使用神经网络求解离散中子扩散方程

Toby R.F. Phillips [a], Claire E. Heaney[a], Boyang Chen[a], Andrew G. Buchan[b], Christopher C. Pain [a]

[a] *Applied Modelling and Computation Group, Department of Earth Science and Engineering, Imperial College London, London, SW7 2AZ United Kingdom*[b] *School of Engineering and Materials Science, Queen Mary University of London, London, E1 4NS United Kingdom*

arXiv:2301.09939v1 [cs.CE] 24 Jan 2023

摘要

本文提出一种新方法，利用人工智能（AI）软件库中的工具，作为求解采用标准数值方法离散化的偏微分方程（PDEs）的替代途径。具体而言，我们介绍如何通过预先设定神经网络中卷积层的权重，来表示由有限体积法和有限元法产生的数值离散。由于权重由离散格式决定，因此无需训练网络，所得解与使用标准代码（通常以 Fortran 或 C 编写）得到的解完全相同（考虑求解器容差）。我们还说明如何利用人工智能库提供的函数实现雅可比方法和多重网格求解器。对于后者，我们采用能够表示锯齿多重网格方法的 U-Net架构。以这种方式使用人工智能库的一个优势是，可以利用其强大功能和内置技术。例如，这些库的执行已针对不同的计算机架构进行优化，无论是中央处理器、图形处理器还是新一代人工智能处理器。

本文将所提出的方法应用于反应堆物理中的特征值问题，其中中子输运由扩散理论描述。对于一个燃料组件基准算例，我们证明，采用新方法得到的解与使用 Fortran 按标准方式编写相同离散化方案所得的解相同（考虑求解器容差）。随后，我们使用新方法求解一个反应堆堆芯基准算例。

关键词： 偏微分方程数值解；有限差分法；有限体积法；卷积神经网络；多重网格求解器；U-Net；中子扩散方程；反应堆物理

1. 引言

新型计算硬件的开发也带来了一个挑战：必须调整代码，才能将其成功部署到这些新架构上。在人工智能（AI）领域，这一

挑战在很大程度上已由广泛使用的人工智能库（例如 TensorFlow [1] 和 PyTorch [2]）的编写者和贡献者解决。在这些库中，与架构相关的代码已被抽象出来，用户可以专注于自己想要实现的算法，而不必考虑或理解与计算机体系结构相关的代码。因此，用户只需对代码进行少量修改，就能在中央处理器（CPU）、图形处理器（GPU）甚至张量处理单元（TPU）上运行代码。在科学计算等其他领域，可能是因为代码和库的标准化程度较低且数量较多，用户要让代码在新架构上运行就得付出更多努力。如今，将代码移植到 CPU 集群相对容易，但在集群上运行所能带来的计算性能提升，会受到内存访问和数据传输的限制。尽管图形处理器的性能已证明优于中央处理器，但必须使用 CUDA 或 OpenCL 等语言编写图形处理器指令，而许多从事科学计算的人并不熟悉这些语言。额外的编码工作阻碍了图形处理器的普及，不过也有成功应用的例子，例如在计算流体力学中模拟 [3], 声波 [4] ，以及在辐射输运中应用于蒙特卡罗中子输运代码 [5] 和特征值问题 [6]。借助 CUDA 和 OpenCL，图形处理器已被用于加速非结构网格 [7–10] 以及间断伽辽金方法 [11]的有限元矩阵生成。最近，专为人工智能相关任务（如矩阵乘法和向量运算）设计的新型处理器问世。因此，这些处理器也适用于科学计算领域中的线性代数计算 [12]。此外，这些新型处理器的设计比中央处理器或图形处理器更加节能，单个芯片上拥有数十万个处理器核心，因此研究人员可借助这种理想的硬件，以节能的方式运行计算密集型问题。人工智能库已经能够在这些所谓的人工智能处理器上运行，其中包括谷歌的张量处理单元 [12], 、Graphcore 的智能处理单元（IPU） [13] 和 Cerebras 的 CS-2 [14]。为了利用图形处理器或人工智能处理器的速度进行科学计算，本文提出一种方法，将数值离散表示为人工智能库中的运算或函数，例如离散卷积。以这种方式编写离散化代码，意味着无论可用平台是中央处理器、图形处理器还是新型人工智能处理器，都可以部署代码，而无需对代码进行重大修改。

