arXiv:2303.01163v1 [math.NA] 2 Mar 2023

一种用于求解偏微分方程的域分解策略

Ken Trotti 信息学学院
瑞士意大利大学
ken.trotti@usi.ch

2023年3月3日

摘要

在本研究中，我们提出了一种用于求解偏微分方程的新型域分解策略。针对 d 维对流-扩散方程的均匀离散化，我们的方案采用一种两层算法，将在高度各向异性子网格上对方程进行离散化所得到的解进行合并，以计算细网格解的初始近似。随后，该算法利用残差结构对初始猜测进行迭代细化。在各向异性子网格上执行代价高昂的计算，使我们能够将问题的维数降低一维；而合并过程涉及在互不相交的定义域上计算解，因此支持并行实现。

关键词 偏微分方程 · 对流-扩散方程 · Richardson外推 · 域分解 ·时间并行积分

1 引言

如今的超级计算机是拥有数千乃至数百万个核心的高性能计算机，旨在处理极其庞大而复杂的计算任务。因此，并行化——即将任务拆分为可在不同核心上同时运行的较小部分的能力——是充分发挥超级计算机性能的重要因素。要有效利用这一特性，必须相应地设计算法。对于高维偏微分方程，已有多种标准方法，包括深度学习技术、 [6, 7, 11], 蒙特卡洛方法 [20, 19], 稀疏网格 [1], 以及其他利用定义域性质、对称性和正则性的方法，参见例如 [12]。

稀疏网格（SG）方法首先构造一个具有 N^d 个网格点的均匀网格，然后按照特定规则移除某些点，从而得到一个具有 $O\left(N \log^{d-1} N\right)$ 个网格点的稀疏网格。虽然这种方法可以降低内存需求，但与使用均匀网格相比，也会导致近似误差增大。具体而言，当采用二阶精度格式时，所得近似在 L^2 范数下的误差为$O\left(N^{-2}\log(N)^{d-1}\right)$，大于使用均匀网格时达到的$O\left(N^{-2}\right)$ 误差。

我们的目标是开发一种采用比均匀网格更粗的网格、但能避免 SG 特有的误差增大的方法。因此，我们旨在开发一种算法，在保持高精度和潜在较低内存需求的同时，实现高效的并行计算。为实现这一目标，我们提出了各向异性子网格域分解方法（ASDSM），这是一种用于并行求解 d 维偏微分方程的新型两级域分解方法。ASDSM 将细均匀网格划分为多个强各向异性子网格。随后，将从每个子网格获得的解合并，形成均匀网格上离散化偏微分方程的近似解。该过程通过利用残差的结构重复进行，以提高解的精度。该方法能够有效地将问题的维数降低一个量级，从而高效求解规模更大、复杂性更高的问题。此外，如第 3.3 节所述，智能内存处理策略与使用原始均匀网格相比，还可进一步降低内存需求。

在这项初步研究中，我们聚焦于广为人知的对流-扩散方程。该方程是一种基础数学模型，广泛应用于流体力学、热传递和污染物输运等领域 [3, 16, 13]。

考虑以下　　以下带有狄利克雷边界条件（边界条件）的 d 维对流-扩散方程

$$\begin{cases} -\sum_{i=1}^{d} \alpha_i \frac{\partial u}{\partial x_i^2}(\mathbf{x}) + \beta_i \frac{\partial u}{\partial x_i}(\mathbf{x}) = s(\mathbf{x}), & \mathbf{x} \in \Omega \subset \mathbb{R}^d, \\ u(\mathbf{x}) = g(\mathbf{x}), & \mathbf{x} \in \partial\Omega, \end{cases} \tag{1}$$

及其时间依赖型对应形式

$$\begin{cases} \frac{\partial u}{\partial t}(\mathbf{x},t) - \sum_{i=1}^{d} \alpha_i \frac{\partial u}{\partial x_i^2}(\mathbf{x}) + \beta_i \frac{\partial u}{\partial x_i}(\mathbf{x}) = s(\mathbf{x},t), & (\mathbf{x},t) \in \Omega \times [0,1] \subset \mathbb{R}^{d+1}, \\ u(\mathbf{x}) = g(\mathbf{x},t), & (\mathbf{x},t) \in \partial\Omega \times [0,1), \end{cases} \tag{2}$$

其中， s 为源项， g 为边界条件， $\alpha > 0$ 为扩散系数， β 为速度场，Ω为空间区域。

在以下各节中，我们将详细介绍 ASDSM。具体而言，第 2 节介绍解释 ASDSM 所需的基本工具，并详细构造相关网格和投影算子，同时简要概述偏微分方程的离散化过程。第 3 节通过图示逐步讲解 ASDSM 算法，并就如何高效实现该算法提出一些注记。第 4 节给出数值结果，展示我们的方法在每次迭代中的残差降低效果。最后，第 5 节总结全文并概述未来工作的潜在方向。我们还将讨论该方法的意义，以及如何将其应用于本领域的其他问题。

2 ASDSM算法简介

为简便起见，本节将考虑正方形区域 $\Omega = [0,1] \times [0,1]$ 上方程 (1) 的二维情形。

在介绍 ASDSM算法之前，我们先介绍必要的网格并确定符号约定。随后，我们使用有限差分法对偏微分方程进行离散化，并介绍用于将未知量投影到不同网格上的投影算子。这些工具构成了 ASDSM算法的基础，对于理解该方法的细节至关重要。

2.1 网格

令 $N_c^{(x)}, N_{(x)f}, N_{(y)c}, N_{(y)f} \in \mathbb{N}$，使得

$$\begin{aligned} N_f^{(x)} + 1 &= n_x(N_c^{(x)} + 1), \\ N_f^{(y)} + 1 &= n_y(N_c^{(y)} + 1), \end{aligned} \tag{3}$$

对于某些 $n_x, n_y \in \mathbb{N}$，考虑步长

$$\begin{aligned} h_x &= \tfrac{1}{N_f^{(x)}+1}, & h_y &= \tfrac{1}{N_f^{(y)}+1}, \\ H_x &= \tfrac{1}{N_c^{(x)}+1} = n_x h_x, & H_y &= \tfrac{1}{N_c^{(y)}+1} = n_y h_y, \end{aligned} \tag{4}$$

以及以下均匀网格

$$\begin{aligned} \Omega^{h_x,h_y} &= \{(x_i,y_j) \,|\, x_i = ih_x,\ y_j = jh_y,\ i = 1,...,N_f^{(x)},\ j = 1,...,N_f^{(y)}\}; \\ \Omega^{h_x,H_y} &= \{(x_i,y_j) \,|\, x_i = ih_x,\ y_j = jH_y,\ i = 1,...,N_f^{(x)},\ j = 1,...,N_c^{(y)}\}; \\ \Omega^{H_x,h_y} &= \{(x_i,y_j) \,|\, x_i = iH_x,\ y_j = jh_y,\ i = 1,...,N_c^{(x)},\ j = 1,...,N_f^{(y)}\}; \\ \Omega^{H_x,H_y} &= \{(x_i,y_j) \,|\, x_i = iH_x,\ y_j = jH_y,\ i = 1,...,N_c^{(x)},\ j = 1,...,N_c^{(y)}\}; \\ \Omega_{i,j}^{h_x,h_y} &= \left\{ (x_k,y_l) \;\middle|\; \begin{array}{lll} x_k = iH_x + kh_x, & k = 1,...,n_x-1, & i = 0,...,N_c^{(x)} \\ y_l = jH_y + lh_y, & l = 1,...,n_y-1, & j = 0,...,N_c^{(y)} \end{array} \right\}. \end{aligned} \tag{5}$$

我们将方程 (5) 中的网格分别称为细网格、沿 x 和 y 轴密集的各向异性网格、粗网格和“孔洞”。此外，$\Omega^{h_x,H_y} \cup \Omega^{H_x,h_y}$ 的并集称为骨架或粗结构。此外，我们用 $\Omega_{holes}^{h_x\,h_y}$，表示所有孔洞的并集，即 $\Omega_{holes}^{h_x\,h_y} := \cup_{i=1,\,j=0}^{N_c^{(x)},\,N_c^{(y)}} \Omega_{i,\,j}^{h_x\,h_y}$。

以下命题是方程(3)中条件的推论，展示了(5)中网格之间的关系。

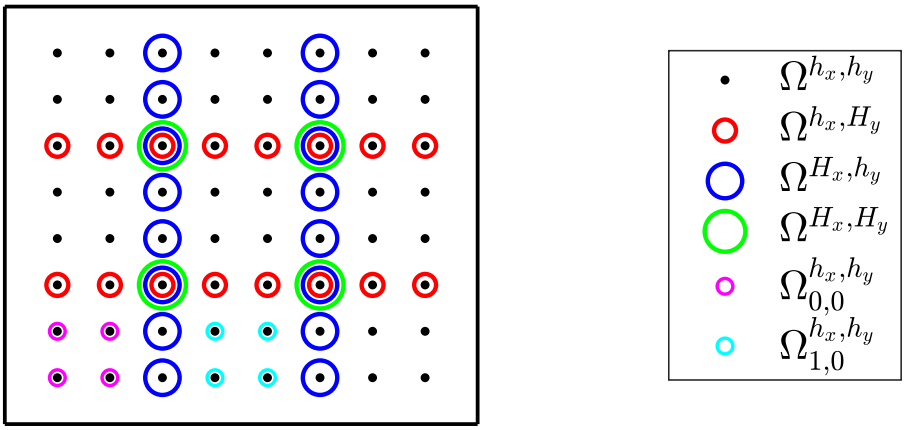


图1：网格的图示。

命题2.1. 考虑方程(5)中的网格，则以下关系成立

a) $\Omega^{h_x,H_y} \subset \Omega^{h_x,h_y}$, $\quad \Omega^{H_x,h_y} \subset \Omega^{h_x,h_y}$;

b) $\Omega^{H_x,H_y} \subset \Omega^{h_x,h_y}$, $\quad \Omega^{H_x,H_y} \subset \Omega^{H_x,h_y}$, $\quad \Omega^{H_x,H_y} \subset \Omega^{h_x,H_y}$;

c) $\Omega^{h_x,h_y}_{i,j} \cap \Omega^{h_x,h_y}_{k,l} = \emptyset \quad if \quad (i,j) \neq (k,l)$;

d) $\Omega^{h_x,h_y}_{holes} = \Omega^{h_x,h_y} \setminus \left(\Omega^{h_x,H_y} \cup \Omega^{H_x,h_y}\right)$.

在命题2.1中，条目(a)表明两个各向异性网格是细网格的子网格，条目(b)表明粗网格是细网格和各向异性网格的子网格。此外，由条目(a)可知，网格 $\Omega^{h_x,H_y} \cup \Omega^{H_x,h_y}$ 是细网格的非均匀子网格；在此基础上加入 Ω^{h_x,h_y}_{holes}，即可得到细网格。

事实上，与条目(d)等价地，我们有

$$\Omega^{h_x,h_y} = \left(\bigcup_{i,j=0}^{N_c^{(x)},N_c^{(y)}} \Omega^{h_x,h_y}_{i,j} \right) \cup \Omega^{h_x,H_y} \cup \Omega^{H_x,h_y}.$$

设 $N_c^{(x)} = N_c^{(y)} = 2$, $N_f^{(x)} = N_f^{(y)} = 8$。那么，图1展示了方程(5)中网格的图示。

2.2 投影算子

ASDSM算法旨在通过在较粗网格上求解方程来计算离散细网格解。为此，我们需要引入投影算子，以便将离散函数从一个网格传递到另一个网格。这些投影算子对于在本方法所使用的不同网格之间准确传递信息至关重要。

我们首先在方程（5）中引入定义在细网格 Ω^{h_x,h_y}上的所有离散函数的集合，记为$V^{h_x,h_y} = \{v \mid v \in \mathbb{R}^{\#\Omega^{h_x,h_y}}\}$。设 $v \in V^{h_x,h_y}$ 为一个有序向量

$$v = [v_{1,1}, v_{2,1}, ..., v_{N_f^{(x)},1}, v_{1,2}, ..., v_{N_f^{(x)},2}, ..., v_{N_f^{(x)},N_f^{(y)}}]^{\mathrm{T}},$$

其中，$v_{i,j}$是离散函数 v在网格点（x_i，y_j） $\in \Omega^{h_x,h_y}$处的值。我们还定义集合V^{h_x,H_y}、V^{H_x,h_y}、V^{H_x,H_y}、$V^{h_x,h_y}_{i,j}$，其元素的排列顺序相同。

注2.2. 显然，对于 m $v \in V^{h_x,h_y}$的每个位置，都存在唯一的索引 $i \in \{1, ..., N_f^{(x)}\}$和 $j \in \{1, ..., N_f^{(y)}\}$，使得 $v_m = v_{i,j}$且 $(i-1)N_f^y + j = m$。

定义 2.3. 下面定义以下投影算子。

1) P_{ff}^{fc}: $V^{h_x,h_y} \to V^{h_x,H_y}$，作为将向量 $v \in V^{h_x,h_y}$映射为向量 $w \in V^{h_x,H_y}$ 的投影算子 $w = P_{ff}^{fc} v$，仅保留满足 $(x_i, y_j) \in \Omega^{h_x,H_y}$ 的分量 $v_{i,j}$；根据注2.2，形式上，

$$\left(P_{ff}^{fc}\right)_{m,n} = \begin{cases} 1 & \text{if } \Omega^{h_x,h_y} \ni (x_{i(m)}, y_{j(m)}) = (x_{i(n)}, y_{j(n)}) \in \Omega^{h_x,H_y}, \\ 0 & \text{otherwise.} \end{cases}$$

2) P_{ff}^{cf}: $V^{h_x,h_y} \to V^{H_x,h_y}$，作为将向量 $v \in V^{h_x,h_y}$映射为向量 $w \in V^{H_x,h_y}$ 的投影算子 $w = P_{ff}^{cf} v$，仅保留满足 $(x_i, y_j) \in \Omega^{H_x,h_y}$ 的分量 $v_{i,j}$；

3) P_{ff}^{cc}: $V^{h_x,h_y} \to V^{H_x,H_y}$，作为将向量 $v \in V^{h_x,h_y}$映射为向量 $w \in V^{H_x,H_y}$ 的投影算子 $w = P_{ff}^{cc} v$，仅保留满足 $(x_i, y_j) \in \Omega^{H_x,H_y}$ 的分量 $v_{i,j}$；

4) P_{ff}^{holes}: $V^{h_x,h_y} \to V^{h_x,{}^{h_y}_{holes}}$，作为将向量 $v \in V^{h_x,h_y}$ 映射为向量 $w \in V^{h_x,{}^{h_y}_{holes}}$ 的投影算子，$w = P_{ff}^{holes} v$，只保留满足（x_i, y_j） $\in \Omega_{holes}^{h_x\ h_y}$, 的分量 $v_{i,j}$；

由于投影算子的转置会交换定义域与陪域，我们注意到：

1) $\left(P_{ff}^{fc}\right)^{\mathrm{T}} = P_{fc}^{ff}$;

2) $\left(P_{ff}^{cf}\right)^{\mathrm{T}} = P_{cf}^{ff}$;

3) $\left(P_{ff}^{cc}\right)^{\mathrm{T}} = P_{cc}^{ff}$;

4) $\left(P_{ff}^{holes}\right)^{\mathrm{T}} = P_{holes}^{ff}$.

由此，我们可以得到命题 2.4 中列出的性质。

命题 2.4。设 I 为适当大小的单位矩阵。那么，上述定义的投影算子具有以下性质：

1) $P_{ff}^{fc}\left(P_{ff}^{fc}\right)^{\mathrm{T}} = P_{ff}^{fc} P_{fc}^{ff} = I;$

2) $P_{ff}^{cf}\left(P_{ff}^{cf}\right)^{\mathrm{T}} = P_{ff}^{cf} P_{cf}^{ff} = I;$

3) $P_{ff}^{cc}\left(P_{ff}^{cc}\right)^{\mathrm{T}} = P_{ff}^{cc} P_{cc}^{ff} = I;$

4) $P_{ff}^{holes}\left(P_{ff}^{holes}\right)^{\mathrm{T}} = P_{ff}^{holes} P_{holes}^{ff} = I;$

*5) Every projector has norm-1 and infinity n*范数等于 1。

2.3 离散化

对式 (5) 中均匀网格上的偏微分方程进行离散化时，采用二阶精度中心有限差分法，从而得到以下线性系统。

$$\begin{array}{ll} A_{ff}u_{ff} = b_{ff}, & u_{ff} \in \mathbb{R}^{N_f^{(x)} N_f^{(x)}}; \\ A_{fc}u_{fc} = b_{fc}, & u_{fc} \in \mathbb{R}^{N_f^{(x)} N_c^{(x)}}; \\ A_{cf}u_{cf} = b_{cf}, & u_{cf} \in \mathbb{R}^{N_c^{(x)} N_f^{(x)}}; \\ A_{cc}u_{cc} = b_{cc}, & u_{cc} \in \mathbb{R}^{N_c^{(x)} N_c^{(x)}}; \\ A_{(ij)}u_{(ij)} = b_{(ij)}, & u_{(ij)} \in \mathbb{R}^{(n_x-1)(n_y-1)},\ i = 1, ..., n_x - 1,\ j = 1, ..., n_y - 1. \end{array} \tag{6}$$

具体而言，

$$A_{ff} = \alpha \left(I_{N_f^{(y)}} \otimes \left(\frac{1}{h_x^2} L_{N_f^{(x)}} \right) + \left(\frac{1}{h_y^2} L_{N_f^{(y)}} \right) \otimes I_{N_f^{(x)}} \right) + \\ + \beta \left(I_{N_f^{(y)}} \otimes \left(\frac{1}{2h_x} D_{N_f^{(x)}} \right) + \left(\frac{1}{2h_y} D_{N_f^{(y)}} \right) \otimes I_{N_f^{(x)}} \right),$$

$$b_{ff} = [s_{1,1}, s_{2,1}, ..., s_{N_f^{(x)},1}, s_{1,2}, ..., s_{N_f^{(x)},N_f^{(y)}}]^{\mathrm{T}},$$

其中 $\otimes$ 是张量积，$s_{i,\ j} = s(x_i, y_j)$、$(x_i, y_j) \in \Omega^{h_x,\ h_y}$ 和

$$D_N = \begin{bmatrix} 0 & 1 & & & \\ -1 & 0 & 1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 0 & 1 \\ & & & -1 & 0 \end{bmatrix}_{N \times N}, \qquad L_N = \begin{bmatrix} 2 & -1 & & & \\ -1 & 2 & -1 & & \\ & \ddots & \ddots & \ddots & \\ & & -1 & 2 & -1 \\ & & & -1 & 2 \end{bmatrix}_{N \times N}.$$

方程 (6) 中的其他矩阵和已知向量定义方式类似。

注2.5。由于 $\Omega_i^{h_x\ h_y}{}_{,\ ,\ j} \subset \Omega^{h_x,\ h_y}$，因此，如果采用相同的离散化方案，系数矩阵 $A_{(ij)}$ 是矩阵 A_{ff} 的一个主对角子块。

3 二维自适应稀疏域分解平滑算法

本节介绍 ASDSM算法，并通过图示提供详细的分步说明。我们考虑方程 (1) 中给出的二维对流扩散方程，其中 $\alpha = 1, \beta = 0$，即泊松问题； $g(x, y)$ 和 $s(x, y)$ 根据精确解 $u(x, y) = \sin(x\pi) \sin(2y\pi)$ 计算。为清晰起见，我们使用非常粗的网格对该方程进行离散化，其中 $N_f^{(x)} = N_{(y)f} = 24$ 和 $N_{(x)c} = N_{(y)c} = 4$。

需要注意的是，以下算法的输入仅包含随每次迭代而变化的变量。为提高可读性，我们省略了算法迭代过程中保持不变的其他输入。此外，有时步骤 k 可以拆分为两个互不重叠、可并行执行的部分，即子步骤 $k.1$) 和 $k.2$)。

3.1 初始猜测的计算

我们提出的方法的核心是一种特殊的初始猜测计算方式。该方法如算法1所示，包括计算方程 (6) 中各向异性线性系统和粗网格线性方程组的解 u_{fc}、u_{cf}、u_{cc}，将这三个解合并，从而构成细网格解 u_{ff} 的近似粗结构（即骨架），最后填充空区域。

算法1 初始猜测构造器

函数 INITIALGUESS($b_{ff}, b_{fc}, b_{cf}, b_{cc}$)
- 1.1) $u_{fc} = (A_{fc})^{-1} b_{fc}$ ▷ 在 Ω 上求解方程$^{h_x,\ H_y}$
- 1.2) $u_{cf} = (A_{cf})^{-1} b_{cf}$ ▷ 在 Ω 上求解方程$^{H_x,\ h_y}$

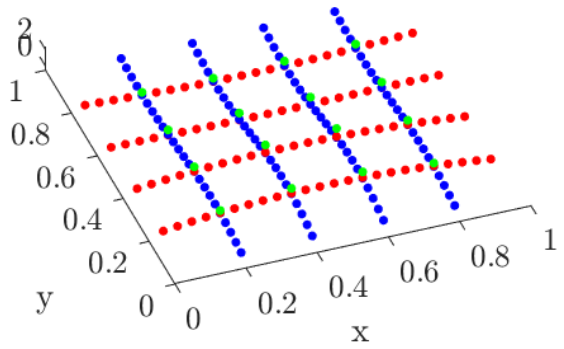
如果输入中有 $*b_{cc}$，则
- 1.3) $u_{cc} = (A_{cc})^{-1} b_{cc}$ ▷ 在 Ω 上求解方程$^{H_x,\ h_y}$

else
- 2.1) $u_{fc} = \frac{u_{fc}}{\|u_{fc}\|}$ ▷ 归一化向量
- 2.2) $u_{cf} = \frac{u_{cf}}{\|u_{cf}\|}$

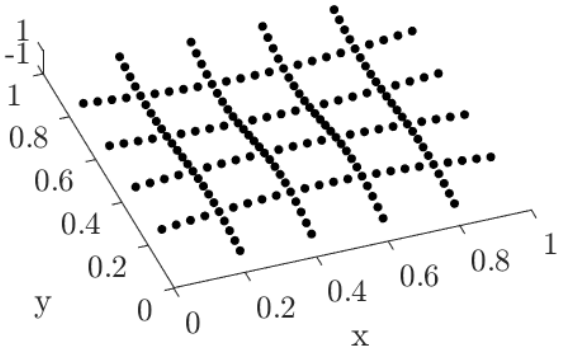
结束条件判断
- 3) $\tilde{u}_{ff} =$ 骨架 (u_{fc}, u_{cf}, u_{cc}) ▷ 构建初始猜测的骨架
- 4) $\tilde{u}_{ff} =$ 填充器 $(\tilde{u}_{ff}, b_{ff})$ ▷ 填充孔洞 $\Omega^{h_x,h}_{yi,j}$
- 返回 $\tilde{u}_{ff}$

函数结束

具体而言，在步骤 1) 中，可以通过专用求解器高效地计算解 u_{fc}、u_{cf}、u_{cc}。例如，循环预条件子、不完全 LU 预条件子和多重网格方法已被证明可作为有效的



(a) u_{fc}、 u_{cf}、 u_{cc} 分别以红色、蓝色和绿色显示，位于图1中的网格上。



(b) 校正后，网格 $\Omega^{h_x,H_y} \cup \Omega^{H_x,h_y}$ 上的曲面骨架。

图2：合并过程前（a）和后（b）的粗结构。

结构化各向异性 [10, 18, 8] 和各向同性 [9, 2, 5] 线性系统的预条件器/求解器。请注意，在计算初始猜测时，已知项 b_{cc} 已作为输入提供，因此满足"if"条件。未提供 u_{cc} 的情况会导致跳过步骤 1.3)，并在步骤 2) 中对两个解进行归一化，稍后将对此进行处理。在分别以图2a 中的红色、蓝色和绿色显示的 u_{fc}、 u_{cf}、 u_{cc} 计算完成后，我们在步骤 3) 中通过骨架构建算法 2 将它们合并为一个唯一的未知量 $\tilde{u}_{ff}$。

算法2 骨架构建器

函数 SKELETON(u_{fc}, u_{cf}, u_{cc})

1.1) $\tilde{u}_{cc}^{(1)} = P_{fc}^{cc} u_{fc}$

1.2) $\tilde{u}_{cc}^{(2)} = P_{cf}^{cc} u_{cf}$ ▷ 将各向异性解投影到粗网格上

如果 *u_{cc} 在输入中* 则 ▷ 可以使用 Richardson外推（理查森外推）

2) $\hat{u}_{cc}$ =理查森外推$\left(\tilde{u}_{cc}^{(1)}, \tilde{u}_{cc}^{(2)}, u_{cc}\right)$ ▷ 外推后的粗解

else

3) $\hat{u}_{cc} = \tilde{u}_{cc}^{(1)} \quad | \quad \hat{u}_{cc} = \tilde{u}_{cc}^{(2)}$ ▷ 选择粗解

结束条件判断

4.1) $e_{cc}^{(1)} = \hat{u}_{cc} - \tilde{u}_{cc}^{(1)}$

4.2) $e_{cc}^{(2)} = \hat{u}_{cc} - \tilde{u}_{cc}^{(2)}$ ▷ 计算相对于 $\hat{u}_{cc}$ 的误差

5.1) $\tilde{u}_{fc} = u_{fc}$+样条($e_{cc}^{(1)}, \Omega^{h_x,\ H_y}$)

5.2) $\tilde{u}_{cf} = u_{cf}$+样条($e_{cc}^{(2)}, \Omega^{H_x,\ h_y}$) ▷ 修正这两个解

6) $\tilde{u}_{ff} = P_{fc}^{ff} \tilde{u}_{fc} + P_{cf}^{ff} \tilde{u}_{cf} - P_{cc}^{ff} P_{cf}^{cc} \tilde{u}_{cf}$

返回 $\tilde{u}_{ff}$

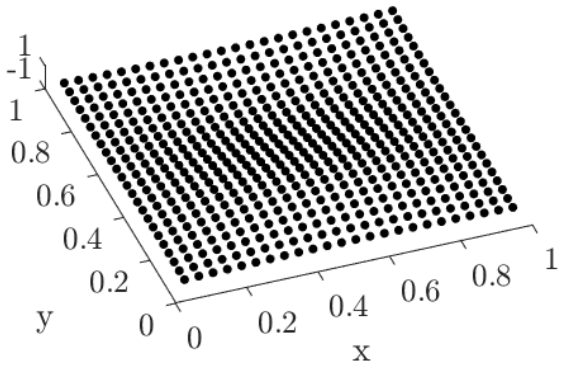
结束函数

在图2a中，我们注意到，由于所涉及的网格不同，三个解在交叉点处并未完全重合。因此，算法2利用Richardson外推计算精确的交叉点，然后调整u_{fc}和 u_{cf}，并将它们合并。

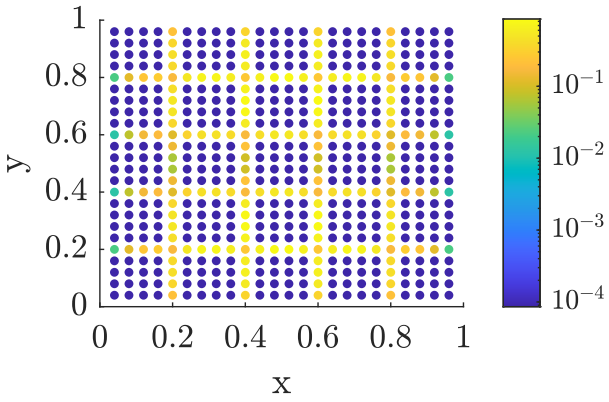
具体而言，在步骤1)中，我们通过投影算子 P_{fc}^{cc}、 P_{cf}^{cc}将两个各向异性解 u_{fc}、 u_{cf} 投影到粗各向同性网格$\Omega^{H_x,\ H_y}$ 上。然后，由于输入中包含 u_{cc}，在步骤3)中我们使用Richardson外推 [14]，在$\Omega^{H_x,\ H_y}$上生成一个更精确的解$\hat{u}_{cc}$。

注记3.1。为了进一步提高$\hat{u}_{cc}$的精度，如果满足光滑性要求，可以计算更多解，其计算成本低于 u_{fc}、 u_{cf}，例如，在各向异性网格$\Omega^{kh_x,\ H_y}$或$\Omega^{H_x,\ kh_y}$上计算，适用于某个 $k \in \mathbb{N}$。信息越多，外推解的精度越高。

现在已经计算出精确的交叉点，我们需要调整两个各向异性解 u_{fc}, u_{cf}，使其变为 $\tilde{u}_{fc}, \tilde{u}_{cf}$，从而满足 $P_{fc}^{cc}\tilde{u}_{fc} = P_{cf}^{cc}\tilde{u}_{cf} = \hat{u}_{cc}$，也就是说，它们在粗各向同性网格上的投影与外推交叉点 $\hat{u}_{cc}$重合。为此，我们在步骤 4) 中计算 $P_{fc}^{cc}u_{fc}, P_{cf}^{cc}u_{cf}$与 $\hat{u}_{cc}$之间的距离。然后，在



(a) 骨架填充过程结束后得到的初始猜测。



(b) 初始猜测的重塑后的残差。

图3：初始猜测 (a) 及其对应的残差 (b)。

步骤 5) 中，我们将该距离以零边界条件插值到网格 $\Omega^{h_x,\ H_y}$、$\Omega^{H_x,\ h_y}$上，并分别将其加到 u_{fc}, u_{cf}上。通过使用高阶样条插值，这一过程得到具有上述性质的光滑未知量 $\tilde{u}_{fc}, \tilde{u}_{cf}$。此时，在步骤 6) 中，我们将校正后的各向异性未知量投影到细网格 $\Omega^{h_x,\ h_y}$上，将它们相加，最后移除重复计数的网格点。新定义的未知量 $\tilde{u}_{ff}$称为初始猜测的骨架或粗结构。
如图2b所示，骨架曲面在所谓的空区域或孔洞 $\Omega_i^{h_x\ h_y}$，，$_j$ （白色区域）上为零，而在粗结构 $\Omega^{h_x,\ H_y} \cup \Omega^{H_x,\ h_y}$ （黑色区域）上是平滑的。这一特性使得孔洞能够被高效填充，如算法3中的填充算法示意图所示。

算法3 填充器

函数 F填充器($\tilde{u}_{ff}, A_{ff}, b_{ff}$)
1) $c = -P_{ff}^{holes} A_{ff} \tilde{u}_{ff}$ ▷ Use $\tilde{u}_{ff}$ as BCs
2) $v = \left(P_{ff}^{holes} A_{ff} P_{holes}^{ff}\right)^{-1} \left(P_{ff}^{holes} b_{ff} + c\right)$
3) $\tilde{u}_{ff} = \tilde{u}_{ff} + P_{holes}^{ff} v$
返回 $\tilde{u}_{ff}$
结束函数

填充过程包括将骨架 $\tilde{u}_{ff}$用作新互不相交的离散偏微分方程的边界条件，并行求解这些方程，最后更新细网格未知量。

具体而言，由于对孔洞进行了零填充，步骤 1) 中的乘积 $A_{ff}\tilde{u}_{ff}$ 将边界（骨架）上的信息投影到孔洞内部。然后，投影算子 P_{ff}^{holes} 通过仅保留空区域，从 c 中移除骨架；这些空区域现在包含了边界信息。因此，向量 c 包含了孔洞上线性系统的边界条件。
在步骤 2) 中，我们将矩阵 A_{ff} 和已知项 b_{ff} 投影到不含骨架的细网格上。新得到的线性系统的解，是各个孔洞内部解的串接。
请注意，矩阵 $P_{ff}^{holes} A_{ff} P_{holes}^{ff}$ 具有块结构。根据注2.5，由于使用了相同的离散格式， $P_{ff}^{holes} A_{ff} P_{holes}^{ff}$ 的每个块都对应于某些 i、j 下的 $A_{(ij)}$。因此，可以轻松地并行执行投影矩阵的求逆。

最后，在步骤3)中，我们利用新信息更新骨架。如图3a所示，这一过程会产生连续未知量（除非涉及时间变量）。

在给出 ASDSM 算法之前，我们总结初始猜测算法1的三个性质。
命题3.2. 设 $u_{ff} = (A_{ff})^{-1} b_{ff}$为方程(6)中细线性系统的数值解， $u^{(0)}$ 为通过算法1计算得到的初始猜测，其中 r 是重塑为离散曲面的残差向量。那么

1) 未知量 wn $u^{(0)}$ 与 u_{ff} 重合，当且仅当 $u^{(0)}$ 的骨架与 u_{ff}的骨架重合 ;

2) r 在孔洞上为零，在其骨架上非零，除非 $u^{(0)} = u_{ff}$；

3） r 的骨架在粗网格点 $\Omega^{H_x,\ H_y}$附近可能不光滑，即两个各向异性网格的交叉点处。

证明。陈述1）如果 $u^{(0)}$ 与 u_{ff}重合，显然两个骨架也重合。反之，如果两个骨架重合，则算法2的输出为

$$\tilde{u}_{ff} = \left(I - P_{holes}^{ff} P_{ff}^{holes}\right) u_{ff}, \tag{7}$$

即，在孔洞 $\Omega_{holes}^{h_x\ h_y}$，上为 $\tilde{u}_{ff} = 0$，并且在骨架 $\Omega^{h_x,\ H_y} \cup \Omega^{H_x,\ h_y}$上为 $\tilde{u}_{ff} = u_{ff}$。然后 $\tilde{u}_{ff}$经过算法3得到

$$u^{(0)} = \tilde{u}_{ff} + P_{holes}^{ff} \left(P_{ff}^{holes} A_{ff} P_{holes}^{ff}\right)^{-1} \left(P_{ff}^{holes} b_{ff} - P_{ff}^{holes} A_{ff} \tilde{u}_{ff}\right),$$

根据方程（7）

$$\begin{aligned} u^{(0)} &= \tilde{u}_{ff} + P_{holes}^{ff} \left(P_{ff}^{holes} A_{ff} P_{holes}^{ff}\right)^{-1} \left(P_{ff}^{holes} A_{ff} P_{holes}^{ff} P_{ff}^{holes} u_{ff}\right) \\ &= \tilde{u}_{ff} + P_{holes}^{ff} P_{ff}^{holes} u_{ff}, \end{aligned}$$

这等于 u_{ff}，从而证明了陈述1）。

陈述 2）这是注2.5的推论。事实上，考虑 $P_{ff}^{holes} r$，即 $\Omega_{holes}^{h_x\ h_y}$ 上的残差投影，。于是有

$$\begin{aligned} P_{ff}^{holes} r &= P_{ff}^{holes} \left(b_{ff} - A_{ff} u^{(0)}\right) \\ &= P_{ff}^{holes} \left(b_{ff} - A_{ff} \left(\tilde{u}_{ff} + P_{holes}^{ff} v\right)\right) \\ &= P_{ff}^{holes} \left(b_{ff} - A_{ff} \left(\tilde{u}_{ff} + P_{holes}^{ff} \left(P_{ff}^{holes} A_{ff} P_{holes}^{ff}\right)^{-1} \left(P_{ff}^{holes} b_{ff} + c\right)\right)\right) \\ &= P_{ff}^{holes} b_{ff} - P_{ff}^{holes} A_{ff} \tilde{u}_{ff} - P_{ff}^{holes} A_{ff} P_{holes}^{ff} \left(P_{ff}^{holes} A_{ff} P_{holes}^{ff}\right)^{-1} \left(P_{ff}^{holes} b_{ff} + c\right) \\ &= P_{ff}^{holes} b_{ff} - P_{ff}^{holes} A_{ff} \tilde{u}_{ff} - P_{ff}^{holes} b_{ff} - c \end{aligned}$$

由于 $c = -P_{ff}^{holes} A_{ff} \tilde{u}_{ff}$，该值为零。因此，孔洞上的残差为零。此外，在 $u^{(0)} \neq u_{ff}$ 的情况下，残差非零；根据前面的结果，它只能在其骨架上非零。

陈述 3）这是因为粗点（x_i，y_j）$\in \Omega^{h_x,\ h_y} \cap \Omega^{H_x,\ H_y}$ 是 4 个子域 $\Omega_i^{h_x\ h_y}$、、$_j$之间的交叉点，其中某些 i、j 对应的 4 个解是通过错误的边界条件计算得到的，因为 $u^{(0)} \neq u_{ff}$。因此，如果（x_i，y_j）是两个各向异性网格之间的交叉点，那么 $r_{i,\ j}$（仅根据骨架上的单元计算）与 $r_{i\pm1,\ j}$ 或 $r_{i,\ j\pm1}$（根据两个定义域上的单元计算）之间可能存在显著差异。□

注记3.3。在命题3.2的证明中，并未明确建立残差结构与各向异性线性系统所用离散方法之间的联系。因此，即使采用其他离散格式，得到的残差仍具有相同的稀疏结构。

3.2 迭代方法

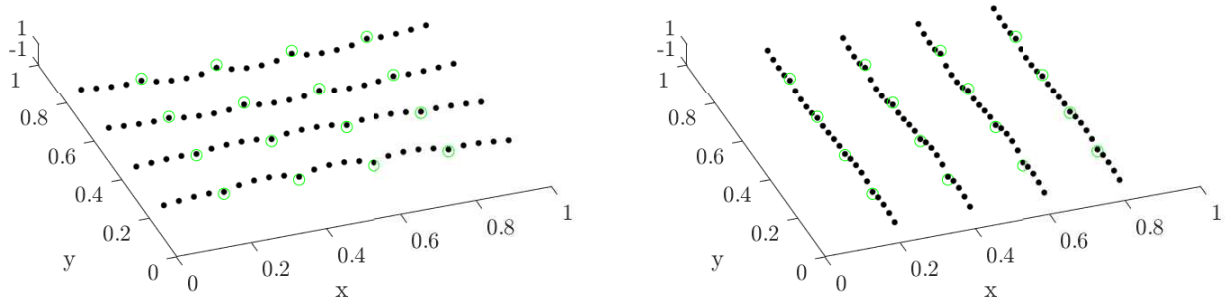
ASDSM算法（总结于算法4）通过利用残差结构，对初始猜测算法1进行迭代。

计算出初始猜测后，我们计算残差，并使用算法1迭代地进行修正，以获得误差的初始猜测。

具体而言，我们在第3步计算残差，并观察到，如命题3.2所示且图3b所示，残差集中在解的骨架上。因此，在第4步中，我们使用投影算子 P_{ff}^{fc}和 P_{ff}^{cf}将残差投影到各向异性网格$\Omega^{h_x,\ H_y}$和$\Omega^{H_x,\ h_y}$上，分别如图4a和图4a所示。在第5步中，我们通过初始猜测算法1计算误差方程 $A_{ff} e_{ff} = r_{ff}$ 的近似解 $\tilde{e}_{ff}$。

算法4 ASDSM算法

ASDSM函数
1) $u^{(0)} =$ 初始猜测 $(b_{ff}, b_{fc}, b_{cf}, b_{cc})$ ▷ 计算初始猜测
2) $k = 0$
当 $\|b_{ff} - A_{ff}u^{(k)}\| <$tol | *残差停滞* 时执行
3) $r_{ff} = b_{ff} - A_{ff}u^{(k)}$ ▷ 计算 Ω^{h_x} 中的残差，h_y
4.1) $r_{fc} = P_{ff}^{fc} r_{ff}$ ▷ 将残差投影到 Ω^{h_x} 中，H_y
4.2) $r_{cf} = P_{ff}^{cf} r_{ff}$ ▷ 将残差投影到 Ω^{H_x}，h_y
5) $\tilde{e}_{ff} =$ 初始猜测 $(0 \cdot b_{ff}, r_{fc}, r_{cf})$ ▷ 近似计算误差
6) $\hat{s} = \arg\min_{s \in \mathbb{R}^+} \|b_{ff} - A_{ff}\left(u^{(k)} + s\tilde{e}_{ff}\right)\|$
7) $u^{(k+1)} = u^{(k)} + \hat{s}\tilde{e}_{ff}$ ▷ 应用缩放校正
8) $k = k + 1$
结束 while
返回 $u^{(k)}$
结束 function



（a） r_{fc}：残差 r_{ff} 投影到网格 $\Omega^{h_x,\ H_y}$。 (b) r_{cf}：残差 r_{ff} 投影到网格 Ω^{H_x,h_y} 上。

图 4：投影到两个各向异性网格上的残差。

然而，这一过程中我们面临两个挑战：残差不够光滑，无法使用Richardson外推；两个各向异性误差方程 $A_{fc}e_{fc} = r_{fc}$ 和 $A_{cf}e_{cf} = r_{cf}$ 的不同尺度可能导致解 e_{fc} 和 e_{cf} 的范数相差很大。事实上，正如命题3.2所述且图4a和图4b所示，我们观察到残差曲面存在不规则之处，这些不规则处靠近 $\Omega^{H_x,\ H_y}$ 的粗网格点（绿色），也就是两个各向异性网格的交点。此外，观察图5a、5b中的 z 轴，我们发现尺度略有差异；随着网格点增多，这种差异可能会进一步扩大。

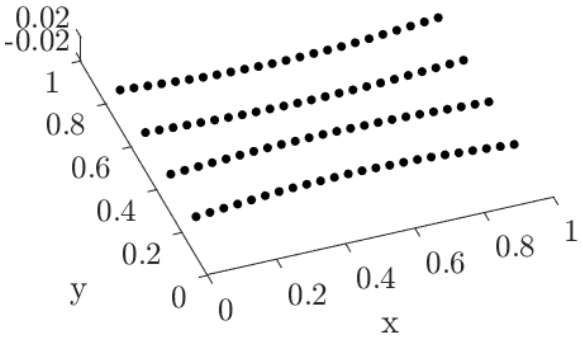
因此，为了解决这些问题，在算法2的步骤3)中，我们在 $P_{fc}^{cc}u_{fc}$ 和 $P_{cf}^{cc}u_{cf}$ 之间随机选择一个作为 $\hat{}u_{cc}$，因为 $\hat{}u_{cc}$ 无法进行外推；而在算法1的步骤2)中，我们在构造骨架之前对两个解进行归一化。归一化确保两个粗解具有近似相同的范数，因为它们都是较细网格上一个解的子采样。
在主算法的最后一步（步骤6)）中，我们对更新量 $\tilde{e}_{ff}$ 进行最优缩放，使残差最小化。此步骤对于在归一化后正确缩放解是必要的，并且对于确保残差不增加至关重要。

注3.4。ASDSM 算法 4 的步骤7)中采用的缩放确保算法不会发散，但无法避免停滞。此外，由于误差是通过算法1计算的，我们推断命题3.2对 $u^{(k)}, \forall k \geq 0$同样成立。

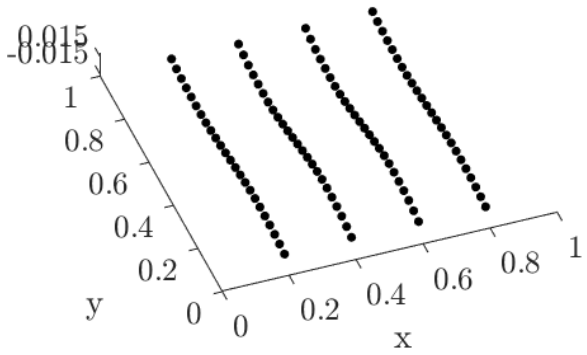
3.3 高效实现

为了高效实现 ASDSM，应考虑以下注记。

- 可以利用残差稀疏性，将计算限制在非零元素上，从而降低计算成本；



(a) e_{fc}：在网格 $\Omega^{h_x,\ H_y}$上计算的误差。



(b) e_{cf}：在网格 $\Omega^{H_x,\ h_y}$上计算的误差。

图5：两个各向异性误差方程的解。

- 计算 $\hat{s}$ 需要求解一个含一个未知量和 $O\left(N^{(x)c}\ \ N_{(x)f}\ \ + N_{(y)c}\ \ N_{(y}_{f}\right)$ 个方程的超定线性系统；与考虑全部方程相比，选取方程的随机子集可以提高计算效率；

- 计算误差的初始猜测时，填充过程包括并行求解许多齐次偏微分方程。在某些情况下，齐次偏微分方程的解可以解析求得，这样基本上就能消除填充过程的成本；

- 算法3第2步中的块矩阵求逆可以通过矩阵自由方法 [15]实现；
- 根据命题3.2，可以通过智能内存管理来降低迭代未知量 $u(k)$ </style}的存储需求：只存储骨架及其附近的元素，而非所有$(x)f\ \ N_{(y)f}$分量。这样可以将存储需求降低至 $O\left(N_{(x)c}\ \ N_{(y)f}\ \ + N_{(y)c}\ \ N_{(x}_{f}\right)$) </style}。

3.4 扩展至不同的离散化方法和偏微分方程

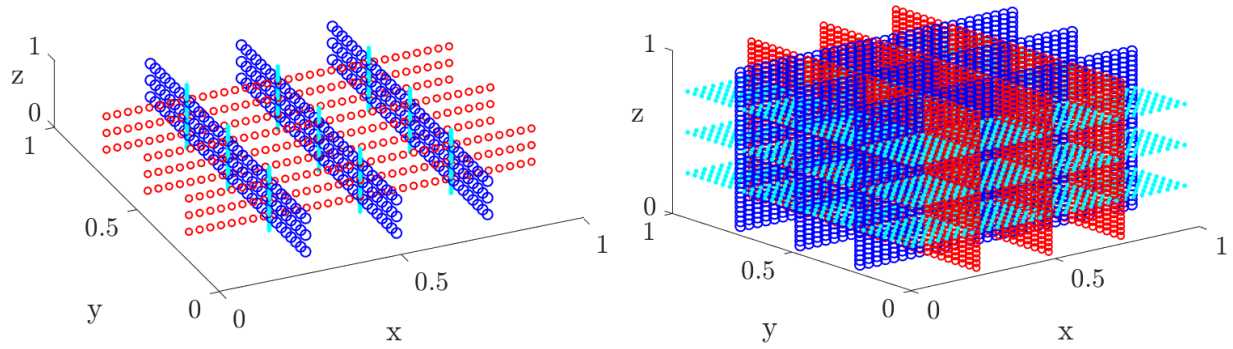
对于高阶偏微分方程或离散格式精度阶数较高的情况，离散化过程会产生带宽大于3的带状系数矩阵，并导致边界条件数量随之增加。这可能会造成难以恰当填充孔洞 $\Omega_{holes}^{h_x\ h_y}$，因为对每个孔洞上的偏微分方程进行离散化所得到的线性系统可能需要多个边界条件。一种可能的解决方案是在定义域边缘附近使用不同的离散方法。例如，通过泰勒展开，我们可以得到以下二阶导数的三阶精度有限差分离散化

$$
\begin{aligned}
u''(x_i) &= \frac{-u_{i-2} + 16u_{i-1} - 30u_i + 16u_{i+1} - u_{i+2}}{12h^2} + O(h^3);\\
u''(x_i) &= \frac{10u_{i-1} - 15u_i - 4u_{i+1} + 14u_{i+2} - 6u_{i+3} + u_{i+4}}{12h^2} + O(h^3);\\
u''(x_i) &= \frac{u_{i-4} - 6u_{i-3} + 14u_{i-2} - 4u_{i-1} - 15u_i + 10u_{i+1}}{12h^2} + O(h^3).
\end{aligned}
$$

因此，我们可以通过上述等式定义孔洞中的线性系统，从而只需一个边界条件。然后，为确保注2.5成立，我们构造细网格系数矩阵 A_{ff}，使其子块与孔洞中的线性系统一致。由于注2.5对于命题3.2的证明至关重要，因此从理论上讲，可以将ASDSM扩展到高阶偏微分方程或具有更高阶精度的离散格式。

3.5 三维情形的扩展

在三维情形中，我们旨在细三维均匀网格上求解偏微分方程 (1)。扩展第 2.1 节中引入的记号后，我们将此类网格记为 $\Omega^{h_x,h_y,\ h_z}$，其中 h_x、h_y、h_z 分别表示各维度的网格宽度。扩展 ASDSM 的一种最优方法，是在四种不同的网格上对偏微分方程进行离散：三种强各向异性网格 $\Omega^{H_x,h_y,\ h_z}$、$\Omega^{h_x,H_y,\ h_z}$、$\Omega^{h_x,h_y,\ H_z}$，以及一种粗各向同性网格 $\Omega^{H_x,H_y,\ H_z}$。然而，如图 6a 所示，问题在于嵌套网格。在这种情况下，立方体空区域仅有 8 个网格点作为边界条件，即



(a) 红色网格 $\Omega^{h_x,\ H_y,\ H_z}$ ，蓝色网格 $\Omega^{H_x,\ h_y,\ H_z}$ ；(b) 红色网格 $\Omega^{H_x,\ h_y,\ h_z}$ ，蓝色网格 $\Omega^{h_x,\ H_y,\ h_z}$ ，青色网格 $\Omega^{H_x,\ H_y,\ h_z}$。青色网格 $\Omega^{h_x,\ h_y,\ H_z}$。

图 6：各向异性网格的不同选择。

立方体的角点处，且各个面上都没有给出解。因此，所获得的骨架无法用作新互不相交子问题的边界条件。

一种可能的解决方案是考虑更密集的各向异性网格，例如 $\Omega^{H_x,\ h_y,\ h_z}$、$\Omega^{h_x,\ H_y,\ h_z}$、$\Omega^{h_x,\ h_y,\ H_z}$。从图 6b 可以看出，使用更密集的网格后，孔洞内互不相交子问题的边界条件定义得更加明确。不过需要注意的是，由于网格密度增加，这种方法也会导致计算成本显著上升。

在数值结果一节中，讨论三维情形时，我们将通过依次遍历网格 $\Omega^{H_x,\ h_y,\ h_z}$、$\Omega^{h_x,\ H_y,\ h_z}$、$\Omega^{h_x,\ h_y,\ H_z}$ 来实现 ASDSM，并将 $\Omega^{H_x,\ H_y,\ H_z}$ 作为粗交集网格。请注意，与二维情形不同，网格 $\Omega^{H_x,\ H_y,\ H_z}$ 并不包含先前网格之间的所有交点。例如，$\Omega^{h_x,\ H_y,\ H_z} = \Omega^{h_x,\ h_y,\ H_z} \cap \Omega^{h_x,\ H_y,\ h_z}$。因此，为确保合并过程定义良好，我们还对其他交点进行外推。

4 数值结果

本节给出数值结果，以评估 ASDSM 的性能。该方法使用 Matlab 2020b 实现，测试在配备 AMD Ryzen 5950X 处理器和 64GB 随机存取存储器的机器上运行。

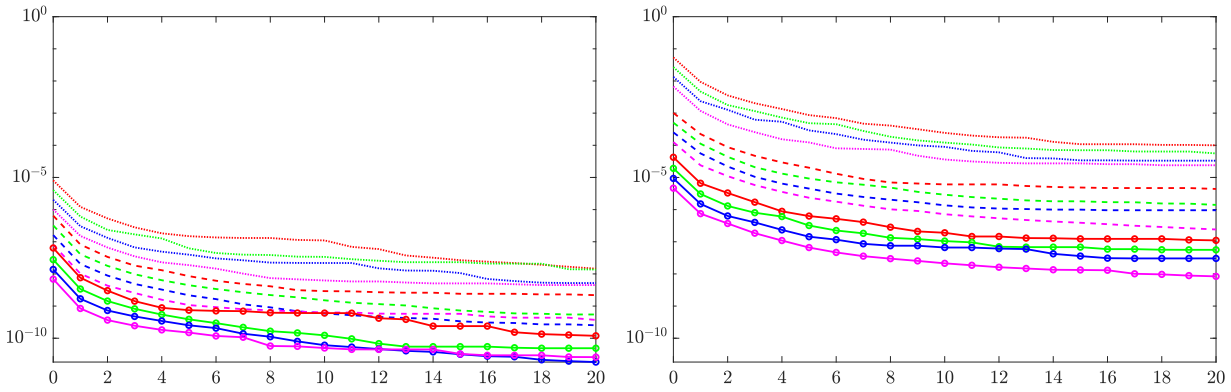
本节使用的算法可在 [17] 中获取。在以下示例中，我们设定 $N_c^{(x)} = N_{(y)\,c} = N_{(z)\,c} = N_c$以及 $N_f^{(x)} = N_f^{(y)} = N_f^{(z)} = N_f$。此外，每个示例都将测试 ASDSM 在两种情形下求解 (1) 或 (2) 的表现：一种情形下解是光滑的，另一种情形下解为振荡解。因此，我们使用记号 $r_k^{(1)}, r_k^{(2)}$ 分别表示 ASDSM 在第一种和第二种情形下求解偏微分方程时第 k 次迭代的残差。请注意，当 $k = 0$, $r_0^{(j)}$，$j = 1$ 时，2 表示通过算法1计算得到的初始猜测的残差。

示例1：这里我们测试 ASDSM 在以下设置下求解二维稳态对流扩散方程 (1) 的表现：

1) $\alpha_1 = \alpha_2 = \beta_1 = \beta_2 = 1$ 以及由精确解 $u(x, y) = \sin(x\pi + y\pi)$ 计算得到的函数 s, g；2) $\alpha_1 = 1 + x^2$, $\alpha_2 = 2 + xy$, $\beta_1 = 2 - x$, $\beta_2 = 1 + y$ 以及由精确解$u(x, y) = \sin(4x\pi + 4y\pi)$计算得到的函数 s, g。

我们注意到，设置 1) 和 2) 所选的系数 α_i, β_i 不同：在设置 1) 中它们是相等的常数，而在设置 2) 中它们是正函数。此外，设置 2) 中的解比设置 1) 中的解更具振荡性，因此问题更难求解，尤其是在网格极度各向异性的情况下，因为需要更多的点才能更好地逼近解。

图 7a 和图 7b 分别展示了上述两种设置下，迭代次数为 $k = 0,...,$ 20 时的残差 $r_k^{(1)}, r_k^{(2)}$。点线、虚线和带圆圈的线分别表示 $N_c = 5$、10、20，而红色、绿色、蓝色和洋红色线分别表示 $N_f = 400$、800、1600、3200。每种情况下，所求解的线性系统的规模为 $N_f \times N_f$，因此，相同颜色的不同类型线表示求解同一线性系统时 ASDSM 的残差



(a) 设置 1)　　(b) 设置 2)

图 7：将 ASDSM 应用于方程 (1) 时，两种设置下每次迭代的残差 $r_k^{(1)}, r_k^{(2)}$。点线、虚线和带圆圈的线分别表示 $N_c = 5$、10、20，而红线、绿线、蓝线和洋红线分别表示$N_f = 400$、800、1600、3200。

。这是因为在 ASDSM 中只需要 N_c 来定义各向异性线性系统的规模。因此，较大的 N_c 会导致 ASDSM 的迭代计算开销更大。

在图 7a 中，考虑的是设置 1)。可以观察到，初始猜测算法中的 Richardson外推带来了显著的初始提升，使每种 N_c, N_f 组合的 $r_0^{(1)}$ 都非常小。随后，ASDSM 的第一次迭代将残差进一步大幅降低，约一个数量级。然而，第一次迭代之后，降低因子开始减小，并且随着 k 增大，似乎趋近于 1，从而导致停滞。这可能是由于各向异性线性系统的精度较低。随着 ASDSM 越来越接近解，误差的初始猜测变得越来越不准确。事实上，我们注意到，固定 N_f 并增大 N_c，即同时降低各向异性并提高精度时，停滞点会降低。另一个原因可能与残差的光滑性有关（更多细节参见例2）。

此外，在固定 N_c 并增大 N_f（这会增加各向异性）时，停滞点似乎仍会下降。这表明，增大主线性系统的规模，进而提高其解的精度，可以改善 ASDSM 的性能。

考虑设置 2) 时，我们从图 7b 中注意到，$r_k^{(2)}$ 相对于 $r_k^{(1)}$ 呈现出类似的行为。主要区别在于起始点位于 $k = 0$。在这种情况下，由于解呈振荡状态，外推交叉点的精度低于前一种情况。因此，初始猜测算法得到的解精度较低。精度损失的另一个原因可能是使用了变系数 α_i, β_i, $i = 1, 2$，而不是设置 1) 中使用的常系数。事实上，当 $N_c = 5$ 时，残差 $r_0^{(1)}$ 比 $r_0^{(2)}$ 小近 4 个数量级；而当 $N_c = 10, 20$ 时，则小 3 个数量级。

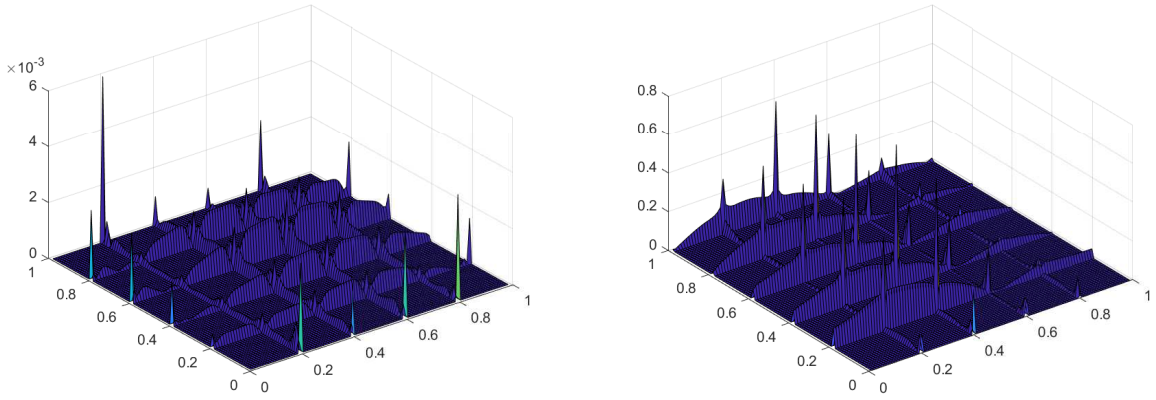
例2：在本例中，我们展示ASDSM的一个重要缺点，即会使粗点处的残差变得高度不规则。考虑以下两个玩具问题：

1）二维空间稳态对流扩散方程（1），其中 $\alpha_i = \beta_i = 1$、 $i = 1$、2和 s、 g由精确解 $u(x, y) = \sin(x\pi + y\pi)$计算得到；

2）一维空间、时间相关的对流扩散方程（2），其中 $\alpha_1 = \beta_1 = 1$ 以及 s、 g 由精确解 $u(x, t) = \sin(x\pi + t\pi)$计算得到。

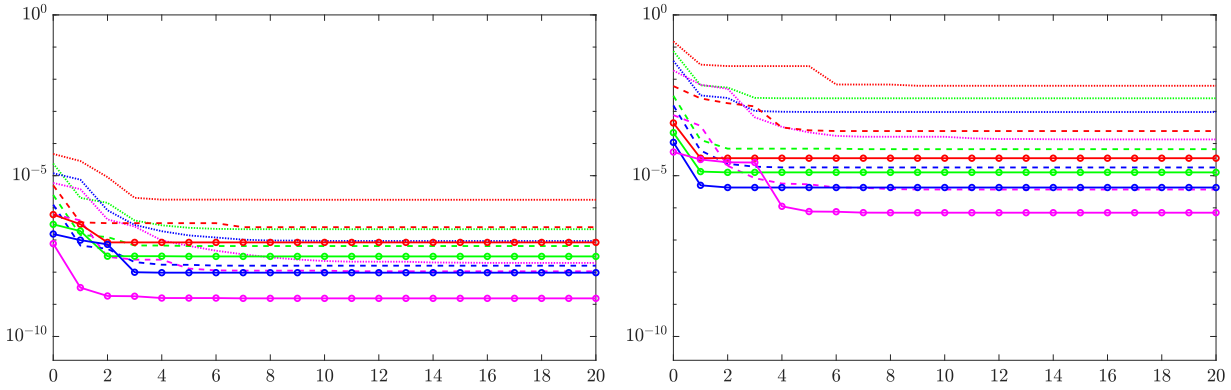
图 8展示了ASDSM迭代20次后得到的重塑后的残差，即ASDSM发生停滞时的结果。在这两种情况下，都可以观察到粗点处存在明显的不规则性。在时变情形下，时间方向稠密网格Ω^{H_x,h_t}沿线也观察到较不严重的不规则性。这导致ASDSM 算法 4的步骤4.1）和4.2）中的投影残差高度不规则，进而使算法1的步骤1.1）和1.2）中的解不准确，最终导致停滞。

本文未展示的其他测试表明，对残差进行人工平滑可减少停滞点。



(a) 稳态偏微分方程 1)　　(b) 时变偏微分方程 2)

图 8：ASDSM 迭代 20 次后的重塑绝对残差



(a) 设置 1)　　(b) 设置 2)

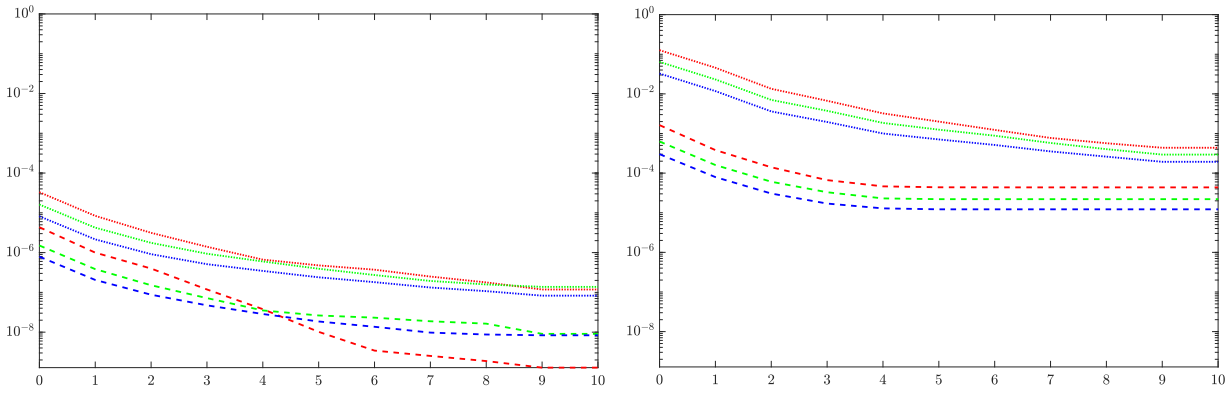
图 9：ASDSM 应用于方程 (2) 时，两种设置下每次迭代的残差 $r_k^{(1)}, r_k^{(2)}$。点线、虚线和带圆圈的线分别表示 $N_c = 5$、10、20，而红线、绿线、蓝线和品红线分别表示$N_f = 400$、800、1600、3200。

示例3：这里我们测试 ASDSM 在以下设置下求解一维时空依赖对流扩散方程 (2)：

1) $\alpha_1 = \beta_1 = 1$ 以及根据精确解 $u(x,t) = \sin(x\pi + t\pi)$计算得到的函数 s, g；2) $\alpha_1 = 1 + x^2$, $\beta_1 = 2 - x$ 以及根据精确解 $u(x,t) = \sin(4x\pi +$计算得到的函数 s, g $4t\pi)$.

请注意，这两种设 置分别涉及一个光滑解和一个振荡解，与示例1中的相同，但 h g 用变量 t 替代 y.

采用与示例1相同的记号，图 9a 和图 9b 分别显示了第 k次迭代时的绝对残差 $\left|r_k^{(1)}\right|, \left|r_k^{(2)}\right|$。比较图 9 和图 7 可知，总体而言，示例1中的相同观察结果也适用于这一时变情形。不过，主要区别在于残差的变化趋势：图 9 中的趋势比图 7 中的更不平滑。这种行为的原因在于骨架构建算法 2 的第 3) 步：误差曲面的交叉点无法外推，因此将一个随机解作为“最精确的解”。在这种情况下，在时间稠密网格 $\Omega^{H_x,\ h_t}$ 上计算得到的解 $e_{H_x,\ h_t}$ 比在空间稠密网格 $\Omega^{h_x,\ H_t}$上计算得到的解 $e_{h_x,\ H_t}$ 更精确。因此，当随机选取的交叉点属于误差 $e_{H_x,\ h_t}$时，残差会大幅减小。然而，由于示例1中所述的相同原因，残差似乎只会如此大幅减小几次，之后便会出现停滞。



(a) 设置 1)　　(b) 设置 2)

图10：ASDSM 应用于方程 (1) 时，在两种设置下每次迭代的残差 $r_k^{(1)}$, $r_k^{(2)}$ 。点线、虚线分别表示 $N_c = 5$,10；红色、绿色、蓝色和洋红色线表示$N_f = 70$, 150,300。

示例4：这里我们测试 ASDSM 在以下设置下求解三维稳态对流扩散方程 (1)：

1) $\alpha_i = \beta_i = 1$, $i = 1, 2$、3 以及函数 s, g 根据精确解 $u(x, y, z) = \sin(x\pi + y\pi + z\pi)$计算得出；2) $\alpha_1 = 1 + x^2$, $\alpha_2 = 2 + xy$, $\alpha_3 = 3 - xyz$, $\beta_1 = 2 - x$, $\beta_2 = 1 + y$, $\beta_3 = 2 - x + yz$ 以及函数 s, g根据精确解 $u(x, t) = \sin(4x\pi + 4y\pi + 4z\pi)$计算得出。

设置1)和2)中的平滑解和振荡解是练习1中设置的扩展，新增了时间变量 t。图10a、10b分别展示了在设置1)和2)下，将ASDSM应用于偏微分方程(1)时，第 k次迭代的残差 $r_k^{(1)}$ 、 $r_k^{(2)}$ 。为缩短测试的计算时间，未提供此前以带圈曲线表示的 $N_c = 20$的结果，研究仅限于 $N_f \in \{70, 150, 300\}$，分别以红色、绿色和蓝色表示。
从图10可以看出，尽管从算法角度来看，三维情形更为复杂，而且包含更多交叉点，但残差的行为与示例1类似。因此，同样的观察结果也适用，并且预计对于更大的 N_c、 N_f，也会得到类似的结果。

5 结论

ASDSM算法为基于结构化稀疏残差的方法开辟了新视野。这些初步结果表明，通过将问题分解为互不相交的子问题并合并所得解，我们能够有效地将问题的维数降低1，并提高计算效率。尽管仍需进一步研究以解决收敛性和粗点最优选择等问题，但ASDSM算法有望成为处理大型线性系统的有用工具，既可作为求解器，也可为其他迭代求解器（如区域分解方法）提供良好的初始猜测。它还可用于为射击法提供快速解，因为射击法并不总是要求高精度。

在未来的工作中，我们将探索克服当前算法局限性的替代方案，并研究其在非线性问题和非结构网格中的潜在应用。

致谢

本研究得到 EuroHPC TIME-X 项目 n. 955701 的支持。

参考文献

[1] H. Bungartz, M. Griebel, 《稀疏网格》, 《数值学报》, 第13卷, 第147-269页, 2004年。

[2] S. Doi, 《不完全LU分解的并行性与收敛性》, 《应用数值数学》, 第7.5卷, 第417-436页（1991年）。[3] L. Formaggia, S. Micheletti, S. Perotto, 《计算流体动力学中的各向异性网格自适应：应用于对流-扩散-反应和Stokes问题》, 《应用数值数学》, 第51卷, 第511-533页（2004年）。[4] M.J. Gander, J. Liu, S.L. Wu, X. Yue, T. Zhou, 《Paradiag：基于对角化技术的时间并行算法》, arXiv预印本 arXiv:2005.09158（2020年）。[5] W. Hackbusch, 《多重网格方法及其应用》, Springer Science & Business Media, 第4卷, 2013年。[6] J. Han, A. Jentzen, E. Weinan, 《利用深度学习求解高维偏微分方程》, 《美国国家科学院院刊》115.34, 第8505-8510页（2018年）。[7] C. Hur´e, H. Pham, X. Warin, 《用于高维非线性偏微分方程的深度反向格式》, 《数学计算》89.324, 第1547-1579页（2020年）。[8] J. van Lent, S. Vandewalle, 《各向异性偏微分方程的多重网格波形松弛》, Springer, 《数值算法》31, 第361–380页（2002年）。[9] I.D. Lirkov, S.D. Margenov, P.S. Vassilevski, 《椭圆问题的循环块分解预条件子》, Springer, 《计算》53, 第59-74页（1994年）。[10] I.D. Lirkov, S.D. Margenov, L.T. Zikatanov, 《各向异性椭圆问题的循环块分解预条件》, Springer, 《计算》58, 第245-258页（1997年）。[11] Z. Liu, W. Cai, Z.Q.J. Xu, 《用于求解复杂定义域中Poisson-Boltzmann方程的多尺度深度神经网络（MscaleDNN）》, 《计算物理通讯》28卷第5期, 第1970-2001页（2020年）。[12] I.G. Lisle, S.L.T. Huang, G.J. Reid, 《偏微分方程对称性的结构：利用部分积分系统》, 《2014年符号-数值计算研讨会论文集》（2014年）。[13] M.M. Rahaman, H. Takia, Md.K. Hasan, Md.B. Hossain, S. Mia, 和 K. Hossen, 《对流扩散方程在水溶液污染物测定中的应用：数学分析》, 《应用数学与物理》10卷第1期（2022年）。[14] S.A. Richards, 《空间和时间上的完整Richardson外推》, 《工程数值方法通讯》13.7, 第573-582页（1997年）。[15] J. Sch¨afer, X. Huang, S. Kopecz, P. Birken, M.K. Gobbert, A. Meister, 《一种适用于具有非光滑源项的对流-扩散-反应系统的内存高效有限体积法》, 《偏微分方程数值方法》31卷第1期, 第143-167页（2015年）。[16] R.N. Singh, 《近地表地球物理与环境科学中的对流扩散方程模型》, 《印度地球物理联盟期刊》17卷第2期, 第117-127页（2013年）。[17] K. Trotti, ASDSM算法二维和三维版, https://github.com/ken13889/ASDSM[18] S.A. Vinogradova, L.A. Krukier, 《不完全LU分解在各向异性介质对流扩散过程建模中的应用》, 《数学模型与计算机模拟》5.2, 第190-197页（2013年）。[19] X. Warin, 《用于高维非线性偏微分方程的嵌套蒙特卡罗》, 《蒙特卡罗方法与应用》24.4, 第225-247页（2018年）。[20] E. Weinan, J. Han, A. Jentzen, 《求解高维偏微分方程的算法：从非线性蒙特卡罗到机器学习》, 《非线性》35.1, 第278号（2021年）。