Understanding Belief Propagation and its Generalizations

2001

MITSUBISHI ELECTRIC RESEARCH LABORATORIES

Abstract

- Explain belief propagation (BP)
- Developed unified approach
- Compares BP to Bethe approximation of statistical physics
- BP can only converge to a fixed point (which is also the stationary point of the Bethe approximation to free energy)
- Belief propagation is used to solve inference problems (based on local message passing)

Inference Problems

- Many algorithms are just special cases of the BP algorithm
    - Viterbi
    - Forward-backward
    - Iterative decoding algorithms for Gallager codes and turbocodes
    - Pearl's belief propagation algorithm for Bayesian networks
    - Kalman filter
    - Transfer-matrix approach

1 INFERENCE AND GRAPHICAL MODELS

Bayes Networks

Inference problem: Get probability of patient having disease; or any marginal probability in problem.

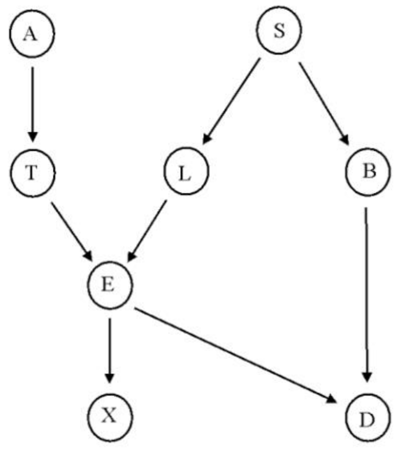
- 'Bayesian Networks' are the most popular type of graphical model
- Used in expert systems of medicine, languages, search, etc.

Example

Want to construct a machine that will automatically give diagnoses for patients. Each patient has some (possibly incomplete) information: symptoms and test result. We infer the probability of a disease. We know statistical dependencies between symptoms, test results, and disease

- Recent trip to Asia (A), increases chances of tuberculosis (T)
- Smoking (S) is risk factor for lung cancer (L) and bronchitis (B)
- Presence of either E tuberculosis or lung cancer can be detected by an X-ray result (X) but the X-ray alone cannot distinguish between them.
- Dyspnoea (D) shortness of breath may be caused by bronchitis (B) or either tuberculosis or lung cancer, E.

PGM would look like



Nodes

- Represent variable that exists in discrete number of possible states
- Filled-in are "observable" nodes, when we have information about patient
- Empty nodes, "hidden" nodes

$$x_i$$

- Variable representing different possible states of node i

Arrow

- Conditional Probability
- $p(x_L|x_S)$
- Conditional probability of lung cancer given they do/do not smoke
- S is "parent" of L because x_L is conditionally dependent on x_s
- D has more than one parent, so $p(x_D|x_E, x_B)$

Joint Probability

- Product of all probabilities of the parent nodes and all conditional probabilities

$$p(x_A, x_T, x_E, x_L, x_S, x_B, x_D, x_X) = p(x_A)p(x_S)p(x_T|x_A)p(x_L|x_S)p(x_B|x_S)p(x_E|x_T, x_L)p(x_D|x_E, x_B)p(x_X|x_E)$$

Directed acyclic graph

- arrows do not loop around in a cycle
- $Par(x_i)$ is the states of the parents of node i and if node i has no parents, it is simply $p(x_i)$
- Directed acyclic graph of N random variables that defines a joint probability function

$$p(x_1, x_2, \dots, x_N) = \prod_{i=1}^{N} p(x_i|Par(x_i))$$

Goal: compute marginal probabilities

- Probability that patient has certain disease
- **"inference" =** computation of marginal probabilities
- Marginal probabilities are defined in terms of sums over all possible states of all other nodes

$$p(x_N) = \sum_{x_1}\sum_{x_2}\dots\sum_{x_{N-1}} p(x_1, x_2, \dots, x_N) = b(x_N)$$

- Marginal Probabilities are the **beliefs**

$$b(x_i)$$

Challenge

- Number of terms in the marginal probability sum **grows exponentially w/ # of nodes in network** -> Back Propagation
- BP computes in time that grows only linearly with number of nodes in system
- BP = "inference engine" acting on statistical data encoded in large Bayesian network

Pairwise Markov Random Fields

Inference problem: inferring state of underlying scene in image

Challenge

- Computer vision

Pairwise MRFs

- Want to infer 'what is really out there' in vision problem, using only a 2D array of numbers

Example:

- Infer the distance of objects in a scene from the viewer
- Given 1000x1000 gray-scale image
- i ranges over million possible pixel positions

$$y_i$$

- *Observed* quantities
- filled in circles

$$x_i$$

- *Latent* variable
- Quantities we want to infer about in underlying scene
- Needs to have some structure, otherwise the problems are ill-posed
- We encode structure by saying the nodes i are arranged in 2D grid, and scene variables x_i are 'compatible' with nearby scene variables x_j

- empty circles

i

- Single pixel position, or small patch of pixels

$\boldsymbol{\phi_i(x_i, y_i)}$

- Assume there is **statistical dependency** between x_i an y_i
- $\phi_i(x_i, y_i)$ is the "**evidence**" for x_i

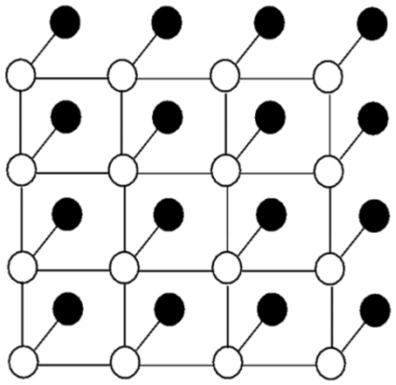
$\boldsymbol{\psi_{ij}(x_i,\ x_j)}$

- *Compatibility function / potential function*
- Only connects nearby positions - why we call it "pairwise"
- Undirected compatibility function instead of directed arrows as in Bayesian network because no node is another node's parent

Take overall joint probability of scene x_i and image y_i to be:

$$p(\{x\},\{y\}) = \frac{1}{Z}\prod_{(ij)} \psi_{ij}(x_i, x_j) \prod_i \phi(x_i, y_i)$$

Iterates over each set of adjacent pixels (pairs - "**pairwise**") and computes the statistical dependence $\boldsymbol{\phi_i(x_i, y_i)}$ between that pixel y and it's underlying inference quantity x
Result is -> **Markov Random Field**



Inference problem: compute belief, $b(x_i)$ for all positions i to infer about underlying scene (similar to that for Bayes Networks)

Potts and Ising Models

Inference problem: computing local 'magnetizations'

- Example of MRF in physics
- Inference problem is the computing local 'magnetizations'

Tanner Graphs and Factor Graphs - Error Correcting Codes (ECC)

Inference Problem: receiver of coded message that has been corrupted by noisy channel is trying to **infer** the message that was initially transmitted

Block parity-check codes

- Try to send k information bits in a block of N bits

- $N > k$ to provide redundancy that can be used to recover from errors induced by the noisy channel

$N = 6, k = 3$ binary code - Tanner Graph Example

Circles = Bits (6 bits)

- Bit that participates in the parity check

Squares

- Parity check

Parity Checks:

- [Square 1] Forces sum of bits 1 and 2 and 4 to be even
- [Square 2] Forces sum of bits 1, 3, and 5 to be even
- [Square 3] Forces sum of bits 2, 3, and 6 to be even

There are 8 codewords that satisfy 3 parity-check constraints: 000000, 001011, 010101, 011110, 100110, 101101, 110011, 111000.