先前利用人工智能处理器的线性代数能力、借助人工智能库解决科学问题的工作包括：分布式傅里叶变换 [15, 16]；用于金融领域的蒙特卡洛模拟 [17]；多体量子物理 [18]；以及密度泛函理论 [19]。我们发现了四项先前工作，它们利用人工智能库中可用的神经网络相关运算来解决科学问题 [15, 16]– [17]。

Zhao 等人 [20] 首次将纳维–斯托克斯方程的有限差分离散化与卷积神经网络对应起来，其中网络权重由离散化确定。为进行验证，他们使用了多项基准测试，包括顶盖驱动方腔流和绕圆柱流动。Wang 等人 [21] 提出了与 Zhao 等人 [20], 类似的思路，同样使用 TensorFlow 实现计算流体力学问题的有限差分离散化，但采用的是张量处理单元，而非图形处理器。他们求解了变密度纳维–斯托克斯方程，并展示了良好的弱扩展性和强扩展性。Chen 等人 [22] 通过卷积神经网络实现有限差分离散和有限元离散，以求解多个计算流体力学问题。他们开发了一种求解离散方程组的方法，将锯齿多重网格方法与雅可比方法相结合，并将其实现为 U-Net [24]（一种具有特定架构的卷积神经网络）。Phillips 等人 [23] 使用权重预先确定的卷积神经网络，实现了迎风有限体积离散，以及若干种源自一种新型卷积有限元法（ConvFEM）的有限元离散。他们研究的辐射输运问题需要开发四维多重网格法，该方法同样基于 U-Net。研究人员此前已指出，多重网格方法与编码器—解码器类型的神经网络（如 U-Net） [24] 之间存在相似之处。因此，人们已探索将多重网格启发式架构用于（经训练的）神经网络，并证实在计算机视觉 [25, 26] 和计算流体力学（CFD） [27, 28] 应用中，相较于传统卷积神经网络架构，这种方法能够提升性能。Margenberg 等人 [29] 则采用了不同的方法：使用（经训练的）神经网络生成多重网格较细层级上的解，并使用标准计算流体力学求解器生成较粗层级上的解。与先前将多重网格方法和神经网络相结合的示例 [25–29], 不同，Chen 等人 [22] 使用（未经训练的）神经网络及预先确定的权重实现多重网格方法，在粗层级上求解偏微分方程，并在较细层级上计算残差和进行雅可比松弛。我们当前的研究采用的正是这种方法。

本文介绍了如何使用卷积神经网络实现中子扩散方程的有限体积离散，该网络的权重由特定离散格式预先确定。（在本例中，我们的有限体积离散等价于有限差分离散。）我们还使用这一方法实现了中子扩散方程的二次有限元离散。雅可比方法和锯齿形多重网格方法被用作求解器，并通过人工智能软件库中的标准操作加以实现。我们使用燃料组件基准算例展示了该方法，并将中子通量空间变化的解与采用 Fortran 按照标准方式编写相同离散所得的解进行比较。随后，我们使用所提出的方法求解反应堆堆芯基准算例。本文所述方法是一种全新且不同的

利用人工智能技术求解控制偏微分方程的本土化方法。最终，通过这一新方法获得的解与标准代码得到的解完全一致，但通过人工智能库执行所有操作的优势在于，代码能够在所有架构上高效运行。此外，借助神经网络，还可以将灵敏度 [30] , 不确定性量化 [31, 32] 和数据同化 [33]等方法的最新发展付诸实践。尽管人工智能正在核工程领域日益普及，但其应用通常是通过代理建模实现的，而这需要训练神经网络。当前工作的一些实例包括：使用物理信息神经网络处理点堆动力学 [34] 和非光滑非均匀中子扩散问题 [35]; 用于瞬态分析 [36], 特征值问题 [37] 和数字孪生 [33]的代理模型；以及使用神经网络进行截面生成 [38]。然而，本文提出的方法从根本上有所不同：它提供了一种精确表示给定偏微分方程组离散化的替代方式，而代理模型提供的则是离散化偏微分方程组的近似。在将这些离散化表述为神经网络之后，一个显而易见的扩展是结合未训练网络（即本文所述具有预先确定权重的网络）和训练网络，以构建更高效、更强大的数字孪生，正如先前已有研究所观察到的那样 [20, 22, 39]。

