Variational Methods
Convex Relations
Monte Carlo Methods

Admin: tutorials this week!
Midterm solutions: tonight
Marked A7: due Friday
A8: out tonight, due Dec. 17th
Coding project: due Dec. 17th
Final Project: marking scheme out this week
report due due Dec. 17 th

## Loopy Belief Propagation

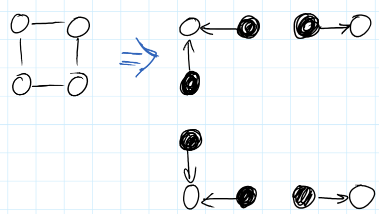
- does not require $q(x)$ to be valid distribution, only "locally consistent"
  (adjacent nodes agree on marginals)
- relation to "turbo codes" in information theory
- algorithm: apply sum-product updates repeatedly (exact for trees)
- convergence issues, but now exist convex/convergent versions.
- LBP is only for Gaussian/Discrete, generalization is "expectation propagation"

## ⊛ Pseudo-likelihood

- For learning parameters of UGM, common and consistent and convex approximation is "pseudo-likelihood"

$$p(x) \propto \prod_c \Psi_c(x) \approx \prod_j p(x_j \mid x_{-j})$$

- training is now easy as in DAGs
  convex, fast



## Convex Relaxations to Decoding

multistate

$$\max_{x \in \{1,2,\dots,s\}^d} \prod_{j=1}^{d} \phi_j(x_j) \prod_{(j,k)\in E} \phi_{jk}(x_j, x_k)$$

$$\Updownarrow$$

binary

$$\max_{\substack{z_j^s \in \{0,1\} \\ z_{jk}^{ss'} \in \{0,1\}}} \sum_{j=1}^{d} \sum_{s=1}^{S} z_j^s \log \phi_j(s) + \sum_{(j,k)\in E} z_{jk}^{ss'} \log \phi_{jk}(s, s') \Rightarrow \text{"Integer linear program"}$$

↳ NP-Hard

$z_j = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{bmatrix}$ ← the one we choose

$Z_{jk} = \begin{bmatrix} 0&0&0&0&0 \\ 0&0&1&0&0 \\ 0&1&0&0&0 \\ 0&0&0&0&0 \end{bmatrix}$

subject to

$\sum_{s=1}^{S} z_j^s = 1$ (can only pick one state)

$\sum_{s=1}^{S} z_{jk}^{ss'} = z_j^s$ (edge labellings agree with node labellings)

$\sum_{s'=1}^{S} z_{jk}^{ss'} = z_j^{s'}$

$\Downarrow$

$z_j^s \in [0, 1]$
$z_{jk}^{ss'} \in [0, 1]$ interval, not binary

↓

"Linear Programing Relaxation"
↳ polynomial time

if get 0, 1 only: equivalent solution to integer (but unlikely to get)

whole values: part of "some optimal solution"

# Monte Carlo Methods

Methods for intractable inference problems

variational: approximate $p(x)$ with simple "simple" $q(x)$

Monte Carlo : - approximate $p(x)$ with S samples $x^s \sim p(x)$

- use $x^1, x^2, \dots x^S$ to approximate

good for:

- $p(x_j)$
- $p(\hat{\theta} \mid D)$
- $\max_x p(x)$

Ex: $\theta \sim$ Dirichlet

| $\theta_1$ | $\theta_2$ | $\theta_3$ |
|---|---|---|
| 0.1 | 0.9 | 0 |
| 0.2 | 0.8 | 0 |
| 0.2 | 0.7 | 0.1 |
| 0.1 | 0.7 | 0.2 |

$E[\theta_1]$ ?

$= (0.1 + 0.2 + 0.2 + 0.1)/4 = 0.15$

But the hard part is selecting good samples!

