# A 用于贝叶斯反问题的域分解变分自编码器方法

Zhihang Xu[a]、Yingzhi Xia[b]、Qifeng Liao[a],*

[a]上海科技大学信息科学与技术学院，中国上海 *201210*[b]新加坡科技研究局（*A*STAR*）高性能计算研究所，新加坡 *138632*

arXiv:2301.05708v2 [stat.ML] 7 Feb 2023

**摘要**

当正向模型由复杂的偏微分方程（PDE）支配时，贝叶斯反问题通常面临巨大的计算挑战。这通常是由于正向模型评估成本高昂，以及先验分布的参数化维度过高所致。本文提出一种域分解变分自编码器马尔可夫链蒙特卡罗（DD-VAE-MCMC）方法，以同时应对这些挑战。该方法将全局物理域划分为多个小型子域，首先基于局部历史数据构建局部确定性生成模型，从而提供高效的局部先验表示。利用带主动学习的高斯过程模型处理域分解界面条件。随后，在各子域上独立并行地进行反演，并在低维潜在参数空间中求解。通过泊松图像融合过程对局部推断结果进行后处理，从而得到高效的全局推断结果。数值示例展示了所提方法的性能。

关键词：B 贝叶斯推断，马尔可夫链蒙特卡罗，域分解，深度生成模型 s.

## 1. 引言

反问题 [1] 广泛存在于科学和工程的许多领域，包括地震反演 [2], 、热传导 [3] 和地下水反演建模 [4]。正向模型通常由偏微分方程（PDE）定义，用参数作为输入来描述特定的物理现象。通常，求解正向模型的计算开销很大，但问题定义明确。相比之下，相关的逆问题旨在根据有限且含噪的观测值推断无法直接观测的隐藏参数，通常是病态的：不同的参数集可能产生相似的传感器测量结果；可能不存在能够拟合观测数据的可行解；或者微小误差也可能导致正向模型出现不可预测的变化。贝叶斯方法 [5, 6],将未知参数视为随机变量，把逆问题构建为一个概率问题，以刻画观测值、正向模型和先验知识中的不确定性。我们可以指定一个先验分布，以反映测量前对参数的了解。似然函数由正向模型刻画。收集观测值后，贝叶斯公式会更新后验分布。该解

*通讯作者电子邮箱： xuzhh@shanghaitech.edu.cn（Zhihang Xu），Xia_Yingzhi@ihpc.a-star.edu.sg（Yingzhi Xia），liaoqf@shanghaitech.edu.cn（Qifeng Liao）

贝叶斯反问题的解，即后验分布，并非单一数值，而是能够提供有关推断参数的统计信息的分布。

贝叶斯视角为刻画后验不确定性提供了一个框架，但在实践中应用贝叶斯公式时仍存在若干重大瓶颈。首先，后验分布通常没有闭式解，只能确定到一个归一化常数。因此，通常需要通过马尔可夫链蒙特卡罗（MCMC）方法等采样算法对其进行近似 [7, 8, 9]。其次，当正向模型由偏微分方程描述时，需要反复评估正向模型的采样方法会造成沉重的计算负担。最后，未知量通常是函数或场，离散化后可能具有很高的维度；在某些情况下，先验信息仅以历史数据的形式提供。因此，需要采用适当的参数化方法，通过相对低维的随机变量来表示空间相关场。

总之，理想的参数化方法应当：（1）降低未知量的维度；（2）仅利用历史数据来表示未知量。深度生成模型的目标恰好与此相符：生成模型可以利用样本，通过概率模型近似未知分布；而深度生成模型（DGMs） [10] 则利用神经网络构建概率模型，以增强学习能力。更重要的是，无需其他假设或限制，即可通过训练数据自然地挖掘先验信息。近年来，深度生成模型与反问题的结合受到广泛关注。文献 [11] 提出将变分自编码器（VAE） [12] 用作复杂地质介质反演的一种高效先验采样方法。文献 [13]介绍了一种基于变分自编码器的多尺度贝叶斯推断流程。在地球物理反演的不确定性量化中，研究者采用结合深度卷积架构的变分自编码器，并通过变分推断 [14]，利用深度混合模型探索后验分布。文献 [15]对生成对抗网络（GAN） [16] 和变分自编码器进行了实验，并指出，变分自编码器能够兼顾生成模式的准确性与基于梯度的反演的可行性，因此可能是更好的反演选择。

尽管将深度生成模型作为一种参数化方法这一思路颇具吸引力，但在实际应用中仍面临诸多挑战。例如，潜在参数空间仍可能是高维的，而且每次局部正演计算的计算成本都很高。通过将空间域分解为一系列较小的子域，可以将原问题转化为一组域和尺度更小、易于处理的子问题。这样一来，每个局部子域中定义的潜在参数空间维度都会降低。此外，与各子域相关的计算负担也会大幅减轻。

本文提出了一种用于贝叶斯反演的新方法，以应对上述挑战。我们的贡献如下。首先，我们提出域分解变分自编码器（DD-VAE）方法，以进一步增强深度生成先验分布的表示能力。我们的结果基于简单的全连接神经网络（FCNN），表明较小的子域使网络架构更具适应性且更浅，并能提高超参数调优的效率。此外，由于空间域经过划分，训练数据量得以增加；更重要的是，与全局潜变量相比，局部潜变量的维度有所降低。

其次，区域划分后，计算效率得到显著提升。如前所述，对计算成本高昂的正向模型进行重复评估，会给采样方法带来沉重的计算负担。通过分解空间域，全局正向模型被转换为多个局部正向模型，从而大幅降低计算成本。此外，区域分解策略所具备的并行能力进一步减少了计算开销。第三，我们利用结合主动学习的高斯过程（GP）模型来处理界面条件。最后，我们提出了一种重建未知全局场的新方案。由于直接拼接局部场会在界面处产生明显接缝，我们采用泊松融合技术对重叠的局部场进行融合，从而有效表示全局场。

本文其余部分安排如下。第2节建立问题，并介绍贝叶斯反问题和原始变分自编码器。第3节介绍我们提出的区域分解变分自编码器马尔可夫链蒙特卡罗（DD-VAE-MCMC）算法。第4节给出数值示例，以验证所提方法的效率。第5节总结全文。

## 2. 问题设定

本节简要回顾贝叶斯反问题的一般描述以及我们所考虑的控制问题。

### *2.1.* 贝叶斯反问题

令 $D \subset \mathbb{R}^{N_D}$ $(N_D = 1, 2, 3)$ 表示一个有界、连通且边界为多边形的空间域$\partial D$，并令 $s \in \mathbb{R}^{N_D}$ 表示空间变量。我们考虑从$N_d$维观测数据 $\boldsymbol{d} \in \mathbb{R}^{N_d}$中恢复一个随空间变化的参数函数 $x(s)$。出于实际考虑，我们使用标准有限元方法 [17] 将 $x(s)$离散为有限维向量 $x \in \mathbb{R}^{N_x}$ $(N_x \gg N_d)$。观测值与参数通常通过所谓的正向 问题相关联：

$$\boldsymbol{d} = F(x) + \epsilon_{\text{obs}}\,, \tag{1}$$

其中，$F$ 表示正向模型，$\epsilon_{\text{obs}}$ 表示加性噪声。贝叶斯公式 [6] 将未知参数$x$ 视为随机变量。假设$x$的先验分布$p(x)$已知，它体现了任何测量发生之前关于该参数的认知。噪声 $\epsilon_{\text{obs}} \in \mathbb{R}^{N_d}$ 的概率密度函数（PDF）记为 $\pi_{\epsilon_{\text{obs}}}(\epsilon_{\text{obs}})$。在本文中，假设噪声 $\epsilon_{\text{obs}}$ 服从均值为零、协方差矩阵为对角矩阵 $\sigma^2{}_{\text{obs}} I_{N_d}$的高斯分布，即 $\pi_{\epsilon_{\text{obs}}}(\epsilon_{\text{obs}}) = \mathcal{N}(\mathbf{0}, \sigma^2{}_{\text{obs}} I_{N_d})$，其中 $\sigma_{\text{obs}} > 0$ 为标准差，$I_{N_d}$ 为大小为 $N_d \times N_d$ 的单位矩阵，$\mathcal{N}(\mu, \boldsymbol{\Sigma})$ 表示均值向量为 $\mu$、协方差矩阵为 $\boldsymbol{\Sigma}$的高斯分布。似然函数用于评估模型预测与观测值之间的差异，可根据噪声的定义写为：

$$\boldsymbol{L}(\boldsymbol{d}|x) = \pi_{\epsilon_{\text{obs}}}(\boldsymbol{d} - F(x)) \propto \exp\left(-\frac{1}{2\sigma^2_{\text{obs}}}\|F(x) - \boldsymbol{d}\|_2^2\right), \tag{2}$$

其中， $\|\cdot\|_2$ 表示标准欧几里得范数。根据贝叶斯公式， $x$ 的后验分布可写为

as

$$\pi(x|\boldsymbol{d}) = \frac{\overbrace{\boldsymbol{L}(\boldsymbol{d}|x)}^{\text{likelihood}}\overbrace{p(x)}^{\text{prior}}}{\underbrace{\pi(\boldsymbol{d})}_{\text{evidence}}} \propto \boldsymbol{L}(\boldsymbol{d}|x)p(x)\,, \tag{3}$$

其中，式（3）中的证据 $\pi(\boldsymbol{d}) \coloneqq \int \boldsymbol{L}(\boldsymbol{d}|x)p(x)\,\mathrm{d}x$ 通常被视为归一化常数。

对于非线性正向模型，后验分布（3）通常没有解析形式，因此必须使用马尔可夫链蒙特卡罗（MCMC） [9]等采样方法进行近似。MCMC 生成马尔可夫链以近似目标分布，在此语境下，目标分布即为后验分布。为确保 MCMC 收敛，必须满足细致平衡条件。对于任意两个连续状态$x$ 和$x^\star$，细致平衡条件定义为

$$\pi(x|\boldsymbol{d})h(x,x^\star) = \pi(x^\star|\boldsymbol{d})h(x^\star,x)\,, \tag{4}$$

其中$h$ 表示转移核。其定义为

$$h(x,x^\star) = Q(x,x^\star)ar(x,x^\star)\,,$$

其中，$Q(x,x^\star)$ 是提议分布，$ar(x,x^\star)$ 是相应的接受概率。这里，$x^\star$ 由提议分布在给定当前状态$x$的条件下提出。为确保细致平衡条件（4）对于任意先验分布、似然函数和提议分布均成立，接受概率可定义为

$$ar(x,x^\star) \coloneqq \min\left\{1, \frac{\pi(x^\star|\boldsymbol{d})Q(x^\star,x)}{\pi(x|\boldsymbol{d})Q(x,x^\star)}\right\} = \min\left\{1, \frac{\boldsymbol{L}(\boldsymbol{d}|x^\star)p(x^\star)Q(x^\star,x)}{\boldsymbol{L}(\boldsymbol{d}|x)p(x)Q(x,x^\star)}\right\}.$$

在基本的马尔可夫链蒙特卡罗框架中，标准的 Metropolis-Hastings（MH）算法 [7, 8] 因其简单易用，是最常用的采样技术之一。然而，MH-MCMC 方法的主要缺点是，随着$x$的维度增加，采样效率会迅速下降。为此，本文采用预条件 Crank–Nicolson 马尔可夫链蒙特卡罗（pCN-MCMC）方法 [18, 19]。提议分布定义为满足

$$Q(x,x^\star)p(x) = Q(x^\star,x)p(x^\star)\,. \tag{5}$$

则接受概率$ar$ 可表示为

$$ar(x,x^\star) = \min\left\{1, \frac{\boldsymbol{L}(x^\star|\boldsymbol{d})}{\boldsymbol{L}(x|\boldsymbol{d})}\right\}. \tag{6}$$

pCN-MCMC 方法假设先验分布$p(x)$ 为高斯分布，这里令$p(x) = \mathcal{N}(\mu,\boldsymbol{C})$。对于当前状态$x$，pCN-MCMC 提议$x^\star$，其中

$$x^\star = \sqrt{1-\gamma^2}(x-\boldsymbol{\mu}) + \gamma\zeta + \boldsymbol{\mu}\,, \tag{7}$$

其中， $\zeta \sim \mathcal{N}(\mu,\boldsymbol{C})$， $\gamma$ 为控制随机移动的步长。可以看出，提议式 (7) 满足式 (5)。

pCN-MCMC 的过程如下。首先，可以从高斯先验分布中随机抽取一个初始状态 $x^{(1)}$ 。然后，对于第$k$ 个状态 $x^{(k)}$，根据式（7）抽取一个候选状态 $x^\star$ 。该候选

状态 $x^\star$ 随后以接受概率$ar$ （见式（6））被接受为下一个状态。详细的 pCN-MCMC 算法总结于算法 1 中。

**算法1** 预条件 Crank–Nicolson 马尔可夫链蒙特卡罗算法

**输入：** 正向模型 $F(x)$、观测数据 $\boldsymbol{d}$以及先验分布 $p(x) = \mathcal{N}(\mu,\boldsymbol{C})$。

1：从先验分布中生成一个初始状态 $x^{(1)}$。

2: **对于**$k = 1, \ldots, N_C - 1$ **执行**

3: 以步长 $\gamma$，提出

$$x^\star = \sqrt{1-\gamma^2}(x^{(k)} - \boldsymbol{\mu}) + \gamma\zeta + \boldsymbol{\mu},$$

其中 $\zeta \sim \mathcal{N}(\mu,\boldsymbol{C})$。

4: 计算接受率

$$ar(x, x^\star) = \min\left\{1, \frac{\boldsymbol{L}(x^\star|\boldsymbol{d})}{\boldsymbol{L}(x|\boldsymbol{d})}\right\}.$$

其中，(2) 中定义的似然函数 $\boldsymbol{L}$ 需要用到正向模型 (1)。

5: 从均匀分布 $\nu \sim \mathcal{U}[0,1]$ 中抽取 $\nu$。

6: **如果** $\nu < ar(x,x^\star)$ **则**

7: 接受提议状态，即令 $x^{(k+1)} = x^\star$。

8: **else**

9: 拒绝提议状态，即令 $x^{(k+1)} = x^{(k)}$。

10: **结束 if**

11: **结束 for**

**输出：** 后验样本 $\{x^{(k)}\}_{k=1}^{N_C}$。

### *2.2.* 带随机输入的偏微分方程

在本文中，我们考虑涉及偏微分方程的正向模型，即$F$ 包含求解一个适定的偏微分方程组。$x$ 的图像记为 $\Gamma$。所考虑问题的物理规律由空间域$D$ 上的偏微分方程组以及边界 $\partial D$上的边界条件支配，具体表述如下：求$u(s,x) : D \times \Gamma \to \mathbb{R}$，使得

$$\mathcal{L}(s, x; u(s,x)) = f(s), \qquad \forall (s,x) \in D \times \Gamma, \tag{8a}$$

$$\mathcal{B}(s, x; u(s,x)) = g(s), \qquad \forall (s,x) \in \partial D \times \Gamma, \tag{8b}$$

其中， $\mathcal{L}$ 是偏微分算子， $\mathcal{B}$ 是边界算子，二者均依赖于随机输入$x$。此外，$f$ 是源函数，$g$ 规定了边界条件。令 O 表示观测算子，例如，在给定的传感器位置获取解的值，则观测数据$\boldsymbol{d}$ 定义为$\boldsymbol{d} = \mathrm{O}(u(s,x))$。$x$ 的先验分布记为$p_x(x)$，在不致引起歧义时，简记为$p(x)$。

### *2.3.* 深度生成式先验模型

在贝叶斯反问题中，先验分布的选择至关重要，却也十分复杂。例如，潜在的先验分布可能无法用标准分布或其变体来刻画，或者

先验分布的信息以历史数据的形式隐含地存在 .

