SAILING LAB
Laboratory for Statistical Artificial Intelligence & INtegrative Genomics

Learning Stochastic Models from Empirical Data

Eric Xing

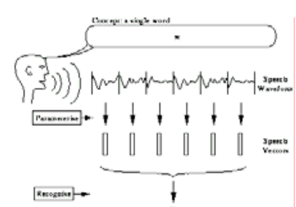
epxing@cs.cmu.edu

Machine Learning Dept./Language Technology Inst./Computer Science Dept.

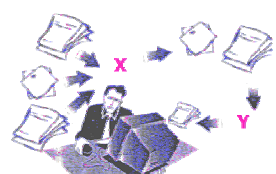
Carnegie Mellon University

SAILING LAB

Reasoning under Uncertainty!

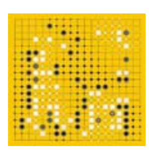


Speech recognition



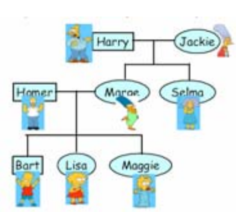
Information retrieval

Computer vision



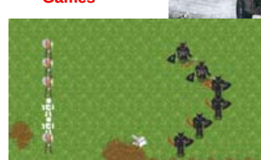
Games

Robotic control



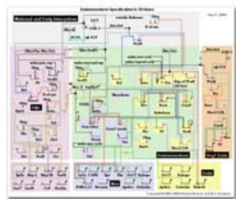
Pedigree

Evolution

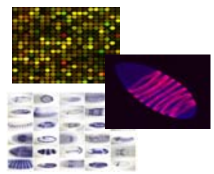


Planning

Statistical Inference & Learning

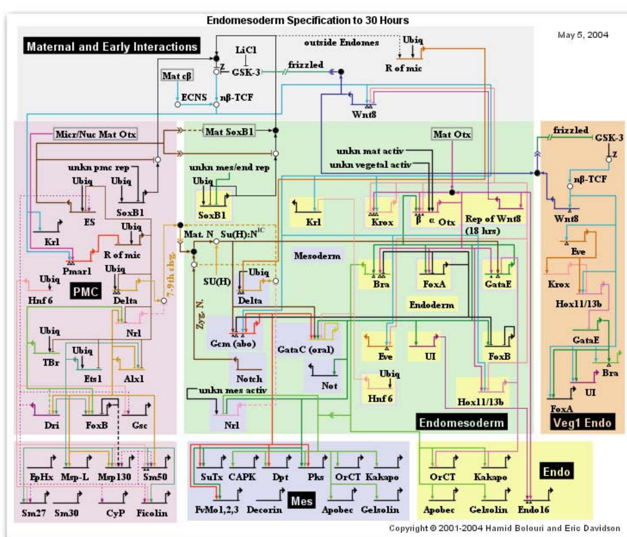


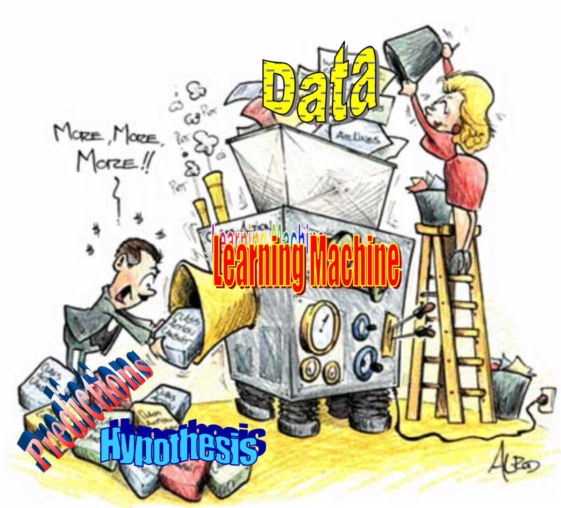
probabilistic generative model



gene expression profiles

Statistical Inference & Learning



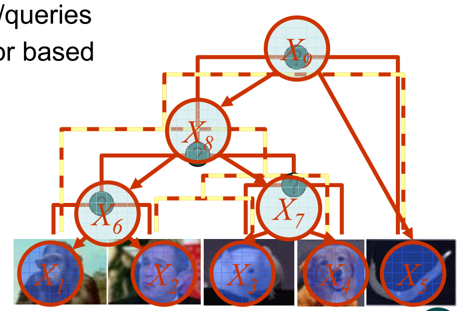


The Fundamental Questions

- Representation
  - How to encode our domain knowledge/assumptions/constraints?
  - How to capture/model uncertainties in possible worlds?
- Inference
  - How do I answers questions/queries according to my model and/or based given data?

    e.g.: $P(X_i \mid \boldsymbol{D})$
- Learning
  - What model is "right" for my data?

  e.g.: $\mathcal{M} = \arg\max_{\mathcal{M} \in \mathrm{M}} F(\boldsymbol{D}; \mathcal{M})$



Fitting Stochastic Models to Data

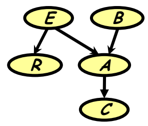
$(x_1^{(1)},\ldots,x_n^{(1)})$

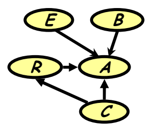
$(x_1^{(2)},\ldots,x_n^{(2)})$

$\ldots$

$(x_1^{(M)},\ldots,x_n^{(M)})$

Possible structures





⋮

Learn parameters

prediction error

Maximum likelihood

Bayesian

Conditional likelihood

Margin

…

Score struc/param

10^{-5}

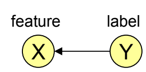
10^{-3}

10^{-15}

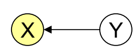
$\ldots$

Fitting Stochastic Models to Data

- Scenarios:
  - All variables are observed in all data (supervised)
  - Some observed, some not, in all data (unsupervised)
  - Some fully and some partially observed data (semi-supervised)



e.g., classification



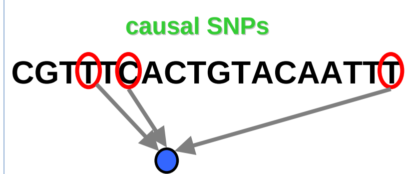
e.g., clustering

- Estimation principles (loss functions):
  - Least mean squared error of prediction
  - Maximal likelihood estimation (MLE)
  - Bayesian estimation
  - Maximal conditional likelihood
  - Maximal "Margin"

- We use **learning** as a name for the process of estimating the parameters, and in some cases, the structure of the model, from data.

E.g. Learning genome-trait association

The genetic picture



a univariate phenotype:

i.e., the expression intensity of a gene

The Data:

	Phenotype (BMI)	Genotype
Individual 1	2.5	..C.....T..C.......T... ..C.....A..C.......T...
Individual 2	4.8	..G.....A..G.......A... ..C.....T..C.......T...
⋮		
Individual N	4.7	..G.....T..C.......T... ..G.....T..G.......T...

