Probabilistic Graphical Models

Lecture 11 – CRFs, Exponential Family

CS/CNS/EE 155
Andreas Krause

Announcements

- Homework 2 due today
- Project milestones due next Monday (Nov 9)
  - About half the work should be done
  - 4 pages of writeup, NIPS format
  - http://nips.cc/PaperInformation/StyleFiles

So far

- Markov Network **Representation**
  - Local/Global Markov assumptions; Separation
  - Soundness and completeness of separation
- Markov Network **Inference**
  - Variable elimination and Junction Tree inference work exactly as in Bayes Nets
- How about **Learning** Markov Nets?

MLE for Markov Nets

- Log likelihood of the data

$$\log P(D \mid \theta) = \sum_{\ell} \log P(x^{(\ell)} \mid \theta)$$

$$= \sum_{\ell=1}^{m} \log \frac{1}{Z} \prod_i \psi_i(c_i^{(\ell)})$$

$$= \sum_{\ell} \sum_i \log \psi_i(c_i^{(\ell)}) - m \log Z$$

$$= m \sum_i \sum_{c_i} \hat{P}(c_i) \underbrace{\log \psi_i(c_i)}_{\text{in BN } \log P(X_i \mid Pa_i)} \underbrace{- m \log Z}_{\text{not in B}}$$

$$Z = Z(\theta) = \sum_x \prod_i \psi_i(c_i) \qquad \log Z = \log \sum_x \prod_i \psi_i(c_i)$$

wrt θ

Log-likelihood doesn't decompose

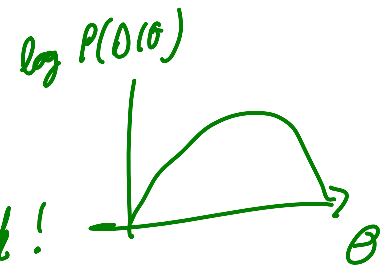
- Log likelihood

$$\log P(\mathcal{D} \mid \theta) = \underbrace{m \sum_i \sum_{\mathbf{c}_i} \hat{P}(\mathbf{c}_i) \log \psi_i(\mathbf{c}_i)}_{\text{decomposes nicely}} - \underbrace{m \log Z(\theta)}_{\text{does not decompose}}$$

- $l(D \mid \theta)$ is concave function!

No local optima!

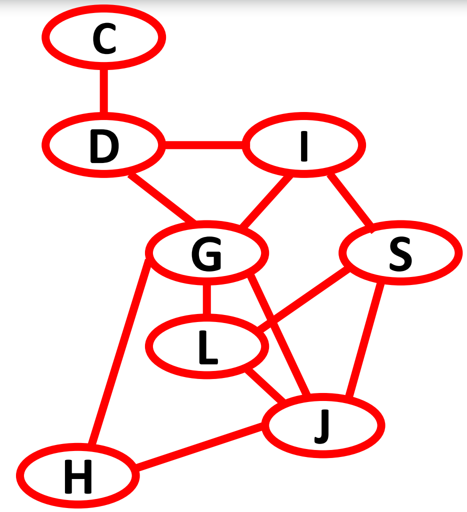
Gradient ascent won't get stuck!



- Log Partition function $\log Z(\theta)$ doesn't decompose

Computing the derivative

- Derivative



$$\frac{\partial \log P(\mathcal{D} \mid \theta)}{\partial \psi_i(\mathbf{c}_i)} = m \frac{\hat{P}(\mathbf{c}_i)}{\psi_i(\mathbf{c}_i)} - m \frac{P(\mathbf{c}_i \mid \theta)}{\psi_i(\mathbf{c}_i)}$$

$$\frac{\partial \log P(D \mid \theta)}{\partial \psi_G(G=1, D=0, I=1)} = m \frac{\hat{P}(1,0,1)}{\psi_i(1,0,1)} - m \frac{P(1,0,1 \mid \theta)}{\psi_i(1,0,1)}$$

Can do this using VE
Junction tree, ...

- Computing P(c_i | θ) requires inference!

- Can optimize using conjugate gradient etc.

Alternative approach: Iterative Proportional Fitting (IPF)

- At optimum, it must hold that

$$\frac{\partial \log P(\mathcal{D} \mid \theta)}{\partial \psi_i(\mathbf{c}_i)} = m\frac{\hat{P}(\mathbf{c}_i)}{\psi_i(\mathbf{c}_i)} - m\frac{P(\mathbf{c}_i \mid \theta)}{\psi_i(\mathbf{c}_i)} = 0$$

At opt.: $\frac{\hat{P}(c_i)}{\psi_i(c_i)} = \frac{P(c_i \mid \theta)}{\psi_i(c_i)}$ "Data agrees with model on marginals"

➔ Solve fixed point equation $\psi_i^{(0)}(c_i) = 1$

$$\psi_i^{(t+1)}(c_i) = \psi_i^{(t)}(c_i) \cdot \frac{\hat{P}(c_i)}{P(c_i \mid \theta)}$$

- Must recompute parameters every iteration

$P(c_i \mid \theta)$

Parameter learning for log-linear models

- Feature functions $\phi_i(C_i)$ defined over cliques
- Log linear model over undirected graph G
  - Feature functions $\phi_1(C_1),\ldots,\phi_k(C_k)$
  - Domains C_i can overlap
- Joint distribution

$$P(X_1,\ldots,X_n) = \frac{1}{Z}\underline{\exp\Big(\sum_i w_i^T \phi_i(C_i)\Big)}$$

- How do we get weights w_i?

