Probabilistic inference from Complex Data (David Dunson)

January 21, 2016

Abstract

1 Notes

Theoretical basis & applied focus Algorithm development

Outline:

- Motivation and background
- Primary interest: new methods for high dimensional data with structure.
- Project in art history **

There's a big emphasis on big data: most of the emphasis has been on penalized optimization methods, thinking for industrial problems, like if you have an enormous dataset in Google. There's been a lot less thinking on high dimensional data ins cience. In industry problems, rapidly obtaining a point estimate and large sample size problems is okay: but we can't do that in science; it doesn't really tell us about uncertainty. We might be propoagating false positives in the literature.

There's also been a huge focus on specific settings: identifying things in images, linear regression.. when we have high dimesional scientific data, it's difficult to apply these methods in that setting.

My focus: probability models where we can characteristically describe uncertainty. I'd like to have a probabilistic programming framework where peple come in, they can fit their models with minimal user specification. Hopefully we can figure out how to do inference in that model regardless of what the model is. If it involves big data, hopefully we can automatically adapt the algorithm to handle that data. We would like also to be able to handle arbitrarily complex probability models such as high dimensional surfaces, or differential equations - and we'd like to scale to huge data.

We want to accurately characterize uncertainty. Robustness of inference is also crucial.

1.1 Bayes approaches

I'm not religiously Bayesian; I'd like to do whatever works, but they offer an attractive general approach for modeling complex data. [Brief review on Bayesian inference: choosing a prior and likelihood, parameter might be infinite dimensional, it can be pretty complicated. Bayes rule also has a an annoying term in the denominator, integrating out the terms in the denominator. We usually can't deal with the integral in the denominator - it's difficult to approximate computationally or analytically.]

So what can we do? MCMC and posterior sampling algorithms remain teh gold standard. Variational methods may be the gold standard in industrial applications but not in scientific applications where you have to characterize uncertainty well since you don't have any guarantees about uncertainty, and that's problematic. But scaling MCMC to big and complex settings is challenging.

MCMC constructs a Markov chain with the stationary distribution that corresponds to the posterior distribution. [notes on board describing the process, transition kernel] We had a recent paper where we define a new class of Hamiltonian Monte Carlo algorithms which gives us an improved way of making these jumps which are very efficient. We run into a number of problems: the first is that the time per iteration increases with the number of parameters and unknowns. For example, the gaussian process - we might have a model involving an unknown regression function or surface. We'll see that usual MCMC algorithms might have cubic complexity at every iteration of the algorithm; n might be enormous. Also what happens is that mixing gets worse as dimension of data increases, where mixing rate describes efficient samples. People study scaling limits of these Markov chains and the scaling limit is pretty poor.

We'd really like to improve both of these problems. Also storing and basic processing on big data sets is problematic.

1.2 Solution

Cbayes notion: one of the problems we run into is the lack of robustness. Say we have a parameter in the model controlling the complexity. You can say, let's generate data from some small topological neighborhood of the model. If you allow the model complexity parameter to be unknown, as the model increases in dimension, the complexity will increase without bound - asymptotically we'll add more mixture components as we add more data.

CBayes: we start conditioning on the observed data being within a small neighborhood of data from the model. This leads to robustness and allows you to improve computational time by using subsets of the data.

Hybrid algorithms: run MCMC for a subset of the parameters and use a fast estimate for the others.

EPMCMC: embarassingly parallel MCMC. Run MCMC in parallel for different subsets of data and combine. Then we can get the best of both worlds.

Approximate MCMC: approximate expensive to evaluate transition kernels.

Embarassingly parallel MCMC:

LEt's say we have a huge dataset that's too large to do all the processing on one machine. We want to d something simple: we want to shard the data in advance, run MCMC in an embarassingly parallel manner, estimate the subset posterior, and combine the results to evaluate the joint posterior distribution.

How do we do it? A toy example: let's say we have a logistic regression model. Let's say we have a probability that the label is 1 given some featuers, and the parameters - we would put a prior on the parameters and we would have a joint posterior distribution on all of these θ. Instead of running MCMC for a long time break the data into subsets. If we can somehow estimate the center of the green contours (subset posteriors); maybe that gives us an accurate estimate of the actual posterior. Using the notion of mean in probability space - Wasserstein centers - we can get the center, which we can compute fast using a sparse linear program.

The game: we'd like to approximate this full data posterior distribution. This is completely analytically intractable; all we can do is draw samples and estimate posterior summaries for the parameters that we care about. Divide the data into k subsets of size m: they don't have to be all the same size; there can be more elaborate subsetting schemes. Throw the data in separate machines and do that in advance. Subset posterior density for the *jth* data subset - the product is just taken over the datapoints in the subset. We take the likelihood contribution adn we raise it to a large power γ which might be the order of the number of subsets. It's almost replicating this data to the full size of the dataset. We can choose γ to minimize approximation error; $O(k)$.