Benign SNPs **Causal SNP**

SAILING LAB

Association Mapping as Regression

	Phenotype (BMI)	Genotype
Individual 1	2.5	. . 0 . . . . . 1 . . 0 . . . . 0 . . .
Individual 2	4.8	. . 1 . . . . . 1 . . 1 . . . . 1 . . .
⋮		
Individual N	4.7	. . 2 . . . . . 2 . . 1 . . . . 0 . . .

$$\mathbf{y}_i = g\left(\sum_{j=1}^{J} x_{ij}\theta_j\right)$$

SNPs with large $|\theta_j|$ are relevant

SAILING LAB

Linear Regression

- Assume that Y (target) is a linear function of X (features):
  - e.g.:

$$\begin{aligned}\hat{y}_i &= \theta_0 + \theta_1 x_{i,1} + \theta_2 x_{i,2} \\ &= \theta \cdot \mathbf{x}_i\end{aligned}$$

- Our goal is to pick the optimal θ that minimize the following **cost function**:

$$\begin{aligned}J(\theta) &= \frac{1}{2}\sum_{i=1}^{n}(\hat{y}_i(\mathbf{x}_i) - y_i)^2 \\ &= \frac{1}{2}\sum_{i=1}^{n}(\theta \cdot \mathbf{x}_i - y_i)^2\end{aligned}$$

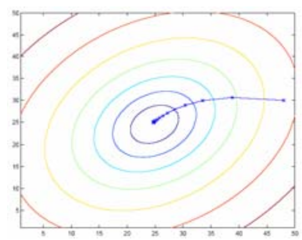
- This is known as the “least mean square” (LMS) estimate

Gradient Descent

- The LMS (coordinate descent) algorithm:

$$\theta_j^{t+1} = \theta_j^{\,t} - \alpha \frac{\partial}{\partial \theta_j} J(\theta)\Big|_t = \theta_j^{\,t} + \alpha \sum_{i=1}^{n} (y_i - \bar{\mathbf{x}}_i^{\,T} \theta^t) x_i^j$$

- Steepest descent:



$$\nabla_\theta J = \left[\frac{\partial}{\partial \theta_1} J, \ldots, \frac{\partial}{\partial \theta_k} J \right]^T = -\sum_{i=1}^{n} (y_n - \mathbf{x}_n^{\,T} \theta) \mathbf{x}_n$$

$$\theta^{t+1} = \theta^t + \alpha \sum_{i=1}^{n} (y_n - \mathbf{x}_n^{\,T} \theta^t) \mathbf{x}_n$$

- Normal equation: $\theta^* = \left(X^T X\right)^{-1} X^T \vec{y}$

Convergence rate

- **Theorem**: the steepest descent equation algorithm converge to the minimum of the cost characterized by normal equation:

$$\theta^{(\infty)} = (X^T X)^{-1} X^T y$$

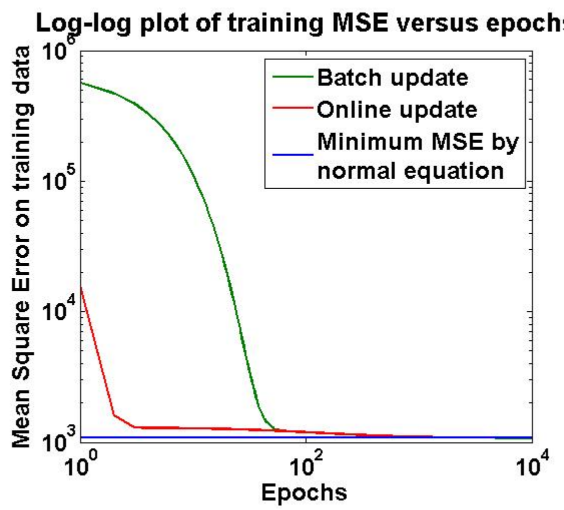
If

$$0 < \alpha < 2/\lambda_{\max}[X^T X]$$

- A formal analysis of LMS need more math-mussels; in practice, one can use a small α, or gradually decrease α.

Convergence Curves

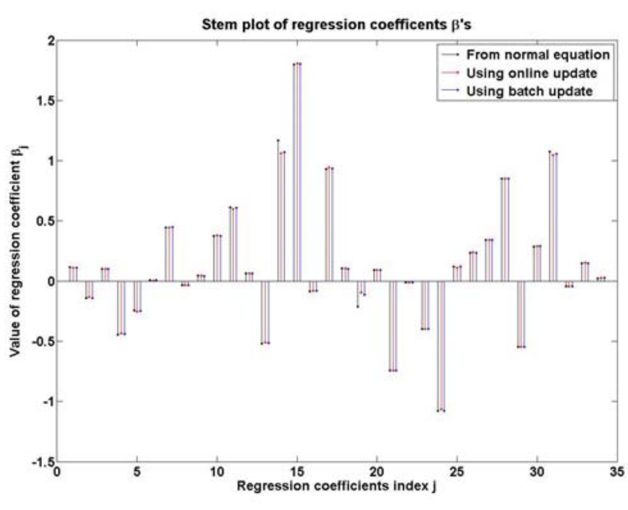
$$\theta_j^{t+1} = \theta_j^{t} + \alpha \sum_{i=1}^{n} (y_i - \vec{\mathbf{x}}_i^{T} \theta^{t}) x_i^{j}$$



- For the batch method, the training MSE is initially large due to uninformed initialization
- In the online update, N updates for every epoch reduces MSE to a much smaller value.

Network Workshop @ Santa Fe 12/5/2008 15

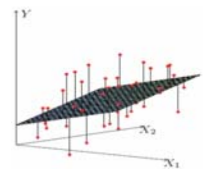
The Learned Coefficients



Network Workshop @ Santa Fe 12/5/2008 16

Probabilistic Interpretation of LMS

- Let us assume that the target variable and the inputs are related by the equation:



$$y_i = \theta^T \mathbf{x}_i + \varepsilon_i$$

where ε is an error term of unmodeled effects or random noise

- Now assume that ε follows a Gaussian $N(0, \sigma)$, then we have:

$$p(y_i \mid x_i; \theta) = \frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{(y_i - \theta^T \mathbf{x}_i)^2}{2\sigma^2} \right)$$

- By independence assumption:

$$L(\theta) = \prod_{i=1}^{n} p(y_i \mid x_i; \theta) = \left(\frac{1}{\sqrt{2\pi}\sigma} \right)^n \exp\left(-\frac{\sum_{i=1}^{n} (y_i - \theta^T \mathbf{x}_i)^2}{2\sigma^2} \right)$$

Network Workshop @ Santa Fe 12/5/2008 17

Probabilistic Interpretation of LMS

- Hence the log-likelihood is:

$$l(\theta) = n \log \frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2} \frac{1}{2} \sum_{i=1}^{n} (y_i - \theta^T \mathbf{x}_i)^2$$

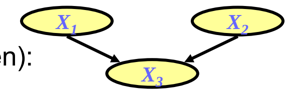
- Do you recognize the last term?

