《概率论与随机过程》第一次 **Project**

无 42 陈佳榕 2014011050

2016 年 11 月 14 日　星期一

1. 摘要（Abstract）

本文首先使用到了 Metropolis-Hastings 采样算法，它是基于马氏链蒙特卡洛 MCMC 方法的采样算法，该方法被广泛运用于各种随机模拟实验中。基于该方法我设计了完整的对二维高斯分布进行随机采样的采样算法。我使用一个二维高斯分布对所设计的算法进行了测试，并利用生成的随机样本对此二维高斯的相关系数作了估计，得到了与理论值误差在 0.3%以下的估计结果。

然后使用到了 AIS 算法、TAP 算法、RTS 算法，这几个方法是常用的 RBM 模型归一化常数估计方法，被广泛运用于深度学习之中。基于这几个方法我设计了有效的采样方法，对四种不同的 RBM 模型进行归一化常数估计，并对几种算法的准确性和效率作了对比。最后利用估计得到的归一化常数在真实的测试数据上算出四种不同模型的似然值并作了相应比较。

2. 介绍（Introduction）

为了模拟服从给定分布的随机变量，用生成一个易于实现的不可约遍历链作为随机样本并使其平稳分布为所需分布的方法称为马氏链蒙特卡洛（MCMC）方法。它被广泛应用于各种学科领域的科学计算，是马氏链理论的一个重要应用，发展日益蓬勃。特别是随着深度学习领域的蓬勃发展，该方法也得到了更多的关注。

在《应用随机过程》中林元烈对 MCMC 作了一个简短的介绍，给出了状态空间为至多可数情形下的 Metropolis-Hastings 算法的一种实现方法；《Pattern Recognition and Machine Learning》的作者 Christopher M Bishop 也简单介绍了 MCMC，并给出 Metropolis-Hastings 算法的关键思想；Jun S Liu 则在《Monte Carlo strategies in scientific computing》中对 MCMC 作了更系统的介绍，并且给出了更一般化的 Metropolis-Hastings 算法的实现步骤。

RBM 模型是深度学习中最重要和基础的模型之一，自然而然地，估计 RBM 模型的归一化常数就成了深度学习中一个必不可少的需要，随着深度学习越来越广泛的运用到各个领域中去，该归一化常数的估计也越发具有重要的作用。

目前，常用的 RBM 模型归一化常数的估计方法有 AIS，SAMS，Wang-Landau 算法，TAP 方法，RTS 等。在 Ruslan Salakhutdinov 的《Learning deep generative models》中作者介绍了 RMB 的概念，并给出了 AIS 算法的具体过程；在 Zhiqiang Tan 的《Optimally adjusted mixture sampling and locally weighted histogram analysis》中给出了 SAMS 算法的详细解释；《Training Restricted Boltzmann Machines via the Thouless-Anderson-Plamer Free Energy》中作者给出了 TAP 算法的思想及具体内容；《Partition Functions from Rao-Blackwellized Tempered Sampling》的作者则在 AIS 算法的基础上提出了 RTS 算法，并给出了 RTS 算法的具体实施步骤。对比中可以发现 TAP 方法的不足之处在于准确度较低，RTS 方法的不足在于计算效率较低而且准确度也不高，AIS 方法虽然准确度较高但是同时也提高了计算的代价。

3. 模型（Model）

(a)二维高斯的相关系数

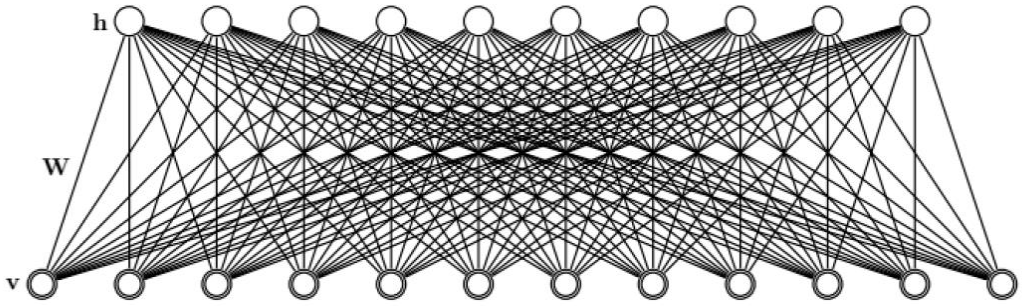
这里用到的 Metropolis-Hastings 采样算法是基于马氏链这一模型的，离散时间马氏链的定义为：对于∀k，

$$\mathrm{P}(X_{k+1}=n_{k+1}|X_k=n_k,\cdots,X_1=n_1)=\mathrm{P}(X_{k+1}=n_{k+1}|X_k=n_k)$$