The first 3 bits are information bits. Remaining bits are uniquely determined as the codeword.

Assume the codeword is 010101, but the word '011101' was received. We can assume that 010101 is the real codeword because it is only a single bit-flip away.

Usually, N and k are too large for this - # of codewords is exponentially huge and we cannot look through them all. -> Turn to BP algorithm.

Redefine in Probabilistic Formulation

- Received sequence of N bits y_i and we are tryhing to find the N bits of the true transmitted codeword x_i (latent variable).
- Assume noisy channel is memoryless - so each bit flip is independent of all other bit flips
- Each bit has a conditional probability $p(x_i|y_i)$
- Example: 1st bit received is 0, and bits are flipped with probability f; probability that the first bit was transmitted as 0 is $1 - f$. $p(x_1 = 0|y_1 = 0) = 1 - f$
- Overall probability of codeword is product of each bit's (node's) probability

$$\prod_{i=1}^{N} p(x_i|y_i)$$

*But, we need to make sure the **parity check constraints hold.***

-> Joint probability function that combines conditional probabilities w/ parity check constraints

y_i = observed bit , x_i = actual bit

$$p(\{x\},\{y\}) = \psi_{135}(x_1, x_5, x_3)\psi_{124}(x_1, x_4, x_2,)\psi_{236}(x_3, x_6, x_2)\prod_{i=1}^{6} p(y_i|x_i)$$

Each ψ is a parity check function.

- Value 1 if sum is even
- Value 0 if sum is odd

Note: the parity functions are in terms of the ACTUAL bits because THOSE are what the code was built for. When sending the message, the code was created such that the original numbers fulfill the parity checks (not an error-prone arrival bit, y_i)

Note: Because the observed value is based on the real value (not the other way around) we write the product at the end in terms of $y|x$.

Joint probability distribution, genericized for N bits with $N-k$ parity checks:

$$p(\{x\},\{y\}) = \prod_{j=1}^{N-k} \psi_j(\{x\}_j) \prod_{i=1}^{N} p(y_i|x_i)$$

$\psi_j(\{x\}_j)$ represents the jth parity check ψ_j and all nodes in it as $\{x\}_j$

Marginalizing:

- Minimize # of bits decoded incorrectly
- Compute marginal probability for each bit i and threshold bit to most probable value, for example:

$$p(x_N, y_N) = \sum_{x_1}\sum_{x_2}\cdots\sum_{x_{N-1}}\sum_{y_1}\sum_{y_2}\cdots\sum_{y_{N-1}} p(\{x\},\{y\}) = \sum_{x_1}\sum_{x_2}\cdots\sum_{x_{N-1}}\sum_{y_1}\sum_{y_2}\cdots\sum_{y_{N-1}} \prod_{j=1}^{N-k}\psi_j(\{x\}_j)\prod_{i=1}^{N} p(y_i|x_i)$$

- but this may not yield a valid codeword, since each bit is minimized independently
- Above computation will take exponentially huge time
- --> Resort to BP algorithm

Note: BP is not exact, but effective in practice. Achieves 'near-Shannon' limit performance.

Factor Graph

- Generalization of Tanner graph
- Square represents any function of its variables (its variables are the nodes attached to it)
  - Can have any number of variables, even just 1
- Joint probability function of factor graph of N variables with M functions:

$$p(\{x\}) = \frac{1}{Z}\prod_{a=1}^{M}\psi(\{x_a\})$$

The observed nodes y_i are absorbed into the functions ψ and not written out.

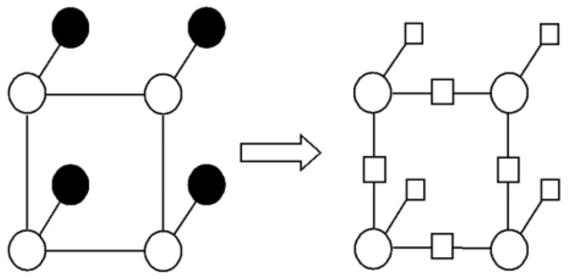
Converting Graphical Models

Can convert arbitrary pairwise MRF's or Bayesian networks into equivalent factor graphs.

MRF -> Factor Graph

- Observable node is replaced by factor function of single variable F
- Factor graph function $\psi(\{x_a\})$ is equivalent to $\psi_{ij}(x_i, x_j)$ when they link latent nodes, or single node function $\phi(x_i, y_i)$ when attatched to single latent node.

Below shows a MRF -> Factor Graph

- 

Bayesian Networks, pairwise Markov Random Fields and factor graphs are all **mathematically equivalent**

- Bayesian networks & pairwise MRFs can be converted into factor graphs (see above)
- reverse is also true
- for future work, choose to use pairwise MRFs

2 Standard Belief Propagation

"observed" nodes y_i

joint probability distribution of unknown variable x_i:

$$p(\{x\}) = \frac{1}{Z} \prod_{(i,j)} \phi_{ij}(x_i, x_j) \prod_i \phi_i(x_i)$$

message from hidden node i to hidden node j about what state j should be in: $m_{ij}(x_j)$

- $m_{ij}(x_j)$ is same dimensionality as x_j
- each component is proportional to how likely node i thinks node j will be in corresponding state

belief at node i is proportional to product of local evidence at that node & all message

$$b_i(x_i) = k\phi_i(x_i) \prod_{j\in N(i)} m_{ji}(x_i)$$

normalization constant: k

- beliefs must sum to 1

nodes neighboring i: $N(i)$

Messages are determined self-consistently by message update rules:

$$m_{ij}(x_j) \leftarrow \sum_{x_i} \phi_i(x_i)\psi_{ij}(x_i, x_j) \prod_{k\in N(i)\setminus j} m_{ki}(x_i)$$

Which states

- Message from $x_i \to x_j$, $m_{ij}(x_j)$ is the sum of all previous current states multiplied by the product of all messages from neighbors of i to i except for j, because that is the node few are sending a message to now; that sum of messages is displayed below

Above is an example pairwise-MRF (Figure 12). The belief at node 1 is

$$b_1(x_1) = k\phi_1(x_1)m_{21}(x_1)$$

where k is probably a constant, phi_1 is the evidence (the relationship between the observed data x, filled-in node, and latent data y, empty node). Evidence is necessary to dictate what we think y is based on x. m_21 is the message from 2 to 1.

