UC Berkeley
Department of Electrical Engineering and Computer Science
Department of Statistics

EECS 281A / STAT 241A Statistical Learning Theory

Solution to Problem Set 7
Fall 2012

Issued: Tues. Nov. 13, 2012 **Due:** Thurs. Nov. 29, 2012

Reading: Sampling chapter. Notes on mean field algorithm.

Problem 7.1

Gibbs sampling and mean field: Consider the Ising model with binary variables $X_s \in \{-1, 1\}$, and a factorization of the form

$$p(x; \theta) \propto \exp\Big\{ \sum_{s \in V} \theta_s x_s + \sum_{(s,t) \in E} \theta_{st} x_s x_t \Big\}. \tag{1}$$

To make the problem symmetric, assume a 2-D grid with toroidal (donut-like) boundary conditions, as illustrated in Figure 1(a).

(a) Derive the Gibbs sampling updates for this model. Implement the algorithm for $\theta_{st} = 0.25$ for all edges, and $\theta_s = (-1)^s$ for all $s \in \{1, \ldots, 49\}$ (using the node ordering in Figure 1(a)). Run a burn-in period of 1000 iterations (where one iteration amounts to updating each node once). For each of 5000 subsequent iterations, collect a sample vector, and use the 5000 samples to form Monte Carlo estimates $\widehat{\mu}_s$ of the moments $\mathbb{E}[X_s]$ at each node. Output a 7×7 matrix of the estimated moments. Repeating this same experiment a few times will provide an idea of the variability in your estimate. Hand in print-outs of your code, as well as your results.

Solution: Fix a node $s \in V$ and let $\alpha_s = \theta_s + \sum_{t \in \mathcal{N}(s)} \theta_{st} x_t$, where $\mathcal{N}(s)$ is the set of neighbors of s. Then the Gibbs update for node s is

$$X_s = \begin{cases} 1 & \text{w.p.} \quad \frac{\exp\{\alpha_s\}}{\exp\{\alpha_s\}+\exp\{-\alpha_s\}} \\ -1 & \text{w.p.} \quad \frac{\exp\{-\alpha_s\}}{\exp\{\alpha_s\}+\exp\{-\alpha_s\}} \end{cases}$$

A sample output of the estimated moment matrix from Gibbs sampling

is as follows:

$$\begin{pmatrix} -0.7668 & 0.5072 & -0.6176 & 0.5552 & -0.6028 & 0.5220 & -0.7768 \\ 0.5400 & -0.4188 & 0.3964 & -0.4232 & 0.4288 & -0.4048 & 0.5628 \\ -0.5432 & 0.4312 & -0.4432 & 0.4696 & -0.4340 & 0.3904 & -0.5464 \\ 0.5772 & -0.4212 & 0.4272 & -0.4440 & 0.4264 & -0.4436 & 0.5884 \\ -0.5836 & 0.4020 & -0.4484 & 0.4384 & -0.3972 & 0.4268 & -0.5608 \\ 0.5584 & -0.4340 & 0.4160 & -0.4700 & 0.4240 & -0.3748 & 0.5460 \\ -0.7820 & 0.4516 & -0.7132 & 0.5244 & -0.6328 & 0.5024 & -0.7952 \end{pmatrix}.$$

(b) Derive the naive mean field updates (based on a fully factorized approximation), and implement them for the same model. Compute the average ℓ_1 distance $\frac{1}{49}\sum_{i=1}^{49} |\tau_s - \widehat{\mu}_s|$ between the mean field estimated moments τ_s, and the Gibbs estimates $\widehat{\mu}_s$. Hand in print-outs of your code, as well as your results.

Solution: Taking the naive approach, we want to find

$$\sup_{\tau_s \in [-1,1]} \Big\{ \sum_{s \in V} \theta_s \tau_s + \sum_{st \in E} \theta_{st} \tau_s \tau_t + \sum_{s \in V} H(\tau_s) \Big\},$$

where

$$H(X_s) = -\left(\frac{1+\tau_s}{2}\right) \log\left(\frac{1+\tau_s}{2}\right) - \left(\frac{1-\tau_s}{2}\right) \log\left(\frac{1-\tau_s}{2}\right).$$