Optimizing parameters

- Gradient of log-likelihood

$$\frac{\partial \log P(\mathcal{D} \mid w)}{\partial w_i} = \underbrace{m \sum_{\mathbf{c}_i} \hat{P}(\mathbf{c}_i)\phi_i(\mathbf{c}_i)}_{\hat{\mathbb{E}}(\phi_i)} - \underbrace{m \sum_{\mathbf{c}_i} P(\mathbf{c}_i \mid w)\phi_i(\mathbf{c}_i)}_{\mathbb{E}_w(\phi_i)}$$

- Thus, w is MLE ⇔ $\hat{\mathbb{E}}[\phi_i] = \mathbb{E}_w[\phi_i]$

Regularization of parameters

- Put prior on parameters w $P(w)$

$$\frac{\partial \log P(\mathcal{D} \mid w)P(w)}{\partial w_i} = \underbrace{m \sum_{\mathbf{c}_i} \hat{P}(\mathbf{c}_i)\phi_i(\mathbf{c}_i) - m \sum_{\mathbf{c}_i} P(\mathbf{c}_i \mid w)\phi_i(\mathbf{c}_i)}_{\text{last slide}} + \underbrace{\frac{\partial \log P(w)}{\partial w_i}}_{(*)}$$

Prior: $P(w) = \mathcal{N}(w; 0, I) \propto \exp\left(-\sum_i w_i^2\right)$

$(*)$ $\log P(w) = -\sum_i w_i^2$

$\frac{\partial}{\partial w_i} \log P(w) = -2w_i$

Summary: Parameter learning in MN

- MLE in BN is easy (score decomposes)
- MLE in MN requires inference (score doesn’t decompose)
- Can optimize using gradient ascent or IPF

Generative vs. discriminative models

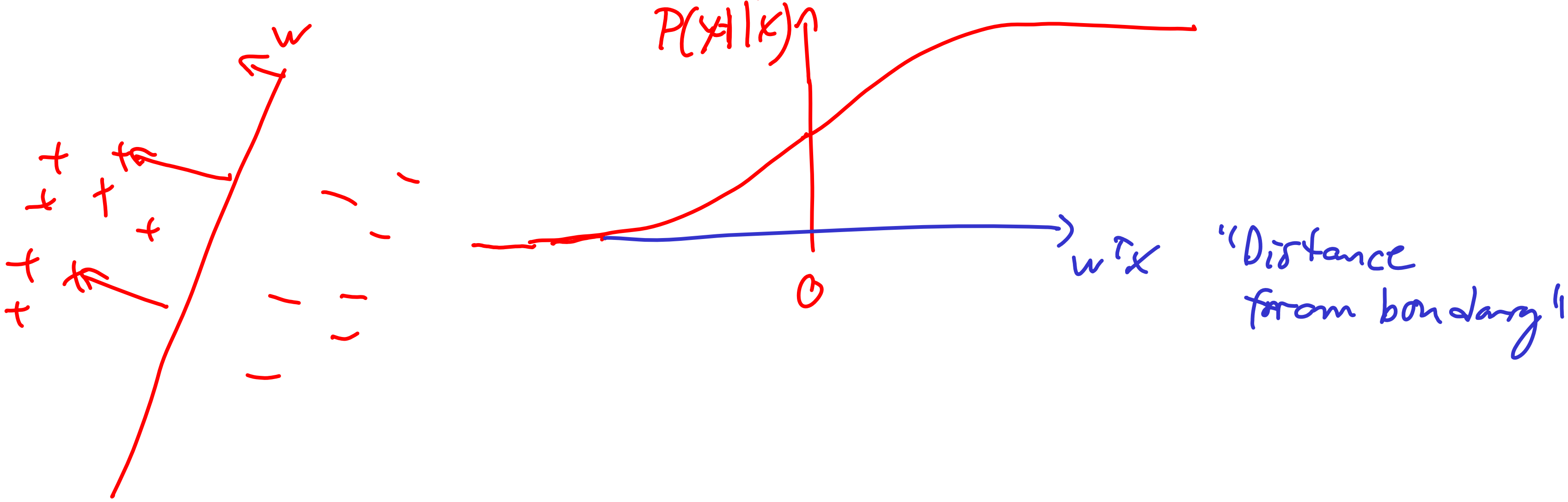
- Often, want to predict Y for inputs X
  - Bayes optimal classifier: Predict according to $P(Y \mid X)$
- Generative model
  - Model $P(Y)$, $P(X|Y)$
  - Use Bayes' rule to compute $P(Y \mid X)$
- Discriminative model
  - Model $P(Y \mid X)$ directly!
  - Don't model distribution $P(X)$ over inputs X
  - Cannot "generate" sample inputs
  - Example: Logistic regression