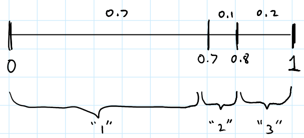
Variational: fast but baised

Monte Carlo: slow but consistent (add more samples - slower but better)

## Sampling from Standard Distribution

(⊛ assume you can generate $u \in U(0,1)$)

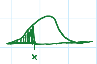
If $x \sim$ Multinuli $(0.7, 0.1, 0.2)$




Generate $u \sim U(0,1)$, if it falls in "2", $x^S =$ "2"

For univariariate continuous, use inverse CDF

Ex: Gaussian

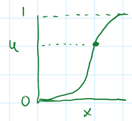



For R.V. X, CDF is $F(x) = p(X \le x)$

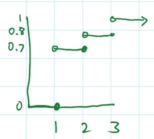
Inverse CDF: $F^{-1}(p) = \{x : F(x) = p\}$ "quantile function"

ex: $F^{-1}(0.5) =$ median

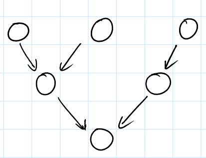
- generate u, put in $F^{-1}$, get value in x




(discrete represented in this form:) Equivalency:




## Sampling From DAG



← sample first

← sample conditioned on parents

← sample last

"Ancestral Sampling"

Special case:

X is a multivariate normal

$X \sim N(\mu, \Sigma)$

- Generate $x \sim N(0, I)$
- Set: $y = \mu + Lx$, where $LL^T = \Sigma$

## Rejection Sampling

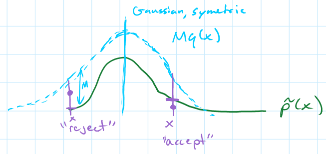
- don't have inverse CDF or too expensive

- sample from proposal $q(x)$, where

$$Mq(x) \geqslant \tilde{p}(x)$$

(M should be small)

Sample $x$ from $q(x)$, $u \in \mathcal{U}(0,1)$
"accept sample if $u \geqslant \dfrac{\tilde{p}(x)}{Mq(x)}$

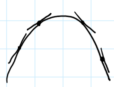



Eg: Sample from posterior: $p(\theta|D) = p(D|\theta)p(\theta)/p(D)$

Target: $\tilde{p}(\theta) = p(D|\theta)\,p(\theta)$
Proposal: $q(\theta) = p(\theta)$

We have: $M = p(D|\theta_{MLE})$ (MLE: satisfies inequality)

Eg 2



$\log(p(x))$ is concave

"Adaptive Rejection Sampling"

## Importance Sampling

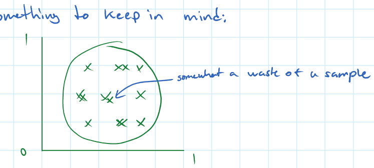
Want to compute $E[h(x)] = \int_x h(x)p(x)\,dx$

- represent as: $\int_x h(x)\dfrac{p(x)}{q(x)}q(x)\,dx$ — $q$ non zero if $p$ non zero

- sample from $q(x)$, weigh each sample by $\dfrac{p(x)}{q(x)}$

$$E[h(x)] \approx \sum_{s=1}^{S} w_s h(x^s)$$

Something to keep in mind:



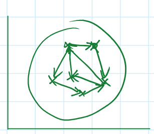

## Sequential Importance Sampling (SMC, Particle filter)

- generate from sequence of proposed distributions $q_t(x)$
- used for tracking, non-Gaussian/non-discrete time series modelling.

## MCMC (Markov Chain - Monte Carlo)

High-dimensional: integration

key idea: construct Markov Chain with $p(x)$ as stationary distribution
Eg we will do a random walk through $\mathcal{X}$, where asymptotically we spend $p(x)$ time at $x$.
- use these <u>dependent</u> samples for Monte Carlo integration



## Gibbs Sampling

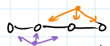
(sampling version of coordinate descent)

- sample $x_1 \sim p(x_1 \mid x_2, x_3)$
- sample $x_2 \sim p(x_2 \mid x_1, x_3)$
- sample $x_3 \sim p(x_3 \mid x_1, x_2)$
- repeat

- Reduces high dimensional sampling to 1D sampling

- May require "burn in" (if you start in area of low probability) (don't want to be biased by start point)

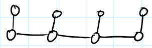
- For graphical models, only depend on neighbours



(and co-parents for DAGs)

- "Block Gibbs Sampling"

- "Rao-Blackwellized"



Similar to F/B

- forward
- backwards, but similar to viterbi decoding

## Metropolis-Hastings Algorithm

- Generalization that allows arbitrary "proposal"

Metropolis algorithm for symmetric proposal:

- at some $x$, propose $x'$
- accept with probability $\min\left(1, \frac{p(x')}{p(x)}\right)$
- sample from arbitrary density $p(x)$
- adapted to a-symetric

Simulated Annealing

$$\min\left(1, \exp\left(\frac{f(x) - f(x')}{T}\right)\right)$$

T large, can go anywhere

T small "only uphill"