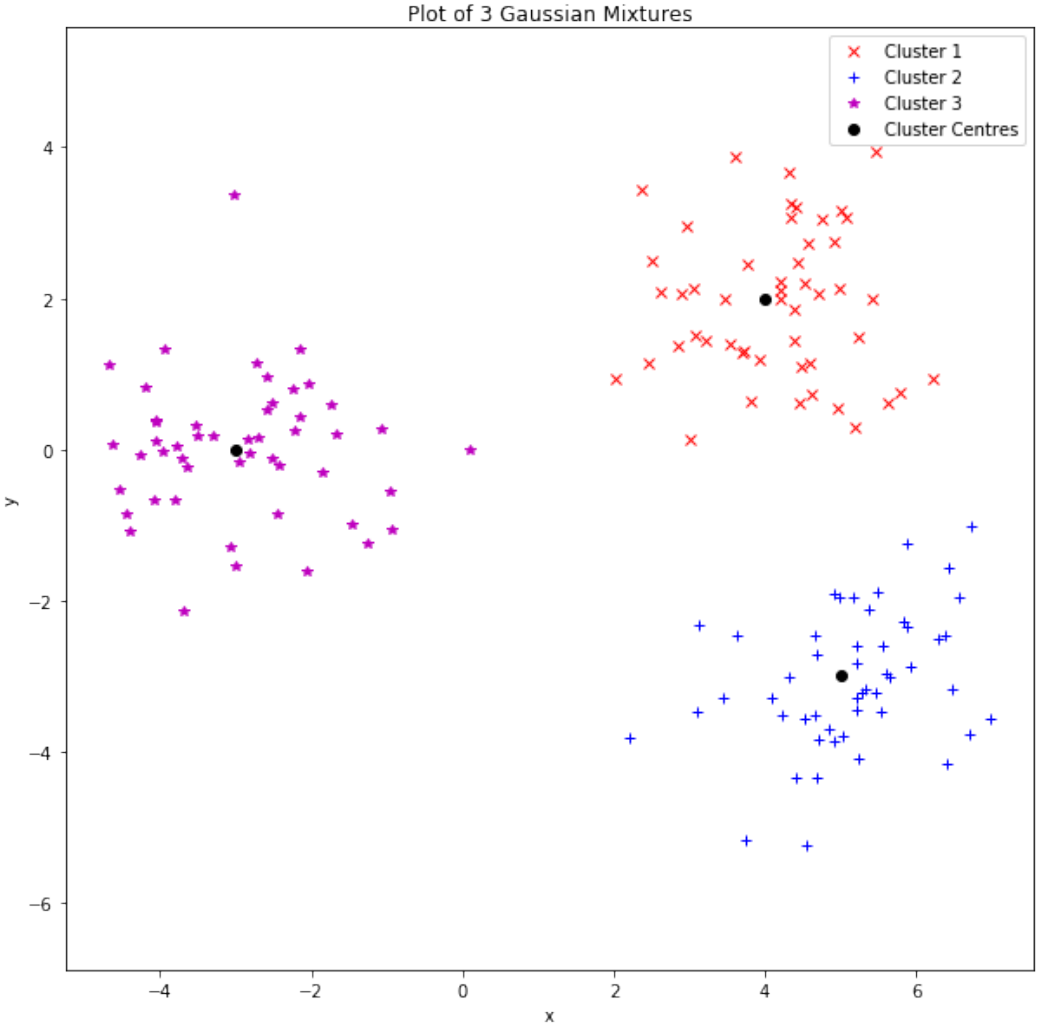
Foundations of Graphical Models
Problem Set #2

