Bayesian Statistics - Lecture 9

Lecture 9: HMC, Stan, Variational.

Mattias Villani

Department of Statistics
Stockholm University
and
Department of Computer and Information Science
Linköping University

Lecture overview

- **Hamiltonian Monte Carlo**
- **Stan**
- **Variational Bayes**

Hamiltonian Monte Carlo

- Assume $\theta = (\theta_1, ..., \theta_p)$. If p is large, then most of the mass of $p(\theta|\mathbf{y})$ is usually located on some subregion in $\mathbb{R}^p$ with complicated geometry.
- MH: hard to find good proposal distribution $q\left(\cdot|\theta^{(i-1)}\right)$.
- MH: have to use very small step sizes otherwise too many rejections.
- **Hamiltonian Monte Carlo** (**HMC**): **distant proposals** with **high acceptance** probabilities.
- HMC adds an auxiliary **momentum** parameter $\phi = (\phi_1, \ldots, \phi_p)$ and samples from $p(\theta, \phi|\mathbf{y}) = p(\theta|\mathbf{y})\, p(\phi)$.

- Physics: **Hamiltonian** system $H(\theta, \phi) = U(\theta) + K(\phi)$, where U is the **potential energy** and K is the **kinetic energy**.
- **Hamiltonian Dynamics**

$$\frac{d\theta_i}{dt} = \frac{\partial H}{\partial \phi_i} = \frac{\partial K}{\partial \phi_i},$$
$$\frac{d\phi_i}{dt} = -\frac{\partial H}{\partial \theta_i} = -\frac{\partial U}{\partial \theta_i}$$

- Think: Hockey puck sliding over a friction-less surface: illustration.
- Use $U(\theta) = -\log\left[p(\theta)\, p(\mathbf{y}|\theta)\right]$.
- Use $\phi \sim N(0, \mathbf{M})$ and $K(\phi) = -\log\left[p(\phi)\right] = \frac{1}{2}\phi^T \mathbf{M}^{-1}\phi + \text{const}$, where $\mathbf{M}$ is the mass matrix (often diagonal).
- If we could propose θ in continuous time (spoiler: we can't), the acceptance probability would be one.

- **Hamiltonian Dynamics**

$$\frac{d\theta_i}{dt} = \left[\mathbf{M}^{-1}\phi\right]_i,$$
$$\frac{d\phi_i}{dt} = \frac{\partial \log p\left(\theta|\mathbf{y}\right)}{\partial \theta_i}$$

which can be simulated using the **leapfrog algorithm**

$$\phi_i\left(t + \frac{\varepsilon}{2}\right) = \phi_i\left(t\right) - \frac{\varepsilon}{2}\frac{\partial \log p\left(\theta(t)|\mathbf{y}\right)}{\partial \theta_i},$$
$$\theta\left(t+\varepsilon\right) = \theta\left(t\right) + \varepsilon\mathbf{M}^{-1}\phi(t),$$
$$\phi_i\left(t+\varepsilon\right) = \phi_i\left(t + \frac{\varepsilon}{2}\right) - \frac{\varepsilon}{2}\frac{\partial \log p\left(\theta(t)|\mathbf{y}\right)}{\partial \theta_i},$$

where ε is the step size.

- Discretization $\Rightarrow$ acceptance probability drops with ε.

- Initialize $\theta^{(0)}$ and iterate for $i = 1, 2, \dots$

  1. Sample the starting **momentum** $\phi_s \sim N(0, \mathbf{M})$
  2. Simulate new values for (θ_p, ϕ_p) by iterating the **leapfrog algorithm** L times, starting in $\left(\theta^{(i-1)}, \phi_s\right)$.
  3. Compute the **acceptance probability**

$$\alpha = \min\left(1, \frac{p(\mathbf{y}|\theta_p)p(\theta_p)}{p(\mathbf{y}|\theta^{(i-1)})p(\theta^{(i-1)})} \frac{p(\phi_p)}{p(\phi_s)}\right)$$

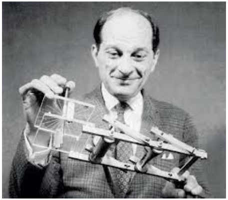
  4. With probability α set $\theta^{(i)} = \theta_p$ and $\theta^{(i)} = \theta^{(i-1)}$ otherwise.