本文各章节安排如下。第 2 节介绍如何利用卷积神经网络表示有限体积离散，以及如何利用神经网络构建雅可比迭代和多重网格方法。第 3 节介绍使用神经网络求解器求解反应堆物理特征值问题的三个数值算例，并将结果与标准有限体积法进行比较。最后，第 4 节总结本文的研究发现。

2. 方法论

本节第一部分介绍控制方程、采用有限体积法或控制体积法对其进行离散化，以及求解所得方程组的雅可比方法。随后，我们说明如何利用具有预定义权重的神经网络卷积层来表示这种离散化。为了求解所得方程组，我们将雅可比方法嵌入多重网格方法中。雅可比方法和多重网格方法均在神经网络中实现，后者基于 U-Net架构。最后，概述多能群情况下的求解过程，包括如何确定特征值。

2.1. 扩散方程

用于临界性计算的多群稳态扩散方程可写为:

$$-\nabla\cdot(D_g\nabla\phi_g)+\Sigma_g^a\phi_g+\sum_{\substack{g'=1\\g'\neq g}}^{N_g}\Sigma^s_{g\to g'}\phi_g=\sum_{\substack{g'=1\\g'\neq g}}^{N_g}\Sigma^s_{g'\to g}\phi_{g'}+\lambda\,\chi_g\sum_{g'=1}^{N_g}\nu_{g'}\Sigma^f_{g'}\phi_{g'},\qquad \forall g\in\{1,2,\ldots,N_g\}, \tag{1}$$

其中， ϕ_g 是中子群体的标量通量， Σ_g^a 表示吸收截面， Σ_g^f表示裂变截面，ν_g 是每次裂变事件产生的平均中子数， Σ_g^s表示散射截面，χ_g 是每次裂变事件中每个能群产生的中子所占的比例， N_g 是所使用的能群数。下标 g 表示特定的能群。扩散系数 D_g定义如下:

$$D_g=\frac{1}{3(\Sigma_g^a+\Sigma_g^s)}\,. \tag{2}$$

特征值 λ取为 k_{eff} 的倒数（即 $\lambda=1/k_{\text{eff}}$），其中 :

$$k_{\text{eff}}=\frac{\text{number of neutrons in one generation}}{\text{number of neutrons in the preceding generation}}\,. \tag{3}$$

反射性边界条件以及真空或裸表面边界条件可按如下方式实现 :

$$D_g\,(\boldsymbol{n}\cdot\nabla\phi_g)\;=0 \qquad \textit{(reflective)} \tag{4}$$

$$-D_g\,(\boldsymbol{n}\cdot\nabla\phi_g)=\frac{1}{2}\phi_g \qquad \text{（真空或裸露表面）} \tag{5}$$

其中 $\boldsymbol{n}$ 是边界的外法向量。

2.2. 离散化

二维扩散方程可在由 $N_x \times N_y$ 个单元组成的规则网格上采用有限体积法进行离散，具体如下:

$$
\begin{aligned}
&-\frac{(D_{i-1,j,g}+D_{i,j,g})}{2\Delta x^2}\phi_{i-1,j,g}-\frac{(D_{i,j,g}+D_{i+1,j,g})}{2\Delta x^2}\phi_{i+1,j,g}-\frac{(D_{i,j-1,g}+D_{i,j,g})}{2\Delta y^2}\phi_{i,j-1,g}\\
&-\frac{(D_{i,j,g}+D_{i,j+1,g})}{2\Delta y^2}\phi_{i,j+1,g}+\left(\frac{(D_{i-1,j,g}+2D_{i,j,g}+D_{i+1,j,g})}{2\Delta x^2}+\frac{(D_{i,j-1,g}+2D_{i,j,g}+D_{i,j+1,g})}{2\Delta y^2}\right)\phi_{i,j,g}\\
&+{\Sigma^a}_{i,j,g}\phi_{i,j,g}\ +\sum_{\substack{g'=1\\ g'\neq g}}^{N_g}\Sigma^s_{i,j,g\to i,j,g'}\phi_{i,j,g}=\sum_{g'=1}^{N_g}\Sigma^s_{i,j,g'\to i,j,g}\phi_{i,j,g'}+\lambda\chi_g\sum_{g'=1}^{N_g}\nu_{g'}\Sigma^f_{i,j,g'}\phi_{i,j,g'}, \qquad (6)\\
&\qquad \forall i\in\{2,3,\ldots,N_x-1\}, \qquad \forall j\in\{2,3,\ldots,N_y-1\}, \qquad \forall g\in\{1,2,\ldots,N_g\},
\end{aligned}
$$