Si Kai Lee sl3950@columbia.edu

October 31, 2016

Problem 1

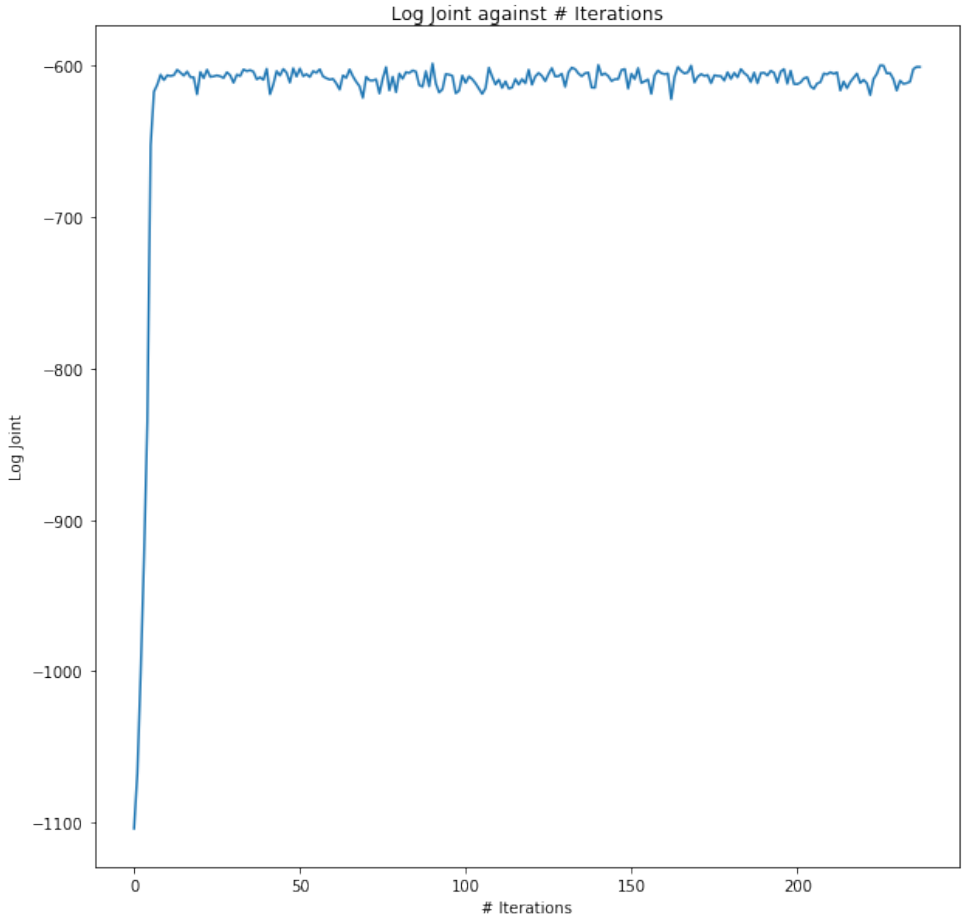
I implemented an uncollapsed Gibbs sampler for a mixture of two and three Gaussians. All the work reflected here are for mixtures of three Gaussians as I have made my code work with any number of mixtures. I simulated 150 data points with numpy's multivariate_normal function with means $\mu_1 = [2, 2],\ \mu_2 = [2, -2],\ \mu_3 = [-2, 0]$ and $\Sigma = [[1, 0], [0, 1]]$ as shown below:



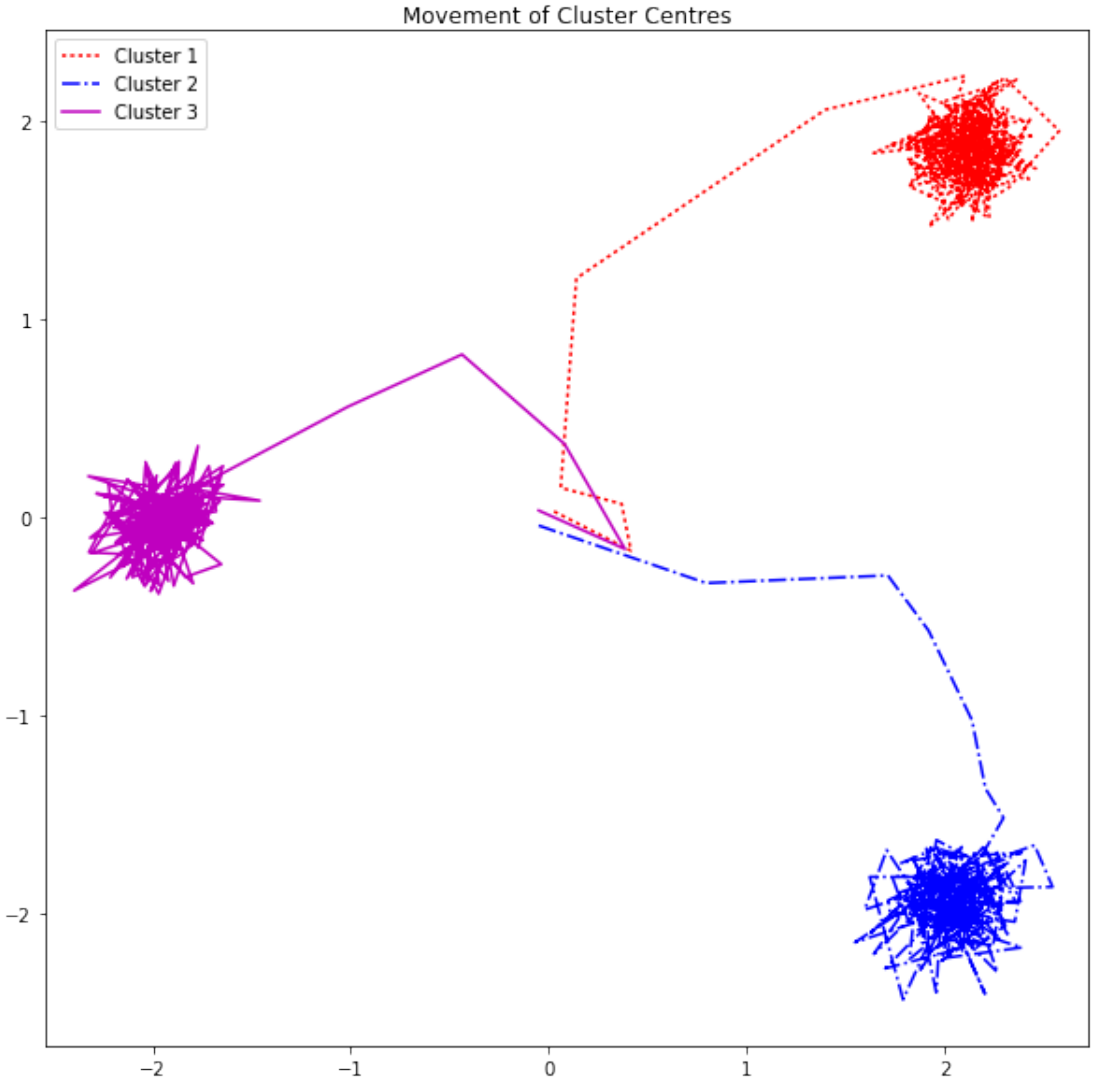
An interesting observation was that the choice of means is actually really important for obtaining convergence. When I was trying to my code to work for a mixture of two Gaussians, I set my means to be $\mu_1 = [2, 2], \mu_2 = [-2, 2]$, symmetrical around $x = y$. Within the first iteration, the sampler found means close to the ones I set but as converged to means close to the origin quite quickly and stayed close to it even after 800 iterations. In addition the log joint peaked then and never recovered. As the sampled assignments come from a random Bernoulli distribution, the randomness from the sampling process led to configurations where the data points are relatively well mixed. This led to overall means obtained, when averaged, to be close to the origin and from then on, it would be almost impossible to escape from such regions as the Bernoulli distribution would be parameterised by probability half. By setting my means to be $\mu_1 = [2, 2], \mu_2 = [2, -2]$, the sampler quickly converged to the set means.