深度生成模型可作为一种灵活且可扩展的数据驱动参数化方法。变分自编码器和生成对抗网络[16] 能够用低维潜变量表示高维参数。相反，基于流的模型 [20] 和扩散模型 [21] 等生成模型，则用与场维度相同的潜变量来表示场，这意味着反演的维度问题依然存在。在这项工作中，与生成对抗网络相比，我们主要关注变分自编码器，因为其训练效率更高且更简单。

考虑历史数据集 $\mathrm{X} = \{\mathrm{x}^{(1)}, \ldots, \mathrm{x}^{(K)}\} = \{\mathrm{x}^{(k)}\}_{k=1}^{K}$，其中 $\mathrm{x}^{(k)} \in \mathbb{R}^{N_x}$ 是从先验分布 $\mathrm{p}(\mathrm{x})$ 中独立同分布抽取的样本，即对于 $\mathrm{k} = 1, \ldots, \mathrm{K}$，$\mathrm{x}^{(k)} \overset{\text{i.i.d.}}{\sim} \mathrm{p}(\mathrm{x})$。设 $\mathrm{p}_\theta(\mathrm{x})$ 是一个概率模型，其中 $\theta$ 表示其参数；我们的目标是用 $\mathrm{p}_{\theta^\star}(\mathrm{x})$ 表示先验分布 $\mathrm{p}(\mathrm{x})$，其中 $\theta^\star$ 表示模型的优化参数。概率模型常用的准则是最大化对数似然，即 $\theta^\star := \arg\max_\theta \log \mathrm{p}_\theta(\mathrm{X}) = \arg\max_\theta \sum_{k=1}^{K} \log \mathrm{p}_\theta(\mathrm{x}^{(k)})$。变分自编码器引入一个低维潜变量 $\alpha \in \mathbb{R}^{N_\alpha}(\mathrm{N}_\alpha < \mathrm{N}_x)$，以扩展概率模型的表征能力。这里，潜变量是指无法直接观测的变量；我们将由神经网络参数化的联合分布 $\mathrm{p}_\theta(\mathrm{x}, \alpha)$ 称为深度潜变量模型。边缘分布为 $\mathrm{p}_\theta(\mathrm{x}) = \int \mathrm{p}_\theta(\mathrm{x}, \alpha)\, \mathrm{d}\alpha$。然而，由于边缘似然是一个积分，且没有解析解或高效的估计方法，因此通常难以计算，导致无法通过最大化对数似然来进行优化。

由于这一问题难以处理，我们无法直接使用梯度下降法优化目标函数。变分自编码器框架通过引入深度参数化模型 $q_\phi(\alpha|x)$ 来近似 $p_\theta(\alpha|x)$，其中 $\phi$ 是模型的参数。模型 $q_\phi(\alpha|x)$ 通常称为编码器。对于任意给定的 $q_\phi(\alpha|x)$，我们有

$$
\begin{aligned}
\log p_\theta(x) &= \mathbb{E}_{q_\phi(\alpha|x)}[\log p_\theta(x)] = \mathbb{E}_{q_\phi(\alpha|x)}\left[\log\left[\frac{p_\theta(x,\alpha)}{p_\theta(\alpha|x)}\right]\right] \\
&= \mathbb{E}_{q_\phi(\alpha|x)}\left[\log\left[\frac{p_\theta(x,\alpha)}{q_\phi(\alpha|x)}\frac{q_\phi(\alpha|x)}{p_\theta(\alpha|x)}\right]\right] \\
&= \underbrace{\mathbb{E}_{q_\phi(\alpha|x)}\left[\log\left[\frac{p_\theta(x,\alpha)}{q_\phi(\alpha|x)}\right]\right]}_{\mathcal{L}_{\theta,\phi}(x)\ \text{(ELBO)}} + \underbrace{\mathbb{E}_{q_\phi(\alpha|x)}\left[\log\left[\frac{q_\phi(\alpha|x)}{p_\theta(\alpha|x)}\right]\right]}_{D_{KL}(q_\phi(\alpha|x)\|p_\theta(\alpha|x))},
\end{aligned}
\tag{9}
$$

其中 $\mathbb{E}(\cdot)$ 表示期望算子，$D_{KL}$ 表示 Kullback-Leibler（KL）散度。式（9）中的第二项非负，因此第一项是数据对数似然的下界：

$$
\mathcal{L}_{\theta,\phi}(x) = \log p_\theta(x) - D_{KL}(q_\phi(\alpha|x)\|p_\theta(\alpha|x)) \le \log p_\theta(x)\,, \tag{10}
$$

这一项也称为证据下界 (证据下界)。根据式（10），我们可以看出，最大化证据下界 $\mathcal{L}_{\theta,\ \phi}$ 关于参数 $\theta, \phi$，相当于近似最大化边缘似然函数$p_\theta(x)$，并最小化近似分布$q_\phi(\alpha|x)$与真实后验分布$p_\theta(\alpha|x)$之间的KL散度。

从另一个角度来看，将证据下界（10）改写为

$$\mathcal{L}_{\theta,\phi}(x) = \mathbb{E}_{q_\phi(\alpha|x)}\left[\log\left[\frac{p_\theta(x,\alpha)}{q_\phi(\alpha|x)}\right]\right] = \mathbb{E}_{q_\phi(\alpha|x)}\left[\log\left[\frac{p_\theta(x|\alpha)p(\alpha)}{q_\phi(\alpha|x)}\right]\right]$$
$$= \underbrace{\mathbb{E}_{q_\phi(\alpha|x)}\left[\log p_\theta(x|\alpha)\right]}_{\mathcal{L}^{rec}(x)} + \underbrace{\left\{-D_{KL}\left[q_\phi(\alpha|x)\|p(\alpha)\right]\right\}}_{\mathcal{L}^{KL}(x)}. \tag{11}$$

因此，证据下界由“重构项”和“正则化项”组成。前者旨在使编码-解码方案尽可能高效，后者则通过使编码器返回的分布接近给定的先验分布$p(\alpha)$，对潜在空间进行正则化。为了最大化（11），需要计算其梯度$\nabla_{\theta,\phi}\mathcal{L}_{\theta,\phi}(x)$。然而，这些梯度无法以闭式形式求得，只能进行近似。对于连续潜变量，我们可以使用重参数化技巧 来计算$\nabla_{\theta,\phi}\mathcal{L}_{\theta,\phi}(x)$的无偏估计。令 $\epsilon$表示一个分布与$x$ 或 $\phi$无关的随机变量，则可将 $\alpha \sim q_\phi(\alpha|x)$表示为 $\epsilon$的可微且可逆变换： $\alpha = G(\epsilon,\phi,x)$。本文为 $\epsilon$指定标准高斯分布，并考虑一种简单的因子化高斯编码器，即，

$$q_\phi(\alpha|x) = \mathcal{N}(\mu_\phi(x), \mathrm{diag}(\sigma_\phi^2(x))), \quad \alpha = \mu_\phi(x) + \sigma_\phi(x)\odot\epsilon, \quad p(\epsilon) = \mathcal{N}(\mathbf{0}, \boldsymbol{I}_{N_\alpha}), \tag{12}$$

其中，$\mathrm{p}(\epsilon)$ 表示 $\epsilon$ 的概率密度函数，$\mu_\phi(\mathrm{x})$ 和 $\sigma_\phi(\mathrm{x})$ 由编码器神经网络计算得出，$\mathrm{diag(a)}$ 表示由向量 a 构造的对角矩阵，$\odot$ 表示逐元素乘积。总之，可以先从 $\mathrm{p}(\epsilon)$ 中采样 $\epsilon$，然后计算 $\alpha = \mathrm{G}(\epsilon,\phi,\mathrm{x}) = \mu_\phi(\mathrm{x}) + \sigma_\phi(\mathrm{x})\odot\epsilon$，从而生成一个来自 $\mathrm{q}_\phi(\alpha|\mathrm{x})$ 的样本 $\alpha$。我们为 $\mathrm{p}(\alpha)$ 指定标准高斯分布，即 $\mathrm{p}(\alpha) = \mathcal{N}(0, \mathrm{I}_{N_\alpha})$。此外，我们将解码器 $\mathrm{p}_\theta(\mathrm{x}|\alpha)$ 设为具有单位协方差矩阵的多元高斯分布，即 $\mathrm{p}_\theta(\mathrm{x}|\alpha) = \mathcal{N}(\mathcal{G}_\theta(\alpha), \mathrm{I}_{N_x})$，其中 $\mathcal{G}_\theta(\alpha)$ 是解码器网络的输出。因此，我们可以通过蒙特卡罗估计轻松近似重构项，

$$\mathcal{L}^{rec}(x) \approx \widetilde{\mathcal{L}^{rec}}(x) := \frac{1}{N_{\mathrm{rec}}}\sum_{k=1}^{N_{\mathrm{rec}}}\log p_\theta(x|\alpha^{(k)}) = \frac{1}{N_{\mathrm{rec}}}\sum_{k=1}^{N_{\mathrm{rec}}} -\frac{1}{2}\|\mathcal{G}_\theta(\alpha^{(k)}) - x\|_2^2 + \mathrm{constant},$$

其中 $\alpha^{(k)} \sim q_\phi(\alpha|x)$、$k = 1$、.....、$N_{\mathrm{rec}}$，最后一项为常数，在优化过程中可以忽略。此外，取$N_{\mathrm{rec}} = 1$后，我们可以构造一个简单的蒙特卡罗估计量，

$$\widetilde{\mathcal{L}^{rec}}(x) = -\frac{1}{2}\|\mathcal{G}_\theta(\alpha) - x\|_2^2 + \mathrm{constant} \propto -\|\mathcal{G}_\theta(\alpha) - x\|_2^2, \quad \epsilon \sim p(\epsilon), \quad \alpha = G(\epsilon,\phi,x), \tag{13}$$

由于$p(\alpha)$和$q_\phi(\alpha|x)$均为高斯分布，因此可以对 $\mathcal{L}^{KL}(x)$进行解析计算，

$$\mathcal{L}^{KL}(x) = \frac{1}{2}\sum_{l=1}^{N_\alpha} 1 + \log(\sigma_\phi(x))_l^2 - (\mu_\phi(x))_l^2 - (\sigma_\phi(x))_l^2,$$

其中，$(\mu_\phi(x))_l$ 和$(\sigma_\phi(x))_l$ 分别是向量$\mu_\phi(x)$和$\sigma_\phi(x)$的第$l$个元素。然后，单个数据点$x$ 对应的目标证据下界可以通过以下方式估计：

$$\widetilde{\mathcal{L}}_{\theta,\phi}(x) := \widetilde{\mathcal{L}^{rec}}(x) + \mathcal{L}^{KL}(x).$$

随后，可以利用自动微分轻松获得梯度 $\nabla_\theta\mathcal{L}_{\theta,\phi}(x), \nabla_\phi\mathcal{L}_{\theta,\phi}(x)$。数据集 $\boldsymbol{X}$ 的证据下界目标是每个数据点的证据下界之和，即 $\mathcal{L}_{\theta,\phi}(\boldsymbol{X}) = \sum_{x\in\boldsymbol{X}}\mathcal{L}_{\theta,\phi}(x)$。为了高效地最小化$-\mathcal{L}_{\theta,\phi}(\boldsymbol{X})$，我们采用小批量随机梯度下降（SGD）方法。

