Metropolis-Hastings 算法, HMC 算法与 SMC 算法

王璐

有些 Bayesian 模型不存在 conjugate priors, 甚至参数的 full conditional distributions 也不是常见的分布或很难进行抽样。这种情况下，无法使用 Gibbs sampler 估计参数的后验分布。本章介绍一种更通用的 MCMC 方法 — Metropolis-Hastings 算法，它几乎适用估计任何 priors 下的 Bayesian 模型。我们首先以一个 Bayesian Poisson 回归模型为例引出该方法。

1 A Bayesian Poisson Regression Model

一项针对麻雀的研究记录了 52 只雌雀在一个夏天的繁殖数据，图1用 boxplot 展示了这些雌雀繁殖的后代数与它们年龄的关系。从图1可以看到，2 岁的雌雀繁殖后代数的中位数 (median) 最高。

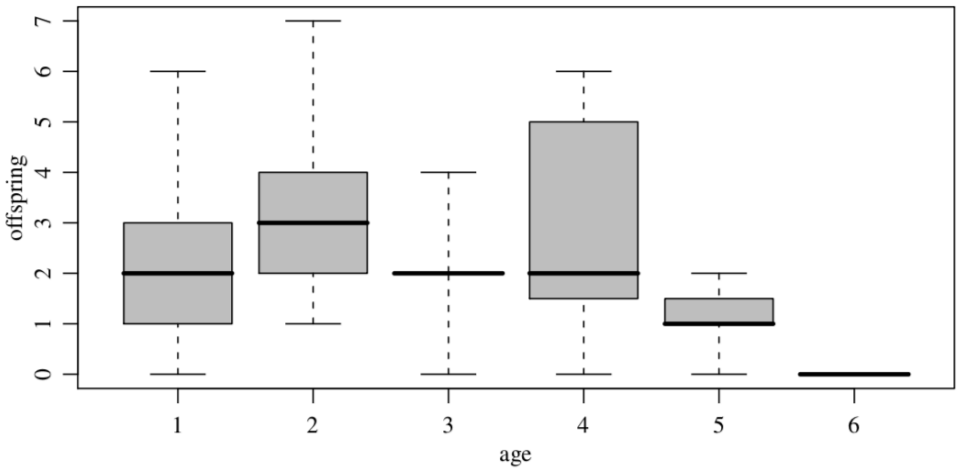


Figure 1: 雌雀繁殖的后代数与年龄的 boxplot. Picture sourse: Hoff (2009)

我们希望用一个概率模型拟合该数据以预测雌雀各年龄繁殖后代数的期望。响应变量 (response) y_i 对应雌雀 i 繁殖的后代数，是一个非负整数 $y_i \in \{0, 1, 2, \dots\}$；解释变量 x_i 为雌雀 i

的年龄。考虑采用如下的 Poisson 模型:

$$y_i \mid x_i \sim Po(\theta(x_i)), \quad i = 1, \ldots, n. \tag{1}$$

根据图1，可以假设 y_i 的期望 $\theta(x_i)$ 是 x_i 的二次函数，即 $\theta(x_i) = \beta_1 + \beta_2 x_i + \beta_3 x_i^2$. 但是该模型有一个问题：估计的系数 $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3)$ 可能使 $\theta(x_i) < 0$. 一种解决的方法是假设 $\log \theta(x_i)$ 是 x_i 的二次函数：

$$\log \theta(x_i) = \beta_1 + \beta_2 x_i + \beta_3 x_i^2 \tag{2}$$

此时 y_i 的期望 $\theta(x_i) = \exp(\beta_1 + \beta_2 x_i + \beta_3 x_i^2) > 0, \forall \boldsymbol{\beta}$.

为了更好地描述估计的系数 $\hat{\boldsymbol{\beta}} = (\hat{\beta}_1, \hat{\beta}_2, \hat{\beta}_3)$ 的不确定性 (如方差、置信区间等), 考虑采用 Bayesian 分析。为上述 Poisson 回归模型(1)-(2)的参数 $\boldsymbol{\beta}$ 设定如下的多元正态 prior:

$$\boldsymbol{\beta} \sim N_3(\mathbf{0}, 100 I_3) \tag{3}$$

在 prior (3)下，$\boldsymbol{\beta}$ 的后验分布不是多元正态分布，其各分量的 full conditional distribution 也不是常见的容易抽样的分布，此时无法使用 Gibbs sampler, 但 Metropolis 方法仍然能通过构建 Markov chain 获得 $\boldsymbol{\beta}$ 后验分布的样本。

2 Metropolis 算法

对一般的 Bayesian 模型，用 $p(\boldsymbol{y} \mid \boldsymbol{\theta})$ 表示观察值的似然函数，参数 $\boldsymbol{\theta} \in \mathbb{R}^p$ 的 prior 为 $p(\boldsymbol{\theta})$. $\boldsymbol{\theta}$ 的后验分布为

$$p(\boldsymbol{\theta} \mid \boldsymbol{y}) \propto p(\boldsymbol{y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta})$$

但该分布的 normalizing constant $\int p(\boldsymbol{y} \mid \boldsymbol{\theta}) p(\boldsymbol{\theta}) d\boldsymbol{\theta}$ 通常很难计算，也很难直接从该后验分布抽样。

Metropolis 算法通过持续地在参数空间随机 “游走” 寻找目标后验分布 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ 概率密度较高的区域。假设在当前时刻 Markov chain 得到的样本为 $\boldsymbol{\theta}^{(t)}$, 在 $\boldsymbol{\theta}^{(t)}$ 附近随机产生一点，如果该点对应的 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ 的值高于 $p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})$, 则让 Markov chain 沿该点的方向继续移动；反之以一定概率决定是否沿较低概率密度的方向移动，这点对于有效地探索多峰值的后验分布很重要。

运行 Metropolis 算法需要先选取一个对称的 proposal 分布 $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$, $g(\cdot \mid \cdot)$ 满足 $g(\boldsymbol{\theta}_a \mid \boldsymbol{\theta}_b) = g(\boldsymbol{\theta}_b \mid \boldsymbol{\theta}_a)$. 比如常用的 proposal 分布为:

- $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) \sim \boldsymbol{U}\left(\boldsymbol{\theta}^{(t)} - \boldsymbol{\delta}, \boldsymbol{\theta}^{(t)} + \boldsymbol{\delta}\right)$
- $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)}) \sim N_p(\boldsymbol{\theta}^{(t)}, \mathrm{diag}(\boldsymbol{\delta}))$

后面我们会讨论如何选取 $\boldsymbol{\delta}$ 使算法运行更有效率。

选定 proposal 分布 $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ 后，Metropolis 算法如下产生样本 $\boldsymbol{\theta}^{(t+1)}$:

1. 抽取 $\boldsymbol{\theta}^{\star} \sim g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$

2. 计算接受比率 (acceptance ratio)

$$r_t = \frac{p(\boldsymbol{\theta}^{\star} \mid \boldsymbol{y})}{p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})} = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta}^{\star}) p(\boldsymbol{\theta}^{\star})}{p(\boldsymbol{y} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)})}.$$

3. 令

$$\boldsymbol{\theta}^{(t+1)} = \begin{cases} \boldsymbol{\theta}^{\star} & \text{概率 } \min(r_t, 1) \\ \boldsymbol{\theta}^{(t)} & \text{概率 } 1 - \min(r_t, 1) \end{cases}$$

2.1 Metropolis 算法的收敛性

根据 Markov chain 理论，一条遍历的 Markov chain 会收敛到唯一的 stationary distribution. 此时对于 transition density $p(\boldsymbol{x} \mid \boldsymbol{x}')$, 如果能找到分布 $\pi(\boldsymbol{x})$ 满足

$$\pi(\boldsymbol{x}) = \int p(\boldsymbol{x} \mid \boldsymbol{x}') \pi(\boldsymbol{x}') d\boldsymbol{x}', \quad \forall \boldsymbol{x}, \boldsymbol{x}' \tag{4}$$

则 $\pi(\boldsymbol{x})$ 就是 Markov chain 收敛到的 stationary distribution.

(4)成立的一个充分条件如下：

$$p(\boldsymbol{x}' \mid \boldsymbol{x}) \pi(\boldsymbol{x}) = p(\boldsymbol{x} \mid \boldsymbol{x}') \pi(\boldsymbol{x}') \quad \forall \boldsymbol{x}, \boldsymbol{x}' \tag{5}$$

对(5)两边关于 $\boldsymbol{x}'$ 积分可得(4). 称(5)为 **detail balance** 条件。因此对一条遍历的 Markov chain, 找到满足 detail balance (5)的分布 $\pi(\boldsymbol{x})$ 就找到了 Markov chain 的 stationary distribution.

接下来我们证明 Metropolis 算法产生的 Markov chain 的 stationary distribution 为参数的后验分布 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$.

- 如果 $\boldsymbol{\theta}^{(t+1)} \neq \boldsymbol{\theta}^{(t)}$, 说明接受了候选样本，对应的 transition density 为

$$\begin{aligned} p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) &= g(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) P(\boldsymbol{\theta}^{(t+1)} \text{ is accepted}) \\ &= g(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) \min\left\{ \frac{p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{y})}{p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})}, 1 \right\} \end{aligned}$$

此时

$$p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y}) = g(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) \min\{p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{y}), p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})\} \tag{6}$$

由于 proposal density $g(\cdot \mid \cdot)$ 是对称的，因此(6)的右端关于 $(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)})$ 对称，所以

$$p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t+1)})p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{y}) = p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)})p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})$$

即 detail balance 条件对后验分布 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ 成立。

- 如果 $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)}$，不论 transition density 具有何种形式，detail balance 条件对后验分布总是成立的，因为 $p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)})p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y}) = p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)})p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})$.

因此如果 Metropolis 方法生成的 Markov chain 是遍历的，由于 detail balance 条件对后验分布成立，该 Markov chain 会收敛到后验分布。

Remarks

1. 如果参数 $\boldsymbol{\theta}$ 是一个 p 维连续向量，将 proposal distribution $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ 选为 $N_p(\boldsymbol{\theta}^{(t)}, \mathrm{diag}(\boldsymbol{\delta}))$ 可以得到遍历的 Markov chain.

2. 在 Metropolis 算法中，接受比率并不是越高越好。如果 proposal distribution $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ 中选取的 $\|\boldsymbol{\delta}\|$ 很小，每一步迭代产生的候选样本 $\boldsymbol{\theta}^{\star}$ 会很接近当前样本 $\boldsymbol{\theta}^{(t)}$，这导致 $p(\boldsymbol{\theta}^{\star} \mid \boldsymbol{y}) \approx p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})$，因此候选样本很容易被接受。但会导致 Markov chain 在参数空间中移动地非常缓慢，需要很长时间才能收敛到 stationary distribution, 如图2的左图所示。此时 Markov chain 上样本之间的相关性较高，这使得样本均值对后验期望的近似精度下降（回顾 effective sample size 的定义）。在 Gibbs sampling 中，我们无法直接控制 Markov chain 的相关性，但在 Metropolis 算法中，可以通过调节 $\|\boldsymbol{\delta}\|$ 来调整 Markov chain 的相关性。

3. 如果在 proposal distribution $g(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ 中令 $\|\boldsymbol{\delta}\|$ 很大，每步产生的候选样本 $\boldsymbol{\theta}^{\star}$ 虽然能更自由地探索参数空间，其对应的 $p(\boldsymbol{\theta}^{\star} \mid \boldsymbol{y})$ 很可能非常小，因而很容易被拒绝。这会造成 Markov chain 在一段时间 “困在” 局部某点处不动，导致样本间相关性较高，如图2的右图所示。

4. 实践中经常采用以下做法为 proposal distribution 选取合适的 $\boldsymbol{\delta}$: 在不同 $\boldsymbol{\delta}$ 取值下，先运行 Metropolis 算法产生一些较短的 Markov chains, 选取 $\boldsymbol{\delta}$ 使得 Markov chain 的接受比率大致在 20% 到 50% 之间 (Hoff, 2009)；确定一个合理的 $\boldsymbol{\delta}$ 后，再运行 Metropolis 算法产生较长的 Markov chain 做统计推断。

