{Z_{\boldsymbol{x}}(\boldsymbol{\theta})} \tilde{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{Y}) \qquad Z_{\boldsymbol{x}}(\boldsymbol{\theta}) = \sum_{\boldsymbol{Y}} \tilde{P}_{\boldsymbol{\theta}}(\boldsymbol{x}, \boldsymbol{Y})$$

log conditional likelihood

$$\mathcal{D} = \{(\boldsymbol{x}[m], \boldsymbol{y}[m])\}_{m=1}^{M} \qquad \ell_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{\theta} : \mathcal{D}) = \sum_{m=1}^{M} \ln P_{\boldsymbol{\theta}}(\boldsymbol{y}[m] \mid \boldsymbol{x}[m], \boldsymbol{\theta})$$

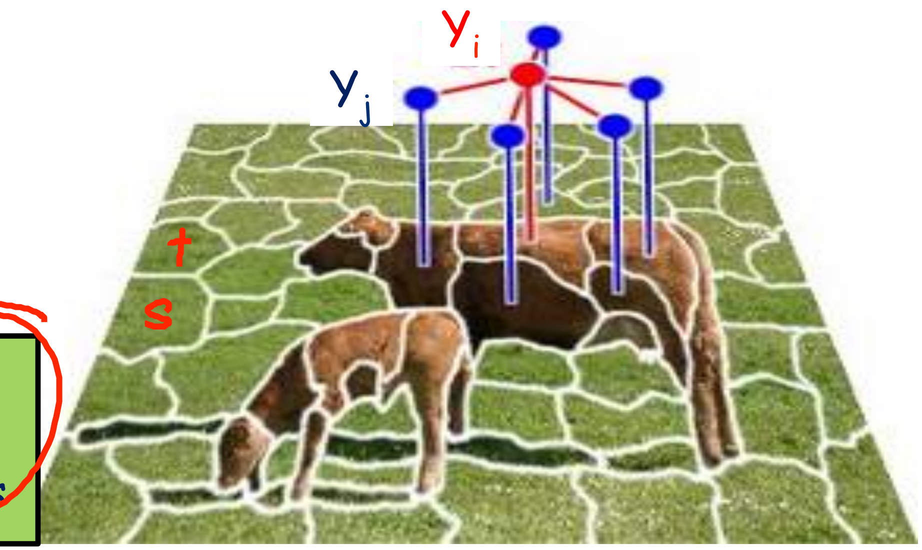
$$\ell_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{\theta} : (\boldsymbol{x}[m], \boldsymbol{y}[m])) = \left(\sum_i \theta_i f_i(\boldsymbol{x}[m], \boldsymbol{y}[m])\right) - \ln Z_{\boldsymbol{x}[m]}(\boldsymbol{\theta})$$

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell_{\mathbf{Y}|\mathbf{X}}(\boldsymbol{\theta} : \mathcal{D}) = \frac{1}{M} \sum_{m=1}^{M} \left(\boxed{f_i(\boldsymbol{x}[m], \boldsymbol{y}[m])} - \boxed{\boldsymbol{E}_{\boldsymbol{\theta}}[f_i(\boldsymbol{x}[m], \boldsymbol{Y})]}\right)$$

Y

Daphne Koller

Example



$f_1(Y_s, X_s) = \mathbf{1}(Y_s = g) \times G_s$

average intensity of green channel for pixels in superpixel s

$f_2(Y_s, Y_t) = \mathbf{1}(Y_s = Y_t)$

$$\frac{\partial}{\partial \theta_i} \ell_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{\theta} : (\boldsymbol{x}[m], \boldsymbol{y}[m])) = (f_i(\boldsymbol{x}[m], \boldsymbol{y}[m]) - \boldsymbol{E}_{\boldsymbol{\theta}}[f_i(\boldsymbol{x}[m], \boldsymbol{Y})])$$

m

$$\frac{\partial}{\partial \theta_1} = \sum_s \mathbf{1}\{y_s[m] = g\} G_s[m] - \sum_s P_{\boldsymbol{\theta}}(Y_s = g \mid \boldsymbol{x}[m]) G_s[m]$$

$$\frac{\partial}{\partial \theta_2} = \sum_{(s,t) \in \mathcal{N}} \mathbf{1}\{y_s[m] = y_t[m]\} - \sum_{(s,t) \in \mathcal{N}} P_{\boldsymbol{\theta}}(Y_s = Y_t \mid \boldsymbol{x}[m])$$