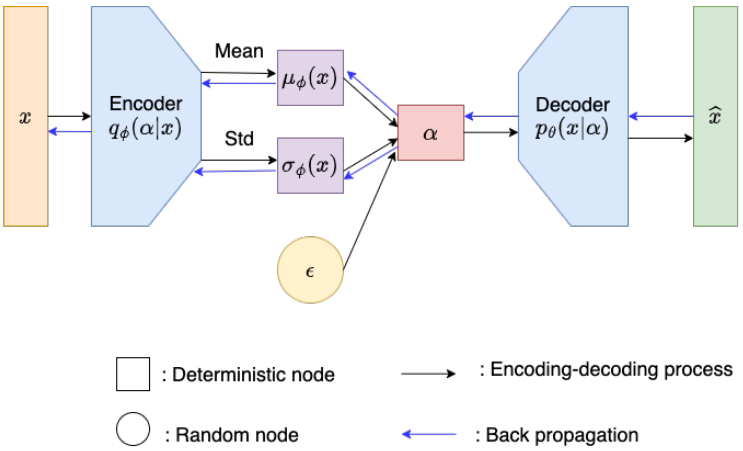



图1：原始变分自编码器示意图。

在训练过程中，每个训练轮次开始时，我们首先从完整数据集 $\boldsymbol{X}$ 中随机抽取一个数据集批次 $\boldsymbol{X}^b$，批量大小为 $N_b$。然后，对于 $\boldsymbol{X}^b$中的每个数据点 $x^{(k)}$，我们从 $p(\epsilon)$中随机采样噪声 $\epsilon^{(k)}$。接着，我们可以使用编码器和式（12）计算 $x^{(k)}$ 对应的 $\alpha^{(k)}$。$\boldsymbol{X}^b$ 的目标可近似为

$$
\begin{aligned}
-\widetilde{\mathcal{L}}_{\theta,\phi}(\boldsymbol{X}^b) &= -\sum_{k=1}^{N_b} \widetilde{\mathcal{L}}_{\theta,\phi}(x^{(k)}) = -\sum_{k=1}^{N_b} \widetilde{\mathcal{L}^{rec}}(x^{(k)}) + \mathcal{L}^{KL}(x^{(k)}) \\
&= \sum_{k=1}^{N_b} \left[ \|\mathcal{G}_\theta(\alpha^{(k)}) - x^{(k)}\|_2^2 - \frac{1}{2} \sum_{l=1}^{N_\alpha} 1 + \log(\sigma_\phi(x^{(k)}))_l^2 - (\mu_\phi(x^{(k)}))_l^2 - (\sigma_\phi(x^{(k)}))_l^2 \right] .
\end{aligned} \tag{14}
$$

最后，我们可以使用随机梯度下降更新 $\theta$ 和 $\phi$。原始变分自编码器算法 [12] 的训练过程总结于算法2中。原始变分自编码器的示意图如图1所示。

在这项工作中，我们仅关注解码器网络的均值。总之，训练过程结束后，若要从目标分布中生成样本，可以先从给定的简单低维概率分布 $p(\alpha)$中采样 $\alpha \in \mathbb{R}^{N_\alpha}$，然后将其输入训练好的确定性生成模型，以获得生成样本$x$，即 $x = \mathcal{G}_{\theta^\star}(\alpha)$。因此，对 $x$ 的推断已转变为对 $\alpha$的推断，

$$
\pi(\alpha|\boldsymbol{d}) \propto \boldsymbol{L}(\boldsymbol{d}|\mathcal{G}_{\theta^\star}(\alpha))p(\alpha) . \tag{15}
$$

总体而言，通过深度生成先验推断高维参数$x$ 大致包含以下步骤：(a) 利用训练数据$\boldsymbol{X}$，可以通过算法2学习确定性生成器 $\mathcal{G}_{\theta^\star}$；(b) 给定观测数据$\boldsymbol{d}$、生成器 $\mathcal{G}_{\theta^\star}$、变更后的前向模型$F(\mathcal{G}_{\theta^\star}(z))$以及先验分布$p(\alpha) = \mathcal{N}(\mathbf{0},\boldsymbol{I}_{N_\alpha})$，可以使用算法1为变更后的后验分布(15)生成后验潜变量样本 $\{\alpha^{(k)}\}^{N_C}{}_{k=1}$；(c) 利用后验潜变量样本，可以通过生成器 $\mathcal{G}_{\theta^\star}$生成后验样本 $\{x^{(k)}\}^{N_C}{}_{k=1}$，进而计算后验均值和后验方差等后验统计信息。为便于区分，我们将训练数据集定义在全局域上的原始变分自编码器称为全局*VAE* 方法（G-VAE）。此外，我们将采用G-VAE先验的全局反演称为全局-VAE-马尔可夫链蒙特卡罗（G-VAE-MCMC）方法。图2展示了该框架的示意图，其中后验均值和方差的定义见第4.4节。

**算法2** 原始变分自编码器算法

**输入：** 训练数据集 $\boldsymbol{X}$、编码器$q_\phi(\alpha|x)$、解码器 $p_\theta(x|\alpha)$、最大训练轮数 $N_e$，以及批量大小 $N_b$。

初始化编码器和解码器网络的 $\theta$ 和 $\phi$。

将 $\boldsymbol{X} = \{x^{(k)}\}^N_{k=1}$ 划分为 $n_b$ 个小批次 $\{\boldsymbol{X}^{i_b}\}^{n_b}_{i_b=1}$.

**对于** $t = 1, \ldots, N_e$ **do**

  **对于** $i_b = 1, \ldots, n_b$ **执行**

    随机从$\boldsymbol{X}$ 中抽取大小为$N_b$的小批量训练数据集$\boldsymbol{X}^{i_b}$。

    为$\boldsymbol{X}^{i_b}$中的每个数据点$x^{(k)}$ 随机抽取噪声 $\epsilon^{(k)} \sim p(\epsilon)$。

    使用编码器网络为$\boldsymbol{X}^{i_b}$ 中的每个数据点$x^{(k)}$ 计算潜变量 $\alpha^{(k)}$ （见（12））。

    计算 $-$ 根据式 (14) 计算 $\mathcal{L}_{\theta,\ \phi}(\boldsymbol{X}^{i_b})$ 及其梯度 $\nabla_\theta - \widetilde{\mathcal{L}}_{\theta,\phi}(\boldsymbol{X}^{i_b})$, $\nabla_\phi - \widetilde{\mathcal{L}}_{\theta,\ \phi}(\boldsymbol{X}^{i_b})$。

    使用随机梯度下降优化器更新 $\theta$ 和 $\phi$。

  **结束循环**

**结束循环**

**输出：** 训练得到的编码器网络 $q_{\phi^\star}(\alpha|x)$ 和解码器网络 $p_{\theta^\star}(x|\alpha)$。

### 3. 一种 DD-VAE-MCMC 方法

正向系统(8)通常可以通过经典数值方法求解，例如有限元方法（FEM） [17]。然而，相应的反问题仍然具有挑战性。求解这类适定性不良问题时，主要存在三个瓶颈。第一个瓶颈是反演过程中高昂的计算成本。马尔可夫链蒙特卡罗为确保收敛所需的混合时间通常很长，因此需要进行大量正向计算。采用传统数值方法求解前向模型时，计算成本可能非常高。因此，人们提出了代理模型 [22, 23, 24] ，以牺牲计算精度为代价缓解这一问题。第二个瓶颈是所谓的维数灾难。离散化向量$x$ 通常是高维的，因此会造成极大的采样困难。包括Karhunen-Lo`ève（KL）展开 [25],在内的参数化方法应运而生，以应对这一问题。即便如此，维度问题只是得到缓解，并未彻底消除。正如 [26], 所指出的，特征值的衰减率取决于空间域的相对长度。在我们先前的工作 [27], 中，我们提出了局部KL展开方法来解决维数问题。第三个瓶颈是难以描述先验分布，因为先验分布的信息通常没有封闭形式，只能通过历史样本获得。

在第2.3节中，我们回顾原始变分自编码器的基本设置和训练过程。我们可以利用变分自编码器的生成属性，从给定的训练数据中编码先验信息。确定性生成器定义了一个从潜变量空间到复杂先验分布空间的映射。变分自编码器虽然颇具吸引力，但这类深度模型的主要局限之一是训练神经网络所需的计算成本极高。此外，从反问题的角度来看，它仍然存在维数问题：即使潜变量的维度大幅降低，对于基于马尔可夫链蒙特卡罗的

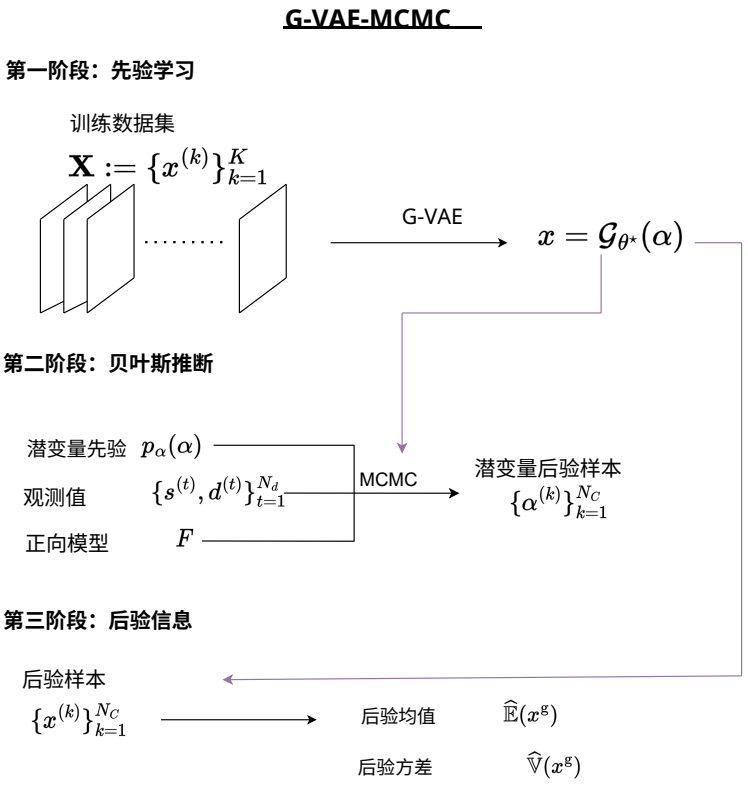



图2：G-VAE-MCMC方法示意图。

采样方法。

在本研究中，我们开发了一种域分解变分自编码器马尔可夫链蒙特卡罗（DD-VAE-MCMC）方法，以同时解决这些问题。我们采用的域分解策略的详细设置见第3.1节。第3.2节介绍我们提出的域分解变分自编码器（DD-VAE）方法。第3.3节讨论用于处理未知界面的自适应高斯过程方法。第3.4节回顾图像融合技术，并在第3.5节总结我们的整体DD-VAE-MCMC算法。

### *3.1.* 域分解设置

我们的物理域$D \subset \mathbb{R}^{N_D}$被分解为$M$ ($M > 1$个重叠子域，使得

$$\overline{D} = \overline{D_1} \cup \overline{D_2} \cup \cdots \cup \overline{D_M},$$

其中，上横线表示闭包。对于每个子域 $\mathrm{D}_i$（$\mathrm{i} = 1, \ldots, \mathrm{M}$），其边界集合记为$\partial \mathrm{D}_i$，其相邻子域索引集合记为 $\mathrm{N}_i := \{\mathrm{j} | \mathrm{j} \in \{1, \ldots, \mathrm{M}\}, \mathrm{j} \neq \mathrm{i}, \text{且 } \mathrm{D}_i \cap \mathrm{D}_j \neq \emptyset\}$。将域分解引入的界面记为 $\partial \mathrm{jD}_i := \partial \mathrm{D}_i \cap \mathrm{D}_j$，边界集合 $\partial \mathrm{D}_i$可分为两部分： $\partial \mathrm{D}_i = \partial_{ex} \mathrm{D}_i \cup \partial_{in} \mathrm{D}_i$，其中 $\partial_{ex} \mathrm{D}_i := \partial \mathrm{D}_i \cap \partial \mathrm{D}$ 为外部边界， $\partial_{in} \mathrm{D}_i := \cup_{\mathrm{j} \in \mathrm{N}_i} \{\partial_j \mathrm{D}_i\}$ 为内部边界。汇集与所有子域 $\{\mathrm{D}_i\}_{i=1}^{M}$相关的全部界面索引，定义 $\mathrm{N} := \{(\mathrm{i}, \mathrm{j}) | \mathrm{i} \in \{1, 2, \ldots, \mathrm{M}\} \text{ 且 } \mathrm{j} \in \mathrm{N}_i\}$。图 3 展示了重叠双子域系统的符号示意图。

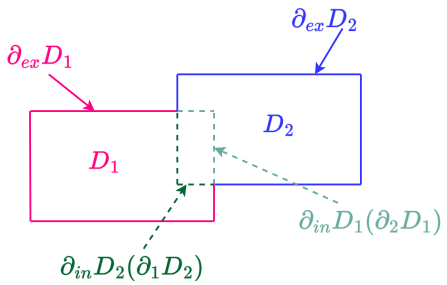



图3：重叠双子域系统示意图：$D = D_1 \cup D2$, $\partial_{in}D_1 = \partial D_1 \cap \overline{D}_2 = \overline{\partial_2 D1}$, $\overline{\partial_{in}D_2} = \partial D_2 \cap D_1 = \partial_1 D_2$,$\partial_{ex}D_1 = \partial D_1 \cap \partial D$, $\partial_{ex}D_2 = \partial D_2 \cap \partial D$, $N_1 = \{2\}$、$N_2 = \{1\}$ 和$N = \{(1,2),(2,1)\}$。

我们引入分解后的局部算子 $\{\mathcal{L}_i := \mathcal{L}|_{D_i}\}_{i=1}^{M}$, $\{\mathcal{B}_i := \mathcal{B}|_{\partial_{ex}D_i}\}_{i=1}^{M}$ 和局部函数 $\{f_i := f|_{D_i}\}_{i=1}^{M}$,$\{g_i := g|_{D_i}\}_{i=1}^{M}$，它们是限制到各子域 $D_i$ 上的全局算子和函数。x 在各子域上的限制记为 $x_i := x|_{D_i}$。$x_i$ 的维度记为 $N_{x_i}$。本文考虑每个子域上的随机变量维度相同的情形，即 $N_{x_1} = \cdots = N_{x_M}$。假设 $\{x_i\}_{i=1}^{M}$ 的像分别记为 $\{\Gamma_i\}_{i=1}^{M}$，其概率密度函数记为 $p_{x_i}(x_i)$；为避免歧义，简记为 $p(x_i)$。对于 $(i, j) \in N$，$h_{ij}$ 表示定义在界面 $\partial_j D_i$ 上的界面函数，本文将其定义为 $h_{ij}(s, x) := u(s, x)|_{\partial_j D_i}$，其中 $u(s, x)$ 是全局问题 (8) 的解。注意，界面函数取决于随机输入 x。于是，每个局部问题定义如下：对于 $i = 1, \ldots, M$，求 $u_i(s, x_i) : D_i \times \Gamma_i \to \mathbb{R}$，使得

$$\mathcal{L}_i(s, x_i; u_i(s, x_i)) = f_i(s), \quad \forall (s, x_i) \in D_i \times \Gamma_i, \tag{16a}$$

$$\mathcal{B}_i(s, x_i; u_i(s, x_i)) = g_i(s), \quad \forall (s, x_i) \in \partial_{ex}D_i \times \Gamma_i, \tag{16b}$$

$$\mathcal{B}_{ij}(s, x_i; u_i(s, x_i)) = h_{ij}(s, x), \quad \forall (s, x_i) \in \partial_j D_i \times \Gamma_i, j \in \mathfrak{N}_i, \tag{16c}$$