Let $s \in V$; we now want to solve the problem above for τ_s by holding $(\tau_t,\ t \neq s)$ fixed. It is easy to see that the above problem is concave in τ_s, so by taking derivative with respect to τ_s and setting it equal to 0 we obtain

$$\theta_s + \sum_{t \in \mathcal{N}(s)} \theta_{st} \tau_t - \frac{1}{2} \log\left(\frac{1+\tau_s}{1-\tau_s}\right) = 0.$$

Solving the equation above for τ_s gives us

$$\tau_s = \frac{\exp\{2\beta\} - 1}{1 + \exp\{2\beta\}},$$

where $\beta = \theta_s + \sum_{t \in \mathcal{N}(s)} \theta_{st} \tau_t$. This is the naive mean field update for τ_s. Note the relationship between parts (a) and (b). Namely, that if X_s is sampled as in part (a) and for each $t \in \mathcal{N}(s)$ we have $x_t = \tau_t = \mathbb{E}[X_t]$, then

$$\mathbb{E}[X_s] = \frac{\exp\{\beta\} - \exp\{-\beta\}}{\exp\{\beta\} + \exp\{-\beta\}} = \tau_s.$$

A sample output of the estimated moment matrix from the naive mean field is as follows:

$$\begin{pmatrix} -0.8058 & 0.5862 & -0.6363 & 0.6196 & -0.6077 & 0.6573 & -0.8141 \\ 0.5905 & -0.4437 & 0.4576 & -0.4570 & 0.4675 & -0.4257 & 0.5920 \\ -0.6289 & 0.4585 & -0.4861 & 0.4812 & -0.4821 & 0.4645 & -0.6277 \\ 0.6203 & -0.4596 & 0.4792 & -0.4767 & 0.4842 & -0.4521 & 0.6218 \\ -0.6308 & 0.4556 & -0.4890 & 0.4813 & -0.4753 & 0.4802 & -0.6287 \\ 0.5824 & -0.4529 & 0.4454 & -0.4603 & 0.4868 & -0.3507 & 0.5720 \\ -0.8223 & 0.5733 & -0.6709 & 0.5985 & -0.5654 & 1.0000 & -1.0000 \end{pmatrix}.$$

In this case the average ℓ_1 distance is 0.05921.

```
% Script for Problem 7.1

k = 7;  % width of toroidal graph
thetaNodes = (-1).^(1:k^2);  % potential for each node
thetaEdge = 0.25;  % constant potential for all edges

% Generate adjacency matrix
adjMat = generateToroidal(7);

% Run Gibbs sampling and estimate moments
gibbsSamples = GibbsSampling(adjMat,thetaNodes,thetaEdge,1000,5000);
gibbsMean = mean(gibbsSamples,2);
gibbsMatrix = reshape(gibbsMean, [k k])'

% Run naive mean field
naiveEst = NaiveMeanField(adjMat,thetaNodes,thetaEdge);
naiveMatrix = reshape(naiveEst, [k k])'
dist = mean(abs(gibbsMean-naiveEst));
fprintf('Average ell_1 distance is %g.\n', dist);
```

```
function adjMat = generateToroidal(k)
% Generate the adjacency matrix of the k-by-k toroidal graph.

adjMat = zeros(k^2, k^2);
for i = 1:k
    for j = 1:k
        ind = toInd(i,j);  % linear index of the pair
        nbhs = getNeighbors(i,j);  % get neighbors of the node
        for p = 1:length(nbhs)
            nbh_ind = toInd(nbhs{p}(1), nbhs{p}(2));
            adjMat(ind, nbh_ind) = 1;
```

```
        end
    end
end

% Helper function to convert from pair (i,j) to linear
% index in {1, ..., k^2}
function ind = toInd(i,j)
    ind = (i-1) * k + j;
end

% Helper function to get the neighbors of node (i,j)
function neighbors = getNeighbors(i,j)
    neighbors = {[i-1,j], [i+1,j], [i,j-1], [i,j+1]};
    if (i == 1), neighbors{1} = [k,j]; end
    if (i == k), neighbors{2} = [1,j]; end
    if (j == 1), neighbors{3} = [i,k]; end
    if (j == k), neighbors{4} = [i,1]; end
end

end
```