其中， Δx 和 Δy 分别是 x 和 y 方向上的均匀单元宽度， N_x 和 N_y 分别是 x 和 y 方向上的单元数，下标 i 和 j 分别表示x 和 y 方向上的单元，$\phi_{i,j,g}$ 表示单元 i, j中能群 g 的标量通量。这种离散化等效于有限差分离散。边界条件施加于 x 和 y 方向上的首末单元，因此不对这些单元求解方程（6）。我们希望以一种方式施加边界条件，避免改变边界附近的离散模板，从而最大限度地提高实现效率。考虑到这一点，可通过以下约束对左边界（$i=1$）施加反射边界条件:

$$\phi_{1,j,g}=0, \quad D_{1,j,g}=-D_{2,j,g} \quad \forall j \qquad (7)$$

右边界（$i=N_x$）则为:

$$\phi_{N_x,j,g}=0, \quad D_{N_x,j,g}=-D_{N_x-1,j,g} \quad \forall j\,. \qquad (8)$$

可根据需要对上、下边界施加类似的约束。这种实现边界条件的方式可确保边界单元与其相邻单元之间的界面处平均扩散率为零，从而避免扩散穿过该界面。对于裸表面边界条件（见公式 (5)），当边界法向与 x 方向对齐时，吸收项修改如下:

$$\Sigma^a_{i,j,g}\leftarrow\Sigma^a_{i,j,g}+\frac{1}{2\Delta x}\,. \qquad (9)$$

对于边界与 y 方向对齐的裸表面边界条件：

$$\Sigma^a_{i,j,g} \leftarrow \Sigma^a_{i,j,g} + \frac{1}{2\Delta y}\,. \tag{10}$$

对于同时具有两种边界条件的单元，作如下修改：

$$\Sigma^a_{i,j,g} \leftarrow \Sigma^a_{i,j,g} + \frac{1}{2\Delta x} + \frac{1}{2\Delta y}\,. \tag{11}$$

对于与方程（6）相关的五点模板，通过一层“光晕单元”来实现边界条件。对于采用更大模板的高阶离散化，需要更多层单元作为光晕单元。

方程（6）及其相关边界条件通常写作：

$$\boldsymbol{A}\boldsymbol{\phi} = \lambda \boldsymbol{B}\boldsymbol{\phi}. \tag{12}$$

其中，矩阵 $\boldsymbol{A}$ 包含吸收、扩散和散射项；矩阵 $\boldsymbol{B}$ 表示裂变项；向量 $\boldsymbol{\phi}$ 包含每个能群中各单元的标量通量值。接下来，我们仍采用迄今为止使用的记法，将每个能群的未知标量通量存储在一个二维数组中。尽管这种问题表述方式可能不太常见，但其动机将在下一节变得清晰：届时我们会比较离散模板与卷积算子。考虑到这一点，我们将方程（6）中的方程组改写为

$$\sum_{u=-l}^{l}\sum_{v=-l}^{l} a^{u,v}_{i,j,g}\,\phi_{i+u,j+v,g} = s_{i,j,g}\,, \quad \forall i \in \{2,3,\ldots,N_x-1\},\ \forall j \in \{2,3,\ldots,N_y-1\},\ \forall g \in \{1,2,\ldots,N_g\}, \tag{13}$$

其中

$$a_{i,j,g}^{u,v} = \begin{cases} -\dfrac{(D_{i,j,g}+D_{i+u,j+v,g})}{2\Delta x^2} & \text{for } |u|=1,\ v=0 \\ -\dfrac{(D_{i,j,g}+D_{i+u,j+v,g})}{2\Delta y^2} & \text{for } u=0,\ |v|=1 \\ \dfrac{D_{i-1,j,g}+2D_{i,j,g}+D_{i+1,j,g}}{2\Delta x^2}+\dfrac{D_{i,j-1,g}+2D_{i,j,g}+D_{i,j+1,g}}{2\Delta y^2}+\Sigma_{i,j,g}^{as} & \text{for } u=0=v \\ 0 & \text{for } |u|=1=|v| \end{cases} \tag{14}$$

$$\Sigma_{i,j,g}^{as} = \Sigma_{i,j,g}^{a}+\sum_{\substack{g'=1 \\ g'\neq g}}^{N_g}\Sigma_{i,j,g\to i,j,g'}^{s} \tag{15}$$

$$s_{i,j,g} = \sum_{g'=1}^{g-1}\Sigma_{i,j,g'\to i,j,g}^{s}\phi_{i,j,g'}^{(k+1)}+\sum_{g'=g}^{N_g}\Sigma_{i,j,g'\to i,j,g}^{s}\phi_{i,j,g'}^{(k)}+\lambda\chi_g\sum_{g'=1}^{N_g}\nu_{g'}\Sigma_{i,j,g'}^{f}\phi_{i,j,g'}^{(k)}\ . \tag{16}$$