其中，(16c)定义了界面上的边界条件，$\mathcal{B}_{ij}$ 是作用于界面 $\partial_j D_i$上的适当边界算子。关于界面函数的详细讨论见 [28, 29, 26]。

### *3.2. DD-VAE方法*

将神经网络与域分解相结合的思路目前正得到积极发展。Jagtap等人基于域分解提出了扩展物理信息神经网络（XPINNs） [30] 和并行物理信息神经网络（并行PINNs） [31]。正如他们所指出的，域分解策略在基于深度学习的方法中的优势包括并行化能力、强大的表示能力以及高效的超参数调优。 [32] 和 [33] 采用深度Ritz方法，在各子域中构建神经网络，发展了深度域分解方法。受益于域分解策略所带来的优势，本文提出一种域分解变分自编码器（DD-VAE）方法，将域分解的思想融入变分自编码器。

然而，与求解偏微分方程的大多数基于区域分解的方法不同，例如 [30, 33], ，神经网络的数量与子域的数量一致，因为每个神经网络都需要包含不同的

物理现象。在深度生成先验的背景下，训练过程不受边界条件约束，因此可以采用堆叠方式进行。对于原始训练数据集 X，我们将每个训练数据点 $\mathrm{x}^{(k)}$ 划分为 $\{\mathrm{x}^{(k)}\mathrm{i}\ \}_{i=1}^{M}$，其中 $\mathrm{x}^{(k)}\mathrm{i} := \mathrm{x}^{(k)}|_{D_i}$ 是根据全局域的分解策略得到的（见第3.1节），由此可以获得每个子域上的训练数据集 $\mathrm{X}_i := \{\mathrm{x}^{(k)}\mathrm{i}\ \}_{k=1}^{K}$，其中 $\mathrm{i} = 1, \ldots, \mathrm{M}$。我们假设，在维度一致的情况下，先验分布中的任意有限个随机变量都服从相同的分布。具体而言，x 是一个随机变量集合，其概率密度函数为 p(x)；$\mathrm{x}_i$、$\mathrm{x}_j$ 是从 x 中取出的两个具有相同有限数量的随机变量。因此，假设 $\mathrm{x}_i$ 和 $\mathrm{x}_j$ 服从相同的分布。在本研究中，我们考虑局部随机变量维度相同的情况。这样，训练得到的变分自编码器便可在全局域内共享。为保持一致，将子域上的随机变量记为 y，将局部域的目标分布记为 p(y)，即 $\mathrm{p}_{x_1}(\mathrm{x}_1) = \ldots = \mathrm{p}_{x_M}(\mathrm{x}_M) = \mathrm{p}_y(\mathrm{y})$；在没有歧义时，下标可以省略。因此，将所有子域的训练数据点汇集为一个增强训练数据集 $\mathrm{Y} := \{\mathrm{y}^{(k)}|\mathrm{y}^{(k)} \in \mathrm{X}_i$，其中 $\mathrm{i} = 1, \ldots, \mathrm{M}, \mathrm{k} = 1, \ldots, \mathrm{K}\}$。

基于上述假设，我们不再针对$\boldsymbol{X}_i$训练不同的生成模型，而是基于增强训练数据集$\boldsymbol{Y}$训练一个变分自编码器，称为域分解变分自编码器（DD-VAE）。$\boldsymbol{Y}$ 包含$MK$个维度为$N_y := N_{x_1} = \cdots = N_{x_M}$的训练数据点。为简便起见，$\boldsymbol{Y}$重写为$\boldsymbol{Y} = \{y^{(k)}\}_{k=1}^{MK}$。遵循第2.3节中的设置，我们构建 DD-VAE，利用训练数据$\boldsymbol{Y}$ 学习局部先验的局部生成模型。具体而言，此处的目标是利用概率模型$p_\psi(y)$学习分布$p(y)$。我们用 $\beta \in \mathbb{R}^{N_\beta}$ 表示 DD-VAE 中的潜变量。此外，编码器、解码器和确定性生成器分别记为$q_\chi(\beta|y)$、$p_\psi(y|\beta)$和$\mathcal{H}_{\psi^\star}(\beta)$。

除了降维之外，使用我们的 DD-VAE 进行先验建模还具有以下优势：（1）训练数据集的规模大幅扩大；（2）对于区域规模较小的训练任务，网络可以采用更浅层的结构；（3）通过叠加局部先验，整体表征能力得以提升。总之，得益于更大的训练集和更简单的网络结构，训练效率大幅提高。

本文使用全连接神经网络（FCNN）。设 $\mathcal{F}: \mathbb{R}^{N_i} \to \mathbb{R}^{N_o}$ 是一个具有 $L$ 层的前馈神经网络，其中 $N_i$ 和 $N_o$ 分别表示输入和输出的维度。输入变量记为 $\eta \in \mathbb{R}^{N_i}$，第 $\ell$ 层记为 $\mathcal{F}_\ell: \mathbb{R}^{N_{\ell-1}} \to \mathbb{R}^{N_\ell}$ ($N_0 = N_i$ 和 $N_L = N_o$)。第 $\ell$ 层（$1 \le \ell \le L$）的权重矩阵和偏置向量分别记为 $W_\ell$ 和 $b_\ell$，整体神经网络由 $L$ 层组合而成 $\mathcal{F}(\eta) = \mathcal{F}_L \circ \cdots \circ \mathcal{F}_1(\eta)$，其中

$$\mathcal{F}_1(\eta) = \Phi(W_1\eta + b_1), \quad \mathcal{F}_\ell = \Phi(W_\ell\mathcal{F}_{\ell-1} \circ \cdots \mathcal{F}_1(\eta) + b_\ell),$$

其中 $\Phi(\cdot)$ 表示激活函数。将所有参数 $\Theta := \{W_\ell, b_\ell\}_{\ell=1}^{L}$汇集起来，参数化神经网络记为 $\mathcal{F}_\Theta$。本文中，编码器和解码器均采用全连接神经网络。

### *3.3.* 自适应高斯过程回归界面处理

空间域分解后，需要指定局部问题 (16c) 的界面条件。我们提出了一种用于界面条件的自适应高斯过程回归方法 [27], ，并在此给出该方法以供参考。

将所有观测数据组成的集合记为 $\mathcal{D} = \{(s^{(t)}, d^{(t)})\}_{t=1}^{N_d}$，其中 $s^{(t)}$ 是第 t 个传感器的位置，$d^{(t)} \in \mathbb{R}$ 是在 $x^{(t)}$ 处采集的观测值。所有传感器位置组成的集合定义为 $s := \{s^{(t)} | (s^{(t)}, d^{(t)}) \in \mathcal{D}\}$，观测数据集合记为 $d = [d^{(1)}, \ldots, d^{(N_d)}]$。对于每个子域 $D_i$（$i = 1, \ldots, M$），局部观测数据组成的集合为 $\mathcal{D}_i := \{(s^{(t)}, d^{(t)}) | (s^{(t)}, d^{(t)}) \in \mathcal{D}$，且 $s^{(t)} \in D_i \cup \partial D_i\}$；集合 $\mathcal{D}_i$ 的大小记为 $N_{d_i} := |\mathcal{D}_i|$。类似地，$s_i := \{s^{(t)} | (s^{(t)}, d^{(t)}) \in \mathcal{D}_i\}$，而 $d_i \in \mathbb{R}^{N_{d_i}}$ 汇集了 $\mathcal{D}_i$中包含的观测数据。

对于任意 $(i, j) \in \mathbb{N}$，都需要指定合适的界面函数 $\{h_{ij}(s, x)\}_{(i,j)\in\mathbb{N}}$。界面函数实际上依赖于全局随机参数 $x$。真实值 $x$ 对应的界面函数定义为 $h_{ij}(s) := h_{ij}(s, x)$。基于有限的观测数据，本文提出了一种自适应高斯过程（GP）[34]策略，用于 [27]近似其概率形式中的 $h_{ij}$。

假设用于近似目标函数 $h_{ij}(s)$ 的训练数据集记为 $\Lambda_{ij} = \{(s^{(t)}, d^{(t)})\}_{t=1}^{N_{d_{ij}}}$，其大小为 $N_{d_{ij}} := |\Lambda_{ij}|$。由 $\Lambda_{ij}$ 中的传感器位置构成的集合记为 $s_{ij} := \{s^{(t)} | (s^{(t)}, d^{(t)}) \in \Lambda_{ij}\}$，而 $d_{ij} \in \mathbb{R}^{N_{d_{ij}}}$ 汇集了 $\Lambda_{ij}$ 中的所有观测值。高斯过程是一组随机变量，其任意有限组合都服从联合高斯分布。其基本思想是假设目标函数 $h_{ij}(s)$ 是一个均值函数为 $m(s)$、协方差由核函数 $\mathcal{K}(s, s')$指定的高斯随机场的一个实现，即先验高斯过程模型记为 $h_{ij}(s) \sim \mathcal{GP}(m(s), \mathcal{K}(s, s'))$。给定训练数据后，我们希望预测任意位置 s 处 $h_{ij}(s)$ 的值，该值也服从高斯分布，

$$\widehat{h}_{ij}(s) | \Lambda_{ij} \sim \mathcal{N}(m_{N_{d_{ij}}}(s), \sigma_{N_{d_{ij}}}(s)), \tag{17}$$

其中

$$m_{N_{d_{ij}}}(s) = k_\star^T (K_{N_{d_{ij}}} + \sigma_{\text{obs}}^2 \boldsymbol{I}_{N_{d_{ij}}})^{-1} \boldsymbol{d}_{ij}, \tag{18a}$$

$$\sigma_{N_{d_{ij}}}(s) = k(s, s) - k_\star^T (K_{N_{d_{ij}}} + \sigma_{\text{obs}}^2 \boldsymbol{I}_{N_{d_{ij}}})^{-1} k_\star, \tag{18b}$$

其中$k_\star \in \mathbb{R}^{N_{d_{ij}}}$，其各项定义为（$k_\star$）$_t = \mathcal{K}$（$s^{(t)}$，$s^{(t)}$）对于$s^{(t)} \in \boldsymbol{s}_{ij}$；$K^{N_{d_{ij}}}$ 是协方差矩阵，其各项为[$K_{N_{d_{ij}}}$]$_{rt} = \mathcal{K}$（$s^{(r)}$，$s^{(t)}$），对于$s^{(r)}$、$s^{(t)} \in \boldsymbol{s}_{ij}$ 以及$r$、$t$ = 1、.....、$N^{d_{ij}}$（参见 [34]）。

从认知角度而言，高斯过程模型的效率在很大程度上取决于训练数据集 $\Lambda_{ij}$。因此，为了最大限度地发挥高斯过程模型的潜力，我们采用自适应方式构建高斯过程模型。首先，从$\mathcal{D}_i \cup \mathcal{D}_j$中随机选择一个数据对 {$s^{(t)}$、$d^{(t)}$}，构建初始训练数据集 $\Lambda_{ij}$。同时，构建测试数据集$\Delta_{ij} \subset \partial_j D_i$。随后，利用当前训练数据集初始化高斯过程模型（17）。其次，使用（18b）计算当前高斯过程模型在每个测试点$s \in \Delta_{ij}$ 处的方差，并将方差最大的测试点记为

$$\bar{s} := \arg\max_{s \in \Delta_{ij}} \sigma_{N_{d_{ij}}}(s). \tag{19}$$

然后，我们追踪与 $s$ 最近的观测值的位置，即，

$$s^\star := \arg\min_{s \in \boldsymbol{s}} \|s - \bar{s}\|_2.$$

在第三步中，将数据对（$s^\star$，$d^\star$）增广到训练数据集 $\Lambda_{ij}$ 中，其中 $d^\star$ 是从 $s^\star$处收集的观测值。随后，使用当前训练数据集更新高斯过程模型。重复第二步和第三步，直到

测试数据集的最大后验方差小于给定阈值 $\delta_{\text{tol}}$。该过程在第3.5节的主算法中表示。

利用界面高斯过程模型，第3.1节讨论的局部问题（16）可重新表述如下：对于$i = 1, \ldots, M$，求$u_i^{\text{GP}}(s,x_i) : D_i \times \Gamma_i \to \mathbb{R}$，使得

$$\mathcal{L}_i(s, x_i; u_i^{\text{GP}}(s, x_i)) = f_i(s), \quad \forall (s, x_i) \in D_i \times \Gamma_i, \tag{20a}$$

$$\mathcal{B}_i(s, x_i; u_i^{\text{GP}}(s, x_i)) = g_i(s), \quad \forall (s, x_i) \in \partial_{ex} D_i \times \Gamma_i, \tag{20b}$$

$$\mathcal{B}_{ij}(s, x_i; u_i^{\text{GP}}(s, x_i)) = m_{N_{d_{ij}}}(s), \quad \forall (s, x_i) \in \partial_j D_i \times \Gamma_i, j \in \mathfrak{N}_i, \tag{20c}$$

其中$m^{N_{d_{ij}}}(s)$是式（18）中定义的界面高斯过程模型的均值函数。至此，与式（20）关联的局部正向模型记为$F_i^{\text{GP}}(x_i) := \mathsf{O}_i(u_i^{\text{GP}})$，其中$\mathsf{O}_i$是局部观测算子。

### *3.4.* 图像融合

给定局部样本$x_i, i = 1, \ldots, M$，直接将局部样本拼接起来以获得全局样本，可能会在界面处产生明显的接缝。为解决这一问题，我们采用计算机视觉中广泛使用的图像融合技术。本文重点介绍泊松融合 [35] 方法（也称为梯度域合成），以无缝融合多个样本。泊松融合方法的核心思想是，拉普拉斯算子能够表征变化缓慢的强度梯度，而将这些梯度叠加到图像上几乎不会产生明显影响。随后， [35] 将原始图像编辑问题表述为带狄利克雷边界条件的泊松偏微分方程。求解相应的偏微分方程即可得到融合图像。