Yes it is: $$J(\theta) = \frac{1}{2} \sum_{i=1}^{n} (\mathbf{x}_i^T \theta - y_i)^2$$

- Thus under independence and Gaussian noise assumptions, LMS is equivalent to Maximum Likelihood Estimation (MLE) of θ !

Network Workshop @ Santa Fe 12/5/2008 18

The Basic Idea Underlying MLE



- Likelihood
  (for now let's assume that the structure is given):

  $L(\boldsymbol{\theta} \mid X) = p(X \mid \boldsymbol{\theta}) = p(X_1 \mid \theta_1) p(X_2 \mid \theta_2) p(X_3 \mid X_3, X_3; \theta_3)$

- Log-Likelihood:

  $l(\boldsymbol{\theta} \mid X) = \log p(X \mid \boldsymbol{\theta}) = \log p(X_1 \mid \theta_1) + \log p(X_2 \mid \theta_2) + \log p(X_3 \mid X_3, X_3, \theta_3)$

- Data log-likelihood

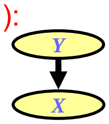
  $$l(\boldsymbol{\theta} \mid DATA) = \log \prod_n p(X_n \mid \boldsymbol{\theta})$$
  $$= \sum_n \log p(X_{n,1} \mid \theta_1) + \sum_n \log p(X_{n,2} \mid \theta_2) + \sum_n \log p(X_{n,3} \mid X_{n,1} X_{n,2}, \theta_3)$$

- MLE

  $$\{\theta_1, \theta_2, \theta_3\}_{MLE} = \arg\max l(\boldsymbol{\theta} \mid DATA)$$

$$\theta_1^* = \arg\max \sum_n \log p(X_{n,1} \mid \theta_1), \quad \theta_2^* = \arg\max \sum_n \log p(X_{n,2} \mid \theta_2), \quad \theta_3^* = \arg\max \sum_n \log p(X_{n,3} \mid X_{n,1} X_{n,2}, \theta_3)$$

E.X.: Gaussian Discriminative Classifier



- A Conditional Gaussian Model (completely observed):
  - Y is a class indicator vector

    $$Y = \begin{bmatrix} Y^1 \\ Y^2 \\ \vdots \\ Y^K \end{bmatrix}, \qquad \text{where } Y^k = [0,1], \text{ and } \sum_k Y^k = 1$$

    and a datum is in class i w.p. π_i

    $$p(y^i = 1 \mid \pi) = \pi_i = \pi_1^{y^1} \times \pi_2^{y^2} \times \ldots \times \pi_K^{y^K}$$

    All except one of these terms will be one

    $$p(y) = \prod_k \pi_k^{y^k}$$

  - X is a conditional Gaussian variable with a class-specific mean

    $$p(x \mid y^k = 1, \mu, \sigma) = \frac{1}{(2\pi\sigma^2)^{1/2}} \exp\left\{ -\tfrac{1}{2\sigma^2} (x - \mu_k)^2 \right\}$$

    $$p(x \mid y, \mu, \sigma) = \prod_k N(x \mid \mu_k, \sigma)^{y^k}$$

SAILING LAB

E.X.: Gaussian Discriminative Classifier

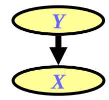
- Data log-likelihood

$$l(\boldsymbol{\theta} \mid D) = \log \prod_n p(y_n, x_n) = \log \prod_n p(y_n \mid \pi) p(x_n \mid y_n, \mu, \sigma)$$

$$= \sum_n \log p(y_n \mid \pi) + \sum_n \log p(x_n \mid y_n, \mu, \sigma)$$

$$= \sum_n \log \prod_k \pi_k^{y_n^k} + \sum_n \log \prod_k N(x_n \mid \mu_k, \sigma)^{y_n^k}$$

$$= \sum_n \sum_k y_n^k \log \pi_k - \sum_n \sum_k y_n^k \frac{1}{2\sigma^2}(x_n - \mu_k)^2 + C$$



- MLE

$$\pi_k^* = \arg\max l(\boldsymbol{\theta} \mid D), \qquad \Rightarrow \frac{\partial}{\partial \pi_k} l(\boldsymbol{\theta} \mid D) = 0, \forall k, \quad \text{s.t. } \sum_k \pi_k = 1$$

$$\Rightarrow \pi_k^* = \frac{\sum_n y_n^k}{N} = \frac{n_k}{N}$$

the fraction of samples of class *k*

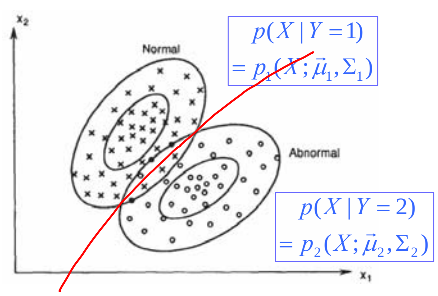
$$\mu_k^* = \arg\max l(\boldsymbol{\theta} \mid D), \qquad \Rightarrow \mu_k^* = \frac{\sum_n y_n^k x_n}{\sum_n y_n^k} = \frac{\sum_n y_n^k x_n}{n_k}$$

the average of samples of class *k*

SAILING LAB

Suppose you know the following …

- Class-specific Dist.: P(X|Y)



- Class prior (i.e., "weight"): P(Y)

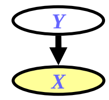
Bayes classifier:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

- This is a generative model of the data!

What if Z is not observed

- A *Mixture of* Gaussian Model (partially observed):
  - Z is a *latent* class indicator vector



$$p(y) = \text{multi}(y : \pi) = \prod_k (\pi_k)^{y^k}$$

  - X is a conditional Gaussian variable with a class-specific mean/covariance

$$p(x \mid y^k = 1, \mu, \Sigma) = \frac{1}{(2\pi)^{m/2} |\Sigma_k|^{1/2}} \exp\left\{-\tfrac{1}{2}(x - \mu_k)^T \Sigma_k^{-1} (x - \mu_k)\right\}$$

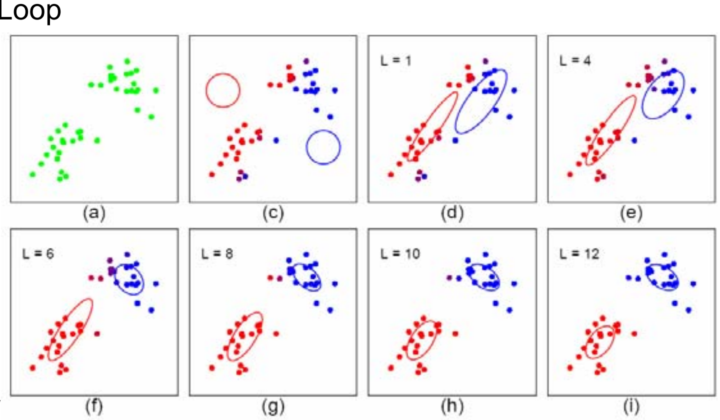
  - The likelihood of a sample:

$$\begin{aligned} p(x \mid \mu, \Sigma) &= \sum\nolimits_k p(y^k = 1 \mid \pi)\, p(x \mid y^k = 1, \mu, \Sigma) \\ &= \sum\nolimits_k \pi_k N(x \mid \mu_k, \Sigma_k) \end{aligned}$$