也即当前的状态只与前一个状态有关。

(b)RBM (Restricted Boltzmann Machine)模型的归一化常数的估计

受限波尔兹曼机（RBM）是一种生成式随机神经网络，它由一些可见单元（对应观测变量）和隐藏单元（对应隐变量）构成，其中观测变量和隐变量都是二元变量，即其状态取自{0,1}。这个网络是一个二部图，只有可见单元和隐藏单元之间才存在边。RBM 是深度学习中最重要和基础的模型之一，如下图所示是一种常用的无向图模型：



一层观测变量和隐变量构成了 RBM 模型，变量取值范围为{0,1}，模型的参数为$\theta=\{\mathrm{W},\mathrm{b},\mathrm{a}\}$，RBM 模型的概率分布（变量的联合分布）为

$$\mathrm{p}(\mathrm{v},\mathrm{h};\theta)=\frac{1}{Z(\theta)}e^{-E(v,h;\theta)}$$

其中$\mathrm{Z}(\theta)=\sum_v\sum_h e^{-E(v,h;\theta)}$即为 RBM 的归一化系数，文献[5]的第 2 章对 RBM 有更加详细具体的介绍。

4. 基本理论和方法（Basic Theory and Method）

Metropolis-Hastings 算法：

首先假设给定的概率分布为π(x)，当前状态为$x^{(t)}$。

●第一步，从一个易于实现的条件转移矩阵 P 中抽取一个新的状态 y，并计算接受概率$\rho\left(x^{(t)},y\right)=\min\{1,\frac{\pi(y)P(y,x^{(t)})}{\pi(x)P(x^{(t)},y)}\}$；

●第二步，抽取 u~Uniform[0,1]，并作状态转移如下：

$$x^{(t+1)}=\begin{cases}y & u\le\rho\left(x^{(t)},y\right)\\ x^{(t)} & u>\rho\left(x^{(t)},y\right)\end{cases}$$

关于 Metropolis-Hastings 算法更详细的解释和正确性的证明可参看文献[4]。

AIS 方法：

1：选择一系列β_k满足$0=\beta_0<\beta_1<\cdots<\beta_K=1$

2：从$P_A=P_0=\prod_i\frac{1}{1+e^{-b_i}}$中采样得到$x_1$

3：对于$k=1:K-1$，根据x_k以及$T_K(x_k\to x_{k+1})$采样得到x_{k+1}

4：计算$\omega_{AIS}=\prod_{k=1}^{K}\frac{P_k^*(x_k)}{P_{k-1}^*(x_k)}$

关于 AIS 方法的具体思想和理论证明可参看文献[5]。

TAP 方法：

1：初始化m^v和m^h

2：迭代计算m^v和m^h

3：计算$\ln Z=-\Gamma(m^v,m^h)$

关于 TAP 方法更详尽的解释和证明可参看文献[3]。

RTS 方法：

1：初始化一系列β_k，r_k和$\hat{Z}_k$，k 取值从 1 到 K

2：从一系列β_k中随机抽取一个β，并初始化$\hat{c}_k=0$，k 取值从 1 到 K

3：重复 N 次，根据已有的β及 x，仿照 AIS 算法采样得到一个新的 x，计算

$q(\beta_k|x)$，然后从$q(\beta|x)$中采样得到$\beta|x$，更新$\hat{c}_k = \hat{c}_k + \frac{1}{N}q(\beta_k|x)$

