Advanced Machine Learning Subsidary Notes

Lecture 20: MCMC

Adam Prügel-Bennett

January 28, 2021

1 Keywords

- Monte Carlo methods, MCMC, Variational Methods

2 Main Points

2.1 Monte Carlo

- Except in a few cases Bayesian inference is hard
  - It is easy when we have a conjugate prior giving us a simple posterior
  - But, when we want to model complicated situations our likelihood function usually doesn't have a conjugate prior
  - This means the posterior is not expressible as a simple probability distribution
  - Furthermore it is usually impossible to compute the marginal likelihood or evidence

$$\mathbb{P}[\mathcal{D}] = \sum_{\boldsymbol{\Theta}} \mathbb{P}[\boldsymbol{\Theta}|\mathcal{D}]\ \mathbb{P}[\boldsymbol{\Theta}]$$

  - The sum or integral over variables $\boldsymbol{\Theta}$ is often too large to make this the feasible
  - You could try to obtain a histogram for the posterior, but if $\boldsymbol{\Theta}$ has many components the curse of dimensionality makes this difficult
- There is a simple answer which is to obtain samples from the posterior
  - We can use this to estimate expectations of any function, $g(\boldsymbol{\Theta})$, of $\boldsymbol{\Theta}$

$$\mathbb{E}_{\boldsymbol{\Theta}|\mathcal{D}}[g(\boldsymbol{\Theta})] \approx \frac{1}{n}\sum_{i=1}^{n} g(\boldsymbol{\Theta}_i)$$

    * $\boldsymbol{\Theta}_i$ are random deviates drawn from the posterior $\mathbb{P}[\boldsymbol{\Theta}_i|\mathcal{D}]$
    * (random deviates or random variates are values drawn from a probability distibution)
    * When the posterior distribution is well behaved (doesn't have a thick tail) then the error in the approximation fall off as $1/\sqrt{n}$
- The problem is you have to be able to sample random deviates from $P(\mathcal{D}|\boldsymbol{\Theta})\, f(\boldsymbol{\Theta})$
- For simple distributions there are two classic solutions to generating random deviates
  1. *Transformation Methods*

- If we can compute the cumulative probability function

$$F_X(x) = \int_{-\infty}^{x} f(t)\,\mathrm{d}t$$

and invert $F_X(x)$, then $X = F_X^{-1}(U)$ will be a random deviate with density f, where U is a uniform standard deviate between 0 and 1
- This only works for a few classic distributions

2. *Rejection Method*
   - In the rejection methods we find another distribution $g_Y(y)$ that we can sample from that satisfies

$$c\,g_Y(x) > f_X(x)$$

   - Then we generate a deviate $Y \sim g_Y$ and accept it with a probability $f_X(Y)/(c\,g_Y(Y))$
   - The expected rejection rate is $c - 1$
   - This is a more general method, but is efficient only if we have a good approximating function $g_Y(y)$ (good in the sense that c is not too large)
   - Particularly for multivariate (i.e. high dimensional) random variables it is often very hard to find a good approximating function

2.2 Markov-Chain Monte Carlo (MCMC)

- Fortunately there is another method for generating random deviates which is very general
- This uses a Markov-Chain where we update an initial state of our system
- If we choose the dynamics correctly eventually we will reach a state that is correctly distributed
- Suppose we have a set of states, $\mathcal{S}$, and we want to choose a random sample from a distribution $\boldsymbol{\pi} = (\pi_i | i \in \mathcal{S})$
  - π_i is the probability of being in state i—we get to choose this (for Bayesian inference we will want to choose the posterior probability)
- Let M_{ij} be the transition probability from state j to state i
  - $\sum_i M_{ij} = 1$ (if we start any state, j say, we have to end up in another state)
  - $M_{ij} \geq 0$ (it is a probability)
  - Matrices of non-negative elements whose rows sum to one are known as **stochastic matrices**
- If we choose M_{ij} so that

$$M_{ij}\,\pi_j = M_{ji}\,\pi_i$$

  - then after some time the probability of being in state i will be π_i (there are some mild conditions on M for this to be true)
  - This condition is known as **detailed balance**
  - Note that if we sum the detailed balance equation then

$$\sum_{j\in\mathcal{S}} M_{ij}\,\pi_j = \sum_{j\in\mathcal{S}} M_{ji}\,\pi_i = \pi_j$$