    - This objective is much harder to optimization than $p(x,y)$ w.r.t. the parameter

Expectation-Maximization

- Start:
  - "Guess" the centroid μ_k and coveriance Σ_k of each of the K clusters
- Loop



SAILING LAB

How is EM derived?

$$\cancel{\begin{aligned}\ell(\boldsymbol{\theta};x) &= \log p(x|\mu,\Sigma) \\ &= \log\left(\sum_k p(y^k=1|\pi)\,p(x\,|\,y^k=1,\mu,\Sigma)\right) \\ &= \log\left(\sum_k \pi_k N(x\,|\,\mu_k,\Sigma_k)\right)\end{aligned}}$$

- The complete log likelihood:

$$\begin{aligned}\ell(\boldsymbol{\theta};D) &= \log\prod_n p(y_n,x_n) = \log\prod_n p(y_n\,|\,\pi)p(x_n\,|\,y_n,\mu,\sigma) \\ &= \sum_n \log\prod_k \pi_k^{y_n^k} + \sum_n \log\prod_k N(x_n;\mu_k,\sigma)^{y_n^k} \\ &= \sum_n\sum_k y_n^k \log\pi_k - \sum_n\sum_k y_n^k \tfrac{1}{2\sigma^2}(x_n-\mu_k)^2 + C\end{aligned}$$

- The expected complete log likelihood

$$\begin{aligned}\left\langle \ell_c(\boldsymbol{\theta};x,y)\right\rangle &= \sum_n \left\langle \log p(y_n\,|\,\pi)\right\rangle_{p(z|x)} + \sum_n \left\langle \log p(x_n\,|\,y_n,\mu,\Sigma)\right\rangle_{p(z|x)} \\ &= \sum_n\sum_k \left\langle y_n^k\right\rangle \log\pi_k - \frac{1}{2}\sum_n\sum_k \left\langle y_n^k\right\rangle\left((x_n-\mu_k)^T\Sigma_k^{-1}(x_n-\mu_k)+\log|\Sigma_k|+C\right)\end{aligned}$$

Network Workshop @ Santa Fe 12/5/2008 25

SAILING LAB

E-step

- We maximize $\left\langle l_c(\boldsymbol{\theta})\right\rangle$ iteratively using the following iterative procedure:
  - Expectation step: computing the expected value of the sufficient statistics of the hidden variables (i.e., z) given current est. of the parameters (i.e., π and μ).

$$\tau_n^{k(t)} = \left\langle y_n^k\right\rangle_{q^{(t)}} = p(y_n^k=1\,|\,x,\mu^{(t)},\Sigma^{(t)}) = \frac{\pi_k^{(t)} N(x_n,|\,\mu_k^{(t)},\Sigma_k^{(t)})}{\sum_i \pi_i^{(t)} N(x_n,|\,\mu_i^{(t)},\Sigma_i^{(t)})}$$

  - Here we are essentially doing **inference**

Network Workshop @ Santa Fe 12/5/2008 26

SAILING LAB

M-step

- We maximize $\langle l_c(\theta) \rangle$ iteratively using the following iterative procedure:
  - Maximization step: compute the parameters under current results of the expected value of the hidden variables

$$\pi_k^* = \arg\max \langle l_c(\boldsymbol{\theta}) \rangle, \qquad \Rightarrow \pi_k^* = \frac{\sum_n \langle y_n^k \rangle_{q^{(t)}}}{N} = \frac{\sum_n \tau_n^{k(t)}}{N} = \frac{\langle n_k \rangle}{N}$$

$$\mu_k^* = \arg\max \langle l(\boldsymbol{\theta}) \rangle, \qquad \Rightarrow \mu_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} x_n}{\sum_n \tau_n^{k(t)}}$$

$$\Sigma_k^* = \arg\max \langle l(\boldsymbol{\theta}) \rangle, \qquad \Rightarrow \Sigma_k^{(t+1)} = \frac{\sum_n \tau_n^{k(t)} (x_n - \mu_k^{(t+1)})(x_n - \mu_k^{(t+1)})^T}{\sum_n \tau_n^{k(t)}}$$

- This is isomorphic to **MLE** except that the variables that are hidden are replaced by their expectations (in general they will by replaced by their corresponding "**sufficient statistics**")

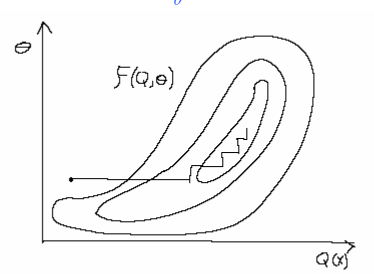
SAILING LAB

Lower Bounds and Free Energy

- For fixed data x, define a functional called the free energy:

$$F(q,\theta) \stackrel{\text{def}}{=} \sum_z q(z \mid x) \log \frac{p(x,z \mid \theta)}{q(z \mid x)} \le \ell(\theta; x)$$

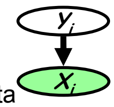
- The EM algorithm is coordinate-ascent on F:
  - **E-step:** $q^{t+1} = \arg\max_q F(q, \theta^t)$
  - **M-step:** $\theta^{t+1} = \arg\max_\theta F(q^{t+1}, \theta^t)$



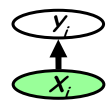
Generative vs. Discriminative Models

- Goal: Wish to learn f: X → Y, e.g., P(Y|X)

- Generative classifiers (e.g., Naïve Bayes):
  - Assume some functional form for P(X|Y), P(Y)
    This is a ***'generative'*** model of the data!
  - Estimate parameters of P(X|Y), P(Y) directly from training data
  - Use Bayes rule to calculate P(Y|X= x)



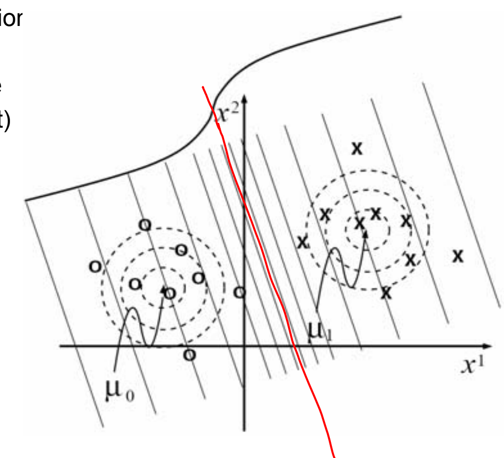
- Discriminative classifiers:
  - Directly assume some functional form for P(Y|X)
    This is a ***'discriminative'*** model of the data!
  - Estimate parameters of P(Y|X) directly from training data



Generative vs. Discriminative Models

- Generative:
  - Modeling the joint distributior of all data
  - → Gaussian discriminative analysis (we've just seen it)

- Discriminative:
  - Modeling only points at the boundary
  - Logistic regression