4：更新$\hat{Z}_k^{RTS} = \hat{Z}_k \frac{r_1 \hat{c}_k}{r_k \hat{c}_1}$

关于 RTS 方法的更严密的过程和证明可参看文献[2]。

5. 方法（Method）

(a)二维高斯的相关系数的估计

这里使用 Metropolis-Hastings 算法对下述二维高斯分布进行随机采样

$$\mathcal{N}\left\{\begin{pmatrix}x_1\\x_2\end{pmatrix}\middle|\begin{pmatrix}5\\10\end{pmatrix},\begin{pmatrix}1 & 1\\1 & 4\end{pmatrix}\right\}$$

然后使用生成的随机样本估计二维高斯的相关系数ρ。

为了方便，将条件概率转移矩阵各个元素置为相等的，也就是$P\left(y, x^{(t)}\right) = P\left(x^{(t)}, y\right)$，因此可以简化接受概率的计算为$\rho\left(x^{(t)}, y\right) = \min\left\{1, \frac{\pi(y)}{\pi(x)}\right\}$，因此可以具体设计算法如下：

①初始化时间 t=0

②初始化状态$x^{(0)} = [0,0]$

③从已知分布$P\left(y|x^{(t)}\right)$中生成一个候选状态y^*

④计算接受概率$\rho\left(x^{(t)}, y\right) = \min\left\{1, \frac{\pi(y^*)}{\pi(x)}\right\}$

⑤从均匀分布 Uniform(0, 1)中生成一个随机值 u

⑥如果$u \le \rho\left(x^{(t)}, y\right)$, 接受新生成的值，即令$x^{(t+1)} = y$，否则保留在原状态，即令$x^{(t+1)} = x^{(t)}$

⑦如果$t = T$(人为设置的最大时间长度)，则结束，否则调回第③步

这里需要说明一下的是第③步中生成一个候选状态的方法，考虑到已经将条件概率转移矩阵各个元素都置为相等的了，也就是说从已知分布$P\left(y|x^{(t)}\right)$中生成一个候选状态y^*其实就相当于在状态空间随机生成一个 y。又考虑到高斯分布的“3σ原则”，所以实际上可以将状态空间压缩，假设第一个分量x_1的均值为μ_1，标准差为σ_1，第二个分量x_2的均值为μ_2，标准差为σ_2，那么可

以选择从$[\mu_1 - a\sigma_1, \mu_1 + a\sigma_1]$中随机抽取$x_1$，从$[\mu_2 - a\sigma_2, \mu_2 + a\sigma_2]$中随机抽取$x_2$，这里$a$为一个比 3 稍大的数。因为虽然“3σ原则”表明了高斯分布中，数值分布$[\mu - 3\sigma, \mu + 3\sigma]$范围中的概率为 99.73%，但是考虑到精度问题应该令a比 3 稍大一点。

另外，这里的$\pi(x)$即为需要的二维高斯分布概率密度，记$x = [x_1, x_2]$, 则有

$$\pi(x) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp\left\{-\frac{1}{2(1-\rho^2)}\left[\frac{(x_1-\mu_1)^2}{{\sigma_1}^2} - 2\rho\frac{(x_1-\mu_1)(x_2-\mu_2)}{\sigma_1\sigma_2} + \frac{(x_2-\mu_2)^2}{{\sigma_2}^2}\right]\right\}$$

(b)RMB 模型的归一化常数的估计

AIS 算法：（注：算法的代码实现参考了文献[5]原作者 Ruslan Salakhutdinov 的开源代码，具体参见附录里的代码文件）

为了方便，算法实际实施的时候作了一些约定，将$\theta_A = \{W, a, b\}$中的W, a均置为零，b^A则是通过学习测试数据得到的。

①选择一系列β_k满足$0 = \beta_0 < \beta_1 < \cdots < \beta_K = 1$的时候就简单地选择 0 到 1 的一个等差数列即可

②要从$P_A = P_0 = \prod_i \frac{1}{1+e^{-b_i}}$中采样得到$x_1$，先利用$b^A$计算得到概率分布，然后和一个随机生成的 Uniform$[0, 1]$概率分布作比较，大于的情况x_1相应置一，否则置零，这相当于作了一个服从P_A分布的采样