Daphne Koller

Computation

MRF

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell(\boldsymbol{\theta} : \mathcal{D}) = \boldsymbol{E}_{\mathcal{D}}[f_i(\boldsymbol{X})] - \boldsymbol{E}_{\boldsymbol{\theta}}[f_i]$$

- Requires inference at each gradient step

CRF

$$\frac{\partial}{\partial \theta_i} \frac{1}{M} \ell_{\boldsymbol{Y}|\boldsymbol{X}}(\boldsymbol{\theta} : \mathcal{D}) = \frac{1}{M} \sum_{m=1}^{M} \left(f_i(\boldsymbol{x}[m], \boldsymbol{y}[m]) - \boxed{\boldsymbol{E}_{\boldsymbol{\theta}}[f_i(\boxed{\boldsymbol{x}[m],} \boldsymbol{Y})]} \right)$$

- Requires inference for each x[m] at each gradient step

M = # training instances

Daphne Koller

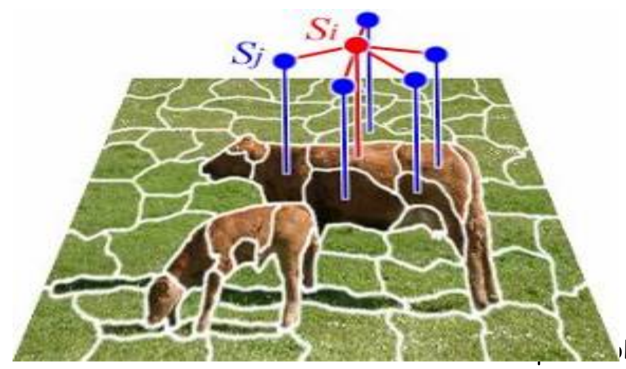
However…

- For inference of $P(\mathbf{Y} \mid \mathbf{x})$, we need to compute distribution only over $\mathbf{Y}$
- If we learn an MRF, need to compute $P(\mathbf{Y},\mathbf{X})$, which may be much more complex

$f_1(Y_s, X_s) = \mathbf{1}(Y_s = g) \times G_s$

$f_2(Y_s, Y_t) = \mathbf{1}(Y_s = Y_t)$

average intensity of green channel for pixels in superpixel i

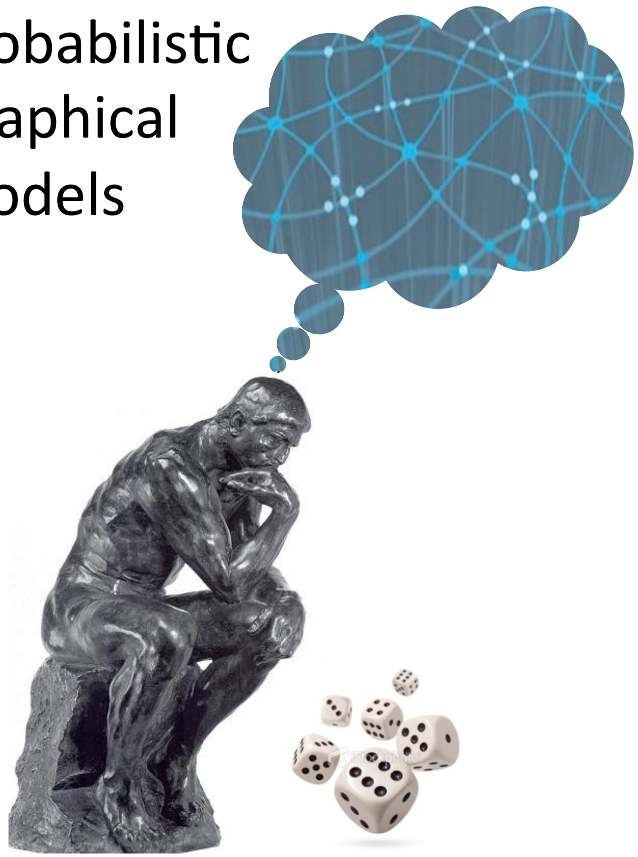


ller

Summary

- CRF learning very similar to MRF learning
  - Likelihood function is concave
  - Optimized using gradient ascent (usually L-BFGS)
- Gradient computation requires inference: one per gradient step, data instance
  - c.f., once per gradient step for MRFs
- But conditional model is often much simpler, so inference cost for CRF, MRF is not the same