    * since $\sum_j M_{ji} = 1$
  - Writing this in matrix form

$$\mathsf{M}\boldsymbol{\pi} = \boldsymbol{\pi}$$

* That is, $\boldsymbol{\pi}$ is an eigenvalue of $\mathbf{M}$ with eigenvalue 1

– Because $\mathbf{M}$ is a stochastic matrix all eigenvalues will be less than or equal to 1

* To prove this let $\boldsymbol{v}$ be an eigenvector of $\mathbf{M}$ with eigenvalue λ

$$\sum_{j\in\mathcal{S}} M_{ij}\, v_j = \lambda\, v_i$$

taking the absolute value of both sides

$$\left|\sum_{j\in\mathcal{S}} M_{ij}\, v_j\right| = |\lambda\, v_i| = |\lambda|\,|v_i|$$

* But

$$\left|\sum_{j\in\mathcal{S}} M_{ij}\, v_j\right| \leq \sum_{j\in\mathcal{S}} |M_{ij}\, v_j| = \sum_{j\in\mathcal{S}} M_{ij}\,|v_j|$$

where we have used $M_{ij} \geq 0$ (and essentially $|a+b| \leq |a| + |b|$)

* Thus

$$\sum_{j\in\mathcal{S}} M_{ij}\,|v_j| \geq |\lambda|\,|v_i|$$

* Summing both sides with respect to i

$$\begin{aligned}
\sum_{i\in\mathcal{S}}\sum_{j\in\mathcal{S}} M_{ij}\,|v_j| &\geq |\lambda| \sum_{i\in\mathcal{S}} |v_i| \\
\sum_{j\in\mathcal{S}}\sum_{i\in\mathcal{S}} M_{ij}\,|v_j| &\geq |\lambda| \sum_{i\in\mathcal{S}} |v_i| \\
\sum_{j\in\mathcal{S}} |v_j| &\geq |\lambda| \sum_{i\in\mathcal{S}} |v_i|
\end{aligned}$$

where we used $\sum_{i\in\mathcal{S}} M_{ij} = 1$

* It follows that $\lambda \leq 1$
* Let $\boldsymbol{p}(t)$ be a vector whose elements $p_i(t)$ equal the probability of being in state i at time t
* If the set of eigenvectors span the whole space, then any starting vector $\boldsymbol{p}(0)$ can be written in terms of the eigenvectors

$$\boldsymbol{p}(0) = \sum_i c_i\, \boldsymbol{v}_i$$

(this is just a change of basis)

* The probability vector after one step is given by

$$\boldsymbol{p}(1) = \mathbf{M}\,\boldsymbol{p}(0) = \mathbf{M}\sum_i c_i\,\boldsymbol{v}_i = \sum_i c_i\,\mathbf{M}\,\boldsymbol{v}_i = \sum_i c_i\,\lambda_i\,\boldsymbol{v}_i$$

* After t steps

$$\boldsymbol{p}(t) = \mathbf{M}^t\,\boldsymbol{x} = \sum_i c_i\,\lambda_i^t\,\boldsymbol{v}_i$$

* If $|\lambda| < 1$ will then λ_i^t will shrink exponentially fast so that

$$\lim_{t\to\infty} \boldsymbol{p}(t) = \sum_{i:\lambda_i=0} c_i \boldsymbol{v}_i$$

* That is we converge onto the set of eigenvectors with eigenvalue 1
* But if detailed balance is conserved then $\boldsymbol{\pi}$ satisfies this

        * There are some conditioned to ensure that there is only one such eigenvector with eigenvalue 1
            · We have to ensure that we can get from any state to any other through some series of transitions
            · We have to prevent periodic behaviour (e.g. if our set of states where from 1 to n and we could only move from state i to state $i-1$ or $i+1$ then if we started on an odd state we would always be on an odd state after an even number of moves)
            · It is very easy to insure both conditions