③初始化$ln\omega_{AIS} = -lnP_0^*(x_1) = -\sum_i b_i^A v_i - \sum_{j=1}^{F_B} ln2$，（因为$\beta_0 = 0$）注意这里没有计算$\sum_{j=1}^{F_A} ln2$是因为每一项$lnP^*(v)$都含有相同的这一项，在上下相减的时候可以消去

④对于$\mathrm{k} = 1{:}\,\mathrm{K} - 1$，首先利用当前的观测变量$x_k$，

计算$ln\omega_{AIS} = ln\omega_{AIS} + lnP_k^*(x_k)$，其中

$lnP_k^*(v) = (1 - \beta_k)\sum_i b_i^A v_i + \beta_k \sum_i b_i^B v_i + \sum_{j=1}^{F_B} ln(1 + e^{\beta_k(\sum_i W_{ij}^B v_i + a_j^B)})$，

将v代为x_k即可，

然后再计算得到已知x_k分布下隐变量 h 的概率分布

$P(h_j^A = 1|v) = g((1 - \beta_k)(\sum_i W_{ij}^A v_i + a_j^A))$及

$P(h_j^B = 1|v) = g(\beta_k(\sum_i W_{ij}^B v_i + a_j^B))$，

并据此采样得到 h 的样本，之后便可以通过已知的 h 分布计算得到下一个观测变量x_{k+1}的概率分布

$P(v_i' = 1|h) = g((1-\beta_k)b_i^A + \beta_k(\sum_j W_{ij}^B h_j^B + b_i^B))$，

之后即可通过采样得到下一个观测变量x_{k+1}（x_{k+1}即v'）

再次计算$ln\omega_{AIS} = ln\omega_{AIS} - lnP_k^*(x_{k+1})$，$lnP_k^*(x_{k+1})$的计算同上。

需要说明的是上面用到的$\mathrm{g(x)} = \frac{1}{1+e^{-x}}$。

⑤最后一次计算$ln\omega_{AIS} = ln\omega_{AIS} + \sum_i b_i^B v_i + \sum_{j=1}^{F_B} ln(1 + e^{\sum_i W_{ij}^B v_i + a_j^B})$，（因为$\beta_K = 1$）

⑥根据得到的$ln\omega_{AIS}$(实际是一个 M 维向量)求出$ln\hat{r}_{AIS} = ln\sum_{i=0}^{M}\omega_{AIS}^{(i)}$，并计算$lnZ_A = \sum_j ln2 + \sum_i ln(1 + e^{b_i})$，据此得到所需的$lnZ = lnZ_A + ln\hat{r}_{AIS}$。

TAP 算法：

①初始化m^v向量和m^h向量

②进行 T 次迭代，每次先计算

$$m_j^h[t+1] = \mathrm{sigm}\left[b_j + \sum_i W_{ij}m_i^v[t] - W_{ij}^2\left(m_j^h[t] - \frac{1}{2}\right)(m_i^v[t] - (m_i^v[t])^2)\right]$$

然后计算

$$m_i^v[t+1] =$$

$$\mathrm{sigm}[a_i + \sum_j W_{ij}m_j^h[t+1] - W_{ij}^2(m_i^v[t] - \frac{1}{2})(m_j^h[t+1] - (m_j^h[t+1])^2)]$$

其中$\mathrm{sigm(x)} = \frac{1}{1+e^{-x}}$

③可以计算得到最终结果

$$lnZ =$$

$$-\sum_i [m_i^v lnm_i^v + (1-m_i^v)\ln(1-m_i^v)]$$

$$-\sum_j [m_j^h lnm_j^h + (1-m_j^h)\ln(1-m_j^h)]$$

$$+\sum_i a_i m_i^v + \sum_i b_j m_j^h + \sum_{i,j}[W_{ij}m_i^v m_j^h + \frac{W_{ij}^2}{2}(m_i^v - (m_i^v)^2)(m_j^h - \left(m_j^h\right)^2)]$$