Example: Logistic Regression

Inputs $X_1 \dots X_n \in \mathbb{R}$

Output $Y \in \{0, 1\}$

$$P(Y=1 \mid X) = \frac{1}{1 + \exp(-w^T X)}$$



Log-linear conditional random field

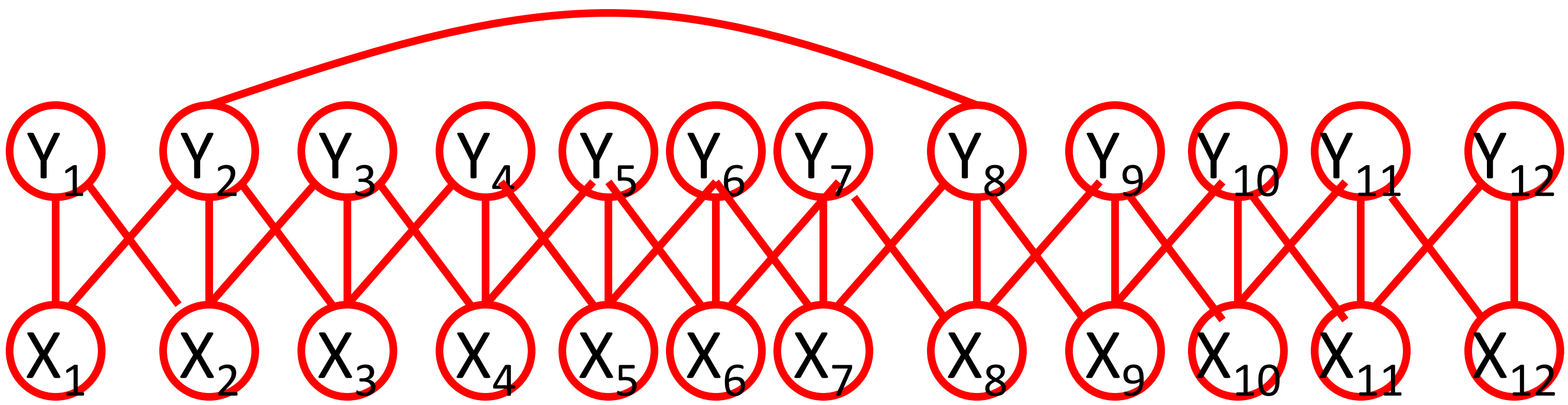
- Define log-linear model over outputs Y
  - No assumptions about inputs X
- Feature functions $\phi_i(C_i, x)$ defined over cliques and inputs

$C_i \subseteq Y$

- Joint distribution

$$P(Y_1, \ldots, Y_n \mid x) = \frac{1}{Z(x, w)} \exp\Big(\sum_i w_i^T \phi_i(C_i, x)\Big)$$

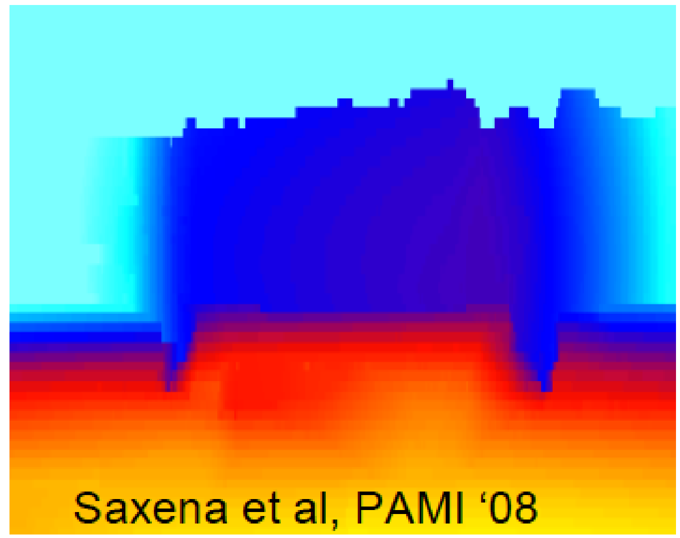
Example: CRFs in NLP



Mrs. Greene spoke today in New York. Green chairs the finance committee

- Classify into Person, Location or Other

Example: CRFs in vision



Parameter learning for log-linear CRF

$$P(Y_1, \ldots, Y_n \mid x) = \frac{1}{Z(x)} \exp\Big(\sum_i w_i^T \phi_i(C_i, x)\Big)$$

- Conditional log-likelihood of data

$$\log P(D_y \mid w, D_x) = \sum_\ell \sum_i w_i^T \phi_i(C_i, x^{(\ell)}) - \sum_\ell \log Z(x^{(\ell)})$$

- Can maximize using conjugate gradient

Gradient of conditional log-likelihood

- Partial derivative

$$\frac{\partial \log P(\mathcal{D}_Y \mid w, \mathcal{D}_X)}{\partial w_i} = \sum_j \left[\phi_i(\mathbf{c}_i^{(j)}, x^{(j)}) + \underbrace{\sum_{\mathbf{c}_i} P(\mathbf{c}_i \mid w, \underline{x^{(j)}}) \underline{\phi_i}(\mathbf{c}_i, \underline{x^{(j)}})}_{\text{Req. Inference}} \right]$$

- Requires one inference per feature <u>and</u> per data point

  Can be very expensive

  Can do "pseudo" likelihood est. (approximate)