3 Bayesian Poisson 回归模型的 Metropolis 算法

本节介绍如何使用 Metropolis 算法估计 Section 1中的 Bayesian Poisson 回归模型(1)-(3). 在每步迭代中，我们整体对参数 $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3)$ 进行抽样，因为 block sampling 可以减少样本间

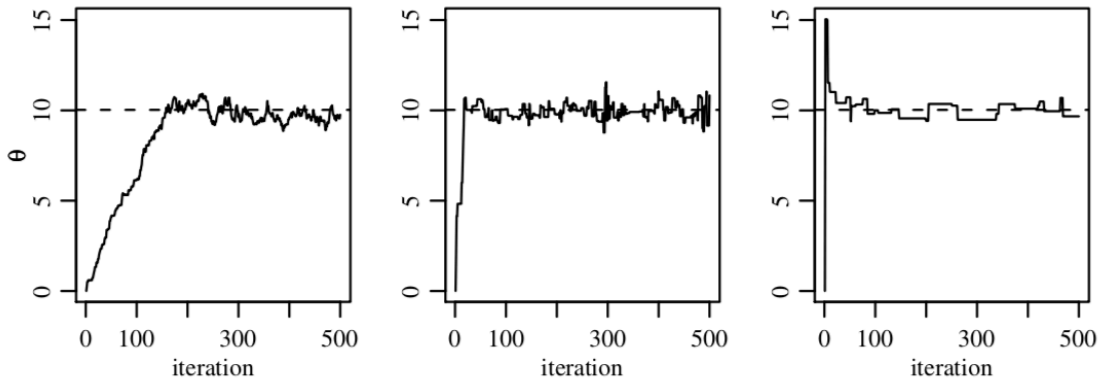


Figure 2: 在 proposal distribution $N(\theta^{(t)}, \delta^2)$ 中选取不同的 δ 得到的 Markov chains. 从左到右分别对应 $\delta^2 = 1/32,\ 2,\ 64$. Markov chain 的总体接受比率从左到右分别为 87%, 35%, 5%. Picture sourse: Hoff (2009)

的相关性。从 proposal distribution $g(\boldsymbol{\beta} \mid \boldsymbol{\beta}^{(t)})$ 产生一个候选样本 $\boldsymbol{\beta}^\star$，令 $\boldsymbol{x}_i = (1, x_i, x_i^2)^\top$，矩阵 $X = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^\top$, 此时 Metropolis 算法的接受比率为:

$$\begin{aligned} r &= \frac{p(\boldsymbol{\beta}^\star \mid X, \boldsymbol{y})}{p(\boldsymbol{\beta}^{(t)} \mid X, \boldsymbol{y})} \\ &= \frac{N_3(\boldsymbol{\beta}^\star \mid \mathbf{0}, 100 I_3) \prod_{i=1}^n \mathrm{Po}\left(y_i \mid \exp(\boldsymbol{x}_i^\top \boldsymbol{\beta}^\star)\right)}{N_3(\boldsymbol{\beta}^{(t)} \mid \mathbf{0}, 100 I_3) \prod_{i=1}^n \mathrm{Po}\left(y_i \mid \exp(\boldsymbol{x}_i^\top \boldsymbol{\beta}^{(t)})\right)}. \end{aligned}$$

在本例中，我们选取的 proposal distribution 为 $N(\boldsymbol{\beta}^{(t)}, \hat{\sigma}^2 (X^\top X)^{-1})$, 其中 $\hat{\sigma}^2$ 是 $\{\log(y_1 + 1/2), \ldots, \log(y_n + 1/2)\}$ 的样本方差。因为对于线性回归模型，$\boldsymbol{\beta}$ 的后验方差接近 $\sigma^2 (X^\top X)^{-1}$, 其中 σ^2 是 Y 的方差。如果该分布产生的 Markov chain 的接受比率过高或过低，总可以相应调整 proposal distribution 中的方差。实现本例 Metropolis 算法的 R code 如下。

```
n <- length(y)
p <- dim(X)[2]

pmn.beta <- rep (0 ,p) #prior expectation
psd.beta <- rep (10 ,p) #prior sd

var.prop <- var ( log (y+1/2) ) * solve ( t(X) %*% X ) #proposal var

S <- 10000 #length of Markov chain
beta <- rep(0,p) #initial value
```

```
acs <- 0 #number of acceptances
BETA <- matrix(0, nrow=S, ncol=p)

# Metropolis algorithm
set.seed (1)
for(s in 1:S){
  beta.p <- t( rmvnorm(1, beta, var.prop) )

  log_ar <- sum( dpois(y, exp(X %*% beta.p), log=T) ) -
            sum( dpois(y, exp(X %*% beta), log=T) ) +
            sum( dnorm(beta.p, pmn.beta, psd.beta, log=T) ) -
            sum( dnorm(beta, pmn.beta, psd.beta, log =T) )

  if( log(runif(1)) < log_ar ){
    beta <- beta.p
    acs <- acs+1
  }

  BETA[s,] <- beta
}
```

代入雌雀的数据，上述 Metropolis 算法得到的 Markov chain 的接受比率为 43%. 图3的左图展示了 β_3 对应的 Markov chain 的 traceplot. 可以看到该 Markov chain 很快从初始值 0 移动到 posterior mode 附近。图3中间的图展示了该 Markov chain 的 autocorrelation function (ACF), 可以看到相邻样本的自相关系数很高。如果我们从该 Markov chain 上每 10 步取一个样本组成一条 *thinned* Markov chain，图3展示了新的 Markov chain 的 ACF, 可以看到此时样本间的相关性很小。经过 thinning 的 Markov chain 只保留了原 Markov chain 上 10000 个样本中的 1000 个，但这 1000 个样本接近相互独立，它们的 ESS 为 726. 这些样本对于估计本例的后验分布是足够的，图4的左图用虚线展示了由离散网格法估计的 β_3 的边际后验分布的 pdf，这与从上述 thinned Markov chain 上估计的边际分布几乎完全相同。

最后，图4的右图展示了雌雀在年龄 x 下期望繁殖的后代数 $E(Y \mid x)$ 的 posterior median 及 95% credible interval, 这些结果反映出该雀种产生的后代数与雌雀年龄的二次函数的关系。

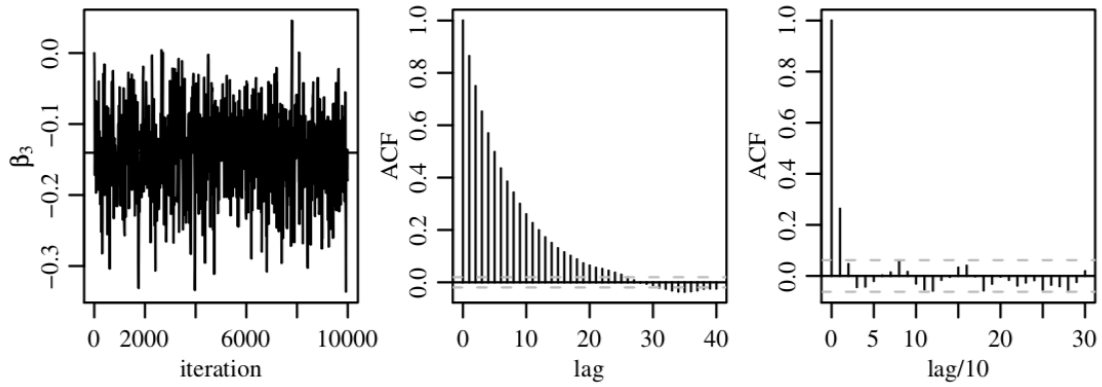


Figure 3: β_3 的 Markov chain 及其 autocorrelation functions. Picture sourse: Hoff (2009)

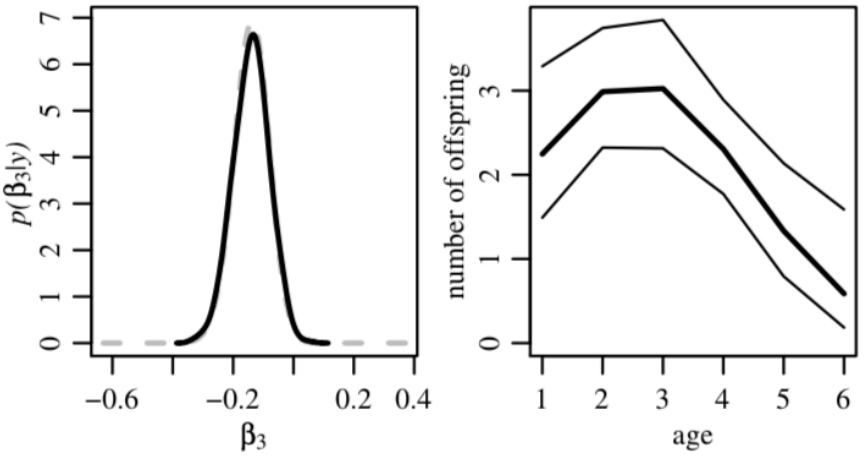


Figure 4: 左图：实线代表由 β_3 的 thinned Markov chain 估计的边际 pdf，虚线是由数值方法估计的 β_3 的边际 posterior density; 右图：从上到下的三条线分别对应 $\exp(\boldsymbol{x}^\top\boldsymbol{\beta})$ 的 2.5%, 50% 和 97.5% posterior quantiles. Picture sourse: Hoff (2009)

4 Metropolis-Hastings 算法

前面介绍的 Gibbs sampler 和 Metropolis 算法是两种使用 Markov chain 近似目标分布的方法。事实上，它们是更一般的 Metropolis-Hastings (M-H) 算法的两个特例。M-H 算法与 Metropolis 算法很相似，它也需要在每步迭代中从 proposal distribution 产生一个候选样本，然后按照一定规则接受或拒绝该样本，但是 M-H 算法允许任何形式的 proposal distribution, 不一定是对称的条件分布。

假设我们的目标分布是参数 $\boldsymbol{\theta}$ 的后验分布 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$. 选取 proposal distribution $\tilde{g}(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ 后，

M-H 算法如下产生样本 $\boldsymbol{\theta}^{(t+1)}$:

1. 抽取 $\boldsymbol{\theta}^\star \sim \tilde{g}(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$

2. 计算接受比率

$$r_t = \frac{p(\boldsymbol{\theta}^\star \mid \boldsymbol{y})}{p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})} \cdot \frac{\tilde{g}(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^\star)}{\tilde{g}(\boldsymbol{\theta}^\star \mid \boldsymbol{\theta}^{(t)})} = \frac{p(\boldsymbol{y} \mid \boldsymbol{\theta}^\star)p(\boldsymbol{\theta}^\star)\tilde{g}(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^\star)}{p(\boldsymbol{y} \mid \boldsymbol{\theta}^{(t)})p(\boldsymbol{\theta}^{(t)})\tilde{g}(\boldsymbol{\theta}^\star \mid \boldsymbol{\theta}^{(t)})}.$$

3. 令

$$\boldsymbol{\theta}^{(t+1)} = \begin{cases} \boldsymbol{\theta}^\star & \text{概率 } \min(r_t, 1) \\ \boldsymbol{\theta}^{(t)} & \text{概率 } 1 - \min(r_t, 1) \end{cases}$$

M-H 算法的适用范围更广，因为对称的 proposal distribution 有时并不合理，比如对于方差参数，对称的 proposal distribution 可能无法保证产生的样本为正。与 Metropolis 算法类似，我们可以用 detail balance 条件证明 M-H 算法产生的 Markov chain 收敛到参数的后验分布 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$。

- 如果 $\boldsymbol{\theta}^{(t+1)} \neq \boldsymbol{\theta}^{(t)}$，从 $\boldsymbol{\theta}^{(t)}$ 到 $\boldsymbol{\theta}^{(t+1)}$ 的 transition density 为

$$\begin{aligned} p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) &= \tilde{g}(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) P(\boldsymbol{\theta}^{(t+1)} \text{ is accepted}) \\ &= \tilde{g}(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) \min\left\{ \frac{p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{y})}{p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})} \cdot \frac{\tilde{g}(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t+1)})}{\tilde{g}(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)})}, 1 \right\} \end{aligned}$$

  此时

$$p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y}) = \min\{p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{y})\tilde{g}(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t+1)}),\, p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})\tilde{g}(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)})\} \quad (7)$$

  注意到(7)的右端关于 $(\boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)})$ 对称，所以

$$p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t+1)}) p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{y}) = p(\boldsymbol{\theta}^{(t+1)} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y}), \quad \forall \boldsymbol{\theta}^{(t)}, \boldsymbol{\theta}^{(t+1)}$$

  即 detail balance 条件关于后验分布 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ 成立。

- 如果 $\boldsymbol{\theta}^{(t+1)} = \boldsymbol{\theta}^{(t)}$，不论 transition density 是何种形式，detail balance 条件对后验分布 $p(\boldsymbol{\theta} \mid \boldsymbol{y})$ 总是成立的，因为 $p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y}) = p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{\theta}^{(t)}) p(\boldsymbol{\theta}^{(t)} \mid \boldsymbol{y})$.