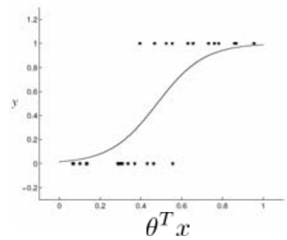
Logistic Regression

- The condition distribution: a Bernoulli

$$p(y \mid x) = \mu(x)^{y}(1-\mu(x))^{1-y}$$

where μ is a logistic function

$$\mu(x) = \frac{1}{1+e^{-\theta^T x}}$$



- Estimate parameters θ=<θ_0, θ_1, ... θ_m> to maximize the conditional likelihood of training data $\mathcal{D} = \{(x_1, y_1), \ldots, (x_N, y_N)\}$

- Data conditional likelihood = $\prod_{i=1}^{N} P(y_i|x_i;\theta)$

$$\theta = \arg\max_{\theta} \ln \prod_{i} P(y_i|x_i;\theta)$$

Generative vs. Discriminative Models

- Under naïve Bayes assun

Gaussian discriminative

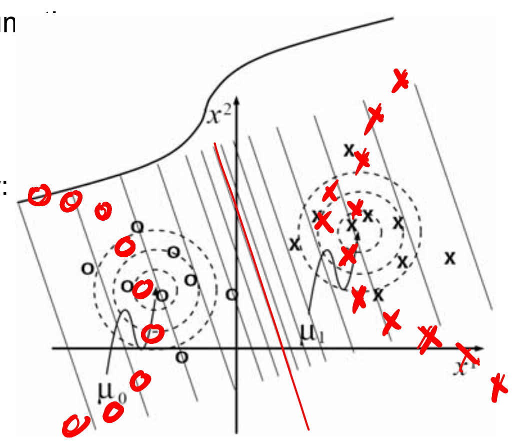
and

Logistic regression

give the same decision law:

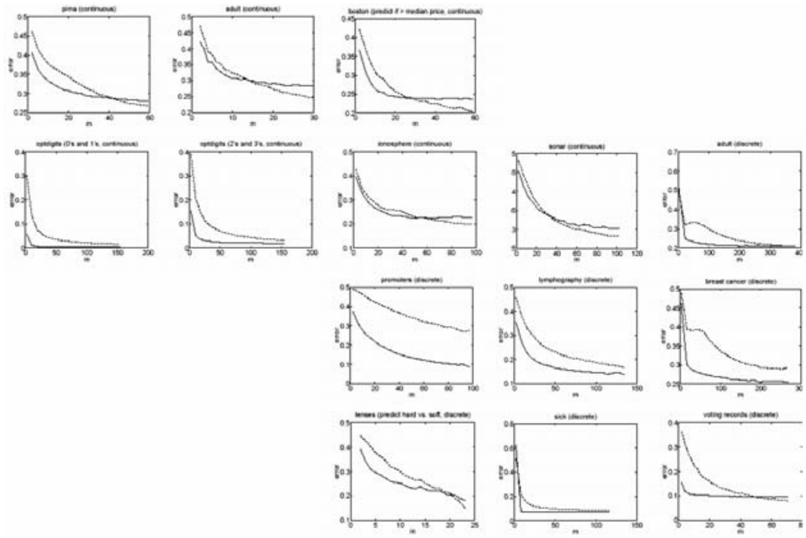
$$p(y_n^1 = 1 \mid x_n)$$

$$= \frac{1}{1+e^{-\theta^T x_n}}$$

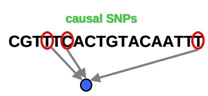


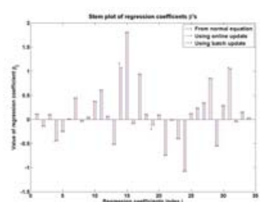
But LR relies much less of assumption of data distribution (because it only focus on the decision boundary)

NB and LR on 14 Data Sets



Regularization





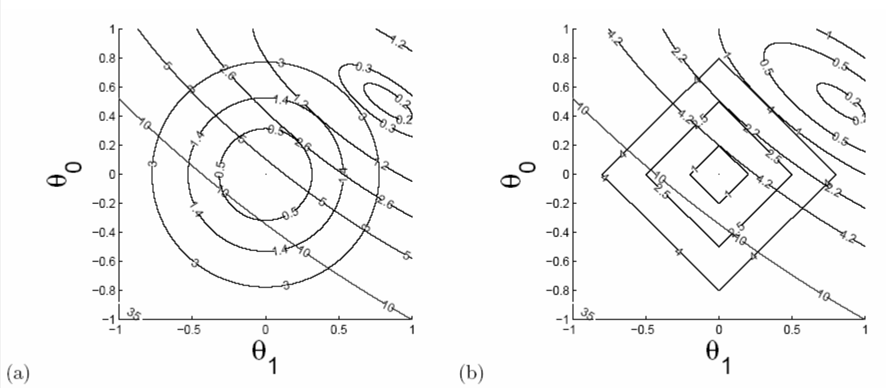
- Maximum-likelihood estimates are not always t and Stein showed a counter example in the ea
- Alternative: we "regularize" the likelihood objective (also known as penalized likelihood, shrinkage, smoothing, etc.), by adding to it a penalty term:

$$\hat{\theta}_{\text{shrinkage}} = \arg\max_{\theta}\left[l(\theta; D) + \lambda\|\theta\|\right]$$

where λ>0 and $||\theta||$ might be the L_1 or L_2 norm.

- The choice of norm has an effect
  - using the L_2 norm pulls directly towards the origin,
  - while using the L_1 norm pulls towards the coordinate axes, i.e it tries to set some of the coordinates to 0.
  - This second approach can be useful in a feature-selection setting.

L_2 vs. L_1 Regularization



Bayesian Interpretation of Regulation

- Regularized Linear Regression
  - Recall that LMS with Gaussian noise is equivalent to MLE of θ

$$l(\theta) = n\log\frac{1}{\sqrt{2\pi}\sigma} - \frac{1}{\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y_i - \theta^T\mathbf{x}_i)^2$$

  - Now assume that vector θ follows a normal prior with 0-mean and a diagonal covariance matrix $\theta \sim N(0, \tau^2 I)$
  - The *posterior distribution* of θ is:

$$p(\theta|D) \propto p(D|\theta)p(\theta) = (2\pi\sigma^2)^{-n/2}\exp\left\{-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_n - \theta^T x_i)^2\right\}\times C\exp\{-(\theta^T\theta/2\tau^2\}$$

  - This leads to a now objective

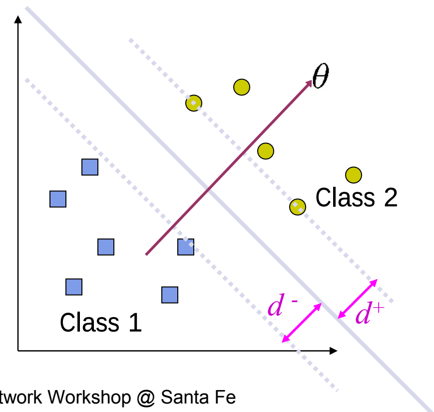
$$l_{MAP}(\theta; D) = -\frac{1}{2\sigma^2}\frac{1}{2}\sum_{i=1}^{n}(y_i - \theta^T\mathbf{x}_i)^2 - \frac{1}{\tau^2}\frac{1}{2}\sum_{k=1}^{K}\theta_k^2$$
$$= l(\theta; D) + \lambda\|\theta\|$$