RTS 算法：

算法实施的时候将r_k均置 1，β_k选为 0 到 1 的等差数列

①初始化$\hat{Z}_k$均为 1

②初始化$\hat{c}_k$和x均为 0

③循环 N 次，先随机从 β 中抽取一个β_k（原算法是从（β|x）中抽取，但是实际上发现这种方法的性能较差，故我还是改为随机抽取，事实证明这样的效果较好），

然后同 AIS 方法一样根据已有的 β 和 x 生成下一个 x，并计算每个$lnf_k(x) =$

$$(1-\beta_k)\sum_i b_i^A x_i + \sum_{j=1}^{F_A} ln2 + \beta_k \sum_i b_i^B x_i + \sum_{j=1}^{F_B} ln(1+e^{\beta_k(\sum_i W_{ij}^B x_i + a_j^B)})$$

然后计算$q(\beta_k|x) = e^{lnf_k(x) - \sum_k lnf_k(x)}$，

并更新$\hat{c}_k = \hat{c}_k + \frac{1}{N} q(\beta_k|x)$

④更新$\hat{Z}_k^{RTS} = \hat{Z}_k \frac{\hat{c}_k}{\hat{c}_1}$，然后返回第②步继续计算直至迭代次数够了

⑤最后$lnZ = ln(\hat{Z}_k^{RTS}(end))$就是需要的结果

6. 方法分析（Analysis of Method）

(a)二维高斯的相关系数的估计

我设计的 Metropolis-Hastings 算法的性能总体来说在时间上受制于所选取的时间长度也即整个状态链的长度，这主要是因为在算法中为了生成指定长度的状态链需要作与状态链长度等量的循环。根据马氏链蒙特卡洛方法的性质，算法执行至产生足够长度的状态链之后，状态分布会趋于平稳分布，这里的平稳分布也即为我们所需要的给定分布，所以算法的稳定性是可以保证的，即多次独立重复实验中，我们总是能够得到一条趋于平稳分布的状态链。虽然算法是针对高斯分布提出的，但是算法本身是具有广泛的适用性

的，即对于各种分布都能很好地工作，因为只需要修改状态中的概率分布密度函数即可，而且因为算法本身就是针对多维状态提出的，所以也是适用于更高维的状态空间的。

(b) RMB 模型的归一化常数的估计

AIS 算法的性能在几个方法之中来说属于比较好的，因为这个方法没有经过近似，所以在理论上这个方法总是可以达到精确值的，当然实际上不可能让循环次数达到无限。由于 AIS 算法的收敛速度比较快，实际上在循环次数达到一个值（实测大约在 10000）之后就趋于收敛了。而且这个算法的稳定性比较好，进行多次的独立重复实验可以得到较为稳定的结果。