Remarks

1. 在 M-H 算法中，如果选取的 proposal distribution $\tilde{g}(\boldsymbol{\theta} \mid \boldsymbol{\theta}^{(t)})$ 是一个对称的条件分布，则算法退化为 Metropolis 算法。与 Metropolis 算法类似，M-H 算法产生的 Markov chain 也可能出现重复的样本。

2. 如果 M-H 算法的目标分布 $\pi(\boldsymbol{\theta})$ 可以分解为如下两部分：

$$\pi(\boldsymbol{\theta}) \propto \alpha(\boldsymbol{\theta}) g(\boldsymbol{\theta})$$

其中 $g(\boldsymbol{\theta})$ 在 $\pi(\boldsymbol{\theta})$ 中占主导地位且对应一个容易抽样的分布。可将 proposal distribution 选为 $g(\boldsymbol{\theta})$, 此时 M-H 算法的接受比率简化为

$$r_t = \frac{\pi(\boldsymbol{\theta}^\star)}{\pi(\boldsymbol{\theta}^{(t)})} \cdot \frac{g(\boldsymbol{\theta}^{(t)})}{g(\boldsymbol{\theta}^\star)} = \frac{\alpha(\boldsymbol{\theta}^\star)}{\alpha(\boldsymbol{\theta}^{(t)})}.$$

下面以一个简单的例子说明为什么 Gibbs sampler 也是 M-H 算法的一个特例。假设要抽样的目标分布是一个二元分布 $\pi(u, v)$. Gibbs sampler 通过轮流从每个变量的 full conditional distribution 抽样产生一列收敛到目标分布的样本：给定当前样本 $(u^{(t)}, v^{(t)})$,

- 抽取 $u^{(t+1)} \sim \pi(u \mid v^{(t)})$
- 抽取 $v^{(t+1)} \sim \pi(v \mid u^{(t+1)})$.

M-H 算法可以如下构造：分别为随机变量 U, V 选取 proposal distribution $g_u(u \mid u', v')$ 和 $g_v(v \mid u', v')$. 给定当前样本 $(u^{(t)}, v^{(t)})$,

1. 更新 U:

   (a) 抽取 $u^\star \sim g_u(u \mid u^{(t)}, v^{(t)})$

   (b) 计算接受比率

$$ru_t = \frac{\pi(u^\star, v^{(t)})}{\pi(u^{(t)}, v^{(t)})} \cdot \frac{g_u(u^{(t)} \mid u^\star, v^{(t)})}{g_u(u^\star \mid u^{(t)}, v^{(t)})}$$

   (c) 令

$$u^{(t+1)} = \begin{cases} u^\star & \text{概率 } \min(ru_t, 1) \\ u^{(t)} & \text{概率 } 1 - \min(ru_t, 1) \end{cases}$$

2. 更新 V:

   (a) 抽取 $v^\star \sim g_v(v \mid u^{(t+1)}, v^{(t)})$

   (b) 计算接受比率

$$rv_t = \frac{\pi(u^{(t+1)}, v^\star)}{\pi(u^{(t+1)}, v^{(t)})} \cdot \frac{g_v(v^{(t)} \mid u^{(t+1)}, v^\star)}{g_v(v^\star \mid u^{(t+1)}, v^{(t)})}$$

(c) 令

$$v^{(t+1)} = \begin{cases} v^\star & \text{概率 } \min(rv_t, 1) \\ v^{(t)} & \text{概率 } 1 - \min(rv_t, 1). \end{cases}$$

如果在上述 M-H 算法中，将 proposal distributions 选为目标分布 $\pi(u, v)$ 的 full conditional distributions, 即 $g_u(u \mid u', v') = \pi(u \mid v')$, $g_v(v \mid u', v') = \pi(v \mid u')$，则 M-H 算法每步的接受比率为

$$\begin{aligned} ru_t &= \frac{\pi(u^\star, v^{(t)})}{\pi(u^{(t)}, v^{(t)})} \cdot \frac{g_u(u^{(t)} \mid u^\star, v^{(t)})}{g_u(u^\star \mid u^{(t)}, v^{(t)})} \\ &= \frac{\pi(u^\star, v^{(t)})}{\pi(u^{(t)}, v^{(t)})} \cdot \frac{\pi(u^{(t)} \mid v^{(t)})}{\pi(u^\star \mid v^{(t)})} \\ &= \frac{\pi(v^{(t)})}{\pi(v^{(t)})} = 1 \end{aligned}$$

同理可得 $rv_t = 1$. 因此如果能从目标分布的 full conditional distribution 抽取候选样本，则该样本被接受的概率为 1，此时 M-H 算法退化为 Gibbs sampler.

5 Hamiltonian Monte Carlo (HMC) 方法

Metropolis 算法随机游走的行为有时会导致 Markov chain 在空间中曲折移动很长时间才能收敛到目标分布，且样本之间的相关性较高使得估计目标期望的有效样本很少。

考虑如下一个简单的 Bayesian 模型:

$$y_i = \beta_0 + \beta_1 x_i + \varepsilon_i,\ \varepsilon_i \overset{iid}{\sim} N(0, \sigma^2),\ i = 1, \ldots, n.$$

为参数选取的 priors 为

$$\begin{aligned} \beta_0 &\sim N(0, 10^2) \\ \beta_1 &\sim N(0, 10^2) \\ \sigma &\sim p(\sigma) = \frac{2}{\pi(1 + \sigma^2)} \cdot \mathbf{1}(\sigma > 0) \quad \text{(half-Cauchy)} \end{aligned}$$

在真实值 $\beta_0 = 1$, $\beta_1 = 2$ 以及 $\sigma = 0.5$ 下生成 $n = 100$ 个观察值，分别用 M-H 算法和 Hamiltonian Monte Carlo (HMC) 算法估计参数的后验分布。两种方法生成的 Markov chains 在截距 β_0 和斜率 β_1 所在的参数空间的动态移动情况分别见 `https://github.com/wangronglu/statcomp/blob/gh-pages/video/MCMC.mp4` 和 `https://github.com/wangronglu/statcomp/blob/gh-pages/video/NUTS.mp4`. 可以看到，M-H 算法生成的 Markov chains 通过随机游走的方式探索参数空间效率较低，因为每一步移动的幅度较小（如果增大样本移动的幅度又可能会使大部分候选样本被拒绝），需要运行很长时间才能收敛到后验分布且样本间的相关性较高; 而 HMC 方法是一种基于梯度的抽样

方法，可以使样本每步移动的幅度更大，因此在探索参数空间时更高效，且能有效减少样本间的相关性。

5.1 Hamiltonian Dynamics

HMC 方法借鉴了物理中 Hamiltonian 运动系统的想法。假设在一个无摩擦的环境下，小球沿着高低起伏的光滑表面运动。用 $\boldsymbol{\theta}_t \in \mathbb{R}^d$ 表示小球在 t 时刻的位置，$U(\boldsymbol{\theta}_t)$ 表示小球的势能；用 $\boldsymbol{q}_t \in \mathbb{R}^d$ 表示小球在 t 时刻的动量（$\boldsymbol{q}_t = m \cdot \boldsymbol{v}_t$），$K(\boldsymbol{q}_t)$ 表示小球的动能，则 $K(\boldsymbol{q}_t) = \boldsymbol{q}_t^\top \boldsymbol{q}_t/(2m)$.

在 Hamiltonian 系统中，小球的运动由 Hamiltonian 函数 $H(\boldsymbol{\theta}_t, \boldsymbol{q}_t)$ 描述。$\boldsymbol{\theta}$ 和 $\boldsymbol{q}$ 的每个分量的变化满足如下的 **Hamilton's Equations**:

$$\begin{aligned} \frac{d\theta_j}{dt} &= \frac{\partial H}{\partial q_j} \\ \frac{dq_j}{dt} &= -\frac{\partial H}{\partial \theta_j} \end{aligned} \tag{8}$$

$j = 1, \ldots, d$. 此时 Hamiltonian $H(\boldsymbol{\theta}_t, \boldsymbol{q}_t)$ 不随时间变化：

$$\frac{dH}{dt} = \sum_{j=1}^{d} \frac{d\theta_j}{dt}\frac{\partial H}{\partial \theta_j} + \frac{dq_j}{dt}\frac{\partial H}{\partial q_j} = \sum_{j=1}^{d} \frac{\partial H}{\partial q_j}\frac{\partial H}{\partial \theta_j} - \frac{\partial H}{\partial \theta_j}\frac{\partial H}{\partial q_j} = 0.$$

在 HMC 中，上述模型的 $\boldsymbol{\theta}$ 对应我们要抽样的变量，如果抽样的目标分布为 $\pi(\boldsymbol{\theta})$, 势能函数 $U(\boldsymbol{\theta})$ 对应 $-\log \pi(\boldsymbol{\theta})$. HMC 为每一个分量 θ_j 引入一个辅助的 "动量" 变量 q_j, 一般令 $\boldsymbol{q} \sim N_d(\mathbf{0}, M)$. 此时动能函数 $K(\boldsymbol{q})$ 对应

$$K(\boldsymbol{q}) = -\log p(\boldsymbol{q}) = \boldsymbol{q}^\top M^{-1} \boldsymbol{q}/2.$$

在 HMC 中，协方差矩阵 M 被称为 "mass matrix", 一般取为对角阵。Hamiltonian 函数通常取为如下形式：

$$H(\boldsymbol{\theta}, \boldsymbol{q}) = U(\boldsymbol{\theta}) + K(\boldsymbol{q}).$$

根据 Hamilton 方程(8), 此时

$$\begin{aligned} \frac{d\theta_j}{dt} &= \left[M^{-1}\boldsymbol{q}\right]_j \\ \frac{dq_j}{dt} &= \frac{\partial \log \pi(\boldsymbol{\theta})}{\partial \theta_j} \end{aligned} \tag{9}$$

$j = 1, \ldots, d$. HMC 方法在每步迭代中使用 Metropolis 方法更新 $(\boldsymbol{\theta}, \boldsymbol{q})$, 而候选样本是从 Hamiltonian 运动系统(9)中产生，这种方式产生的候选样本可以与当前样本距离较远，同时还能保证较高的接受概率。