- Can optimize using conjugate gradient

Exponential Family Distributions

- Distributions for log-linear models

$$P(X_1, \ldots, X_n) = \frac{1}{Z} \exp\left(\sum_i w_i^T \phi_i(C_i)\right)$$

- More generally: **Exponential family distributions**

$$P(x) = h(x) \exp\left(w^T \phi(x) - A(w)\right)$$

  - h(x): Base measure ← Often constant
  - w: natural parameters
  - ϕ(x): Sufficient statistics
  - A(w): log-partition function ← $A(w) = \log Z(w)$
- Hereby x can be continuous (defined over any set)

Examples

- Exp. Family:

$$P(x) = h(x)\exp\big(w^T\phi(x) - A(w)\big)$$

h(x): Base measure
w: natural parameters
ϕ(x): Sufficient statistics
A(w): log-partition function

- Gaussian distribution

$$P(x) = \frac{1}{\sqrt{2\pi\sigma^2}}\exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

$$-\frac{x^2}{2\sigma^2} + \frac{x\mu}{\sigma^2} - \frac{\mu^2}{2\sigma^2}$$

$$h(x) = \frac{1}{\sqrt{2\pi}}$$

$$\phi(x) = \left[-\frac{x^2}{2}, x\right], \quad A(w) = \frac{\mu^2}{2\sigma^2} - \log\sigma$$

$$w = \left[\frac{1}{\sigma^2}, \frac{\mu}{\sigma^2}\right],$$

- Other examples: Multinomial, Poisson, Exponential, Gamma, Weibull, chi-square, Dirichlet, Geometric, …

Moments and gradients

$$P(x) = h(x)\exp\big(w^T\phi(x) - A(w)\big)$$

- Correspondence between moments and log-partition function (just like in log-linear models)

$$\frac{\partial A(w)}{\partial w_i} = \int p(x \mid w)\phi_i(x)dx = \mathbb{E}[\phi_i \mid w]$$

$$\frac{\partial^2 A(w)}{\partial w_i \partial w_j} = Cov(\phi_i, \phi_j \mid w)$$

- Can compute moments from derivatives, and derivatives from moments!
- MLE ⇔ moment matching

Recall: Conjugate priors

- Consider parametric families of prior distributions:
  - P(θ) = f(θ; α)
  - α is called "hyperparameters" of prior
- A prior P(θ) = f(θ; α) is called **conjugate** for a likelihood function P(D | θ) if P(θ | D) = f(θ; α')
  - Posterior has same parametric form
  - Hyperparameters are updated based on data D
- Obvious questions (answered later):
  - How to choose hyperparameters??
  - Why limit ourselves to conjugate priors??

Conjugate priors in Exponential Family

$$P(x \mid w) = h(x) \exp\big(w^T \phi(x) - A(w)\big)$$

Any exponential family likelihood has a conjugate prior

$$P(w|\alpha, \beta) = \exp\big(\alpha^T w - \beta A(w) - B(\alpha, \beta)\big)$$

Maximum Entropy interpretation

Theorem: Exponential family distributions maximize the entropy over all distributions satisfying

$$\max_p - \int p(x) \log p(x) dx$$

$$\text{such that } t_i = \mathbb{E}[\phi_i(x)] = \int p(x)\phi_i(x) dx$$

Summary exponential family

- Distributions of the form

$$P(x) = h(x)\exp\big(w^T\phi(x) - A(w)\big)$$

- Most common distributions are exponential family
  - Multinomial, Gaussian Poisson, Exponential, Gamma, Weibull, chi-square, Dirichlet, Geometric, …
  - Log-linear Markov Networks
- All exponential family distributions have conjugate prior in EF
- Moments of sufficient stats = derivatives of log-partition function
- Maximum Entropy distributions ("most uncertain" distributions with specified expected sufficient statistics)

Exponential family graphical models

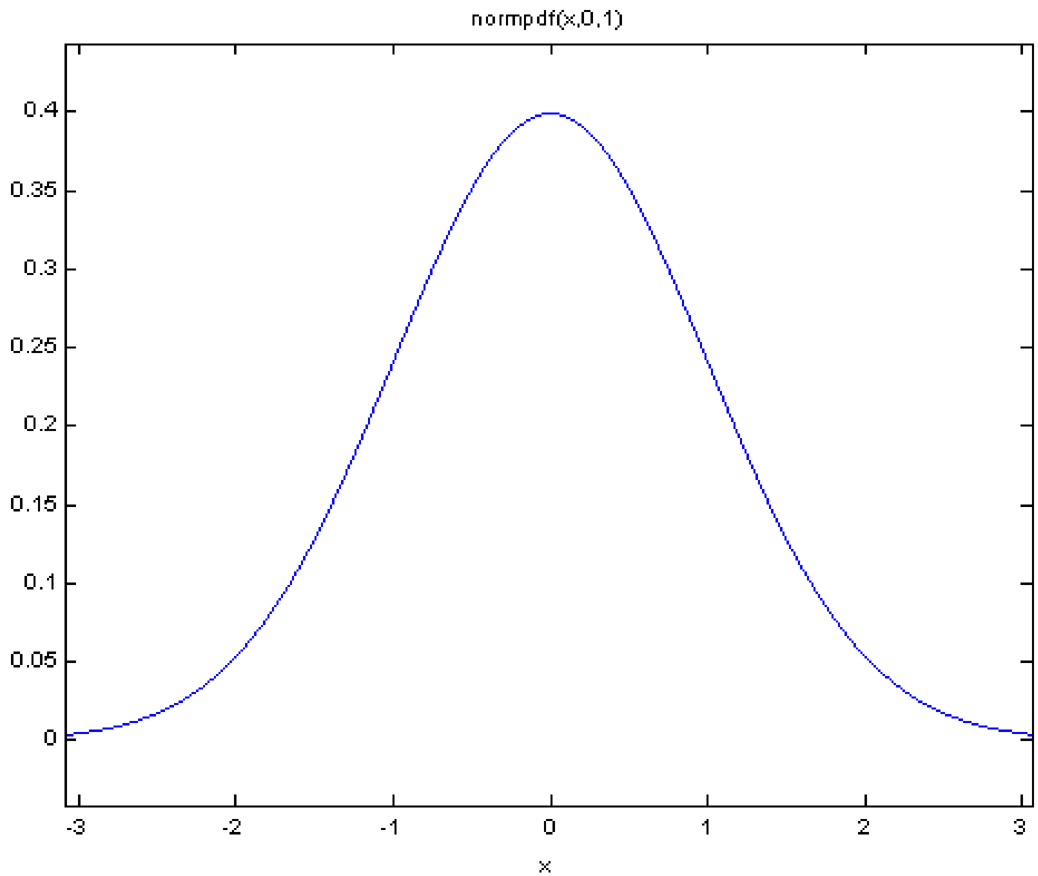
- So far, only defined graphical models over discrete variables.
- Can define GMs over continuous distributions!
- For exponential family distributions:

$$p(X_1, \ldots, X_n) = \prod_i h_i(C_i) \exp\Big(\sum_i w_i^T \phi_i(C_i) - A(w)\Big)$$

$$\exp A(w) = \int\int \ldots \int \prod_i h_i(C_i) \exp\Big(\sum_i w_i^T \phi_i(C_i) - A(w)\Big) dx_1 \ldots dx_n$$

- Can do much of what we discussed (VE, JT, parameter learning, etc.) for such exponential family models
- Important example: **Gaussian Networks**

Gaussian distribution

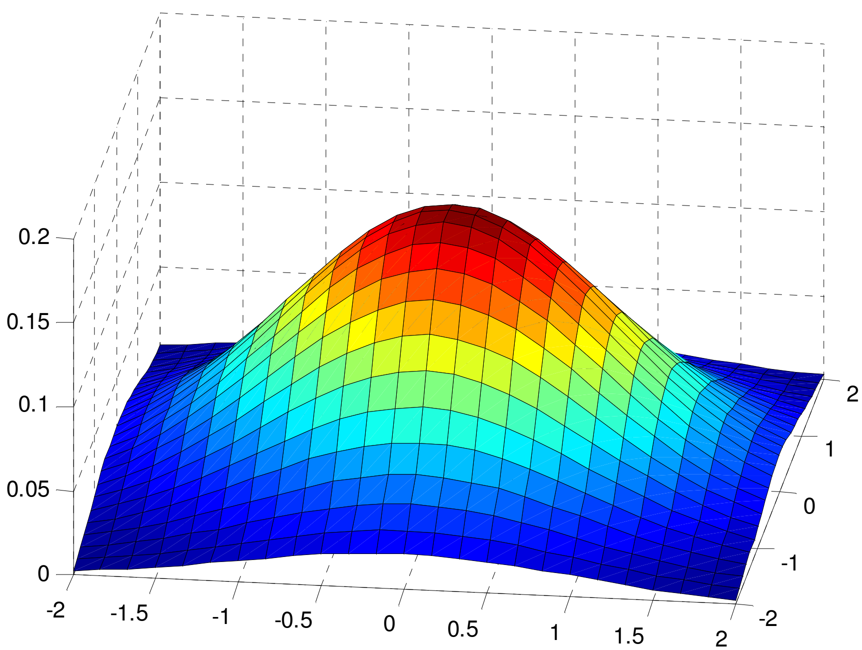


- σ = Standard deviation
- μ = mean

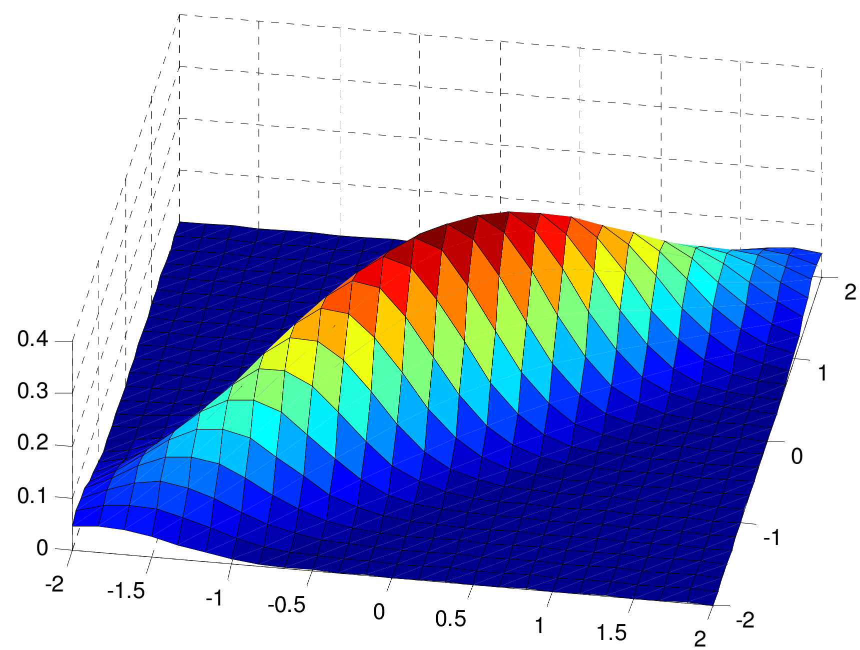
$$\frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{(x-\mu)^2}{2\sigma^2}\right)$$