- **Tuning parameters**: 1. **stepsize** ε, 2. **number of leapfrog** iterations L and 3. **mass matrix** M. **No U-turn**.

STAN

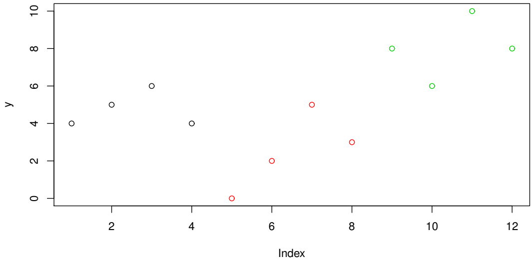
- **Stan** is a probabilistic programming language based on HMC.
- Allows for Bayesian inference in many models with automatic implementation of the MCMC sampler.
- Named after Stanislaw Ulam (1909-1984), co-inventor of the Monte Carlo algorithm.
- Written in C++ but can be run from R using the package `rstan`

Stan logo



Stanislaw Ulam

- Three plants were observed for four months, measuring the number of flowers



Stan Model 1: iid normal

$$y_i \overset{iid}{\sim} N\left(\mu, \sigma^2\right)$$

```
library(rstan)
y = c(4,5,6,4,0,2,5,3,8,6,10,8)
N = length(y)

StanModel = '
data {
  int<lower=0> N; // Number of observations
  int<lower=0> y[N]; // Number of flowers
}
parameters {
  real mu;
  real<lower=0> sigma2;
}
model {
  mu ~ normal(0,100); // Normal with mean 0, st.dev. 100
  sigma2 ~ scaled_inv_chi_square(1,2); // Scaled-inv-chi2 with nu 1, sigma 2
  for(i in 1:N)
    y[i] ~ normal(mu,sqrt(sigma2));
}'
```

$$y_{i,p} \sim N\left(\mu_p, \sigma_p^2\right), \quad \mu_p \sim N\left(\mu, \sigma^2\right)$$

```
StanModel = '
data {
  int<lower=0> N; // Number of observations
  int<lower=0> y[N]; // Number of flowers
  int<lower=0> P; // Number of plants
}
transformed data {
  int<lower=0> M; // Number of months
  M = N / P;
}
parameters {
  real mu;
  real<lower=0> sigma2;
  real mup[P];
  real sigmap2[P];
}
model {
  mu ~ normal(0,100); // Normal with mean 0, st.dev. 100
  sigma2 ~ scaled_inv_chi_square(1,2); // Scaled-inv-chi2 with nu 1, sigma 2
  for(p in 1:P){
    mup[p] ~ normal(mu,sqrt(sigma2));
    for(m in 1:M)
      y[M*(p-1)+m] ~ normal(mup[p],sqrt(sigmap2[p]));
  }
}'
```

Stan Model 3: multilevel Poisson

$$y_{i,p} \sim Poisson\left(\mu_p\right), \quad \mu_p \sim logN\left(\mu, \sigma^2\right)$$

```
StanModel = '
data {
  int<lower=0> N; // Number of observations
  int<lower=0> y[N]; // Number of flowers
  int<lower=0> P; // Number of plants
}
transformed data {
  int<lower=0> M; // Number of months
  M = N / P;
}
parameters {
  real mu;
  real<lower=0> sigma2;
  real mup[P];
}
model {
  mu ~ normal(0,100); // Normal with mean 0, st.dev. 100
  sigma2 ~ scaled_inv_chi_square(1,2); // Scaled-inv-chi2 with nu 1, sigma 2
  for(p in 1:P){
    mup[p] ~ lognormal(mu,sqrt(sigma2)); // Log-normal
    for(m in 1:M)
      y[M*(p-1)+m] ~ poisson(mup[p]); // Poisson
  }
}'
```

```
data = list(N=N, y=y, P=P)
burnin = 1000
niter = 2000
fit = stan(model_code=StanModel,data=data,
           warmup=burnin,iter=niter,chains=4)

# Print the fitted model
print(fit,digits_summary=3)

# Extract posterior samples
postDraws <- extract(fit)

# Do traceplots of the first chain
par(mfrow = c(1,1))
plot(postDraws$mu[1:(niter-burnin)],type="l",ylab="mu",main="Traceplot")

# Do automatic traceplots of all chains
traceplot(fit)

# Bivariate posterior plots
pairs(fit)
```