5.2 Leapfrog 方法: 离散化 Hamilton 方程

为了模拟 Hamiltonian 系统的运动，我们以时间间隔 ϵ 对 Hamilton 方程进行离散化近似: 从 $t=0$ 开始，依次计算 $t=\epsilon, 2\epsilon, \ldots$ 时的 $\boldsymbol{\theta}(t)$ 和 $\boldsymbol{q}(t)$. 令 $H(\boldsymbol{\theta}, \boldsymbol{q}) = U(\boldsymbol{\theta}) + K(\boldsymbol{q})$, 其中 $K(\boldsymbol{q}) = \boldsymbol{q}^\top M^{-1}\boldsymbol{q}/2$. 取 M 为对角阵 $M = \mathrm{diag}(m_1, \ldots, m_d)$, 则 $K(\boldsymbol{q})$ 可写为

$$K(\boldsymbol{q}) = \sum_{i=1}^{d} \frac{q_j^2}{2m_j}.$$

- **Euler 方法**。对微分方程最简单的离散近似方法是 Euler 方法。根据 Hamilton 方程(8), Euler 方法对 Hamiltonian 运动系统的近似形式如下:

$$\begin{aligned} q_j(t+\epsilon) &= q_j(t) + \epsilon \frac{dq_j(t)}{dt} = q_j(t) - \epsilon \frac{\partial U\left(\boldsymbol{\theta}(t)\right)}{\partial \theta_j} \\ \theta_j(t+\epsilon) &= \theta_j(t) + \epsilon \frac{d\theta_j(t)}{dt} = \theta_j(t) + \epsilon \frac{q_j(t)}{m_j}. \end{aligned} \tag{10}$$

  $j = 1, \ldots, d$. 从 $t=0$ 开始，在初始值 $\boldsymbol{\theta}(0)$ 和 $\boldsymbol{q}(0)$ 下，按照(10)不断迭代，就得到了 $\boldsymbol{\theta}$ 和 $\boldsymbol{q}$ 在时间 $t=\epsilon, 2\epsilon, \ldots$ 的一条轨迹。

- **改进的 Euler 方法**。对 Euler 形式(10)稍作修改可以获得更好的近似效果:

$$\begin{aligned} q_j(t+\epsilon) &= q_j(t) - \epsilon \frac{\partial U\left(\boldsymbol{\theta}(t)\right)}{\partial \theta_j} \\ \theta_j(t+\epsilon) &= \theta_j(t) + \epsilon \frac{q_j(t+\epsilon)}{m_j} \end{aligned} \tag{11}$$

  即在更新 θ_j 时代入新的 q_j 的值.

- **Leapfrog 方法**。Leapfrog 方法的近似效果更好，它的形式为:

$$\begin{aligned} q_j(t+\epsilon/2) &= q_j(t) - \frac{\epsilon}{2} \cdot \frac{\partial U\left(\boldsymbol{\theta}(t)\right)}{\partial \theta_j} \\ \theta_j(t+\epsilon) &= \theta_j(t) + \epsilon \frac{q_j(t+\epsilon/2)}{m_j} \\ q_j(t+\epsilon) &= q_j(t+\epsilon/2) - \frac{\epsilon}{2} \cdot \frac{\partial U\left(\boldsymbol{\theta}(t+\epsilon)\right)}{\partial \theta_j} \end{aligned} \tag{12}$$

  Leapfrog 方法在每步更新时，先对动量变量 q_j 更新半步，代入新的 q_j 再对位置变量 θ_j 更新一整步，代入新的 θ_j 后再对 q_j 更新剩下的半步。

我们来看以上三种方法对一个一维 Hamiltonian 运动系统的近似效果。假设

$$H(\theta, q) = U(\theta) + K(q), \quad U(\theta) = \frac{\theta^2}{2}, \quad K(q) = \frac{q^2}{2}.$$

根据 Hamilton 方程(8),

$$\frac{d\theta}{dt} = q, \quad \frac{dq}{dt} = -\theta. \tag{13}$$

上述微分方程组有如下形式的解:

$$\theta(t) = r\cos(a+t), \quad q(t) = -r\sin(a+t). \tag{14}$$

其中 r 和 a 是常数。图5展示了使用 Euler 方法、改进的 Euler 方法和 leapfrog 方法得到的 $(\theta(t), q(t))$ 连续 20 步的近似轨迹。可以看到，在选取时间步长 $\epsilon = 0.3$ 下，leapfrog 方法精确地近似了真实轨迹，而 Euler 方法得到的轨迹严重偏离真实值且有发散的趋势，改进的 Euler 方法得到的轨迹不发散、与真实轨迹较接近但近似效果没有 leapfrog 方法好。

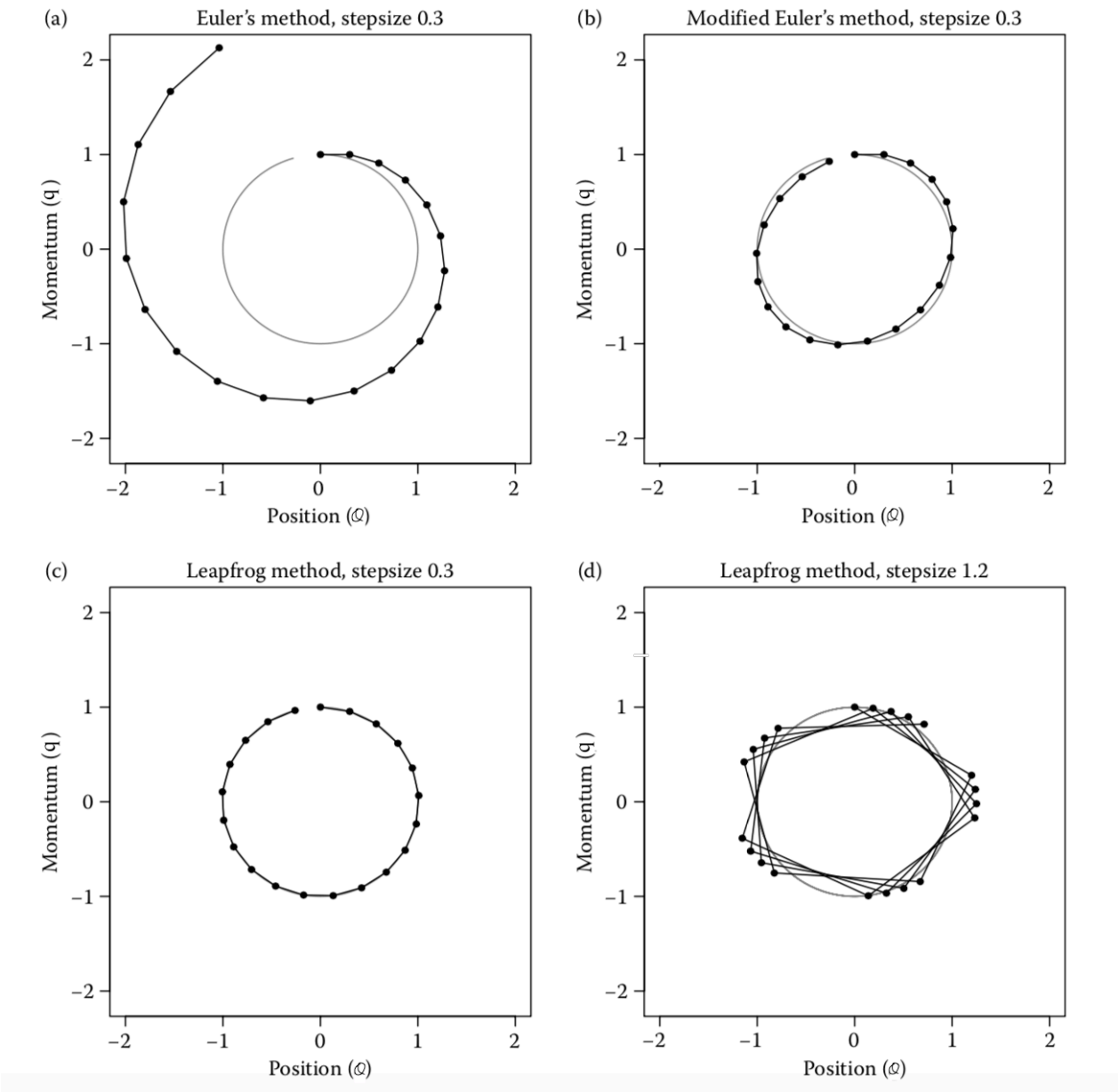


Figure 5: 使用三种离散方法对 Hamiltonian 系统(13)的近似效果。初始值为 $\theta = 0$, $q = 1$, 灰色曲线代表 (θ, q) 真实的轨迹(14)；(a)-(d) 分别展示了每种方法得出的 (θ, q) 连续 20 步的运动轨迹。在 (a), (b), (c) 中，时间间隔 $\epsilon = 0.3$, (d) 中 $\epsilon = 1.2$. Picture source: Neal et al. (2011)

对于微分方程的离散方法，人们用**局部误差** (local error) 描述从时间 t 到 $t+\epsilon$ 的近似误差，用**全局误差** (global error) 描述经过一段时间 T（经过 T/ϵ 个时间步）后的近似误差。如果方法的局部误差为 $O(\epsilon^p)$，则全局误差为 $O(\epsilon^{p-1})$. Euler 方法及改进的 Euler 方法的局部误差都是 $O(\epsilon^2)$, 全局误差 $O(\epsilon)$; leapfrog 方法的局部误差为 $O(\epsilon^3)$, 全局误差 $O(\epsilon^2)$ (Neal et al., 2011).

5.3 HMC 算法

在 HMC 算法中，目标分布 $\pi(\boldsymbol{\theta})$ 的归一化常数可以未知。因此在估计 Bayesian 模型时，可以令

$$U(\boldsymbol{\theta}) = -\log\left[L(\boldsymbol{y} \mid \boldsymbol{\theta})p(\boldsymbol{\theta})\right]$$

其中 $p(\boldsymbol{\theta})$ 是 prior density, $L(\boldsymbol{y} \mid \boldsymbol{\theta})$ 是观察值的似然函数。

给定初始值 $\boldsymbol{\theta}^{(0)}$, 假设已获得当前样本 $\boldsymbol{\theta}^{(t)}$, HMC 算法如下产生新的样本 $\boldsymbol{\theta}^{(t+1)}$:

1. 抽取 $\boldsymbol{q} \sim N_d(\mathbf{0}, M)$.

2. 从 $(\boldsymbol{\theta}^{(t)}, \boldsymbol{q})$ 出发，使用 leapfrog 方法按 Hamiltonian 机制(9)移动 L 步，每步时间间隔 ϵ, 得到候选样本 $(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)$.

   具体地，令 $\tilde{\boldsymbol{\theta}} = \boldsymbol{\theta}^{(t)}$, $\tilde{\boldsymbol{q}} = \boldsymbol{q}$, for $s = 1, \ldots, L$:

   (a) 将 $\tilde{\boldsymbol{q}}$ 更新半步:

   $$\tilde{\boldsymbol{q}} \leftarrow \tilde{\boldsymbol{q}} - \frac{\epsilon}{2}\nabla U(\tilde{\boldsymbol{\theta}})$$

   (b) 使用更新的 $\tilde{\boldsymbol{q}}$ 更新 $\tilde{\boldsymbol{\theta}}$:

   $$\tilde{\boldsymbol{\theta}} \leftarrow \tilde{\boldsymbol{\theta}} + \epsilon M^{-1}\tilde{\boldsymbol{q}} \tag{15}$$

   (c) 代入更新的 $\tilde{\boldsymbol{\theta}}$, 再对 $\tilde{\boldsymbol{q}}$ 更新半步:

   $$\tilde{\boldsymbol{q}} \leftarrow \tilde{\boldsymbol{q}} - \frac{\epsilon}{2}\nabla U(\tilde{\boldsymbol{\theta}})$$

   经过 L 个 leapfrog steps 后，（将动量向量反向，即令 $\tilde{\boldsymbol{q}} = -\tilde{\boldsymbol{q}}$）令 $(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star) = (\tilde{\boldsymbol{\theta}}, \tilde{\boldsymbol{q}})$.