Probabilistic
Graphical
Models



Learning

Parameter Estimation

MAP

Max a posteriori

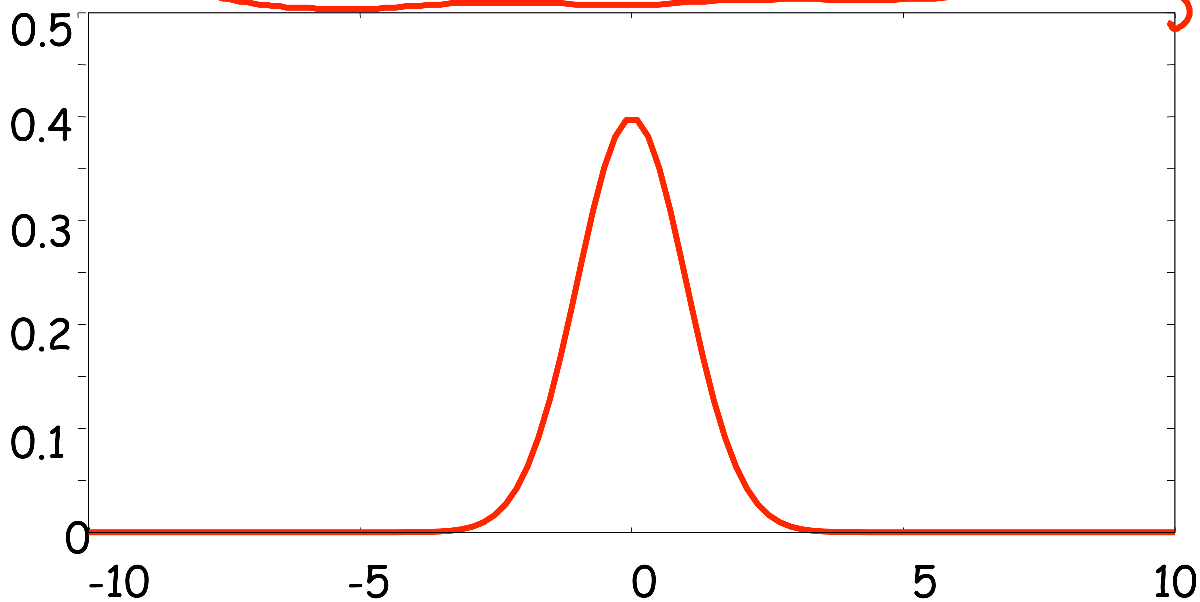
Estimation for MRFs, CRFs

Gaussian Parameter Prior

$$P(\boldsymbol{\theta} : \sigma^2) = \prod_{i=1}^{k} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{ -\frac{\theta_i^2}{2\sigma^2} \right\}$$