- Now the problem is to choose the transition probabilities to satisfy detailed balance
- This is easy and can be done in different ways
- **Metropolis Algorithm**
    - A very easy way to achieve detailed balance is that starting from state i we choose a neighbouring state j
        * We have to make sure the probability of choosing state i when in state j is the same as the probability of choosing state j when starting in state i
    - We now move to state j if
        * $\pi_j \geq \pi_i$
        * or we move anyway with a probability π_i/π_j
    - It is a simple exercise to show that this satisfies detailed balance
- **Metropolis-Hastings Algorithm**
    - Sometimes it is difficult to insure that choosing state i form state j is the same as choosing state j from state i
        * As an example considering our states were non-negative integers, we move from state i to state $i-1$ and $i+1$ with probability $\frac{1}{2}$, but we have a problem at state 0 where we can't move to state $i-1$
    - We can modify the Metropolis algorithm. Suppose $p(i|j)$ is the probability of choosing state i starting is state j then let $r = p(j|i)\,\pi_i/(p(i|j)\,\pi_j)$
        * we accept the move if $r > 1$
        * or with a probability r
        * in practice we can just just draw a random number U uniformly between 0 and 1 and accept the move if $U < r$
        * this is the same as the Metropolis algorithm if $p(j|i) = p(i|j)$
- We can equally well apply this to (multi-dimensional) continuous variables, $\boldsymbol{\theta}$
    - If $\boldsymbol{\theta}$ is our current set of parameters, we choose a new set of parameter $\boldsymbol{\theta}'$ with probability $p(\boldsymbol{\theta}'|\boldsymbol{\theta})$ and then accept this proposal if $r = (p(\boldsymbol{\theta}|\boldsymbol{\theta}')\,\pi(\boldsymbol{\theta}'))/(p(\boldsymbol{\theta}'|\boldsymbol{\theta})\,\pi(\boldsymbol{\theta})) \geq 0$ or with a probability r
    - We now choose $p(\boldsymbol{\theta}'|\boldsymbol{\theta})$ so that with high probability $\boldsymbol{\theta}'$ is close to $\boldsymbol{\theta}$, this ensures that there is a reasonable high acceptance rate
- For Bayes calculations then $\pi(\boldsymbol{\theta})$ would be our posterior $f(\boldsymbol{\theta}|\mathcal{D})$
    - MCMC has the nice property that the update depends only on the ratio

$$\frac{\pi(\boldsymbol{\theta}')}{\pi(\boldsymbol{\theta})} = \frac{f(\boldsymbol{\theta}'|\mathcal{D})}{f(\boldsymbol{\theta}|\mathcal{D})} = \frac{f(\mathcal{D}|\boldsymbol{\theta}')\,f(\mathcal{D}|\boldsymbol{\theta}')}{f(\mathcal{D}|\boldsymbol{\theta})\,f(\mathcal{D}|\boldsymbol{\theta})}$$

  - That is, the normalising factor $f(\mathcal{D})$ that appears in Bayes' rule cancels
  - This is important because this is usually incomputable (although we can use MCMC to estimate this)
  - There is also often another advantage of this ratio: if we only change one variable at a time then frequently we can compute the ratio much more efficiently than the full likelihood and priors
  - When this happens then it pays to choose a single variable, θ_i, at a time, chose a neighbour θ_i' from a distribution $p(\theta_i'|\theta_i)$ and decide whether to accept this update
  - This one-variable-at-a-time procedure goes by the name of *Gibbs' sampling*

- **Convergence**
  - A problem with MCMC is that your initial parameter, $\boldsymbol{p}(0)$, is different from $\boldsymbol{\pi}$ so you have to wait some considerable time before your samples are from posterior distribution
  - As we saw earlier the convergence rate is determined by the second largest eigenvalue of the transition matrix $\mathbf{M}$ (we never explicit calculate $\mathbf{M}$ so we don't know what this eigenvalue is)
    * If it is very close to 1 then your convergence can be slow
  - Furthermore if you posterior distribution is broad you need lots of independent sample to accurately compute expectations
  - To obtain an independent sample we have to run our MCMC a long time
  - In practice you have to throw away some samples in a *burn-in* period
  - You can then average over all the remaining samples you have (including repetitions where you don't accept a move)
  - If you do this long enough then you should get samples that cover values of $\boldsymbol{\theta}$ with a high posterior probability
  - MCMC is slow because of this
  - There are a lot of clever tricks to speed up convergence and decorrelation times