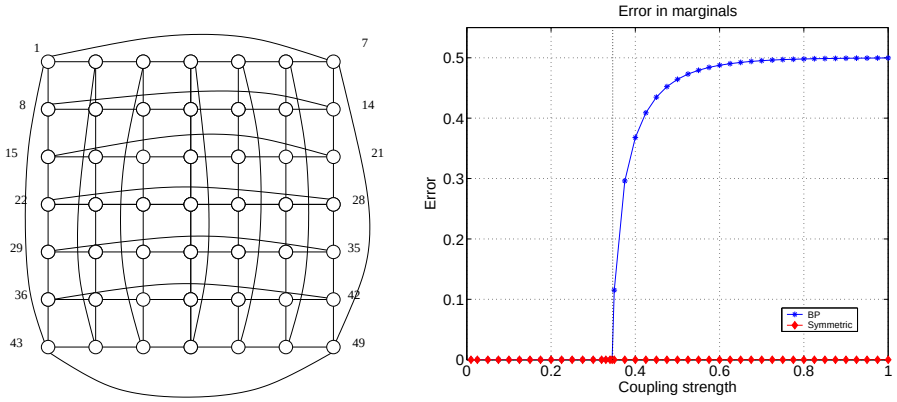
```
function samples = GibbsSampling(...
    adjMat, thetaNodes, thetaEdge, burnIn, numSamples)
% Implement Gibbs sampling for the {-1,+1} Ising model with
% constant edge potentials.

n = size(adjMat,1);  % number of vertices
samples = zeros(n, numSamples);  % placeholder for samples
X = ones(n,1);  % initial configuration
X(rand(n,1) > 0.5) = -1;  % randomize

numIters = burnIn + numSamples;
for it = 1:numIters
    perm = randperm(n);  % random ordering of the indices
    for s = 1:perm  % iterate in order
        Xnbhs = X(adjMat(s,:) ≠ 0);  % neighbors of s
        alpha = thetaNodes(s) + thetaEdge * sum(Xnbhs);
        if (rand < exp(alpha)/(exp(alpha) + exp(-alpha)))
            X(s) = 1;
        else
            X(s) = -1;
        end
    end
    if (it > burnIn)
        samples(:, it-burnIn) = X;  % record sample
    end
end
```

```
function est = NaiveMeanField(adjMat, thetaNodes, thetaEdge)
% Implement naive mean field for the Ising model with constant
% edge potentials.

n = size(adjMat,1);  % number of vertices
eps = 1e-6;  % threshold for convergence
est = ones(n,1);  % initial configuration
est(rand(n,1) > 0.5) = -1;  % randomize
Δ = 1;  % change from previous estimate
numIter = 0;  % number of iterations so far
while (Δ > eps)
    oldEst = est;  % old estimate
    numIter = numIter + 1;
    perm = randperm(n);  % random ordering of the indices
    for s = 1:perm  % iterate in order
        estNbhs = est(adjMat(s,:) ≠ 0);  % neighbors of s
        beta = thetaNodes(s) + thetaEdge * sum(estNbhs);
        est(s) = (exp(2*beta) - 1) / (1 + exp(2*beta));
    end
    Δ = norm(est - oldEst, Inf);
end
fprintf('Naive mean field converges in %d iterations!\n', numIter);
```



(a) (b)

Figure 1: (a) A two-dimensional grid graph with toroidal boundary conditions. (b) Break-down of the sum-product algorithm on the Ising model on the toroidal grid. For all $\gamma < \gamma^*$, the sum-product algorithm computes the correct symmetric marginals. Beyond this point, it outputs increasingly inaccurate answers.

Problem 7.2

Neighborhood regression for Gaussian graphical models: In this problem, we explore properties of jointly Gaussian random vectors that underlie the success of the neighborhood-based Lasso approach to estimating Gaussian graphical models (as discussed in lecture).

Let $(X_1, X_2, \ldots, X_p)$ be a zero-mean jointly Gaussian random vector with positive definite covariance matrix Σ. Letting $T = \{2, 3, \ldots, p\}$, consider the conditioned random variable $Z = (X_1 \mid X_T)$.

(a) Show that there is a vector $\theta \in \mathbb{R}^{p-1}$ such that $Z = \langle \theta, X_T \rangle + W$ where W is a zero-mean Gaussian variable independent of X_T. *Hint:* Consider the best linear predictor of X_1 given X_T.