Bivariate Gaussian distribution

$$\frac{1}{2\pi\sqrt{|\Sigma|}}\exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right) \qquad \Sigma = \begin{pmatrix}\sigma_1^2 & \sigma_{12}\\ \sigma_{21} & \sigma_2^2\end{pmatrix} \qquad \mu = \begin{pmatrix}\mu_1\\ \mu_2\end{pmatrix}$$



$$\Sigma = \begin{pmatrix}1 & 0\\ 0 & 1\end{pmatrix}$$



$$\Sigma = \begin{pmatrix}1 & 0.9\\ 0.9 & 1\end{pmatrix}$$

Multivariate Gaussian distribution

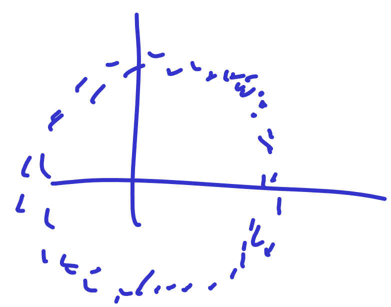
$$\mathcal{N}(x;\Sigma,\mu) = \frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T\Sigma^{-1}(x-\mu)\right)$$

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \dots & \sigma_{1n} \\ \vdots & & & \vdots \\ \sigma_{n1} & \sigma_{n2} & \dots & \sigma_n^2 \end{pmatrix} \qquad \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_n \end{pmatrix}$$

$\Sigma^T = \Sigma$

pos. def.: $x^T \Sigma x \geq 0 \quad \forall x$

- Joint distribution over n random variables $P(X_1, \ldots X_n)$
- $\sigma_{jk} = E[\,(X_j - \mu_j)\,(X_k - \mu_k)\,]$ $= Cov(X_j, X_k)$
- X_j and X_k independent $\Leftrightarrow$ $\sigma_{jk}=0$



Marginalization

- Suppose $(X_1,\ldots,X_n) \sim N(\mu, \Sigma)$
- What is $P(X_1)$??

Σ_{AA} μ_A

$$\Sigma = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1n} \\ \vdots & & & \vdots \\ \sigma_{n1} & \sigma_{n2} & \cdots & \sigma_n^2 \end{pmatrix} \quad \mu = \begin{pmatrix} \mu_1 \\ \mu_2 \\ \vdots \\ \mu_n \end{pmatrix}$$

$P(X_1) = \iiiint \cdots \int P(X_1 \ldots X_n)\, dx_2 \ldots dx_n$

$P(X_1) = \mathcal{N}(X_1; \mu_1, \sigma_1^2)$

- More generally: Let $A=\{i_1,\ldots,i_k\} \subseteq \{1,\ldots,N\}$
- Write $X_A = (X_{i_1},\ldots,X_{i_k})$

- $X_A \sim N(\mu_A, \Sigma_{AA})$

$$\Sigma_{AA} = \begin{pmatrix} \sigma_{i_1}^2 & \sigma_{i_1 i_2} & \cdots & \sigma_{i_1 i_k} \\ \vdots & & & \vdots \\ \sigma_{i_k i_1} & \sigma_{i_k i_2} & \cdots & \sigma_{i_k}^2 \end{pmatrix} \qquad \mu_A = \begin{pmatrix} \mu_{i_1} \\ \mu_{i_2} \\ \vdots \\ \mu_{i_k} \end{pmatrix}$$

Conditioning

- Suppose $(X_1,\ldots,X_n) \sim N(\mu, \Sigma)$
- Decompose as (X_A, X_B)
- What is $P(X_A \mid X_B)$??

$$\Sigma = \begin{pmatrix} \Sigma_{AA} & \Sigma_{AB} \\ \Sigma_{BA} & \Sigma_{BB} \end{pmatrix}$$