My main objective for this assignment was learn how to most efficiently in terms of wall-clock time compute the normaliser and the probabilities of a point being from the different Gaussians in log-space since I would probably have to figure that out at some point when working with high-dimensional data. As I have been recently been playing with Python's lambda calculus, I used the map function to calculate all the probabilities in log-space, converted them into a numpy array, ran scipy's logsumexp to add them all up in log-space, then use numpy array's broadcasting properties to subtract the normaliser from the probabilities. By doing so, convergence to a log-joint difference of less 0.04 (it was set as that as with my random seed the sampler converged to 0.044 in 18 iterations) took around 35 seconds which is about 6.7 iterations per second. Next time, I hope to test out numpy.vectorize which numpy's equivalent to the map function.

Here, the log-joint is plotted against the number of iterations:



As a sanity check, I plotted the paths taken by the means as a function of iterations as seen in the following figure. It actually was really useful as I accidentally set the data points belonging cluster 3 as those belonging to cluster 2. The average means after 237 iterations (excluding the first 50 points which I discarded as burn-in) were $\mu_1 = [2.11733163, 1.85737745]$, $\mu_2 = [2.04162896, -1.93626671]$, $\mu_3 = [-1.95429518, -0.03094358]$ which are pretty close to those I set at the beginning.



Problem 2

Currently, I am working on extending variational RNNs to tackle applications of interest. An possible idea is to model traffic data in Manhattan. I looked at the datasets provided by the NYC Department of Transport (NYC DOT) for this brainstorming exercise.

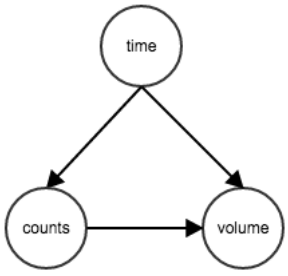
a

NYC DOT has several datasets[1]: street construction, street network changes and real-time traffic camera, real-time traffic speed and on top of these it has raw traffic data[2]. For this exercise, I looked exclusively at the raw traffic data which is quite sparse within Manhattan and would have to be supplemented by real-time traffic camera and traffic speed data when actual analysis is conducted. Locations of monitors are included the dataset too.

The variables listed in the raw traffic data are:

- Continuous Vehicle Classification
- Continuous Volume
- Short Count Vehicle Classification
- Short Count Volume
- Short Count Speed
- Average Weekday Vehicle Classification
- Average Weekday Volume
- Average Weekday Speed

I believe that time $\rightarrow$ vehicle counts, time $\rightarrow$ volume of traffic and vehicle counts $\rightarrow$ volume of traffic is a reasonable assumption with the graphical model shown below. The variables would be highly correlated as the number of vehicles on the road and the volume of vehicles per hour is generally dependent on the hour of the year and the volume of traffic is dependent on the number of vehicles on the road. From the volume of traffic, the counts of different sized vehicles and the length of the stretch of the road monitored, the speed of traffic can be derived (assuming vehicles of different sizes conform to some set of average lengths). After running the Bayes ball algorithm, I found no conditional independences.



[1] http://www.nyc.gov/html/dot/html/about/datafeeds.shtml
[2] https://www.dot.ny.gov/divisions/engineering/technical-services/highway-data-services/hdsb

b

A possible latent variable is the importance of the road with respect to the overall traffic infrastructure. This latent variable would determine counts and volume of traffic as roads of high importance would be key to traffic running smoothly and typically carry more traffic. Another latent variable that I am interesting in modelling is the effect of specific events on traffic i.e. poor weather, road works and accidents and compare how much they adversely affect traffic. The presence of such events at specific times might also be able to summarise the data close in time. I believe that within the data there might exist good partitions of New York where traffic in one part does not affect other parts and also some sequences of traffic patterns that lead to unusually smooth or congested traffic. If the latter is known, by perturbing the sequence slightly and observing the results, I might be able to understand the conditions determining good and bad traffic in specific areas and uncover possible ways to alleviate slow traffic or prevent them altogether.

c

1. Can realistic sequences of traffic patterns be simulated, especially those leading to unusually smooth and congestion traffic?
2. How streets can be ranked in terms of how important they are in ensuring smooth traffic within a certain area?
3. What are the effects of specific events at specific times on traffic locally and globally?