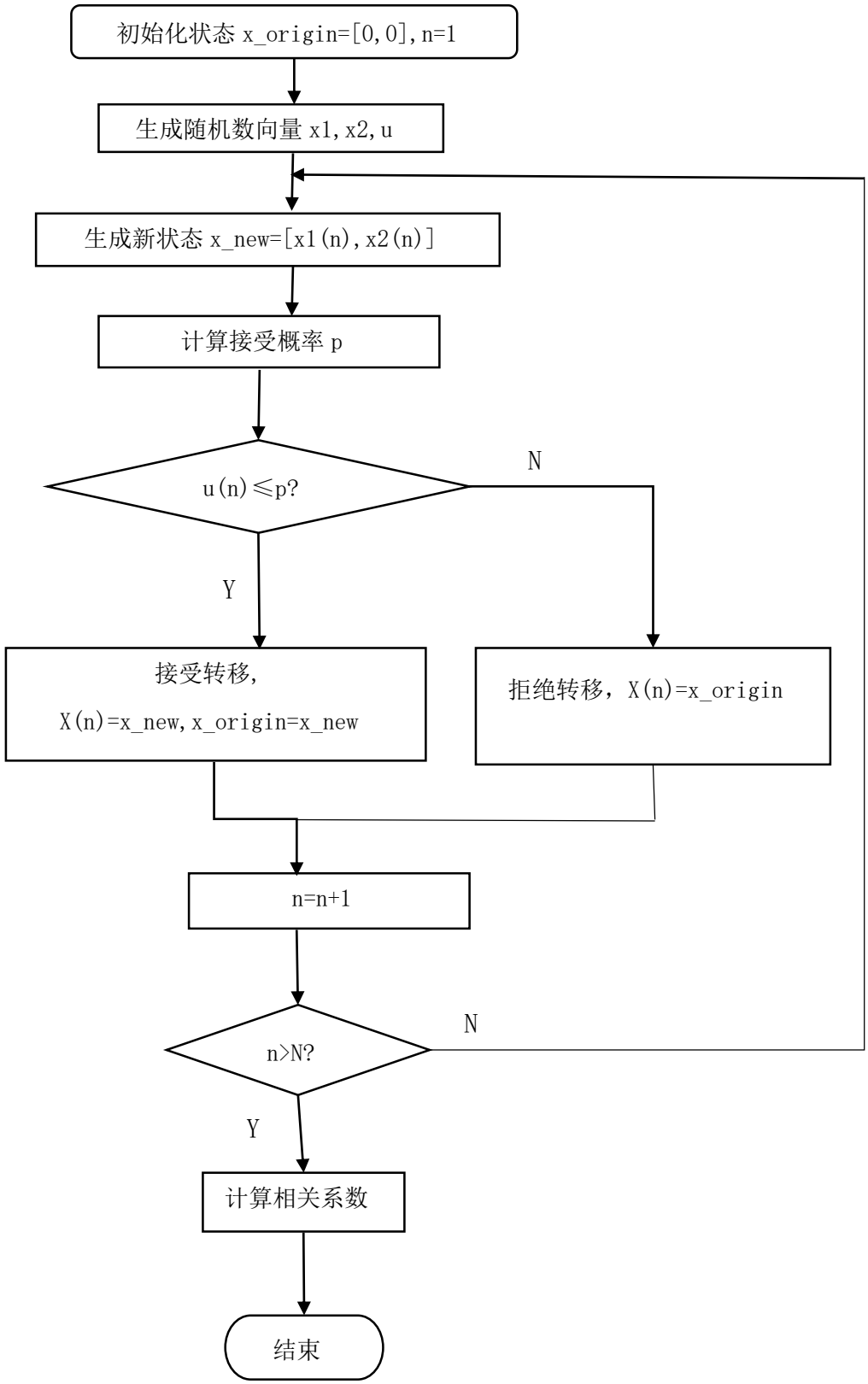
TAP 算法的优势在于计算效率较高，运行速度也是几者中最快的，但是由于其在原理上就已经是经过了近似的，所以在迭代次数实际为有限次的时候计算结果的准确度较差。由于算法本身没有引入随机性，所以其实给定输入之后总是能得到一样的结果，也就是说算法是很稳定的。而且收敛也很快，在迭代次数达到一个值（实测也大约在 10000）之后就趋于收敛了。

RTS 算法最大的劣势在于其计算效率太低，直接导致了运行速度比较慢。而且这个算法的收敛速度也比较慢，稳定性也不好，在多次独立重复独立实验中结果的变化较大。

7. 算法（Algorithm）

(a) 二维高斯的相关系数的估计

可以给出算法的流程图如下：


初始化状态 x_origin=[0,0],n=1
生成随机数向量 x1,x2,u
生成新状态 x_new=[x1(n),x2(n)]
计算接受概率 p
u(n)≤p?
N
Y
接受转移,
X(n)=x_new,x_origin=x_new
拒绝转移，X(n)=x_origin
n=n+1
n>N?
N
Y
计算相关系数
结束

(b) RMB 模型的归一化常数的估计

AIS 算法：

Set β =[0:$\frac{1}{K}$:1]

From P_A sample x_1

---

for k = 1 : K - 1

given x_k, sample x_{k+1} through T_k $(x_{k+1}|x_k)$

end

Calculate $\omega_{AIS} = \prod_{k=1}^{K} P_k^*(x_k)/P_{k-1}^*(x_k)$

TAP 算法：

Initialize m^v and m^h

---

for t = 1 : T

Given $m^v[t]$ and $m^h[t]$, calculate $m^h[t+1]$

then calculate $m^v[t+1]$

end

Calculate $\Gamma(m^v, m^h)$

RTS 算法：

Set $\{\boldsymbol{\beta}_k\}$=[0:$\frac{1}{K}$:1], $\{\boldsymbol{r}_k\}$ = 1

Initialize $\widehat{\boldsymbol{Z}}_k = \mathbf{1}$

Initialize $\hat{\boldsymbol{c}}_k = \mathbf{0}$, x = 0

for i = 1 : N

Select a β from $\{\boldsymbol{\beta}_k\}$

Sample a new x given the origin x

Update $\hat{\boldsymbol{c}}_k = \hat{\boldsymbol{c}}_k + \frac{\mathbf{1}}{\boldsymbol{N}}\boldsymbol{q}(\boldsymbol{\beta}_k|\boldsymbol{x})$