- Often MCMC seems daunting because of the apparent difficulties
  - But don't be put off
  - You only need to use the tricks for extremely complicated problems
  - Usually a simple implementation works fine
  - Modern computers are so fast that convergence isn't usually a problem
    * Convergence can be a problem in pathological cases—these occur when using MCMC to model some physical systems—but for many Bayesian problems the Markov Chain is *rapidly mixing* meaning the convergence time is not excessively long

2.3 Variational Techniques

- This is a completely different approach to MCMC, but it tries to solve the same problem, but uses a different strategy
- This is again technically challenging and I don't expect you to memorise the derivation
- We saw in earlier lectures that we can cheat by seeking the parameters that maximise the posterior (the, so called, MAP solution)
- But this sacrifices all the probabilistic information

- In variational techniques we try to approximate the posterior with a simpler distribution
- That is, we approximate the prior $f(\boldsymbol{\theta}|\mathcal{D})$ with a simpler distribution $q(\boldsymbol{\theta}|\boldsymbol{w})$ where we get choose $\boldsymbol{w}$ to make q close to f
- Usually we choose q to be a factorisable distribution

$$q(\boldsymbol{\theta}|\boldsymbol{w}) = \prod_i q(\theta_i|w_i)$$

- These techniques were first developed in the physics community and so come with a strange language
- In the *variational approximations* we consider the **variational free energy**

$$\Phi(\boldsymbol{w}) = \int q(\boldsymbol{\theta}|\boldsymbol{w}) \, \log\left(\frac{q(\boldsymbol{\theta}|\boldsymbol{w})}{f(\boldsymbol{\theta}, \mathcal{D})}\right) \mathrm{d}\boldsymbol{\theta}$$

  - writing $f(\boldsymbol{\theta}, \mathcal{D}) = f(\boldsymbol{\theta}|\mathcal{D})\, f(\mathcal{D})$ we can rearrange the variational free energy as

$$\begin{aligned}\Phi(\boldsymbol{w}) &= \int q(\boldsymbol{\theta}|\boldsymbol{w}) \, \left(\log\left(\frac{q(\boldsymbol{\theta}|\boldsymbol{w})}{f(\boldsymbol{\theta}|\mathcal{D})}\right) - \log(f(\mathcal{D}))\right) \mathrm{d}\boldsymbol{\theta} \\ &= \mathrm{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{w})\big\|f(\boldsymbol{\theta}|\mathcal{D})\big) - \log(f(\mathcal{D}))\end{aligned}$$

  - where $\mathrm{KL}\big(q(\boldsymbol{\theta}|\boldsymbol{w})\big\|f(\boldsymbol{\theta}|\mathcal{D})\big)$ is the KL divergence that measures the "distance" between $q(\boldsymbol{\theta}|\boldsymbol{w})$ and $f(\boldsymbol{\theta}|\mathcal{D})$
  - we have previously shown that the KL-divergence is non-negative and equal to zero if the two distributions are the same
  - The term $\log(f(\mathcal{D}))$ is the *marginal likelihood* or *evidence* and does not depend on the parameters $\boldsymbol{w}$
  - If we choose $\boldsymbol{\theta}$ to minimise $\Phi(\boldsymbol{w})$ then we minimise the KL-divergence and force $q(\boldsymbol{\theta}|\boldsymbol{w})$ to approximate the posterior $f(\boldsymbol{\theta}|\mathcal{D})$
  - The variational free energy can also be written as

$$\begin{aligned}\Phi(\boldsymbol{w}) &= -\int q(\boldsymbol{\theta}|\boldsymbol{w}) \, \log(f(\boldsymbol{\theta}, \mathcal{D}))\, \mathrm{d}\boldsymbol{\theta} + \int q(\boldsymbol{\theta}|\boldsymbol{w}) \, \log(q(\boldsymbol{\theta}|\boldsymbol{w}))\, \mathrm{d}\boldsymbol{\theta} \\ &= -\big(L_q(\boldsymbol{w}) + H_q(\boldsymbol{w})\big)\end{aligned}$$