3. 计算接受比率

   $$r = \exp\left[H(\boldsymbol{\theta}^{(t)}, \boldsymbol{q}) - H(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)\right] = \exp\left[U(\boldsymbol{\theta}^{(t)}) + K(\boldsymbol{q}) - U(\boldsymbol{\theta}^\star) - K(\boldsymbol{q}^\star)\right] \tag{16}$$

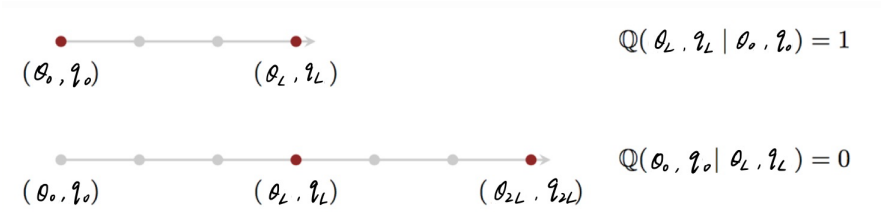
   令

   $$\boldsymbol{\theta}^{(t+1)} = \begin{cases} \boldsymbol{\theta}^\star & \text{概率 } \min(1, r) \\ \boldsymbol{\theta}^{(t)} & \text{概率 } 1 - \min(1, r). \end{cases}$$

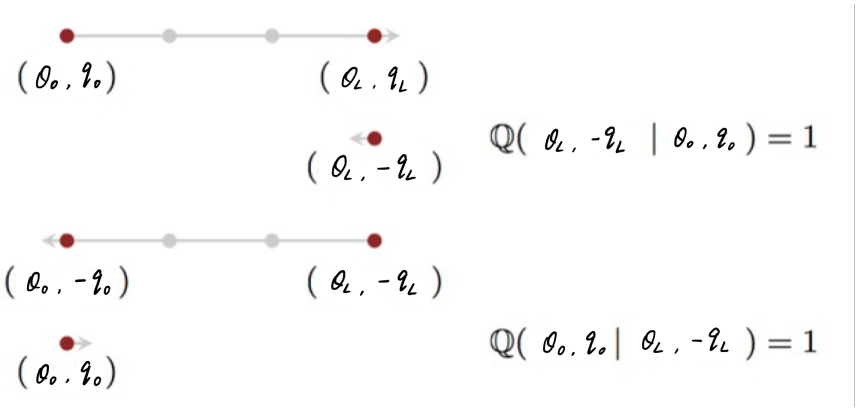
Remarks

1. 视频 `https://github.com/wangronglu/statcomp/blob/gh-pages/video/HMC.mp4` 展示了 HMC 生成 Markov chain 的动态过程。

2. 在 leapfrog 轨迹的最后将动量变量反向，是为了保证 proposal 分布是对称的。HMC 每步迭代使用的 “proposal distribution” 对应 L leapfrog steps, 而这一过程其实是确定的，没有随机性。用 $Q()$ 表示运行 L leapfrog steps 对应的变换，假设在 L 步 leapfrog “跳跃” 之前，$(\boldsymbol{\theta}, \boldsymbol{q})$ 的初始状态为 $(\boldsymbol{\theta}_0, \boldsymbol{q}_0)$, 经过 L leapfrog steps 到达 $(\boldsymbol{\theta}_L, \boldsymbol{q}_L)$, 则 $Q(\boldsymbol{\theta}_L, \boldsymbol{q}_L \mid \boldsymbol{\theta}_0, \boldsymbol{q}_0) = 1$, 但 $Q(\boldsymbol{\theta}_0, \boldsymbol{q}_0 \mid \boldsymbol{\theta}_L, \boldsymbol{q}_L) = 0$，如下图所示：



如果我们在 L leapfrog steps 结束后再将动量变量 $\boldsymbol{q}$ 反向，由于 Hamiltonian 系统的 leapfrog 轨迹是可逆的，此时接着运行 L leapfrog steps, $(\boldsymbol{\theta}, \boldsymbol{q})$ 将沿 “原路” 返回到初始位置，再将动量变量反向即得 $(\boldsymbol{\theta}_0, \boldsymbol{q}_0)$，如下图所示：



可以看到此时的 “proposal distribution” $Q()$ 是对称的。但在实践中可以省略对 $\boldsymbol{q}_L$ 的反向操作，因为 HMC 在后续迭代中只用到 $K(\boldsymbol{q}_L)$ 的值，而 $K(\boldsymbol{q}_L)$ 对应 $N_d(\mathbf{0}, M)$ 的 density, 因此 $K(-\boldsymbol{q}_L) = K(\boldsymbol{q}_L)$.

3. 由于 HMC 产生候选样本的机制是对称的，在选取的接受比率(16)下，HMC 生成的 Markov chain 满足 detail balance 条件。注意到 $(\boldsymbol{\theta}, \boldsymbol{q})$ 在 stationary distribution 下的联合概率密度为 $\exp\left[-H(\boldsymbol{\theta}, \boldsymbol{q})\right]$.

   - 在上述 HMC 迭代中，如果 $(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)$ 被接受，则 $(\boldsymbol{\theta}^{(t)}, \boldsymbol{q})$ 的 stationary density 乘以 $(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)$ 的 transition probability 为：

$$\exp\left[-H(\boldsymbol{\theta}^{(t)}, \boldsymbol{q})\right] Q(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star \mid \boldsymbol{\theta}^{(t)}, \boldsymbol{q}) \min\left\{1, \exp\left[H(\boldsymbol{\theta}^{(t)}, \boldsymbol{q}) - H(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)\right]\right\}$$

$$= Q(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star \mid \boldsymbol{\theta}^{(t)}, \boldsymbol{q}) \min\{\exp\left[-H(\boldsymbol{\theta}^{(t)}, \boldsymbol{q})\right], \exp\left[-H(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)\right]\} \tag{17}$$

$Q()$ 是对称的，因此在(17)中交换 $(\boldsymbol{\theta}^{(t)}, \boldsymbol{q})$ 和 $(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)$ 的位置结果不变，即 detail balance 成立。

- 如果 $(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)$ 被拒绝，detail balance 显然成立。

虽然 $\boldsymbol{q}^\star$ 和 $\boldsymbol{\theta}^\star$ 同时被接受或拒绝，由于 HMC 在每步迭代的开始都会更新 $\boldsymbol{q}$, 所以实践中并不需要保存被接受的 $\boldsymbol{q}$ 值。

4. 如果 HMC 算法产生的 Markov chain 不会陷入到局部区域，则一般是遍历的 (ergodic). 此时 detail balance 保证了 $(\boldsymbol{\theta}, \boldsymbol{q})$ 的 Markov chain 会收敛到 stationary distribution, 因此 marginal Markov chain $\{\boldsymbol{\theta}_t\}$ 会收敛到目标分布 $\pi(\boldsymbol{\theta})$.

   但是如果 leapfrog 轨迹具有某种周期性，如图5所示，且乘积 $L\epsilon$ 接近周期，此时经过 L leapfrog steps 产生的候选样本 $(\boldsymbol{\theta}^\star, \boldsymbol{q}^\star)$ 几乎与原点 $(\boldsymbol{\theta}^{(t)}, \boldsymbol{q}^{(t)})$ 重合, 则生成的 Markov chain $\{\boldsymbol{\theta}_t\}$ 不具有遍历性 (ergodicity). 解决的办法之一是：每次产生候选样本时，在较小的范围内随机选取 ϵ 和 L. Hoffman and Gelman (2014) 提出了 no U-turn sampler (NUTS)，在检测到 leapfrog 轨迹有 "调头" 趋势时 (动量向量 $\boldsymbol{q}$ 与位置的改变 $(\boldsymbol{\theta} - \boldsymbol{\theta}_0)$ 点积为负) 就停止移动。

5. Columbia University 的 Andrew Gelman 教授团队开发了 **Stan** 软件 (Carpenter et al., 2017) 及 R package `rstan` 和 `rstanarm`，可以为输入的 Bayesian 模型自动运行 HMC 算法 (auto-tuned HMC, no U-turn sampler).

6. 在 HMC 算法中，有三处可调参数 (tuning parameters): (i) 动量变量 $\boldsymbol{q}$ 的协方差矩阵 M; (ii) leapfrog steps 中的时间间隔 ϵ; (iii) leapfrog 的步数 L.

   具体调节方法是：先给 M 设定一个简单的形式，一般默认值是 $M = I$; 然后设定 ϵ 和 L 的值，Gelman et al. (2013) 建议选取 ϵ 和 L 使得 $\epsilon L = 1$, 比如 $\epsilon = 0.1$, $L = 10$. 如果目标分布 $\pi(\boldsymbol{\theta})$ 接近正态分布且 mass matrix M 与 $\boldsymbol{\theta}$ 在目标分步下的 inverse covariance matrix 较接近，按照(15) 将 $\boldsymbol{\theta}$ 移动 L 步，每步幅度约为 ϵM^{-1} 大致可以将 $\boldsymbol{\theta}$ 从目标分布的一端移动到另一端。Girolami and Calderhead (2011) 提出了 Riemannian adaptation, 可以根据目标分布的局部曲率 ($\approx$ inverse covariance matrix) 自动调整 mass matrix M, 使 Markov chain 更有效地探索状态空间。

   在选定的参数下，通过观察一些较短的 Markov chain 的接受比率和相关性再对参数做进一步调整。理论表明在满足一定假设条件下，HMC 最优接受比率约为 65% (Neal et al., 2011). 如

果 Markov chain 的整体接受比率较低，可能是 leapfrog 每步 “跳跃” 得太大，可以减小 ϵ, 增加 L. 相反如果接受比率过高，可以增加 ϵ, 减小 L.

7. 如果有些变量 θ_j 的 support 是某个特定区域，比如对于标准差参数 $\sigma > 0$, HMC 算法产生的候选样本可能超出该 support，此时目标分布 $\pi(\boldsymbol{\theta})$ 在候选样本处的概率密度为 0，接受比率 $r = 0$，候选样本一定会被拒绝。注意(16)中的 r 可写为

$$r = \frac{\pi(\boldsymbol{\theta}^\star)N(\boldsymbol{q}^\star \mid \mathbf{0}, M)}{\pi(\boldsymbol{\theta}^{(t)})N(\boldsymbol{q}^{(t)} \mid \mathbf{0}, M)}.$$

   - 对上述情形，另一种处理方法是 reparametrization, 比如对 $\sigma > 0$, 可以用 $\xi = \log\sigma$ 替换，对概率 $P \in (0,1)$, 可以引入 $\eta = \text{logit}(P)$.
   - 还有一种办法被称为 “反弹” (bouncing) , 即在每一个 leapfrog step 中, 检查目标 density 是否变为 0. 如果变为 0，则令动量变量 $\boldsymbol{q}$ 反向, 因为 Hamiltonian 系统的 leapfrog 轨迹是可逆的，这种方法仍然可以保证 proposal 机制是对称的。有时这种方法比直接拒绝更有效。

8. HMC 在每步迭代都先抽取一个新的 $\boldsymbol{q}$ 是为了改变 Hamiltonian 的值 $H(\boldsymbol{\theta}, \boldsymbol{q}) = -\log\pi(\boldsymbol{\theta}) - \log p(\boldsymbol{q})$, 即 $(\boldsymbol{\theta}, \boldsymbol{q})$ 的联合概率密度。因为在 Hamiltonian 运动机制下，当 leapfrog 时间步长 ϵ 足够小时，$H(\boldsymbol{\theta}, \boldsymbol{q})$ 的值几乎是不变的；如果在每步迭代中不对 $\boldsymbol{q}$ 重新抽样，则 $H(\boldsymbol{\theta}^{(t)}, \boldsymbol{q}^{(t)}) = U(\boldsymbol{\theta}^{(t)}) + K(\boldsymbol{q}^{(t)})$ 的值几乎不随迭代 t 变化，同时 $U(\boldsymbol{\theta}^{(t)})$ 和 $K(\boldsymbol{q}^{(t)})$ 一般都非负，因此 $U(\boldsymbol{\theta}^{(t)})$ 总是无法超过 $H(\boldsymbol{\theta}^{(t)}, \boldsymbol{q}^{(t)})$ 的初始值 $H(\boldsymbol{\theta}^{(0)}, \boldsymbol{q}^{(0)})$.