If we write out what message 2 to 1 is, we find it composes of the belief at node 3 and belief at node 4:

$$b_1(x_1) = k\phi_1(x_1) \sum_{x_2} \psi_{12}(x_1, x_2)\phi_2(x_2) \sum_{x_3} \phi_3(x_3)\psi_{23}(x_2, x_3) \sum_{x_4} \phi_4(x_4)\psi_{24}(x_2, x_4) \quad (17)$$

And simplifying it all into 1 sum we find

$$b_1(x_1) = k \sum_{x_2,x_3,x_4} p(\{x\}) = p_1(x_1)$$

which shows that the belief at node 1 $b_1(x_1)$ is the marginal probability at node 1 (meaning it's the belief at node 1 with all the states of the other nodes marginalized out by summing over them).

Order to compute?

- usually start at edges and compute message only when we have all available messages necessary
- for above example, we should've started at 3 and 4 to compute belief for 1.

Why is message passing so awesome?

- look at example above, if we start at nodes 3 and 4, we only need to compute the message once per node. And, each message only depends on the node before. The time complexity is proportional to the **number of links (edges) in the graph** this is WAYYYY LESS than the time it takes to compute marginal probabilities naively (aka, doing the equation above for belief 1 for every single node separately).

Why do loops mess everything up?

- because each belief depends on the chain of messages before it being computed. If there was a loop back to the current node then it would get all messed up because that chain would not terminate. That's why we consider only **loop-free pairwise MRFs**

2-node marginal probabilities

$$p_{ij}(x_i, x_j)$$

- for neighbors i and j
- obtained by marginalizing joint probability function over every node except for these 2

$$p_{ij}(x_i, x_j) \equiv \sum_{z:z_{ij}=(x_i,x_j)} p(\{z\}).$$

equation 20:

$$b_{ij}(x_i, x_j) = k\psi_{ij}(x_i, x_j)\phi_i(x_i)\phi_j(x_j) \prod_{k \in N(i)\backslash j} m_{ki}(x_i) \prod_{l \in N(i)\backslash i} m_{lj}(x_j)$$

except there is a typo, second product should be all neighbors of j except for i

If we marginalize equation 20 over j we will get the belief of i, as shown above in equation 13:

$$b_i(x_i) = k\phi_i(x_i) \prod_{j \in N(i)} m_{ji}(x_i)$$

thus:

$$b_i(x_i) = \sum_{x_j} b_{ij}(x_i, x_j)$$

And what do we do about **loops?**

- run it as normal and wait until it converges
- but it may circulate indefinitely and it may never converge

3 FREE ENERGIES

Fixed points of belief propagation = the stationary points of Bethe free, for pairwise MRF

1. Derive Bethe Free Energy (because Gibbs is intractable)
    a. Define KL-Distance with approximating distribution in terms of energy via Boltzmann's
    b. Derive Gibbs Free Energy as equal to minimum KL-distance
    c. Derive Mean Field Gibbs Free Energy to make #2 more tractable
    d. Derive Bethe Free Energy, which consists of all tree-like distributions
2. Show stationary points (minima) of Bethe Free Energy are equal to fixed points of Belief Propagation

But first… Distribution Bounding

We are trying to find the distribution q within the space of all distributions Q, for which $q = p$ almost everywhere (then it is a minimum of the KL). We limit q to space of distributions in order to make the problem tractable. $p(x, D)$ is the joint distribution of D data and x latent variable, and $q(x)$ is the approximating distribution (Expectation is over X, but follow functional form of KL) : $KL\big(q(x) \parallel p(x, D)\big) = \mathbb{E}_x\left[\log\left(\frac{q(x)}{p(x,D)}\right)\right]$

$$\log(Z) = \min_{q \in Q^{Gibbs}} KL(q(x)||p(x, D)) \leq \min_{q \in Q^{Bethe}} KL(q(x)||p(x, D)) \leq \min_{q \in Q^{Mean\ Field}} KL(q(x)||p(x, D))$$

	Constraint Set	*Tractability based on constraint set*
$\log(Z)$		
$\min_{q \in Q^{Gibbs}} KL(q(x)\|\|p(x, D))$	*All distributions*	*intractable*
$\min_{q \in Q^{Mean\ Field}} KL(q(x)\|\|p(x, D))$		
$\min_{q \in Q^{Bethe}} KL(q(x)\|\|p(x, D))$	Distributions that obey tree-like factorization	Tractable by optimization (can find local minimia)

1 Deriving Bethe Free Energy

1. Define KL- Distance w/ the approximating distribution in terms of energy (via Boltzmann's)
    a. KL-Distance, ***Equation 22*** ($KL_D = D$)

- $$D\left(b(\{x\})||p(\{x\}\right) = \sum_{\{x\}} b(\{x\}) \ln \frac{b(\{x\})}{p(\{x\})}$$

b. Replace $\mathrm{p(x)}$ in Equation 22 with Boltzmann's Distribution $\mathrm{p(\{x\}) = \frac{1}{Z} e^{-E(\{x\})}}$, allow $\mathrm{T = 1}$. But first, rewrite Eq 22 in terms of expectation $\left(\mathrm{KL}(\mathrm{q(x)} \parallel \mathrm{p(x)}) = \mathbb{E}_{\mathrm{q(x)}}\left[\log\left(\frac{\mathrm{q(x)}}{\mathrm{p(x)}}\right)\right] = \sum_{\mathrm{x}\in\mathcal{X}} \mathrm{q(x)} \log\left(\frac{\mathrm{q(x)}}{\mathrm{p(x)}}\right)\right)$ to get ***Equation 23***:

- $\boldsymbol{D(b(\{x\}) \parallel p(\{x\}) = \sum_{\{x\}} b(\{x\})\, ln\left(\frac{b(\{x\})}{p(\{x\})}\right) = \mathbb{E}_{b(x)}\left[log\left(\frac{b(x)}{p(x)}\right)\right]}$

$$\boldsymbol{= \mathbb{E}_{b(x)}\left[log\left(\frac{b(x)}{\frac{1}{Z}e^{-E(\{x\})}}\right)\right]}$$

We can turn the division of the log into a subtraction, and distribute the Expectation linearly:

$$= \mathbb{E}_{b(x)}\left[\log(b(x)) - \log\left(\frac{1}{Z}e^{-E(\{x\})}\right)\right]$$

$$= \mathbb{E}_{b(x)}[\log(b(x))] - \mathbb{E}_{b(x)}\left[\log\left(\frac{1}{Z}e^{-E(\{x\})}\right)\right]$$

$$= \mathbb{E}_{b(x)}[\log(b(x))] - \mathbb{E}_{b(x)}[\log(e^{-E(\{x\})}) - \log(\mathrm{Z})]$$

$$= \mathbb{E}_{b(x)}[\log(b(x))] - \mathbb{E}_{b(x)}[\log(e^{-E(\{x\})})] - \mathbb{E}_{b(x)}[-\ln(Z)]$$

$$= \mathbb{E}_{b(x)}[\log(b(x))] + \mathbb{E}_{b(x)}[E(\{x\})] + \mathbb{E}_{b(x)}[\ln(Z)]$$