由于方程（6）中使用的模板是五点模板（也可以等价地写成 3×3 模板），因此公式（13）中的 l 值为 1。公式（13）的右侧可以通过对 $\phi_{i,\ j,\ g}$作出“最佳猜测”来确定。这样便对公式（13）进行了线性化，现在可以用雅可比方法求解：

$$\phi_{i,j,g}^{(k+1)} = \frac{1}{a_{i,j,g}^{0,0}}\left(s_{i,j,g}-\sum_{u=-l}^{l}\sum_{v=-l}^{l}a_{i,j,g}^{u,v}\,\phi_{i+u,j+v,g}^{(k)}+a_{i,j,g}^{0,0}\phi_{i,j,g}^{(k)}\right), \tag{17}$$

其中， $2l+1$ 是模板的宽度， k 是雅可比迭代， $\{\{a_{i,j,g}^{u,v}\}_{u=-l}^{l}\}_{v=-l}^{l}$ 表示用于计算单元 i, j中标量通量的模板系数。雅可比方法可用于对角占优系统；给定初始猜测后，依次求解每个对角分量。迭代持续进行，直到方程组收敛 [40]。公式（17）的常规矩阵向量形式（见公式（12））中的对角项现在记为 $a^{0,}$ $_{i}{}^{0}{}_{,j,g}$（对应单元 i, j 和能群 g）；其余项为非对角项（$a^{u,i}{}^{v}{}_{,j,g}$ $\forall u, v$，满足 $|u|=1$ 和 $v=0$，以及 $u=0$ 和$|v|=1$），这些项将从公式（17）中的源项中减去。

2.3. 使用卷积神经网络实现离散化

神经网络的卷积层配有一个滤波器或卷积核，它是一个小型网格（二维滤波器通常为 3×3, 5×5 或 7×7），其中各单元都具有称为权重的数值。滤波器通过将输入值与重叠单元中的权重相乘，作用于输入的一部分。将乘积相加即可得到输出。图1展示了这一过程，其中可以看到滤波器作用于输入数据的一部分。滤波器在输入数据的各个部分上移动，这一过程会不断重复，直到滤波器遍历所有输入数据，并且所有

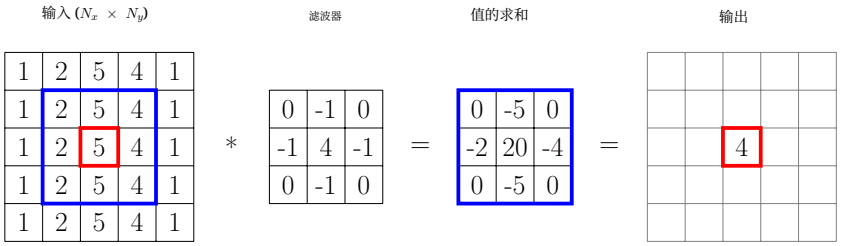


图1：一个 3×3 卷积滤波器，它将二维离散扩散算子（五点有限体积模板）应用于 9 个单元。首先将该滤波器应用于左侧蓝色方块中的所有单元；其结果如等号后方的蓝色方块所示。然后将这 9 个值相加，得到右侧红色方块中的值，即输出值。该值表示在中心单元处近似的输入上作用扩散算子所得的值。

输出值均已确定。二维卷积层作用于二维输入的过程可表示如下

$$x_{i,j}^{(k+1)} = \sum_{u=-l}^{l} \sum_{v=-l}^{l} w^{u,v}\, x_{i+u,j+v}^{(k)}, \tag{18}$$