设$C \subset \mathbb{R}^2$为图像定义域，$\Omega$是$C$的一个闭子集，其边界为 $\partial\Omega$。令$B$表示源图像，即我们要插入 $\Omega$ 中的图像；令$\mathbf{v}$表示其梯度，也称为引导场。令$I^\star$表示定义在$C$去掉 $\Omega$ 内部后的区域上的原始图像，令$I$表示定义在 $\Omega$ 内部的生成图像的未知标量函数。图 4 展示了详细设置。

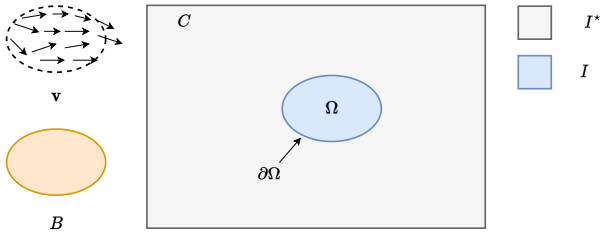



图 4：泊松图像融合方法示意图。

泊松图像编辑技术插入一个向量$\mathbf{v}$，以融合源图像中的像素，因此可以通过以下最小化问题实现两幅图像的融合：

$$\min_I \iint_\Omega |\nabla I - \mathbf{v}|^2, \quad I|_{\partial\Omega} = I^\star|_{\partial\Omega},$$
$$\mathbf{v} = \nabla g.$$

上述优化问题的解是以下带狄利克雷边界条件的泊松方程的唯一解：

$$\Delta I = \Delta B \text{ over } \Omega, \quad \text{with} \quad I|_{\partial\Omega} = I^{\star}|_{\partial\Omega}, \tag{21}$$

其中，边界条件约束生成图像的像素强度，使其等于目标图像的边界像素。通过最小化融合图像的梯度，源图像 $B$被克隆到目标图像中，呈现出自然的效果。

接下来，我们可以将 (21) 表示为离散形式。对于 $C$ 中的每个像素 $s$，令$C$中与其相邻的四邻域像素构成集合$N_s$，并令 $\langle s,s'\rangle$ 表示一对像素，满足 $s' \in N_s$。于是，$\Omega$ 的边界表示为 $\partial\Omega = \{s \in C/\Omega : N_s \cap \Omega \neq \emptyset\}$。令 $I(s)$ 表示$I$ 在像素$s$处的强度。(21) 的离散形式为

$$\min_{I|_{\Omega}} \sum_{\langle s,s'\rangle \cap \Omega \neq 0} (I(s) - I(s') - v_{s,s'})^2, \quad \text{with } I(s) = I^{\star}(s) \text{ for all } s \in \partial\Omega,$$

其中，$v_{s,s'}$ 是源图像的一阶导数。因此，生成图像中 $\Omega$ 内每个像素的强度由以下方程确定：

$$|N_s| I(s) - \sum_{s' \in N_s \cap \Omega} I(s') = \sum_{s' \in N_s \cap \partial\Omega} I^{\star}(s') + \sum_{s' \in N_s} v_{s,s'}, \quad \forall s \in \Omega.$$

求解离散泊松方程后，即可生成无缝图像。由此可见，泊松图像融合技术实际上是一个灵活的框架，可通过不同设置实现不同的融合方式；本研究采用的具体设置见第4.4节。给定局部样本 $\{x_i\}_{i=1}^{M}$，我们将融合图像记为$x_{\text{ble}}$。

### *3.5. DD-VAE-MCMC 方法*

本节详细介绍 DD-VAE-MCMC 方法的具体实现。首先，我们将全局域 D 划分为 M 个大小相同且彼此重叠的局部域 $\{\mathrm{D}_i\}_{i=1}^{M}$（见第3.1节）。第二阶段是先验学习。根据区域划分策略，我们将原始训练数据集 $\mathrm{X} = \{\mathrm{x}^{(k)}\}_{k=1}^{K}$ 重新排列为 $\mathrm{Y} = \{\mathrm{y}^{(k)}\}_{k=1}^{MK}$。利用 Y，我们可以通过 DD-VAE 得到局部生成器 $\mathcal{H}_{\psi^{\star}}$（见第3.2节）。第三阶段是使用自适应高斯过程模型为局部问题构造界面条件（见第3.3节）。第四阶段，对于每个局部子域 $\mathrm{D}_i$（$\mathrm{i} = 1, \ldots, \mathrm{M}$），使用算法1，结合局部正向模型 $\mathrm{F}_i^{\mathrm{GP}}(\mathrm{x}_i)$（通过式（20）定义）、局部观测值 $\mathrm{s}_i$ 和先验分布 $\mathrm{p}(\beta) = \mathcal{N}(0, \mathrm{I}_{N_\beta})$，生成局部潜变量后验样本$\{\beta^{(k)}\,\mathrm{i}\}, \mathrm{k} = 1, \ldots, \mathrm{N}_C$。最后一个阶段是后处理。给定局部潜变量后验样本 $\{\beta^{(k)}\mathrm{i}\ \}^{N_C}\mathrm{k}{=}1$（$\mathrm{i} = 1, \ldots, \mathrm{M}$），我们首先利用局部生成器 $\mathcal{H}_{\psi^{\star}}$生成局部后验样本 $\{\mathrm{x}^{(k)}\mathrm{i}\ \}^{N_C}\mathrm{k}{=}1$（$\mathrm{i} = 1, \ldots, \mathrm{M}$）。然后，使用泊松图像融合技术（见第3.4节）构造全局后验样本 $\{\mathrm{x}^{(k)}\,\mathrm{ble}\}^{N_C}\mathrm{k}{=}1$。给定全局后验样本后，可以计算其后验统计信息。图5展示了 DD-VAE-MCMC 方法的示意图。算法3总结了 DD-VAE-MCMC 方法的详细步骤，其中 $\delta_{\text{tol}}$ 表示给定的容差。

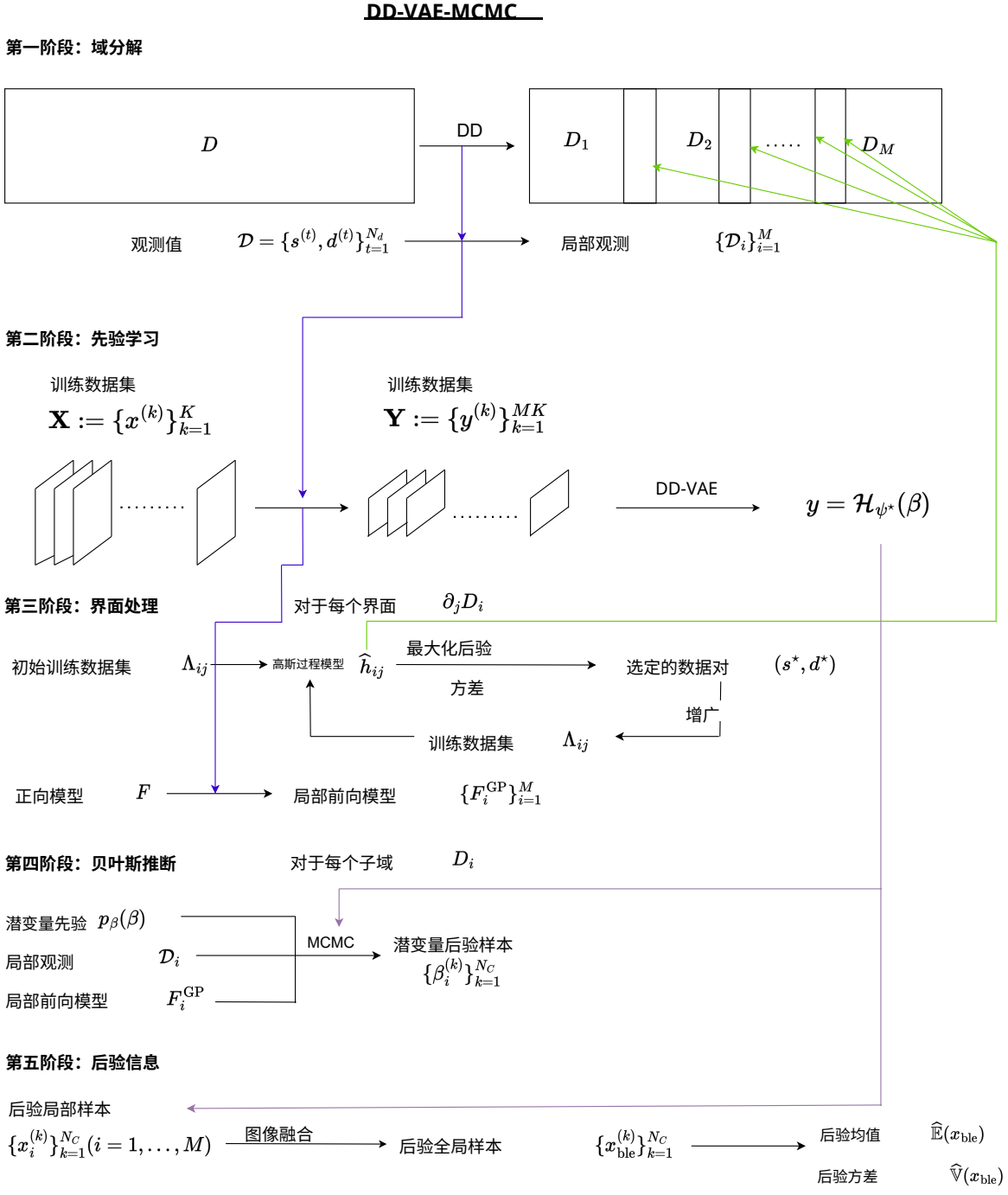



图5：DD-VAE-MCMC 方法示意图。

算法3 DD-VAE-MCMC算法 输入：训练数据集 $X = \{x^{(k)}\}_{k=1}^{N}$、全局域 D，以及观测数据 $\mathcal{D} = \{(s^{(t)}, d^{(t)})\}_{t=1}^{N_d}$。1：将全局域 D 划分为 M 个相互重叠的局部域 $\{D_i\}_{i=1}^{M}$，其界面为 $\partial_j D_i$，其中 $j \in N_i$（具体设置见第3.1节）。2：将观测数据集 $\mathcal{D}$ 划分为 $\{\mathcal{D}_i\}_{i=1}^{M}$，其中 $\mathcal{D}_i := \{(s^{(t)}, d^{(t)}) | (s^{(t)}, d^{(t)}) \in \mathcal{D}$，且 $s^{(t)} \in D_i \cup \partial D_i\}$。3：将训练数据集 X 重新排列为 Y（详情见第3.2节）。4：使用算法2，基于经过调整的训练数据集 Y、编码器 $q_\chi(\beta|y)$、解码器 $p_\psi(y|\beta)$、最大训练轮数 $N_e$和批量大小 $N_b$，获得局部生成器 $\mathcal{H}_{\psi^\star}$。5：对于每个界面 $\partial_j D_i$（$(i, j) \in N$），执行以下操作：6：用 $\mathcal{D}_i \cup \mathcal{D}_j$ 中任意一个数据点初始化训练集 $\Lambda_{ij}$。7：构建有限测试集 $\Delta_{ij} \subset \partial D_i$。8：建立高斯过程模型 $h_{ij}(s)|\Lambda_{ij} \sim \mathcal{N}(m^{N_{d_{ij}}}(s), \sigma^{N_{d_{ij}}}(s))$（见式（18））。9：获得最大后验方差 $\sigma^{\max}_{\Delta_{ij}} := \max_{s\in\Delta_{ij}} \sigma^{N_{d_{ij}}}(s)$。10：当 $\sigma^{\max}_{\Delta_{ij}} \geq \delta_{tol}$ 时，执行以下操作：11：求取 $s := \arg\max_{s\in\Delta_{ij}} \sigma^{N_{d_{ij}}}(s)$。12：求取 $s^\star := \arg\min_{s\in\mathbf{s}} \|s - s\|_2$。13：更新训练数据集 $\Lambda_{ij} = \Lambda_i j \cup \{(s^\star), d^\star\}$，其中 $d^\star$是在 $s^\star$处采集的观测值。14：更新高斯过程模型 $h_{ij}(s)|\Lambda_{ij} \sim \mathcal{N}(m^{N_{d_{ij}}}(s), \sigma^{N_{d_{ij}}}(s))$（见式（18））。15：获得最大后验方差 $\sigma^{\max}_{\Delta_{ij}} := \max_{s\in\Delta_{ij}} \sigma^{N_{d_{ij}}}(s)$。16：结束 while 循环。17：结束 for 循环。18：构建局部正向模型 $F_i^{GP}(x_i)$，其中 $i = 1, \ldots, M$（见式（20））。19：对于 $i = 1, \ldots, M$，执行以下操作：20：使用算法1，基于局部模型 $F_i^{GP}(\mathcal{H}_{\psi^\star}(\beta))$、局部观测数据 $\mathcal{D}_i$ 以及局部先验分布 $p(\beta) = \mathcal{N}(0, I^{N_\beta})$，获得局部潜变量后验样本 $\{\beta^{(k)}\}_{k=1}^{N_C}$。21：使用 $\{\beta^{(k)}\}_{k=1}^{N_C}$ 和局部生成器 $\mathcal{H}_{\psi^\star}$，获得局部后验样本 $\{x^{(k)}\}_{k=1}^{N_C}$。22：结束 for 循环。23：使用泊松融合技术（见第3.4节）获得全局后验样本 $\{x^{(k)}\}_{k=1}^{N_C}$。输出：全局后验样本 $\{x^{(k)}\}_{k=1}^{N_C}$。

## 4. 数值示例

本节通过数值示例展示所提出的 DD-VAE- MCMC 方法的有效性。测试问题的设置列于第4.1节。第4.2节讨论变分自编码器的生成特性。第4.3节介绍高斯过程（GP）接口处理的效果。反演结果见第4.4节。

