若干马氏链蒙特卡洛采样方法的比较与分析

王禹　无 48　2014011241

***Abstract*—本文对估计受限玻尔兹曼机 (RBM) 的归一化常数经常使用的四种马氏蒙特卡洛抽样方法进行了复现改进，一定程度上提高了抽样方法的效率，并对这四种抽样方法的稳定性进行了比较分析。我使用 MNIST 手写数字数据库对四种方法进行了测试与比较，以此来对四种抽样方法的效率与性能进行评估与分析。**

I. Introduction

手写体数字识别一直是一个研究热门问题，因为其应用广泛，且对识别的误识别率有着较高的要求。传统的分析方法是通过提取手写数字像素获得高维数特征集，再利用类似 PCA 等特征选择方法，筛选出维度较低的特征，在利用这些较低维度的特征训练神经网络，从而获得分类器。而本文所进行比较的四种方法，则均是直接将 $28 \times 28 = 784$ 个像素点作为特征，输入到受限玻尔兹曼机 (RBM) 的观测变量端，并将该一层网络训练为分类器，训练过程中需要计算 RBM 的归一化常数。由于观测变量有 784 个，隐变量可以取任意数目，故最少共有 $2^7 84$ 种观测变量与隐变量的组合情况。这个数字对于计算机系统而言是巨大的，因此使用遍历方法计算归一化常数会有巨大的代价，所以需要使用合适的抽样方法对归一化常数进行逼近与估计。

本文所比较的四种采样方法分为三种。以 AIS(Annealed Importance Sampling)[1] 为代表的方法采用的是构造了一个易于计算归一化常数的受限玻尔兹曼机 RBM_A，然后构造一系列从 0 到 1 的温度过渡因子，通过温度过渡，逐渐计算出目标受限玻尔兹曼机 RBM_B 的归一化常数。以 RTS(Rao-Blackwellized Tempered Sampling)[2] 和 SAMS(Self-adjusted mixture sampling)[3] 为代表的方法，同样是构造了一个易于计算归一化常数的 RBM_A，也同样构造出了一系列由 0 到 1 的温度过渡因子。不同的是，其通过迭代将所有中间过程的归一化常数同时计算。经过足够多的次数后，归一化常数趋于稳定，而归一化常数数列的最后一项，即为目标 RBM_B 的归一化常数。对于上述的两种方法，作者修改了部分迭代过程中的概率分布数学运算式，使得结果更快收敛。而 TAP(Thouless-Anderson-Palmer)[4] 所采用的是则是将归一化常数的 ln 值，即自由能进行展开。因为无论是玻尔兹曼机还是受限玻尔兹曼机 (RBM)，都应当是系统能量最低的状态，即稳定状态。因此使用迭代法自由能的 Legendre 变换形式的最低状态所对应的参数后，将其带入总 RBM 自由能的展开式，进行一定的近似截取，即可获得总的自由能，即归一化常数的值。对于该种方法，笔者将其从原文献的二阶形式，精确到三阶迭代，获得了更加稳定的算法。

II. Model

本实验的基本模型为受限玻尔兹曼机 (RBM)，是深度学习的重要基础模型之一，其结构如图 Fig. 1所示：

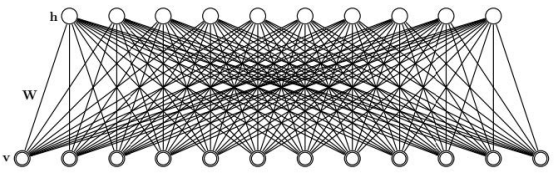

Fig. 1: RBM 结构模型

RBM 由一层观测变量和一层隐变量构成，变量均为 0,1 取值。模型的能量符合物理中的玻尔兹曼分布，即：

$$\begin{aligned} &E(v,h;\theta) \\ &= -v^T W h - b^T v - a^T h \\ &= -\sum_i \sum_j W_{ij} v_i h_j - \sum_i b_i v_i - \sum_j a_j h_j \end{aligned} \tag{1}$$

而 RBM 模型观测变量和隐变量的联合分布为：

$$p(v,h;\theta) = \frac{1}{Z(\theta)} e^{-E(v,h;\theta)} \tag{2}$$

其中，

$$Z(\theta) = \sum_v \sum_h e^{-E(v,h;\theta)} \tag{3}$$

为归一化常数。

III. Methods

A. Basic Method - MCMC

在叙述计算 RBM 归一化常数的采样方法之前，将先描述一般的 MCMC(Markov Chain Monte Carlo) 采样方法。

***Definition** 1:* 为要模拟服从给定分布 T 的随机变量，用生成一个易于实现的不可约遍历链 $X = \{X_n, n \geq O\}$ 作为随机样本，使其平稳分布为 π 的方法，称为马氏链蒙特卡罗方法.

而 Metropolis-Hastings 算法则为 MCMC 方法的一个改进，具体表述如下:

Algorithm 1: Metropolis-Hastings Algorithm	
1:	设 $\pi = (\pi(i), i \in S)$ 为任意给定的概率分布，而 $T = (T(i,j), i,j \in S)$ 为任选的易于实现的条概率转移矩阵
2:	给定 $\pi = (X_n \in S, n \geq 0)$，由 $T(X_n, \cdot)$ 抽取 Y，并计算 $\rho = min\{1, \pi(Y)T(Y,X_n)/(\pi(x)T(X_n,Y)\}$
3:	抽取 $U \sim U[0,1]$，若 $U < \rho$，则 $X_{n+1} = Y$，否则舍去 Y，返回步骤 2。

TABLE I: Metropolis-Hastings Algorithm

使用上述算法可以有效的获得大量目标分布的抽样样本，具体的算法性能分析，请见下一小节。

B. RBM 归一化参数与最似然概率估计

在详细描述四种具体的估计归一化常数的方法之前，需要首先说明一下抽样方法基于的模型。四种归一化参数的估计方法中，除了 TAP 外，都采用了相似的转移采样模型，具体模型如图 Fig. 2所示:

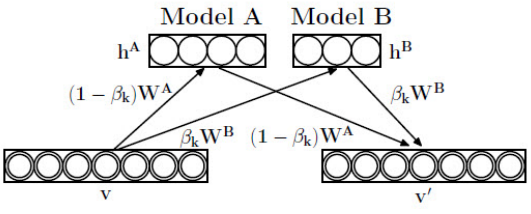

Fig. 2: Transition Operation Model

即构造一组 $[0,1]$ 的温度过渡因子，从易于计算归一化常数 ($\beta = 0$) 的 RBM_A，过渡到目标模型 ($\beta = 1$)RBM_B。在某一个中间过程，即某一个特定温度因子 β 下，上述模型转移抽样应当如下数学表达式所示:

$$p(h_j^A = 1|\mathbf{v}) = g\left((1-\beta_k)\left(\sum_i W_{ij}^A v_i + a_j^A\right)\right) \tag{4}$$

$$p(h_j^B = 1|\mathbf{v}) = g\left(\beta_k\left(\sum_i W_{ij}^B v_i + a_j^B\right)\right) \tag{5}$$

$$p(v_i' = 1|\mathbf{h}) = g((1-\beta_k)(\sum_i W_{ij}^A v_i + a_j^A) \tag{6}$$

$$+ \beta_k(\sum_i W_{ij}^B v_i + a_j^B)) \tag{7}$$

其中，$g(x) = \frac{1}{1+e^{-x}}$，为深度学习模型中常用的 logistic function。

TAP 算法虽然没有用到上述模型，但是仍然采用了恢复温度因子再做处理的的思想。

1) AIS[1]: AIS 方法使用一系列过渡的 β_k 值，计算相邻两个状态之间的归一化常数的比值，通过比值的连乘，将归一化常数从易于计算的 RBM_A 过渡到 RBM_B。本文实现的 AIS 具体算法如 Table II所示:

Algorithm 2: Annealed Importance Sampling	
1:	从 $0 = \beta_0 < \beta_1 < ... < \beta_K = 1$ 中随机选择一个 β_k
2:	用选择的 β_k 采样一个 x_1，满足最初的易于计算的 RBM 的 P_A
3:	for $k = 1 : K-1$ do
4:	通过 T_k 转移矩阵，以及目前的 x_k，采样 x_{k+1}。
5:	end for
6:	令 $\omega_k = \frac{Z_{k+1}}{Z_k} = \frac{P_{k+1}^*(x)}{P_{k+1}^*(x)}$，其中 $x \sim P_k$
7:	$Z_B = Z_A \prod_{k=1}^{K-1} \omega_k$，便可以获得最终结果

TABLE II: AIS Algorithm

原始的算法中，ω_k 一项表示成: $\omega_k = \frac{Z_{k+1}}{Z_k} = \frac{1}{M}\sum_{i=1}^{M} \frac{P_{k+1}^*(x^{(i)})}{P_{k+1}^*(x^{(i)})}$，$x^{(i)} \sim P_k$。即原始算法对每一种 β_k 值均作了 M 次抽样，之后将 M 次的结果作了平均。但是本算法中，将 M 取为 1，也得到了较好的结果。而 $P_k^*(\mathbf{v})$ 可以用如下的等式计算:

$$P_k^*(\mathbf{v}) = e^{\beta_k \sum_i b_i^B v_i}\left[\prod_{j=1}^{F_B}\left(1 + e^{\beta_k(\sum_i W_{ij}^B v_i + a_j^B)}\right)\right]$$

$$\times e^{(1-\beta_k)\sum_i b_i^A v_i}\left[\prod_{j=1}^{F_A}\left(1 + e^{(1-\beta_k)(\sum_i W_{ij}^A v_i + a_j^A)}\right)\right] \tag{8}$$

具体的实验，实验结果以及分析见下一节。

2) TAP[4]: 由前述可知，RBM 的联合分布为 $P(v,h)=Z^{-1}e^{-E(v,h)}$,Z 为归一化常数。取对数后，可得

$$\mathcal{L}=lnP(v)=-F^c(v)+F \tag{9}$$

其中，$F=-\ln Z$ 为 RBM 的自由能，$F^c(v)=-\ln(\sum_h e^{-E(v,h)})$ 为 RBM 的钳制自由能。本文关心重点为归一化常数的估计，即 RBM 的**自由能**。而吉布斯-波尔兹曼分布的**能量**在基于配置 s 的情况下应为 $E(s)=-\sum_i a_i s_i-\sum_{(i,j)} W_{ij}s_i s_j$。为了降低计算**自由能**的计算量。首先恢复玻尔兹曼分布中的温度 β 对于基于玻尔兹曼分布的模型的影响，因为大多数模型中，将温度设定为常数 1，从而忽略温度的影响。之后，将**能量**用外部辅助场 q 重写，可以得到 $-\beta F[q]=ln\sum_s e^{-\beta E(s)+\beta\sum_i q_i s_i}$ 而在经过 Legendre 变换，引入共轭变量 $m=m_i$ 后，可以获得

$$-\beta\Gamma[m]=-\beta\max_q[F[q]+\sum_i q_i m_i] \tag{10}$$

而**自由能**则为应该是经过 Legendre 变换后 $q=0$ 的状态在经过反 Legendre 变换可得，即

$$-\beta F=-\beta F[q=0]=-\beta\min_m[\Gamma[m]]=-\beta\Gamma[m^*] \tag{11}$$

而对于任意的 $A(\beta,m)\equiv-\beta\Gamma[m]$，展开后可得：

$$\begin{aligned}A(\beta,m)=&A(0,m)+\beta\frac{\partial A(\beta,m)}{\partial\beta}\Big|_{\beta=0}\\&+\frac{\beta^2}{2}\frac{\partial^2 A(\beta,m)}{\partial\beta^2}\Big|_{\beta=0}+...\end{aligned} \tag{12}$$

因此可以得到：

$$\begin{aligned}-\beta\Gamma(m)=&-\sum_i[m_i\ln m_i+(1-m_i)\ln(1-m_i)]\\&+\beta\sum_i a_i m_i+\beta\sum_{(i,j)}W_{ij}m_i m_j\\&+\frac{\beta^2}{2}\sum_{(i,j)}W_{ij}^2(m_i-m_i^2)(m_j-m_j^2)\\&+\frac{2\beta^2}{3}\sum_{(i,j)}W_{ij}^3(m_i-m_i^2)(\frac{1}{2}-m_i)\\&(m_j-m_j^2)(\frac{1}{2}-m_j)+...\end{aligned} \tag{13}$$

而回到 RBM 模型，我将其**能量**的 Legendre 变换由文献 [4] 中的二阶推广的三阶形式，如下：

$$\begin{aligned}\Gamma(m^v,m^h)\approx&-S(m^v,m^h)-\sum_i a_i m_i^v-\sum_i b_j m_j^h\\&-\sum_{i,j}W_{ij}m_i^v m_j^h\\&-\sum_{i,j}\frac{W_{ij}^2}{2}(m_i^v-(m_i^v)^2)(m_j^h-(m_j^h)^2)\\&-\sum_{i,j}\frac{2}{3}W_{ij}^3(m_i-m_i^2)(\frac{1}{2}-m_i)\\&(m_j-m_j^2)(\frac{1}{2}-m_j)-...\end{aligned} \tag{14}$$

这里将温度 β 设置为 1，即我们所需要的 RBM 模型。上式精确到了三阶近似，若是只需要二阶近似，则把最后一项略去即可。而为了得到**自由能**，需要式 (14) 能够取的最小值。因此需要 $\frac{d\Gamma}{dm}=0$。因此可以通过如下经过推广的三阶迭代式，得到带入计算的 m^v,m^h:

$$\begin{aligned}m_j^h[t+1]\leftarrow g[&b_j+\sum_i W_{ij}m_i^v[t]\\&-\sum_i W_{ij}^2\left(m_j^h[t]-\frac{1}{2}\right)(m_i^v[t]-m_i^v[t]^2)\\&+\sum_i W_{ij}^3(m_i^v[t]-m_i^v[t]^2)(\frac{1}{2}-m_i^v[t])\\&(2(m_j^h[t]^2-m_j^h[t])+\frac{1}{3})]\end{aligned} \tag{15}$$

$$\begin{aligned}m_i^v[t+1]\leftarrow g[&a_i+\sum_j W_{ij}m_j^h[t+1]\\&-\sum_j W_{ij}^2\left(m_i^v[t]-\frac{1}{2}\right)(m_j^h[t+1]-(m_j^h[t+1])^2)\\&+\sum_j W_{ij}^3(m_j^h[t+1]-m_j^h[t+1]^2)(\frac{1}{2}-m_j^h[t+1])\\&(2(m_i^v[t]^2-m_i^v[t])+\frac{1}{3})]\end{aligned} \tag{16}$$

该 m^v,m^h 也为三阶近似项，若需要二阶近似，则将两式的最后一项舍去即可。

3) RTS[2] and SAMS[3]: RTS 方法与 SAMS 方法比较类似，在此一并介绍。首先先介绍 RTS 方法。该方法也是构造出一系列温度过渡因子 $0=\beta_1<\beta_2<...<\beta_K=1$ 与 AIS 一致。在某一温度因子情况下的中间过程，概率密度分布也与 AIS 一致，即

$$p(x|\beta_k)=\frac{f_k(x)}{Z_k} \tag{17}$$

式中的 $f_k(x)=P_A^*(x)^{1-\beta_k}P_B^*(x)^{\beta_k}$，其中 P_A^*,P_B^* 即如同式 (8) 中所示。式中的 Z_k 为温度处在 β_k 的情况下的归一化常数。RTS 算法的核心便是通过多次迭代，同时更新所有温度过渡因子下的归一化

常数。每次迭代的过程中，会同时对温度过渡因子 β 和观测变量 v 进行抽样。本算法中对观测变量的采样方法与本节开头的方法相同，即式 (5),(6),(7)。这里主要叙述 β 的抽样方法。

当 $\beta \in \beta_k$ 是一个随机变量时，假定有一个先验概率 $r(\beta_k) = r_k$，那么 x, β_k 的联合分布则如下表示：

$$p(x, \beta_k) = p(x|\beta_k) r_k, \tag{18}$$

$$= \frac{f_k(x) r_k}{Z_k} \tag{19}$$

由于 Z_k 是我们需要求的，并不知道，但我们可以定义 $\hat{Z}_k$ 则

$$q(x, \beta_k) \propto \frac{f_k(x) r_k}{\hat{Z}_k} \tag{20}$$

由于联合分布已经知道，文献 [2] 定义出：

$$q(\beta_k|x) = \frac{f_k(x) r_k / \hat{Z}_k}{\sum_{l=1}^{K} f_l(x) r_l / \hat{Z}_l} \tag{21}$$

有了上述的概率分布，我们便可以通过采样出的 x_k，对下一次迭代使用的 β_s 进行抽样。具体来说，便是令 $q(\beta_1), q(\beta_2), ..., q(\beta_K)$ 依次按照其值，在 $[0,1]$ 的轴上占据一定位置。取一符合 $U(0,1)$ 的随机变量，该变量落入哪一个 $q(\beta_k)$ 的区间，则将该 β_k 作为下一次迭代使用的 β。而每次迭代中，则采用 $Z_k^{\hat{RTS}} = \hat{Z}_k \frac{r_1}{r_k} \frac{\hat{c_k}}{\hat{c_1}}$ 对归一化常数进行更新，其中 $\hat{c_k} = \frac{1}{N} \sum_{i=1}^{N} q(\beta_k|x^{(i)})$。

但是原始文章所采用的 $q(\beta_k|x; \hat{Z}_k)$ 收敛效果并不好，因此我将 $q(\beta_k|x; \hat{Z}_k)$ 定义时进行了改进。由文献 [1] 中指出，对于任意一个 $beta_k$，其未归一化的中间概率分布应当为式 (8) 所示。将其归一化之后，应当与 $q(\beta_k|x; \hat{Z}_k)$ 等价。而将 $q(\beta_k|x; \hat{Z}_k)$ 中的易于计算归一化常数的 RBM 的影响项从 $q(\beta_k|x; \hat{Z}_k)$ 中去除，即将 $q(\beta_k|x; \hat{Z}_k)$ 除以 $Z_A^{1-\beta_k}$。经过这样的处理后，收敛效果得到了较好的提升。

具体的算法步骤见下表 TABLE III：

而 SAMS 与 RTS 大体相似，算法步骤也基本相似，但是具体来说，根据文献 [3] 所示，其更新 Z_k 的方法以及抽样 β 的方法不一样。这里先指出 β 的抽样方法不同之处。如上所示，RTS 的 β 抽样方法，是全局抽样 (global jump)，即下一次抽样的 β_k 的位置可以是那组 β 的任意值，但是根据文献 [1] 所示，SAMS 的抽样方法是局部抽样 (local jump)，即每次抽取的下次的 β 值只能是本次 β 的相邻的 β 值。根据测试，global jump 使得归一化参数更新速度过快，容易造成过冲，导致预估的归一化常数过大，此处使用 local jump，能有效的控制收敛的速度在一个合适的范围。而 SAMS 的归一化常数更新的方法与 RTS 也不相同。RTS 通过两层循环，先更新 $\hat{c}$，再新 $\hat{Z}$，从而实现归一化常数的估计。而 SAMS 的更新方法则如下描述：

令 $\zeta = \ln Z$，则：

$$\zeta^{(t-\frac{1}{2})} = \zeta^{(t-1)} + \lambda\{\frac{q_1(\cdot; \hat{Z}^{(t-1)})}{r_1}, \frac{q_2(\cdot; \hat{Z}^{(t-1)})}{r_2}, ..., \frac{q_K(\cdot; \hat{Z}^{(t-1)})}{r_K}\}, \tag{22}$$

$$\zeta^{(t)} = \zeta^{(t-\frac{1}{2})} - \zeta_1^{(t-\frac{1}{2})}$$

其中 $\zeta_1^{(t-\frac{1}{2})}$ 为 $\zeta^{(t-\frac{1}{2})}$ 的第一项，而 λ 满足如下的关系，以控制收敛的速度：

$$\lambda = \begin{cases} \min(\pi_j, t^{-\beta}) & if\ t \le t_0 \\ \min\{\pi_j, (t - t_0 + t_0^{\beta})^{-1}\} & if\ t > t_0 \end{cases}$$

而此处的 q_k 仍采用了我在 RTS 中叙述过的改进，将 RBM A 的能量部分除去，以获得更好的收敛效果。

Algorithm 3: Rao-Blackwellized Tempered Sampling

初始化 $0 = \beta_0 < \beta_1 < ... < \beta_K = 1$
初始化 $\ln Z_k = 0, k = 2, ..., K$
初始化 $\hat{c_k} = 0, k = 2, ..., K$
while $maxabs(\hat{c_k} - \frac{1}{K}) > Tlimit$
  for $i = 1 : N$ do
    通过转移矩阵 (即式 (5)(6)(7))，以及目前的 β_k，采样 x。
    通过 $q(\beta_k|x; \hat{Z}_k)$ 采样下一次迭代的 β
    更新 $\hat{c_k} \leftarrow \hat{c_k} + \frac{1}{N} q(\beta_k|x)$
  end for
  更新 $\hat{Z_k}^{RTS} \leftarrow \hat{Z}_k \frac{r_1}{r_k} \frac{\hat{c_k}}{\hat{c_1}}$
end while

TABLE III: RTS Algorithm

IV. Analysis of Method & Results

由于程序的运行时间和机器的配置有着较大的关系，本文的仿真实验，都是在 CPU 为 i7-6700 @3.4GHz，内存 16GB 的电脑上运行的。

A. MCMC

为了验证 MCMC，我们进行估计二维高斯的相关系数的实验。

***Problem** 1:* 用 Metropolis-Hastings 算法，对下述二维高斯分布进行随机采样

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

使用生成的随机样本来估计二维高斯函数的相关系数，并与真值比较。

由于这里抽样的采样点为一个二维向量，且在 $\mathbb{R}^2$ 上式连续。因此 T 取有限大小会对转移产生误差。因此在本题中，没有构造出真实的转移矩阵 T，而是假定转移矩阵 T 为一个所有元素都相同且符合马尔科夫转移性质的矩阵，即每一行相加为 1。这种情况下，上述算法 Table I中的 $T(Y, X_n)$ 与 $T(X_n, Y)$ 相消，故只需要计算即将抽样的样本点的概率密度与前一个已经确认的样本点的概率密度作比值，在与一个在 $[0, 1]$ 均匀分布的随机变量作比较后，来判断是否转移成功即可。而由于 T 为完全均匀的转移矩阵，因此下一个抽样点的抽取应当完全随机抽取。这里为了提高抽取的命中率，我将 x_1 与 x_2 分别按照独立的两个一维高斯分布抽取随机数，一维高斯分布的均值与标准差均来自于二维随机分布中的均值与协方差矩阵。同样，这样的抽样方法，也使得最终的结果的相关系数与抽样点的个数无关，不会随着抽样点的数目增加而变得更加接近真值。

理论的相关系数真值应当为 0.5。使用 MATLAB 仿真相关系数分布在 $[0.484, 0.497]$ 之间。为评判 MCMC 的准确性，多次重复独立试验，每次试验均抽取 5 万个采样点。讲这些实验所得的相关系数计算所得结果的方差，结果如下：

运行次数	相关系数均值	结果方差
5	4.912	5.49×10^{-6}
10	4.902	2.20×10^{-5}
20	4.899	3.69×10^{-5}
50	4.934	1.74×10^{-5}
100	4.923	1.67×10^{-5}

首先分析本方法的准确性，由上述结果可见，多次运行本 MCMC 方法的方差非常小，为 10^{-5} 量级，而相关系数为 10^{-1} 量级。因此可以认为该 MCMC 抽样方法准确性相当高，几乎没有误差，同时，其抽样出的样本的相关系数与真值也非常接近，最大误差在 0.1 左右。

其次是对算法效率的分析。运行一次生成 5 万个抽样点的 MATLAB 程序需要大概 1s 到 2s 的时间，主要时间开销在抽样主程序上。其次为了实现该算法，完成了一个计算二维正太分布概率密度的函数，该函数调用较为高效，3s 的时间可以调用 300 万次。总体程序的效率较高。

B. AIS

在本实验实现 AIS 方法时，划分的温度过渡因子越细致，即 β 的集越大，所得到的结果越加精细，估计值更加贴近真值，但相对的运算的代价会变大，计算速度会变慢。具体的算法，在前述部分已经得到了讲解，在此对算法进行分析。

首先分析程序运行效率。为了提高程序效率，令 $\omega_k = \frac{Z_{k+1}}{Z_k} = \frac{1}{M}\sum_{i=1}^{M}\frac{P^*_{k+1}(x^{(i)})}{P^*_{k+1}(x^{(i)})}$ 中的 $M = 1$，既可以减少抽样次数，提高程序效率，也可以获得较好的结果。在尽可能获得准确的结果的情况下，令 β 的取值更加精细。根据实验结果来看，β 的精细程度需要根据 RBM 的隐变量数目来动态决定。根据实验结果，对于 10 个隐变量取 β 数组的元素数为 3 万，20 个隐变量模型的为 30 万，100,500 个隐变量的模型均取为 6 万。而根据 MATLAB 本身的运算特性，尽可能的将所有的循环改写成为矩阵运算后，程序的运行效率得到了大大的提高。经过 MATLAB 仿真测试，h10 的模型，运行时间为 10s；h20 的模型运行时间为 60s；h100 的模型运行时间为 100s；h500 的模型运行时间为 264s。而该函数主要耗费效率的部分是抽样算法，抽出下一个观测变量的运算，以及一些求和运算，这些运算都无法避免或者使用矩阵乘法提高运算效率。

再分析程序的准确性，将 AIS 函数循环 20 次，计算归一化常数的方差值，所得的结果如下表 TABLE 和下图 Fig. 所示：

Model	Normalized Coef	Result Var
h10	225.89	0.1939
h20	218.23	19.4639
h100	348.5343	0.1089
h500	460.9204	0.0962

通过上述结果可以看出，除了 h20 的模型之外，AIS 的结果是非常稳定的，每次产生的归一化常数的值也较为相似，方差非常小。而 h20 的结果方差较大，可能的原因在于训练模型时，获得参数结果不太好，导致了计算归一化常数的值得结果方差较大。可以看出，AIS 算法除了花费的时间非常长之外，获得结果的准确性非常高。

C. TAP

而对 TAP 算法进行分析时，可以发现由于 TAP 算法是一种近似算法。近似造成其估计的归一化常数值与真值之间可能会存在一个恒定的误差，多次运行试验时得到的一组估计归一化常数的方差也非常小。根据原始公式，显然结果方差与与真值的误差会随着近似程度的增加，会逐渐变小。

本文实现了 TAP 的二阶近似算法 TAP2 与三阶近似算法 TAP3。两个算法的运行性能结果如下表所示：

Model	TAP2 Var	TAP3 Var	Norm Coef
h10	0.0213	0.0001	207.2
h20	1.0136	0.0001	221.1
h100	7.35×10^{-11}	0.0036	335.7857
h500	1.04×10^{-4}	0.0002	432.6158

通过上述结果可以发现，TAP 算法获得的归一化常数与真值之间有一定误差，但是每一次 TAP 得到的结果的方差较小，因此 TAP 算法是一个稳定的算法，但是得到的结果偏差较远。TAP 算法的时间开销较小，每次调用的时间都在 1s 以内，这和他不需要逐步提高温度因子来逼近最终结果的玻尔兹曼机有关。通过快速迭代，TAP 可以较快的得到结果下图 Fig.3为重复试验调用 TAP3 函数 50 次时，每次的结果图，Fig.4为一次调用 TAP3 中，其当前自由能能量与上次自由能能量之差的结果。通过该图可以发现，TAP3 算法能够以较快的速度收敛，但是估计出的归一化常数，与真实值有着一定的差距。

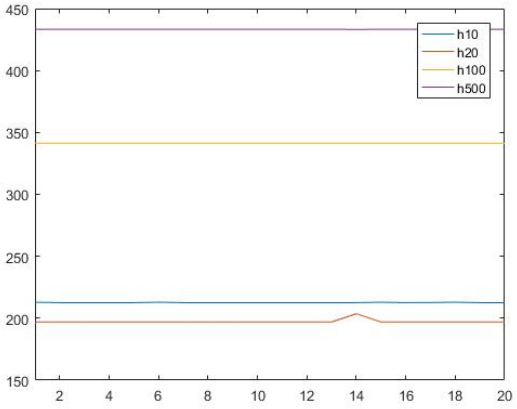

Fig. 3: 重复试验 50 次结果

D. RTS

RTS 的结果收敛性较差，波动性也较高。这个和 RTS 算法中的易于计算归一化常数的 RBM 参数设置有关。最开始，作者尝试使用全为 0 的参数设置为 RBM A 的参数。然而，这种情况下 RTS 算法并不收敛，其估算出的误差呈现周期性的波动。因此，我尝试改变 RBM A 的一个或两个参数。由文献 [1] 中指出，当 W_A 取为全 0 矩阵时，

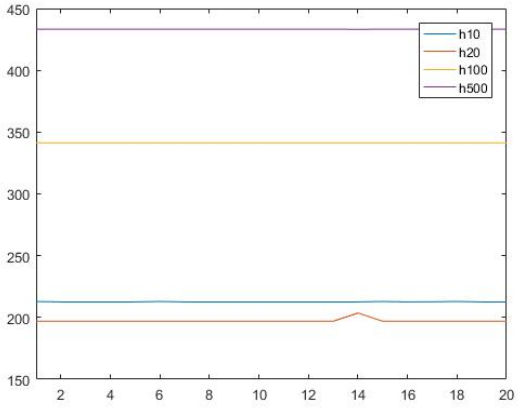

Fig. 4: TAP3 收敛效果

有 $P^A(\mathbf{v}) = \prod_i p_A(v_i) = \prod \frac{1}{1+e^{-b_i}}$。由此可以解出 $b_i = \ln \frac{P_i^A}{1-P_i^A}$。因此，在此将 b^A 设定为如下的值。令 $b^A = \ln \frac{f}{1-f}$，其中 f 为 1×784 的向量，其计算方法如下所述。读取训练 RBM 参数的训练集，应当为 $N \times 784$ 的数据集，统计 784 个变量的 N 组数据中 1 出现的频率，即为所需要的 f。而为避免出现 inf 与 $-\inf$ 的情况，可以将 784 个 1 的频数都增加一个较小的值如 5。通过该方法，获得不含有 inf 与 $-\inf$ 的 b^A。将其带入运算后，收敛性得到了较好的提高。以 h10 为例，其收敛速度如下图 Fig. 5所示

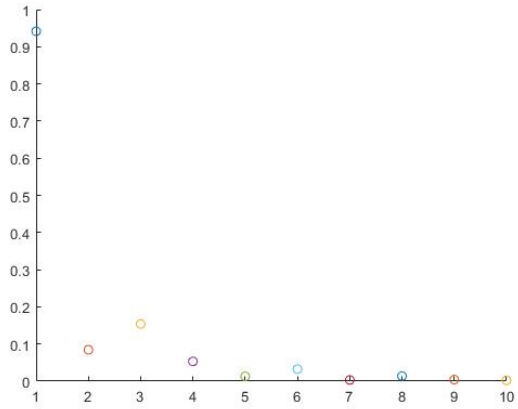

Fig. 5: h10 模型收敛速度

RTS 的运行性能如下表所示：

Model	RTS Var	Norm Coef
h10	0.2963	226.1005
h20	2.1839	220.6264
h100	12.3514	347.7237
h500	0.2188	463.1204

仿真过程中发现，若不能快速达到收敛条件，RTS 容易陷入循环中，最终使得算法的时间开销较长，因此设置合适的阈值与 RBM_A 的参数，对于 RTS 得到快速正确的结果非常的重要。而综合来看，因为作者本人没有花费足够的时间去寻找所有合适的参数，因此对于隐变量数目较少的模型，收敛效果较好，对于隐变量数目较多的模型，最后难以收敛。但是一旦能够成功地脱离循环，得到最终的归一化结果，可以发现，RTS 迭代得到的结果距离真值精确程度与 AIS 一致。因此对于 RTS 算法而言，设计合适的阈值与迭代参数是最为重要的设计部分。

E. SAMS

SAMS 的估计结果相较 AIS 偏离不远，且其运行速度比 AIS 要快，收敛性也比 RTS 要好。因此也是一种比较优质的估计归一化参数的方法。SAMS 算法的结果如下表所示：

Model	SAMS Var	Norm Coef
h10	0.7467	226.45
h20	0.2188	210.17
h100	301.54	329.79
h500	486.2	512.70

通过上述结果可以发现，SAMS 的结果相对而言也比较稳定。与真值的误差随着隐变量的个数增加而变大。SAMS 时间开销较小，h500 的模型在 30s 内可以计算完成。但是，随着隐变量数量快速增加，SAMS 的准确性也会随之降低，方差也随之增加，算法的稳定性下降，此时应当使用新的参数进行。

而经过上述四个算法得到了归一化常数之后，我们可以对实验样本进行似然值的估计。似然值估计的方法，通过下式进行：

$$\begin{aligned} P(v,\theta) &= \frac{1}{Z(\theta)} \sum_{h} e^{-E(v,h;\theta)} \\ &= \frac{1}{Z(\theta)} e^{b^\top v} \prod_{j=1}^{F} \left(1 + e^{a_j + \sum_{i=1}^{D} W_{ij} v_i}\right) \end{aligned} \quad (23)$$

通过该表达式，可以将总似然值计算得出。对于每一个测试集，$\frac{1}{Z(\theta)}$ 后面的部分是恒定的，只有归一化常数会对最后的结果产生影响。而最似然估计值越大，意味着测试集与模型匹配的越好。

V. Conclusion

综上所述，所有采样方法中，AIS 方法为精确度最高的方法，但是其时间复杂度太高，且随着模型隐变量数目的增加而增长，若需要精确地估计归一化常数，则使用此方法较为合适。而 RTS 方法是收敛性最差的算法，需要对初始的参数做精巧的设计，才有可能使得 RTS 方法以较快的速度收敛。TAP 方法为所有方法中速度最快的方法，低时间复杂度的代价便是归一化常数估计值与真值之间的误差较大。若对归一化常数不需要较为精确地结果，且需要处理大量的模型数据，使用此方法较为合适。而 SAMS 的相对于前述的三种算法而言，获得了折中的结果，即速度相比 AIS 获得了较大的提高，结果相对于 TAP 更为准确，而且易于收敛。该方法适合用于快速对归一化参数进行近似的估计。

VI. Acknowledgement

感谢随机过程课程的欧智坚老师与戴音培助教对我学习过程中的困惑的解释。

感谢无 47 班余东翰同学，无 48 班黄佳新同学，无 48 班徐瑞同学等对我学习 RBM 归一化参数的估计算法的帮助！

References

[1] R. Salakhutdinov, “Learning deep generative models,” Ph.D. dissertation, University of Toronto, 2009.

[2] D. Carlson, P. Stinson, A. Pakman, and L. Paninski, “Partition functions from rao-blackwellized tempered sampling,” *arXiv preprint arXiv:1603.01912*, 2016.

[3] Z. Tan, “Optimally adjusted mixture sampling and locally weighted histogram analysis,” *Journal of Computational and Graphical Statistics*, no. just-accepted, 2015.

[4] M. Gabrié, E. W. Tramel, and F. Krzakala, “Training restricted boltzmann machine via the thouless-anderson-palmer free energy,” in *Advances in Neural Information Processing Systems*, 2015, pp. 640–648.