其中，输入和输出分别是具有分量 $x_i^{(k)}{}_{,\,j}$ 和 $x_i^{(k+1)}{}_{,\,j}$ 的二维网格。滤波器的权重表示为 $w^{u,\,v}$，滤波器的尺寸为 $(2l+1)\times(2l+1)$。图1中的示例对应于在 $\Delta x = 1 = \Delta y$ 的情况下，将离散扩散算子应用于输入，且常数扩散率 $(\boldsymbol{D}_g)$ 为 1。对于一般网格尺寸，采用方程（18）中的记号，对于一组特定权重 $\boldsymbol{w}$，将离散扩散算子应用于能群 g和单元 $i,\ j$ 的标量通量，可写为

$$-\nabla^2 \boldsymbol{\Phi}_g\Big|_{i,\,j} = \sum_{u=-l}^{l} \sum_{v=-l}^{l} w^{u,v} \phi_{i+u,j+v,g} = -\left(\frac{\phi_{i-1,j,g} - 2\phi_{i,j,g} + \phi_{i+1,j,g}}{\Delta x^2} + \frac{\phi_{i,j-1,g} - 2\phi_{i,j,g} + \phi_{i,j+1,g}}{\Delta y^2} \right), \tag{19}$$

这等价于

$$\sum_{u=-l}^{l} \sum_{v=-l}^{l} w^{u,v} \phi_{i+u,j+v,g} = \sum_{\text{entries}} \begin{bmatrix} 0 & \frac{-1}{\Delta y^2} & 0 \\ \frac{-1}{\Delta x^2} & \frac{2}{\Delta x^2} + \frac{2}{\Delta y^2} & \frac{-1}{\Delta x^2} \\ 0 & \frac{-1}{\Delta y^2} & 0 \end{bmatrix} \odot \begin{bmatrix} \phi_{i-1,j+1,g} & \phi_{i,j+1,g} & \phi_{i+1,j+1,g} \\ \phi_{i-1,j,g} & \phi_{i,j,g} & \phi_{i+1,j,g} \\ \phi_{i-1,j-1,g} & \phi_{i,j-1,g} & \phi_{i+1,j-1,g} \end{bmatrix} =: \boldsymbol{f}(\boldsymbol{\Phi}_g; \boldsymbol{w})\Big|_{i,j}. \tag{20}$$

其中，$l = 1$，$\odot$ 表示 Hadamard 积，即逐元素乘法；符号$\sum_{\text{entries}}$ 表示矩阵所有元素的总和（见公式 (B.2)）；$\boldsymbol{f}$ 表示对场 $\boldsymbol{\Phi}_g$ 应用权重为 $\boldsymbol{w}$的 3×3 滤波器所进行的离散卷积。公式 (20) 中方括号内的分量，排列方式与图像中的像素相同，而非矩阵中的分量。公式 (19) 展示了扩散算子作用于场 $\boldsymbol{\Phi}_g$ 时的一种有限体积离散写法；公式 (20) 则是完全相同的离散化写法，只不过表示为卷积（使用 Hadamard 积）。这说明离散格式可以表示为卷积

神经网络。

通过比较公式（13）和公式（18），我们可以看到，扩散方程（具有空间变化的$\boldsymbol{D}_g$）目前还不能写成卷积层，因为方程（13）中的权重随空间变化（对于两个单元 $i,\ j$和 $i^*,\ j^*$, $a^{u,\ i}{}^{v}{}_{,\ j} \neq a^{u}{}_{,\ i^*}{}^{v}{}_{,\ j^*}$），而在方程（18）中，权重与其作用于输入数据的哪一部分无关（即 $w^{u,\ v}$ 独立于 $i,\ j$）。回想方程 (1) 中的扩散算子可以写成三个都涉及拉普拉斯算子的项（见附录 A），因此我们可以将扩散算子写成三个卷积:

$$-\nabla\cdot(D_g\nabla\phi_g) = \frac{1}{2}\left(-\nabla^2(D_g\phi_g) - D_g\nabla^2\phi_g + \phi_g\nabla^2 D_g\right) \qquad \textit{analytical form} \qquad (21)$$

$$\boldsymbol{f}^{\text{Diff}}(\boldsymbol{\Phi}_g, \boldsymbol{D}_g; \boldsymbol{w}) = \frac{1}{2}\left(\boldsymbol{f}(\boldsymbol{D}_g \odot \boldsymbol{\Phi}_g; \boldsymbol{w}) + \boldsymbol{D}_g \odot \boldsymbol{f}(\boldsymbol{\Phi}_g; \boldsymbol{w}) - \boldsymbol{\Phi}_g \odot \boldsymbol{f}(\boldsymbol{D}_g; \boldsymbol{w})\right) \qquad \textit{discretised form} \qquad (22)$$