### *4.1.* 测试问题的设置

考虑矩形区域$D = (0, 2) \times (0, 1) \subset \mathbb{R}^2$上的椭圆型偏微分方程，其边界为$\partial D$，

$$\nabla \cdot (-\exp(x(s))\nabla u(s)) = f(s), \quad s \in D,$$

其中$s \in D$表示空间坐标，$f$是已知源项。该方程描述了地下水情境下多孔介质中的达西流。左右边界采用狄利克雷边界条件，上下边界采用齐次诺伊曼边界条件，即

$$\exp(x(s))\nabla u(s) \cdot \vec{n} = 0, \quad s \in \{(0, 2) \times \{0\}\} \cup \{(0, 2) \times \{1\}\},$$
$$u(s) = 0, \quad s \in \{\{0\} \times (0, 1) \cup \{1\} \times (0, 1)\},$$

其中$\vec{n}$表示诺伊曼边界的单位法向量。源项设为

$$f(s) = 3 \exp\left(-\|s^{sr} - s\|_2^2\right),$$

其中$s^{sr} = [s_1^{sr}, s_2^{sr}]^T$表示污染物中心，其位置设为$s^{sr} = [1, 0.5]^T$。采用 FEniCS [36] 实现的双线性有限元法，在 $129 \times 65$ 网格上对问题进行离散（空间自由度数为 8385）。测量噪声设为均值为零的独立同分布高斯分布，标准差设为观测均值的 1%。

设$D_1 = (0, 1.1875) \times (0, 1)$、$D_2 = (0.8125, 2) \times (0, 1)$，界面为$\partial_2 D_1 = \{1.1875\} \times (0, 1)$、$\partial_1 D_2 = \{0.8125\} \times (0, 1)$，如图6所示。本文所有结果均在配备 2.20 GHz Intel(R) Xeon(R) E5-2630 图形处理器的工作站上使用 Python 获得。对于每个局部子域，采用均匀的 $77 \times 65$ 网格，通过双线性有限元法对局部问题 (16) 进行离散（空间自由度数为 5005）。

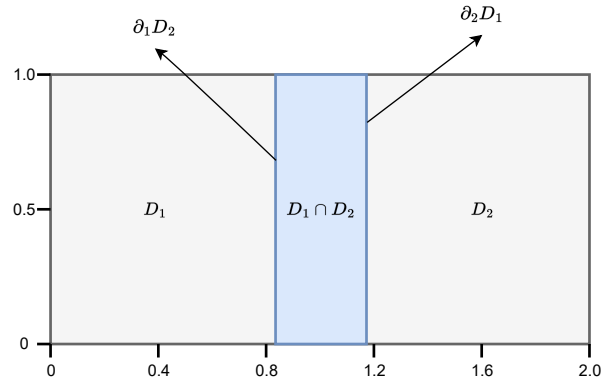



图6：包含$M = 2$个子域的重叠区域分解。

### *4.2.* 变分自编码器的生成特性

令$x(s)$表示高斯随机场，即$x(s) \sim \mathcal{GP}(m(s), \mathcal{K}(s, s'))$，其中$m(\cdot)$和$\mathcal{K}(\cdot, \cdot)$分别表示均值函数和核函数。我们通过KL展开 [25]对$x(s)$进行参数化。协方差函数

$C(s,s'): D \times D \to \mathbb{R}$ 定义为

$$C(s,s') = \mathbb{E}[(x(s,\omega) - m(s))(x(s',\omega) - m(s'))] .$$

本文将协方差函数设为指数核，即，

$$C(s,s') = \sigma_f^2 \exp\left(-\frac{\|s-s'\|_2}{\tau}\right), \tag{22}$$

其中， $\tau$ 是协方差函数的相关长度，$\sigma_f$ 是随机场的标准差。假设 $\tau$ 固定，令 $\{\lambda_r, \psi_r(s)\}_{r=1}^{\infty}$ 为协方差函数（22）的特征值及其对应的正交归一特征函数，则高斯随机场$x(s)$可通过截断的KL展开表示为，

$$x(s) \approx m(s) + \sum_{r=1}^{N_{\mathrm{KL}}} \sqrt{\lambda_r}\psi_r(s)\kappa_r , \tag{23}$$

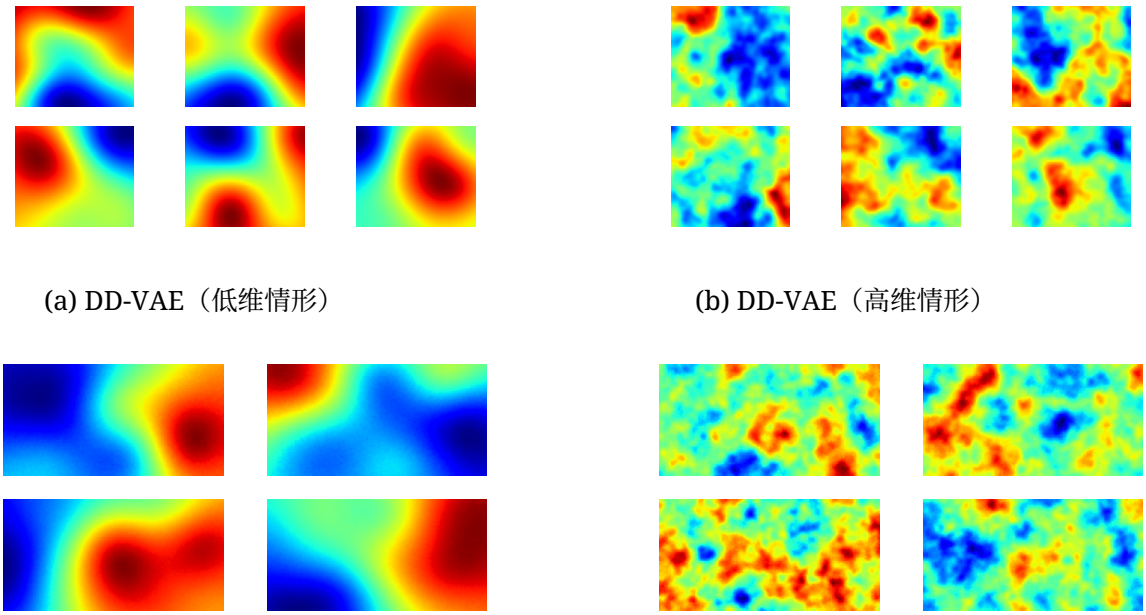
其中， $\kappa_r \sim \mathcal{N}(0,1)$、$r = 1$、……、$N_{\mathrm{KL}}$为不相关的标准正态随机变量。我们的训练数据由相关长度不确定的KL展开生成。令 $\sigma_f^2 = 0.5$且$m(s) = 1$，本文考虑两种情形：低维情形中，式（23）中的相关长度 $\tau$ 取自1到2；高维情形中， $\tau$ 取自0.2到0.65。为捕获95%的总方差，低维情形保留的KL项数$N_{\mathrm{KL}}$约为 12~ 21，而高维情形中$N_{\mathrm{KL}}$ 取自166到1462。然后，我们采用标准有限元方法在 $129 \times 65$ 网格上对$x(s)$进行离散化。

两种情形的训练数据集$\boldsymbol{X}$ 数量均设为$K = 1 \times 10^4$，大小为 $129 \times 65$。随后，增广训练数据集$\boldsymbol{Y}$（见第3.2节）包含 $2\times10^4$ 个大小为 $77\times65$的图像。对于不同的区域尺度和不同的训练数据集，采用不同的网络结构是很自然的。在本文中，编码器和解码器模型均采用全连接神经网络（见第3.2节），因为其结构简单。具体设置见附录A。对于低维情形和高维情形，G-VAE在图12中采用B 型网络结构，而DD-VAE采用A 型网络结构。数值结果表明，在考虑较小区域时，神经网络通常可以设计得更浅。其他超参数设置见附录A。G-VAE的潜变量维度设为512，而DD-VAE的潜变量维度设为256。对于低维情形和高维情形，G-VAE和DD-VAE的训练时间分别约为56分钟和12分钟。

训练过程完成后，可以先从潜在分布中采样，再将样本输入确定性生成模型，从而从先验分布生成样本。从标准高斯分布中随机采样 $\beta$，图7(a) 和图7(b) 展示了使用 D-VAE 在低维情形和高维情形下生成的图像。相比之下，图7(c) 和图7(d) 展示了使用 G-VAE 生成的图像。可以看出，通常即使计算成本更高，G-VAE 生成的图像仍比 DD-VAE 生成的图像更加模糊。

我们引入 Fr´echet Inception Distance（FID）[37]，以进一步量化生成属性。FID 根据原始图像的计算机视觉特征，衡量两组图像的相似程度。分数越低，表示两组图像越相似，或其统计特征越相似（理想分数为 0.0，这表示

两组图像完全相同）。表1展示了不同情况下以及不同 VAE模型设置下的 FID分数。值得注意的是，由于全局域和局部域采用了不同的区域，用于计算 FID 的训练数据并不一致。可以看出，在计算资源相同的情况下，DD-VAE 的生成属性显著优于 G-VAE。



(a) DD-VAE（低维情形）　(b) DD-VAE（高维情形）

(c) G-VAE（低维情形）　(d) G-VAE（高维情形）

图7：使用 DD-VAE 和 G-VAE 随机生成的图像。

表1：G-VAE 和 DD-VAE 的比较。

| 字段类型 | 模型 | 维度<br>输入变量 | 维度<br>潜变量 | FID | 大小<br>训练数据集 | 训练<br>time |
|---|---|---|---|---|---|---|
| 低维 | G-VAE | $129 \times 65$ | 512 | 188.006 | 10000 | 56 分钟 |
| 高斯分布 | DD-VAE | $77 \times 65$ | 256 | 3.107 | 20000 | 12 分钟 |
| 高维 | G-VAE | $129 \times 65$ | 512 | 393.560 | 10000 | 56 分钟 |
| 高斯分布 | DD-VAE | $77 \times 65$ | 256 | 89.351 | 20000 | 12 分钟 |

### *4.3.* 界面处理的结果

对于$(i,j) \in \mathbb{N}$，为界面函数$h_{ij}(s)$在界面 $\partial_j D_i$上构建高斯过程模型时，测试集 $\Delta_{ij}$设为界面上的网格点，每个界面有65个网格点，最大后验方差的阈值 $\delta_{\text{tol}}$ 设为 $10^{-7}$。我们使用 Python 软件包 GPy [38]实现高斯过程回归。所用核为径向基函数核，即 $\mathcal{K}(s,s') = \exp(-\|s-s'\|_2^2/(2\sigma_{\text{GP}}^2))$，其中 $\sigma_{\text{GP}}$ 为高斯过程模型的超参数，由训练数据集自动优化。

所需训练数据点的最大数量及相应的最大后验方差见表2。由于目标界面函数通常较为平滑，表2显示，最多经过五次迭代即可达到阈值。为评估界面处理的准确性，我们计算了与真实渗透率场对应的 GP界面模型和精确界面函数之间的差异。对于每个界面 $\partial_j D_i$，若（$i, j$） $\in$ N，则相对界面误差通过以下方式计算

$$\epsilon_{ij}^{(\mathrm{int})} := \|h_{ij}(s,x) - \widehat{h}_{ij}(s)\|_2 / \|h_{ij}(s,x)\|_2 \tag{24}$$

其中，$h_{ij}(s, x)$ 是定义为 $h_{ij}(s, x) := u(s, x)|_{\partial_j D_i}$ 的精确界面函数（参数 $x$ 与 $\widehat{\ }$ 真实值场相关），而 $h^{ij}(s)$ 是训练后的GP界面模型。此外，对于每个局部子域 $D_i$，$(i = 1, 2)$，还评估了使用GP界面模型获得的局部解的相对状态误差，其计算公式为

$$\epsilon_i^{(\mathrm{state})} := \|u_i^{\mathrm{GP}}(s, x_i) - u_i(s, x_i)\|_2 / \|u_i(s, x_i)\|_2 \,,$$

其中，$u_i^{\mathrm{GP}}(s,x_i)$ 是公式（20）中定义的局部解，$u_i(s, x_i)$ 是公式（16）中定义的精确局部解。表3展示了相对界面误差和状态误差，可以看出这些误差都很小。

表 2：两个测试场景的最大训练数据点数量及其对应的最大后验方差 s.

| 场景 | 高斯场<br>低维情形 | 高斯场<br>高维情形 |
|---|---|---|
| $\lvert\Lambda_{12}\rvert$ | 2 | 3 |
| $\lvert\Lambda_{21}\rvert$ | 5 | 3 |
| $\sigma^{\max}_{\Delta_{12}}$ | $9.0460 \times 10^{-8}$ | $3.4867 \times 10^{-10}$ |
| $\sigma^{\max}_{\Delta_{21}}$ | $2.1025 \times 10^{-10}$ | $1.856 \times 10^{-10}$ |

表3：不同界面的相对误差。

| | 场景 | 高斯场<br>低维情形 | 高斯场<br>高维情形 |
|---|---|---|---|
| $\epsilon_{ij}^{(\text{整数})}$ | $\partial_2 D_1$ | $1.6721 \times 10^{-2}$ | $5.2275 \times 10^{-2}$ |
| | $\partial_1 D_2$ | $3.3415 \times 10^{-2}$ | $1.1325 \times 10^{-2}$ |
| $\epsilon_i^{(\text{状态})}$ | $D_1$ | $1.8193 \times 10^{-2}$ | $2.8392 \times 10^{-2}$ |
| | $D_2$ | $8.3981 \times 10^{-3}$ | $7.1597 \times 10^{-3}$ |

### *4.4.* 反演性能：相关长度不确定的高斯随机场

低维情形和高维情形的真实场分别如图 8(a) 和图 10(a) 所示。这两者均未包含在训练数据中。相应的压力场如图 8(b) 和图 10(b) 所示，黑点表示传感器的位置。求解低维情形和高维情形中与全局模型相关的线性系统约需 1.3529 秒。此外，

两种情形下的局部模型可在 0.3299 秒内求解。全局模型评估与局部模型评估的耗时比约为 4.1009。因此，若将一次局部正向模型评估的计算成本定义为一个成本单位，则每次全局模型评估的成本约为 4 个成本单位。马尔可夫链蒙特卡罗的接受率定义为接受的样本数与总样本数之比。我们仔细调整了 DD-VAE-MCMC 和 G-VAE-MCMC 的步长 $\gamma$ （见 (7)），使接受率处于适当范围。为确保计算成本大致相等，G-VAE-MCMC 的马尔可夫链长度设为 DD-VAE-MCMC 的四倍。低维和高维情形下的步长、马尔可夫链长度及接受率分别汇总于表4和表5。