  - Minimising the variational free energy is equivalent to maximising $L_q(\boldsymbol{w}) + H_q(\boldsymbol{w})$
  - $L_q(\boldsymbol{w}) = \int q(\boldsymbol{\theta}|\boldsymbol{w}) \, \log(f(\boldsymbol{\theta}, \mathcal{D}))\, \mathrm{d}\boldsymbol{\theta}$ is the expected log-likelihood of the data, where we have marginalised with respect to our approximate posterior $q(\boldsymbol{\theta}|\boldsymbol{w})$
  - $H_q(\boldsymbol{w}) = -\int q(\boldsymbol{\theta}|\boldsymbol{w}) \, \log(q(\boldsymbol{\theta}|\boldsymbol{w}))\, \mathrm{d}\boldsymbol{\theta}$ is the entropy of our approximate posterior (this measure the uncertainty in $\boldsymbol{\theta}$)
  - We thus balance maximising the likelihood of the data with maximising the uncertainty in our approximate posterior
  - This means we fit the data, but guard against over fitting by allowing there to be a non-zero probability wherever it does not strongly contradict the data
  - In practice both $L_q(\boldsymbol{w})$ and $H_q(\boldsymbol{w})$ tend to be quite easy to compute
  - We are left with an optimisation problem for the parameters, $\boldsymbol{w}$, but this is usually quite quick to compute

3 Exercise

3.1 Mysterious Disease

- Recall in the lecture on probabilistic inference, we defined $Z(t)$ to be the number of people that catch a disease on day t
- We assumed the growth rate is given by

$$\mathbb{P}[Z(t+1)] = \mathrm{Poi}\Big(Z(t+1)\Big|\frac{r_0}{3}\left(Z(t)+Z(t-1)+Z(t-2)\right)\Big)$$

  - that is people of contagious for three days

- The observed number of new people with the disease, $X(t)$, on day t is

$$\mathbb{P}[X(t)=k] = \mathrm{Binom}(k|Z(t),p) = \binom{Z(t)}{k} p^k\,(1-p)^{Z(t)-k}$$

- Here is $X(t)$ for the first 30 days
  0,0,0,0,1,1,2,1,0,0,3,10,19,34,44,93,117,221,376,633,
  1021,1643,2701,4503,7414,12091,19999,33286,54612,90283

- Estimate p and r_0 from the data
- (If you prefer simulate your own data and estimate your parameters)

4 Answers

4.1 Mysterious Disease

- We need to decide on a prior
  - A reasonable prior for p is a Beta distribution perhaps with $a=b=1$ (this is a uniform prior)
  - For r_0 we could use a Gamma prior with $a=2$ and $b=1$, this has a mean of $r_0=2$ (it seems to be considerably greater than 1)—we need to include $r_0^{19}\mathrm{e}^{-20\,r_0}$ as our prior (the normalisation is irrelevant)
- Now we have to choose proposal distributions for the new values of p and r_0
  - You could use $p' = p + U(-0.01, 0.01)$ and $r_0' = r_0 + U(-0.1, 0.1)$ where $U(a,b)$ is a uniform random deviate between a and b
  - This could have a problem in that p and r_0 could take illegal values (i.e. we should have $0 \le r \le 1$ and $r_0 \ge 0$), but for my data this is unlikely to happen
  - We could choose $p' \sim \mathrm{Beta}(10\,p, 10-10\,p)$ and $r_0' \sim r_0\,\mathrm{Gamma}(5,5) = \mathrm{Gamma}(5, 5.r_0)$, this ensures that in expectation p' equals p and r_0' equals r_0 with small variations
    - Beware that there are two common definitions of Gamma distributions

$$\mathrm{Gamma}(z|,a,b) = \frac{b^a}{\Gamma(a)}\,z^{a-1}\,\mathrm{e}^{-b\,z}$$

$$\mathrm{Gamma}^*(z|,\alpha,\beta) = \frac{1}{\beta^\alpha\,\Gamma(\alpha)} z^{\alpha-1}\,\mathrm{e}^{-z/\beta}$$

    - You have to check which of these your libraries use
    - In this case we would have Metropolis-Hastings to ensure we get unbiased samples

- The variables $Z(t)$ are latent variables (they will vary for each iteration) we can ignore them, but we could also estimate their mean as they tell us the actual number of cases of people with the virus
  - My data was generated with $p = 0.1$ and $r_0 = 2.5$