    - This is L_2 regularized LR! --- a MAP estimation of θ
    - L_1 regularized LR correspond to a MAP est. under a Laplace prior of θ

Classification and Margin

- We can represent a linear decision boundary as:

$$\theta^T x + b = 0$$

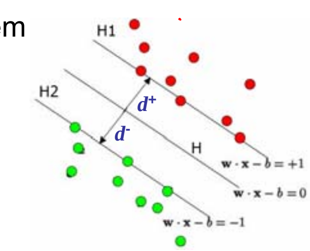


- The Margin between the points closest to the decision boundaries is:

$$m = \frac{2c}{\|\theta\|}$$

Support Vector Machine

- A convex quadratic programming problem with linear constrains:



$$\min_{\theta,b} \quad \frac{1}{2}\theta^T\theta + C\sum_{i=1}^{m}\xi_i$$

$$\text{s.t} \quad \begin{array}{l} y_i(\theta^T x_i + b) \geq 1 - \xi_i, \quad \forall i \\ \xi_i \geq 0, \quad \forall i \end{array}$$

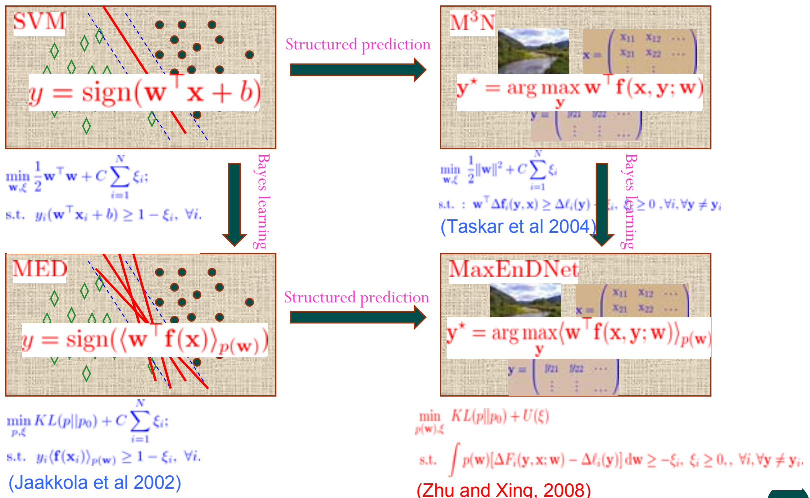
  - Only a few of the classification constraints are relevant ➔ **support vectors**

- Constrained optimization
  - We can directly solve this using commercial quadratic programming (QP) code
  - But we want to take a more careful investigation of Lagrange duality, and the solution of the above in its dual form.

  ➔ deeper insight: support vectors, kernels …

  ➔ more efficient algorithm

Margin-based Learning Paradigms



Probabilistic Inference

- Computing statistical queries regarding the model, e.g.:
  - What is the probability of X=true if (Y=false and Z=true)?
  - What is the joint distribution of (X,Y) if Z=false?
  - What is the likelihood of some full assignment?
  - What is the most likely assignment of values to all or a subset the nodes of the network?
  - **Inferring hidden variables (recall EM!!)**
- General purpose algorithms exist to fully automate such computation
  - Computational cost depends on the structure of the model
  - Exact inference:
    - The junction tree algorithm
  - **Approximate inference;**
    - Loopy belief propagation, mean-field inference, Monte Carlo sampling

Inference as Optimization

- For a distribution $p(X|\theta)$ associated with a complex graph, computing the **marginal (or conditional) probability** of arbitrary random variable(s) is intractable

- Variational methods
  - formulating probabilistic inference as an optimization problem:

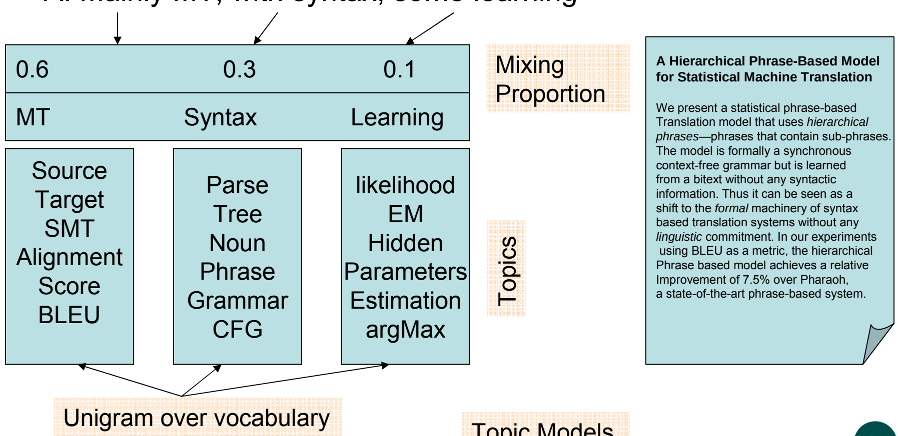
$$e.g.\quad f^{*} = \arg\max_{f \in S} \text{ or } \min \left\{ F(f, P) \right\}$$

$$f: \quad \begin{array}{l} \text{a (tractable) probability distribution} \\ \text{or, solutions to certain probabilistic queries} \end{array}$$

Hierachical Bayesian Models

Topic Models:How to model semantic?

- Q: What is it about?
- A: Mainly MT, with syntax, some learning



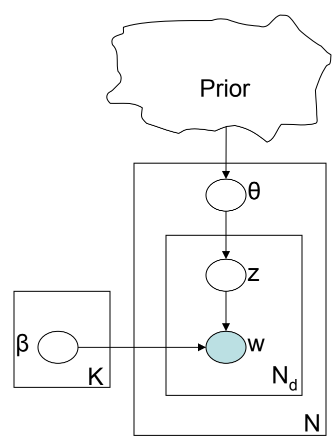
Admixture Models (Topic Models)

Generating a document

– *Draw* θ from the prior

For each word n

- Draw z_n from $multinomial(\theta)$
- Draw $w_n \mid z_n, \{\beta_{1:k}\}$ from $multinomial(\beta_{z_n})$