Solution: As suggested in the hint, we define θ to be the best linear predictor of X_1 given X_T:

$$\theta = \arg \min_{\gamma \in \mathbb{R}^{p-1}} \mathbb{E}[(X_1 - \langle \gamma, X_T \rangle)^2]. \tag{2}$$

Then it is easy to see that $W = Z - \langle \theta, X_T \rangle$ is a zero-mean Gaussian random variable independent of X_T, for if W still has a nonzero correlation with X_T then we can decrease the mean-squared error in (2) further.

Alternatively, we can use the conditioning formula for multivariate Gaussian distribution. Recall that since $X = (X_1, X_T)$ is jointly Gaussian:

$$\begin{pmatrix} X_1 \\ X_T \end{pmatrix} \sim \mathcal{N}\left(\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \Sigma_{11} & \Sigma_{1T} \\ \Sigma_{T1} & \Sigma_{TT} \end{pmatrix} \right),$$

the random variable $Z = (X_1 \mid X_T)$ has the distribution (e.g. see equations (13.26) and (13.27) in the course reader)

$$Z \sim \mathcal{N}\big(\Sigma_{1T}\Sigma_{TT}^{-1} X_T,\ \Sigma_{11} - \Sigma_{1T}\Sigma_{TT}^{-1}\Sigma_{T1}\big).$$

This means we can write

$$Z = \Sigma_{1T}\Sigma_{TT}^{-1} X_T + W = \langle \Sigma_{TT}^{-1}\Sigma_{T1}, X_T \rangle + W$$

for a random variable $W \sim \mathcal{N}\big(0, \Sigma_{11} - \Sigma_{1T}\Sigma_{TT}^{-1}\Sigma_{T1}\big)$ that is independent of X_T.

(b) Show that $\theta = (\Sigma_{TT})^{-1}\Sigma_{T1}$, where $\Sigma_{T1} \in \mathbb{R}^{p-1}$ is the vector of covariances between X_1 and X_T.

Solution: If we use the multivariate Gaussian conditioning approach in part (a), then we already know that $\theta = (\Sigma_{TT})^{-1}\Sigma_{T1}$. On the other hand, it is also easy to compute the value of θ if we use the best linear predictor technique. We can write the objective function in (2) as

$$\begin{aligned} L(\gamma) &= \mathbb{E}[(X_1 - \langle \gamma, X_T \rangle)^2] \\ &= \gamma^\top \mathbb{E}[X_T X_T^\top]\, \gamma - 2\mathbb{E}[X_1 X_T^\top]\gamma + \mathbb{E}[X_1^2] \\ &= \gamma^\top \Sigma_{TT}\, \gamma - 2\Sigma_{T1}^\top \gamma + \Sigma_{11}, \end{aligned}$$

which is a convex quadratic function of γ. Thus, the minimizer θ of L satisfies the stationary condition

$$\nabla L(\theta) = 2\Sigma_{TT}\theta - 2\Sigma_{T1} = 0,$$

which gives us $\theta = (\Sigma_{TT})^{-1}\Sigma_{T1}$, as desired.

(c) Letting $N(1) = \{j \in V \mid (1, j) \in E\}$ be the neighborhood set of node i, show that $\theta_j = 0$ if and only if $j \notin N(1)$. *Hint:* The following elementary fact could be useful: let A be an invertible matrix, given in the block-partitioned form

$$A = \begin{bmatrix} A_{11} & A_{12} \\ A_{21} & A_{22} \end{bmatrix}.$$

Then letting $B = A^{-1}$, we have $B_{12} = A_{11}^{-1} A_{12} \big[A_{21} A_{11}^{-1} A_{12} - A_{22}\big]^{-1}$.

Solution: Recall that from the Hammersley-Clifford theorem applied to multivariate Gaussian graphical model (e.g. see the solution to Problem 3.4(b) in Homework 3), the sparsity pattern of the inverse covariance matrix $\Lambda = \Sigma^{-1}$ encodes the graph structure, i.e. for each $i \neq j \in V$ we have $(i, j) \notin E$ if and only if $\Lambda_{ij} = 0$. Thus, to apply this theorem we only need to express θ in terms of Λ, which we do via block matrix inversion.