- Getting started with RStan
- RStan vignette
- Stan Modeling Language User's Guide and Reference Manual
- Stan Case Studies

- Let $\theta = (\theta_1, ..., \theta_p)$. Approximate the posterior $p(\theta|y)$ with a (simpler) distribution $q(\theta)$.
- Before: **Normal approximation** from optimation: $q(\theta) = N\left[\tilde{\theta}, J_{\mathbf{y}}^{-1}(\tilde{\theta})\right]$.
- **Mean field Variational Bayes** (**VB**)

$$q(\theta) = \prod_{i=1}^{p} q_i(\theta_i)$$

- Find the $q(\theta)$ that **minimizes the Kullback-Leibler distance** between the true posterior p and the approximation q:

$$KL(q, p) = \int q(\theta) \ln \frac{q(\theta)}{p(\theta|y)} d\theta = E_q \left[\ln \frac{q(\theta)}{p(\theta|y)} \right].$$

- **Mean field VB** is based on factorized approximation:

$$q(\theta) = \prod_{i=1}^{p} q_i(\theta_i)$$

- **No specific functional forms** are assumed for the $q_i(\theta)$.
- **Optimal densities** can be shown to satisfy:

$$q_i(\theta) \propto \exp\left(E_{-\theta_i} \ln p(\mathbf{y}, \theta)\right)$$

where $E_{-\theta_i}(\cdot)$ is the expectation with respect to $\prod_{i \neq j} q_j(\theta_j)$.

- **Structured mean field approximation**. Group subset of parameters in tractable blocks. Similar to Gibbs sampling.

Mean field approximation - algorithm

- Initialize: $q_2^*(\theta_2), \ldots, q_M^*(\theta_p)$
- Repeat until convergence:
  - $q_1^*(\theta_1) \leftarrow \frac{\exp\left[E_{-\theta_1} \ln p(\mathbf{y},\theta)\right]}{\int \exp\left[E_{-\theta_1} \ln p(\mathbf{y},\theta)\right] d\theta_1}$
  - $\vdots$
  - $q_p^*(\theta_p) \leftarrow \frac{\exp\left[E_{-\theta_p} \ln p(\mathbf{y},\theta)\right]}{\int \exp\left[E_{-\theta_p} \ln p(\mathbf{y},\theta)\right] d\theta_p}$
- Note: no assumptions about parametric form of the $q_i(\theta)$.
- Optimal $q_i(\theta)$ often **turn out** to be parametric (normal etc).
- Just update hyperparameters in the optimal densities.

- **Model**: $X_i|\theta, \sigma^2 \overset{iid}{\sim} N(\theta, \sigma^2)$.
- **Prior**: $\theta \sim N(\mu_0, \tau_0^2)$ **independent** of $\sigma^2 \sim Inv - \chi^2(\nu_0, \sigma_0^2)$.
- **Mean-field approximation**: $q(\theta, \sigma^2) = q_\theta(\theta) \cdot q_{\sigma^2}(\sigma^2)$.
- Optimal densities

$$q_\theta^*(\theta) \propto \exp\left[E_{q(\sigma^2)} \ln p(\theta, \sigma^2, \mathbf{x})\right]$$
$$q_{\sigma^2}^*(\sigma^2) \propto \exp\left[E_{q(\theta)} \ln p(\theta, \sigma^2, \mathbf{x})\right]$$

- **Variational density for σ^2**

$$\sigma^2 \sim Inv - \chi^2\left(\tilde{\nu}_n, \tilde{\sigma}_n^2\right)$$

where $\tilde{\nu}_n = \nu_0 + n$ and $\tilde{\sigma}_n = \frac{\nu_0\sigma_0^2 + \sum_{i=1}^n (x_i - \tilde{\mu}_n)^2 + n \cdot \tilde{\tau}_n^2}{\nu_0 + n}$

- **Variational density for θ**

$$\theta \sim N\left(\tilde{\mu}_n, \tilde{\tau}_n^2\right)$$

where

$$\tilde{\tau}_n^2 = \frac{1}{\frac{n}{\tilde{\sigma}_n^2} + \frac{1}{\tau_0^2}}$$