By the definition of Expectation: $\mathbb{E}_{P(x)}[f(x)] = \sum_{i=1}^{K} p_i\, f(a_i)$, *get* ***Equation 23****:*

$$= \sum_{\{x\}} b(\{x\}) \log(b(\{x\})) + \sum_{\{x\}} b(\{x\}) E(\{x\}) + \ln(Z)$$

ii. Note that $\sum_{\{x\}} b(\{x\}) \log(b(\{x\}))$ is the ***entropy*** of $b(\{x\})$

iii. optimal $\mathrm{b(x)}$ approximating distribution is given by $-\ln(Z)$ (next eq.)

iv.

Boltzmann's Distribution* / *Gibbs distribution

- comes from assuming ***Boltzmann's Law***
  - Boltzmann's law describes power radiated from a black body in terms of its temperature: total energy radiated per unit surface area of a black body across all wavelengths per unit time is directly proportional to the fourth power of the black body's temperature
  - "The initial state in most cases is bound to be highly improbable and from it the system will always rapidly approach a more probable state until it finally reaches the most probable state, i.e. that of the heat equilibrium. If we apply this to the second basic theorem we will be able to identify that quantity which is usually called ***entropy*** with the probability of the particular state." (Boltzmann, 1877) (1) from here
- distribution is a probability distribution that a system will be in a certain state as a function of the state's energy and temperature (T) of the system
- $p_i \propto \frac{1}{Z} e^{-\frac{E(x)}{T}}$
  - Z is the normalization

        - T is temperature
        - E(x) is energy of state
    - Also known as Gibbs distribution
    - Physicists specialize on this class of distributions
    - Any distribution can be expressed in this form by raising it to an exponent and normalizing using Z
    - $probability \propto e^{-Energy} \rightarrow energy = -\log(probability)$
        - put into terms of energy because systems decrease in entropy

2. Derive Gibbs Free Energy
    a. Solve Eq 23 for $\ln(Z)$ to get ***Equation 24,*** Gibbs Free Energy:
    b. $G(b\{x\}) = \sum_{\{x\}} b(\{x\})E(\{x\}) + \sum_{\{x\}} b(\{x\}) \log\big(b(\{x\})\big)$

$$G(b\{x\}) = \mathrm{U}(\mathrm{b}\{\mathrm{x}\}) - \mathrm{S}(\mathrm{b}\{\mathrm{x}\})$$

$$G(b\{x\}) = average\ energy\ - entropy$$

        i. where $average\ energy = U(b\{x\}) = \sum_{\{x\}} b(\{x\})E(\{x\})$
            1. ***average energy*:** expectation (average) of Boltzmann's Energy
        ii. where negative of the $entropy = S(b\{x\}) = \sum_{\{x\}} b(\{x\}) \log\big(b(\{x\})\big)$
        iii. balance, because average energy wants to keep q small, and entropy wants to spread out q (because we minimize the entire eq, therefore maximizing the negative entropy)
    c. KL distance is 0 when Equation 24 achieves its **minimal** value of $-\ln(Z)$.
    d. $G(b\{x\})$ is a functional of b, we want to **find the b that minimizes the Gibbs Free Energy (**same as maximizing likelihood; remember, high energy means unstable, we want stable so we want lower energy)
    e. This minimization is a balance between U and S. Negative entropy, -S, is considered, thus entropy wants to be as large as possible. However, U wants to keep q small.

    Free Energy Background
    Process will only happen spontaneously (without added energy) if it *increases the entropy of the universe* - 2nd Law of Thermodynamics ("total entropy of an isolated system can never decrease over time")
        - need a metric to capture the effect of a **reaction** on the **entropy** of the **universe** --> ***Gibbs Free Energy***
        - provides measure of how much usable ("useful") energy is released (or consumed) when the reaction takes place ; the amount of "Free" or "useful" energy available to do work (associated with chemical reaction)

    In biology, Gibbs Free Energy is defined as *Gibbs Free Energy Change Equation*:
            - $\Delta G = G_{final} - G_{initial}$
            - tells us maximum usable energy released (or absorbed) in going from initial to final state
        - if ΔG (Gibbs Free Energy) is negative, it means there is a negative energy release which means they do increase entropy and can happen sponatenously
        - else, they are non-spontaneous and need input energy

3. Derive Mean Field Gibbs Free Energy
    a. Although we know p is not independent, we assume that the approximating distribution b is, and so we define dependency structure as **independent**
    b. Assume b is independent and we are working with a pairwise MRF:
        i. $b(\{x\}) = \prod_i b_i(x_i)$ and $\sum_i b_i(x_i) = 1$

c. Energy of Markov Random Field, ***Eq 26***

   i. $$E(\{x\}) = -\sum_{(ij)} \ln \psi_{ij}(x_i, x_j) - \sum_i \ln \phi_i(x_i)$$

a. Eq 27 uses the definition of $U(b\{x\})$ from Eq 24 to derive mean-field average energy, ***Eq 27***

   i. $$U_{MF}(\{b_i\}) = -\sum_{(ij)} \sum_{x_i, x_j} b_i(x_i) b_j(x_j) \ln \psi_{ij}(x_i, x_j) - \sum_i \sum_{x_i} b_i(x_i) \ln \phi_i(x_i)$$

a. Only a function of one-node beliefs, therefore we derive Bethe, to get a function of 2-node beliefs.