0-mean univariate Gaussian

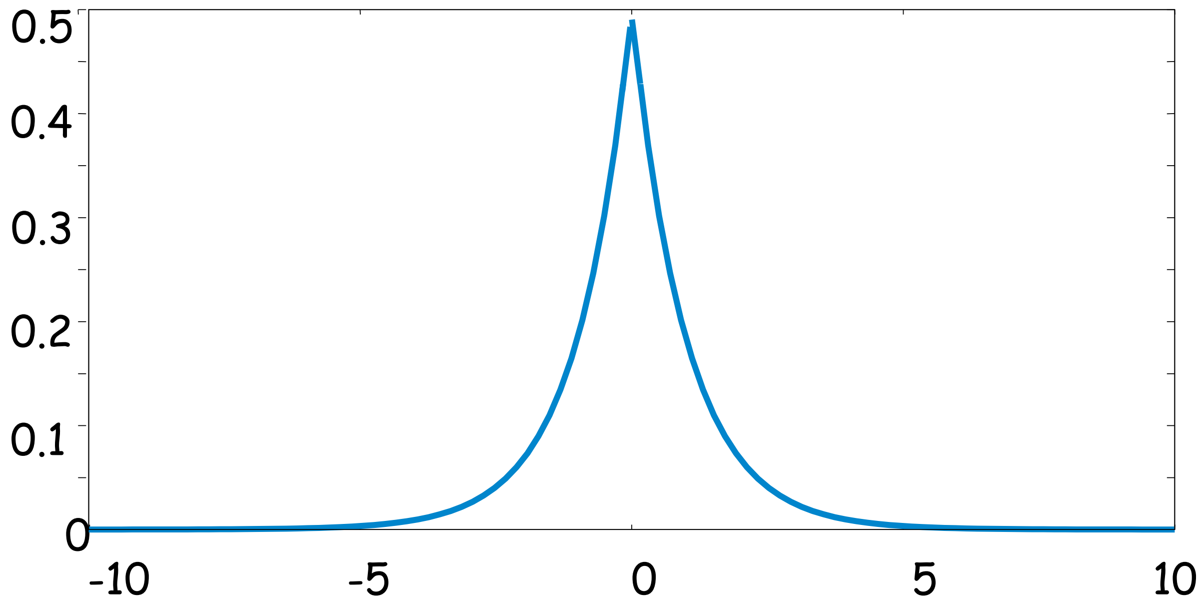
hyperparameter



Laplacian Parameter Prior

hyperparameter

$$P(\boldsymbol{\theta} ; \beta) = \prod_{i=1}^{k} \frac{1}{2\beta} \exp\left\{ -\frac{|\theta_i|}{\beta} \right\}$$



MAP Estimation & Regularization

$$P(\boldsymbol{\theta} : \sigma^2) = \prod_{i=1}^{k} \frac{1}{\sqrt{2\pi}\sigma} \exp\left\{-\frac{\theta_i^2}{2\sigma^2}\right\} \qquad P(\boldsymbol{\theta} : \beta) = \prod_{i=1}^{k} \frac{1}{2\beta} \exp\left\{-\frac{|\theta_i|}{\beta}\right\}$$

likelihood

prior

$$\underline{\mathrm{argmax}_{\boldsymbol{\theta}} \boxed{P(\mathcal{D}, \boldsymbol{\theta})} = \mathrm{argmax}_{\boldsymbol{\theta}}} \, \underline{P(\mathcal{D} \mid \theta)} P(\boldsymbol{\theta})$$

log likelihood

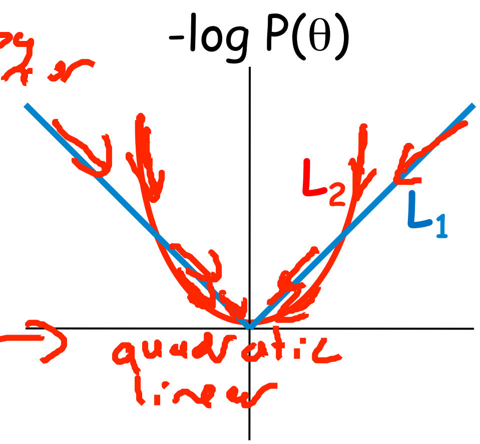
log prior

$$= \underline{\mathrm{argmax}_{\boldsymbol{\theta}}} \left(\underline{\ell(\boldsymbol{\theta} : \mathcal{D})} + \log P(\boldsymbol{\theta})\right)$$

many $\theta_i \neq 0$

dense → L2-regularization →

sparse L1- regularization



Daphne Koller

Summary

- In undirected models, parameter coupling prevents efficient Bayesian estimation
- However, can still use parameter priors to avoid overfitting of ~~MLE~~ MAP
- Typical priors are L_1, L_2
  - Drive parameters toward zero
- L_1 provably induces sparse solutions
  - Performs feature selection / structure learning