收稿：由生产环节添加 修订：由生产环节添加 接收：由生产环节添加DOI：xxx/xxxx

arXiv:2302.10917v1 [cs.CE] 18 Feb 2023

研究论文

混合间断伽辽金方法的无矩阵宏单元变体

Vahid Badrkhani* | René R. Hiemstra | Michał Mika | Dominik Schillinger

[1]德国达姆施塔特工业大学力学研究所计算力学组

通信作者
*Vahid Badrkhani，力学研究所，计算力学组，达姆施塔特工业大学，德国。电子邮箱：vahid.badrkhani@tu-darmstadt.de

资助信息 本研究得到德国研究联合会（Deutsche Forschungsgemeinschaft）资助，资助编号：1249/2-1

摘要

我们研究了混合间断 Galerkin（HDG）方法的一种宏单元变体，该方法采用由标准单纯形单元组成的片区，且这些片区的接口可以不匹配。通过HDG技术进行耦合，我们的方法能够通过对每个宏单元进行一致单纯形细分来实现局部细化。通过对所有宏单元采用相同的空间离散，我们得到每个宏单元对应的局部问题；这些问题不仅易于并行处理，而且负载均衡良好。因此，我们的宏单元变体能够高效扩展至 n节点集群，并可通过调整局部问题规模以适应单个节点的处理能力，从而匹配可用硬件，同时仍采用二次多项式或三次多项式等适中的多项式阶数。增大局部问题规模，意味着相对而言同时缩小全局问题规模，从而有效控制自由度数量的增长。全局问题通过一种无矩阵迭代技术求解，该技术也大量依赖宏单元局部运算。我们通过一个对流-扩散模型问题，探讨并讨论了宏单元混合间断伽辽金方法的优势与局限性。

关键词：
混合间断伽辽金方法、宏单元、无矩阵、局部自适应细化、区域分解、负载均衡、可扩展性

1 引言

间断伽辽金（DG）公式[1] 在求解守恒问题时具有诸多理想特性，例如严谨的数学基础、能够在一般非结构网格上使用任意阶基函数，以及对流算子的自然稳定性[2,3,4,5]。大约十年前，混合间断 Galerkin（HDG）方法在[6], 中被提出；与其他 DG方法相比，该方法具有若干独特优势。首先，对于光滑且扩散占优的问题，该方法可提供所有变量的近似解，且在L^2 范数[7]下以最优阶 $p+1$ 收敛。此外，通过一种计算成本低廉、逐单元进行的后处理过程[8,9]，该方法还能计算新的近似速度场，其对 $p \geq 1$的收敛阶为 $p+2$ 。其次，该方法提供了一种新颖且系统化的方式来施加不同类型的边界条件[10,11]。第三，由混合间断伽辽金方法产生的线性系统等价于两个不同的线性系统：第一个系统在全局范围内耦合单元边界上的解的数值迹，从而显著减少自由度；第二个系统则在单元层面（局部）耦合守恒量及其梯度，因此可以逐单元求解[12,13]。鉴于这些

优势，许多研究工作已将混合间断伽辽金方法扩展到各种初边值问题和偏微分方程组中，参见例如[6,7,8,9,11,13,14,15,16,17,18,19]。

DG方法的一个普遍缺点是，与标准连续有限元方法相比，其自由度数量更多，且全局耦合更强。混合间断伽辽金方法的目标之一，是提高DG格式及其在间断有限元空间中所得解的计算效率。具体而言，基于单元界面上的附加未知量进行混合化，可减少全局耦合，并便于采用静态凝聚策略。然而，对于大规模计算[20,21],，仅靠混合化往往不足以克服内存和求解时间方面的限制，因此围绕算法的研究仍在持续[22,23,24,25]。

与其间断形式相比，连续Galerkin（CG）方法，即经典有限元公式[26,27],，在相同网格[28]上所需的未知量更少。CG解也可以通过静态凝聚加速，该方法生成的全局耦合系统仅包含网格骨架上的自由度。非结构网格生成器通常会生成顶点价数较高（即与给定顶点相邻的单元数量较多）的网格。因此，在分布式内存系统上并行运行时，连续伽辽金方法涉及相当复杂的通信模式，这会对可扩展性产生不利影响[29,30]。混合间断伽辽金方法为单元边界上的近似生成一个全局系统；尽管其阶数高于CG中传统静态凝聚系统，但在多项式次数适中的情况下，其带宽显著更小[13,29]。

本文研究一种结合连续方法与混合间断方法要素的离散化策略。其核心思想不是在各个单元之间应用 HDG 概念，而是在较大的宏单元之间应用该概念，每个宏单元本身可包含数量灵活的有限元。标准 HDG 方法中，全局问题和局部问题的结构与性质由给定网格及各单元上的多项式次数严格决定；而宏单元混合间断伽辽金概念则使我们能够灵活地影响这种结构。凭借这种灵活性，所提出的宏单元变体带来了标准 HDG 方法无法以同样方式提供的额外算法可能性。本文将探讨如何利用这些可能性，在现代异构计算系统上实现实用且有效的自适应加密、区域分解和负载均衡方法。

值得注意的是，宏单元HDG概念与区域分解方法有相似之处[31,32,33,34],，这类方法旨在将给定的有限元离散化[35,36] 划分为子域块。然而，其显著特点在于代数结构：它由一个基于宏单元之间迹变量的全局耦合系统，以及多个彼此解耦的系统构成，每个宏单元对应一个系统；这使 HDG 方法在并行计算方面具备独特优势。

本文结构安排如下：第 2 节简要介绍模型问题，并明确全文后续部分所需的基本概念和符号。第 3 节将宏单元上的混合间断伽辽金变体与标准混合间断伽辽金离散化进行对比，并从精度和计算成本的角度说明其基本性质。第 4 节将讨论并重点介绍该离散化策略在精度、自适应加密、区域分解和负载均衡方面的实际优势。第 5 节和第 6 节将深入研究宏单元HDG变体的计算效率和可扩展性，并与标准混合间断伽辽金方法进行比较：首先考察理论估计，随后通过并行实现进行数值测试。

2 背景与符号

下面我们将简要介绍模型问题。随后，我们将在一般设置下回顾一些相关背景并明确本文采用的符号；这一设置既适用于完全不连续单元上的标准混合间断伽辽金方法，也适用于我们在宏单元上的变体。

2.1 模型问题

本文通过三维空间中的定常线性对流扩散方程，展示这些方法及其数值性质。设 $\Omega \subset \mathbb{R}^d$ 为具有分片光滑边界 $\partial\Omega = \Gamma_D \cup \Gamma_N$ 的开集，$\emptyset = \Gamma_D \cap \Gamma_N$ 为其外法向单位向量。考虑不可压缩流体中某种物质的浓度 u，其模型为

$$\nabla \cdot (\boldsymbol{a}u) - \nabla \cdot (\kappa \nabla u) = f \qquad \text{于 } \Omega, \tag{1a}$$
$$u = g_D \qquad \text{on } \Gamma_D, \tag{1b}$$
$$(\boldsymbol{a}u - \kappa \nabla u) \cdot \boldsymbol{n} = g_N \qquad \text{on } \Gamma_N. \tag{1c}$$

这里， $\boldsymbol{a} \in (L^\infty(\Omega))^d$ 表示无散对流速度场， $\kappa > 0$ 为常数扩散系数， $f \in L^2(\Omega)$为标量源项， g_D 和 g_N 分别为给定的狄利克雷边界数据和诺伊曼边界数据。

考虑辅助变量 $\boldsymbol{q} = -\nabla u$，我们可以将（1）重写为以下一阶方程组

$$\boldsymbol{q} + \nabla u = 0 \qquad \text{于 } \Omega, \tag{2a}$$
$$\nabla \cdot (\boldsymbol{a}u + \kappa \boldsymbol{q}) = f \qquad \text{in } \Omega, \tag{2b}$$
$$u = g_D \qquad \text{on } \Gamma_D, \tag{2c}$$
$$(\boldsymbol{a}u + \kappa \boldsymbol{q}) \cdot \boldsymbol{n} = g_N \qquad \text{on } \Gamma_N. \tag{2d}$$

2.2 网格、网格骨架和迹变量

我们首先简要规定用于描述分片计算域的记号。令 $\mathcal{T}_h$ 表示将区域 Ω 剖分为一组互不相交的子域 $\{ K \ : \ \bigcup K = \Omega,\ K_i \cap K_j = \emptyset \text{，其中 } K_i \neq K_j \}$。子域边界的集合记为 $\partial\mathcal{T}_h = \{ \partial K \ : \ K \in \mathcal{T}_h \}$。相邻子域 K^+ 和 K^- 在$\mathcal{T}_h$ 中共享的公共界面 $F = \partial K^+ \cap \partial K^-$ 称为内部面。所有内部面的集合为集合 $\mathcal{E}_h^\circ$，称为“网格骨架”。类似地，边界面是子域 $K \in \mathcal{T}_h$ 与边界 $\partial\Omega$ 的非空交集 $\partial K \cap \partial\Omega$，其集合记为$\mathcal{E}_h^\partial$。所有面 $\mathcal{E}_h^\circ \cup \mathcal{E}_h^\partial$ 的集合记为 $\mathcal{E}_h$。

我们回顾跳跃算子的定义。令 K^+ 和 K^- 表示相邻的子域，令 $\boldsymbol{n}^+$ 和 $\boldsymbol{n}^-$ 分别表示 ∂K^+ 和 ∂K^- 的外向单位法向量。令 $u^\pm$ 和 $\boldsymbol{q}^\pm$ 表示从 $K^\pm$ 内部在 F 上的 u 和 $\boldsymbol{q}$ 的迹。我们将标量值函数和向量值函数的跳跃 $[\![\cdot]\!]$ 定义为

$$[\![u\boldsymbol{n}]\!] := u^+\boldsymbol{n}^+ + u^-\boldsymbol{n}^- \quad \text{on } F \in \mathcal{E}_h^\circ, \qquad [\![u\boldsymbol{n}]\!] := u\boldsymbol{n} \quad \text{on } F \in \mathcal{E}_h^\partial, \tag{3a}$$
$$[\![\boldsymbol{q}\cdot\boldsymbol{n}]\!] := \boldsymbol{q}^+\cdot\boldsymbol{n}^+ + \boldsymbol{q}^-\cdot\boldsymbol{n}^- \quad \text{on } F \in \mathcal{E}_h^\circ, \qquad [\![\boldsymbol{q}\cdot\boldsymbol{n}]\!] := \boldsymbol{q}\cdot\boldsymbol{n} \quad \text{on } F \in \mathcal{E}_h^\partial. \tag{3b}$$

注意，标量函数的跳跃是向量值的，而向量值函数的跳跃是标量值的。

2.3 函数空间框架与插值空间

我们简要固定相关不连续试验与检验空间的记号。令 $L^2(D)$ 表示一般子域 $D \subset \mathbb{R}^d$ 上平方可积函数构成的空间。对于函数 $\boldsymbol{q}, \boldsymbol{v} \in \left(L^2(D)\right)^d$，我们记 $(\boldsymbol{q}, \boldsymbol{v})_D := \int_D \boldsymbol{q}\cdot\boldsymbol{v}$。对于标量函数，若 D 是一个 d 维子域，则记 $u, w \in L^2(D)$ $(u, w)_D := \int_D uw$；若 D 的维数为 $d-1$，则记 $\langle u, w\rangle_D := \int_D uw$。我们还引入

$$(u,w)_{\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h} (u,w)_K, \qquad \langle \rho,\eta\rangle_{\partial\mathcal{T}_h} = \sum_{K\in\mathcal{T}_h} \langle \rho,\eta\rangle_{\partial K}, \qquad \langle \lambda,\mu\rangle_{\mathcal{E}_h} = \sum_{F\in\mathcal{E}_h} \langle \lambda,\mu\rangle_F, \tag{4}$$

对于定义在 $\mathcal{T}_h$ 上的函数 u， w、 ρ， η（定义在 $\partial\mathcal{T}_h$ 上）以及 λ， μ（定义在 $\mathcal{E}_h$ 上）。

现在介绍定义在分片计算域上的有限维空间。令 $W(K)$、 $\boldsymbol{V}(K)$ 和 $M(F)$ 分别表示定义在子域 $K \in \mathcal{T}_h$ 和 $F \in \mathcal{E}_h$ 上的 C^0 连续分片多项式空间，其阶数为 $p \geq 1$。我们在 $\mathcal{T}_h$ 上定义以下不连续有限元空间

$$W^h := \left\{ w \in L^2(\Omega) \ : \ w|_K \in W(K)\ \forall K \in \mathcal{T}_h \right\}, \tag{5a}$$
$$\boldsymbol{V}^h := \left\{ \boldsymbol{v} \in \left(L^2(\Omega)\right)^d \ : \ \boldsymbol{v}|_K \in \boldsymbol{V}(K)\ \forall K \in \mathcal{T}_h \right\}. \tag{5b}$$

在网格骨架 $\mathcal{E}_h$ 上，我们为迹变量定义一个不连续有限元空间

$$M^h := \left\{ \mu \in L^2(\mathcal{E}_h) \ : \ \mu|_F \in M(F) \ \forall F \in \mathcal{E}_h \right\}. \tag{5c}$$

我们还需要带有预设狄利克雷条件和诺伊曼条件的空间， $M^h(g_\alpha) = \{ \mu \in M^h \ : \ \mu = P(g_\alpha)$ 在 $\Gamma_\alpha \}$、$(\alpha = D, N)$上，这些条件通过 L^2-投影施加： $P \ : \ L^2(\Gamma_\alpha) \mapsto L^2(\Gamma_\alpha)$、 $P \circ P = \mathrm{Id}$。

3 宏单元上的混合间断伽辽金方法

接下来，我们将讨论混合间断伽辽金方法在宏单元上的变体，并着重说明它与采用全间断单元的标准混合间断伽辽金方法之间的差异。为此，我们将先关注理论和公式构造方面，随后详细介绍用于求解全局问题的无矩阵求解方法。

3.1 混合间断伽辽金方法

我们首先回顾混合间断伽辽金公式构造，并讨论其若干数学性质。令 $\hat{u}_h = \hat{v}_h + \hat{g}_h$，其中$\hat{v}_h \in M^h(0)$，且对所有 $F \in \mathcal{E}_h^\circ$， $\hat{g}_h \in M^h(g_D)$ 均为零。弱公式为：求 $(\boldsymbol{q}_h, u_h, \hat{v}_h) \in \boldsymbol{V}^h \times W^h \times M^h(0)$使得

$$\left(\boldsymbol{q}_h, \boldsymbol{v}_h\right)_{\mathcal{T}_h} - \left(u_h, \nabla \cdot \boldsymbol{v}_h\right)_{\mathcal{T}_h} + \langle \hat{v}_h, \boldsymbol{v}_h \cdot \boldsymbol{n} \rangle_{\partial \mathcal{T}_h} = -\langle g_D, \boldsymbol{v}_h \cdot \boldsymbol{n} \rangle_{\Gamma_D} \qquad \forall \boldsymbol{v}_h \in \boldsymbol{V}^h, \tag{6a}$$

$$-\left(\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h, \nabla w_h\right)_{\mathcal{T}_h} + \langle \left(\widehat{\boldsymbol{a}u_h} + \widehat{\kappa \boldsymbol{q}_h}\right) \cdot \boldsymbol{n}, w_h \rangle_{\partial \mathcal{T}_h} = \left(f, w_h\right)_{\mathcal{T}_h} \qquad \forall w_h \in W^h, \tag{6b}$$

$$\langle [\![\left(\widehat{\boldsymbol{a}u_h} + \widehat{\kappa \boldsymbol{q}_h}\right) \cdot \boldsymbol{n}]\!], \hat{\mu}_h \rangle_{\mathcal{E}_h} = \langle g_N, \hat{\mu}_h \rangle_{\Gamma_N} \qquad \forall \hat{\mu}_h \in M^h(0). \tag{6c}$$

这里，“数值通量” $\widehat{\boldsymbol{a}u_h} + \widehat{\kappa \boldsymbol{q}_h}$ 和 ∂K 上的迹 $\hat{u}_h$ 分别是 $\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h$ 和 u_h的合适近似，并满足

$$\langle (\widehat{\boldsymbol{a}u_h} + \widehat{\kappa \boldsymbol{q}_h}) \cdot \boldsymbol{n}, w_h \rangle_{\partial K} = \langle (\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h) \cdot \boldsymbol{n}, w_h \rangle_{\partial K}, \tag{6d}$$

$$\langle \hat{u}_h, \boldsymbol{v}_h \cdot \boldsymbol{n} \rangle_{\partial K} = \langle u_h, \boldsymbol{v}_h \cdot \boldsymbol{n} \rangle_{\partial K}. \tag{6e}$$

根据[7,13], , 我们为数值通量选择如下关系

$$\widehat{\boldsymbol{a}u_h} + \widehat{\kappa \boldsymbol{q}_h} = \boldsymbol{a}\hat{u}_h + \kappa \boldsymbol{q}_h + \tau_K \left(u_h - \hat{u}_h\right) \boldsymbol{n} \qquad \text{on } \partial K. \tag{6f}$$

这里， τ_K 是“局部稳定化参数”，它提供了一种控制该方法精度和稳定性的机制。数值通量的这种选择对应于所谓的局部 DG 方法。文献[7,8] 证明了所述弱形式解的存在性和唯一性，并讨论了稳定化参数的适当取值。

为了理解强制施加约束与弱施加约束的区别，将上述弱陈述转换为残差形式会很有帮助。利用方程 6d 和 6e 以及分部积分，我们可以将方程 (6a)、(6b) 和 (6c) 重述为每个子域 K上的残差形式：

$$\left(\boldsymbol{q}_h + \nabla u_h, \boldsymbol{v}_h\right)_K = \boldsymbol{0} \qquad \forall \boldsymbol{v}_h \in \boldsymbol{V}(K), \ K \in \mathcal{T}_h, \tag{7a}$$

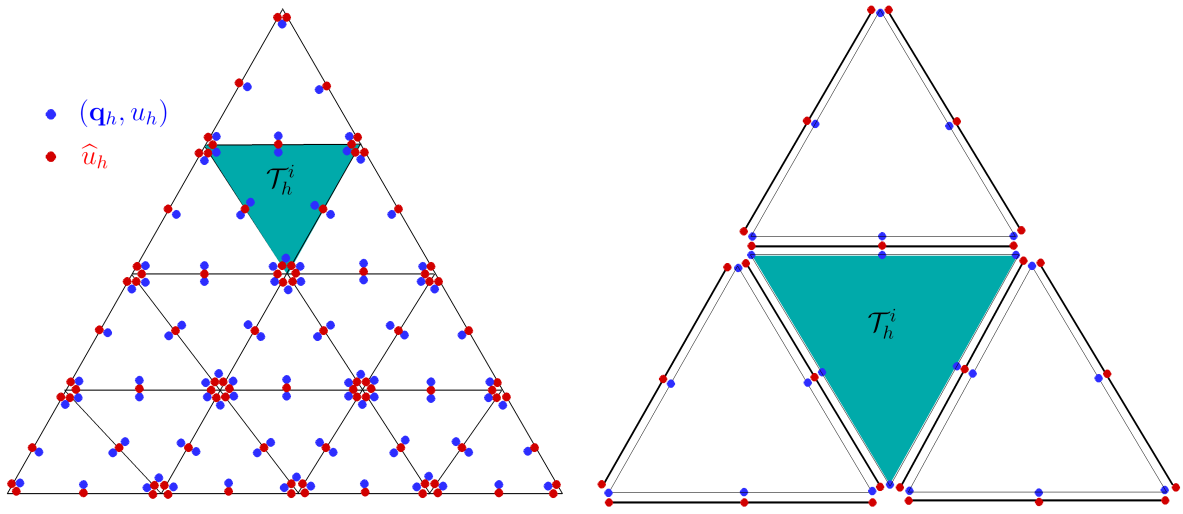
$$\left(\nabla \cdot \left(\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h\right), w_h\right)_K = \left(f, w_h\right)_K \qquad \forall w_h \in W(K), \ K \in \mathcal{T}_h, \tag{7b}$$

$$\langle [\![\left(\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h\right) \cdot \boldsymbol{n}]\!], \hat{\mu}_h \rangle_F = 0 \qquad \forall \hat{\mu}_h \in M(F), \ F \in \mathcal{E}_h^\circ, \tag{7c}$$

$$\langle u_h, \hat{\mu}_h \rangle_F = \langle g_D, \hat{\mu}_h \rangle_F \qquad \forall \hat{\mu}_h \in M(F), \ F \in \Gamma_D, \tag{7d}$$

$$\langle \left(\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h\right) \cdot \boldsymbol{n}, \hat{\mu}_h \rangle_F = \langle g_N, \hat{\mu}_h \rangle_F \qquad \forall \hat{\mu}_h \in M(F), \ F \in \Gamma_N. \tag{7e}$$

如果 $[\![\left(\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h\right) \cdot \boldsymbol{n}]\!]$是 $M(F)$的一个元素，那么 (7c) 表明，在所有面 $F \in \mathcal{E}_h^\circ$上， $[\![\left(\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h\right) \cdot \boldsymbol{n}]\!] = 0$逐点成立，这意味着 $\boldsymbol{a}u_h + \kappa \boldsymbol{q}_h$ 的法向分量在内部面上连续。否则，法向连续性是在弱意义下满足的。



(a) 标准 HDG **(b)** 局部自由度放大图

图 1 使用 $p = 2$ 的标准混合间断伽辽金方法的自由度示意图。黑线表示标准间断单元的边界。

3.2 插值空间

标准 HDG 方法采用全间断单元，其中空间 $W(K)$，$\boldsymbol{V}(K)$和 $M(F)$均为多项式空间。有几种选择能够得到适定的离散系统，参见[6]。我们在单纯形单元上使用等阶插值空间，其中 $W(K)$和 $\boldsymbol{V}(K)$分别由单元 K上总次数为 p的标量多项式和向量值多项式构成，而 $M(F)$是面 F上总次数为 p的标量多项式空间，即

$$W(K) := \mathcal{P}_p(K), \tag{8a}$$
$$\boldsymbol{V}(K) := \left(\mathcal{P}_p(K)\right)^d, \tag{8b}$$
$$M(F) := \mathcal{P}_p(F). \tag{8c}$$

我们的混合间断伽辽金方法宏单元变体采用由标准 C^0 连续单元组成的片区，这些单元仅在片区边界处不连续。因此，在宏单元上，我们使用连续分片多项式，即，

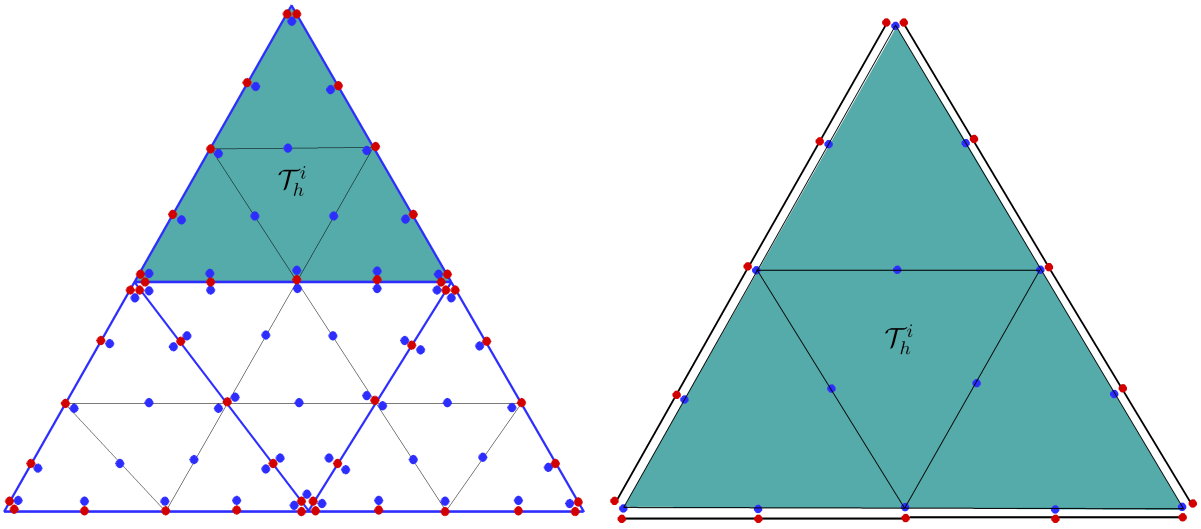
$$W(K) := \{\, w \in C^0(K) \;:\; w|_{\mathrm{e}} \in \mathcal{P}_p(\mathrm{e}),\ \forall \mathrm{e} \in K \,\}, \tag{9a}$$
$$\boldsymbol{V}(K) := \{\, v \in \left(C^0(K)\right)^d \;:\; w|_{\mathrm{e}} \in \left(\mathcal{P}_p(\mathrm{e})\right)^d,\ \forall \mathrm{e} \in K \,\}, \tag{9b}$$
$$M(F) := \{\, m \in C^0(F) \;:\; m|_{\mathrm{f}} \in \mathcal{P}_p(\mathrm{f}),\ \forall \mathrm{f} \in F \,\}. \tag{9c}$$

图 1 和图 2 分别展示了标准 HDG 与宏单元混合间断伽辽金的自由度结构。很容易看出，宏单元混合间断伽辽金方法包含标准混合间断伽辽金方法这一特例。

在 HDG 框架中使用宏单元这一概念，旨在构造一种 HDG 变体，使其能够根据可用计算资源进行轻松调整，无论是在共享内存系统还是分布式内存系统中。此外，HDG 方法的宏单元变体还隐含地包含了一种简单的区域分解方法。

利用数值通量的关系，我们可以将弱公式概括如下：求 $U_h = (\boldsymbol{q}_h, u_h) \in \boldsymbol{V}^h \times W^h$ 和 $\hat{v}_h \in M^h(0)$，使得

$$A\left(U_h, V_h\right) + B\left(\hat{v}_h, V_h\right) = l\left(V_h\right) \qquad \forall V_h = (\boldsymbol{v}_h, w_h) \in \boldsymbol{V}^h \times W^h, \tag{10a}$$
$$C\left(U_h, \hat{\mu}_h\right) + D\left(\hat{v}_h, \hat{\mu}_h\right) = m\left(\hat{\mu}_h\right) \qquad \forall \hat{\mu}_h \in M^h(0), \tag{10b}$$



(a) 宏单元混合间断伽辽金 **(b)** 局部自由度放大图

图 2 带有 $p = 2$ 的宏单元混合间断伽辽金方法自由度示意图。蓝线表示宏单元片区的边界，黑线表示每个宏单元内 C^0- 连续单元的边界。

其中，对于任意 U_h，$V_h \in \boldsymbol{V}^h \times W^h$ 和 $\hat{v}_h$，$\hat{\mu}_h \in M^h(0)$，我们有

$$A\left(U_h, V_h\right) = \left(\boldsymbol{q}_h, \boldsymbol{v}_h\right)_{\mathcal{T}_h} - \left(u_h, \nabla \cdot \boldsymbol{v}_h\right)_{\mathcal{T}_h} - \left(\boldsymbol{a} u_h + \kappa \boldsymbol{q}_h, \nabla w_h\right)_{\mathcal{T}_h} + \langle \kappa \boldsymbol{q}_h \cdot \boldsymbol{n} + \tau u_h, w_h \rangle_{\partial \mathcal{T}_h}, \tag{10c}$$

$$B\left(\hat{v}_h, V_h\right) = \langle \hat{v}_h, \boldsymbol{v}_h \cdot \boldsymbol{n} \rangle_{\partial \mathcal{T}_h} + \langle (\boldsymbol{a} \cdot \boldsymbol{n} - \tau)\, \hat{v}_h, w_h \rangle_{\partial \mathcal{T}_h}, \tag{10d}$$

$$l\left(V_h\right) = \left(f, w_h\right)_{\mathcal{T}_h} - \langle g_D, \boldsymbol{v}_h \cdot \boldsymbol{n} \rangle_{\Gamma_D}, \tag{10e}$$

$$C\left(U_h, \hat{\mu}_h\right) = \langle [\![\kappa \boldsymbol{q}_h \cdot \boldsymbol{n} + \tau u_h]\!], \hat{\mu}_h \rangle_{\mathcal{E}_h}, \tag{10f}$$

$$D\left(\hat{v}_h, \hat{\mu}_h\right) = \langle [\![(\boldsymbol{a} \cdot \boldsymbol{n} - \tau) \hat{v}_h]\!], \hat{\mu}_h \rangle_{\mathcal{E}_h}, \tag{10g}$$

$$m\left(\hat{\mu}_h\right) = \langle g_N, \hat{\mu}_h \rangle_{\Gamma_N}. \tag{10h}$$

(10) 中的变分陈述只包含一阶微分算子。因此，在每个宏单元上定义的 C^0-连续插值空间下，弱形式 (10) 仍然是良定义的。因此我们得出结论：弱形式 (10) 对标准混合间断伽辽金变体和宏单元混合间断伽辽金变体均成立。

3.3 线性方程组全局部分的无矩阵求解

3.3.1 矩阵方程与求解方法

式（10）中的弱形式自然地转化为以下矩阵方程组

$$\begin{bmatrix} \mathbf{A} & \mathbf{B} \\ \mathbf{C} & \mathbf{D} \end{bmatrix} \begin{bmatrix} \mathbf{u} \\ \hat{\mathbf{u}} \end{bmatrix} = \begin{bmatrix} \mathbf{R}_u \\ \mathbf{R}_{\hat{u}} \end{bmatrix}, \tag{11}$$

其中，$\mathbf{u} \in \mathbb{R}^z$ 表示将区域离散为不连续子区域（标准单元或宏单元）所对应的自由度，$\hat{\mathbf{u}} \in \mathbb{R}^{\hat{z}}$ 表示与网格骨架上的迹变量离散化相关联的自由度向量。由于近似空间具有不连续性，且片区通过界面变量耦合，这些矩阵具有特殊的块结构。具体而言，矩阵 $\mathbf{A}$ 是一个块对角矩阵，其中每个块都对应一个宏单元。$\mathbf{A}$ 的所有块矩阵均可逆。

利用舒尔补，我们可以通过消去自由度 $\mathbf{u}$，求解上述关于 $\hat{\mathbf{u}}$ 的矩阵问题，从而得到

$$\left(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)\hat{\mathbf{u}}=\mathbf{R}_{\hat{u}}-\mathbf{C}\mathbf{A}^{-1}\mathbf{R}_{u} \quad \text{(Solve for interface dofs - global problem),} \tag{12a}$$

$$\mathbf{A}\,\mathbf{u}=\mathbf{R}_{u}-\mathbf{B}\hat{\mathbf{u}} \quad \text{(Solve for interior dofs - local problem).} \tag{12b}$$

通常使用迭代求解器来求解全局问题。在并行计算环境中，采用无矩阵求解方法可以特别高效地完成这一任务。

3.3.2 无矩阵求解方法

我们考虑以下针对全局问题 (12a) 的迭代求解方法。首先计算 $\mathbf{f}=\mathbf{R}_{\hat{u}}-\mathbf{C}\mathbf{A}^{-1}\mathbf{R}_{u}$。然后使用迭代求解器求解 $\left(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)\hat{\mathbf{u}}=\mathbf{f}$。这里，我们利用矩阵 $\mathbf{A}$，$\mathbf{B}$，$\mathbf{C}$，$\mathbf{D}$ 的块结构，在局部并行地执行所有运算。令下标 (e) 表示将运算局部化到（宏）单元层级，(f) 表示将运算局部化到（宏）单元面。这样，无矩阵算法的每次迭代都可以针对每个（宏）单元 e 并行执行，具体分为以下四个步骤

1. $\mathbf{x}_{(\mathrm{e})}=\mathbf{B}_{(\mathrm{e})}\hat{\mathbf{u}}_{(\mathrm{e})}$,
2. $\mathbf{y}_{(\mathrm{e})}=\mathbf{A}_{(\mathrm{e})}^{-1}\mathbf{x}_{(\mathrm{e})}$,
3. $\hat{\mathbf{v}}_{(\mathrm{e})}=\mathbf{C}_{(\mathrm{e})}\mathbf{y}_{(\mathrm{e})}$.

第四步涉及一项归约操作，需要使用与网格骨架面相关的数据以及来自（宏）单元的数据，因此需要在处理器之间进行通信。

4. $\hat{\mathbf{w}}_{(\mathrm{f})}=\mathbf{D}_{(\mathrm{f})}\hat{\mathbf{u}}_{(\mathrm{f})}-\hat{\mathbf{v}}_{(\mathrm{f}1)}-\hat{\mathbf{v}}_{(\mathrm{f}2)}$.

注*1.* 通过就地计算并存储其 LU分解来应用逆算子 $\mathbf{A}_{(\mathrm{e})}$。

注*2.* 一种简单的预条件　子是将 $\mathbf{D}_{(\mathrm{f})}^{-1}$ 应用于无矩阵算法的第四步。

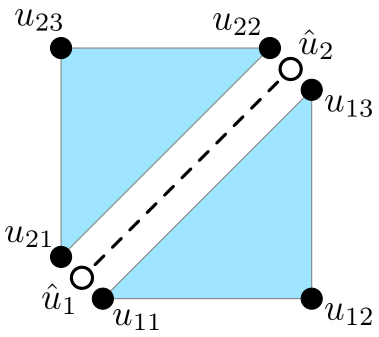


图3 具有齐次边界条件和一个宏单元界面的双宏单元网格示例离散化。

例1。 考虑图3所示的双宏单元网格。我们关注的是在求解 (12a) 的迭代过程中执行一次矩阵-向量乘法。前三个步骤可以如下进行

$$\begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ \hline x_{21} \\ x_{22} \\ x_{23} \end{bmatrix} = \underbrace{\left[\begin{array}{cc} * & * \\ * & * \\ * & * \\ \hline * & * \\ * & * \\ * & * \end{array}\right]}_{\mathbf{B}} \begin{bmatrix} \hat{u}_1 \\ \hat{u}_2 \end{bmatrix}, \qquad \begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ \hline y_{21} \\ y_{22} \\ y_{23} \end{bmatrix} = \underbrace{\left[\begin{array}{ccc|ccc} * & * & * & & & \\ * & * & * & & & \\ * & * & * & & & \\ \hline & & & * & * & * \\ & & & * & * & * \\ & & & * & * & * \end{array}\right]}_{\mathbf{A}^{-1}} \begin{bmatrix} x_{11} \\ x_{12} \\ x_{13} \\ \hline x_{21} \\ x_{22} \\ x_{23} \end{bmatrix}, \qquad \begin{bmatrix} \hat{v}_{11} \\ \hat{v}_{12} \\ \hline \hat{v}_{21} \\ \hat{v}_{22} \end{bmatrix} = \underbrace{\left[\begin{array}{ccc|ccc} * & * & * & & & \\ * & * & * & & & \\ \hline & & & * & * & * \\ & & & * & * & * \end{array}\right]}_{\mathbf{C}} \begin{bmatrix} y_{11} \\ y_{12} \\ y_{13} \\ \hline y_{21} \\ y_{22} \\ y_{23} \end{bmatrix}.$$

此处的水平线和垂直线表示矩阵和向量的剖分，这些矩阵和向量可能分布在不同的处理器上。因此，网格中每个宏单元的所有计算都可以在本地完成。

最后一步，即第四步，可以针对网格骨架中的每个宏单元面执行。该步骤涉及归约，因此需要处理器间通信。

$$\begin{bmatrix} \hat{w}_1 \\ \hat{w}_2 \end{bmatrix} = \underbrace{\begin{bmatrix} * & * \\ * & * \end{bmatrix}}_{\mathbf{D}} \begin{bmatrix} \hat{u}_1 \\ \hat{u}_2 \end{bmatrix} - \begin{bmatrix} \hat{v}_{11} \\ \hat{v}_{12} \end{bmatrix} - \begin{bmatrix} \hat{v}_{21} \\ \hat{v}_{22} \end{bmatrix}.$$

3.4 对流主导问题的额外稳定化

对于对流主导问题，我们必须预期，标准 C^0 连续有限元中常见的解振荡会局部出现在每个宏单元内，尤其是在包含大量 C^0 连续单元的片区中。为减轻宏单元内的这些振荡，我们在每个宏单元内采用流线迎风 Petrov-Galerkin（SUPG）方法[37,38,39,40,41]，加入标准的基于残差的稳定化。修改后的双线性型和线性型如下：

$$B_{supg}\left(U_h, V_h\right) = B\left(U_h, V_h\right) + \sum_{\mathrm{e}\in K} \int_{\mathrm{e}} \mathcal{P}(w_h)\tau_{supg} \nabla \cdot \left(\boldsymbol{a} u_h - \kappa \nabla u_h\right), \tag{13a}$$

$$l_{supg}\left(V_h\right) = l\left(V_h\right) + \sum_{\mathrm{e}\in K} \int_{\mathrm{e}} \mathcal{P}(w_h)\tau_{supg} f \tag{13b}$$

with

$$\mathcal{P}(w_h) = \boldsymbol{a} \cdot \nabla w_h, \tag{13c}$$

$$\tau_{supg} = \frac{h}{2|\boldsymbol{a}|}\left(\coth \mathrm{Pe}_{\mathrm{e}} + 1/\mathrm{Pe}_{\mathrm{e}}\right), \tag{13d}$$

其中，Pe_{e} 为单元佩克莱数。

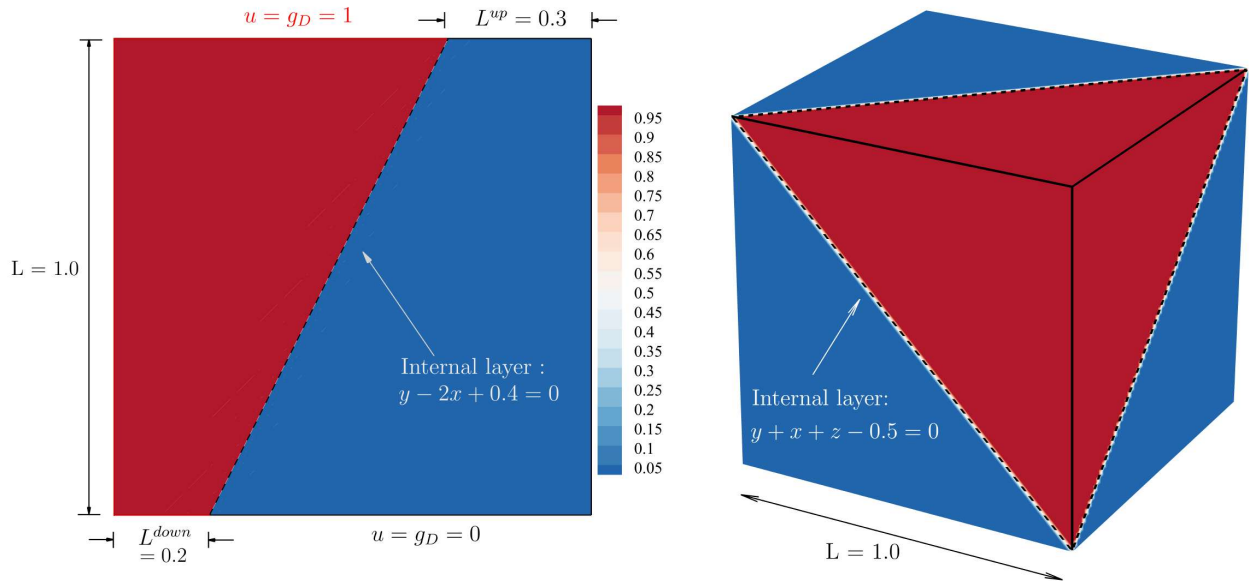
4 宏单元混合化 DG 方法的性质

下面，我们首先展示所提出的混合间断伽辽金方法宏单元变体的精度和收敛性。随后，我们将讨论并突出介绍该离散化策略在自适应加密、区域分解和负载均衡方面的实际优势。

4.1 精度与收敛

4.1.1 一项具有可自定义内部层网格倾斜度的基准测试

基于模型问题 (1)，我们采用具有以下精确解的基准测试，



(a) 二维域 **(b)** 三维域

图4 具有可自定义且相对于网格倾斜的内部层的对流-扩散基准问题的精确解（Pe $\gg 1$），分别定义在单位正方形和单位立方体上。

$$u_{exact} = 0.5\left(1 + \tanh\left(\frac{y - 2x + 0.4}{\kappa}\right)\right) \qquad \text{for } d = 2, \tag{14}$$

$$u_{exact} = 0.5\left(1 + \tanh\left(\frac{x + y + z - 0.5}{\kappa}\right)\right) \qquad \text{for } d = 3, \tag{15}$$

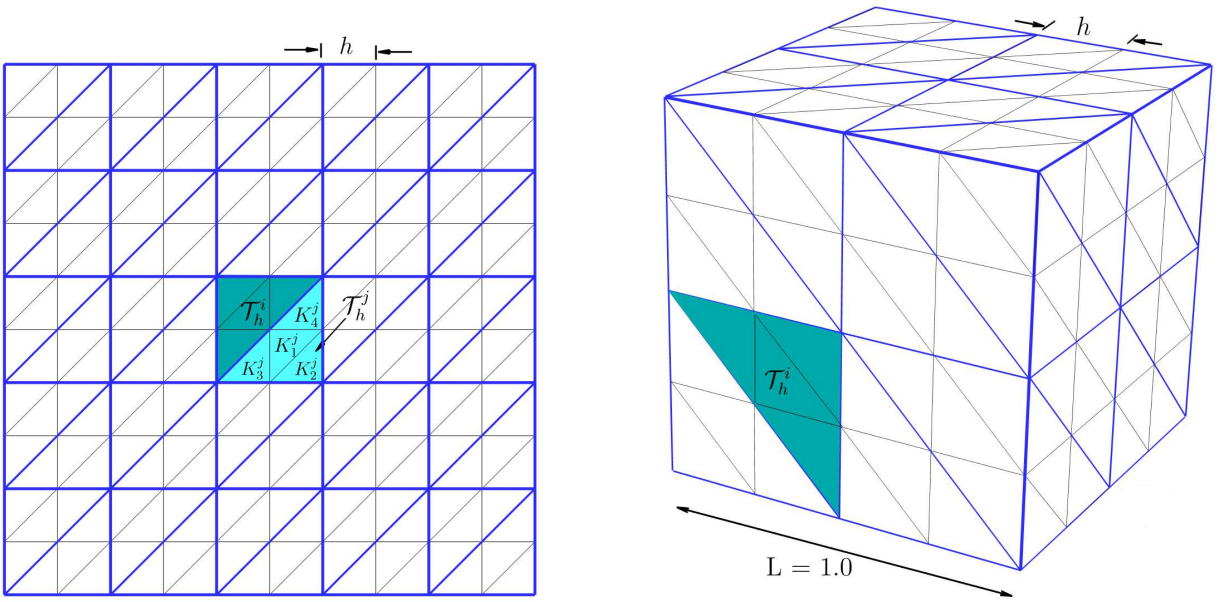
该精确解的灵感来自一维 Allen-Cahn问题的稳态相场解，该解具有弥散界面，参见例如[42] 及其中的参考文献。场解(14)和(15)分别为空间平移和旋转后的变体，因此会形成与（结构化）网格倾斜、锐度各异的内部层。对于二维和三维情形，我们分别选择直线$2x - y = 0.4$ 和曲面 $x + y + z = 0.5$ 作为弥散内部层的中心。相应的边界条件可直接由(14)和(15)推导得到；将解(14)和(15)代入对流-扩散方程(1)，并求解 f，即可确定相应的源项。我们强调，只要选择的对流速度场 $\boldsymbol{a}$ 为常量，且方向与内部层一致， f 就与 $\boldsymbol{a}$无关，因为 $\boldsymbol{a} \cdot \nabla u = 0$。

4.1.2 扩散主导与对流主导情形

解的状态由佩克莱数 Pe $= |\boldsymbol{a}| L/\kappa$决定，其中 L 表示区域 Ω的特征长度尺度。在我们的基准测试中，对于给定的速度和长度尺度，可以通过选择不同的扩散系数 κ 来调整佩克莱数。图4a和图4b绘出了单位正方形和单位立方体上的精确解（14）和（15），展示了对流主导状态（Pe $\gg 1$）下的尖锐内部层。

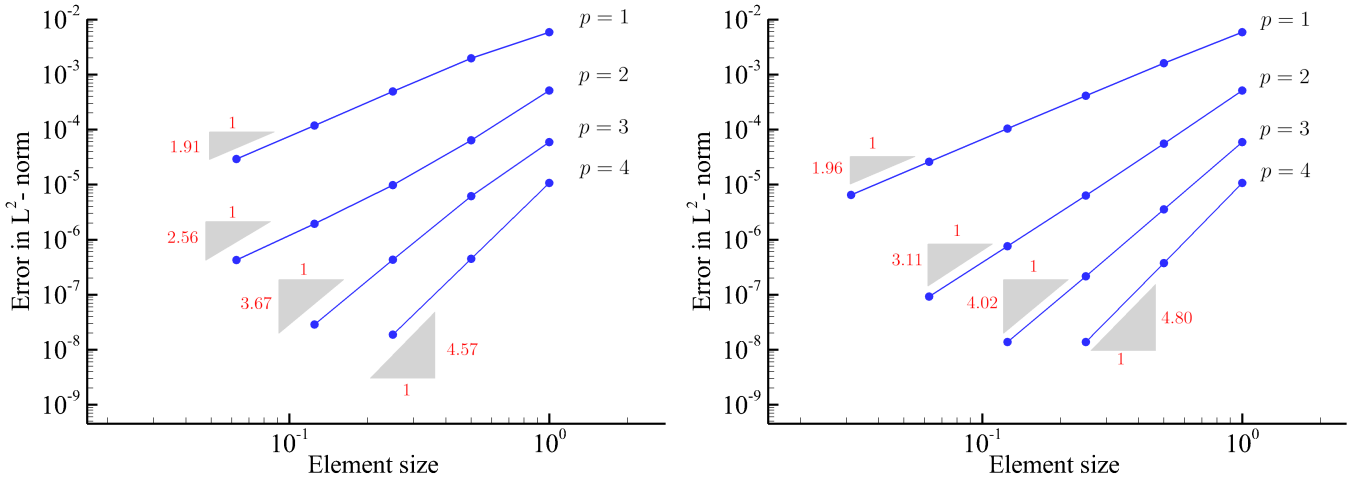
我们首先评估宏单元HDG变体能否达到最优收敛阶。为此，我们在二维中将 $\kappa = 0.4$ 固定为 $\boldsymbol{a} = (1, 1)$，在三维中固定为 $\boldsymbol{a} = (1, 1, 1)$，两者均与内部层平行，从而得到扩散主导的解。我们考虑从 $p = 1$ 到 $p = 4$ 的多项式阶数，并采用图 5a 和图 5b 所示的网格，其中每个方向上沿宏单元边的单元数 m 均取为二。

在图 6a 和 7a 中，我们分别绘制了二维基准问题和三维基准算例在宏单元均匀细化下， L^2 范数误差随单元尺寸的变化。我们观察到，所有情况下均达到了最优收敛阶 $p+1$。我们还使用标准 HDG 方法计算了相同的算例，并绘制了所得的



(a) 二维结构化网格。 **(b)**三维结构化网格。

图 5 标准单元（黑色）和宏单元（蓝色）的示意图。

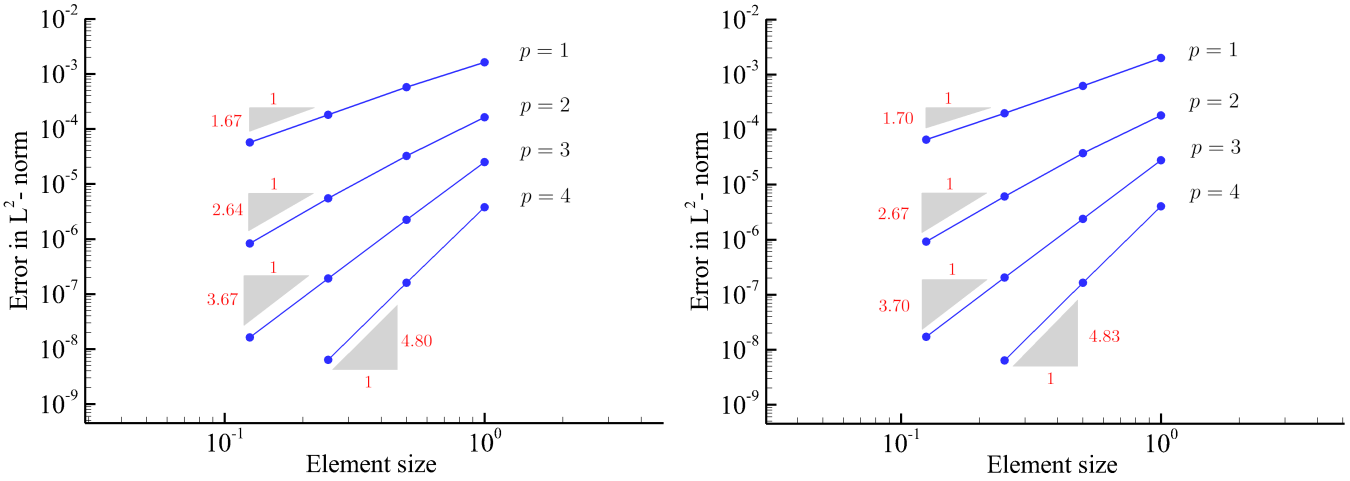


(a) 宏单元混合间断伽辽金（$m = 2$）。 **(b)** 标准 HDG。

图 6 **误差** L^2 对于采用均匀细化的二维基准算例，在 $\kappa = 0.4$ (Pe $= 3.5$) 下得到的误差 nt.

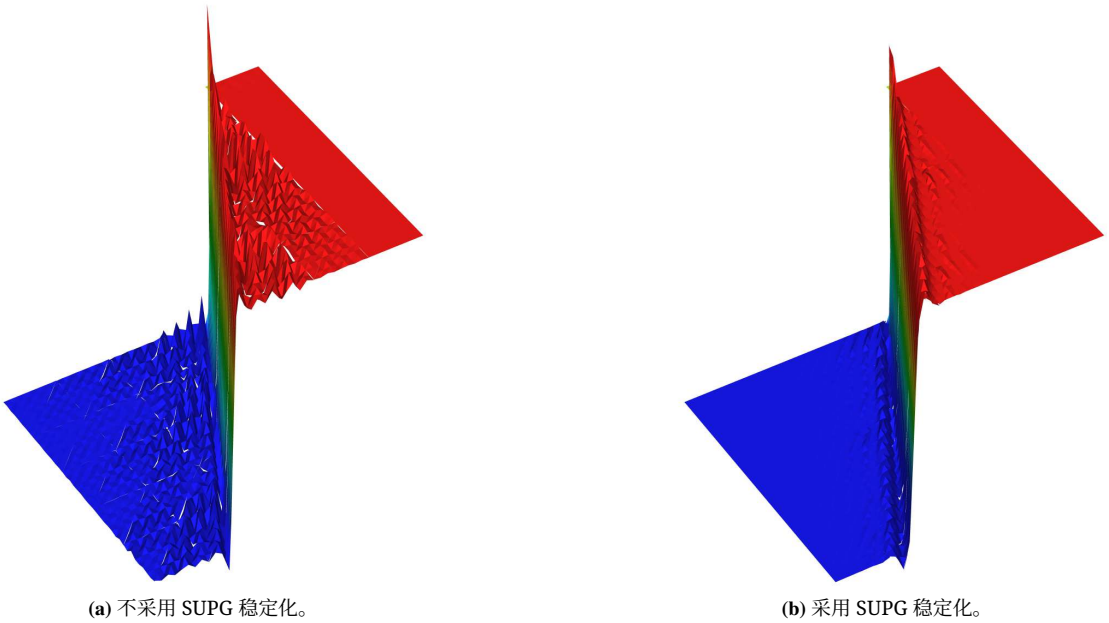
收敛曲线，如图 6b 和 7b 所示。我们观察到，这两种 HDG 变体在误差水平和收敛率方面几乎没有差异。

接下来，我们展示在对流占优情形下，采用宏单元混合间断伽辽金方法时，加入额外稳定化对于获得准确解的重要性。为此，我们将扩散系数降低至 $\kappa = 10^{-5}$。首先，我们绘制图 8a 中的解，其中使用图 5a 所示的网格，且未加入额外稳定化。总体而言，解是稳定的，但我们观察到，每个宏单元内的解都表现出强烈的振荡行为。加入 SUPG 稳定化后，图 8b 中的这些振荡大幅消失。过冲和欠冲有所减轻，较差的行为仅限于



(a) 宏单元混合间断伽辽金（$m = 2$）。 **(b)** 标准 HDG。

图 7 L^2 在均匀细化下，使用 $\kappa = 0.4$ （Pe $= 4.3$）进行三维基准测试所获得的误差 ent.



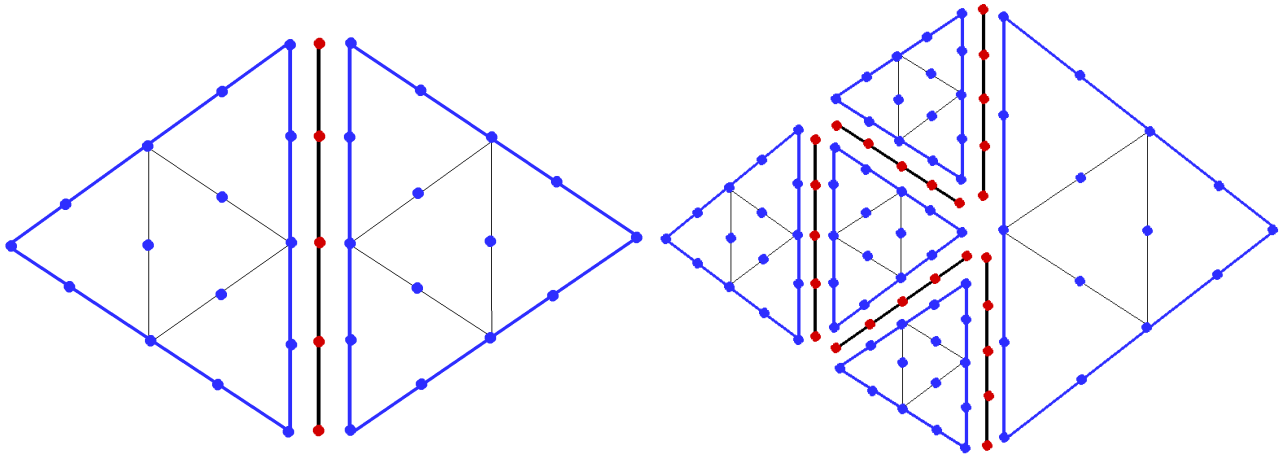
(a) 不采用 SUPG 稳定化。 **(b)** 采用 SUPG 稳定化。

图 8 对流占优情形下（$\kappa = 10^{-5}$，Pe $= \sqrt{2} \cdot 10^5$）应用和未应用 SUPG 稳定化的宏单元混合间断伽辽金方法。

尖锐层附近的一带宏单元。本文余下部分中，所有采用宏单元混合间断伽辽金方法获得的结果均使用 SUPG 稳定化。

4.2 局部自适应加密

标准混合间断伽辽金方法通过不连续基函数和迹变量的存在，提供了一种简单的局部细化机制。单元可以递归细分，而选择合适的迹空间是保持



(a)局部细分之前　　**(b)** 局部细分之后

图 9　$p = 2$和 $m = 2$对应的宏单元及其迹变量的局部细化 .

单元之间弱连续性的关键[43]。这种局部细化能力自然推广到宏单元混合间断伽辽金方法。同样，选择合适的迹空间是保持跨宏单元边界弱连续性的关键。

图 9 展示了我们采用的一种细化方法示例。宏单元采用二分细化。相应的迹变量也会细化，使其能够在局部表示细化后宏单元变量的迹。这意味着法向上的连续性得以精确保留。还有其他选择，但根据我们的计算测试，这种选择通常最准确，在实践中也效果良好。

在涉及边界层和内部层的问题中，局部细化至关重要。图 10 展示了为捕捉对流占优情形（$\mathrm{Pe} = 10^{10}$）中出现的内部层而生成的一系列网格。为避免图示过于繁杂，我们仅绘制宏单元。这里，我们采用自动化的基于梯度的细化方案，网格规模为 $m = 2$，次数为$p = 2$。图 11a 和 11b 分别展示了均匀加密与自适应加密对应的收敛曲线，以及最细的第 9 级网格上的解图。我们观察到，局部细化大幅提高了该方法的有效性。

4.3 区域分解与负载均衡

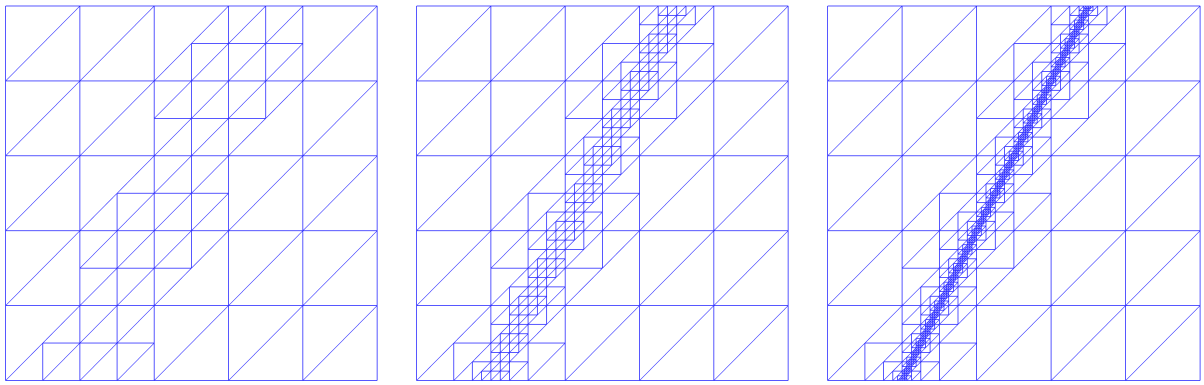
当计算需要在大型分布式内存系统上运行时，必须考虑各处理器之间计算负载的差异，并确保各处理器的负载尽可能均衡。未能处理负载不均衡会严重影响性能，并限制强扩展性和弱扩展性。改善负载均衡可以显著缩短运行时间并提升加速性能。

为分析实际性能，我们采用“负载均衡性能因子”（LBF）来衡量负载均衡效率[44],，其计算方式为各处理器平均耗时与所有处理器中最长耗时之比：

$$\mathrm{LBF} = \frac{\text{Avg. workload of all proc's}}{\text{Max. workload among proc's}} .$$

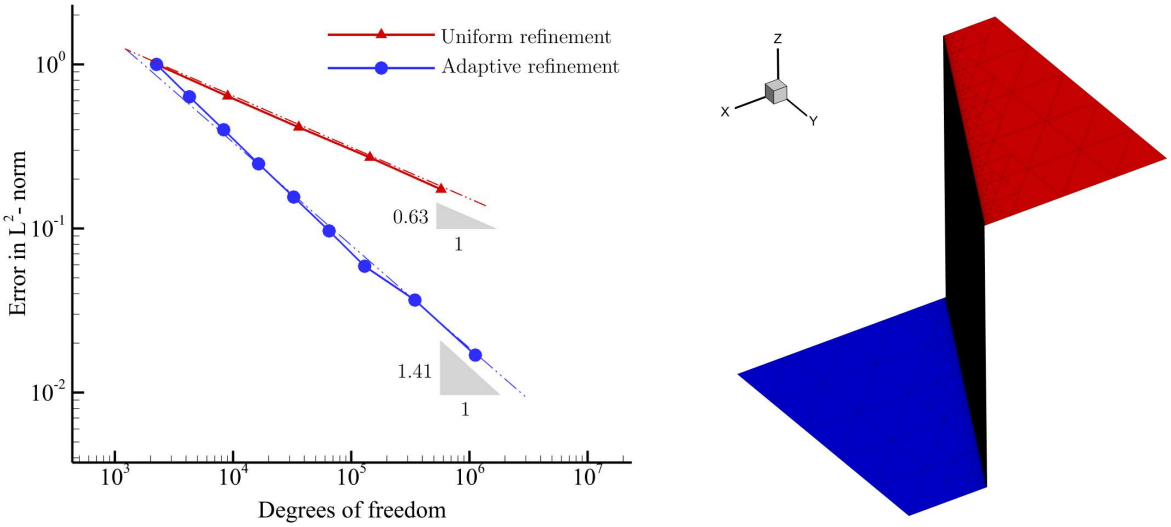
完全平衡的方案所得比值为 1.0，而负载均衡效果较差的方案所得比值则小于 1.0。文献中对此有不同定义，但几乎所有定义都基于所用时间的平均值和最大值。

使用将在第 6.1 节中详细介绍的计算配置和环境，我们计算了三维基准测试，其离散化包含总计 $N^{Elm} = 196,608$ 个单元。随后，我们收集了混合间断伽辽金方法宏单元变体的 LBF，其中宏单元沿宏单元边包含两个单元（$m = 2$）。我们首先选用线性基函数（$p = 1$），此时全局求解器面对的系统有 $\mathrm{dof}^{global} = 608{,}256$ 个自由度。图 12 展示了在单个计算节点上使用 8、16 和 32 个处理器并行执行时，局部和全局求解器的执行时间分布。根据这些结果，无论是局部还是全局计算，负载均衡都非常出色，LBF 始终高于 0.9。我们还考察了每个处理器自身的工作负载如何随着计算



(a) 第1级。 **(b)** 第3级。 **(c)** 第5级。

图 10 局部宏单元细化序列，适用于对流占优情形下的二维基准算例（$\mathrm{Pe} = 10^{10}$）。



(a)均匀加密和自适应加密下的收敛。 **(b)** 细化级别9下得到的解。

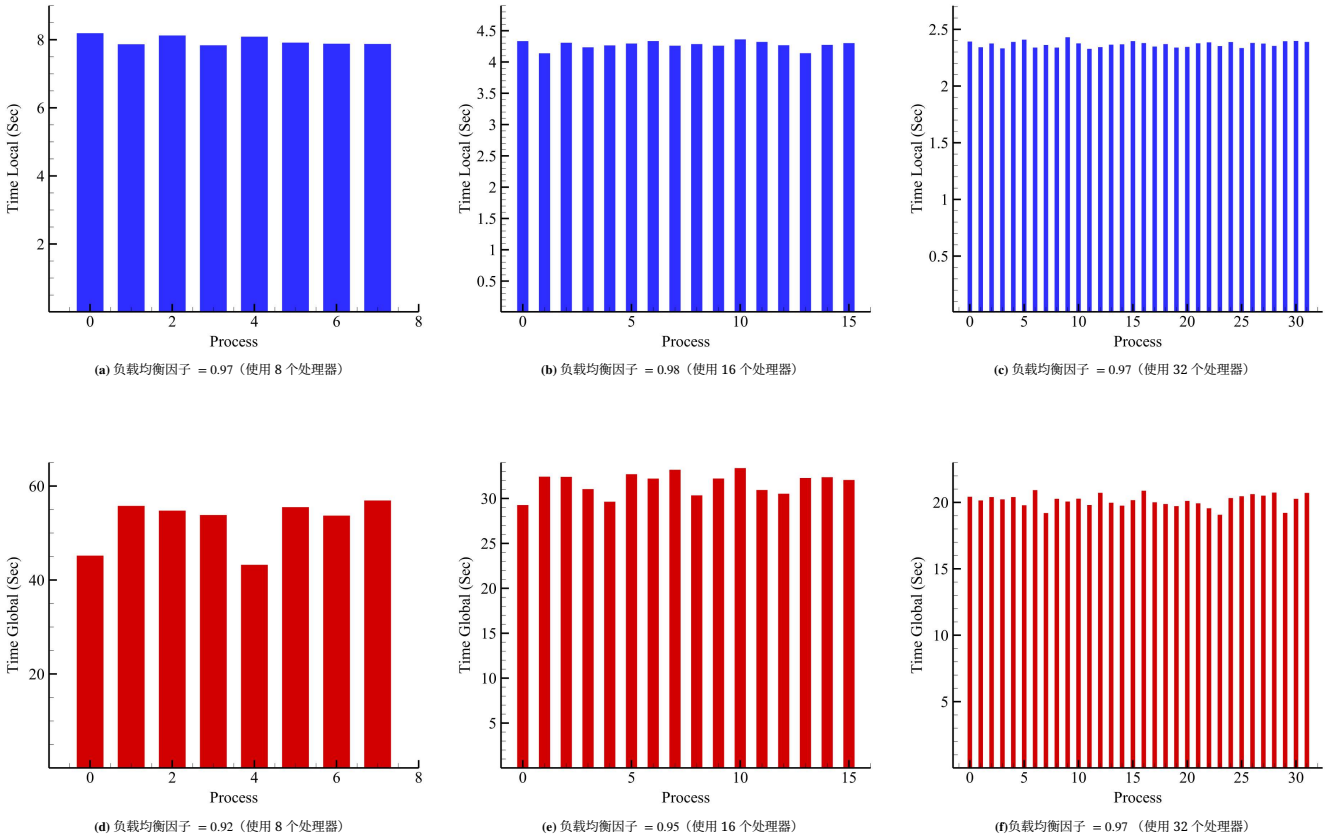
图11 收敛性：二维基准算例（$\mathrm{Pe} = 10^{10}$）在均匀和局部细化下的表现，其中 $m = 2$和 $p = 2$。

节点数量的变化而扩展，其中每个节点均使用全部 32 个处理器。为了产生足够的负载，我们将多项式次数提高到 $p = 2$，此时局部和全局求解器分别面对具有 $\mathrm{dof}^{local} = 3{,}440{,}640$ 和 $\mathrm{dof}^{global} = 1{,}216{,}512$ 个自由度的系统。结果如图13所示。我们观察到节点间通信带来轻微影响，但 LBF 仍约为 0.9。对于 4 个节点和 128 个处理器，局部求解器和全局求解器的 LBF 分别为 0.87 和 0.91。

5 计算效率：理论估计

接下来要回答的关键问题是，我们提出的混合间断伽辽金方法宏单元变体能否在计算成本和可扩展性方面实现有竞争力的计算性能。本节首先通过理论估计来探讨这一问题。

我们注意到，近期一篇预印本针对一种非常相似的方法研究了这一问题[45]，，并得出结论：“在三维情形下，组装和求解离散传输条件的成本仍高得令人难以承受。”一种宏单元混合间断伽辽金



(a) 负载均衡因子 = 0.97（使用 8 个处理器）
(b) 负载均衡因子 = 0.98（使用 16 个处理器）
(c) 负载均衡因子 = 0.97（使用 32 个处理器）
(d) 负载均衡因子 = 0.92（使用 8 个处理器）
(e) 负载均衡因子 = 0.95（使用 16 个处理器）
(f)负载均衡因子 = 0.97（使用 32 个处理器）

图 12 单个**计算节点**上宏单元混合间断伽辽金的负载均衡——局部（蓝色）和全局（红色）求解器的时间以及负载均衡因子（LBF）（自由度^{global} = 608，256）。

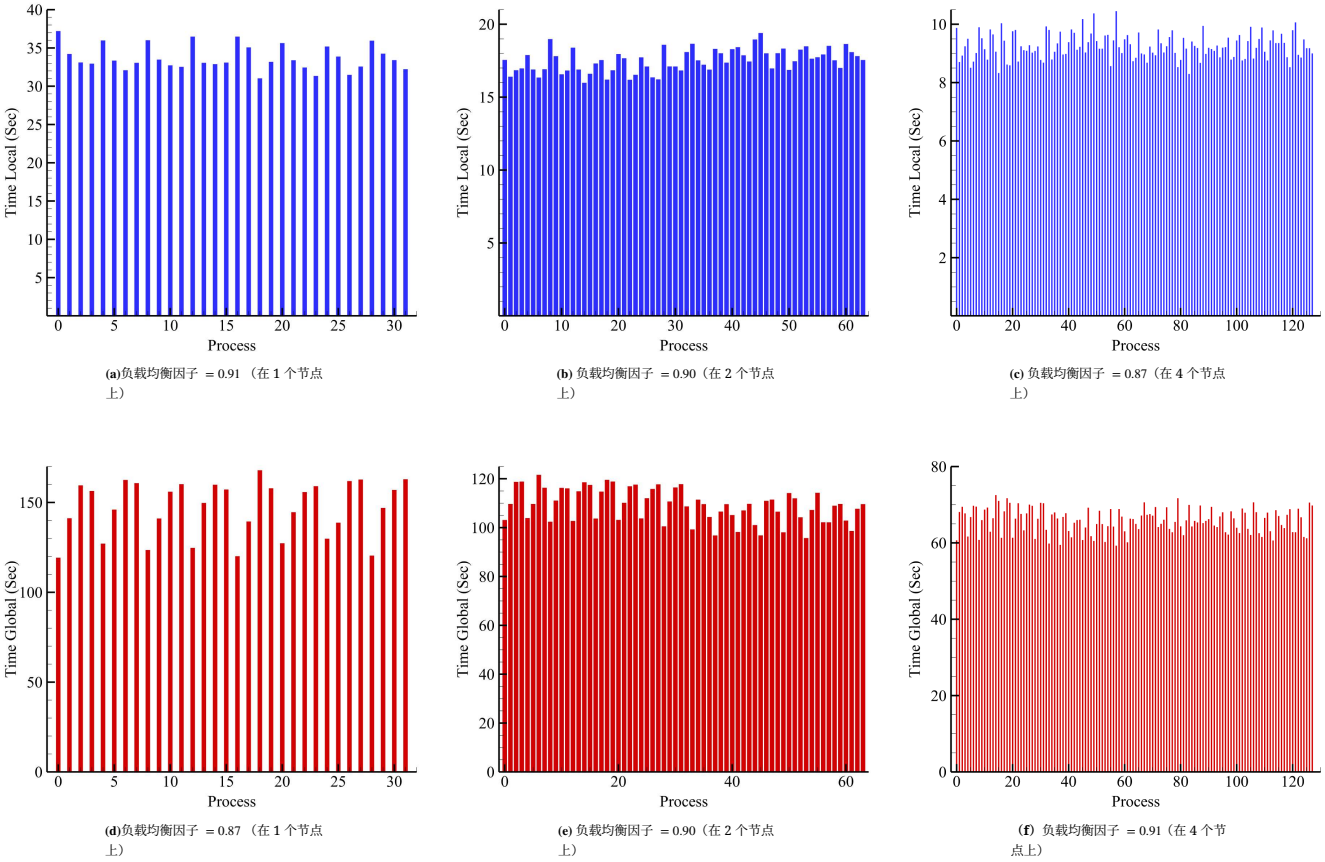
方法因此被认为通常效率低下，不受推荐。我们强调，我们的复杂度分析得出了不同的结果，也揭示了局限性，但与[45]中报告的结果相比，前景要乐观得多。

5.1 复杂度分析

回顾一下，d (= 2, 3)表示空间维度。我们假设将单位立方体 $[0,1]^d$ 结构化离散为一组 $N = d!\,n^d$ 单纯形宏单元，其中 n 表示每个空间方向上的宏单元数量。每个宏单元再划分为一组 m^d 单纯形 C^0连续有限元，其中 m 表示每个宏单元在每个空间方向上的单元数量。我们进一步假设，每个单纯形单元上采用一个总次数为 p的多项式基。表1和表2总结了我们在分析中使用的定义量和相关量。

表1 定义量。

d	空间维度（= 2，3）
n	单位立方体每条边上的宏单元数量 $[0,1]^d$
m	宏单元边上的单元数量
p	单元多项式阶数



(a)负载均衡因子 = 0.91（在 1 个节点上）
(b) 负载均衡因子 = 0.90（在 2 个节点上）
(c) 负载均衡因子 = 0.87（在 4 个节点上）
(d)负载均衡因子 = 0.87（在 1 个节点上）
(e) 负载均衡因子 = 0.90（在 2 个节点上）
(f) 负载均衡因子 = 0.91（在 4 个节点上）

图13 宏单元混合间断伽辽金在**多个计算节点**上的负载均衡，每个节点包含 32 个进程——局部求解器（蓝色）和全局求解器（红色）的时间以及负载均衡因子（LBF）（$\mathrm{dof}^{local} = 3$，440、640、$\mathrm{dof}^{global} = 1$，216、512）。

我们关注宏单元混合间断伽辽金方法（$m > 1$）与标准混合间断伽辽金方法（$m = 1$）之间的比较。我们假设，只要单元总数保持相等，即 $n \cdot m =$ 为常数，就能达到相近的精度；后文将对此进行验证。

5.1.1 全局求解器的成本

上述“全局”问题的时间复杂度可以很容易地计算出来。矩阵-向量乘积可以局部化到宏单元层级，即在宏单元 e 上，矩阵-向量乘积为：

1. $\mathbf{x}_{(\mathrm{e})} = \mathbf{B}_{(\mathrm{e})}\hat{\mathbf{u}}_{(\mathrm{e})}$,
2. $\mathbf{y}_{(\mathrm{e})} = \mathbf{A}^{-1}_{(\mathrm{e})}\mathbf{x}_{(\mathrm{e})}$,
3. $\hat{\mathbf{v}}_{(\mathrm{e})} = \mathbf{C}_{(\mathrm{e})}\mathbf{y}_{(\mathrm{e})}$.

第 1 步和第 3 步中的矩阵-向量乘法最多需要 $\mathcal{O}\left((d+1)Q_{(d-1)} \cdot Q_{(d)}\right)$ 次运算。对 $\mathbf{A}_{(\mathrm{e})}$ 进行一次 LU分解，所需成本为 $\mathcal{O}\left(Q^3_{(d)}\right)$ 次运算。在所有后续步骤中应用该分解需要 $\mathcal{O}\left(Q^2_{(d)}\right)$ 次运算。这意味着在初始化过程之后，求解器每次迭代的成本为 $\mathcal{O}\left(Q^2_{(d)}\right)$ 次运算。

注 *3.* $N = d!\,n^d$ 局部问题可以独立并行运行！

表2 相关量。

$N = d!\, n^d$	宏单元总数
$M_{(d)} = m^d$	每个 d 维宏单元中的单元总数
$\begin{cases} Q_{(d)} &= \binom{m\,p+d}{d} \\ &= \frac{1}{d!}(m\cdot p+1)\ \cdots\ (m\cdot p+d) \\ &< \frac{1}{d!}(m\cdot p+d)^d \end{cases}$	每个 d 维宏单元中的自由度总数—— 单元
$\begin{cases} D = A_d\, n^d + B_d\, d\, n^{d-1}(n-1) \\ \text{其中 } (A_2=1, B_2=1, A_3=6, B_3=2) \end{cases}$	构成网格骨架的宏单元面数量 （不包括边界）。 A_d 是 由单纯形划分的立方体内共享面的数量，而 B_d 是 两个 d 维立方体之间共享面的数量。
$d+1$	d 维单纯形的面数
$(2p+1)^d$	C^0的非空交集数量上限 一个宏单元内的测试函数和试探函数。
$C^{\text{sparsity}}_{(d)} = \begin{cases} 1 & \text{对于稠密运算。} \\ (2p+1)^d / Q_{(d)} & \text{对于稀疏运算。} \end{cases}$	矩阵稀疏性的上限估计。

最后的约化步骤针对网格骨架中的每个子域执行。假设 $\hat{\mathbf{v}}_{(\mathrm{f}1)}$ 和 $\hat{\mathbf{v}}_{(\mathrm{f}2)}$ 是与网格骨架中面 f 相邻的两个宏单元所对应的已计算向量。那么，宏单元面上的约化步骤为：

4. $\hat{\mathbf{w}}_{(\mathrm{f})} = \mathbf{D}_{(\mathrm{f})}\hat{\mathbf{u}}_{(\mathrm{f})} - \hat{\mathbf{v}}_{(\mathrm{f}1)} - \hat{\mathbf{v}}_{(\mathrm{f}2)}$.
矩阵乘法和减法至多需要 $\mathcal{O}\left(Q^2_{(d-1)}\right)$ 次运算。

备注 *4.* D 局部问题可以独立运行，并行求解！

备注 *5.* 当 $\hat{\mathbf{u}}$ 的部分内容作为输入传递给步骤 1–3，以及步骤 3 的输出被传回以执行步骤 4 中的归约时，会发生通信。

5.1.2 局部求解器的计算成本

全局问题求解完成后，也就是说，计算出界面自由度 $\hat{\mathbf{u}}$ 后，可以通过分别求解以下问题，重构每个宏单元的局部解

$$\mathbf{A}_{(\mathrm{e})}\mathbf{u}_{(\mathrm{e})} = \mathbf{R}_{u(\mathrm{e})} - \mathbf{B}_{(\mathrm{e})}\hat{\mathbf{u}}_{(\mathrm{e})}. \tag{16}$$

同样，计算得到的 LU 分解会被使用，这需要 $\mathcal{O}\left(Q^2_{(d)}\right)$ 次运算。

5.2 内存估计

现在我们总结网格中每个宏单元以及网格骨架中的宏单元面所需的各个子矩阵的内存估算。

1. $\mathbf{B}_{(\mathrm{e})}$ 和 $\mathbf{C}_{(\mathrm{e})}$: $\text{const}^{\text{sparsity}}_{(d)} \cdot (d+1) \cdot Q_{(d-1)} \cdot Q_{(d)} \cdot 8$ 字节，2. $\mathbf{A}_{(\mathrm{e})} = \mathbf{L}_{(\mathrm{e})}\mathbf{U}_{(\mathrm{e})}$: $\text{const}^{\text{sparsity}}_{(d)} \cdot Q^2_{(d)} \cdot 8$ 字节，3. $\mathbf{D}_{(\mathrm{f})}$: $\text{const}^{\text{sparsity}}_{(d-1)} \cdot Q^2_{(d-1)} \cdot 8$ 字节。

这里假设采用双精度运算。参数 $\text{const}^{\text{sparsity}}_{(d)} \in (0,1]$ 表示关联矩阵的非零元素比例。此外，我们假设在计算出 $\mathbf{A}_{(\mathrm{e})}$ 后，其 LU 分解会原位存储，无需额外内存。

网格中 N 个宏单元对应的矩阵（1. 和 2.）均会存储，并可能分布在多个处理器上。矩阵（3.）则会为网格骨架中的每个 D 宏单元面存储，同样分布在多个处理器上。

注 *6*。当 m 较小时，使用稠密线性代数效率更高。不过，当 m 较大时，例如 $m > 2$，稀疏求解器可能效率更高。这取决于矩阵 $\mathbf{A}$ 的稀疏性（由 m 和 p 决定）以及 $L1 - L2$ 缓存的大小。

5.3 与标准 HDG 的比较

我们关注上述针对不同 m，n 且满足 $n \cdot m = const$ 的运算次数估计。本例着眼于有限元总数相同的情形，因此宏单元混合间断伽辽金与标准 HDG 之间的精度预计具有可比性。设 $\bar{n}$，$\bar{m}$ 表示宏单元混合间断伽辽金方法中 n 和 m 的维数。于是，我们可以根据这些参数，将运算次数与标准混合间断伽辽金方法 $m = 1$、$n = \bar{m} \cdot \bar{n}$ 进行比较。结果见下表 3。

	宏单元混合间断伽辽金	标准 HDG
初始化	$\bar{n}^d(\bar{m} \cdot p + d)^{3d}$	$\bar{m}^d \cdot \bar{n}^d(p + d)^{3d}$
步骤 1	$\bar{n}^d(\bar{m} \cdot p + d)^{2d-1}$	$\bar{m}^d \cdot \bar{n}^d(p + d)^{2d-1}$
步骤 2	$\bar{n}^d(\bar{m} \cdot p + d)^{2d}$	$\bar{m}^d \cdot \bar{n}^d(p + d)^{2d}$
步骤 3	$\bar{n}^d(\bar{m} \cdot p + d)^{2d-1}$	$\bar{m}^d \cdot \bar{n}^d(p + d)^{2d-1}$
步骤 4	$\bar{n}^d(\bar{m} \cdot p + d - 1)^{2d-2}$	$\bar{m}^d \cdot \bar{n}^d(p + d)^{2d-2}$

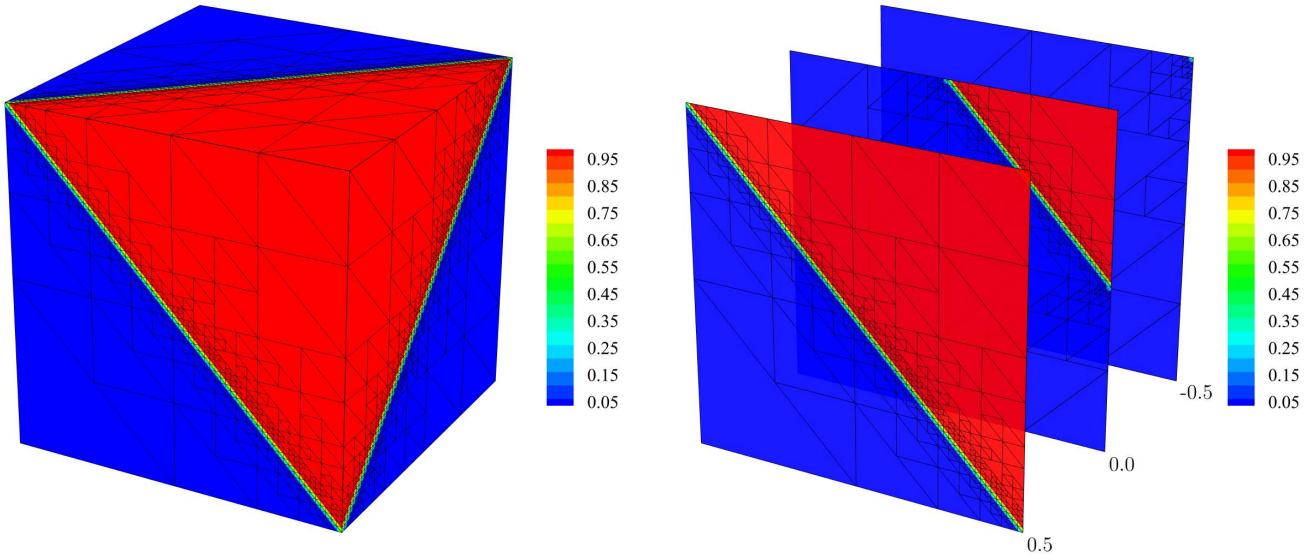
表 3阶数　　初始化一次迭代并在每次迭代中执行步骤 1 至 4 所需的运算次数。

初始化步骤中会计算 LU分解，这是开销最大的步骤，但只需执行一次，而且可以轻松地并行处理。其余步骤则在线性求解器的每次迭代中执行。这里，步骤二是主要开销项。

现在假设标准混合间断伽辽金方法和宏单元混合间断伽辽金方法的迭代次数相同。在这种情况下，采用 $\bar{m} > 1$ 似乎并无益处。这意味着，从运算次数的角度来看，宏单元混合间断伽辽金方法似乎不如标准混合间断伽辽金方法高效。不过，这项分析并未考虑若干参数。标准混合间断伽辽金方法需要更多处理器间通信，因为其局部问题数量要多得多；而且，通常全局问题的迭代次数在宏单元变体中要少得多。本文余下部分将在现代异构计算集群上运行的完整并行实现中研究这些方面。

6 计算效率：采用并行实现的数值测试

接下来，我们将通过并行实现进行数值测试所测得的计算时间，补充上一节报告的理论估计。为此，我们重新考察第4.1节中介绍的三维基准算例，其中内部层与（结构化）网格呈倾斜关系。几何形状和宏单元基网格如图4b和5b所示。我们分别取扩散系数和对流场为 $\kappa = 10^{-8}$ 和 $\boldsymbol{a} = (1, 1, 1)$，得到佩克莱数 $\mathrm{Pe} \gg 1$。为了准确捕捉内部层，我们再次采用基于梯度的误差估计器生成具有五个细化层级的自适应网格。图14展示了宏单元细化模式，以及表明内部层得到准确分辨的解图。



(a)第5级的宏单元细化及其解。 **(b)** 体积切片。

Fig图 14 三维基准测试的宏单元细化模式及对应解。

6.1 计算设置和预条件方法

我们的实现使用 Julia[1], 编写，包含数值积分、单元构造和全局组装例程。对于局部求解器，我们需要组装局部矩阵、计算 $\mathbf{A}_{(e)}$ 的局部 LU分解，并将局部值传递到全局解中。对于 $m = 1$ 和 $m = 2$，我们使用 LAPACK[2], 软件包提供的稠密线性代数；对于 $m = 4$和 $m = 8$，我们使用 UMFPACK[46] 库提供的稀疏线性代数。

对于全局求解器，我们使用开源库 PETSc[3]中的 GMRES迭代求解器。这里可以采用两种不同的求解方案。一方面，我们可以选择第 3.3 节详述的无矩阵求解方法。另一方面，我们可以选择基于矩阵的求解方法，该方法使用显式组装并存储的系数矩阵。在这两种方法中，我们都通过分块矩阵 $\mathbf{D}$的逆对方程 (12a) 进行左预条件。需要强调的是，矩阵 $\mathbf{D}^{-1}$ 从未被实际计算。该预条件子在无矩阵算法的第四步中应用，该步骤针对网格骨架中的每个面 f计算

$$\hat{\mathbf{w}}_{(\mathrm{f})} = \hat{\mathbf{u}}_{(\mathrm{f})} - \mathbf{D}_{(\mathrm{f})}^{-1} \left(\hat{\mathbf{v}}_{(\mathrm{f}1)} + \hat{\mathbf{v}}_{(\mathrm{f}2)} \right). \tag{17}$$

这里， f1 和 f2 表示面 f两侧相邻宏单元的面。由于所有块 $\mathbf{D}_{(\mathrm{f})}$ 均为正定且对称，我们就地计算并存储其 Cholesky分解。注意，右端项也以类似方式进行调整。为简化记号，下文将此变体称为 $\mathbf{D}^{-1}$ 预条件子。

我们注意到，对于基于矩阵的方法，我们测试了其他几种预条件变体。具体而言，我们考虑了代数加性 Schwarz 区域分解，并采用一级重叠，即 ASDD(1)[47], , 其中每个重叠子问题都通过零填充的不完全 LU 分解 ILU(0) 进行近似。根据我们的经验，对于基于矩阵的求解方法，在各种流动状态、精度阶数和处理器数量[24,48] 下，ASDD(1) 与 ILU(0) 的组合在通信成本和 GMRES 迭代次数之间取得了令人满意的平衡。然而，我们发现，与 $\mathbf{D}^{-1}$ 预条件子相比，这种方法并未显著提高效率，因此本文的所有计算均采用了后者。

[1]Julia 编程语言，https://julialang.org/ [2]线性代数软件包，https://netlib.org/lapack/ [3]可移植、可扩展科学计算工具包，https://petsc.org/

表4 表5和表6中所述全局问题的自由度（dof）数量。

	$p=2$		$p=3$		$p=4$		$p=5$	
	$m=1$	$m=2$	$m=1$	$m=2$	$m=1$	$m=2$	$m=1$	$m=2$
dof^{glob}	1,517,376	483,000	2,528,960	901,600	3,793,440	1,449,000	5,310,816	2,125,200

我们的实现已移植到达姆施塔特工业大学高性能计算中心提供的并行计算环境 Lichtenberg II（第一阶段），并使用 GCC（版本 9.2.0）、可移植硬件局部性库（版本 2.7.1）和 OpenMPI（版本 4.1.2）进行编译。我们的结果是在集群系统上获得的，我们使用了多个计算节点。每个计算节点配备两颗英特尔至强铂金 9242 处理器，每颗处理器拥有 48 个核心，基准时钟频率为 2.3 GHz。每个计算节点提供最多 384 GB 的主存。有关可用计算系统的更多详情，请参阅 Lichtenberg 网页[4]。

6.2 全局求解器的基于矩阵方法与无矩阵方法

我们首先评估全局求解器的效率，比较基于矩阵的方法和无矩阵方法，见[49,50,51,52]。在基于矩阵的方法中，必须在宏单元层级显式形成全局系统 $\left(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)\hat{\mathbf{u}}=\mathbf{f}$，在全局层级组装并存储，然后通过迭代求解器求解。宏单元混合间断伽辽金方法继承了标准混合间断伽辽金方法的优点。由于 $\hat{\mathbf{u}}$ 在单元面上定义且取单值，全局系统矩阵 $\left(\mathbf{D}-\mathbf{C}\mathbf{A}^{-1}\mathbf{B}\right)$ 比许多其他DG方法的全局系统矩阵更小[11,12,53]。此外，其结构紧凑，也就是说，只有共享同一单元的相邻面之间的自由度彼此相连[11]。

我们考虑对立方体进行离散化，其中包含 124,096 个单元，多项式次数为 $p=2$ 至 $p=5$，并在 2,848 个处理器上进行计算。表 4 汇报了所考虑的两种混合间断伽辽金变体中，全局问题的自由度数量。表 5 汇报了丢弃容差为 10e-2 时 GMRES 线性求解器所需的时间和迭代次数。我们比较了标准混合间断伽辽金方法与每个宏单元包含八个单元的宏单元混合间断伽辽金方法（$m=2$）。此外，对于每种混合间断伽辽金变体，我们还比较了基于矩阵（MB）方法和无矩阵（MF）方法；这两种变体均采用上述的 $\mathbf{D}^{-1}$ 预条件子。

我们发现，在两种方法中，随着 p 增加，全局求解器所需的计算时间也会增加；这符合预期，因为自由度数量增加了，且全局矩阵的结构发生了变化。我们观察到，当迭代次数较少时，对于两种混合间断伽辽金变体，无矩阵方法都比基于矩阵的方法高效得多。此外，基于矩阵的方法的计算时间增幅远大于无矩阵方法。我们还发现，宏单元混合间断伽辽金方法中的无矩阵方法略快，这是因为与标准混合间断伽辽金方法相比，其自由度数量减少，且计算节点之间的通信量降低。

我们进行了相同的数值实验，但将丢弃容差设为 10e-6，并在表 6 中报告结果。我们观察到，降低 GMRES 线性求解器的丢弃容差，也就是提高线性系统解的精度后，基于矩阵的方法变得更具竞争力。原因在于，较小的丢弃容差会显著增加所需的迭代次数。基于矩阵的方法需要先付出较大的矩阵构建成本，而无矩阵方法无需承担这项成本；但由于矩阵向量乘积次数更多，无矩阵方法的每次迭代成本更高。所需迭代次数越多，初始成本的重要性就越低。由于使用相同的预条件子，基于矩阵方法和无矩阵方法的迭代次数实际上相同。比较标准 HDG 和宏单元混合间断伽辽金变体可知，随着系统规模增大，后者在全局求解器的计算时间更短、迭代次数更少（因而条件性更好）方面的优势愈发明显，这是因为自由度减少，且计算节点之间的通信量降低。

另一个值得注意的重要方面是，与标准混合间断伽辽金方法相比，宏单元变体在中等多项式次数（如二次多项式和三次多项式）下的相对优势尤为明显。表 6 显示，对于$p=2$，无矩阵宏单元混合间断伽辽金方法的速度是无矩阵标准混合间断伽辽金方法的三倍以上；而对于 $p=5$，这一比例降至约两倍，不过在后一种情况下，标准混合间断伽辽金方法所需的迭代次数超过三倍

[4]https://www.hhlr.tu-darmstadt.de

表 5 全局求解器：使用**GMRES 求解器（丢弃容差 10^{-2}）**和 $\mathbf{D}^{-1}$ 预条件子的时间及迭代次数。我们比较了标准 HDG 与宏单元混合间断伽辽金，以及针对不同 $p(N^{Elm} = 124,096,\ Proc's = 2,848)$ 的基于矩阵 (MB) 与无矩阵 (MF) 方法。

	全局求解器时间 [秒]				迭代次数			
	标准 HDG		宏单元 HDG ($m = 2$)		标准 HDG		宏单元 HDG（$m = 2$）	
	MB	MF	MB	MF	MB	MF	MB	MF
$p = 2$	0.94	0.05	0.75	0.01	12	12	10	10
$p = 3$	1.26	0.10	1.32	0.03	11	11	9	9
$p = 4$	3.23	0.18	4.19	0.08	11	11	9	9
$p = 5$	4.31	0.22	13.00	0.14	10	10	9	9

表 6 全局求解器：时间和**GMRES 求解器（丢弃容差 10^{-6}）**以及 $\mathbf{D}^{-1}$ 预条件子的迭代次数。我们比较不同 p 下的标准 HDG 与宏单元混合间断伽辽金，以及基于矩阵（MB）与无矩阵（MF）方法，$(N^{Elm} = 124,096, Proc's = 2,848)$。

	全局求解器时间 [秒]				迭代次数			
	标准 HDG		宏单元混合间断伽辽金（$m = 2$）		标准 HDG		宏单元混合间断伽辽金（$m = 2$）	
	MB	MF	MB	MF	MB	MF	MB	MF
$p = 2$	1.95	3.04	1.09	0.92	1259	1259	939	939
$p = 3$	4.05	8.23	3.98	3.26	1735	1741	1304	1304
$p = 4$	11.74	15.26	9.75	8.98	1829	1830	1259	1258
$p = 5$	40.53	58.37	24.73	23.69	4717	4766	1417	1419

于全局求解器中的迭代次数。因此可以预期，对于采用中等多项式次数单元进行离散化的较大规模系统，宏单元混合间断伽辽金方法相较于标准混合间断伽辽金方法的优势将最为显著。基于这一观察，在本研究余下部分中，我们将重点研究二次单元。

6.3 总体系统规模：全局运算与局部运算的计算时间

下一步，我们比较标准混合间断伽辽金方法和宏单元混合间断伽辽金方法在网格细化下的运行时间性能，比较依据包括局部运算的计算时间（局部求解器加上矩阵自由全局求解器的第2步）、全局运算的时间（矩阵自由全局求解器的其余步骤），以及广义最小残量法迭代次数。根据上一小节的观察，宏单元混合间断伽辽金方法在中等多项式次数下效率较高，因此我们将多项式阶数固定为 $p = 2$，并使用固定数量的512个处理器。

表7报告了这样一组网格的自由度数量：根据图14所示的自适应细化结构，对初始网格进行全局细化后生成这些网格。两种混合间断伽辽金变体使用相同的网格，其中标准混合间断伽辽金方法采用完全不连续的单元（$m = 1$），而宏单元混合间断伽辽金方法则将八个四面体单元（$m = 2$）或64个四面体单元（$m = 4$）组合成一个 C^0 连续宏单元。我们观察到，随着 m增加，宏单元混合间断伽辽金变体显著减少了局部问题和全局问题中的自由度总数。特别是，它将全局系统的自由度数量减少了约一个数量级。

在表8中，我们可以观察到，对于给定的网格，缩减全局求解器的规模可显著减少计算时间。较小系统条件性的改善也是一个重要因素，这使得 GMRES 求解器所需的迭代次数显著减少。对于包含 992,768 个单元的最大网格，采用 $m = 4$ 的宏单元混合间断伽辽金方法进行全局运算所需的时间约为

表7 总体问题规模：我们比较标准 HDG（$m = 1$）与宏单元混合间断伽辽金（$m = 2$，4）在不同 N^{Elm}（均采用 $p = 2$）下的自由度（dof）。

	$m = 1$		$m = 2$		$m = 4$	
	dof^{local}	dof^{global}	dof^{local}	dof^{global}	自由度 $_{local}$	自由度 $_{global}$
$N^{Elm} = 45,504$	1,820,160	560,448	796,320	179,640	469,260	70,740
$N^{Elm} = 124,096$	4,963,840	1,517,376	2,171,680	483,000	1,279,740	187,740
$N^{Elm} = 364,032$	14,561,280	4,425,984	6,370,560	1,401,120	3,754,080	538,920
$N^{Elm} = 992,768$	39,710,720	12,026,112	17,373,440	3,793,440	10,237,920	1,449,000

表8 全局与局部运算的时间：我们比较局部求解器和矩阵自由全局求解器的局部部分（步骤 2）所需的时间，与全局求解器其余部分所需的时间（广义最小残量法丢弃容差 10^{-6}）。我们比较标准 HDG（$m = 1$）与宏单元混合间断伽辽金（$m = 2, 4$）（均采用 $p = 2$、$\mathbf{D^{-1}}$ 预条件子，$Proc's = 512$）。

	局部操作时间 [秒]			全局操作时间 [秒]			迭代次数		
	$m = 1$	$m = 2$	$m = 4$	$m = 1$	$m = 2$	$m = 4$	$m = 1$	$m = 2$	$m = 4$
$N^{Elm} = 45,504$	0.4	0.1	0.2	4.8	1.0	0.8	1970	1066	551
$N^{Elm} = 124,096$	2.4	0.5	1.0	23.4	2.6	2.2	3118	1231	1048
$N^{Elm} = 364,544$	11.1	2.2	3.9	101.6	10.0	5.7	5036	1673	1412
$N^{Elm} = 992,768$	46.2	26.0	24.3	433.4	120.5	36.2	8392	7526	3304

标准混合间断伽辽金方法的十分之一，这部分归因于迭代次数从 8,392 次减少到仅 3,304 次。此外，我们观察到，对于 $m = 4$，局部运算和全局运算在时间方面处于同一数量级。因此，我们得出结论：宏单元混合间断伽辽金方法能够灵活调整每个宏单元所需的计算负载，从而实现局部运算与全局运算之间的平衡。

6.4 宏单元尺寸：全局与局部操作的计算时间

接下来，我们重点研究宏单元混合间断伽辽金方法，考察不同宏单元尺寸的影响，以及该方法灵活平衡局部与全局操作的能力。为此，我们不再固定处理器数量，而是改为使用可变数量的处理器，同时将宏单元数量与处理器数量之比固定为 1（宏单元 / 处理器 $= 1$）。根据我们的观察，宏单元混合间断伽辽金方法在多项式次数适中时效率较高，因此我们再次将多项式阶数固定为 $p = 2$。表9报告了网格序列的自由度数量。这些网格是通过对图14所示的、具有自适应细化结构的初始网格进行全局细化生成的；在这些网格中，宏单元混合间断伽辽金方法将由八个四面体单元（$m = 2$）、64 个四面体单元（$m = 4$）和 512 个四面体单元（$m = 8$）组成的片区分别合并为一个 C^0 连续宏单元。我们再次观察到，随着 m 增大，自由度总数显著减少。全局问题的系统规模缩小且条件性改善，进而大幅减少了所需的 GMRES迭代次数。

表10报告了随着宏单元尺寸增大，局部运算（局部求解器加上无矩阵全局求解器的第2步）和全局运算（无矩阵全局求解器的其余步骤）所需的计算时间 m。我们观察到，通过调整 m，进而调整每个宏单元的局部计算负载，可以灵活调节局部运算与全局运算所需的时间。在我们的研究中，对于较粗的两个网格，宏单元尺寸为 $m = 4$ 时效果最佳；而对于最细的网格，$m = 8$ 则明显更优。这一观察表明，对于非常大型的系统，较大的宏单元更可取，因为它们能够平衡局部与全局运算，并使整个问题的计算时间最短。

表9 宏单元尺寸：对于逐渐增大的 m，全局问题的自由度数（dof）以及 GMRES 求解器的迭代次数。‘*’表示 GMRES 未能在给定的迭代次数内收敛。

	dof^{global}			迭代次数		
	$m=2$	$m=4$	$m=8$	$m=2$	$m=4$	$m=8$
$N^{Elm}=24,576$	97,920	38,880	18,360	999	648	297
$N^{Elm}=196,608$	760,320	293,760	132,192	2,256	1,101	764
$N^{Elm}=1,572,864$	5,990,400	2,280,960	998,784	*3,000	2,990	1,885

表10 全局运算与局部运算的时间：我们比较局部求解器和矩阵自由全局求解器的局部部分（步骤 2）所需的时间，与全局求解器其余部分所需的时间（GMRES 丢弃容差 10^{-6}）。我们采用具有不同 m 的宏单元 HDG（使用 $p=2$、$\mathbf{D^{-1}}$ 预条件子，**每个处理器上的宏单元数 = 1**）。‘*’表示 GMRES 在 3,000 次迭代内未能收敛。

	局部运算时间 [秒]			全局运算时间 [秒]		
	$m=2$	$m=4$	$m=8$	$m=2$	$m=4$	$m=8$
$N^{Elm}=24,576$	0.9	1.3	2.7	2.2	1.2	0.7
$N^{Elm}=196,608$	2.1	2.2	5.5	7.8	2.6	2.2
$N^{Elm}=1,572,864$	2.9	6.6	11.3	*28.5	21.9	14.8

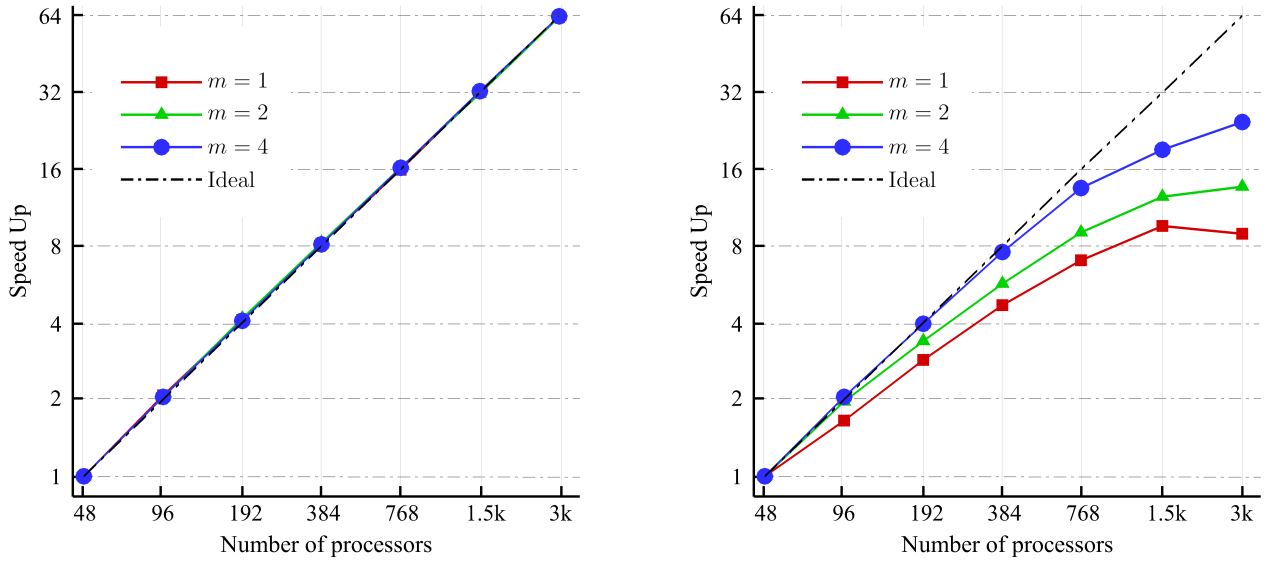
6.5 全局求解器和局部求解器的并行可扩展性

迄今报告的计算时间和迭代次数，能够估计宏单元混合间断伽辽金方法在宏单元结构、单元尺寸和多项式次数方面的表现，但并未直接涵盖评估求解器性能的另一项关键指标，即并行可扩展性[54]。现在，我们考察局部求解器和全局求解器的强可扩展性；前者采用基于LU分解的直接线性求解器，后者采用无矩阵方法，并使用GMRES 线性求解器和 $\mathbf{D^{-1}}$ 预条件子。我们使用以下数量的处理器开展可扩展性研究：48、96、192、384、768、1,536和3,072。我们使用相同的自适应四面体网格，其中包含637,888个单元。宏单元混合间断伽辽金方法将8个和64个单元分组，组成宏单元（$m=2,\ 4$）；标准 HDG 方法则使用标准的不连续单元（$m=1$）。对于 $m=1$、2、4，各自对应的自由度约为：局部未知量分别为2,600万、1,100万和700万，全局未知量分别为800万、200万和90万。关于局部/全局可扩展性，我们报告相对于基准计算时间的加速比；该基准计算时间是在48个处理器上获得的。

图15绘制了标准混合间断伽辽金方法（$m=1$）以及采用$m=2$ 和 $m=4$的宏单元混合间断伽辽金方法的强并行扩展性。我们从图15a中观察到，对于这两种混合间断伽辽金变体，局部求解器的计算时间在强扩展意义上都具有完美的可扩展性。然而，对于全局求解器，我们从图15b中看到，与标准混合间断伽辽金方法相比，宏单元混合间断伽辽金方法提高了强并行扩展性。标准混合间断伽辽金方法的并行效率在仅使用96个处理器时便降至80%以下，而采用 $m=4$ 的宏单元混合间断伽辽金方法在多达763个处理器时仍保持近乎完美的并行效率。一方面，这可归因于标准混合间断伽辽金方法的全局求解器具有较长的通信时间[13,43]。另一方面，宏单元混合间断伽辽金方法受益于其矩阵自由全局求解器中较高比例的局部运算（步骤2），以及局部矩阵和全局矩阵更大的带宽。

7 总结与结论

本文研究了一种混合间断伽辽金（HDG）方法的宏单元变体，该方法结合了连续有限元离散化与混合间断有限元离散化的要素。我们展示了若干重要的



(a)局部求解器的扩展性。 **(b)**全局求解器的扩展性。

图15 标准 HDG（$m = 1$) 与宏单元混合间断伽辽金（$m = 2, 4$)($N^{Elm} = 637$，888（$p = 2$）的局部求解器和全局求解器强扩展性。

实际优势。首先，由于在宏单元内使用连续单元，宏单元混合间断伽辽金方法显著减少了给定网格上的自由度数量，从而缓解了标准 HDG 方法中自由度激增的问题。其次，该方法保留了 HDG 方法独特的区域分解机制，将整体问题划分为每个宏单元对应的局部问题和一个全局问题。具体而言，它在定制宏单元离散及其对应的局部问题规模方面提供了更大的灵活性，使其能够适应可用计算系统及其并行架构。第三，该方法为自适应局部细化提供了直接的方法，因为宏单元网格能够自然地容纳悬挂节点，从而支持一致单纯形细分。第四，对所有宏单元（可能处于不同细化层级）采用相同的连续离散后，所有局部运算都可轻松并行且自动实现均衡，因此无需在数值方法之外采用负载均衡过程。从这个意义上说，宏单元混合间断伽辽金方法特别适合采用无矩阵求解方法，因为该方法在很大程度上由宏单元局部运算构成。

关于计算效率，我们通过理论估计表明，在假设两种变体采用相同网格，且全局求解器所需迭代次数相当的情况下，宏单元混合间断伽辽金方法的计算效率似乎低于标准混合间断伽辽金方法。然而，理论估计无法考虑两个重要参数：宏单元HDG变体所需的处理器间通信较少，并且能够减少全局求解器的迭代次数。

为纳入这些参数，我们通过在 Julia 中实现的并行程序，对宏单元混合间断伽辽金方法和标准混合间断伽辽金方法进行了比较。该程序已移植到现代异构计算系统上（达姆施塔特工业大学的 Lichtenberg II 第一阶段，在 2022 年 11 月 TOP500 榜单中排名第 187 位）。我们证明，与标准混合间断伽辽金方法相比，宏单元混合间断伽辽金方法中的全局求解器采用无矩阵方法时效率更高，这得益于自由度和计算节点间通信量的减少。我们还表明，不同于标准混合间断伽辽金方法，宏单元混合间断伽辽金方法对于二次多项式和三次多项式等中等多项式次数同样高效，因为无论多项式次数如何，都可以增加每个宏单元的局部计算负载。就包含最多一百万个高阶单元的实际网格的总体计算时间而言，我们表明，由于自由度减少，尤其是全局问题规模和求解所需迭代次数的减少，宏单元混合间断伽辽金方法能够平衡局部运算与全局运算，并且在本文所考察的情况下，其计算时间比标准混合间断伽辽金方法更短，最高可达一个数量级。我们还表明，这两种变体中的局部求解器都几乎实现了完美的可扩展性。对于

全局求解器而言，由于通信量减少以及局部运算与全局运算之间的比例更为有利，宏单元混合间断伽辽金方法在强可扩展性方面展现出明显优势。

这些性质在此针对对流-扩散模型得到验证，但它们能否推广到更具挑战性的问题，仍有待观察。未来，我们计划将宏单元混合间断伽辽金方法应用于通过纳维-斯托克斯方程建模的湍流直接数值模拟，以研究这一问题。

致谢

作者衷心感谢德国研究联合会（Deutsche Forschungsgemeinschaft）通过 DFG Emmy Noether Grant SCH 1249/2-1 提供的资金支持。作者还衷心感谢 NHR 中心 NHR4CES 在达姆施塔特工业大学的高性能计算机 Lichtenberg 为他们提供的计算时间。该项目由联邦教育与研究部以及黑森州政府资助。

参考文献

1. Arnold DN, Brezzi F, Cockburn B, Marini LD. 椭圆问题的间断伽辽金方法统一分析。*SIAM*数值分析杂志2002年；39(5)：1749–1779。 2. Bassi F, Rebay S. 用于数值求解可压缩Navier–Stokes方程的高阶精度间断有限元方法。《计算物理学杂志》1997年；131(2)：267–279。 3. Cockburn B. 计算流体力学中的间断伽辽金方法。《计算力学百科全书》第二版2018年：1–63。 4. Hesthaven JS, Warburton T. 节点间断伽辽金方法：算法、分析与应用。Springer Science & Business Media。2007年。 5. Peraire J, Persson PO. 用于椭圆问题的紧致间断伽辽金（CDG）方法。*SIAM*科学计算杂志2008年；30(4)：1806–1824。 6. Cockburn B, Gopalakrishnan J, Lazarov R. 用于二阶椭圆问题的间断伽辽金、混合和连续伽辽金方法的统一混合化。*SIAM*数值分析杂志2009年；47(2)：1319–1365。 7. Nguyen NC, Peraire J, Cockburn B. 用于线性对流–扩散方程的隐式高阶可混合间断伽辽金方法。《计算物理学杂志》2009年；228(9)：3232–3254。 8. Cockburn B, Guzmán J, Wang H. 用于二阶椭圆问题的超收敛间断伽辽金方法。《计算数学》2009年；78(265)：1–24。 9. Nguyen NC, Peraire J. 连续介质力学偏微分方程的可混合间断伽辽金方法。《计算物理学杂志》2012年；231(18)：5955–5988。 10. Nguyen NC, Peraire J, Cockburn B. 用于Stokes流的可混合间断伽辽金方法。《应用力学与工程中的计算方法》2010年；199(9-12)：582–597。 11. Peraire J, Nguyen N, Cockburn B. 用于可压缩Euler和Navier-Stokes方程的可混合间断伽辽金方法。载于：；2010年：363。 12. Cockburn B. 静态凝聚、混合化与HDG方法的设计。载于：Springer。2016年（第129–177页）。 13. Cockburn B, Dong B, Guzmán J, Restelli M, Sacco R. 用于稳态对流–扩散–反应问题的可混合间断伽辽金方法。*SIAM*科学计算杂志2009年；31(5)：3827–3846。 14. Nguyen N, Peraire J, Cockburn B. 用于非线性对流–扩散方程的隐式高阶可混合间断伽辽金方法。《计算物理学杂志》2009年；228(23)：8841–8855。doi：10.1016/j.jcp.2009.08.030

15. Nguyen NC, Peraire J, Cockburn B. 用于不可压缩Navier–Stokes方程的隐式高阶可混合间断伽辽金方法。《计算物理学杂志》2011年；230(4)：1147–1170。 16. Fernandez P, Nguyen N, Peraire J. 用于转捩湍流隐式大涡模拟的混合间断伽辽金方法。《计算物理学杂志》2017年；336：308-329。doi：10.1016/j.jcp.2017.02.015 17. Vila-Pérez J, Giacomini M, Sevilla R, Huerta A. 可压缩流的可混合间断伽辽金形式。计算工程方法档案2021年；28(2)：753–784。 18. Nguyen NC, Peraire J, Cockburn B. 用于声学和弹性动力学的高阶隐式可混合间断伽辽金方法。《计算物理学杂志》2011年；230(10)：3695–3718。 19. La Spina A, Kronbichler M, Giacomini M, Wall WA, Huerta A. 用于流固耦合问题的弱可压缩可混合间断伽辽金形式。《应用力学与工程中的计算方法》2020年；372：113392。 20. Foucart C, Mirabito C, Haley PJ, Lermusiaux PF. 面向非静水海洋过程的可混合间断伽辽金方法的分布式实现与验证。载于：电气电子工程师学会。；2018年：1–8。 21. Pazner W, Persson PO. 用于间断伽辽金流体模拟的阶段并行全隐式龙格–库塔求解器。《计算物理学杂志》2017年；335：700–717。 22. Fabien MS, Knepley MG, Mills RT, Rivière BM. 面向可混合间断伽辽金嵌套多重网格方法的众核并行计算。*SIAM*科学计算杂志2019年；41(2)：C73–C96。 23. Kronbichler M, Wall WA. 连续与间断伽辽金方法结合快速多重网格求解器的性能比较。*SIAM*科学计算杂志2018年；40(5)：A3423–A3448。 24. Roca X, Nguyen C, Peraire J. 用于可压缩流的混合间断伽辽金方法的可扩展并行化。载于：；2013年：2939。 25. Roca X, Nguyen NC, Peraire J. 面向可混合间断伽辽金方法的GPU加速稀疏矩阵向量乘积。载于：；2011年：687。 26. Hughes TJ. 有限元方法：线性静态与动态有限元分析。Courier Corporation。2012年。 27. Paipuri M, Tiago C, Fernández-Méndez S. 连续伽辽金方法与可混合间断伽辽金方法的耦合：共轭传热问题的应用。科学计算杂志2019年；78(1)：321–350。 28. Huerta A, Angeloski A, Roca X, Peraire J. 连续与间断伽辽金方法中高阶单元的效率。《国际工程数值方法杂志》2013年；96(9)：529–560。 29. Kirby RM, Sherwin SJ, Cockburn B. 选择CG还是HDG：一项比较研究。科学计算杂志2012年；51(1)：183–212。 30. Yakovlev S, Moxey D, Kirby RM, Sherwin SJ. 选择CG还是HDG：一项三维比较研究。科学计算杂志2016年；67(1)：192–220。 31. Arbogast T, Cowsar LC, Wheeler MF, Yotov I. 非匹配多块网格上的混合有限元方法。*SIAM*数值分析杂志2000年；37(4)：1295–1315。 32. Belgacem FB. 带拉格朗日乘子的Mortar有限元方法。《数值数学》1999年；84(2)：173–197。 33. Klawonn A, Widlund OB. 线性弹性的对偶原始FETI方法。《纯粹与应用数学通讯：库朗数学科学研究所发行的期刊》2006年；59(11)：1523–1572。 34. Bertsekas DP. 约束优化与拉格朗日乘子法。Academic Press。2014年。 35. Toselli A, Widlund O.区域分解方法：算法与理论。34. Springer Science & Business Media。2004年。

36. Toselli A. 标量对流–扩散问题的 FETI 区域分解方法。《应用力学与工程中的计算方法》2001；190(43-44)：5759–5776。37. Brooks A、TJR H. 针对对流主导流动的流线迎风/Petrov-Galerkin 方法。1980。38. Brooks AN、Hughes TJ. 针对对流主导流动的流线迎风/Petrov-Galerkin 公式，特别强调不可压缩 Navier-Stokes 方程。《应用力学与工程中的计算方法》1982；32(1-3)：199–259。39. Christie I、Griffiths DF、Mitchell AR、Zienkiewicz OC. 具有显著一阶导数的二阶微分方程的有限元方法。《国际工程数值方法杂志》1976；10(6)：1389–1396。40. Donea J、Huerta A.《流动问题的有限元方法》。John Wiley & Sons。2003。41. Heinrich J C、Huyakorn PS、Zienkiewicz OC、Mitchell A. 二维对流输运方程的“迎风”有限元格式。《国际工程数值方法杂志》1977；11(1)：131–143。42. Nguyen L、Stoter S、Baum T 等。体素有限单元法的相场边界条件：基于 CT 的骨结构的无表面应力分析。《生物医学工程数值方法国际期刊》2017；33(12)：e2880。43. Samii A、Michoski C、Dawson C. 用于各向异性非均质扩散的并行自适应杂交间断 Galerkin 方法。《应用力学与工程中的计算方法》2016；304：118–139。44. Zheng G.《在超大规模并行计算机上实现高性能：性能预测与负载均衡》。伊利诺伊大学厄巴纳-香槟分校。2005。45. Vymazal M、Moxey D、Cantwell C、Sherwin S、Kirby RM. 椭圆问题的 CG-HDG 方法：性能模型。《arXiv 预印本 arXiv:1811.11855》2018。46. Davis TA. 算法 832：UMFPACK V4. 3——一种非对称模式多前沿法。《ACM 数学软件汇刊》（TOMS）2004；30(2)：196–199。47. Balay S、Abhyankar S、Adams M 等。PETSc 用户手册。2019。48. Diosady LT.《高阶间断 Galerkin 离散的区域分解预条件子》。博士论文。麻省理工学院；2011。49. Franciolini M、Fidkowski KJ、Crivellini A. 隐式非定常可压缩流动模拟的高效间断 Galerkin 实现与预条件子。《计算机与流体》2020；203：104542。50. Kronbichler M、Kormann K. 并行基于单元的有限元算子应用的通用接口。《计算机与流体》2012；63：135–147。51. Kronbichler M、Kormann K. 间断 Galerkin 有限元算子的快速无矩阵求值。《ACM 数学软件汇刊》（TOMS）2019；45(3)：1–40。52. Kronbichler M、Sashko D、Munch P. 增强高阶无矩阵有限元实现中共轭梯度法的数据局部性。《国际高性能计算应用杂志》2022：10943420221107880。53. Nguyen NC、Peraire J、Cockburn B. 一类用于计算流体力学的嵌入式间断 Galerkin 方法。《计算物理学杂志》2015；302：674–692。54. Fischer P、Min M、Rathnayake T 等。高性能偏微分方程求解器的可扩展性。《国际高性能计算应用杂志》2020；34(5)：562–586。