We need to define a metric defining distances between probability measures. We use the Wasserstein center (aka earth-mover's distance). Then the mean is the argmin over all possible probability measures in this space; it's the mean over all possible probability measures in this space. We're taking L2 distances. We come up with thsi notion of Wasserstein barycenter. It has this nice notion that we have a pile of sand representing the distributions - how much do we need to move the piles of sand to move to another distribution? We're going to define this Wasserstein Barycenter as follows:

$$\Pi_n(\cdot|Y^n) = argmin_{\Phi \in \mathcal{P}_2(\Theta)} \frac{1}{k} \sum_{j=1}^{k} W_2^2(\Pi, \{i_m^\gamma(\cdot|Y_{[j]})\}) \tag{1}$$

This turns into a simple optimization problem if we put in the empirical data, since we're based on samples. Plug in $\hat{\Pi}_m^\gamma$ .. minimizing wasserstein is solution to a discrete optimal transport problem. To illustrate this in the simple case let's just say that we have two measures which are atomic (just puts probability weights on different atoms), and we have some matrix of squared differences in adtoms between the distribution.

We can define an optimal transport polytope: set of doubly stochastic matrices with row sums **a** and column sums **b**. Objective is to find the best transport plan minimizing the distances. (Entropy smoothing has been used previously). We can avoid smoothing and use sparse LP solvers - negligible computation cost. **This is a discrete approximation to the Wasserstein barycenter,**

which doesn't change the location of the center but the weights. We have strong theory that in the large sample case this is the case. Our current theory, we have to rely on the sample size increasing slowly per subset. I would like to get rid of the need to increase subset sizes.

Theorem: under usual regularity conditions - concentration rate about subset posteriors. We have a stronger result in the one-dimensional case which leads to a simple algorithm. Describes how the subset posterior concentrates as we get more data in each subset: it's going to concentrate around θ_0, in this decreasing size ball.

Followup theorem: describes how the Wasserstein barycenter (approximation to joint posterior distributioN) - how that concentrates.

One thing that i was worried about for all of this is that it's very general, it's very fast; we have some theory results - they're more asymptotic than I would like them to be. Also statisticians are not used to linear programs and things like that: I'd like to have something more transparent. We got this PIE algorithm - it's fast posterior interval estimation. We usually report a joint posterior of many parameters that we can't viusalize; in practice we compute posterior summaries for one-dimensional functionals. We're predicting something, we get point/interval estimate. What if we just focus on one-dimensional functionals? Actually the Wasserstein barycenter for one-dimensional functionals has an explicit relationship between the CDFs. Then the quantiles of the WASP are just averages of quantiles of the subset posteriors. Then we can just break up the data on the subsets.. compute quantiles for any functional and just average them - trivial communication cost. reminiscent of Mike Jordan's bag of little bootstraps.

I was talking to Joel Tropp about this; he was saying that bag of little bootstraps fails computationally about 30% of the time. In the one-dimensional case we can come up with much better theory that we had in the other case. This is also trivial to implement in STAN, a probabilistic programming language, which allows powered likelihood so that you can scale up to larger dimensions.

Theory on PIE/1-d wasp

We show that this Wasserstein barycenter is highly accurate approximation to the exact posterior distribution. As the subset sample size m increases, the Wasserstein distance between them decreases at rate faster than parameteric rate $o_p(n^{-\frac{1}{2}})$.

The theorem allows the following: we may have $k = O(n^c)$ and $m = O(n^{1-c})$ for any $c \in (0, 1)$, so m can increase very slowly relative to k. So we can blow up the number of subsets (make it more distributed) while keeping the sample size similar.

In direct comparison of WASP and the true posterior: the biases, variances, quantiles only differ in high orders of the total sample size.

Results

We've tried this in practice for a lot of different models and data.

Question: how does your model handle models with variable complexity: none of the subsets have the same number of mixing components; or ..

The thing we haven't quite figured out is network data .. how do you subsample networks?

aMCMC: different way to speed up MCMC: replace expensive transition kernels with approximations. People do this all the time - approximate a conditional distribution in the Gibbs sampler with a Gaussian, orusing a subsample of data. These kinds of tricks can vastly speed up MCMC. Are there any theory guarantees for this approximation? What happens if we start substituting in aprpoximations - diverge etc? Our goal is to get theory guarnatees and use these to target design of algorithms. Two assumptions: define a nice transition kernel, mixes really well. We never have to run that one. Assumption 2: we have a transition kernel that's close to the transition kernel under the exact chain that we have never to run.

What type of results do we get? The approximate kernel is a computational game. Define the computational speed-up in one evaluation of the algorithm. The approximate algorithm might have 10-fold seed up. We'd like to exactly characterize the efficiency tradeoff between computational error and computational time for posterior functionals. We provide provably tight, finite sample bounds on L_2 error for general classes of approximate algorithms.

Bottom line: aMCMC estimators win for low computational budgets but have asymptotic bias.

Question: Do you also analyze how the approximate kernels affect the mixing time?

We don't analyze that so far but it's on the bucket list; but we think they might have better mixing times.

Polya-gamma data augmentation for logistic regression - assumptions hold with high probability for subsets $>$ minimal size. Application to particle physics dataset - 5 million samples, it's just a classification problem. We consider subset sizes ranging from 1000 to 4.5 million.. different loss functions.. if I want to estimate an interval, I need to choose a very large subset.

There's a huge literature on low-rank approximations for Gaussian processes - you need to choose a particular type of approximation - that I don't thinkw ork well at all.

General Conclusions

We need approximations adaptive to the current state of the train. MOre accurate approximations are needed farther from the high probaiblity region of the posterior. Approximations to conditionals of vector parameters are highlysensitive to the 2nd moment.