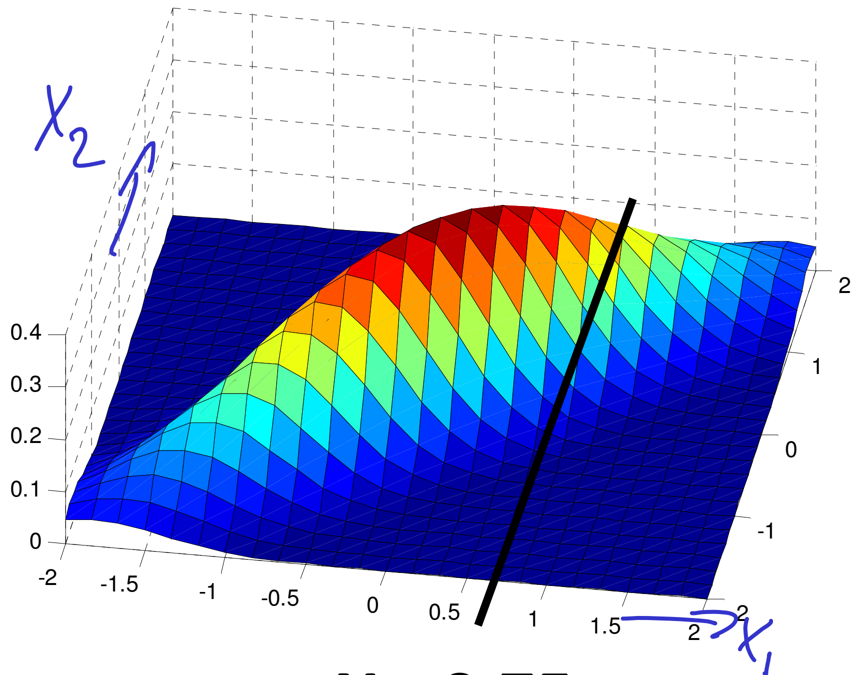
- $P(X_A = x_A \mid X_B = x_B) = N(x_A; \mu_{A|B}, \Sigma_{A|B})$ where

$$\boxed{\begin{aligned} \mu_{A|B=x_B} &= \underline{\mu_A} + \underbrace{\Sigma_{AB}\Sigma_{BB}^{-1}}_{W}(x_B - \mu_B) \\ \Sigma_{A|B} &= \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA} \end{aligned}}$$

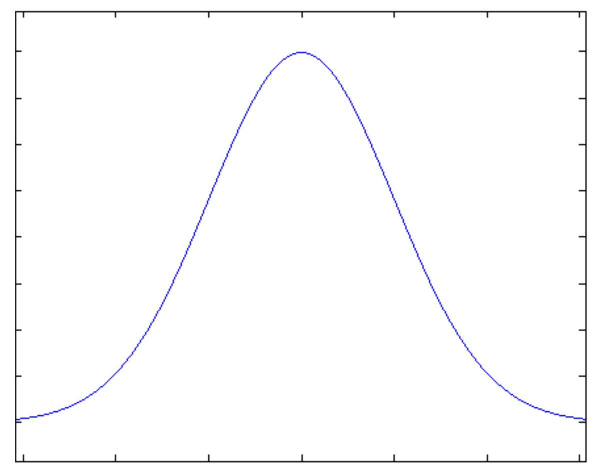
Does not depend on x_B

- Computable using linear algebra!

Conditioning



X_1=0.75



$P(X_2 \mid X_1$=0.75)

Conditional linear Gaussians

$$\mu_{A|B} = \mu_A + \underbrace{\Sigma_{AB}\Sigma_{BB}^{-1}}_{W}(x_B - \mu_B)$$

$$\Sigma_{A|B} = \Sigma_{AA} - \Sigma_{AB}\Sigma_{BB}^{-1}\Sigma_{BA}$$

$$P(x_A | x_B) = \mathcal{N}(x_A ; \mu_{A|B}, \Sigma_{A|B})$$

$$x_A = \mu_A + W x_B - W \mu_B + \varepsilon, \quad \varepsilon \sim \mathcal{N}(0, \Sigma_{A|B})$$

$$= W x_B + b + \varepsilon$$

$$W = \Sigma_{AB}\Sigma_{BB}^{-1}$$

$$b = \mu_A - W\mu_B$$

Canonical representation of Gaussians

$$p(X_1, \ldots, X_n) = \frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1} (x-\mu)\right)$$

$$\propto \exp\left(-\frac{1}{2} x^T \Sigma^{-1} x + \frac{1}{2} x^T \Sigma^{-1} \mu + \frac{1}{2} \mu^T \Sigma^{-1} x - \frac{1}{2} \mu^T \Sigma^{-1} \mu\right)$$

$$= \exp\left(-\frac{1}{2} x^T \Sigma^{-1} x + x^T \Sigma^{-1} \mu - A(\Sigma^{-1}, \mu)\right)$$

$$= \exp\left(-\frac{1}{2} x^T \Lambda x + x^T \eta\right)$$

precision matrix $\Lambda = \Sigma^{-1}$

$\eta = \Sigma^{-1} \mu$

Canonical Representation

$$p(X_1, \ldots, X_n) = \frac{1}{(2\pi)^{n/2}\sqrt{|\Sigma|}} \exp\left(-\frac{1}{2}(x-\mu)^T \Sigma^{-1}(x-\mu)\right)$$

$$\propto \exp(\eta^T x - \frac{1}{2} x^T \Lambda x) = \exp\left(\sum_i \eta_i x_i - \frac{1}{2}\sum_{i,j} x_i x_j \Lambda_{ij}\right)$$

- Multivariate Gaussians in exponential family!
- Standard vs canonical form:

$\mu = \Lambda^{-1} \eta$

$\Sigma = \Lambda^{-1}$

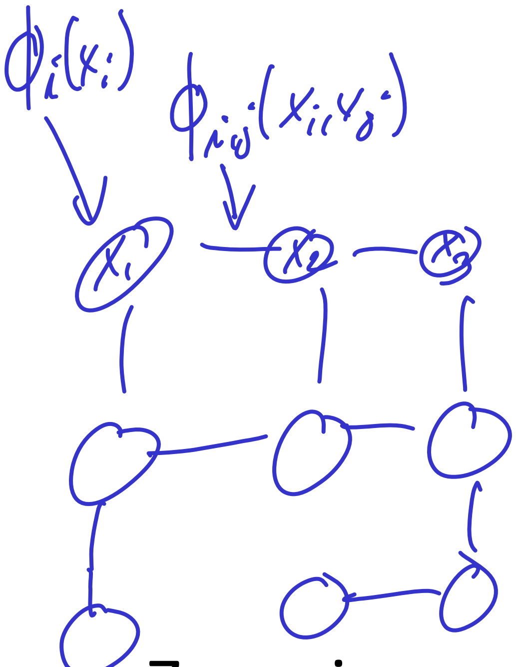
- In standard form: Marginalization is easy
- Will see: In canonical form, multiplication/conditioning is easy!

Gaussian Networks

$$p(X_1, \ldots, X_n) \propto \exp\Big(-\frac{1}{2}\sum_{i,j}\lambda_{i,j}x_i x_j + \sum_i \eta_i x_i\Big)$$

$$= \exp\Big(\sum_{i,j} \lambda_{i,j}\,\phi_{i,j}(x_i,x_j) + \sum_i \eta_i\,\phi_i(x_i)\Big)$$

$$\phi_{i,j}(x_i,x_j) = -\frac{1}{2}x_i x_j \qquad \phi_i(x_i) = x_i$$



No edge between X_i, X_j

$\Longleftrightarrow$

$\lambda_{i,j} = 0$

Zeros in precision matrix Λ indicate missing edges in log-linear model!

Inference in Gaussian Networks

- Can compute marginal distributions in $O(n^3)$!
- For large numbers n of variables, still intractable
- If Gaussian Network has low treewidth, can use variable elimination / JT inference!

- Need to be able to multiply and marginalize factors!

$$g = \int_{x_i} \prod_j f_j$$

Multiplying factors in Gaussians

$P(X_A) = \mathcal{N}(x_A; \Lambda_1, \eta_1)$ $|A| = k,\ |B| = m$

$\Lambda_1 \in \mathbb{R}^{k \times k}$

$P(X_B \mid X_A) \propto \mathcal{N}(x_B, x_A; \Lambda_2, \eta_2)$ ← CLG representation

$\Lambda_2 \in \mathbb{R}^{m \times m}$

$P(X_A, X_B) = P(X_A)\, P(X_B \mid X_A)$

$\propto \exp(x_A^T \Lambda_1 x_A + \eta_1^T x_A)\, \exp((x_A, x_B)^T \Lambda_2 (x_A, x_B) + \eta_2^T (x_A, x_B))$

$= \mathcal{N}(x_A, x_B; \Lambda, \eta)$

$\Lambda = \Lambda_1 + \Lambda_2 = \begin{pmatrix} \boxed{\Lambda_1} & \\ & 0 \end{pmatrix} + \begin{pmatrix} \Lambda_2 \end{pmatrix}$

$\eta = \eta_1 + \eta_2$

Marginalizing in canonical form

- Recall conversion formulas
  - $\mu = \Lambda^{-1} \eta$
  - $\Sigma = \Lambda^{-1}$
- Marginal distribution

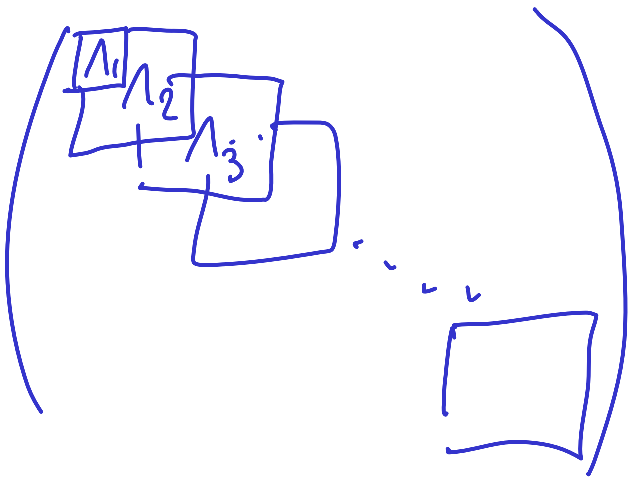
$$\eta_A^m = \eta_A - \Lambda_{AB}\Lambda_{BB}^{-1}\eta_B$$

$$\Lambda_{AA}^m = \Lambda_{AA} - \Lambda_{AB}\Lambda_{BB}^{-1}\Lambda_{BA}$$

Variable elimination

$$P(X_1, \ldots, X_n) = \underbrace{P(X_1)}_{\Lambda_1} \underbrace{P(X_2 \mid X_1)}_{\Lambda_2} \cdots \underbrace{P(X_n \mid X_{n-1})}_{\Lambda_n}$$

$$= \mathcal{N}(\cdot\, ; \Lambda, \eta)$$

$\Lambda =$ 

block diagonal

- In Gaussian Markov Networks, Variable elimination = Gaussian elimination (fast for low bandwidth = low treewidth matrices)

Tasks

- Read Koller & Friedman Chapters 4.6.1, 8.1-8.3