Writing Λ in block-partitioned form as

$$\Lambda = \begin{pmatrix} \Lambda_{11} & \Lambda_{1T} \\ \Lambda_{T1} & \Lambda_{TT} \end{pmatrix},$$

we find an expression of Σ in terms of the blocks of Λ as follows (see equation (13.16) in the course reader):

$$\Sigma = \Lambda^{-1} = \begin{pmatrix} \Lambda_{11}^{-1} + \Lambda_{11}^{-1}\Lambda_{1T}(\Lambda/\Lambda_{11})^{-1}\Lambda_{T1}\Lambda_{11}^{-1} & -\Lambda_{11}^{-1}\Lambda_{1T}(\Lambda/\Lambda_{11})^{-1} \\ -(\Lambda/\Lambda_{11})^{-1}\Lambda_{T1}\Lambda_{11}^{-1} & (\Lambda/\Lambda_{11})^{-1} \end{pmatrix},$$

where $(\Lambda/\Lambda_{11}) = \Lambda_{11} - \Lambda_{1T}\Lambda_{TT}^{-1}\Lambda_{T1}$ is the Schur complement of Λ with respect to Λ_{11}. From this we see that $\Sigma_{T1} = -(\Lambda/\Lambda_{11})^{-1}\Lambda_{T1}\Lambda_{11}^{-1}$ and $\Sigma_{TT} = (\Lambda/\Lambda_{11})^{-1}$, which allows us to write θ in terms of Λ:

$$\theta = (\Sigma_{TT})^{-1}\Sigma_{T1} = -(\Lambda/\Lambda_{11})(\Lambda/\Lambda_{11})^{-1}\Lambda_{T1}\Lambda_{11}^{-1} = -\Lambda_{T1}\Lambda_{11}^{-1}.$$

Thus, for each $j \neq 1$ the component of θ corresponding to X_j is

$$\theta_j = -\frac{\Lambda_{1j}}{\Lambda_{11}}.$$

Therefore, we conclude that $\theta_j = 0$ if and only if $\Lambda_{1j} = 0$, which by the Hammersley-Clifford theorem is equivalent to the condition that $j \notin N(1)$.

Problem 7.3

Sum-product on graphs with cycles In many real-world applications, the sum-product algorithm is applied to graphs with cycles. Unlike the case of trees, the sum-product updates are no longer guaranteed to converge, or to compute the correct marginal distributions. Indeed, the results can be very surprising! As an illustration of this phenomenon, consider the Ising model (1) on the toroidal grid (see Figure 1(a)), with $\theta_s = 0$ for all $s \in V$, and $\theta_{st} = \gamma$ for all edges (s,t); call this distribution $p(x;\gamma)$, since it is parameterized by $\gamma \in \mathbb{R}$.

(a) Show that the single node marginal distributions are uniform for all choices of γ (that is, $\mathbb{P}[X_s = 1;\gamma] = \mathbb{P}[X_s = -1;\gamma] = 0.5$).

Solution: The probability distribution now takes the form

$$p(x;\gamma) \propto \exp\Big(\gamma \sum_{(s,t)\in E} x_s x_t\Big), \quad x \in \{-1,+1\}^{49}.$$

Fix a node s, and let $x_{\setminus s} = (x_t : t \neq s)$ denote the random variables corresponding to all other nodes except node s. For simplicity, we write the exponential term as a function of x_s and $x_{\setminus s}$:

$$g(x_s, x_{\setminus s};\gamma) = \exp\Big(\gamma \sum_{(t,u)\in E} x_t x_u\Big).$$

With this notation, the marginal of node s can be written as

$$p(x_s;\gamma) = \sum_{x_{\setminus s}} p(x_s, x_{\setminus s};\gamma) \propto \sum_{x_{\setminus s}} g(x_s, x_{\setminus s};\gamma),$$

where the summation is over 2^{48} possible configurations of $x_{\setminus s}$. Now observe that since the random variables appear in pairs in the function g, for each $x_s \in \{-1, +1\}$ and each configuration of $x_{\setminus s}$ we have

$$g(x_s, x_{\setminus s}; \gamma) = g(-x_s, -x_{\setminus s}; \gamma).$$

Moreover, summing over all possible configurations of $x_{\setminus s}$ is equivalent to summing over all possible configurations of $-x_{\setminus s}$, so we also have

$$\sum_{x_{\setminus s}} g(x_s, x_{\setminus s}; \gamma) = \sum_{x_{\setminus s}} g(-x_s, -x_{\setminus s}; \gamma) = \sum_{x_{\setminus s}} g(-x_s, x_{\setminus s}; \gamma).$$

Therefore,

$$\mathbb{P}[X_s = 1; \gamma] = \frac{\sum_{x_{\setminus s}} g(1,\ x_{\setminus s}; \gamma)}{\sum_{x_{\setminus s}} g(1,\ x_{\setminus s}; \gamma) + \sum_{x_{\setminus s}} g(-1,\ x_{\setminus s}; \gamma)} = \frac{1}{2},$$

as desired.

