Probabilistic Graphical Models

Daphne Koller, Kevin Murphy
Winter 2011-2012

Feedback — PA5 Quiz

You achieved a score of 38.00 out of 38.00

Question 1

Recall our function **CheckConvergence.m** which uses the difference between a message before and after it is updated (called the ``residual") as a criterion for convergence. While running LBP with our naive message ordering on the network created by **ConstructRandNetwork** with on-diag weight .3 and off-diag weight .7, print out and plot the residuals of the message $19 \rightarrow 3, 15 \rightarrow 40,$ and $17 \rightarrow 2$, with the iteration number on the x-axis (you may want to change the range of the y-axis). Do these messages converge at the same rate? Which converges fastest? (Note, it will be easiest do this assessment within **ClusterGraphCalibrate** and use the helper function **MessageDelta** within that file).

Your Answer		Score	Explanation
Message $19 \rightarrow 3$ converges quickly, followed by $17 \rightarrow 2$ and finally $15 \rightarrow 40$	✔	5.00	
Total		5.00	

Question 2

Which of the following are true about the effects the message passing order can have cluster graph calibration?

Your Answer		Score	Explanation
The value of the final marginals in a graph with no loops can depend on the message passing order.	✔	0.80	Without loops, LBP is equivalent to clique tree inference which is exact and gives the same solution regardless of order.
Different message passing orders can lead to differences in the final full joint distribution given by cluster potentials.	✔	0.80	The joint distribution will never change, this is the cluster graph invariant.
✔ On the same graph, one ordering may converge while another never does.	✔	0.80	In some cases, bad message orderings can cause infinite loops in message changes that could be avoided with a different ordering.
✔ It can affect how long it takes for LBP to reach convergence.	✔	0.80	Consider our naive ordering, it keeps all passing messages even after some have converged. Using a smarter scheme could thus avoid extra work and speed up convergence.
✔ Different orderings can give different values of the final marginals.	✔	0.80	
Total		4.00	

Question 3

Now, consider the toy image network constructed in ConstructToyNetwork.m. Change the values of the on- and off-diagonal weights of the pairwise factors in this network to different values (which can be done by changing the values passed to this function). First try making the the weights on the diagonal much larger than the off-diagonal weights (1 and .2 respectively), then try the opposite where the off-diagonal weights are much larger (.2 and 1), and then finally try the case where the weights are roughly equal (.5 and .5). For each such model, run LBP and exact inference (using your code from PA4). Which of the following occur in this setup? Why? (NOTE: if LBP does not converge within 100,000 iterations it is okay to truncate the run and report on the pseudo-marginals given at that point)

Your Answer		Score	Explanation
☑ The case of low on diagonal weights with high off diagonal weights has poor convergence because of anti-correlation in the variables, which causes LBP to oscillate.	✔	0.83	
☐ All runs instances converge quickly to approximately the correct marginals due to LBP's ability to overcome local maxima.	✔	0.83	All statements are false
☑ The (.5,.5) case performs well relative to the others and converges quickly.	✔	0.83	
☐ The case of high on diagonal weights with low off diagonal weights converges quickly as compared to the others because the strong correlation causes us to quickly enter a strong local optima.	✔	0.83	This will cause positive feedback cycles because of the graph's short loops, preventing hasty convergence.
☐ The case of low on diagonal weights with high off diagonal weights converges quickly as the anti-correlation prevents nodes from strongly influencing one another, meaning our messages start out weak and agree quickly.	✔	0.83	Anti-correlation is still a strong force, which will cause nodes to oscillate and prevent quick convergence.
☑ The case of high on diagonal weights with low off diagonal weights has poor convergence because of high variable correlation coupled with short loops in the network, causing	✔	0.83	

positive feedback loops.

Total	5.00

Question explanation

Question 4

How many iterations did your smart message passing routine take to converge on the RandNewtork (from **ConstructRandNetwork.m**) with on-diagonal weight .3 and off diagonal weight .7?

2688

Your Answer		Score	Explanation
2688	✔	1.00	
Total		1.00	

Question explanation

Question 5

Let's run an experiment using our Gibbs sampling method. As before, use the toy image network and set the on-diagonal weight of the pairwise factor (in ConstructToyNetwork.m) to be 1.0 and the off-diagonal weight to be 0.1. Now run Gibbs sampling a few times, first initializing the state to be all 1's and then initializing the state to be all 2's. What effect does the initial assignment have on the accuracy of Gibbs sampling? Why does this effect occur?

Your Answer		Score	Explanation
The initial state has a significant impact on the result of our sampling, which makes sense as strong correlation makes mixing time long and we remain close to the initial assignment for a long time.	✔	5.00	
Total		5.00	

Question explanation

Question 6

Set the on-diagonal weight of our toy image network to .2 and off-diagonal weight to 1. Now visualize multiple runs with each of Gibbs, MHUniform, Swendsen-Wang variant 1, and Swendsen-Wang variant 2 using VisualizeMCMCMarginals.m (see TestToy.m for how to do this). How do the mixing times of these chains compare? How do the final marginals compare to the exact marginals? Why?

Your Answer		Score	Explanation
Gibbs outperforms the other variants in terms of mixing and the final marginals, but still has signs of a checkerboard pattern. The strong anti-correlation of variables causes the MH methods to overzealously switch adjacent nodes with the same label because of their focus on local properties, leading to an unrealistic checkerboard pattern.	✔	5.00	
Total		5.00	

Question explanation

Question 7

Set the on-diagonal weight of our toy image network to 1 and off-diagonal weight to .2. Now visualize multiple runs with each of Gibbs, MHUniform, Swendsen-Wang variant 1, and Swendsen-Wang variant 2 using VisualizeMCMCMarginals.m (see TestToy.m for how to do this). How do the mixing times of these chains compare? How do the final marginals compare to the exact marginals? Why?

Your Answer		Score	Explanation
The Swendsen-Wang variants outperform the other approaches, with faster mixing and better final marginals. This is likely due to the block-flipping nature of Swendsen-Wang which allows us to flip blocks and quickly mix in environments with strong agreeing potentials.	✔	5.00	
Total		5.00	

Question explanation

Question 8

Set the on-diagonal weight of our toy image network to .5 and off- diagonal weight to .5. Now visualize multiple runs with each of Gibbs, MHUniform, Swendsen-Wang variant 1, and Swendsen-Wang variant 2 using VisualizeMCMCMarginals.m (see TestToy.m for how to do this). How do the mixing times of these chains compare? How do the final marginals compare to the exact marginals? Why?

Your Answer		Score	Explanation
Gibbs and MHUniform perform very well and are somewhat better than the Swendsen-Wang variants. This is because the first two variants use local moves so the local marginals remained consistently close the the true marginals, while SW allows big swings over multiple variables that perturb the distribution.	✔	5.00	

Total	5.00

Question explanation

Question 9

When creating our proposal distribution for Swendsen-Wang, if you set all the $q_{i,j}$'s to zero, what does Swendsen-Wang reduce to?

Your Answer		Score	Explanation
Switching $q_{i,j}$ to 0 is equivalent to a randomized variant of Gibbs sampling where we are allowed to take a random, rather than fixed, order.	✔	3.00	Compare the resulting proposal distribution to our Gibbs proposal distribution to see that the two agree.
Total		3.00	