Homework 5
Daniel Crawford

Note: Due to time constraints and the need for my computer to be available, I was only able to accomplish all calculations on dataset1. Fortunately, this still visualizes the effectiveness of different approaches under different circumstances.

Dataset 1:
MLE Approach:

Training File / Approach	LL Diff	Average LL Diff
dataset1/train-f-1.txt	762957.95	3.814
dataset1/train-f-2.txt	229725.899	1.1486
dataset1/train-f-3.txt	59192.51	0.296
dataset1/train-f-4.txt	18436.23	0.092

EM Approach:

Training File / Approach	LL Diff average over 5 iterations	LL Diff standard deviation over 5 iterations	Average LL Diff
dataset1/train-p-1.txt	10540980.25	51617.50	52.70
dataset1/train-p-2.txt	1819451.92	14221.05	9.09
dataset1/train-p-3.txt	133485.93	85.82	0.667
dataset1/train-p-4.txt	19950.01	0.8698	0.0997

Randomly generated DAGS (Mixture Bayesian Networks)

k=2

Training File / Approach	LL Diff over 5 iterations	LL Diff standard deviation over 5 iterations	Average LL Diff over 5 iterations
dataset1/train-f-1.txt	344551.215	23320.94	1.723
dataset1/train-f-2.txt	102359.80	948.5203	0.5118
dataset1/train-f-3.txt	70290.68	1149.91	0.3514
dataset1/train-f-4.txt	63629.84	2410.84	0.318

k=4

Training File / Approach	LL Diff average over 5 iterations	LL Diff standard deviation over 5 iterations	Average LL Diff
dataset1/train-f-1.txt	340154.06	8080.82	1.700
dataset1/train-f-2.txt	97687.60	2442.78	0.4884
dataset1/train-f-3.txt	67765.899	2038.49	0.3388
dataset1/train-f-4.txt	63251.17	1487.87	0.3162

k=6

Training File / Approach	LL Diff average over 5 iterations	LL Diff standard deviation over 5 iterations	Average LL Diff
dataset1/train-f-1.txt	364766.343	41727.05	1.824
dataset1/train-f-2.txt	95386.67	3147.70	0.4769
dataset1/train-f-3.txt	66958.28	511.71	0.3348
dataset1/train-f-4.txt	61657.133	1973.12	0.3083

Questions:

- Based on your experimental results, compare the FOD-learn algorithm with the POD-EM-learn algorithm. What is the impact of missing data on LLDiff (Think of LLDiff as an error measure)

It shows that POD-EM learning requires a large amount of data in order for it to make a meaningful convergence. When we had a low amount of data (EX: 10 or 100 data points) we reached poor results with our model.

However, we can see around sample sizes of 1000 or higher that we started to reach very good results comparable to our MLE estimate. For example, on dataset1 when MLE and EM had 1000 data points each, we respectively see LLDiffs of approximately 59192, 133485. This LLDiff means that our estimation was off by less than a factor or 3, which is much better than previous sample sizes which had poor estimations which were off by a factor of 9. Most promisingly, we see at 10000 samples we have very close LLDiffs of 18436 and 19950, which are nearly the same.

The impact of missing data on EM seems to be that it requires more samples in order to reach an estimate as good as MLE.

- Based on your experimental results, compare the FOD-learn algorithms with the Mixture-Random-Bayes algorithm. What is the impact of not knowing the precise Bayesian network structure on LLDiff. How does k affect the accuracy of the learned model?

It appears that our Mixture-Random-Bayes algorithm is very good at quickly converging to a decent estimate of our true model. If we compare the results of some of our best performing mixture models on dataset1, we see that larger k's tend to improve performance, but they seem to bottleneck in performance. While using the largest datasets, for all k we see that we are slowly converging at around 60000 LLDiff. Comparatively, our FOD algorithm has about a 20000 LLDiff, meaning that we are off by nearly a factor of 3.

We can describe meeting this bottleneck as a limitation of not being aware about the true structure of our model. While at first, it appears our Mixture-Random-Bayes algorithm performs even better than FOD and EM with their true structures, we can see that they are approaching an LLDiff of 0, while ours may never capture that. Thus, LLDiff will usually be small, but far away from a true estimation.

In terms of k, we see that each model sees some performance improvements depending on k. This is likely because with a larger k, we have more chances to capture true dependencies. However, there is still a limitation because some DAGs will generate dependencies that are not actually there. Generally, we see our model has an easier time fitting the distribution using larger k values.