(b) Figure 1(b) shows empirically that there is some critical threshold $\gamma^* > 0$ such that the sum-product algorithm, when applied to the distribution $p(x; \gamma)$, converges to uniform marginal distributions for all $\gamma \in [0, \gamma^*)$, but produces inaccurate answers for larger γ. Using the analytical form of the sum-product updates, prove that there is some γ^* that sum-product converges from any initial condition, and computes the correct uniform marginals for all $\gamma \in [0, \gamma^*)$. *Hint:* Since each node in the model looks like every other node, it is sufficient to consider a special case of the message-passing updates, in which the message $M_{t \to s}$ along each edge $t \to s$ is the same as every other message.

Solution: We initialize the messages $M_{t \to s} = (M_{t \to s}(1), M_{t \to s}(-1))$ on all edges (s, t) to be the same vector $m_0 = (m_0(1), m_0(-1))$. We then run the sum-product algorithm with flooding schedule, where at each iteration all the nodes simultaneously send messages to all their neighbors. Since the initial messages are the same, this procedure guarantees that at each iteration the messages across all edges are the same. Letting $m_t = (m_t(1), m_t(-1))$ denote the message at time t, we find that the messages change by the following rule:

$$\begin{aligned} m_{t+1}(1) &= e^{\gamma} m_t^3(1) + e^{-\gamma} m_t^3(-1) \\ m_{t+1}(-1) &= e^{-\gamma} m_t^3(1) + e^{\gamma} m_t^3(-1). \end{aligned} \tag{3}$$

The question then is what values of γ make the messages converge for any initial message m_0. At this point it is important to take a step back and think about what we mean with "converge". If we start with the initial message $m_0 = (1, 1)$, then it is easy to see that for any value of $\gamma \geq 0$, at each iteration we still have $m_t(1) = m_t(-1)$, but $m_t(1) \uparrow \infty$ as $t \to \infty$; so the messages themselves may diverge. However, what we ultimately care about is the marginal on each node:

$$p(x_s; \gamma) = \frac{m_\infty^4(x_s)}{m_\infty^4(1) + m_\infty^4(-1)}.$$

Here m_∞ denotes the limit of the messages (m_t), and we take the fourth power of the messages because each node has four neighbors. Thus, we can rescale or normalize the messages as long as the marginals are preserved. In our cautionary tale above, the ratio $\frac{m_t^4(1)}{m_t^4(1)+m_t^4(-1)} = \frac{1}{2}$ is a constant for all t, so in that sense we want to say that the sum product algorithm converges.

There are a variety of ways to normalize the messages, but we will consider the ratio $p_t = m_t(1)/m_t(-1)$; you could also consider other quantities such as the log ratio or normalize the messages so that $m_t(1) + m_t(-1) = 1$. From (3), we see that the ratio p_t evolves by

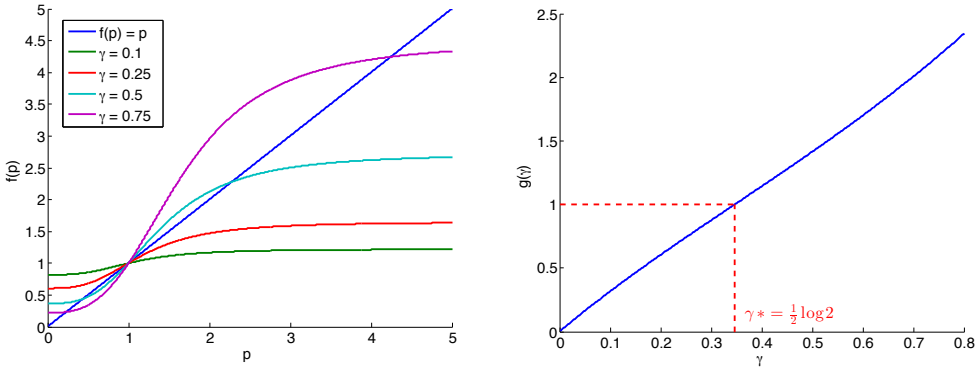
$$p_{t+1} = \frac{m_{t+1}(1)}{m_{t+1}(-1)} = \frac{e^\gamma m_t^3(1) + e^{-\gamma} m_t^3(-1)}{e^{-\gamma} m_t^3(1) + e^\gamma m_t^3(-1)} = \frac{e^\gamma p_t^3 + e^{-\gamma}}{e^{-\gamma} p_t^3 + e^\gamma} = \frac{e^{2\gamma} p_t^3 + 1}{p_t^3 + e^{2\gamma}}.$$