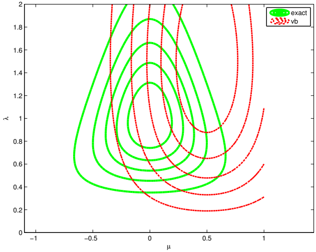
$$\tilde{\mu}_n = \tilde{w}\bar{x} + (1 - \tilde{w})\mu_0,$$

where

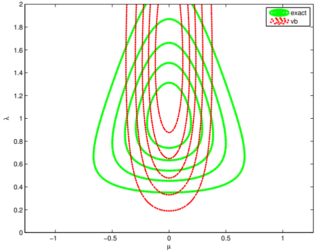
$$\tilde{w} = \frac{\frac{n}{\tilde{\sigma}_n^2}}{\frac{n}{\tilde{\sigma}_n^2} + \frac{1}{\tau_0^2}}$$

Normal example from Murphy ($\lambda = 1/\sigma^2$)

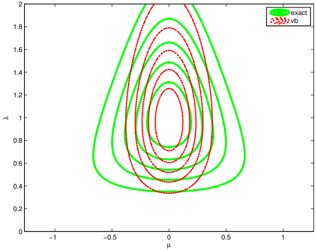
Initial values



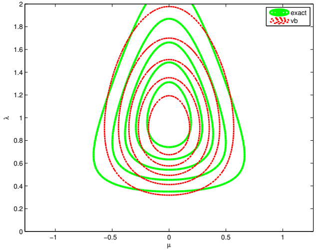
After updating q_μ



After updating q_{σ^2}



At convergence



- **Model**:

$$\Pr(y_i = 1|\mathbf{x}_i) = \Phi(\mathbf{x}_i^T \beta)$$

- **Prior**: $\beta \sim N(0, \Sigma_\beta)$. For example: $\Sigma_\beta = \tau^2 I$.
- **Latent variable formulation** with $u = (u_1, ..., u_n)'$

$$\mathbf{u}|\beta \sim N(\mathbf{X}\beta, 1)$$

and

$$y_i = \begin{cases} 0 & \text{if } u_i \leq 0 \\ 1 & \text{if } u_i > 0 \end{cases}$$

- Factorized **variational approximation**

$$q(\mathbf{u}, \beta) = q_{\mathbf{u}}(\mathbf{u}) q_\beta(\beta)$$

- **VB posterior**

$$\beta \sim N\left(\tilde{\mu}_\beta, \left(\mathbf{X}^T\mathbf{X} + \Sigma_\beta^{-1}\right)^{-1}\right)$$

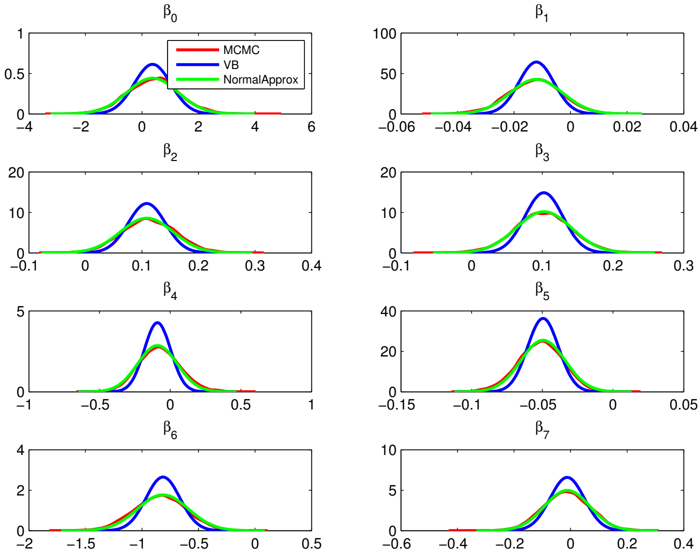
where

$$\tilde{\mu}_\beta = \left(\mathbf{X}^T\mathbf{X} + \Sigma_\beta^{-1}\right)^{-1}\mathbf{X}^T\tilde{\mu}_\mathbf{u}$$

and

$$\tilde{\mu}_\mathbf{u} = \mathbf{X}\tilde{\mu}_\beta + \frac{\phi\left(\mathbf{X}\tilde{\mu}_\beta\right)}{\Phi\left(\mathbf{X}\tilde{\mu}_\beta\right)^\mathbf{y}\left[\Phi\left(\mathbf{X}\tilde{\mu}_\beta\right) - \mathbf{1}_n\right]^{\mathbf{1}_n - \mathbf{y}}}.$$

Probit example (n=200 observations)



Probit example