5.3.1 使用 HMC 估计一个 Bayesian 混合效应模型

当数据涉及分组结构且每个组内有多个观察值时，比较适合用混合效应模型 (mixed effects model) 来分析。混合效应模型可以区分数据在不同组间的变化 (between-group variation) 和同一组内的变化 (within-group variation), 它可以捕捉解释变量对结果的影响如何随分组不同而改变 (heterogenity among coefficients across groups)，且能描述同一组内观察值之间的相关性。

R package `lme4` 提供了一项研究睡眠不足 (sleep deprivation) 与反应时间 (reaction time) 关系的公开数据 sleepstudy (Belenky et al., 2003). 该数据记录了 18 个受试者在前 10 天睡眠不足的情况下每天的反应时间 (ms).

```
library(lme4)
str(sleepstudy)
'data.frame': 180 obs. of  3 variables:
```



```
$ Reaction: num  250 259 251 321 357 ...
$ Days    : num  0 1 2 3 4 5 6 7 8 9 ...
$ Subject : Factor w/ 18 levels "308","309","310",..: 1 1 1 1 1 1 1 1 1 1 ...
```

用 y_{ij} 表示受试者 j 的第 i 个反应时间的观察值，D_{ij} 表示 y_{ij} 对应的睡眠不足的天数。考虑到每个受试者初始的反应时间及睡眠不足对反应时间的影响都可能因人而异，建立如下的 mixed-effects model:

$$y_{ij} = \mu_0 + \gamma_{0j} + (\mu_1 + \gamma_{1j})D_{ij} + \varepsilon_{ij},\ \varepsilon_{ij} \overset{iid}{\sim} N(0, \sigma_e^2),\ i = 1, \ldots, n_j, \tag{18}$$

$$\begin{pmatrix} \gamma_{0j} \\ \gamma_{1j} \end{pmatrix} \overset{iid}{\sim} N_2 \left(\mathbf{0}, \Sigma = \begin{pmatrix} \sigma_0^2 & \sigma_{01} \\ \sigma_{01} & \sigma_1^2 \end{pmatrix} \right),\ j = 1, \ldots, J. \tag{19}$$

其中(18)描述的是同一组内 (within-group) 的数据分布，令

$$\boldsymbol{y}_j = \begin{pmatrix} y_{1j} \\ \vdots \\ y_{n_j,j} \end{pmatrix}, X_j = \begin{pmatrix} 1 & D_{1j} \\ \vdots & \vdots \\ 1 & D_{n_j,j} \end{pmatrix}, \boldsymbol{\mu} = \begin{pmatrix} \mu_0 \\ \mu_1 \end{pmatrix}, \boldsymbol{\gamma}_j = \begin{pmatrix} \gamma_{0j} \\ \gamma_{1j} \end{pmatrix},$$

则受试者 j 所有观察到的反应时间 $\boldsymbol{y}_j$ 服从条件分布:

$$\boldsymbol{y}_j \sim N_{n_j} \left(X_j \boldsymbol{\mu} + X_j \boldsymbol{\gamma}_j, \sigma_e^2 I_{n_j} \right). \tag{20}$$

且给定系数 $\{\boldsymbol{\mu}, \boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_J, \sigma_e^2\}$ 和解释变量 $\{X_j\}$，$\boldsymbol{y}_1, \ldots, \boldsymbol{y}_J$ 是条件独立的。我们称 $\boldsymbol{\mu} = (\mu_0, \mu_1)^\top$ 为固定效应 (fixed effects) 系数, 因为它们只是未知的常数，不是随机变量。系数向量 $\boldsymbol{\mu}$ 在所有组中都一样，μ_0 描述的是第 0 天（实验开始时）受试者的平均反应时间，μ_1 描述的是反应时间随睡眠不足天数增加的平均增长速率。

(19)描述的是不同组间 (between-group) 回归系数 $\boldsymbol{\gamma}_j$'s 的分布，我们称 $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_J$ 为随机效应 (random-effects) 参数，因为它们不是未知的常数 (向量)，而是未知的随机向量。注意(19)不是 $\boldsymbol{\gamma}_1, \ldots, \boldsymbol{\gamma}_J$ 的先验分布 (prior), 它是模型的一部分，其中的协方差矩阵 Σ 也是待估计的参数。(19)还起到了不同组间信息共享 (share information) 的作用，它使得从样本较小的组 (small sample size groups) 估计的 $\boldsymbol{\gamma}_j$'s 更稳定。

如果将条件分布(20)中的随机向量 $\boldsymbol{\gamma}_j$ 积掉，可得 $\boldsymbol{y}_j$ 的边际 (marginal) 分布:

$$\boldsymbol{y}_j \sim N_{n_j} \left(X_j \boldsymbol{\mu}, \sigma_e^2 I_{n_j} + X_j \Sigma X_j^\top \right). \tag{21}$$

从(21)可以看到，此时 $\boldsymbol{y}_j$ 的协方差矩阵不是对角阵，因此通过对 group-specific 系数 $\boldsymbol{\gamma}_j$'s 加入随机性(19)，混合效应模型也实现了描述组内观察值之间的相关性。

为了更好地描述模型参数的不确定性，我们使用 Bayesian 方法估计上述混合效应模型(18) - (19), 为此需要给每个参数设定先验分布 (priors). 为保证先验分布包含参数的真实值，可以为 μ_0, μ_1 选取正态先验分布，为 σ_e^2 和 Σ 分别选取 inverse-gamma 和 inverse-Wishart 分布。`rstan` 在估计 Bayesian 模型时，一般推荐让协方差矩阵服从 LKJ prior (Lewandowski et al., 2009)，它使 HMC 算法运行得更高效，同时保证抽取的协方差矩阵是对称正定的。LKJ prior 一般加在 correlation matrix 的 Cholesky 分解矩阵上，因此令 covariance matrix

$$\Sigma = \begin{pmatrix} \sigma_0 & 0 \\ 0 & \sigma_1 \end{pmatrix} \Omega \begin{pmatrix} \sigma_0 & 0 \\ 0 & \sigma_1 \end{pmatrix}. \tag{22}$$

矩阵 Ω 即是 $\boldsymbol{\gamma}_j$ 的 correlation matrix, 且 Ω 也是对称正定矩阵，因此存在 Cholesky 分解

$$\Omega = LL^\top \tag{23}$$

其中 L 是下三角矩阵。以下我们令 L 服从 LKJ prior.

根据(22)和(23), 我们在 `rstan` 中也对 $\boldsymbol{\gamma}_j$'s 重新参数化，令

$$\boldsymbol{\gamma}_j = \begin{pmatrix} \sigma_0 & 0 \\ 0 & \sigma_1 \end{pmatrix} L\boldsymbol{\eta}_j,\ \boldsymbol{\eta}_j \sim N_2(\mathbf{0}, I_2),\ j = 1, \ldots, J. \tag{24}$$

设置好模型和 priors 后，我们用 `rstan` 估计参数的后验分布 (posteriors). 首先，需要把数据组织成一个 list object:

```
d_stan = list(Subject = as.numeric(factor(sleepstudy$Subject,
              labels=1:length(unique(sleepstudy$Subject)))),
              Days = sleepstudy$Days,
              RT = sleepstudy$Reaction/1000,
              N = nrow(sleepstudy),
              J = length(unique(sleepstudy$Subject)) )
```

这里我们用 `N` 记录 y 的观察值个数 ($N = \sum_j n_j$)，用 `J` 表示受试者的个数，用 `RT` 表示反应时间 (y) 并将单位由毫秒 (ms) 转化为秒 (s). 在 sleepstudy dataset 中，`Subject` 是一个 factor 变量，在使用 `rstan` 时我们将其变成数值变量。

然后打开一个文本文档 (txt file)，输入以下几部分程序，保存为 ".stan" 文件，比如 "sleep_model.stan".

在该文档中首先对数据进行变量声明，比如整数还是实数，向量还需声明长度，因为 rstan 的底层语言是 C++. 变量还可以设置取值的上下界，更多细节可以参考 Stan 手册https://mc-stan.org/docs/2_22/reference-manual/index.html.

```
data {
  int<lower=1> N;            //number of observations
  real RT[N];                //reaction time

  int<lower=0,upper=9> Days[N];   //predictor (days of sleep deprivation)

  // grouping factor
  int<lower=1> J;                   //number of subjects
  int<lower=1,upper=J> Subject[N];  //subject id
}
```

其次列出待估计的参数:

```
parameters {
  vector[2] mu;                   // fixed-effects parameters
  real<lower=0> sigma_e;            // residual std
  vector<lower=0>[2] sigma_gam;       // random effects standard deviations

  // declare L to be the Choleski factor of a 2x2 correlation matrix
  cholesky_factor_corr[2] L;

  matrix[2,J] eta;                  // random effect matrix
}

transformed parameters {
  // this transform random effects so that they have the correlation
  // matrix specified by the correlation matrix above
  matrix[2,J] gamma;
  gamma = diag_pre_multiply(sigma_gam, L) * eta;
}
```


此处我们还增加了 transformed parameters block 以得到模型(18) - (19)中原始的 γ_j's 的估计，这些估计量将用于以下模型设定：

```
model {
  real m_RT; // conditional mean of y

  //priors
  L ~ lkj_corr_cholesky(1.5); // LKJ prior for the Cholesky factor of
                // correlation matrix
  to_vector(eta) ~ normal(0,1); // elementwise prior
  sigma_e ~ normal(0,5);        // prior for residual standard deviation
  mu[1] ~ normal(0.3, 0.5);    // prior for fixed-effect intercept
  mu[2] ~ normal(0.2, 2);      // prior for fixed-effect slope

  //likelihood
  for (i in 1:N){
    m_RT = mu[1] + gamma[1,Subject[i]] + (mu[2]+gamma[2,Subject[i]])*Days[i];
    RT[i] ~ normal(m_RT, sigma_e);
  }
}
```

在模型部分，我们设定参数的先验分布 (priors) 和数据的分布。这里我们为下三角矩阵 L 设定了 Stan 推荐的 LKJ prior, 该分布有一个参数 α, $\alpha = 1$ 相当于矩阵上的均匀分布，$\alpha > 1$ 的分布以单位阵 I 为 mode (概率密度最大的点)。

从文献中可知人的反应时间一般是 300 ms 左右，所以我们将固定效应参数 μ_0 的 prior mean 取为 0.3. 我们给斜率参数 μ_1 设定了一个 weakly informative prior, 让它的 prior 以一个很小的正数 0.2 为中心，但有较大的标准差。

最后我们在文档中加入以下代码储存随机效应的 correlation matrix Ω 的后验样本：

```
generated quantities {
  matrix[2, 2] Omega;
  Omega = L * L'; // so that it return the correlation matrix
}
```


在 R 中将 working directory 设为 sleep_model.stan 文件所在的文件夹，然后调用 `stan` 函数估计上述模型。以下代码将运行 HMC 算法生成 4 条独立的 Markov chains, 每条 chain 有 2000 个样本，其中前 1000 个样本作为 burnin 被丢掉 (也称 warmup)。

```
library(rstan)
# indicate stan to use multiple cores if available
options(mc.cores = parallel::detectCores())
sleep_model <- stan(file = "sleep_model.stan", data = d_stan,
                    iter = 2000, chains = 4)
```

我们首先检查参数后验样本的移动路径判断模型的收敛性:

```
traceplot(sleep_model, pars = c("mu"), inc_warmup = FALSE)
```

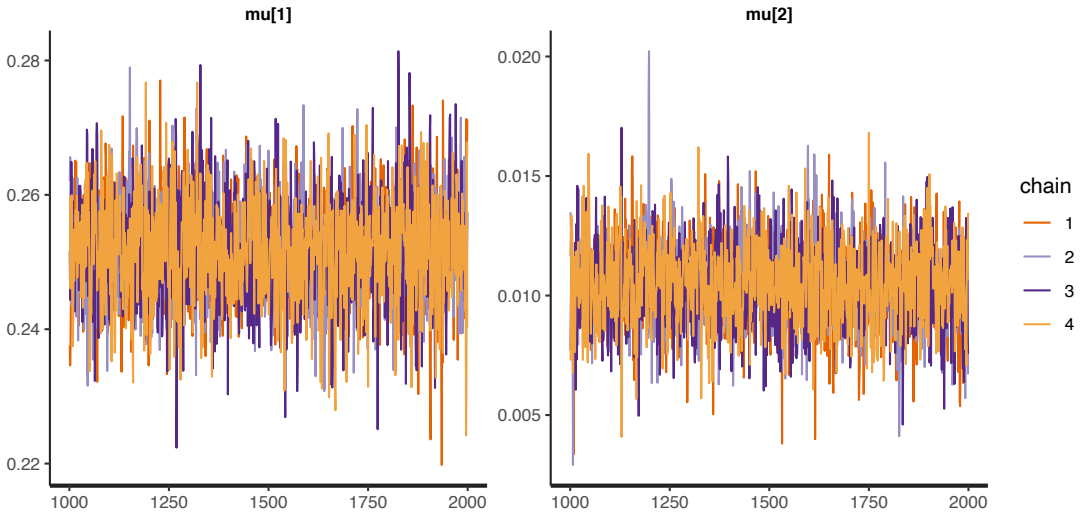


Figure 6: μ_0 和 μ_1 后验样本的 traceplot.