end

Update $\widehat{\boldsymbol{Z}}_k^{\boldsymbol{RTS}} = \widehat{\boldsymbol{Z}}_k \frac{\hat{\boldsymbol{c}}_k}{\hat{\boldsymbol{c}}_1}$

---

8. 算法分析（Analysis of Algorithm）

(a)二维高斯的相关系数的估计

我具体设计出来的 Metropolis-Hastings 算法的时间复杂度为 O(N)，其中 N 为算法设定的循环次数（也即之前提到的时间长度 T）,这是因为在算法中循环部分的执行次数占了最大的比例，其他部分皆可以看成 O(1)复杂度，所以整个算法的复杂度就主要由循环次数决定。至于空间复杂度也是 O(N)，因为通过 N 次循环生成了 N 个样本，每个样本都是二维向量，需要的空间为 2N，加上为了节省运行时间而在循环体外提前生成的三个随机数向量占用了 3N 的空间，而其他的变量占用的空间均可以看出 O(1)，因此空间复杂度为 O(N)。所以事实上我是采用了空间换时间的想法，在这里是很值得的，因为在这里空间一般来说总是足够的，并不是主要限制因素。

在性能方面，由于我利用 MATLAB 提供的随机数生成函数在循环开始之前就将需要用到的随机数先生成，避免了在循环体中每次都要生成随机数，这个细节对于算法的执行效率有很大的提高，经过测试，在 N=30000 的情况下优化之后的执行效率提高了大约 2.5 倍。

在可拓展性方面，只需要修改算法中用到的概率分布密度函数，就可以用于满足其他分布的随机采样。

(b) RMB 模型的归一化常数的估计

AIS 算法:

算法的复杂度主要是受限于循环次数，也即为 O(vhK)，其中 v、h 分别为观测变量和隐变量的长度，K 为算法中用到的 β 向量的长度。由于 MATLAB 对矩阵运算作了很好的优化，所以实际上复杂度要比 O(vhK) 小。

TAP 算法：

同理，TAP 算法的复杂度在理论上也是 O(vhN), 其中 N 为算法迭代的次数，v、h 分别为观测变量和隐变量的长度。当然由于 MATLAB 对矩阵运算的优化，实际上的复杂度要比这个低。

RTS 算法：

RTS 算法的复杂度主要与迭代次数和循环次数相关，理论上为 O(INK)，其中 I 为迭代次数，N 为循环次数, K 为算法中用到的 β 向量的长度。

9. 数值结果（Numerical Result）

仿真环境采用的 CPU 为 Intel Core i5-4210，操作系统为 Windows10，MATLAB 版本为 R2014a。

(a)二维高斯的相关系数的估计

测试的时间长度（即循环次数）为 30000，独立重复试验次数为 100。得到的二维高斯分布的相关系数的估计值为 0.49879，与理论值 0.5 相比相对误差为 0.24%，而方差也只有 0.00012（也即标准差为 0.011），可以看出准确率还是比较高的。这里误差的来源一方面是因为算法的执行次数是有限的，另一方面则是因为对状态空间作了压缩。而平均运行时间为 0.93s，可以看出运行速率还是较快的。

(b) RMB 模型的归一化常数的估计

AIS 算法:

M = 100，K = 10000，进行 50 次独立重复实验，平均运行时间为 245.25s（4 个模型的合计平均时间），结果如下表所示：

模型	归一化常数均值（取 ln 值）	方差
h10	226.10	0.0055
h20	221.06	0.0139
h100	348.38	0.0069
h500	460.00	0.5068

TAP 算法:

迭代次数为 10000，四个模型合计运行时间为 141.49s，得到的结果如下表所示：

模型	归一化常数（取 ln 值）
h10	211.76
h20	196.96
h100	341.31
h500	446.08

RTS 算法:

迭代次数为 250，循环次数 N=100，K=100，进行 32 次独立重复使用，平均运行时间为 458.39s，得到的结果如下表所示：

模型	归一化常数均值（取 ln 值）	方差
h10	225.18	0.7456
h20	213.01	1.2829
h100	335.00	1.7708
h500	442.00	6.2501

四个模型的似然值：

模型	似然值
h10	-173.83
h20	-145.07
h100	-112.99
h500	-99.64

10. 讨论（Discussion）

(a)二维高斯的相关系数的估计

从结果中可以看出，利用马尔科夫蒙特卡洛方法可以得到和理论值很接近的结果，有了原理上的保证，算法的实施才有了坚实的基础。但是在实践上由于受制于计算能力，我们不可能无限地计算下去，所以势必有个截断，这不可避免地会引入误差。所以还要进行多次的独立重复实验，以尽量消除随机误差，得到更精确的结果。从实验结果来看，得到的结果的相对误差较小，说明实验较好地证明了理论的有效性。

(b) RMB 模型的归一化常数的估计

综合三个算法的运行结果，可以看出 AIS 方法的整体性能要优于其他的，这是由 AIS 在原理上的优势决定的。具体地说，估计的准确性方面，AIS 方法优于其他两者，RTS 方法在前两个模型的准确性也要好于 TAP 方法，但是在

后两个模型的准确性上就不如 TAP 了，主要是因为在数据较少的时候 TAP 方法由于截断而引起的误差较大，而且在数据比较大的时候 RTS 算法的稳定性会下降，得到的结果的方差过大。在估计的计算效率上，明显 TAP 方法的计算效率是最好的，AIS 次之而 RTS 的计算效率最差，这也和之前的算法复杂性讨论相吻合。

(c)似然值比较

从结果来看，明显对于规模越小的模型，似然值取 ln 值之后的数值越小，也就是似然值本身越接近于零，说明算得的归一化系数越接近真实值。这是因为对于规模较小的模型，能够进行较为精确的计算，得到准确度较高的归一化常数，因此得到的似然值也越好。

11. 结论（Conclusion）

通过 Metropolis-Hastings 算法可以给出服从给定多维分布的随机样本，实验中通过这个方法得到了服从给定二维高斯分布的随机样本并据此计算得到与理论值吻合程度高的估计值。

然后通过 AIS 算法、TAP 算法以及 RTS 算法分别对四个不同的模型进行归一化系数的估计，发现 AIS 算法在三者之间表现出了最好的性能，其精确度是最好的，但是存在计算效率不够高的不足，这也是一个具有很大突破空间的地方，值得对其进行更进一步的研究，以其改进这个算法得到更好的性能。

12. 致谢（Acknowledgement）

最后，要感谢在本次大作业中及时答疑解惑的欧智坚老师、戴音培助教和陈杭助教，以及为我对算法的理解提供了极大帮助的张元鑫同学。

13. 参考文献（Reference）

[1] Christopher M Bishop. *Pattern recognition and machine learning*. springer, 2006.

[2] David Carlson, Patrick Stinson, Ari Pakman, and Liam Paninski. Partition functions from rao-blackwellized tempered sampling. *arXiv preprint arXiv:1603.01912*, 2016.

[3] Marylou Gabrié, Eric W Tramel, and Florent Krzakala. Training restricted boltzmann machine via the thouless-anderson-palmer free energy. In *Advances in Neural Information Processing Systems*, pages 640–648, 2015.

[4] Jun S Liu. *Monte Carlo strategies in scientific computing*. Springer Science & Business Media, 2008.

[5] Ruslan Salakhutdinov. *Learning deep generative models*. PhD thesis, University of Toronto, 2009.

[6] Zhiqiang Tan. Optimally adjusted mixture sampling and locally weighted histogram analysis. *Journal of Computational and Graphical Statistics*, (just-accepted), 2015.

[7] 林元烈. 应用随机过程. 清华大学出版社, 2002.