Background Mean Field Theory (MFT)

- also known as **variational methods**
- takes optimization problems defined over discrete variables & converts them to continuous -> allows to compute gradients of energy and use **optimization** techniques
- can use **deterministic annealing** methods : define 1-parameter family of optimization problems indexed by temperature parameter T
- approximate distribution $P(x|z)$ using a **simpler distribution** $B^*(x|z)$
  - easy to estimate MAP of $P(\quad)$ from $B^*(\quad)$
  - need to specify
    1. *class of approximating distributions* for the belief, $B^*(\quad)$ --> **Factorizable**
    2. *measure of similarity between B and P* --> **Kullback-Leibler Divergence**
    3. algorithm for finding $B^*(\quad)$ that minimizes similarity distance --> **minimize energy**

Mean Field Free Energies

- Class of approximating distributions:
  - factorizable so $B(x) = \prod_{i \in V} b_i(x_i)$
  - $\{b_i\{x_i\}\}$ are pseudo-marginals, so each belief is greater than 0 and the sum of all beliefs over a certain x (node) is 1
  - MAP estimate of x is the that which maximizes the belief, $\bar{x}_i = \arg\max_{x_i} b^*(x_i)$

4. Derive Bethe Free Energy from Gibbs
   a. Derive Gibbs free energy that is function of 1-node and 2-node beliefs and can work on any tree-like distribution -> Bethe
   b. if graph is ***singly connected*** (connected graph where there is only 1 path between each pair of nodes, same as ***tree***) we use the joint probability distribution given in Eq 30.
   c. Goal: optimize over distributions $q \in Q^{BETHE}$ consistent with **tree**-structured Markov Random Field
      1. Any tree-structured distribution $q(\theta)$ can be written as the factorization below, which is product over unary nodes times a ratio of pairwise and unary nodes as ***reparameterization***

1. $q(\theta) = \prod_{S \in \mathcal{V}} q_s(\theta_s) \prod_{(s,t)\in\mathcal{E}} \frac{q_{st}(\theta_s,\theta_t)}{q_s(\theta_s)q_t(\theta_t)}$

2. ***Equation 30,*** similar except denominator takes each belief to a power, q_i which is the number of nodes neighboring node i; ***reparameterization*** w/ excess variables dropped:
   1. $$b(\{x\}) = \frac{\prod_{(ij)} b_{ij}(x_i, x_j)}{\prod_i b_i(x_i)^{q_i - 1}}$$
   2. ???? How is this equal to the other reparameterization. The denominator clearly has power!?
3. Example: Convert Bayes net factorization & write it using #1 factorization:

$$q(\theta) = \big(q_1(\theta_1)q_2(\theta_2)q_3(\theta_3)q_4(\theta_4)\big)\left(\frac{q_{12}(\theta_1,\theta_2)}{q_1(\theta_1)q_2(\theta_2)}\right)\left(\frac{q_{13}(\theta_1,\theta_3)}{q_1(\theta_1)q_3(\theta_3)}\right)\left(\frac{q_{34}(\theta_3,\theta_4)}{q_3(\theta_3)q_4(\theta_4)}\right)$$

Simplify:

$$q(\theta) = \big(q_1(\theta_1)\cancel{q_2(\theta_2)}\cancel{q_3(\theta_3)}\cancel{q_4(\theta_4)}\big)\left(\frac{q_{12}(\theta_1,\theta_2)}{q_1(\theta_1)\cancel{q_2(\theta_2)}}\right)\left(\frac{q_{13}(\theta_1,\theta_3)}{q_1(\theta_1)\cancel{q_3(\theta_3)}}\right)\left(\frac{q_{34}(\theta_3,\theta_4)}{q_3(\theta_3)\cancel{q_4(\theta_4)}}\right)$$

$$q(\theta) = \big(q_1(\theta_1)\big)\left(\frac{q_{12}(\theta_1,\theta_2)}{q_1(\theta_1)}\right)\left(\frac{q_{13}(\theta_1,\theta_3)}{q_1(\theta_1)}\right)\left(\frac{q_{34}(\theta_3,\theta_4)}{q_3(\theta_3)}\right)$$

Note that conditional probability is given by: $P(B|A) = \frac{P(A,B)}{P(A)}$

$$q(\theta) = \big(q_1(\theta_1)\big)\big(q_2(\theta_2)|q_1(\theta_1)\big)\big(q_2(\theta_3)|q_1(\theta_1)\big)\big(q_2(\theta_4)|q_1(\theta_3)\big)$$

$$\begin{tikzcd} & \theta_1 \arrow[dl] \arrow[dr] & \\ \theta_2 & & \theta_3 \arrow[d] \\ & & \theta_4 \end{tikzcd}$$

4. Distributions of the form R necessarily satisfy Local Marginal Consistency:
   1. $q_s(\theta_s) = \int q_{st}(\theta_s, \theta_t)d\theta_t \ \forall t \in \Gamma(s)$
   2. Where $\Gamma(s)$ is all neighbors of S
5. For SOME REASON? Nodes on their own are not marginals. $q_s(x_s) \neq \int q(x)dx_{\setminus s}$
6. To fix this we **locally constrain** the problem so each node can be **locally marginalized** by its neighbors:
   1. $q_s(\theta_s) = \int q_{st}(\theta_s, \theta_t)d\theta_t \ \forall t \in \Gamma(s)$

d. If q is not a tree, then q is an approximation, otherwise q is the exact approximation

e.

f. From Gibbs Free Energy, ***Equation 24,*** Gibbs Free Energy, we know ***average energy***:

1. $U(b\{x\}) = \sum_{\{x\}} b(\{x\})E(\{x\})$

g. From Gibbs Free Energy, ***Equation 24,*** Gibbs Free Energy, we know entropy:

$$S(b\{x\}) = -\sum_{\{x\}} b(\{x\})\log\big(b(\{x\})\big)$$

1. Deriving S_{Bethe} from Equation 28, the mean-field entropy, combined with the correct joint probability distribution, we get ***Equation 31***

$$S_{Bethe} = -\sum_{(ij)}\sum_{x_i,x_j} b_{ij}(x_i,x_j)\ln b_{ij}(x_i,x_j) + \sum_i (q_i - 1)\sum_{x_i} b_i(x_i)\ln b_i(x_i)$$

2.

3. If b is not a tree, then b is an approximation, otherwise q is the exact approximation

h. Average Energy, U,
   1. of pairwise MRFs with 2-node beliefs that obey conditions
      1. $\sum_{x_i} b_i(x_i) = \sum_{x_i, x_j} b_{ij}(x_i, x_j) = 1$,
   2. is similar to Equation 27, except the belief is of both i, j together, and not each separately multiplied together, ***average energy Eq 29***:
   3. $U = -\sum_{(ij)} b_{ij}(x_i, x_j) \ln\left(\psi_{ij}(x_i, x_j)\right) - \sum_i b_i(x_i) \ln(\phi_i(x_i))$
   4. Replace $-\ln(\phi_i(x_i)) = E_i(x_i)$ where it is the local energy
   5. Replace $-\ln(\phi_{ij}(x_i, x_j) - \ln(\phi_i(x_i)) - \ln\left(\phi_j(x_j)\right) = E_{ij}(x_i, x_j)$
   6. To get ***Equation 32,*** average energy:
   7. $$U = \sum_{(ij)} \sum_{x_i, x_j} b_{ij}(x_i, x_j) E_{ij}(x_i, x_j) + \sum_i (q_i - 1) \sum_{x_i} b_i(x_i) E_i(x_i)$$

i. Bethe Free Energy, altogether:
   1. $G = U - S$
   2. ***Equation 33***
   3. $$\begin{aligned} G_{Bethe}(b_i(x_i), b_{ij}(x_i, x_j)) &= \sum_{(ij)} \sum_{x_i, x_j} b_{ij}(x_i, x_j) \left(E_{ij}(x_i, x_j) + \ln b_{ij}(x_i, x_j)\right) \\ &- \sum_i (q_i - 1) \sum_{x_i} b_i(x_i) \left(E_i(x_i) + \ln b_i(x_i)\right) \end{aligned}$$
   4. Equal to the Gibbs Free Energy for pairwise MRFs when graph has no loops
   5. Belief propagation beliefs are the global minima of Bethe free energy when the graph has no loops -> set of beliefs gives a BP fixed point in any graph only if they local stationary points of Bethe Free Energy