Letting $f_\gamma \colon \mathbb{R}_+ \to \mathbb{R}_+$ denote the function $f_\gamma(p) = (e^{2\gamma} p^3 + 1)/(p^3 + e^{2\gamma})$, we now have the function iteration $p_{t+1} = f_\gamma(p_t)$. If the sequence (p_t) converges, then it necessarily converges to a fixed point of the function f_γ. Note that $p = 1$ is a fixed point of f_γ for all $\gamma \geq 0$, and when the ratio $p_t = 1$ the marginals are uniform, as required. For small γ, $p = 1$ is the only fixed point of f_γ, while for larger values of γ the function f_γ has more than one fixed point; see Figure 2(a). Thus, intuitively, there is a threshold γ^* such that if $0 \leq \gamma < \gamma^*$ then the function f_γ only has one fixed point and the function iteration converges to the fixed point $p \to 1$; let us now prove this.

By the Banach fixed-point theorem, a sufficient condition to guarantee convergence of function iteration is if the function f_γ is contractive:

$$|f_\gamma(p) - f_\gamma(q)| \leq L|p - q|$$

for all $p, q \in \mathbb{R}_+$ and for some constant $0 < L < 1$. Intuitively, if f_γ is contractive then the distance between successive points p_t are



(a) (b)

Figure 2: (a) The plot of the function $f_\gamma(p)$ for several different values of γ. In this case the function f_γ only has one fixed point for $\gamma = 0.1, 0.25$ and it has three fixed points for $\gamma = 0.5, 0.75$. (b) Plot of the function $g(\gamma) = \max_{p>0} |f'_\gamma(p)|$. The function $g(\gamma)$ is increasing and achieves the value 1 at $\gamma^* = \frac{1}{2}\log 2 \approx 0.3466$.

exponentially decreasing to zero. In particular, a contractive function has a unique fixed point, which can be found via repeated function iterations starting from an arbitrary point. The contractive property of f_γ is equivalent to saying that f_γ is L-Lipschitz with $L < 1$, which holds as long as $|f'_\gamma(p)| \leq L$ for all $p \in \mathbb{R}_+$. An easy computation yields

$$f'_\gamma(p) = \frac{3p^2(e^{4\gamma} - 1)}{(p^3 + e^{2\gamma})^2},$$

which, in particular, shows that $f'_\gamma(p) > 0$. Now define

$$g(\gamma) = \sup_{p \in \mathbb{R}_+} f'_\gamma(p).$$

Then the regime of γ that ensures convergence is the set $\{\gamma \colon g(\gamma) < 1\}$. We can evaluate $g(\gamma)$ by considering the second derivative of f_γ:

$$f''_\gamma(p) = \frac{6p(e^{4\gamma} - 1)(e^{2\gamma} - p^3)}{(p^3 + e^{2\gamma})^3}.$$

From this expression we see that $f''_\gamma(p) = 0$ if and only if $p = p^* :=$

$2^{-1/3}e^{2\gamma/3}$, and that f'_γ achieves its maximum at $p = p^*$. Therefore,

$$g(\gamma) = f'_\gamma(p^*) = \frac{2^{4/3}}{3}\frac{(e^{4\gamma}-1)}{e^{8\gamma/3}}.$$

We can quickly check that $g(\gamma)$ is increasing in γ (see Figure 2(b)) and that $g(\gamma) = 1$ at $\gamma = \frac{1}{2}\log 2$. Therefore, we conclude that $\gamma^* = \frac{1}{2}\log 2 \approx 0.3466$, and the sum-product algorithm correctly computes the uniform marginals from any initial condition for all $\gamma \in [0, \gamma^*)$.