泊松图像融合所采用的设置如下：对于两个重叠图像$x_1 \in D_1$ 和$x_2 \in D_2$,，将 $\Omega$ 设为 $\Omega := D_1 \cap D_2$,，我们将已知标量函数定义为

$$I^\star(s) = \begin{cases} x_1(s), & s \in D_1/\Omega, \\ x_2(s), & s \in D_2/\Omega. \end{cases}$$

源图像$B$设为

$$B(s) = \frac{1}{2}(x_1(s) + x_2(s)), \quad s \in \Omega,$$

所得融合图像$I$ 记为$x_{\text{ble}}$。作为比较，图像$x_1$ 和$x_2$ 的拼接场 $x_{\text{sti}}$ 为定义如下

$$x_{\text{sti}} = \sum_{i=1}^{M} x_i \cdot \mathbb{I}_{D_i}(s), \tag{25}$$

其中，示性函数 $\mathbb{I}_{D_i}(s)$ 定义为

$$\mathbb{I}_{D_i}(s) := \begin{cases} 0, & \text{if } s \notin D_i \ , \\ 1, & \text{if } s \in D_i/\left(\cup_{j\in\mathfrak{N}_i}(D_i \cup D_j)\right) \ , \\ \frac{1}{|\mathfrak{N}_i|}, & \text{if } s \in \cup_{j\in\mathfrak{N}_i}(D_i \cup D_j) \ . \end{cases}$$

在本研究中，拼接场为

$$x_{\text{sti}} = \begin{cases} x_1(s), & s \in D_1/\Omega, \\ \frac{1}{2}\left(x_1(s) + x_2(s)\right), & s \in \Omega, \\ x_2(s), & s \in D_2/\Omega. \end{cases}$$

为量化生成的后验样本的不确定性，我们按如下方式计算渗透率场的均值和方差估计。给定定义在全局域$D$中的后验样本 $\{x^{(k)}\}_{k=1}^K$，后验均值和方差估计可通过以下公式计算：

$$\widehat{\mathbb{E}}(x) := \frac{1}{K}\sum_{k=1}^{K} x^{(k)}, \tag{26}$$

$$\widehat{\mathbb{V}}(x) := \frac{1}{K}\sum_{k=1}^{K}\left[x^{(k)} - \widehat{\mathbb{E}}(x)\right]^2 . \tag{27}$$

我们将 G-VAE-MCMC 生成的样本记为 $\{x_g^{(k)}\}_{k=1}^{N_C}$。将样本 $\{x_g^{(k)}\}_{k=1}^{N_C}$ 代入式（26）和式（27），即可计算 G-VAE-MCMC 的后验均值 $\widehat{\mathbb{E}}(x_g)$和方差 $\widehat{\mathbb{V}}(x_g)$。将算法3生成的样本 $\{x_{ble}^{(k)}\}_{k=1}^{N_C}$ 代入式（26）和式（27），即可计算 DD-VAE-MCMC 的后验均值 $\widehat{\mathbb{E}}(x_{ble})$和方差 $\widehat{\mathbb{V}}(x_{ble})$。将局部后验样本 $\{x_i^{(k)}\}_{k=1}^{N_C}$ $(i = 1, 2)$代入式 (25)，即可计算后验拼接样本 $\{x_{sti}^{(k)}\}_{k=1}^{N_C}$。随后，将样本$\{x_{sti}^{(k)}\}_{k=1}^{N_C}$ 代入式（26）和式（27），即可计算拼接场的后验均值 $\mathbb{E}(x_{sti})$和方差 $\mathbb{V}(x_{sti})$。

为了评估估计后验均值场的准确性，定义三种情形的相对误差如下:

$$\epsilon_{sti} := \|\widehat{\mathbb{E}}(x_{sti}) - x_{truth}\|_2 / \|x_{truth}\|_2 , \tag{28}$$

$$\epsilon_{ble} := \|\widehat{\mathbb{E}}(x_{ble}) - x_{truth}\|_2 / \|x_{truth}\|_2 , \tag{29}$$

$$\epsilon_{g} := \|\widehat{\mathbb{E}}(x_{g}) - x_{truth}\|_2 / \|x_{truth}\|_2 , \tag{30}$$

其中$x_{truth}$ 为真实值。表6展示了两个测试问题中均值估计的相对误差。我们的 DD-VAE-MCMC 通常能给出较小的误差。拼接场的相对误差较大，且对应图像中可见明显的接缝。

图9和图 11展示了低维情形和高维情形下拼接场、DD-VAE-MCMC 方法以及 G-VAE-MCMC 方法的反演结果对比。图9(a)、图9(c)和图9(e)对比了低维情形下的估计均值场，图 11(a)、图 11(c)和图 11(e)对比了高维情形下的估计均值场。可以看出，拼接场在界面 $\partial_1 D_2$ 和 $\partial_2 D_1$, 处存在明显接缝，而融合图像融合了两侧的信息，能够得到明显完整的后验均值场。此外，G-VAE-MCMC 在两种情形下都无法给出合理的反演结果。同时，从图9(b)、图9(d)和图9(f)，以及图 11(b)、图 11(d)和图 11(f)可以看出，我们的 DD-VAE-MCMC 通常给出的结果方差较小。直观而言，泊松图像融合技术通过考虑梯度信息，隐式地将两个子域的信息纳入考量，从而在全局域上获得更好的表现。

表4：马尔可夫链蒙特卡罗设置及接受率（低维）。

| 方法 | 区域 | 步长 | 链长 | 接受率 |
|---|---|---|---|---|
| DD-VAE-MCMC | $D_1$ | 0.04 | 4000 | 15.00% |
| | $D_2$ | 0.03 | 4000 | 15.90% |
| G-VAE-MCMC | $D$ | 0.03 | 1000 | 18.00% |

## 5. 结论

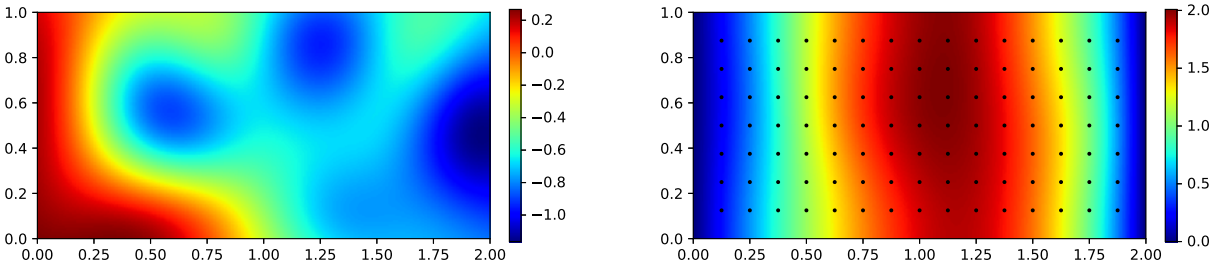
分而治之是利用涉及偏微分方程的正向模型求解高维贝叶斯问题时最基本的理念之一。贝叶斯反问题的主要难点包括维数灾难、隐式先验建模以及高昂的计算成本。特别关注的情况是

表5：MCMC设置和接受率（高维）。

| 方法 | 区域 | 步长 | 链长 | 接受率 |
|---|---|---|---|---|
| DD-VAE-MCMC | $D_1$ | 0.02 | 4000 | 24.47% |
| | $D_2$ | 0.02 | 4000 | 23.38% |
| G-VAE-MCMC | $D$ | 0.03 | 1000 | 21.90% |

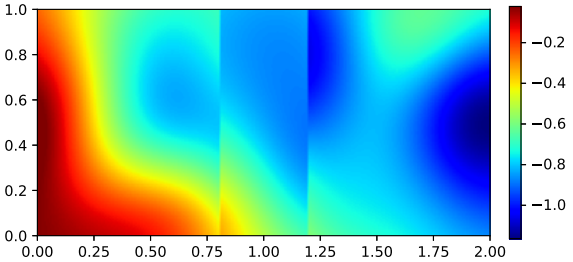
表6：两种情形下均值估计的误差。

| 方法 | 低维情形 | 高维情形 |
|---|---|---|
| DD-VAE-MCMC | $1.3818 \times 10^{-1}$ | $6.2113 \times 10^{-1}$ |
| 拼接场 | $1.4518 \times 10^{-1}$ | $6.6281 \times 10^{-1}$ |
| G-VAE-MCMC | $3.4009 \times 10^{-1}$ | 1.2002 |

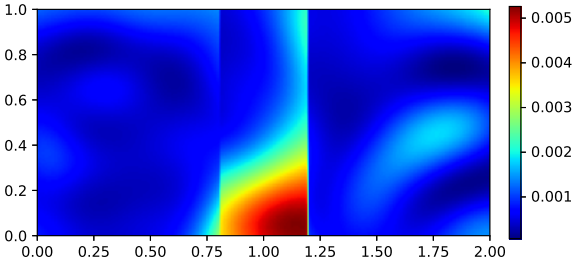



(a) 真实对数渗透率场。　　(b) 真实压力场和传感器。

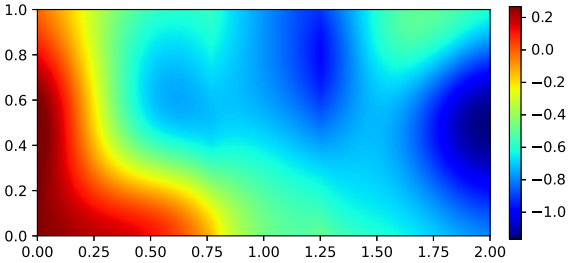
图 8：测试问题设置（低维情形）。



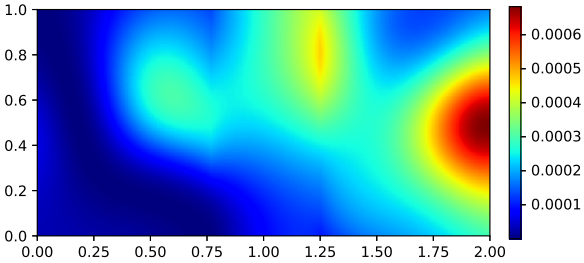

(a) 均值 $\widehat{\mathbb{E}}(x_{\text{sti}})$，拼接后的场。



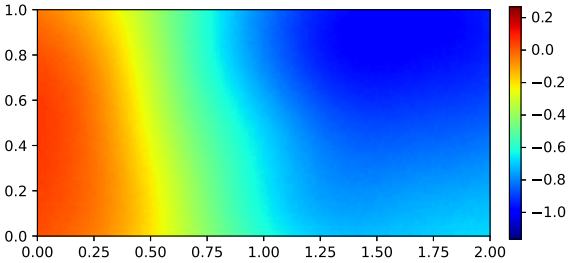

(b) 方差 $\widehat{\mathbb{V}}(x_{\text{sti}})$，拼接后的场。



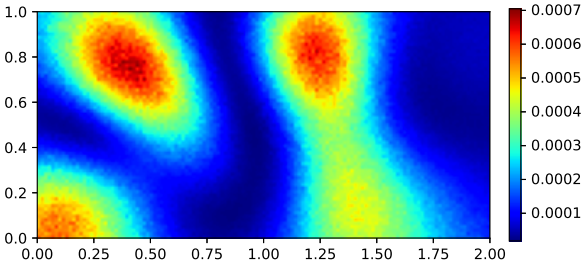

(c) 均值 $\widehat{\mathbb{E}}(x_{\text{ble}})$区域分解变分自编码器马尔可夫链蒙特卡罗。




(d) 方差 $\widehat{\mathbb{V}}(x_{\text{ble}})$，DD-VAE-MCMC。

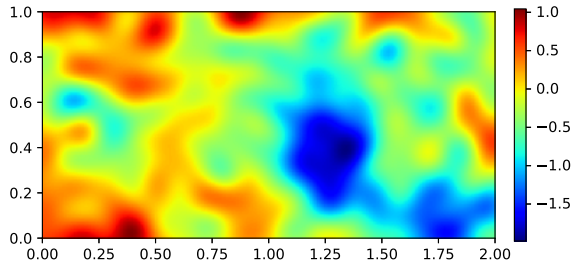



(e) 均值 $\widehat{\mathbb{E}}(x_{\text{g}})$，G-VAE-MCMC。

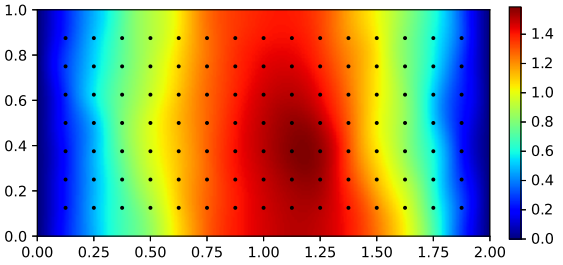



(f) 方差 $\widehat{\mathbb{V}}(x_{\text{g}})$G-VAE-MCMC。

图9：估计均值场和方差场（低维情形）。




(a) 真实对数渗透率场。




(b) 真实压力场和传感器。

图 10：测试问题设置（高维情形）。

针对先验信息无法以闭式形式获得、只能通过历史数据获取的情况，本文提出了一种域分解变分自编码器马尔可夫链蒙特卡罗（DD-VAE-MCMC）方法。变分自编码器可利用历史数据学习未知的先验分布。通过将全局域划分为若干子域，我们提出了 DD-VAE框架，以在低维潜在空间中表示局部先验分布。采用结合主动学习的高斯过程模型处理界面条件。随后，原始全局问题被分解为一系列局部问题，从而能够在维度更低的参数空间和规模更小的物理子域中进行反演。在收集局部后验样本后，利用泊松图像融合技术重建后验全局样本。数值结果表明，基于分解子域的生成属性得到显著改善，并验证了所提方法的效率。