Bethe Background - Bethe Method

- pronounced [bay-duh]
- dates back to 1935 in Bethe's famous approximation method for magnets
- 1951: cluster variation method
- Kikuchi constructed more accurate free energy approximation from Bethe's work
- Yedidia creates generalized belief propagation (GBP) from Kikuchi approximations
- History:
   - Bethe(energy) -> Kikuchi (more accurate free energy) -> Kedidia (generalized belief prop, GBP)

3.3 EQUIVALENCE OF BP to BETHE APPROXIMATION

Bethe Free Energy = Gibbs free energy for pairwise MRFs
Bethe Free Energy is minimal for correct marginals
BP gives correct marginals
∴ BP beliefs are global minimia of Bethe Free Energy

Proof
Set of beliefs gives a BP fixed point in any graph if and only if they are local stationary points of the Bethe Free Energy (belief propagation can only converge to a stationary point of an approximate free energy (Bethe Free Energy in statistical physics)

$$\begin{aligned} G_{Bethe}(b_i(x_i), b_{ij}(x_i,x_j)) &= \sum_{(ij)} \sum_{x_i,x_j} b_{ij}(x_i,x_j)\left(E_{ij}(x_i,x_j) + \ln b_{ij}(x_i,x_j)\right) \\ &- \sum_i (q_i - 1) \sum_{x_i} b_i(x_i)\left(E_i(x_i) + \ln b_i(x_i)\right) \end{aligned}$$

1) Add **marginalization constraint** to beliefs

- need to enforce **marginalization constraint** in G_{Bethe} : a belief over a node is equal to the pair-wise belief of this node and another node marginalized over the other node:
  i. $b_i(x_i) = \sum_j b_{ij}(x_i, x_j)$
- Use Lagrange Multipliers. Brief recap of how they work:

  Method of Lagrange Multipliers:
  - Convert constraint surface to implicit surface function where
    - $g(x) = 0$
    - For example, to constrain to a unit circle, $x^2 + y^2 = 1 \rightarrow \; x^2 + y^2 - 1 = 0$
  - Can write set of points likes this
    - Direction in which points grow is equal to $(-\lambda)$ scaled version of constraint surface $(\nabla_x g)$
    - $\nabla_x f = -\lambda \nabla_x g$
    - $\nabla \; (f + \lambda g) = 0$
  - So for every constraint we add to the problem, we add 1 new variable to the problem .
    - Create new function $L(x, \lambda) = f(x) + \lambda g(x)$
    - Increase dimensionality of problem
    - Unconstrained extrema of L correspond to extrema of f constrained to g

- γ_{ij}, γ_i normalize b_{ij}, b_i
  . $\lambda_{ij}(x_j)$ multiplier that enforces marginalization $b_i(x_i) = \sum_j b_{ij}(x_i, x_j)$
- Turn G_{bethe} into L by adding $\lambda\left(\sum_j b_{ij}(x_i, x_j) - b_i(x_i)\right)$
- Take derivative with respect to $b_{ij}(x_i, x_j)$ and set equal to 0 ($\frac{\partial L}{\partial b_{ij}(x_i, x_j)} = 0$) to get

  Equation 34

f. $$\ln b_{ij}(x_i, x_j) = -E_{ij}(x_i, x_j) + \lambda_{ij}(x_j) + \lambda_{ji}(x_i) + \gamma_{ij} - 1$$

g. do same with $b_i(x_i)$, $\frac{\partial L}{\partial b_i(x_i)} = 0$, ***Equation 35***:

h. $$(q_i - 1)(\ln b_i(x_i) + 1) = (1 - q_i)E_i(x_i) + \sum_{j \in N(i)} \lambda_{ji}(x_i) + \gamma_i$$

i. Differentiating the Lagrangian with respect to the Lagrange multipliers gives the marginalized constraints ($\frac{\partial L}{\partial \lambda_{ij}}$)

j. Suppose we have set of messages and beliefs that are a fixed-point of belief propagation:

k. $\lambda_{ij}(x_j) = \ln \prod_{k \in N(j) \backslash i} m_{kj}(x_j)$

2) Proof

Going from ***Eq 34*** to 20 (Belief equation for 2-node beliefs)

- Use ***Equation 36*** $\lambda_{ij}(x_j) = \ln \prod_{k \in N(j) \backslash i} m_{kj}(x_j)$ in ***Equation 34***

  a. $\ln\left(b_{ij}(x_i, x_j)\right) = -E_{ij}(x_i, x_j) + \lambda_{ij}(x_j) + \lambda_{ji}(x_i) + \gamma_{ij} - 1$

  b. $\ln\left(b_{ij}(x_i, x_j)\right) = -E_{ij}(x_i, x_j) + \ln \prod_{k \in N(j) \backslash i} m_{kj}(x_j) + \ln \prod_{k \in N(i) \backslash j} m_{ki}(x_i) + \gamma_{ij} - 1$

  c. Because p is a pairwise MRF we can replace $-E_{ij}(x_i, x_j) = \log\left(\psi_{i,j}(x_i, x_j)\right) + \log(\phi_i(x_i)) + \log\left(\phi_j(x_j)\right)$

  d. $\ln\left(b_{ij}(x_i, x_j)\right) = \left(\log\left(\psi_{i,j}(x_i, x_j)\right) + \log(\phi_i(x_i)) + \log\left(\phi_j(x_j)\right)\right) + \ln \prod_{k \in N(j) \backslash i} m_{kj}(x_j) + \ln \prod_{k \in N(i) \backslash j} m_{ki}(x_i) + \gamma_{ij} - 1$

  e. Because a sum of logs is a log of their product:

  f. $\log(\psi_{i,j}(x_i, x_j)\phi_i(x_i)\phi_j(x_j) \prod_{k \in N(j) \backslash i} m_{kj}(x_j) \prod_{k \in N(i) \backslash j} m_{ki}(x_i)) + \gamma_{ij} - 1$

  g. Somehow $\gamma_{ij} - 1$ gets turned into the k within the log, and we get:

  h. $\log\left(b_{ij}(x_i, x_j)\right) = \log(k\psi_{i,j}(x_i, x_j)\phi_i(x_i)\phi_j(x_j) \prod_{k \in N(j) \backslash i} m_{kj}(x_j) \prod_{k \in N(i) \backslash j} m_{ki}(x_i))$

  i. Removing logs,

  j. We end up with ***Equation 20***

  k. $$b_{ij}(x_i, x_j) = k\psi_{ij}(x_i, x_j)\phi_i(x_i)\phi_j(x_j) \prod_{k \in N(i) \backslash j} m_{ki}(x_i) \prod_{l \in N(i) \backslash i} m_{lj}(x_j)$$

     1. NOTE: I think **TYPO** is above. Second prod should be over neighbors of j not including i.

3) Proof

Going from ***Eq 35*** to 13 (Belief equation for 1-node beliefs)

Not enough time to prove.