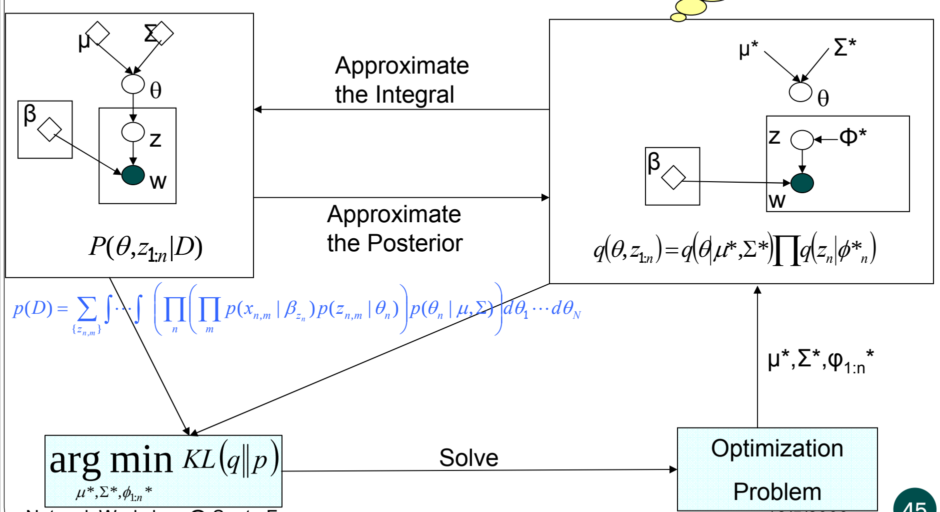


Which prior to use?

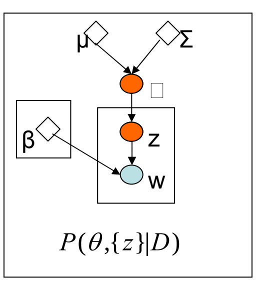
Variational approximation

$f^* = \arg\max_{f \in S} \text{or} \min \{ F(f,P) \}$

f: a (tractable) probability distribution or, solutions to certain queries



Variational Inference With no Tears



Iterate until Convergence

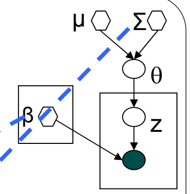
- Pretend you know $E[Z_{1:n}]$
  - $P(\theta | E[z_{1:n}], \mu, \Sigma)$
- Now you know $E[\theta]$
  - $P(z_{1:n} | E[\theta], w_{1:n}, \beta_{1:k})$

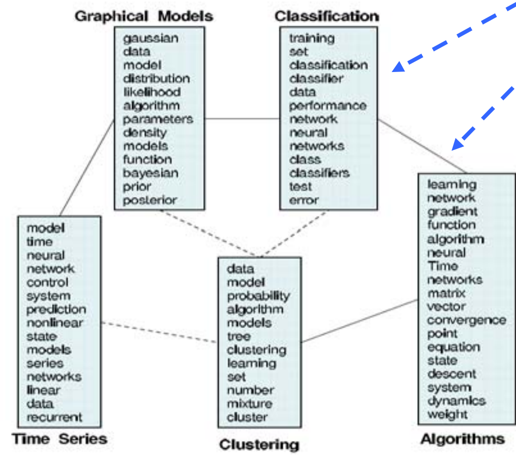
- More Formally: $q^*(X_C) = P\left(X_C \middle| \langle S_Y \rangle_{q_y} : \forall y \in X_{MB} \right)$

Message Passing Scheme (GMF)

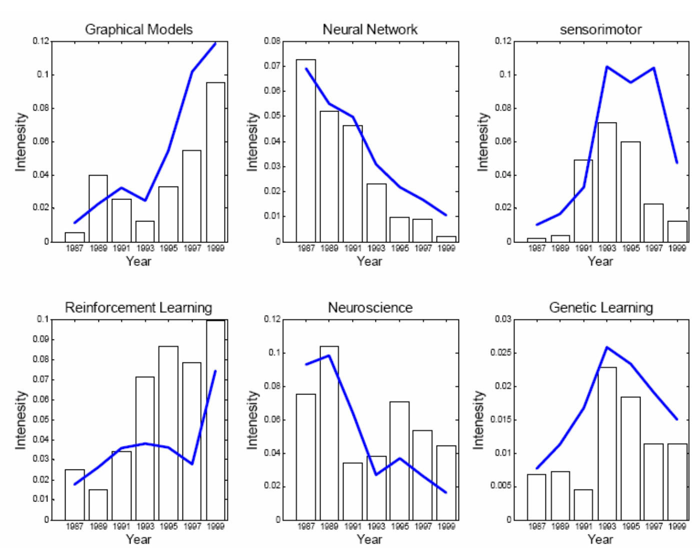
Equivalent to previous method (Xing et. al.2003)

Topics and topic graphs



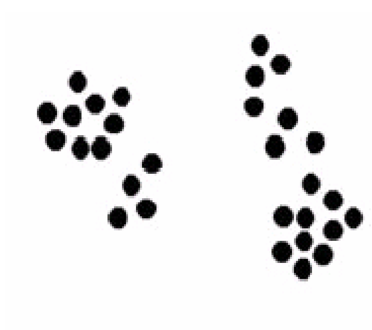


Topic Trends



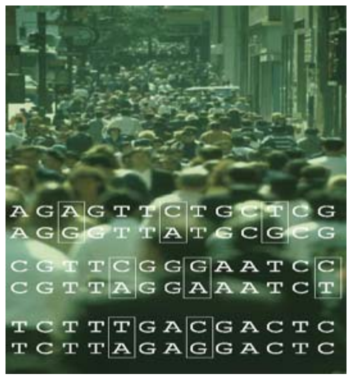
Infinite Models

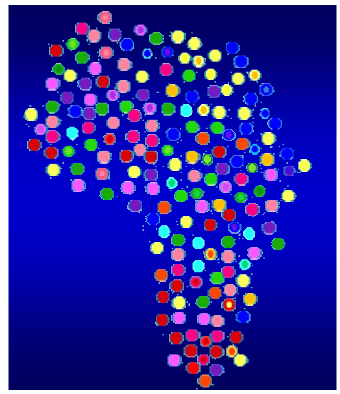
Clustering



- How to label them ?
- How many clusters ???

Genetic Demography





- Are there genetic prototypes among them ?
- What are they ?
- How many ? (how many ancestors do we have ?)

Model Selection vs. Posterior Inference

- Model selection
  - "intelligent" guess: ???
  - cross validation: data-hungry ☹
  - information theoretic:
    - AIC
    - TIC
    - MDL :

    $\left.\right\}$ $\arg\min KL\left(f(\cdot) \,|\, g(\cdot \,|\, \hat{\theta}_{ML}, K)\right)$

    Parsimony, Ockam's Razor
  - Bayes factor: need to compute data likelihood
- Posterior inference:

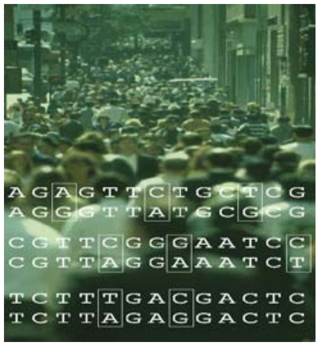
  we want to handle uncertainty of model complexity explicitly

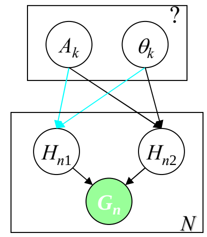
  $$p(M \,|\, D) \propto p(D \,|\, M)\, p(M)$$

  $$M \equiv \{\theta, K\}$$

  - we favor a distribution that constrains M in a "open" space!