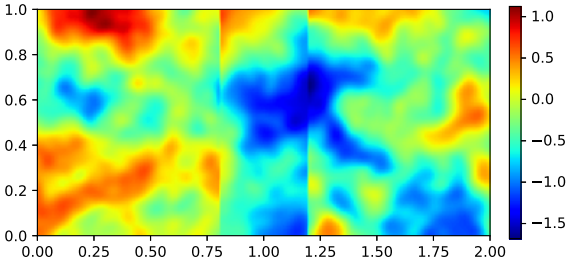
**致谢：**本研究得到国家自然科学基金委员会（编号：12071291）、上海市科学技术委员会（编号：20JC1414300）和上海市自然科学基金（编号：20ZR1436200）的资助。

**参考文献**

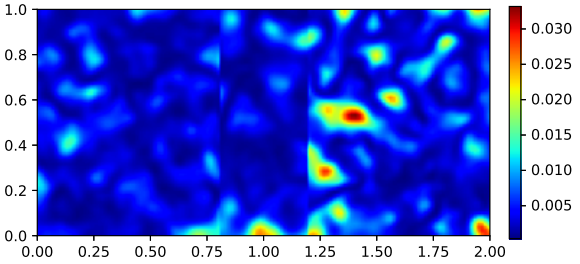
[1] J. Kaipio, E. Somersalo，《统计与计算反问题》，第160卷，Springer Science & Business Media，2006年。

[2] J. Martin, L.C. Wilcox, C. Burstedde, O. Ghattas，适用于大规模统计反问题并应用于地震反演的随机牛顿马尔可夫链蒙特卡罗方法，SIAM科学计算杂志 34 (3) (2012) A1460–A1487。

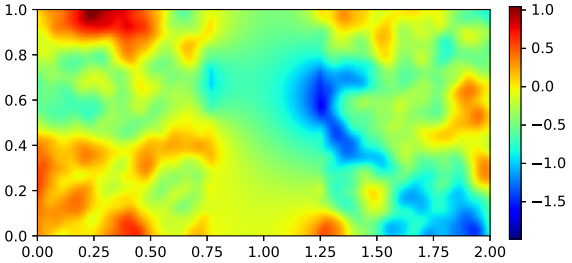
[3] J. Wang, N. Zabaras，热传导反问题的层次贝叶斯模型，反问题 21 (1) (2004) 183。



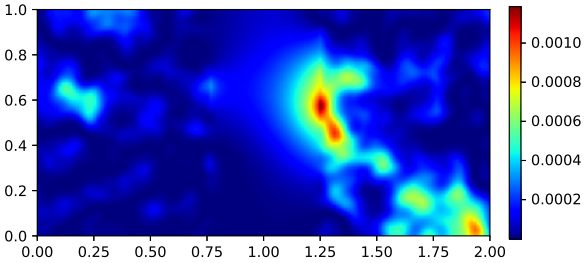

(a) 均值 $\widehat{\mathbb{E}}(x_{\text{sti}})$，拼接后的文件。



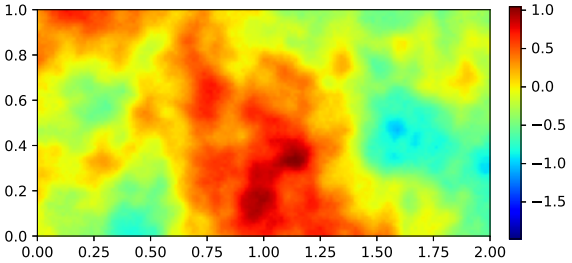

(b) 方差 $\widehat{\mathbb{V}}(x_{\text{sti}})$，拼接后的文件。



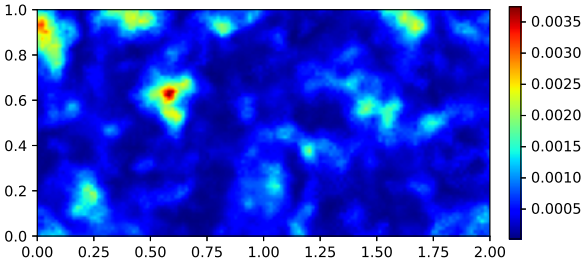

(c) 均值 $\widehat{\mathbb{E}}(x_{\text{ble}})$区域分解-变分自编码器-马尔可夫链蒙特卡罗。




(d) 方差 $\widehat{\mathbb{V}}(x_{\text{ble}})$，DD-VAE-MCMC。




(e) 均值 $\widehat{\mathbb{E}}(x_{\text{g}})$，G-VAE-MCMC。




(f) 方差 $\widehat{\mathbb{V}}(x_{\text{g}})$——G-VAE-MCMC。

图 11：估计均值场和方差场（高维情形）。

[4] W. W.-G. Yeh, 《地下水水文学中参数识别程序综述：逆问题》, 《水资源研究》22 (2) (1986) 95–108。[5] A. Tarantola, 《逆问题理论与模型参数估计方法》, SIAM, 2005。[6] A. M. Stuart, 《反问题：贝叶斯视角》, 《数值学报》19 (2010) 451–559。[7] N. Metropolis、A. W. Rosenbluth、M. N. Rosenbluth、A. H. Teller、E. Teller, 《利用快速计算机进行状态方程计算》, 《化学物理学杂志》21 (6) (1953) 1087–1092。[8] W. K. Hastings, 《使用马尔可夫链的蒙特卡罗采样方法及其应用》, 《生物统计学》57 (1) (1970) 97–109。[9] C. Robert、G. Casella, 《蒙特卡罗统计方法》, Springer Science & Business Media, 2013。[10] R. Salakhutdinov, 《学习深度生成模型》, 《统计学及其应用年度评论》2 (2015) 361–385。[11] E. Laloy、R. H´erault、J. Lee、D. Jacques、N. Linde, 《基于深度神经网络的复杂二元地质介质低维新表示及其反演》, 《水资源进展》110 (2017) 387–405。[12] D. P. Kingma、M. Welling, 《自编码变分贝叶斯》, arXiv 预印本 arXiv:1312.6114。[13] Y. Xia、N. Zabaras, 《用于求解高维逆问题的贝叶斯多尺度深度生成模型》, 《计算物理学杂志》455 (2022) 111008。[14] A. Tewari、B. Wheelock、J. Clark、D. Foster、M. Li、Y. Marzouk, 《利用深度地质先验进行地下不确定性量化：变分贝叶斯框架》, 载于：第二届应用地球科学与能源国际会议, 勘探地球物理学家协会和美国石油地质学家协会, 2022年, 第1745–1749页。[15] J. Lopez-Alvis、E. Laloy、F. Nguyen、T. Hermans, 《反演中的深度生成模型：生成器非线性的影响及基于变分自编码器的新方法》, 《计算机与地球科学》152 (2021) 104762。[16] I. Goodfellow、J. Pouget-Abadie、M. Mirza、B. Xu、D. Warde-Farley、S. Ozair、A. Courville、Y. Bengio, 《生成对抗网络》, 《美国计算机协会通讯》63 (11) (2020) 139–144。[17] H. Elman、D. Silvester、A. Wathen, 《有限元与快速迭代求解器：及其在不可压缩流体动力学中的应用》, 牛津大学出版社（英国）, 2014。[18] A. Beskos、G. Roberts、A. Stuart、J. Voss, 《扩散桥的MCMC方法》, 《随机过程与动力学》8 (03) (2008) 319–350。

[19] S. L. Cotter、G. O. Roberts、A. M. Stuart、D. White，《函数的MCMC方法：改进旧算法以提高速度》，Statistical Science 28 (3) (2013) 424–446。[20] L. Dinh、J. Sohl-Dickstein、S. Bengio，《使用Real NVP进行密度估计》，arXiv 预印本 arXiv:1605.08803。[21] J. Ho、A. Jain、P. Abbeel，《去噪扩散概率模型》，《神经信息处理系统进展》33 (2020) 6840–6851。[22] Y. M. Marzouk、H. N. Najm、L. A. Rahn，《用于高效贝叶斯求解反问题的随机谱方法》，《计算物理学杂志》224 (2) (2007) 560–586。[23] J. Li、Y. M. Marzouk，《用于贝叶斯求解反问题的自适应代理模型构建》，SIAM科学计算杂志36 (3) (2014) A1163–A1186。[24] Q. Liao、J. Li，《一种用于高维贝叶斯逆问题的自适应降阶基ANOVA方法》，《计算物理学杂志》396 (2019) 364–380。[25] P. D. Spanos、R. Ghanem，《随机介质的随机有限元展开》，《工程力学杂志》115 (5) (1989) 1035–1053。[26] Y. Chen、J. Jakeman、C. Gittelson、D. Xiu，《具有高维随机输入的线性微分方程的局部多项式混沌展开》，SIAM科学计算杂志37 (1) (2015) A79–A102。[27] Z. Xu、Q. Liao、J. Li，《基于局部Karhunen-Lo`eve展开的域分解贝叶斯反演》，arXiv 预印本 arXiv: 2211.04026。[28] Q. Liao、K. Willcox，《一种用于不确定性分析的域分解方法》，SIAM科学计算杂志37 (1) (2015) A103–A133。[29] A. M. Quarteroni、A. Valli，《偏微分方程的域分解方法》，牛津大学出版社，1999。[30] A. D. Jagtap、G. E. Karniadakis，《扩展物理信息神经网络（XPINNs）：一种用于非线性偏微分方程的通用时空域分解深度学习框架》，《计算物理学通讯》28 (5) (2020) 2002–2041。[31] A. D. Jagtap、E. Kharazmi、G. E. Karniadakis，《离散区域上的守恒型物理信息神经网络：守恒律在正问题与逆问题中的应用》，计算机方法在应用力学与工程中的应用365 (2020) 113028。[32] K. Li、K. Tang、T. Wu、Q. Liao，《D3M：一种用于偏微分方程的深度域分解方法》，IEEE Access 8 (2019) 5283–5294。[33] S. Li、Y. Xia、Y. Liu、Q. Liao，《一种基于傅里叶特征的深度域分解方法》，计算与应用数学杂志423 (2023) 114963。

[34] C. E. Rasmussen，《机器学习中的高斯过程》，载于：机器学习暑期学校，Springer，2003年，第63–71页。

[35] P. P´erez，M. Gangnet，A. Blake，《泊松图像编辑》，载于：ACM SIGGRAPH 2003论文集，2003年，第313–318页。

[36] M. Alnæs、J. Blechta、J. Hake、A. Johansson、B. Kehlet、A. Logg、C. Richardson、J. Ring、M. E. Rognes、G. N. Wells，《FEniCS项目 1.5 版》，《数值软件档案》3 (100)。

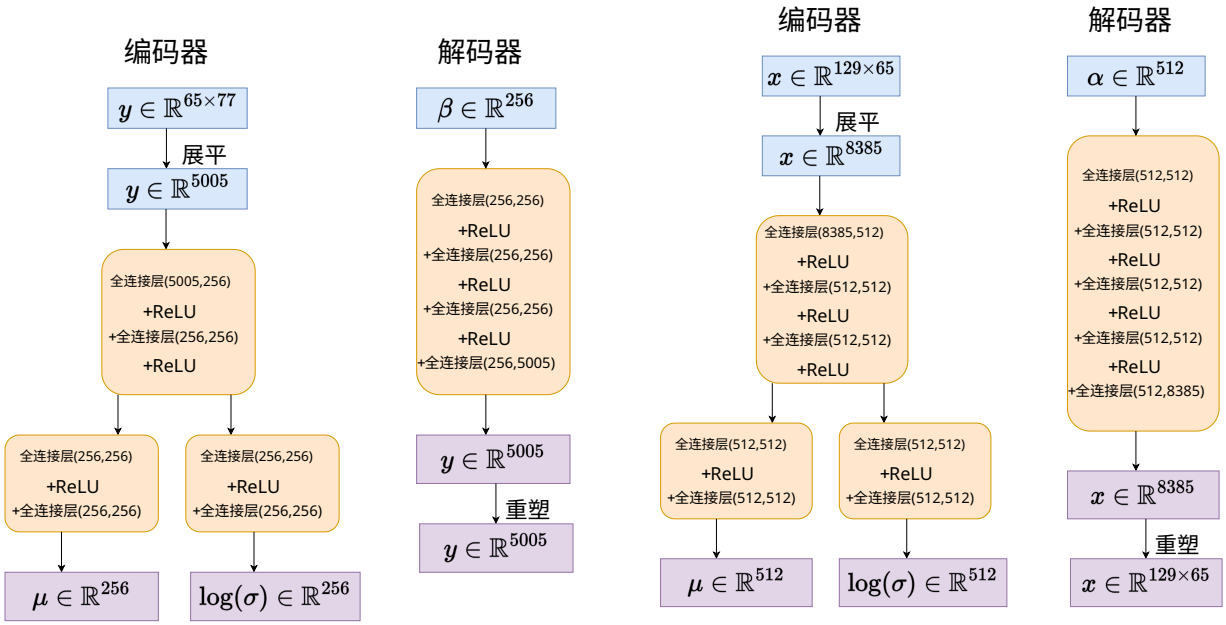
[37] M. Heusel、H. Ramsauer、T. Unterthiner、B. Nessler、S. Hochreiter，《采用双时间尺度更新规则训练的生成对抗网络收敛至局部纳什均衡》，《神经信息处理系统进展》第 30 卷。

[38] GPy，《GPy：Python 中的高斯过程框架》，http://github.com/SheffieldML/GPy（自 2012 年起）。

# 附录

**A. 变分自编码器的编码器和解码器网络的神经网络架构。**

本节将详细介绍我们采用的网络设置。在这项工作中，我们使用全连接神经网络（FCNN）作为变分自编码器中的编码器和解码器模型。我们采用的网络结构如图12所示，训练过程中使用的相关超参数列于表7。




(a) 网络结构 A　　(b) 网络结构 B

图12：用于先验学习的变分自编码器结构（A 型和 B 型）。

表 7：变分自编码器的神经网络设置

| | G-VAE | DD-VAE |
|---|---|---|
| 架构 | B 型 | A 型 |
| 训练数据集大小 | 10000 | 20000 |
| 训练轮数 | 100 | 100 |
| 学习率 | 0.001 | 0.0001 |
| 批量大小 | 32 | 64 |
| 优化器 | Adam | Adam |
| 优化器参数 $(\beta_1, \beta_2)$ | 0.5, 0.999 | 0.5, 0.999 |