Equation 36, Set of messages and beliefs that are fixed-point of BP -$\lambda_{ij}(x_j)$

$$\lambda_{ij}(x_j) = \ln \prod_{k \in N(j) \setminus i} m_{kj}(x_j)$$

Above and the beliefs satisfy equations 34 and 35 (the derivative w.r.t to the belief on xi and xj, and just xi, eq f and h above) using

Equation 13: Belief equation for 1-node beliefs

$$b_i(x_i) = k\phi_i(x_i) \prod_{j \in N(i)} m_{ji}(x_i)$$

Equation 20:

Belief equation for 2-node beliefs

$$b_{ij}(x_i, x_j) = k\psi_{ij}(x_i, x_j)\phi_i(x_i)\phi_j(x_j) \prod_{k \in N(i) \setminus j} m_{ki}(x_i) \prod_{l \in N(i) \setminus i} m_{lj}(x_j)$$

Extended Discussion

Standard BP converges to stationary points of constrained Bethe Free Energy but BP does not minimize the Bethe Free Energy - relationship is one-way!

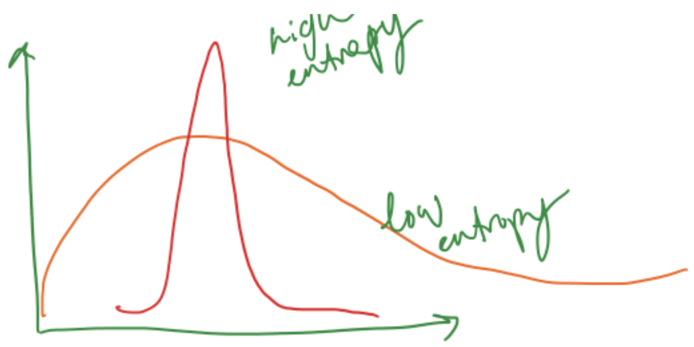
- BP does not decrease Bethe Free Energy at each iteration

Some have made algorithms that minimize free energy on a feasible set of beliefs - these are slower than BP but always converge

Necessary Background – KL Divergence

Entropy

- origins in information theory (goal: quantify how much information is in data)
  - $H = -\sum_{i=1}^{N} p(x_i) \cdot \log p(x_i)$
  - minimum number of bits it would take to encode information
- measure of spread or uncertainty in a distribution



Expectation

- long-run average value of repetitions of the same experiment it represents
  - $expected\ value = mean = \mu = \mathbb{E}[x] = \sum_{i=1}^{k} p_i a_i = \sum_{i=0}^{k} p_0 x_0 + p_1 x_1 + \cdots p_k x_k$
- expectation is property of probability distribution
- expectation of some function f over a probability distribution P of an outcome x; By the ***definition of expectation***:

$$\mathbb{E}_{P(x)}[f(x)] = \sum_{i=1}^{K} p_i\, f(a_i)$$

Rules:

a. Expectation of a constant with respect to x is the constant: $\mathbb{E}[c] = c\sum_{i=1}^{k} p_i = c$
b. Expectation is linear operator $\mathbb{E}[f(x) + g(x)] = \mathbb{E}[f(x)] + \mathbb{E}[g(x)]$ and $\mathbb{E}[cf(x)] = c\mathbb{E}[f(x)]$
c. Expectation of independent outcomes separate: $\mathbb{E}[f(x)g(y)] = \mathbb{E}[f(x)]\ \mathbb{E}[g(y)]$ if x and y are independent.

Kullback-Leibler Divergence

- $KL\big(P(x) \parallel Q(x)\big)$ is the *measure of inefficiency* in using the probability distribution *Q* to approximate the true probability distribution *P* ; similar to the *distance* between distributions Q and P (although **not symmetric**)
- also known as relative entropy, or KL-divergence, or KL-distance
- KL Divergence is very similar to **entropy** - except it tells us how much information is lost when we substitute our observed distribution for a parametrized approximation

$p(x)$	unknown true distribution; intractable (cannot take Expectation of it)
$q(x)$	approximating distribution; can take expectation

Not sure ?

- Infer KL-divergence by taking ***likelihood ratio (LR)*** between the two distributions:

$$LR = \frac{q(x)}{p(x)}$$

- indicates how likely a data-point is to occur in $q(x)$ as opposed to $p(x)$
- take the log to better quantify

$$\log LR = \log\left(\frac{q(x)}{p(x)}\right)$$

- if we have large set of data sampled from q(x), how much will each sample on average indicate that q(x) better describes the data than p(x)?
- calculate average predictive power by sampling N points from p(x) and normalizing the sum for each sample,

$$\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{q(x_i)}{p(x_i)}\right)$$

- using definition of expectation$\sum_{i=1}^{K} p_i\, f(a_i) = \mathbb{E}_{P(x)}[f(x)]$

$$\frac{1}{N}\sum_{i=1}^{N}\log\left(\frac{q(x_i)}{p(x_i)}\right) = \mathbb{E}_{q(x)}\left[\log\left(\frac{q(x)}{p(x)}\right)\right]$$

Not sure ?

$$KL\big(q(x) \parallel p(x)\big) = \mathbb{E}_{q(x)}\left[\log\left(\frac{q(x)}{p(x)}\right)\right]$$

- *Note: The numerator of the log is what the expectation is taken over.*

Can be rewritten by the definition of expectation, $\mathbb{E}_{P(x)}[f(x)] = \sum_{i=1}^{K} p_i\, f(a_i)$

$$KL\big(q(x) \parallel p(x)\big) = \mathbb{E}_{q(x)}\left[\log\left(\frac{q(x)}{p(x)}\right)\right] = \sum_{x\in\mathcal{X}} q(x)\log\left(\frac{q(x)}{p(x)}\right)$$

Converting the division in the log to subtraction we find:

$$KL\big(q(x) \parallel p(x)\big) = \sum_{x_i\in\mathcal{X}} q(x_i)(\log(q(x_i)) - \log(p(x_i)))$$