Ancestral Inference

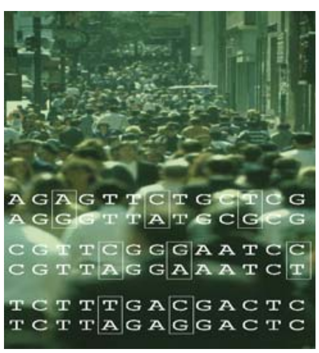


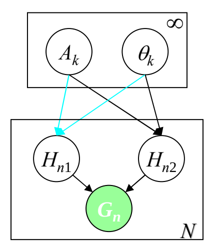


Essentially a clustering problem, but …

- Better recovery of the ancestors leads to better haplotyping results (because of more accurate grouping of common haplotypes)
- True haplotypes are obtainable with high cost, but they can validate model more subjectively (as opposed to examining saliency of clustering)
- Many other biological/scientific utilities

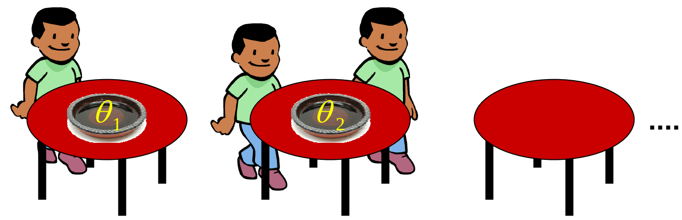
A Infinite (Mixture of) Allele Model





- How?
  - Via a nonparametric hierarchical Bayesian formalism ! (Xing et al 2004,2006)

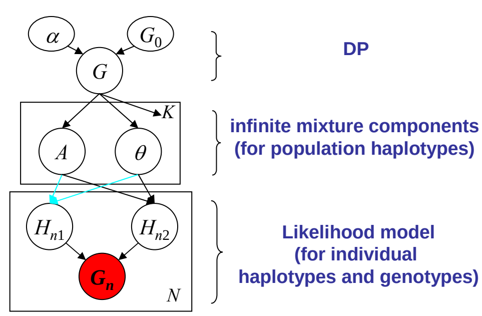
Chinese Restaurant Process



$$P(c_i = k \mid \mathbf{c}_{-i}) = \begin{array}{cccc} 1 & 0 & & 0 \\ \frac{1}{1+\alpha} & \frac{\alpha}{1+\alpha} & & 0 \\ \frac{1}{2+\alpha} & \frac{1}{2+\alpha} & & \frac{\alpha}{2+\alpha} \\ \frac{1}{3+\alpha} & \frac{2}{3+\alpha} & & \frac{\alpha}{3+\alpha} \\ \frac{m_1}{i+\alpha-1} & \frac{m_2}{i+\alpha-1} & \cdots & \frac{\alpha}{i+\alpha-1} \end{array}$$

CRP defines an exchangeable distribution on partitions over an (infinite) sequence of samples, such a distribution is formally known as the Dirichlet Process (DP)

DP-haplotyper



- Inference: Markov Chain Monte Carlo (MCMC)
  - Gibbs sampling
  - Metropolis Hasting

Inference as Stochastic Simulation

Gibbs sampling

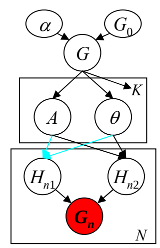
Starting from some initial haplotype reconstruction $H^{(0)}$, pick a first table with an arbitrary $a_1^{(0)}$, and form initial population-hap pool $\mathbf{A}^{(0)} = \{a_1^{(0)}\}$:

i) Choose an individual *i* and one of his/her two haplotypes *t*, uniformly and at random, from all ambiguous individuals;

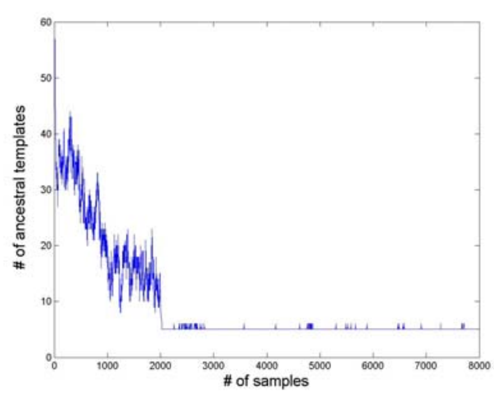
i) Sample $c_{i_t}^{(t+1)}$ from $p(c_{i_t}^{(t+1)} \mid c_{-i_t}^{(t)}, H^{(t)}, \mathbf{A}^{(t)})$, update $c^{(t+1)}$;

iii) Sample $a_k^{(t+1)}$, where $k = c_{i_t}^{(t+1)}$, from $p(a_k^{(t+1)} \mid \forall h_{-i'_{t'}}^{(t)} \text{ s.t. } c_{i'_{t'}}^{(t+1)} = k)$, update $\mathbf{A}^{(t+1)}$;

iii) Sample $h_{i_t}^{(t+1)}$ from $p(h_{i_t}^{(t+1)} \mid c_{i_t}^{(t+1)}, H_{-i_t}^{(t)}, \mathbf{A}^{(t+1)})$, update $H^{(t+1)}$.

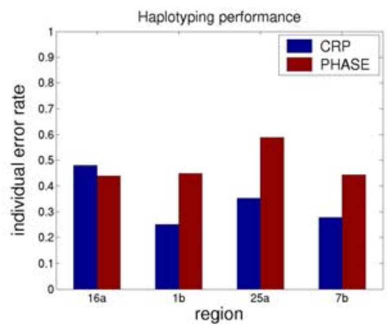


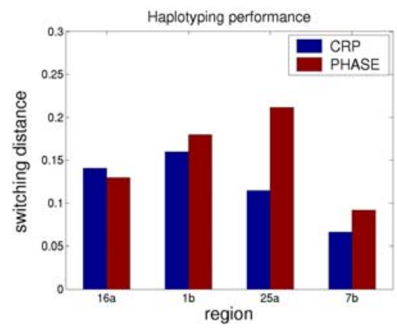
Convergence of Gibbs Sampling



Haplotyping Error

The Gabriel data





Summary

- Fitting Stochastic Models to Empirical Data:
  - All variables are observed in all data (supervised)
  - Some observed, some not, in all data (unsupervised)
  - Some fully and some partially observed data (semi-supervised)
- Estimation principles (loss functions):
  - Least mean squared prediction error
  - Maximal likelihood estimation (MLE)
  - Maximal conditional likelihood
  - Bayesian estimation
  - Maximal "Margin“
  - …
- Learning with hidden variables
  - Exact Inference
  - Variational inference: Inference as optimization
  - Sampling: Inference as stochastic simulation: