arXiv:2302.00564v2 [cs.LG] 1 Jun 2023

Auto概率编程中的自动边缘化马尔可夫链蒙特卡罗

Jinlin Lai [1] **Javier Burroni** [1] **Hui Guan** [1] **Daniel Sheldon** [1]

摘要

哈密顿蒙特卡罗（HMC）是一种强大的算法，可用于从贝叶斯模型中抽取潜变量。概率编程语言（PPL）的出现，使用户不必编写推断算法，而能专注于建模。然而，许多模型难以直接通过 HMC 求解，通常需要采用模型重参数化等技巧。我们的研究受到这样一个事实的启发：许多此类模型可以通过边缘化得到简化。我们提出，在从概率编程语言中提取的图模型中，将自动边缘化作为使用 HMC 进行抽样过程的一部分，从而显著改善对真实世界层次模型的抽样效果。

1. 引言

概率编程语言（PPL）有望实现贝叶斯推理的自动化。用户指定一个概率模型并提供数据，PPL 随即自动执行推断，以近似后验分布。这样，用户无需具备概率推断方面的高度专业知识，也能获得科学见解（van de Meent 等人，2018年）。借助 BUGS（Lunn 等人，2009年）、JAGS（Hornik 等人，2003年）和 Stan（Carpenter 等人，2017年）等工具，这一范式对应用科学产生了巨大影响；与此同时，计算机科学领域也开展了大量研究，以推进概率编程语言的理论基础（Goodman 等人，2008；Wood 等人，2014；Goodman 和 Stuhlmüller，2014；Cusumano-Towner 等人，2019；Minka 等人，2018）。

概率编程语言在许多方面各不相同，包括它们能够表示的分布以及主要采用的推断方法。我们聚焦于一种在实践中影响深远的情形：程序对应一个图模型，并被编译为可微对数密度函数，再通过哈密顿蒙特卡罗（HMC）的一种变体进行推断（Duane 等，1987；Neal，1996）。尽管概率编程语言前景广阔，但用户与其中的推断之间的界限往往并不清晰。模型可能有多种编写方式，而推断性能又严重依赖于具体选择，因此用户仍需要具备专门知识。本文主要关注的一个问题是：通常可以从模型中解析地边缘化某些变量，使推断方法在更小的模型上运行，从而显著提升性能。例如，坍缩吉布斯采样中就采用了这一思路（Liu，1994年）。然而，这类重新表述几乎总是需要手动完成，给用户带来了沉重负担（示例见图 1 和第 2 节）。

我们开发了一种方法，可在使用哈密顿蒙特卡罗进行推断时，自动边缘化用户指定的概率程序中的变量。该方法首先将概率程序编译为图模型。尽管大多数基于哈密顿蒙特卡洛的概率编程语言会直接编译为对数密度，但我们利用 JAX 的程序追踪功能（Bradbury 等人，2018年），提取以 NumPyro 编写的程序的图模型表示。在图模型中，我们识别出可反转边的局部共轭关系。随后，我们对图模型进行操作，创建未观测叶变量，以便对其进行边缘化。我们在简化模型上运行哈密顿蒙特卡罗，然后根据哈密顿蒙特卡罗抽取的变量，通过直接采样恢复边缘化变量。对于不存在共轭关系的模型，我们的方法等同于标准哈密顿蒙特卡洛。重要的是，用户与概率编程语言之间的接口保持不变。

实验表明，我们的方法能够显著提高从层次部分汇聚模型和层次线性回归模型中获得的样本的有效性，并且在两种方法都适用的模型中，表现显著优于模型重参数化（Betancourt & Girolami，2015年）。我们的实现仅支持标量和逐元素数组运算，可能需要用户输入以避免 JAX 编译时间过长，不过这些限制并非根本性的。我们的代码可在 https://github.com/lll6924/automatically-marginalized-MCMC 获取。

1.1. 范围及与先前工作的关系

如上所述，我们聚焦于一类受限的概率程序，这类程序能够提取出图模型

[1]马萨诸塞大学阿默斯特分校。通信请联系：Jinlin Lai <jinlinlai@cs.umass.edu>。

论文集，发表于 40th 国际机器学习会议，美国夏威夷州檀香山。PMLR 202，2023年。版权所有 © 2023 作者。

表示。具体而言，我们聚焦于 (1)生成式 概率编程语言，[2] 其中模型通过编写采样过程来表达；以及 (2) 对应于有向图模型的程序，也就是说，在每次程序执行中，随机变量都按照固定顺序根据条件分布生成（不包含随机分支或无界循环）。这涵盖了使用生成式概率编程语言编写的大多数应用统计模型，例如 Pyro（Bingham 等人，2018年）、NumPyro（Phan 等人，2019年）、PyMC（Patil 等人，2010年）、Edward（Tran 等人，2017年）和 TensorFlow Probability（Piponi 等人，2020年）。这并不直接涵盖 Stan 程序，因为 Stan 程序并非总会指定采样过程，不过大多数程序都可以转换为指定采样过程的形式（Baudart 等人，2021年）。

我们的工作建立在自动边缘化方面的先前研究（Hoffman 等人，2018年）之上，并与针对基于求值的概率编程语言自动对粒子滤波器进行拉奥-布莱克韦尔化的研究共享技术基础（Murray 等人，2018年；Atkinson 等人，2022年）。我们工作的关键区别在于，利用图模型结构，以完全自动化的方式重新表述模型，从而使用哈密顿蒙特卡罗进行推断。Autoconj（Hoffman 等人，2018年）通过在对数密度函数中识别共轭模式，实现了积分消去潜变量这一相同的数学目标。它提供了用于边缘化单个变量和计算完全条件分布的基本操作，但并未提供重新表述模型的整体流程；用户必须自行选择如何应用这些基本操作来转换模型或执行推断。借助我们的流程，用户只需提供模型，端到端算法（算法 1）便会利用图结构，以完全自动化的方式边缘化变量。Murray 等人（2018年）和 Atkinson 等人（2022年）在基于粒子滤波的推断采样过程中执行局部模型转换，这与我们的方法相似，并适用于非常通用的概率程序。相比之下，我们专注于对应图模型的程序，因此能够在推断之前，基于完全符号化的表示执行边缘化，并得到一种普遍适用的模型重新表述方式，可用于多种推断算法。

第 4 节对相关工作进行了更详细的讨论。

2. 动机示例

我们首先介绍几个示例模型，在这些模型中，边缘化能够显著提升基于哈密顿蒙特卡罗的推断效果。

八所学校模型（Gelman 等，1995年）是概率编程语言（Gorinova，2022年）和重参数化（Papaspiliopoulos 等，2007年）中的一个重要示范模型。它是一个层次模型，用于研究教练辅导对八所学校学生学术能力倾向测验成绩的影响。一个概率

[2]定义请参见 Baudart 等人（2021）。

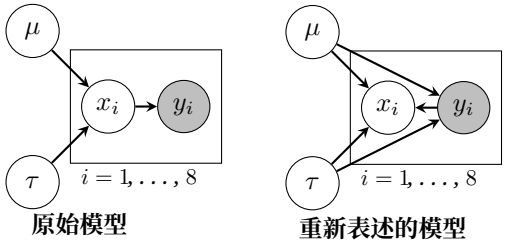

图 *1.* 原始八所学校模型与重新表述的八所学校模型的图模型。我们用板表示不同分支的共享子结构。灰色变量表示观测值。

采用类似 NumPyro 语法的八校模型程序如下：

```
def eight_schools(sigma, y):
    mu = sample(normal(0, 5))
    tau = sample(half_cauchy(5))
    with plate(8):
        x = sample(normal(mu, tau))
        observe(normal(x, sigma), y)
```

从数学上说，该模型为

$$\mu \sim \mathcal{N}(0, 5^2),\ \ \tau \sim \text{HalfCauchy}(5),$$
$$x_i \sim \mathcal{N}(\mu, \tau^2),\ y_i \sim \mathcal{N}(x_i, \sigma_i^2),$$

其中，$i \in \{1, \ldots, 8\}$ 和 $(\sigma_{1:8}, y_{1:8})$ 是给定的数据。我们希望对所有潜变量进行推理：μ、τ 和 $x_{1:8}$。概率编程语言会将模型代码编译为对数联合密度 $\log p(\mu, \tau, x_{1:8}, y_{1:8})$，然后对潜变量 μ、τ 和 $x_{1:8}$ 运行哈密顿蒙特卡罗。[3] 然而，还有另一个具有相同联合密度的模型：

$$\mu \sim \mathcal{N}(0, 5^2), \qquad \tau \sim \text{HalfCauchy}(5),$$
$$y_i \sim \mathcal{N}(\mu, \tau^2 + \sigma_i^2),\ x_i \sim \mathcal{N}\left(\frac{y_i\tau^2 + \mu\sigma_i^2}{\tau^2 + \sigma_i^2}, \frac{\tau^2\sigma_i^2}{\tau^2 + \sigma_i^2}\right).$$

两个模型均以图模型的形式显示在图 1 中：它们具有不同的因果解释，但联合分布相同，因此在执行推断时是等价的。重要的是，在重新表述的模型中，我们不再需要对所有潜变量运行哈密顿蒙特卡罗。由于只有 $y_{1:8}$ 被观测到，因此可以将 $x_{1:8}$ 边缘化，以得到简化模型 $p(\mu, \tau, y_{1:8}) = p(\mu)p(\tau)\prod_{i=1}^{8} p(y_i \mid \mu, \tau)$。我们可以先在简化模型上运行哈密顿蒙特卡罗，对 μ 和 τ 进行采样，然后给定 μ 和 τ，直接从 $p(x_{1:8} \mid \mu, \tau, y_{1:8})$ 中对 $x_{1:8}$ 进行采样。采用这一策略，哈密顿蒙特卡罗只需对 2 个变量进行采样，而不是 10 个，从而显著加快推断速度。

使我们能够转换模型的原理是共轭性（参见 Murphy（2012）第 3.3.2 节）。在贝叶斯模型 $p(x, y) = p(x)p(y \mid x)$ 中，如果后验 $p(x \mid y)$ 与先验 $p(x)$ 属于同一

[3]在实践中，潜变量会经过变换，使其具有实数支持集（Kucukelbir 等，2017年）。

参数族，那么对于所有 y，后验都与 $p(x)$ 属于同一参数族。作为我们的操作性定义，我们假设先验和似然的参数族都具有易于处理的密度函数和抽样过程，并且后验参数可以用 y 表示为解析公式。在这些假设下，也可以从边缘分布 $p(y)$ 中抽样，并高效地计算其密度。共轭性严格来说是分布族的一种性质，但在上下文含义清楚时，我们也会说“x 与 y共轭”。

在八所学校模型中，给定 μ 和τ时， x_i 与 y_i 共轭，因此可以解析地表示重新表述的模型中的分布$p(x_i \mid \mu, \tau, y_i)$和 $p(y_i \mid \mu, \tau)$，并确保它们具有易于处理的密度和抽样器。

分层线性回归。八所学校模型非常简单，但重新表述它已经需要用户付出努力。为强调重新表述更大模型的复杂性，我们在图2中展示了电力公司模型的简化版本（Gelman & Hill，2006）。完整模型见第 5.2 节。观测变量为 y_1和 y_2。可以推测，该模型能够重新表述，因为在给定 σ的条件下，所有变量均服从正态分布，且其均值是其他变量的仿射函数。然而，计算过程很复杂：图2右侧展示了重新表述模型的一部分。在这个版本中，运行哈密顿蒙特卡罗，从给定 y_1 和 y_2的条件下对 μ_a 和 σ 进行抽样（此处未展示其分布）。然后，通过从所示分布中抽样，在给定 μ_a, σ, y_1, y_2 的条件下重建 a、 b_1和 b_2 。将变量数量从5个减少到2个，可以加快哈密顿蒙特卡罗推断。然而，用户要推导出新模型极其繁琐，而且新模型已不再对应最初设想的数据生成过程。我们希望将这一过程自动化，使用户只需编写原始模型，由我们的框架对其进行重新表述。

3. 自动边缘化马尔可夫链蒙特卡罗

给定用户编写的程序，我们的方法将构建一个图模型，然后将其转换为重新表述的模型，使马尔可夫链蒙特卡罗需要抽样的变量更少。关键操作是反转 某些边（基于共轭性），以创建可边缘化的未观测叶节点。例如，在图 1 的八校模型中， x_i 到 y_i 的边被反转（其副作用是创建从 μ 和 τ 指向 y_i的边），此后 x_i 成为叶节点。本节首先在假设存在合适的图模型表示的前提下介绍该算法。随后，在第 3.7 节中，我们将说明在实现中如何使用 JAX 和 NumPyro 获得这种表示。

[4]要进行采样，抽取 $x \sim p(x)$ $y \sim p(y \mid x)$ 并忽略 x；对于密度，使用 $p(y) = {}^{p}\,\cancel{(x_0)\ p\ (y \mid x_0)\ p}\,(x_0 \mid y)$ </style}作为参考值 x_0。

3.1. 图模型表示

假设有 M个随机变量 x_1、 x_2、……、 x_M，其中 x_i 属于定义域 $\mathcal{X}_i$。完整定义域为$\mathcal{X} = \prod_i \mathcal{X}_i$。对于一组索引 A，我们将 $\mathbf{x}_A = (x_i)_{i \in A}$记作 $\mathcal{X}_A = \prod_{i \in A} \mathcal{X}_i$。图模型 G 通过为每个节点指定一个分布族，以及从父节点到参数的映射来定义。具体而言，对于节点 i，令 D_i 表示其所属的分布族（从有限个选项中选取，例如“正态分布”“贝塔分布”等），令 pa(i) $\subseteq \{1, \ldots, M\}$ 表示其父节点，并令 f_i：$\mathcal{X}_{\mathrm{pa}(i)} \to \Theta_i$为一个映射，使得 x_i 服从分布 $D_i(\theta_i)$，其参数为 $\theta_i = f_i(\mathbf{x}^{\mathrm{pa}(i)})$。例如，若 $x_2 \sim \mathcal{N}(x_1, 1)$，则 $D_2 =$ 为“正态分布”，pa(2) $= \{1\}$，且 $f_2(x_1) = (x_1, 1)$。此外，假设每个分布族都提供密度函数和采样程序。令 $p_i(x_i \mid \theta_i)$为节点 i 的密度函数，令 $h_i(u \mid \theta_i)$为采样函数，该函数将随机种子 u 映射为从 $D_i(\theta_i)$中抽取的样本。父节点关系必须无环。初始时，节点将按拓扑顺序排列，使得 pa(i) $\subseteq \{1, \ldots, i-1\}$。我们的算法会操作图模型以保持无环性，但不会保留节点按拓扑顺序编号这一不变量。在示例模型中，我们采用层次模型的标准记法，使用 μ、 τ、 x_i、 y_i等变量名；在这些情况下，如何映射到通用随机变量序列 x_1、……、 x_M、父节点关系以及分布族，均可从上下文中明确得知。

采用这种表示，给定所有变量的具体值后，在节点按拓扑排序的假设下，对数密度可以很容易地计算为 $\sum_{i=1}^{M} \log\ p_i\big(x_i \mid f_i(\mathbf{x}^{\mathrm{pa}(i)})\big)$。生成联合样本的过程类似：遍历各个节点并采样 $x_i = h_i\big(u \mid f_i(\mathbf{x}^{\mathrm{pa}(i)})\big)$。我们方法的一个关键思想是，将对数密度计算分解为针对每个随机变量的一系列条件函数 f_i，从而可以操纵条件分布，实现自动边缘化。

3.2. 计算图表示

我们用于转换图模型的操作需要检查并操纵函数 $f_i(\mathbf{x}^{\mathrm{pa}(i)})$，该函数将父节点映射到分布参数。例如，在图2所示的电力公司模型中，我们需要从符号表达式 $y_2 \sim \mathcal{N}(a + b_2 t_2,\ \sigma^2)$ 中检测出均值参数是 b_2 的仿射函数，这是反转边 $b_2 \to y_2$ 所必需的。同样，我们必须操纵符号表达式，以获得类似于重新表述的模型中的表达式。为此，我们假设函数表示为计算图。

考虑任意函数 $f(x_{i_1},\ x_{i_2},\ \ldots\ldots,\ x_{i_k})$，其中$i_1$，……，$i_k \in \{1, \ldots, M\}$。我们假设 f 的计算图以一系列 N_f 原语操作的形式指定，每个操作都会写入一个值（Griewank & Walther，2008），

原始模型 **重新表述的模型中 a、 b_1和 b_2的分布**

$$\log \sigma \sim \mathcal{N}(0,\ 1),$$
$$\mu_a \sim \mathcal{N}(0,\ 1),$$
$$a \sim \mathcal{N}(100\mu_a,\ 1),$$
$$b_{1:2} \sim \mathcal{N}(0, 100^2),$$
$$y_i \sim \mathcal{N}(a + b_i t_i, \sigma^2)。$$

$$a \sim \mathcal{N}\left(\frac{100\mu_a\sigma^2 + y_1 + y_2 - b_1 t_1 - b_2 t_2}{2 + \sigma^2}, \frac{\sigma^2}{2+\sigma^2}\right),$$
$$b_1 \sim \mathcal{N}\left(\frac{100^2 t_1 (y_1 - 100\mu_a)}{1 + 100^2 t_1^2 + \sigma^2}, \frac{100^2 + 100^2\sigma^2}{1 + 100^2 t_1^2 + \sigma^2}\right),$$
$$b_2 \sim \mathcal{N}\left(\frac{100^2 t_2\ (y_2 + \sigma^2 y_2 - y_1 - 100\sigma^2\mu_a + b_1 t_1)}{2 * 100^2\sigma^2 + 100^2\sigma^4}, \frac{100^2 t_2^2 + 100^2\sigma^2 t_2^2 + 2\sigma^2 + \sigma^4}{100^2 t_2^2 + 100^2\sigma^2 t_2^2 + 2\sigma^2 + \sigma^4}\right).$$

图*2*。原始模型 以及简化电力公司模型的重新表述模型。(t_1, y_1), (t_2, y_2)作为数据给出 .

这与我们可以从 JAX 获得的*JAX* 表达式（*Jaxpr*）表示形式类似。具体而言，值序列 w_1, w_2, w_3, ..., w_{k+N_f}按如下方式计算：（1）前 k 个值是函数的输入，即 $w_j = x_{i_j}$, 其中 $j = 1$ 到 k; （2）之后的每个值都由前面的值计算得出，形式为 $w_j = \phi_j(\mathbf{w}^{\text{pred}(j)})$, 其中 ϕ_j 是作用于值 $\mathbf{w}^{\text{pred}(j)}$ 的原语操作（例如“ADD”“乘法”“SQUARE”），而 pred(*j*) $\subseteq \{1, \ldots, j-1\}$ 是 j 的前驱集合。前驱关系为计算图中的变量定义了一个有向无环图。

我们还需要以算法方式操作计算图。在正文中，我们将用以下符号表示这些操作。假设 $f(\mathbf{x}_A)$ 和 $g(\mathbf{x}_B)$ 是由计算图表示的两个函数，其输入变量集合可能存在重叠。我们使用诸如 $f * g$ 或 $f + g$ 这样的表达式，表示代表该符号表达式的新计算图。例如， $f + g$ 的计算图包含输入变量 $\mathbf{x}_{A\cup B}$，由 f 和 g 的计算图，以及一个用于执行最后加法的额外节点（原语操作）组成。

3.3. 边缘化未观测叶节点

作为对图模型的一种实用变换，我们先考虑存在未观测叶节点时如何改进哈密顿蒙特卡罗。不失一般性，假设该叶节点编号为 M。那么，我们可以将联合分布分解为

$$p(\mathbf{x}_{1:M}) = p(\mathbf{x}_{1:M-1})p(x_M \mid \mathbf{x}_{1:M-1}), \qquad (1)$$

并在边缘化模型 $p(\mathbf{x}_{1:M-1})$ 上运行哈密顿蒙特卡罗，然后通过执行 $h_M(\cdot \mid f_M(\mathbf{x}^{\text{pa}(M)}))$，直接从 $p(x_M \mid \mathbf{x}_{1:M-1})$ 中抽取 x_M。重要的是，边缘分布 $p(\mathbf{x}_{1:M-1}) = \prod_{i=1}^{M-1} p_i(x_i \mid f_i(\mathbf{x}^{\text{pa}(i)}))$ 只是删除叶节点后的原始图模型，因此易于处理。更一般地说，这一论证很容易扩展：反复剥除叶节点，以边缘化所有没有通向观测变量的路径的变量，从而进行哈密顿蒙特卡罗；然后从图模型中通过前向采样 （有时也称为祖先采样）重建这些变量（Koller 和 Friedman，2009年）。

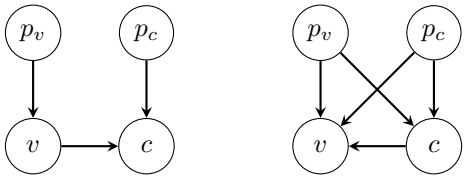

图*3*。反转边 $v \to c$。节点 $p_v \in \text{pa}(v)$ 和 $p_c \in \text{pa}(c) \setminus \{v\}$ 是用于演示这一变换的代表性父节点。左：反转 $v \to c$ 之前的图模型。右：反转 $v \to c$ 之后的图模型。

3.4. 通过边反转对非叶节点进行边缘化

八校模型和电力公司模型等生成模型的原始形式中没有未观测的叶节点，因为这些节点在数据生成过程中不会发挥任何作用。相反，我们的目标是通过一系列边反转来变换模型，从而创建未观测的叶节点。每次边反转都会保留图模型的联合分布，因此不会影响推断。但它不会保留数据生成过程的因果语义（推断并不要求保留这一点），所以变换后的模型出现未观测的叶节点是合理的。

反转单条边。 图3展示了如何反转单条父子边 $v \to c$。从 v 到 c不能存在其他路径；否则反转这条边会形成一个环。在这个例子中，不存在其他路径，因为 v 只有一个子节点。我们将 x_v 和 x_c 的“局部分布”定义为这两个变量在给定其父节点时的条件分布之积，形式为 $p(x_v \mid \cdots)p(x_c \mid x_v, \cdots)$。如果这些分布满足适当的共轭关系，我们就可以推导出形式为$p(x_c \mid \cdots)p(x_v \mid x_c, \cdots)$的替换因子，从而在保留局部分布的同时“反转” $v \to c$ 边。具体而言，该操作如下：

定义 *1*（边反转）。假设 G 是一个图模型，其中节点 v 是 c的父节点，且从 v 到 c不存在其他路径。反转边 $v \to c$ 会将因子 $p(x_v \mid x^{\text{pa}(v)})p(x_c \mid x_v, x^{\text{pa}(c)\setminus\{v\}})$ 替换为$p(x_c \mid x_U)p(x_v \mid x_c, x_U)$，并在 $U = pa(v) \cup pa(c) \setminus \{v\}$的情况下，更新父节点集合，使 $pa'(c) = U$、$pa'(v) = U \cup \{c\}$。

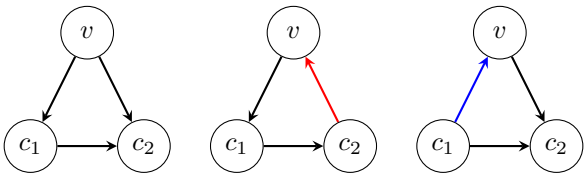

图*4*。反转多个出边的挑战。左：局部结构。中：反转 $v \to c_2$后形成了一个环，这是无效的。右：反转 $v \to c_1$后模型仍然有效，此时也可以反转 $v \to c_2$ 。

容易证明，边反转得到的图模型与原模型具有相同的联合分布。证明见附录A。要理解这一操作的作用，请观察图3：反转 $v \to c$后，节点 v 会变成叶节点，因此可以对其进行边缘化。原则上，任何边都可以反转，但只有在能够推导出替换因子时，这一操作才是可处理的。若分布是局部共轭的，我们就能做到这一点：

定义2（局部共轭）。 设 G 是一个图模型，其中节点 v 是 c的父节点。若$\hat{}p(x_v) := p(x_v \,|\mathbf{x}^{\mathrm{pa}(v)})$ 与$\hat{}p(x_c \,|\, x_v)$ $:=p(x_c \,|\, x_v, \mathbf{x}^{\mathrm{pa}(c)\setminus\{v\}})$ 对所有 $\mathbf{x}^{\mathrm{pa}(v)}$ 和 $\mathbf{x}^{\mathrm{pa}(c)\setminus\{v\}}$ 的取值都构成共轭对，则称 x_v 的分布与x_c 的分布局部共轭。

例如，在模型 $x_1 \sim \mathcal{N}(0,\ 1)$、 $x_2 \sim \mathcal{N}(x_1,\ 1)$、$x_3 \sim \mathcal{N}(x_1 x_2,\ 1)$中，随机变量 x_1 与x_2共轭；当 x_2固定时，它也与 x_3 共轭。因此，它与 x_2 和 x_3都局部共轭。第3.6节将介绍如何利用我们的图模型表示以及特定的分布族共轭对进行边反转。局部共轭不应与相关概念条件共轭（Murphy，2012）混淆。条件共轭指完全条件分布$p(x_v \,|\mathbf{x}^{\{1,\ldots,M\}\setminus\{v\}})$与生成分布 $p(x_v \,|\mathbf{x}^{\mathrm{pa}(v)})$属于同一分布族。条件共轭描述单个变量与完全分布之间的关系，而局部共轭则是一种成对关系。

通过反转节点的所有出边来创建叶节点。接下来，我们考虑是否能够通过反转任意节点 v 的所有出边，将其转换为叶节点。假设 v有 H个子节点 c_1、……、c_H。对于任意子节点 c，即使 x_v 与 x_c 局部共轭，反转 $v \to c$ 也未必安全，因为可能存在从 v 到c的另一条路径，从而形成一个环。图4给出了一个例子。不过，如果在 G的拓扑排序中， c是 c_1、……、c_H中最靠前的节点，那么就不可能存在其他 $v \to c$ 路径，因此反转 $v \to c$是安全的。此外，反转这条边后， v在拓扑排序中的位置会移到 c 之后、其他子节点之前，而其他子节点之间的相对顺序保持不变。这样，另一个子节点就会成为拓扑排序中最靠前的节点。因此，如果能够将 v 转换为叶节点，我们就应当反转这些边

算法 1边缘化并恢复未观测节点

函数边缘化 (G)
  初始化栈 S，并对节点排序，使其编号按拓扑序排列
  对于每个未观测节点 v，按降序**执行**
    如果对所有子节点 c**满足**共轭(G, v, c)，**则**
      // 边缘化 v
      对每个子节点 c 按升序**执行**
        $G =$ 反转$(G,\ v,\ c)$
      从 G 中移除 v
      将 v 添加到 S 的顶部
  返回 G , S
函数 恢复 (S, x_v 针对未移除的节点 v)
  当 S 不为空时 **执行**
    $v =$弹出(S)
    给定 $x_{pa(v)}$，对 x_v 采样
  返回 $x_{1:M}$

从 v 到它的每个子节点，并按照它们的拓扑排序依次进行。这一推理总结为以下定理，该定理的证明见附录 B。

定理 1. 设 G 为一个图模型，其中节点 v 有子节点 c_1、……、 c_H。如果 x_v 与x_{c_1}、……、 x_{c_H}中的每个节点都局部共轭，那么按照子节点的拓扑排序顺序，依次反转从 v 指向每个子节点的边，即可将 v 变为叶节点。

边缘化多个非叶节点。 定理 1 描述了如何修改图模型，同时保持联合分布不变，将一个非叶节点转换为叶节点，以便对其进行边缘化。现在，我们希望利用这一操作尽可能多地边缘化节点。算法 1 中的 MARGINALIZE 函数给出了为此采用的启发式方法：它只是按反转 拓扑序，对每个节点 v 尝试应用定理 1 中的操作进行边缘化。这样做很方便，因为它会自动同时移除所有没有通往观测变量的路径的节点。如果 v 可以被边缘化，就会执行反转操作，并将 v 从 G 中移除，再压入栈 S ，以确定恢复顺序。RECOVER 函数则通过直接采样边缘化变量，为非边缘化变量的样本补充这些变量的样本。接下来的章节将讨论 CONJUGATE 和 REVERSE 的实现。

3.5. 共轭检测

检测 x_a 何时与 x_b 局部共轭时，会使用表 1 中列出的模式，其中 (1) AFFINE(u, v) 表示u 可以写成 $u = pv + q$ ，其中表达式 p 和 q不包含 v；(2) DEPENDENT(u, v) 表示在计算图中存在一条从 v 到 u 的路径；(3) LINEAR(u, v) 表示 u 可以写成 $u = pv$,

其中表达式 p 不包含 v。例如，第一个模式匹配的情形是 x_a 和 x_b都服从正态分布；此时，我们可以从表达式 $f_b(\mathbf{x}^{\text{pa}(b)})$ 中提取参数 μ_b 和σ_b^2 的表达式（计算图），作为该表达式的两个输出。该模式还意味着，如果 AFFINE(μ_b, x_a) 为真且 DEPENDENT(σ_b^2, x_a) 为假，那么 x_a与 x_b局部共轭。AFFINE、LINEAR 和 DEPENDENT 函数需要检查计算图；详情和伪代码见附录 C。虽然我们聚焦于五种常见模式，但也可以将其他模式[5]引入我们的流程，并且有可能利用基于指数分布族性质的自动共轭检测，正如 autoconj 中所做的那样（Hoffman 等人，2018年）。

3.6. 边反转详解

如果检测到共轭性，算法 1 将调用 REVERSE 操作来反转一条边。算法 2 展示了 REVERSE 算法中处理正态-正态共轭的部分。我们强调，诸如 $+$、$-$ 和 $*$这样的操作，是对计算图进行的符号式操作。该算法实现了广为人知的高斯边缘化和条件化公式。第 5 行提取正态分布参数的符号表达式。第 6 行提取表达式 p 和 q，使得 $\mu_c = px_v + q$；共轭性检测已确定此类表达式存在。函数 AFFINE_COEFF 在附录 D 中定义。第 7–12 行计算边缘分布 $p(x_c|\cdots)$ 和条件分布 $p(x_v|x_c, \cdots)$ 参数的符号表达式，并将其写入 f_c 和 f_v。最后，第 15–16 行更新有向无环图，以反映新的依赖关系。

算法 2反转边（正态-正态情形）

1: **函数** REVERSE ($G = (D_i,\ \text{pa}(i)\ f_i)_{i=1}^M$,$v$,$c$)
2: // pa(v)，pa(c) f_v，f_c，D_c原地更新
3: // 所有变量都表示符号表达式
4: **if** D_c 服从正态分布**并且** D_v 服从正态分布**那么**
5: 设 μ_v 和 σ_v^2 为两个输出表达式 f_v，且 μ_c 和 σ_c^2 为 f_c的输出表达式。
6: $p, q =$ 仿射_COEFF(μ_c, x_v)
7: $k = \sigma_v^2 p/(p^2\sigma_v^2 + \sigma_c^2)$
8: $\mu_c' = p\mu_v + q$
9: $\sigma_c'^2 = p^2\sigma_v^2 + \sigma_c^2$
10: $\mu_v' = \mu_v + k(x_c - \mu_c')$
11: $\sigma_v'^2 = (1 - kp)\sigma_v^2$
12: $f_c = (\mu_c',\ \sigma_c'^2)$，$f_v = (\mu_v',\ \sigma_v'^2)$
13: **else**
14: ⋮ （参见附录 D 中的完整算法）
15: pa(c) $=$ (pa(c) $\setminus \{v\}$) $\cup$ pa(v)
16: pa(v) $=$ pa(v) $\cup \{c\} \cup$ pa(c)
17: **返回** G

3.7. 实现

我们已经准备好了自动边缘化哈密顿蒙特卡罗所需的各个部分。完整流程如下：(1) 从用户的程序中提取图模型 G，(2) 调用 MARGINALIZE 函数，得到边缘化模型 G' 和恢复栈 S，(3) 在 G' 上运行哈密顿蒙特卡罗，(4) 对于每个哈密顿蒙特卡罗样本 $\mathbf{x}$，调用 RECOVER(S，$\mathbf{x}$) 对边缘化变量进行采样。

我们的实现使用 JAX（Bradbury 等人，2018年）和 NumPyro（Bingham 等人，2018年；Phan 等人，2019年）提取图模型 G。我们使用 JAX 跟踪工具将 NumPyro 程序转换为 JAX 表达式（Jaxpr），即整个采样过程的计算图。NumPyro 程序必须通过一个围绕 NumPyro 的 sample 语句的薄封装，将模型的随机变量注册到 Jaxpr 中。我们从 NumPyro 跟踪栈中提取分布族，并通过解析 Jaxpr，提取从父随机变量到分布参数的部分计算映射，从而获得参数函数 f_i 参数。如前所述，我们的方法仅适用于可映射为图模型的程序，这意味着它们从固定序列的条件分布中抽样。这与 NumPyro 目前能够执行推断的程序十分吻合，因为 NumPyro 推断中的即时编译步骤需要构建静态计算图。NumPyro 的实验性控制流原语（“scan”和“cond”）不受支持，而且支持它们可能很困难。目前的实现仅适用于包含标量运算和逐元素数组运算的 Jaxpr，不过这一限制并非根本性的。我们预计，我们的方法也兼容使用计算图的其他概率编程语言，但对程序有类似的限制。

对于基于 JAX 的实现，边缘化计算图的复杂度通过 Jaxpr 的行数进行评估。我们尚未证明边缘化后计算图复杂度的界限，但在实验中观察到，其规模增幅较为温和。

4. 相关工作

共轭性与边缘化长期以来一直是概率编程中的重要议题。在 BUGS（Lunn 等人，2000年）和 JAGS（Hornik 等人，2003年）中，利用共轭性来改进自动吉布斯采样。这些系统利用共轭性推导完全条件分布，而不是边缘化变量。Hakaru（Narayanan 等人，2016年）和 PSI（Gehr 等人，2016年；2020年）使用符号式积分器进行边缘化，以实现精确推断。我们利用图模型提供的信息来识别特定模式，这在大规模

[5]关于背景知识和共轭模式的更多内容，请参阅 https://en.wikipedia.org/wiki/Conjugate_prior。

表 *1.* 共轭模式。若满足某一行的条件，则 x_a 的分布与 x_b 的分布局部共轭。

x_a 的分布	x_b 的分布	条件 1	条件 2
$\mathcal{N}(\mu_a,\ \sigma_a^2)$	$\mathcal{N}(\mu_b,\ \sigma_b^2)$	仿射$(\mu_b,\ x_a)$	不依赖$(\sigma_b,\ x_a)$
$\Gamma(\alpha_a,\ \beta_a)$	$\Gamma(\alpha_b,\ \beta_b)$	线性$(\beta_b,\ x_a)$	不依赖$(\alpha_b,\ x_a)$
$\Gamma(\alpha_a,\ \beta_a)$	指数(λ_b)	线性$(\lambda_b,\ x_a)$	-
贝塔分布$(\alpha_a,\ \beta_a)$	二项分布$(n_b,\ p_b)$	$p_b = x_a$	不依赖$(n_b,\ x_a)$
贝塔分布$(\alpha_a,\ \beta_a)$	伯努利(λ_b)	$\lambda_b = x_a$	-

模型中效率更高。Autoconj（Hoffman 等人，2018年）提出了一种项图重写系统，可利用共轭性对数联合密度进行边缘化。我们的方法与之不同：我们针对生成过程的图模型和计算图进行操作，而非对数密度。此外，正如引言中所述，尽管 Autoconj 提供了用于从对数密度中边缘化变量的基本操作，用户还必须指定如何应用这些操作，才能边缘化变量或实现推断目标。相比之下，我们开发了一种用于转换图模型的端到端算法；用户只需提供原始模型。未来，或许可以将 Autoconj 的基本操作与我们的图模型方法结合起来。

Gorinova 等人（2021年）提出了一种信息流类型系统，可用于对离散随机变量进行自动边缘化。在探索表达能力更强的概率编程语言之后，流式模型也吸引了广泛关注。Murray 等人（2018年）提出了延迟采样，该方法通过 Rao-Blackwell化粒子滤波器（RBPF）（Doucet 等人，2000年）实现自动边缘化，从而改进推断。延迟采样已在 Birch（Murray 和 Schön，2018年）、Pyro（Bingham 等人，2018年）中结合 funsors 得到开发（Obermeyer 等人，2019a；b），也应用于 Anglican（Lundén，2017年）和 ProbZelus（Baudart 等人，2020年）。Atkinson 等人（2022年）提出了半符号推断，进一步将延迟采样的适用范围扩展到具有任意结构的模型。我们的工作有所不同：我们在执行推断之前对模型进行静态分析，以改进马尔可夫链蒙特卡罗；这使我们的方法成为“完全符号式”的（不提供具体值），并引出了不同的算法考量。不过，我们的算法 1 与 Atkinson 等人（2022年）的提升算法在技术基础上有共同之处；更多详情请参阅附录E。

在应用概率建模中，通过边缘化变量来手动重构模型，从而实现更高效的推断，是一种广为人知的技巧，在某些语境下也称为折叠采样 。具体而言，折叠吉布斯采样（CGS）（Liu，1994年）是一种边缘化共轭先验变量，并在之后恢复这些变量以便进行吉布斯采样（GS）的算法。它可应用于潜在狄利克雷分配等混合模型（LDA）模型（Porteous 等，2008；Murphy，2012年）。边缘化后剩余的变量通常是离散变量，因此可以通过归一化推导出 GS 所需的任意条件分布。我们工作的关键区别在于，模型是自动重构的，而非手动重构。尽管我们聚焦于哈密顿蒙特卡罗（HMC），但我们的方法也可用于自动折叠模型，以进行吉布斯采样或其他马尔可夫链蒙特卡罗算法。

许多工作从不同角度改进了概率编程语言中的哈密顿蒙特卡罗推断。Stan（Carpenter 等人，2017年）使用哈密顿蒙特卡罗推断处理概率编程语言，对该领域产生了巨大影响。由于 Stan 程序指定的是对数密度，而不是采样过程，因此我们的想法无法直接应用于 Stan 程序。不过，许多 Stan 程序在理念上具有生成性，Baudart 等人（2021年）刻画了 Stan 程序的一个子集，可以将本文的方法直接应用于其中。Papaspiliopoulos 等人（2007年）提出了一个用于马尔可夫链蒙特卡罗中非中心化（重新）参数化的通用框架。Gorinova 等人（2020年）利用变分推断，自动完成模型参数化方式的选择。Parno 和 Marzouk（2018年）以及 Hoffman 等人（2019年）将所有潜变量的参数化学习为归一化流（Papamakarios 等人，2021年；Rezende 和 Mohamed，2015年）。对于某些变量可以边缘化的模型，我们的方法比重新参数化效果更好：示例见第 5.2 节。Mak 等人（2022年）利用对合马尔可夫链蒙特卡罗框架（Neklyudov 等人，2020年；Cusumano-Towner 等人，2020年），将马尔可夫链蒙特卡罗在概率编程语言中的适用范围扩展到了非参数模型。

5. 实验

我们在两类共轭性发挥重要作用的层次模型上评估了该方法的性能。所有实验均使用 NumPyro 的无转向采样器（NUTS）（Hoffman & Gelman，2014年），下文称为 HMC。我们的方法称为“边缘化 HMC”（HMC-M）。所有实验中，我们使用 10,000 个预热样本来调整采样器，使用 100,000 个样本进行评估，并通过有效样本量（ESS）和耗时（包括 JAX 编译时间）来评估性能。

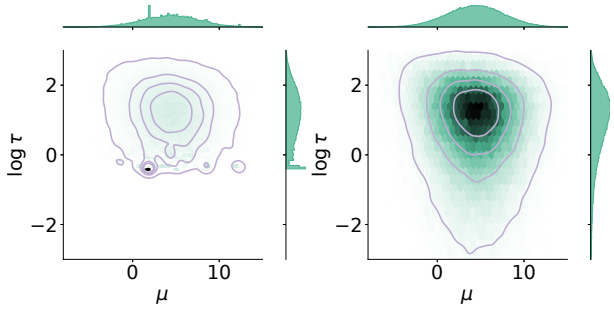

图 5. 采用不同算法从八校模型中抽取的 μ 和 $\log\tau$ 样本的直方图与等高线。两种情况下均抽取了 10 万个样本。左：使用 HMC 时，有效样本量极低，链无法充分混合，卡在某个特定区域，几乎没有探索 τ 的低值。右：使用 HMC-M 时，后验分布得到了均匀探索。

5.1. 层次部分汇聚模型

层次部分汇聚（HPP）模型（Gelman 等，1995）的形式为 $p(\theta,\ z_{1:n},\ y_{1:n}) = p(\theta)\prod_{i=1}^{n} p(z_i\,|\,\theta)p(y_i\,|\,\theta,\ z_i,\ x_i)$，其中 $(x_i,\ y_i)$ 是第 i 个数据点的观测协变量值和响应值，z_i 是局部潜变量，θ 是用于建模共享依赖关系的全局潜变量。在某些 HPP 模型中，z_i 的分布被设定为与 y_i 共轭。八校模型就是其中一个例子。图 5 展示了 HMC 和 HMC-M 对八校模型中的 $\theta = [\mu, \tau]$ 进行采样得到的结果。若不进行边缘化，HMC 难以对 τ 的小值进行采样，并且由于 τ 与 x_i 之间存在“漏斗”关系，链会陷入停滞（Papaspiliopoulos 等，2007年）。进行边缘化后，x_i 和漏斗关系都从 HMC 中消除，最终样本的质量显著提高。

层次部分汇聚的另一种应用是重复二项试验：对于每个单位 i，我们观察到 K_i 次试验中的成功次数 y_i，并假设成功概率具有部分共享的结构，例如（Carpenter 等，2017年）：

$$m \sim \text{Uniform}(0,1),\ \kappa \sim \text{Pareto}(1, 1.5),$$
$$\theta_i \sim \text{Beta}(m\kappa, (1-m)\kappa),\ y_i \sim \text{Binomial}(K_i, \theta_i).$$

应用包括大鼠肿瘤数据集（Tarone，1982年）、1970年棒球安打数据集（Efron 和 Morris，1975）以及 1996年美联棒球安打数据集（Carpenter 等，2017年）。由于 κ 与 θ_i 之间存在漏斗关系，该模型对 HMC 而言同样难以处理（Carpenter 等，2017年）。建议的解决方法包括对 κ 使用指数分布建模（Patil 等，2010），或将模型改写为适用重参数化的形式（Carpenter 等，2017年）。

我们注意到，由于 θ_i (贝塔分布)与y_i (伯努利分布)局部共轭，边缘化是更好的策略。在边缘化模型中，y_i 是一个贝塔-二项随机变量，哈密顿蒙特卡罗仅对 m 和 κ进行采样，而每个 θ_i 都在之后采样——从 $p(\theta_i\,|\,m,\ \kappa,\ y_i)$中采样，该分布为贝塔分布。漏斗问题得以消除，哈密顿蒙特卡罗的维度也从 $n+2$ 降至 2。我们的方法会自动实现这一点。

表 2 展示了结果。在该模型中，抽样 κ 众所周知很困难，但 HMC-M 达到的有效样本量与样本数处于相近的量级。哈密顿蒙特卡罗问题的维度也有所降低，因此运行时间更短。这些因素共同作用，使 1996 年美国联盟棒球击球数据集上的有效样本量/秒提升了 100 倍以上。我们还在 Stan 上进行了相同的实验，手动使用算法 1 重写模型。我们确认，如果在该情境下实现我们的方法，Stan 也能从相同的模型变换中受益。详情请参见附录 F。

5.2. 分层线性回归

与部分汇聚类似，也可以在线性回归模型中引入层级结构。我们通过两个示例展示我们的方法如何改进这类模型中的推断。

电力公司：电力公司模型（Gelman 和 Hill，2006）研究了一档教育电视节目对儿童阅读能力的影响。$C = 192$ 个班级分布在$G = 4$ 个年级中，并划分为 $P = 96$ 个治疗组-对照组配对。班级 k 用（g_k，p_k，t_k，y_k）表示，其中 g_k 是年级，p_k 是配对索引，$t_k \in \{0, 1\}$ 是处理变量，y_k 是平均分。配对 j中的班级属于年级 $\text{gp}[j]$。完整模型为

$$\mu_i \sim \mathcal{N}(0, 1),\ a_j \sim \mathcal{N}(100\mu_{\text{gp}[j]}, 1),$$
$$b_i \sim \mathcal{N}(0, 100^2),\ \log\sigma_i \sim \mathcal{N}(0, 1),$$
$$y_k \sim \mathcal{N}(a_{p_k} + t_k b_{g_k}, \sigma^2_{g_k}).$$

其中 $i \in \{1, \ldots, G\}$、$j \in \{1, \ldots, P\}$ 和 $k \in \{1, \ldots, C\}$。注意，μ_i、a_j、b_i 和 y_k 都服从具有仿射依赖关系的正态分布。因此，可以将 μ_i、a_j 和 b_i 从哈密顿蒙特卡罗过程中边缘化。正如第 2 节所指出的，手动完成这项工作非常困难，但我们的方法可以自动完成。

我们观察到，对 μ_i 进行边缘化会导致 JAX 编译时间非常长，尽管对数密度的计算图并没有比边缘化前大多少（14606 个原语操作，而之前为 9186 个）。我们认为这是当前 JAX 的一项限制。实验性证据请参见附录 G。作为变通方法，我们手动阻止了对 μ_i 的边缘化。

图 6 展示了结果。在该模型中，原始模型上的哈密顿蒙特卡罗表现不佳，但对 a_j变量进行重参数化非常有帮助：这一替代方案称为 HMC-R，其有效样本量表现出色（与样本数量相当）。不过，必须注意的是，这需要用户进行干预，可以采用 Gorinova 等人（2018）提供的工具

表 2. 重复二项试验模型中，HMC 和 HMC-M 在所有维度上的最小有效样本量、运行时间（秒）以及每秒最小有效样本量。报告了 5 次独立运行的均值和标准差。

数据集	算法	最小有效样本量	时间（秒）	最小有效样本量/秒
1970年棒球击球数 ($n = 18$)	HMC	1384.1 (1156.7)	**94.5** (5.7)	14.8 (12.7)
	HMC-M	**39001.8** (20030.4)	110.5 (89.2)	**592.3** (304.2)
大鼠肿瘤 ($n = 71$)	HMC	24632.3 (1494.5)	654.8 (43.9)	37.7 (2.1)
	HMC-M	**77644.5** (9570.8)	**72.4** (0.3)	**1072.7** (134.0)
1996年美国联盟棒球安打 ($n = 308$)	HMC	9592.3 (260.1)	2746.1 (107.6)	3.5 (0.2)
	HMC-M	**61109.0** (3344.9)	**130.9** (1.4)	**467.0** (29.5)

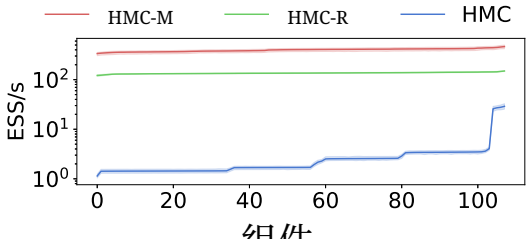

图 6。 电力公司模型上的逐分量有效样本量/s。我们的方法（HMC-M）与 HMC 以及采用重参数化的 HMC（HMC-R）进行比较。各分量按有效样本量/s 排序。

，也可以自行实现重参数化。同样重要的是要认识到，非中心化重参数化并不一定总能改善模型，在某些情况下甚至可能产生不利影响（Gorinova 等人，2018）。此外，HMC-R 不会降低维度，而是解决一个 $3G + P = 108$ 维问题；自动边缘化则将问题维度降至 8，并带来额外 4 倍的加速。

肺纤维化： 肺纤维化数据集（Shahin 等，2020年）包含患者随时间变化的用力肺活量（FVC）和疾病指标观测记录。假设每位患者的 FVC 关于时间呈线性关系，回归斜率和截距由层次模型生成。肺纤维化数据集（Shahin 等，2020年）中的记录形式为（ID_i，t_i，y_i），其中 ID_i 是患者编号，t_i 是观测时间，y_i 是用力肺活量（FVC），即疾病进展的一项指标。假设每位患者的 FVC 关于时间呈线性关系，回归斜率和截距由以下层次模型生成：

$$\mu_\alpha \sim \mathcal{N}(0, 500^2),\ \sigma_\alpha \sim \text{HalfCauchy}(100),$$
$$\mu_\beta \sim \mathcal{N}(0, 3^2),\ \sigma_\beta \sim \text{HalfCauchy}(3),$$
$$\alpha_j \sim \mathcal{N}(\mu_\alpha, \sigma_\beta^2),\ \beta_j \sim \mathcal{N}(\mu_\beta, \sigma_\beta^2),$$
$$\sigma \sim \text{HalfCauchy}(100),\ y_i \sim \mathcal{N}(\alpha_{\text{ID}_i} + t_i\beta_{\text{ID}_i}, \sigma^2),$$

其中 $i \in \{1, \ldots, 549\}$ 和 $j \in \{1, \ldots, 173\}$。我们再次阻止两个顶层变量被边缘化，并

表 3. 肺纤维化模型中 HMC、HMC-R 和 HMC-M 在所有维度上的最小有效样本量、耗时（分钟）及每秒最小有效样本量。报告 5 次独立运行的均值和标准差。

算法	最小有效样本量	时间（分钟）	每秒最小有效样本量
HMC	19817 (1207)	51.8 (0.1)	0.1 (0.0)
HMC-R	11362 (1567)	51.5 (1.3)	0.1 (0.0)
HMC-M	**96135** (557)	**14.9** (1.3)	**1.8** (0.1)

由于 JAX 编译速度较慢，移除 2 3 个数据点。在这些设置下，HMC-M 的表现优于 HMC 和 HMC-R，能在更短时间内生成更多有效样本（表 3）。

6. 讨论

我们提出了一种框架，用于从概率编程语言生成的图模型中自动边缘化变量，以加速马尔可夫链蒙特卡罗。结果表明，在具有共轭性的模型中，性能得到了显著提升。整个过程可以完全自动化，从而使用户免于繁琐的推导和实现。该方法仅限于图模型，因此无法涵盖概率编程语言的部分特性，但能够覆盖范围极其广泛的应用统计模型。当前的实现仅支持标量和逐元素数组运算。未来工作中的一个重要方向是支持更广泛的数组运算，包括矩阵运算、索引、切片和广播。另一个有趣的方向是将自动边缘化引入适用于高阶概率编程语言的马尔可夫链蒙特卡罗。我们相信，这项工作是迈向自动化马尔可夫链蒙特卡罗系统的重要一步；该系统能够在用户提供最少输入的情况下，在广泛的模型上实现良好性能。

致谢

感谢 Justin Domke 和匿名审稿人提出的意见，这些意见极大地改进了本文。本材料基于美国国家科学基金会第 1749854 号和第 1908577 号资助项目的支持。

参考文献

Atkinson, E., Yuan, C., Baudart, G., Mandel, L., 和 Carbin, M. 面向高效流式概率编程的半符号推断。*ACM* 程序语言会议论文集，6（OOPSLA），2022年。

Baudart, G., Mandel, L., Atkinson, E., Sherman, B., Pouzet, M. 和 Carbin, M. 响应式概率编程。载于第 *41* 届 *ACM SIGPLAN* 编程语言设计与实现大会论文集，第 898–912 页，2020年。

Baudart, G., Burroni, J., Hirzel, M., Mandel, L. 和 Shinnar, A. 将 Stan 编译为生成式概率语言，并扩展至深度概率编程。载于第 *42* 届 *ACM SIGPLAN* 国际编程语言设计与实现大会论文集，第 497–510 页，2021年。

Betancourt, M. 和 Girolami, M. 用于层次模型的哈密顿蒙特卡罗。贝叶斯方法论及其应用的当前趋势，79(30):2–4，2015年。

Bingham, E., Chen, J. P., Jankowiak, M., Obermeyer, F., Pradhan, N., Karaletsos, T., Singh, R., Szerlip, P., Hors- fall, P. 和 Goodman, N. D. Pyro：深度通用概率编程。机器学习研究杂志，2018年。

Bradbury, J., Frostig, R., Hawkins, P., Johnson, M. J., Leary, C., Maclaurin, D., Necula, G., Paszke, A., VanderPlas, J., Wanderman-Milne, S. 和 Zhang, Q. JAX：Python+NumPy 程序的可组合变换。2018年。

Carpenter, B., Gelman, A., Hoffman, M. D., Lee, D., Goodrich, B., Betancourt, M., Brubaker, M., Guo, J., Li, P., andRiddell, A. Stan：一种概率编程语言。统计软件杂志，76(1)，2017年。

Cusumano-Towner, M., Lew, A. K. 和 Mansinghka, V. K. 使用概率编程和可微编程自动化对合马尔可夫链蒙特卡罗。*arXiv* 预印本 *arXiv:2007.09871*，2020年。

Cusumano-Towner, M. F., Saad, F. A., Lew, A. K. 和 Mansinghka, V. K. Gen：一种具有可编程推断功能的通用概率编程系统。载于第*40*届 *ACM SIGPLAN* 编程语言设计与实现大会论文集，PLDI 2019年，第221–236页，纽约，纽约州，美国，2019年。ACM。ISBN 978-1-4503-6712-7。

Doucet, A.、De Freitast, N. 和 Russent, S. 动态贝叶斯网络的 Rao-Blackwell 化粒子滤波。载于《第十六届人工智能不确定性会议论文集》，第 176–183 页。Cite-seer，2000年。

Duane, S., Kennedy, A. D., Pendleton, B. J等, d Roweth, D。混合蒙特卡罗。《物理快报*B*》，195(2):216–222，1987年。Efron, B. 和 Morris, C. 使用 Stein 估计量及其推广进行数据分析。《美国统计学会杂志》，70(350):311–319，1975年。

Gehr, T., Misailovic, S. 和 Vechev, M. PSI：概率程序的精确符号推断。载于计算机辅助验证国际会议，第62–83页。Springer，2016年。

Gehr, T., Steffen, S. 和 Vechev, M. λPSI：针对高阶概率程序的精确推断。载于第 *41* 届 *ACM SIGPLAN* 编程语言设计与实现大会论文集。ACM，2020年6月。doi：10.1145/3385412.3386006。

Gelman, A. 和 Hill, J. 使用回归和多层/层次模型进行数据分析。剑桥大学出版社，2006年。

Gelman, A.、Carlin, J. B.、Stern, H. S. 和 Rubin, D. B.贝叶斯数据分析。Chapman and Hall/CRC，1995年。

Goodman, N. D. 和 Stuhlmüller, A. 概率编程语言的设计与实现。`http://dippl.org`，2014年。访问日期：2023-1-24。

Goodman, N. D.、Mansinghka, V. K.、Roy, D. M.、Bonawitz, K. A. 和 Tenenbaum, J. B. Church：一种用于生成模型的语言。载于 McAllester, D. A. 和 Myllymäki, P.（编），*UAI 2008*，第二十四届人工智能不确定性会议论文集，芬兰赫尔辛基，*2008*年*7*月*9*日至*12*日，第 220–229 页。AUAI出版社，2008年。

Gorinova, M., Moore, D. 和 Hoffman, M. 概率程序的自动重参数化。载于国际机器学习会议，第 3648–3657 页。PMLR，2020年。

Gorinova, M. I. 概率程序分析。博士论文，爱丁堡大学，2022年。

Gorinova, M. I., Moore, D. 和 Hoffman, M. D. 概率编程中的自动重参数化。载于第 *1* 届近似贝叶斯推断进展研讨会，第 1–8 页，2018年。

Gorinova, M. I., Gordon, A. D., Sutton, C. 和 Vákár, M. 基于类型标注的条件独立性。《*ACM*编程语言与系统汇刊》（*TOPLAS*），44 (1):1–54，2021年。

Griewank, A. 和 Walther, A. 评估导数：算法微分的原理与技术。SIAM，2008。

Hoffman, M.、Sountsov, P.、Dillon, J.V.、Langmore, I.、Tran, D. 和 Vasudevan, S. 使用神经传输改善哈密顿蒙特卡罗中的不良几何结构。*arXiv* 预印本 *arXiv:1903.03704*，2019年。

Hoffman, M. D. 和 Gelman, A. 无转向采样器：在哈密顿蒙特卡罗中自适应设定路径长度。《机器学习研究期刊》，15(1):1593–1623，2014年。

Hoffman, M. D.、Johnson, M. J. 和 Tran, D. Autoconj：无需领域特定语言即可识别并利用共轭性。神经信息处理系统进展，31，2018年。

Hornik, K.、Leisch, F.、Zeileis, A. 和 Plummer, M. JAGS：使用吉布斯采样分析贝叶斯图模型的程序。载于*DSC*论文集，第2卷，2003年。Koller, D. 和 Friedman, N. 概率图模型：原理与技术。麻省理工学院出版社，2009年。Kucukelbir, A.、Tran, D.、Ranganath, R.、Gelman, A. 和 Blei, D. M. 自动微分变分推断。《机器学习研究期刊》，18:14:1–14:45，2017年。Liu, J. S. 贝叶斯计算中的折叠吉布斯采样器及其在基因调控问题中的应用。《美国统计学会杂志》，89(427): 958–966，1994年。Lundén, D. 概率编程语言 Anglican 中的延迟采样，2017年。Lunn, D.、Spiegelhalter, D.、Thomas, A. 和 Best, N. BUGS项目：演进、评析与未来方向。《医学统计》，28(25):3049–3067，2009年。Lunn, D. J.、Thomas, A.、Best, N. 和 Spiegelhalter, D. WinBUGS——贝叶斯建模框架：概念、结构与可扩展性。《统计与计算》，10 (4):325–337，2000年。Mak, C.、Zaiser, F. 和 Ong, L. 非参数对合马尔可夫链蒙特卡罗。载于国际机器学习会议，第14802–14859页。PMLR，2022年。Minka, T.、Winn, J.、Guiver, J.、Zaykov, Y.、Fabian, D. 和 Bronskill, J. / Infer.NET 0.3，2018年。微软剑桥研究院。http://dotnet.github.io/infer。Murphy, K. P. 机器学习：概率视角。麻省理工学院出版社，2012年。Murray, L. M. 和 Schön, T. B. 使用概率编程语言进行自动化学习：Birch。《控制年度评论》，46:29–43，2018年。

Murray, L. M., Lundén, D., Kudlicka, J., Broman, D., and Schön, T. B. 概率程序的延迟采样与自动 Rao-Blackwell 化。载于人工智能与统计学国际会议（*AISTATS 2018*），*2018*年*4*月*9*日至*11*日，加那利群岛兰萨罗特岛普拉亚布兰卡，西班牙，第 84 卷机器学习研究论文集，第 1037–1046 页。PMLR，2018。Narayanan, P., Carette, J., Romano, W., Shan, C., and Zinkov, R. Hakaru 中通过程序变换进行概率推断（系统描述）。载于国际函数式与逻辑编程研讨会——第 *13* 届国际研讨会，*FLOPS 2016*，日本高知，*2016* 年 *3* 月 *4* 日至 *6* 日，论文集，第 62–79 页。Springer，2016。Neal, R. M. 神经网络的贝叶斯学习，第 118 卷统计学讲义。施普林格出版社，1996。Neklyudov, K., Welling, M., Egorov, E., and Vetrov, D. 对合马尔可夫链蒙特卡罗：一个统一框架。载于国际机器学习会议，第 7273–7282 页。PMLR，2020。Obermeyer, F., Bingham, E., Jankowiak, M., Phan, D., and Chen, J. P. 用于概率编程的函数张量。*arXiv* 预印本 *arXiv:1910.10775*，2019a。Obermeyer, F., Bingham, E., Jankowiak, M., Pradhan, N., Chiu, J., Rush, A., and Goodman, N. 板式因子图的张量变量消元。载于国际机器学习会议，第 4871–4880 页。PMLR，2019b。Papamakarios, G., Nalisnick, E. T., Rezende, D. J., Mohamed, S., and Lakshminarayanan, B. 用于概率建模与推断的归一化流。《机器学习研究期刊》，22:57:1–57:64，2021。Papaspiliopoulos, O., Roberts, G. O., and Sköld, M. 层次模型参数化的通用框架。《统计科学》，第 59–73 页，2007。Parno, M. D. and Marzouk, Y. M. 传输映射加速的马尔可夫链蒙特卡罗。*SIAM/ASA*不确定性量化期刊，6(2):645–682，2018。Patil, A., Huard, D., and Fonnesbeck, C. J. PyMC：Python 中的贝叶斯随机建模。统计软件杂志，35(4):1，2010。Phan, D., Pradhan, N., and Jankowiak, M. 用于灵活且高效概率编程的可组合效应（numpyro）。*arXiv* 预印本 *arXiv:1912.11554*，2019。Piponi, D., Moore, D., and Dillon, J. V. TensorFlow Probability 的联合分布。*arXiv* 预印本 *arXiv:2001.11819*，2020。

Porteous, I., Newman, D., Ihler, A., Asuncion, A., Smyth, P., and Welling, M. 用于潜在狄利克雷分配的快速坍缩吉布斯采样。载于第 *14* 届 *ACM SIGKDD* 知识发现与数据挖掘国际会议论文集，第 569–577 页，2008。

Rezende, D. 和 Mohamed, S. 使用归一化流进行变分推断。载于国际机器学习会议，第 1530–1538 页。PMLR，2015年。Shahin, A.、Wegworth, C.、David、Estes, E.、Elliott, J.、Zita, J.、SimonWalsh、Slepetys 和 Cukierski, W. OSIC 肺纤维化进展，2020年。Tarone, R. E. 在比例趋势检验中使用历史对照信息。*Biometrics*，第 215–220 页，1982年。Tran, D.、Hoffman, M. D.、Saurous, R. A.、Brevdo, E.、Murphy, K. 和 Blei, D. M. 深度概率编程。载于国际学习表征会议，2017年。van de Meent, J.-W.、Paige, B.、Yang, H. 和 Wood, F. 概率编程简介。*arXiv* 预印本 *arXiv:1809.10756*，2018年。Wood, F.、van de Meent, J. W. 和 Mansinghka, V. 概率编程推断的新方法。载于第 *17* 届人工智能与统计国际会议论文集，第 1024–1032 页，2014年。

A. 定义 1 正确性的证明

正文中的定义 1 为

定义 *1*（边反转）。假设 G 是一个图模型，其中节点 v 是 c 的父节点，且不存在从 v 到 c 的其他路径。反转 $v \to c$ 边会将因子 $p(x_v \mid \mathbf{x}_{\text{pa}(v)})p(x_c \mid x_v, \mathbf{x}_{\text{pa}(c)\setminus\{v\}})$ 替换为 $p(x_c \mid \mathbf{x}_U)p(x_v \mid x_c, \mathbf{x}_U)$，并更新父节点集合：$\text{pa}'(c) = U$，$\text{pa}'(v) = U \cup \{c\}$，其中 $U = \text{pa}(v) \cup \text{pa}(c) \setminus \{v\}$。

据称，该操作会产生一个与原始模型具有相同联合分布的图模型。现在我们来证明这一点。

证明。 只需证明：(1) 反转后的图模型仍然有效；(2) 联合分布保持不变，这要求

$$p(x_v \mid \mathbf{x}_{\text{pa}(v)})p(x_c \mid x_v, \mathbf{x}_{\text{pa}(c)\setminus\{v\}}) = p(x_c \mid \mathbf{x}_U)p(x_v \mid x_c, \mathbf{x}_U).$$

对于 (1)，我们需要证明该过程中不会形成环。对于任意 $p_v \in \text{pa}(v)$，都会添加一条边 $p_v \to c$。由于不存在从 c 到 p_v 的路径（否则原始模型中会出现环），这条边不会导致环。对于任意 $p_c \in \text{pa}(c) \setminus \{v\}$，会引入一条边 $p_c \to v$。这条边也不会导致环；否则就会发现从 v 到 c 的另一条路径。最后，边 $v \to c$ 被替换为 $c \to v$，这条边也不会引入环，因为从 v 到 c不存在其他路径。

现在我们证明（2）。由于从 v 到 c不存在其他路径，因此从 v 到 $\mathbf{x}_{\text{pa}(c)\setminus\{v\}}$ 中任一节点都不存在路径。在给定 $\mathbf{x}_{\text{pa}(v)}$ 的条件下，根据条件独立性，我们有 $p(x_v \mid \mathbf{x}_{\text{pa}(v)}) = p(x_v \mid \mathbf{x}_U)$。此外，从 $\text{pa}(v)$ 中的节点到 c 的所有路径，都被 v 或 c 的某个父节点阻断，因此在给定 $\mathbf{x}_{\text{pa}(c)\setminus\{v\}}$ 的条件下，根据独立性，$p(x_c \mid x_v, \mathbf{x}_{\text{pa}(c)\setminus\{v\}}) = p(x_c \mid x_v, \mathbf{x}_U)$。根据共轭性的性质，我们有

$$\begin{aligned} p(x_v \mid \mathbf{x}_{\text{pa}(v)})p(x_c \mid x_v, \mathbf{x}_{\text{pa}(c)\setminus\{v\}}) &= p(x_v \mid \mathbf{x}_U)p(x_c \mid x_v, \mathbf{x}_U) \\ &= p(x_c \mid \mathbf{x}_U)p(x_v \mid x_c, \mathbf{x}_U). \end{aligned}$$

□

B. 定理 1 的证明

我们首先重述定理 1。

定理 1。 设 G 是一个图模型，其中节点 v 有子节点 $c_1, \ldots, c_H$。如果 x_v 与 $x_{c_1}, \ldots, x_{c_H}$ 分别局部共轭，那么通过按任意拓扑排序对 $c_1, \ldots, c_H$ 排序，并按照此顺序反转从 v 到每个子节点的边，就可以将节点 v 变为叶节点。

证明。 不失一般性，假设 $c_1, \ldots, c_H$ 按照拓扑排序排列。我们通过归纳法证明。假设对于 $k \in \{0, \ldots, H\}$，我们已反转边 $v \to c_1, \ldots, v \to c_k$，并且满足以下性质：

(1) v 的子节点为 $c_{k+1}, \ldots, c_H$。

(2) $c_{k+1}, \ldots, c_H$ 按拓扑序排列；(3) 对于每个 $x_{c_{k+1}}, \ldots, x_{c_H}$，$x_v$ 都是局部共轭先验。

我们证明边 $v \to c_{k+1}$ 可反转，并且反转后这些性质对 $k+1$ 仍然成立。根据 (1) 和 (2)，c_{k+1} 在 v 的子节点中按拓扑序最小，因此除 $v \to c_{k+1}$外，不存在从 v 到 c_{k+1} 的路径。根据 (3)，x_v 是 $x_{c_{k+1}}$的局部共轭先验。于是我们可以对 $v \to c_{k+1}$应用边反转。现在检查替换后所有条件是否仍然成立。对于性质 (1)，v的子节点现在变为 $c_{k+2}, \ldots, c_H$。对于性质 (2)，边发生变化（入边或出边）的节点为 v、c_{k+1}及其父节点；在反转之前，这些节点在拓扑序中都排在 $c_{k+2}, \ldots, c_H$ 之前，反转后仍然如此，因此 $c_{k+2}, \ldots, c_H$ 的相对顺序没有变化。对于性质 (3)，x_v 的分布族没有变化，并且 $x_{c_{k+2}}, \ldots x_{c_H}$各自的条件分布也没有变化。因此，表 1 中的局部共轭条件对它们均保持不变，这意味着 x_v 的分布仍然是 $x_{c_{k+2}}, \ldots x_{c_H}$各自分布的局部共轭先验。

总之，通过归纳法，$k = H$ 满足这三个条件，这意味着可以按照上述步骤将 v 转换为叶节点。

□

算法 3确定变量对输入的依赖关系。$\mathbf{x}_{1:M}$ 是所有随机变量的集合。

函数依赖 (w_j, x)
  如果 $w_j = x$ **则**
    返回 真
  如果 $w_j \in \mathbf{x}_{1:M}$ **则**
    返回 假
  对 $p \in$ 预测(j)**执行**
    如果 依赖(w_p, x) **那么**
      返回 真
  返回 假

C. 共轭检测的详细信息

共轭检测需要查看函数的计算图。本节将介绍如何在计算图上进行共轭检测。在程序的跟踪过程中，计算过程会被编译为由基本运算组成的中间表示。例如，函数

```
def f(x, y):
    p = (x - y) ** 2
    q = (x + y) ** 2
    return p + q
```

可以表示为

```
INPUTS: a, b
c = SUB a b
d = SQUARE c
e = ADD a b
f = SQUARE e
g = ADD d f
OUTPUTS: g
```

通过考察中间表示，可以推断输出与输入之间的关系。对于函数 $f(x_{i_1}, x_{i_2}, \ldots, x_{i_k})$，我们在第 3.2 节中将其计算图定义为由一系列 N_f基本操作 构成。计算得到序列 $w_1, w_2, \ldots, w_{k+N_f}$，其中：(1) 前 k 个值是函数的输入，即对于 j= 1 到 k，有 $w_j = x_{i_j}$；(2) 后续每个值都由前面的值计算得出：

$$w_j = \phi_j(\mathbf{w}_{\text{pred}(j)}), \tag{2}$$

其中 ϕ_j 是作用于值集合 $\mathbf{w}^{\text{pred}(j)}$上的一项原语操作，而 pred($j$) $\subseteq \{1, \ldots, j-1\}$ 是j的前驱集合。在第 3.5 节中，我们已将共轭检测归结为仿射性、线性和依赖性检测。下面根据上述定义，首先介绍依赖性检测的细节。

给定一个具有计算图的函数，我们可能想要确定变量 w_j 是否依赖于输入 x_a。我们将结果定义为 DEPENDENT(w_j, x_a)，可通过算法 3 中所示的方程递归求得。请注意，在本文中，函数的输入始终是 $\mathbf{x}_{1:M}$ 中的随机变量。对于 DEPENDENT(w_j, x_a)，如果 w_j 是 $\mathbf{x}_{1:M}$ 中的随机变量，那么它必定是该函数的输入。此时，DEPENDENT 会返回 w_j 是否为 x_a。如果 w_j 是一个基本算子的结果，则会枚举该算子的输入。如果其中任一输入依赖于 x_a，那么结果就为真。在某些特殊情况下，该递归算法的复杂度可能达到指数级。我们将中间结果存储在字典中，并在递归前查询，以避免冗余计算。因此，复杂度与涉及的变量数量呈线性关系。

仿射性和线性可以纳入同一个框架。除了判断 w_j 是否与 x_a仿射外，我们还会返回斜率和截距是否非零。如果检测到仿射关系且截距为零，那么该关系就是线性的。我们将 AFFINE_ALL(w_j, x_a) 定义为一个由三个布尔变量组成的元组——分别表示 w_j 是否关于 x_a仿射，

算法 4确定输入中变量的仿射性和线性。$\mathbf{x}_{1:M}$ 是所有随机变量的集合。

函数仿射_全部（w_j, x）
  如果 $w_j = x$ **则**
    返回 真 , True, 假
  如果 $w_j \in \mathbf{x}_{1:M}$ **那么**
    返回 真 , 假 , True
  如果 $\phi_j \in$ {加，减} **且** 谓词（j） $= \{p_1, p_2\}$ **那么**
    r_1， s_2， $t_1 =$ 仿射_全部(w_{p_1} x)
    r_2， s_2， $t_2 =$ 仿射_全部(w_{p_2} x)
    返回 r_1 **和** r_2, s_1 **或** s_2, t_1 **或** t_2
  如果 $\phi_j =$ 乘法 **且** pred(j) $= \{p_1, p_2\}$ **则**
    $r_1, s_2, t_1 =$ 仿射_ALL(w_{p_1} x)
    $r_2, s_2, t_2 =$ 仿射_ALL(w_{p_2} x)
    如果非 s_1 **则**
      返回 r_1 **和** r_2, t_1 **和** s_2, t_1 **和** t_2
    如果不是 s_2 **则**
      返回 r_1 **和** r_2， s_1 **和** t_2， t_1 **和** t_2
    返回 假 , 假 , 假
  如果 $\phi_j =$ 除法 **并且** 谓词(j) $= \{p_1, p_2\}$ **则**
    r_1， s_2， $t_1 =$ 仿射_全部(w_{p_1} x)
    r_2， s_2， $t_2 =$ 仿射_全部(w_{p_2} x)
    如果不是 s_2，**那么**
      返回 r_1 **和** r_2， s_1， t_1
    返回假 , 假 , 假
  对于 $p \in$ 谓词(j)**执行**
    r， s， $t =$ 仿射_全部(w_p， x)
    如果非 r **或** s **则**
      返回 假 , 假 , 假
  返回 真 , 假 , True

函数仿射 (w_j, x)
  $r, s, t =$ 仿射_全部(w_j, x)
  返回 r

函数 线性 (w_j, x)
  $r, s, t =$ 仿射_全部(w_j, x)
  返回 r **且非** t

斜率是否非零，以及截距是否非零。然后可以从 AFFINE_ALL(w_j, x_a) 得到 LINEAR 和 AFFINE。我们的仿射性检测算法改编自 Atkinson 等人（2022），并针对没有具体取值的情况做了少量修改。伪代码见算法 4。AFFINE_ALL(w_j, x_a) 的结果可通过枚举 ϕ_j 的各种情况，并根据计算图的结构进行归纳得到。例如，假设我们知道 $w_j =$ ADD(w_{p_1}, w_{p_2})，并且 $r_1, s_1, t_1 =$ AFFINE_ALL(w_{p_1}, x_a)，以及 $r_2, s_2, t_2 =$ AFFINE_ALL(w_{p_2}, x_a)，那么当 w_{p_1} 和 w_{p_2} 都与 x_1 仿射时，w_j 就与 x_a 仿射，也就是说（r_1 且 r_2）。如果 p_1 或 p_2 的斜率非零，那么斜率就非零，因此第二个返回值为（s_1 或 s_2）。截距是否非零的判断也相同，即（t_1 或 t_2）。

D. 反转边的细节

我们已经构建了共轭检测的各个部分。本节将讨论边缘化马尔可夫链蒙特卡罗中反转边的细节。在算法 2 中，定义了函数 AFFINE_COEFF，用于在检测到仿射性时获取系数。其伪代码见算法 5，与 AFFINE 类似。我们强调，算法 5 中的计算完全是符号式的。每个变量都对应一系列操作，这些操作可以视为一个计算图。

算法 5 获取输入 x 上变量 w_j 之间仿射关系的系数。$\mathbf{x}_{1:M}$ 是所有随机变量的集合。

函数仿射_系数(w_j, x)
  如果 $w_j = x$，**则**
    返回 1 , 0
  如果 $w_j \in \mathbf{x}_{1:M}$ **那么**
    返回 0 , 1
  如果 $\phi_j =$ 添加 **且** 前置条件$(j) = \{p_1, p_2\}$ **那么**
    $s_1, t_1 =$ 仿射_系数(w_{p_1}, x)
    $s_2, t_2 =$ 仿射_系数(w_{p_2}, x)
    返回 $s_1 + s_2, t_1 + t_2$
  如果 $\phi_j =$ 减去 **并且** 谓词$(j) = \{p_1, p_2\}$ **则**
    $s_1,\ t_1 =$ 仿射_系数(w_{p_1}, x)
    $s_2,\ t_2 =$ 仿射_系数(w_{p_2}, x)
    返回 $s_1 - s_2,\ t_1 - t_2$
  如果 $\phi_j =$ 乘法 **并且** 谓词$(j) = \{p_1, p_2\}$ **则**
    $s_1,\ t_1 =$ 仿射_系数(w_{p_1}, x)
    $s_2, t_2 =$ 仿射_系数(w_{p_2}, x)
    如果 s_1 为 0 **则**
      返回 $t_1 * s_2, t_1 * t_2$
    如果 s_2 为 0 **则**
      返回 $s_1 * t_2, t_1 * t_2$
    引发 错误
  如果 $\phi_j =$ 除法 **并且** 预测（j） $= \{p_1, p_2\}$ **那么**
    $s_1,\ t_1 =$ 仿射_系数(w_{p_1}, x)
    $s_2,\ t_2 =$ 仿射_系数(w_{p_2}, x)
    如果 s_2 为 0 **那么**
      返回 $s_1/t_2,\ t_1/t_2$
    引发 错误
  返回 0 , w_j

因此，每个 $+$、$-$、$*$ 和 $/$ 都是通过合并两个（可能重叠的）计算图来实现的。一个问题是，我们需要判断某些变量是否为零（第 17、19、25 行）。因此，对零值的运算需要特殊处理。例如，如果我们发现一个 $0 + 0$，就不应声明一个将两个零相加的操作，而应直接使用结果 0。

现在只剩下算法 2 的完整版本，可参见算法6。

E. 与提升算法的关系

Atkinson 等人（2022）提出的提升算法是一种在线算法，可用于自动运行 Rao-Blackwell 化粒子滤波器（RBPF）（Doucet 等人，2000年）。我们的算法 1 与提升算法密切相关。两种算法都能执行共轭检测，并对所有可能的随机变量进行边缘化。一个明显的区别是，算法 1 采用循环编写，而提升算法使用递归。不过，这两种算法的主要区别在于其应用场景。在 RBPF 中，所有不可边缘化的随机变量都从模型中采样，因此可以采用半符号化表示：在执行提升算法时，不可边缘化的随机变量会被采样值替代。而在哈密顿蒙特卡罗（HMC）中，不会直接对随机变量进行采样，并且同一个计算图会被多次执行，因此应先进行边缘化，再使用完全符号化表示运行。同时，定理 1 允许我们在两个不同的循环中分别进行共轭检测和逆向操作，从而缩短大规模模型的运行时间。提升算法作为在线算法无法实现这一改进，因此会执行一些不必要的逆向操作。此外，从实现角度来看，在算法 1 的 v 循环中，父节点是固定的。因此，在一次迭代中，所有对共轭和反转的调用都针对同一个 v。这样便可以保存这些函数的中间结果，避免在计算图上进行冗余计算。因此，算法 1 的时间复杂度为 $O(M|C|)$，其中 $|C|$ 是.....的大小

表 4. 在使用 Stan 的重复二项试验模型上，NUTS 和采用手动边缘化的 NUTS（NUTS-M）在所有维度上的最小有效样本量、运行时间（秒）以及每秒最小有效样本量。报告的是 5 次独立运行的均值和标准差。

数据集	算法	最小有效样本量	时间（秒）	每秒最小有效样本量
1970年棒球击球数（$n = 18$）	NUTS	3647.6 (1041.0)	23.3 (0.5)	155.9 (43.4)
	NUTS-M	**29674.5** (3209.4)	**21.1** (0.3)	**1407.9** (142.1)
大鼠肿瘤（$n = 71$）	NUTS	23556.4 (1487.4)	37.9 (1.4)	623.7 (59.0)
	NUTS-M	**45241.9** (1445.4)	**24.8** (0.4)	**1826.2** (84.4)
1996年美国联盟棒球安打 ($n = 308$)	NUTS	9684.6 (529.8)	99.8 (1.9)	97.0 (5.6)
	NUTS-M	**42718.0** (1207.6)	**43.0** (0.6)	**994.1** (27.0)

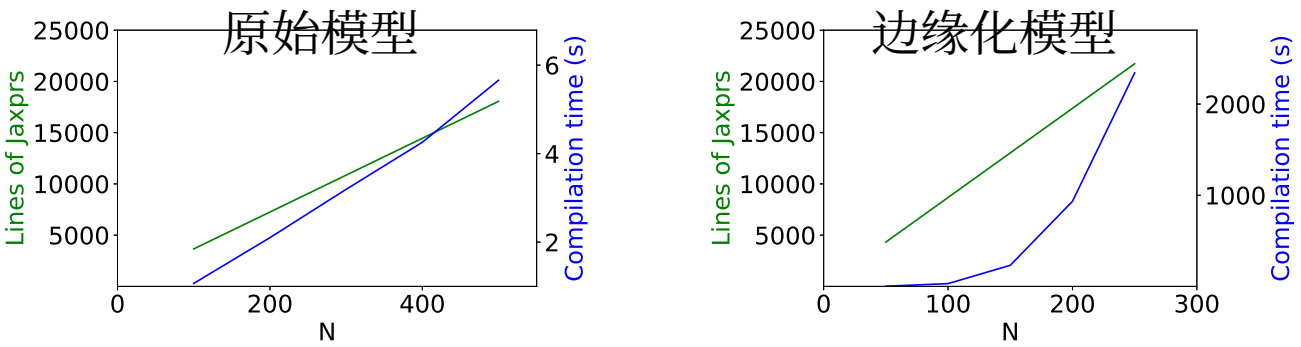

图 7. 简单层次模型中，对数密度函数梯度的 Jaxpr 数量，以及相对于 N 的 JIT 编译时间。尽管 Jaxpr 数量相近，边缘化模型的编译时间可能比原始模型慢数百倍。

计算图。鉴于上述考量，我们认为算法 1 是该领域的一项重要贡献。

F. 使用手动边缘化的 Stan 结果

尽管 Stan 尚未实现自动边缘化，但可以通过在 Stan 中进行手动边缘化测试，来确认其适用性。具体而言，可以将模型改写为边缘化模型，并使用`generated quantities` 部分恢复边缘化的变量。我们复现了表 2 中的实验，结果见表 4。通过引入边缘化，Stan 中的 NUTS 不仅运行得更快，还能为重复二项试验模型获得更好的样本。

G. JAX 编译缓慢

在实验中，我们发现 JAX 对某些模型的编译时间可能很长。我们将问题具体定位在边缘化层级模型的结构上。为此，我们考虑以下简单模型

$$x \sim \mathcal{N}(0, 1),\ \log \sigma \sim \mathcal{N}(0, 1),\ y_i \sim \mathcal{N}(x, \sigma),$$

其中 $i = 1$、……、 N和 $y_i = 0$ 对于所有 i 均作为伪观测给出。通过反转指向各个 y_i的边，可以对 x 进行边缘化。然而，我们发现，对于边缘化模型，JIT 编译时间会随着 N 超线性增长。见图 7。无论性能如何，当 N 足够大时，边缘化模型的梯度函数的 JIT 编译时间可能比原始模型长数百倍，而 Jaxpr 的行数却相近。这可能是因为边缘化会为所有观测创建链状计算图，而 JAX 在这种情况下难以处理。我们不认为这是我们方法的核心局限。

算法6算法 2 的完整版：反转一条边

函数 反转 $(G = (D_i, \text{pa}(i)\ f_i)_{i=1}^{M}, v, c)$
  // $\text{pa}(v)$,$\text{pa}(c)$ f_v, f_c, D_c 原位更新
  // 所有变量都表示符号表达式
  if D_c 服从正态分布**并且** D_v 服从正态分布**则**
    令 μ_v 和 σ_v^2 为两个输出表达式 f_v， μ_c 和 σ_c^2 为 f_c的输出表达式。
    p， $q =$ 仿射_COEFF$(\mu_c,\ x_v)$
    $k = \sigma_v^2 p/(p^2\sigma_v^2 + \sigma_c^2)$
    $\mu_c' = p\mu_v + q$
    $\sigma_c'^2 = p^2\sigma_v^2 + \sigma_c^2$
    $\mu_v' = \mu_v + k(x_c - \mu_c')$
    $\sigma_v'^2 = (1 - kp)\sigma_v^2$
    $f_c = (\mu_c',\ \sigma_c'^2)$， $f_v = (\mu_v',\ \sigma_v'^2)$
  如果 D_v 是贝塔分布**且** $D_c \in$ {是伯努利分布，二项分布} **那么**
    设 α_v 和 β_v 是 f_v的两个输出表达式
    if $D_c =$ 伯努利分布**那么**
      $n_c = 1$
      $p_c = \lambda_c$
    else
      设 n_c 和 p_c 是 f_c的两个输出表达式
    $\alpha_v' = \alpha_v + x_c$
    $\beta_v' = \beta_v + n_c - x_c$
    $D_c =$ 贝塔-二项分布
    $f_c = (n_c,\ \alpha_v,\ \beta_v)$， $f_v = (\alpha_v',\ \beta_v')$
  如果 D_v 是伽马分布**且** $D_c \in$ {是指数分布，伽马分布} **则**
    令 α_v 和 β_v 为 f_v的两个输出表达式
    如果 $D_c =$ 是指数分布**则**
      $\alpha_c = 1$
      $\beta_c = \lambda_c$
    else
      令 α_c 和 β_c 为 f_c的两个输出表达式
    p， $q =$ 仿射_系数$(\beta_c,\ x_v)$
    $\alpha_v' = \alpha_v + \alpha_c$
    $\beta_v' = \beta_v + p * x_c$
    $D_c =$复合伽马分布
    $f_c = (\alpha_c,\ \alpha_v,\ \beta_v/p)$， $f_v = (\alpha_v',\ \beta_v')$
  $\text{pa}(c) = (\text{pa}(c) \setminus \{v\}) \cup \text{pa}(v)$
  $\text{pa}(v) = \text{pa}(v) \cup \{c\} \cup \text{pa}(c)$
  返回 G