从图6可以看出: 4 条 Markov chains 基本都收敛到相同的 mode, 证明算法收敛到参数的后验分布。可以使用 `print` 函数总结参数估计的结果:

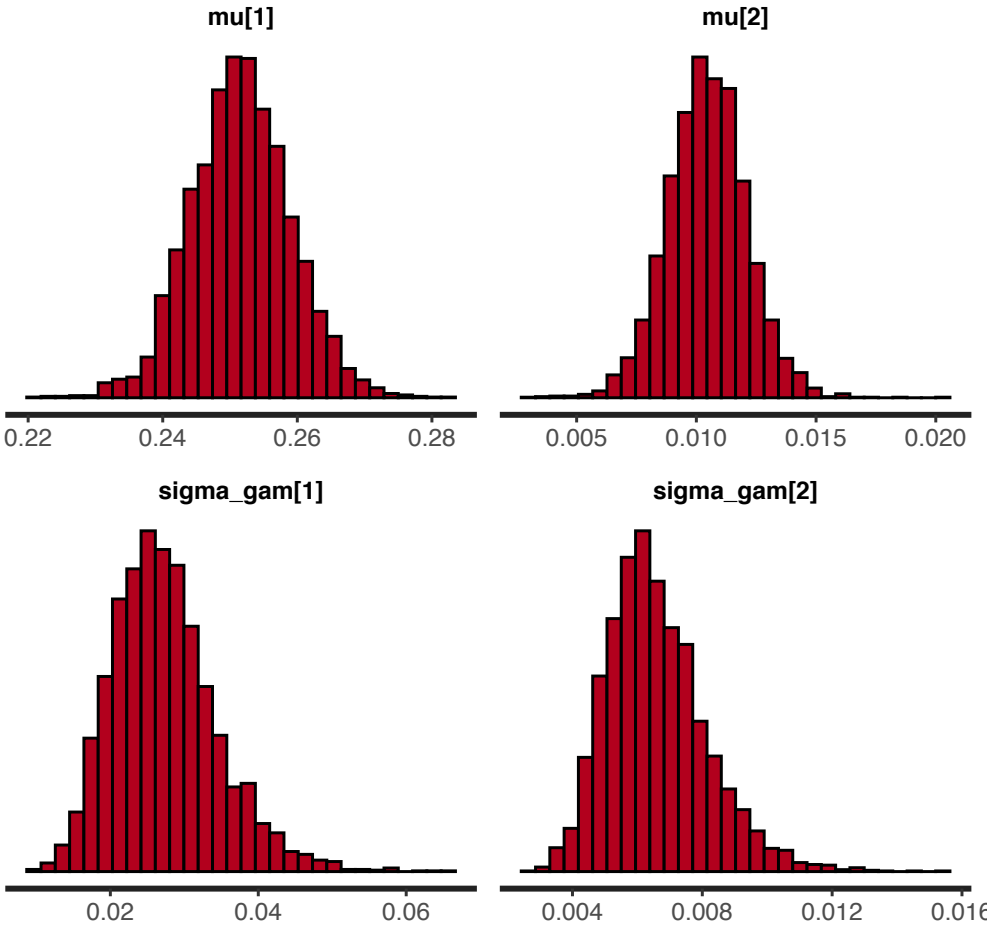
```
print(sleep_model, pars = c("mu"), probs = c(0.025, 0.975),
      digits = 3)
Inference for Stan model: sleep_model.
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.
```



```
       mean se_mean    sd  2.5% 97.5% n_eff  Rhat
mu[1] 0.252       0 0.007 0.237 0.266  2082 1.000
mu[2] 0.010       0 0.002 0.007 0.014  2496 1.001
```

其中 `n_eff` 代表 effective sample size. 如果生成的 Markov chains 收敛, 统计量 `Rhat` $\approx 1 \pm 0.01$. `plot` 函数可以直观地展示参数的后验分布:

```
plot(sleep_model, plotfun = "hist", pars = c("mu", "sigma_gam"))
```



再检查一下随机效应的 correlation matrix 的估计:

```
print(sleep_model, pars = c("Omega"), digits = 3)
Inference for Stan model: sleep_model.
```

```
4 chains, each with iter=2000; warmup=1000; thin=1;
post-warmup draws per chain=1000, total post-warmup draws=4000.

            mean se_mean    sd  2.5%    25%   50%  75% 97.5% n_eff   Rhat
Omega[1,1] 1.000     NaN 0.000  1.00  1.000 1.000 1.00 1.000   NaN    NaN
Omega[1,2] 0.082   0.008 0.288 -0.46 -0.125 0.075 0.29 0.641  1319 1.003
Omega[2,1] 0.082   0.008 0.288 -0.46 -0.125 0.075 0.29 0.641  1319 1.003
Omega[2,2] 1.000   0.000 0.000  1.00  1.000 1.000 1.00 1.000  4045 0.999
```

注意到 `Omega[1,1]` 对应的 `Rhat` 是 `NaN`，这并不意外，因为 `Omega[1,1]` 在抽样过程中始终等于 1.

最后与 frequentist 方法估计的混合效应模型(19)-(18)的 restricted MLE (REML) 做比较。在混合效应模型中，方差部分的 MLE 是有偏的，REML 考虑了估计 fixed effects 损失的自由度，得到的是无偏估计 (Zhang, 2015). R package `lme4` 的 `lmer` 函数可以给出线性混合效应模型的 REML:

```
fm1 = lmer(Reaction/1000 ~ Days + (Days | Subject), sleepstudy)
summary(fm1)
Linear mixed model fit by REML ['lmerMod']
Formula: Reaction/1000 ~ Days + (Days | Subject)
   Data: sleepstudy

REML criterion at convergence: -715.5

Scaled residuals:
    Min      1Q  Median      3Q     Max
-3.9536 -0.4634  0.0231  0.4633  5.1793

Random effects:
 Groups   Name        Variance  Std.Dev. Corr
 Subject  (Intercept) 6.119e-04 0.024737
          Days        3.508e-05 0.005923 0.07
 Residual             6.549e-04 0.025592
Number of obs: 180, groups:  Subject, 18

Fixed effects:
```



```
            Estimate Std. Error t value
(Intercept) 0.251405   0.006824  36.843
Days        0.010467   0.001546   6.771
```

可以看到参数的 posterior means 与它们的 REML 都很接近。

6 Sequential Monte Carlo (SMC) 方法

SMC 方法是一种估计状态空间模型 (state space model) 的常用方法。状态空间模型是研究动态系统的一个时间序列模型，它由状态方程和观测方程组成，描述一列可观测的变量和不可观测的状态变量之间的动态关系，其目标是估计未知的状态变量。该模型在信号处理、计算机视觉、金融等领域有广泛应用。SMC 方法为这类复杂的推断问题提供了有效的近似解。

6.1 线性高斯模型与 Kalman Filter

线性高斯模型是一类最简单的状态空间模型，它的一般形式为:

$$\begin{cases} X_t &= AX_{t-1} + U\epsilon_t,\ \epsilon_t \sim N_p(\mathbf{0}, I_p) \\ Y_t &= BX_t + V\eta_t,\ \eta_t \sim N_q(\mathbf{0}, I_q) \end{cases}$$

其中 X_t 和 Y_t 分别是 p 维和 q 维的随机向量，ϵ_t 和 η_t 是服从标准正态分布的噪声向量，$t = 1, \ldots, T$; A, B, U, V 是已知的常数矩阵。$\{Y_t\}_{t=1}^T$ 是可观测的随机向量，其观察值为 $\{\boldsymbol{y}_t\}_{t=1}^T$. $\{X_t\}_{t=1}^T$ 是不可观测的状态变量，其数目随时间 t 增加。我们的目标是估计给定 $\boldsymbol{y}_1, \ldots, \boldsymbol{y}_t$ 下 X_t 的条件分布，$t = 1, \ldots, T$. Kalman filter 可以给出这列条件分布的具体形式：

- 在初始时刻 $t = 0$，已知 $X_0 \sim N(\boldsymbol{\mu}_0, \Sigma_0)$.
- 对 $t = 1, 2, \ldots, T$
  - 给定 $X_{t-1} \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1} \sim N(\boldsymbol{\mu}_{t-1}, \Sigma_{t-1})$
  - 则 $(X_t, Y_t) \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_{t-1} \sim N(\boldsymbol{\theta}^{(t)}, \Omega^{(t)})$, 其中

$$\boldsymbol{\theta}^{(t)} = \begin{pmatrix} \boldsymbol{\theta}_X^{(t)} \\ \boldsymbol{\theta}_Y^{(t)} \end{pmatrix} = \begin{pmatrix} A\boldsymbol{\mu}_{t-1} \\ BA\boldsymbol{\mu}_{t-1} \end{pmatrix}$$

$$\Omega^{(t)} = \begin{pmatrix} \Omega_{XX}^{(t)} & \Omega_{XY}^{(t)} \\ \Omega_{YX}^{(t)} & \Omega_{YY}^{(t)} \end{pmatrix}$$

$$= \begin{pmatrix} A\Sigma_{t-1}A^\top + UU^\top, & (A\Sigma_{t-1}A^\top + UU^\top)B^\top \\ B(A\Sigma_{t-1}A^\top + UU^\top), & B(A\Sigma_{t-1}A^\top + UU^\top)B^\top + VV^\top \end{pmatrix}$$

– 此时 $X_t \mid \boldsymbol{y}_1, \ldots, \boldsymbol{y}_t \sim N(\boldsymbol{\mu}_t, \Sigma_t)$, 其中

$$\begin{aligned} \boldsymbol{\mu}_t &= \boldsymbol{\theta}_X^{(t)} + \Omega_{XY}^{(t)} \left(\Omega_{YY}^{(t)}\right)^{-1} (\boldsymbol{y}_t - \boldsymbol{\theta}_Y^{(t)}) \\ \Sigma_t &= \Omega_{XX}^{(t)} - \Omega_{XY}^{(t)} \left(\Omega_{YY}^{(t)}\right)^{-1} \Omega_{YX}^{(t)}. \end{aligned}$$

6.2 状态空间模型与 Importance Sampling 方法

一般的状态空间模型可写为以下形式:

$$\begin{cases} X_t \sim g_t(x \mid x_0, \ldots, x_{t-1}) & \text{状态方程} \\ Y_t \sim f_t(y \mid x_0, \ldots, x_t) & \text{观测方程} \end{cases} \tag{25}$$

其中每一时刻 t 的状态分布 g_t 和观测分布 f_t 是已知的，我们希望根据观察到的 $y_1, \ldots, y_t$，对未知的状态变量 X_t 进行实时 (online) 估计，一般是估计条件期望 $E(X_t \mid y_1, \ldots, y_t)$, $t = 1, 2, \ldots$，因为对任意随机变量 X，$\min_c E\left[(c - X)^2\right] = E(X)$. 该过程被称为 **filtering**. 如果希望估计未来时刻 $T > t$ 下的条件期望 $E(X_t \mid y_1, \ldots, y_T)$, 该过程被称为 **smoothing**. SMC 方法可实现对状态空间模型做 filtering 或 smoothing.

注意到

$$\begin{aligned} E(X_t \mid y_1, \ldots, y_t) &= \int x_t p(x_t \mid y_1, \ldots, y_t) dx_t \\ &= \int \cdots \int x_t p(x_0, \ldots, x_t \mid y_1, \ldots, y_t) dx_0 \cdots dx_t \\ &= \int \cdots \int x_t \frac{p(x_{0:t}, y_{1:t})}{p(y_{1:t})} dx_0 \cdots dx_t \\ &= \frac{\int \cdots \int x_t g_0(x_0) \prod_{s=1}^t \left(g_s(x_s \mid x_{0:s-1}) f_s(y_s \mid x_{0:s})\right) dx_0 \cdots dx_t}{\int \cdots \int g_0(x_0) \prod_{s=1}^t \left(g_s(x_s \mid x_{0:s-1}) f_s(y_s \mid x_{0:s})\right) dx_0 \cdots dx_t} \end{aligned} \tag{26}$$

随着时间 t 增加, (26)中的高维积分一般很难有解析形式，而数值积分方法的计算量过大、不可行。SMC 方法可以避免高维积分，是估计状态空间模型的常用方法。

Importance sampling 是 SMC 方法的基础。我们希望估计目标分布 $\pi(x)$（一般为多元分布）的期望 $\mu = E_\pi(X) = \int x\pi(x)dx$, 但是该积分一般没有显式表达式且很难从 $\pi(x)$ 抽样。Importancing sampling 的想法是先从一个 proposal 分布 $q(x)$ 中抽样，然后通过修正样本的权重来近似目标函数:

1. 从 proposal 分布抽取样本 $x_1, x_2, \ldots, x_N$.

2. 计算每个样本的权重：

$$w_j \propto \pi(x_j)/q(x_j),\ j = 1, \ldots, N.$$

3. μ 的估计量为

$$\hat{\mu} = \frac{\sum_{j=1}^{N} w_j x_j}{\sum_{j=1}^{N} w_j} \tag{27}$$

Importance sampling 在计算样本权重 w_j 时，允许 $\pi(x)$ 存在未知的归一化常数。可以证明(27)中的 $\hat{\mu}$ 是 μ 的一致 (consistent) 估计量。

Proof.

$$\hat{\mu} = \frac{\frac{1}{N}\sum_{j=1}^{N} \frac{\pi(x_j)}{q(x_j)} x_j}{\frac{1}{N}\sum_{j=1}^{N} \frac{\pi(x_j)}{q(x_j)}} \to \frac{E_q\left[x\frac{\pi(x)}{q(x)}\right]}{E_q\left[\frac{\pi(x)}{q(x)}\right]} = \frac{\int x\frac{\pi(x)}{q(x)}q(x)dx}{\int \frac{\pi(x)}{q(x)}q(x)dx} = \frac{\int x\pi(x)dx}{\int \pi(x)dx} = E_\pi(x).$$

□

6.3 Sequential Importance Sampling (SIS) and Resampling

使用 importance sampling 估计状态空间模型(25)的条件期望(26), 需要为状态变量选取一个 proposal 分布。由于状态变量的个数随时间增加，我们采用逐维建立 proposal 分布的策略，即选取

$$q(x_{0:t}) = q_0(x_0)q_1(x_1 \mid x_0)\cdots q_t(x_t \mid x_{0:t-1}) \tag{28}$$

其中每个分布 q_s 都是容易抽样的分布。状态空间模型(25)的目标分布为

$$\pi(x_{0:t}) = p(x_{0:t} \mid y_{1:t}) \propto g_0(x_0)\prod_{s=1}^{t} g_s(x_s \mid x_{0:s-1})f_s(y_s \mid x_{0:s})$$

则来自 proposal 分布(28)的样本 $x_{0:t}$ 的权重为

$$w(x_{0:t}) = \frac{\pi(x_{0:t})}{q(x_{0:t})} = \frac{g_0(x_0)\prod_{s=1}^{t} g_s(x_s \mid x_{0:s-1})f_s(y_s \mid x_{0:s})}{q_0(x_0)\prod_{s=1}^{t} q_s(x_s \mid x_{0:s-1})}.$$

可以采用如下迭代的方式计算样本权重：

$$w_t(x_{0:t}) = w_{t-1}(x_{0:t-1})\frac{g_t(x_t \mid x_{0:t-1})f_t(y_t \mid x_{0:t})}{q_t(x_t \mid x_{0:t-1})}, \quad t = 1, 2, \ldots \tag{29}$$

从 proposal 分布(28)抽取大量多元样本 $x_{0:t}^{(j)}$, $j = 1, \ldots, N$, 每个样本也被称为粒子 (particle), 按照(29)计算每个粒子 $x_{0:t}^{(j)}$ 的权重 $w_t^{(j)}$, 则 X_t 的（边际）条件分布的期望(26)可如下估计：

$$\hat{\mu} = \sum_{j=1}^{N} x_t^{(j)} w_t^{(j)} / \sum_{j=1}^{N} w_t^{(j)}. \tag{30}$$

上述方法被称为 **sequential importance sampling** (SIS) 方法。

SIS 方法的一个缺陷是：随着时间 t 增加，粒子的权重 $\{w_t^{(j)}\}$ 往往会变得越来越不均匀 (bias)，即只有少数粒子的权重很大，大部分粒子的权重非常小。使用过多权重很小的粒子计算(30)是一种浪费，因为这些权重很小的粒子对最终结果的贡献微乎其微，为此人们设计了一个**重抽样** (resampling) 步骤：

1. 给每个粒子 $x_{0:t}^{(j)}$ 分配一个概率 $\alpha_t^{(j)}$, $j = 1, \ldots, N$ 且 $\sum_{j=1}^N \alpha_t^{(j)} = 1$.

2. 对 $j = 1, \ldots, N$

   - 从集合 $\{x_{0:t}^{(i)} : i = 1, \ldots, N\}$ 中按概率 $\{\alpha_t^{(i)} : i = 1, \ldots, N\}$ 随机抽取一个样本 $x_{0:t}^{\star(j)}$.
   - 如果 $x_{0:t}^{\star(j)} = x_{0:t}^{(k)}$, 给 $x_{0:t}^{\star(j)}$ 赋予新权重 $w_t^{\star(j)} = w_t^{(k)}/\alpha_t^{(k)}$.

3. 输出新的带权样本集 $\{(x_{0:t}^{\star(j)}, w_t^{\star(j)}) : j = 1, \ldots, N\}$.

Gordon et al. (1993) 建议使用粒子归一化的权重 $\{\alpha_t^{(j)} = w_t^{(j)} / \sum_{j=1}^N w_t^{(j)} : j = 1, \ldots, N\}$ 作为重抽样的概率。Liu (2008) 从保护粒子多样性的角度给出如下形式的重抽样概率：

$$\alpha_t^{(j)} \propto \left[w_t^{(j)}\right]^\alpha, \ \alpha > 0, \ j = 1, \ldots, N.$$

但是当粒子的权重极度偏斜时，重抽样会造成粒子多样性的退化。Fearnhead and Clifford (2003) 为离散的状态空间模型设计了一个 “最优” 重抽样方法，该方法在所有无偏重抽样方法中使一个损失函数达到最小且可以较好地保护粒子多样性。

在 SIS 方法中加入重抽样步骤的算法被称为 **SMC** 算法，总结如下：

1. $t = 0$ 时，抽取 $x_0^{(j)} \sim q_0(x)$, 并令 $w_0^{(j)} = g_0(x_0^{(j)})/q_0(x_0^{(j)})$, $j = 1, \ldots, N$.

2. 对 $t = 1, \ldots, T$

   - 抽样：抽取 $\tilde{x}_t^{(j)} \sim q_t(x \mid x_{0:t-1}^{(j)})$, 并令 $\tilde{x}_{0:t}^{(j)} = (x_{0:t-1}^{(j)}, \tilde{x}_t^{(j)})$, $j = 1, \ldots, N$.
   - 更新权重：令 $\tilde{w}_t^{(j)} = w_{t-1}^{(j)} u_t^{(j)}$, 其中

   $$u_t^{(j)} = \frac{g_t(\tilde{x}_t^{(j)} \mid x_{0:t-1}^{(j)}) f_t(y_t \mid \tilde{x}_{0:t}^{(j)})}{q_t(\tilde{x}_t^{(j)} \mid x_{0:t-1}^{(j)})}, \quad j = 1, \ldots, N.$$

   - 推断：计算目标期望 $E(h(x_{0:t}) \mid y_{1:t})$ 的估计量

   $$\frac{\sum_{j=1}^N \tilde{w}_t^{(j)} h(\tilde{x}_{0:t}^{(j)})}{\sum_{j=1}^N \tilde{w}_t^{(j)}}.$$

- 重抽样：按照权重 $\{\alpha_t^{(j)} : j = 1, \ldots, N\}$ 对粒子集合 $\{\tilde{x}_{0:t}^{(j)} : j = 1, \ldots, N\}$ 进行重抽样，得到一组新的带权粒子集 $\{(x_{0:t}^{(j)}, w_t^{(j)}) : j = 1, \ldots, N\}$.

Lin et al. (2013) 对 SMC 方法做了一个很好的综述并给出一些应用实例。

References

Belenky, G., Wesensten, N. J., Thorne, D. R., Thomas, M. L., Sing, H. C., Redmond, D. P., Russo, M. B., and Balkin, T. J. (2003). Patterns of performance degradation and restoration during sleep restriction and subsequent recovery: A sleep dose-response study. *Journal of sleep research*, 12(1):1–12.

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., and Riddell, A. (2017). Stan: A probabilistic programming language. *Journal of statistical software*, 76(1).

Fearnhead, P. and Clifford, P. (2003). On-line inference for hidden markov models via particle filters. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 65(4):887–899.

Gelman, A., Stern, H. S., Carlin, J. B., Dunson, D. B., Vehtari, A., and Rubin, D. B. (2013). *Bayesian data analysis*. Chapman and Hall/CRC.

Girolami, M. and Calderhead, B. (2011). Riemann manifold langevin and hamiltonian monte carlo methods. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, 73(2):123–214.

Gordon, N. J., Salmond, D. J., and Smith, A. F. (1993). Novel approach to nonlinear/non-gaussian bayesian state estimation. In *IEE Proceedings F (Radar and Signal Processing)*, volume 140, pages 107–113. IET.

Hoff, P. D. (2009). *A first course in Bayesian statistical methods*. Springer Science & Business Media.

Hoffman, M. D. and Gelman, A. (2014). The no-u-turn sampler: adaptively setting path lengths in hamiltonian monte carlo. *Journal of Machine Learning Research*, 15(1):1593–1623.

Lewandowski, D., Kurowicka, D., and Joe, H. (2009). Generating random correlation matrices based on vines and extended onion method. *Journal of multivariate analysis*, 100(9):1989–2001.

Lin, M., Chen, R., Liu, J. S., et al. (2013). Lookahead strategies for sequential monte carlo. *Statistical Science*, 28(1):69–94.

Liu, J. S. (2008). *Monte Carlo strategies in scientific computing*. Springer Science & Business Media.

Neal, R. M. et al. (2011). Mcmc using hamiltonian dynamics. *Handbook of markov chain monte carlo*, 2(11):2.

Zhang, X. (2015). A tutorial on restricted maximum likelihood estimation in linear regression and linear mixed-effects model. *URL http://statdb1. uos. ac. kr/teaching/multi-grad/ReML. pdf*.