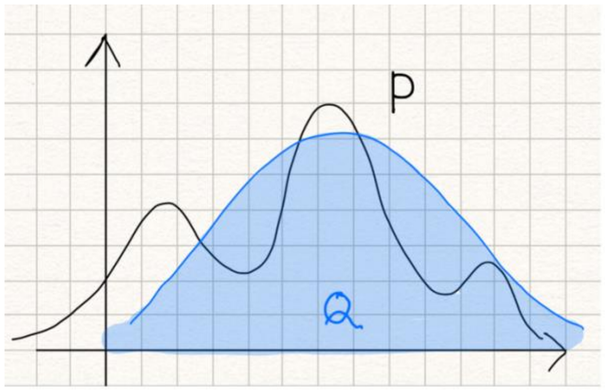
Entropy in KL:

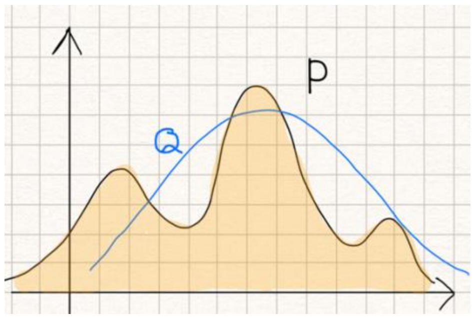
$$\sum_{i=0}^{n} p(x_i)\log\big(p(x_i)\big)$$

- *entropy of distribution p ; expected value of the information content of P*
- note if you break up sum above, you will get entropy of q.

KL-Distance Characteristics:

- not a metric because it is not symmetric, $KL\big(q(x) \parallel p(x)\big) \neq KL\big(p(x) \parallel q(x)\big)$
- $KL\big(q(x) \parallel p(x)\big) \geq 0$ Gibbs Inequality
  -
- $KL\big(q(x) \parallel p(x)\big) = 0$ if $q = p$ almost everywhere
  - q can have area where p does not, even when above = 0





$D_{KL}(P(x) \parallel Q(x))$ $D\_KL\ (Q(x)/\!/P(x))$

Extraneous:
From Bishop, page 55
If we use q(x) to make a coding scheme to transmit value x, the average additional amount of information (measured in nats) required to specify value of x is $KL(p||q)$

$$\begin{aligned}\mathrm{KL}(p\|q) &= -\int p(\mathbf{x})\ln q(\mathbf{x})\,\mathrm{d}\mathbf{x} - \left(-\int p(\mathbf{x})\ln p(\mathbf{x})\,\mathrm{d}\mathbf{x}\right)\\ &= -\int p(\mathbf{x})\ln\left\{\frac{q(\mathbf{x})}{p(\mathbf{x})}\right\}\mathrm{d}\mathbf{x}.\end{aligned}$$

Iff $p(x) = q(x)$ then $KL(p||q) \geq 0$
Using Jensen's Inequality below, where

$$f(x) = \ln\left\{\frac{q(x)}{p(x)}\right\} dx$$

$$KL(p||q) = -p(x)\ln\left(\int \frac{q(x)}{p(x)}\right)\mathrm{dx} \geq -ln\int x$$

convex

- convex function has every chord above or on the function
- the requirement for convexity says that the 2nd derivative of the function be everywhere positive
    - $f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda)f(b)$
    - $x = a,\ x = b$ interval
    - $value\ of\ function \leq point\ on\ chord$

strictly convex

- equality $f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda)f(b)$ holds only for $\lambda = 0\ and\ 1$

concave

- opposite of convex
- every chord lies on or below
- if $f(x)$ is convex, $-f(x)$ is concave

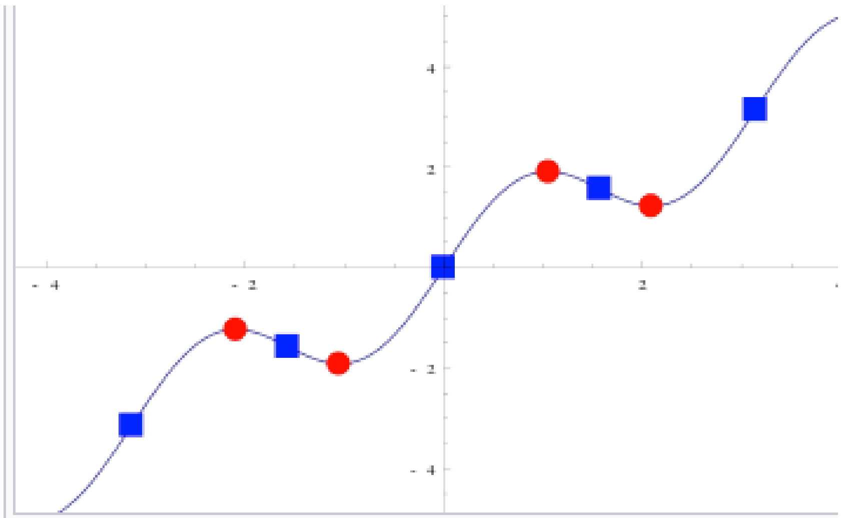
Jensen's inequality

- rewrite $f(\lambda a + (1-\lambda)b) \leq \lambda f(a) + (1-\lambda) f(b)$ using proof by induction:
- $$f\left(\sum_{i=1}^{M} \lambda_i x_i\right) \leqslant \sum_{i=1}^{M} \lambda_i f(x_i)$$
- in the continuous form:
-

Necessary Background – Stationary vs Fixed Points

Stationary point

- point of differentiable function where function's derivative is zero
$$\frac{df}{dx} = 0$$
- minima, maxima, and inflections



The stationary points are the red circles. In this graph, they are all relative maxima or relative minima. The blue squares are inflection points.

- example: in Yedidia, we are looking for the stationary points in the free energy optimization, since it is the point that minimizes the free energy

fixed point

- element of function's domain that is mapped to itself by the function
- $f(x) = x$
- example: in belief propagation, we find fixed points once the model has converged
  - assume we have a sequence of $z: \{z_1, z_2, \dots, z_{k+1}, \dots\}$
  - Once $z_{k+1} = g(z_k)$ we have converged because $z_{k+1} = z_k$ and changing k no longer changes z
  - once all messages stop changing, we have fixed points of z:
    - $m_{t,s}^{k+1}(x_s) = m_{t,s}^{k}(x_s)$
    - message from t to s at iteration k+1 is the same as it was at iteration k
  - we may never reach a fixed point if graph is not a tree or has cycles (same as never converging)

side: in Yedidia, Bethe Free Energy.

- by proving the fixed points of belief propagation (the ultimately converged value of messages) is the same as the stationary point of Bethe Free energy (the minimum of the free energy optimization) we know that minimizing the free energy is the same as converging on belief propagation network

Additional Papers
Long, but still in draft:
Bethe free energy, Kikuchi approximations, and belief propagation algorithms

Later, more updated version:
Constructing Free Energy Approximations and Generalized Belief Propagation Algorithms