其中，$\boldsymbol{\Phi}_g$是一个包含所有 $\phi_{i,\ j,\ g}$分量的 $N_x \times N_y$矩阵，$\boldsymbol{D}_g$是一个包含所有 $D_{i,\ j,\ g}$分量的 $N_x \times N_y$矩阵，而 $\boldsymbol{f}$ 表示应用权重为 $\boldsymbol{w}$的卷积层。方程（22）给出了扩散卷积 $\boldsymbol{f}^{\text{Diff}}$ 和权重 $\boldsymbol{w}$的定义。这种扩散算子有限体积离散形式与方程（6）中所示标准形式之间的等价性见附录 B。现在，能群 g 的离散扩散方程可写为:

$$\boldsymbol{f}^{\text{Diff}}(\boldsymbol{\Phi}_g, \boldsymbol{D}_g; \boldsymbol{w}) + \left(\boldsymbol{\Sigma}_g^a \ + \sum_{\substack{g'=1 \\ g'\neq g}}^{N_g} \boldsymbol{\Sigma}^s_{g\to g'}\right) \odot \boldsymbol{\Phi}_g = \boldsymbol{s}_g, \quad \forall g \in \{1, 2, \ldots, N_g\}, \qquad (23)$$

其中，能群 g 的源项 $\boldsymbol{s}_g$ 还包含除 g 以外其他能群之间的耦合项。$\boldsymbol{\Sigma}_g^a$ 和 $\boldsymbol{\Sigma}^s_{g\to g'}$ 表示包含每个单元的吸收截面和散射截面的矩阵。与之前一样，方程 (23) 可以用雅可比方法求解。不过，为了提高效率，在实现时，我们不使用方程 (23)，而是对其进行改写，以减少一次卷积运算。项 $\sum_{u=-l}^{l}\sum_{v=-l}^{l} a^{u,\ i}{}^{v}{}_{,\ j,\ g}\,\phi^{(k)}_{i+u,\ j+v,\ g} - a^{0}{}_{,\ i}{}^{0}{}_{,\ j,\ g}\,\phi^{(k)}_{i\,j\ ,\ g}$ 可以通过仅包含非对角项的卷积滤波器确定:

$$\sum_{u=-l}^{l}\sum_{v=-l}^{l} a^{u,v}_{i,j,g}\,\phi^{(k)}_{i+u,j+v,g} - a^{0,0}_{i,j,g}\,\phi^{(k)}_{i,j,g} \equiv \frac{1}{2}\left(\boldsymbol{D}_g \odot \boldsymbol{f}(\boldsymbol{\Phi}_g^{(k)}; \boldsymbol{w}_{\text{od}})\,|_{i,j,g} + \boldsymbol{f}(\boldsymbol{D}_g \odot \boldsymbol{\Phi}_g^{(k)}; \boldsymbol{w}_{\text{od}})\right)\Big|_{i,j,g}, \qquad (24)$$

其中， $\boldsymbol{f}$ 是一个权重为 $\boldsymbol{w}_{\mathrm{od}}$ 的卷积层：

$$\boldsymbol{w}_{\mathrm{od}} = \begin{bmatrix} 0 & \frac{-1}{\Delta y^2} & 0 \\ \frac{-1}{\Delta x^2} & 0 & \frac{-1}{\Delta x^2} \\ 0 & \frac{-1}{\Delta y^2} & 0 \end{bmatrix}. \tag{25}$$

其中将 $\boldsymbol{w}$ 的中心项设为零，以得到该结果。因此，方程（17）所写的雅可比方法等价于：

$$\boldsymbol{\Phi}_g^{(k+1)} = (\boldsymbol{A}_g^{0,0})^{\odot -1} \odot \left(\boldsymbol{s}_g - \frac{1}{2} \left(\boldsymbol{D}_g \odot \boldsymbol{f}(\boldsymbol{\Phi}_g^{(k)}; \boldsymbol{w}_{\mathrm{od}}) + \boldsymbol{f}(\boldsymbol{D}_g \odot \boldsymbol{\Phi}_g^{(k)}; \boldsymbol{w}_{\mathrm{od}}) \right) \right), \tag{26}$$

其中（$\boldsymbol{A}^{0,\ g^0}$）$^{\odot -1}$ 是哈达玛逆 [41]，它是一个 $N_x \times N_y$ 数组，其 i^{th}， j^{th} 分量对于能群 g为$a^{0^1,\ i^0,\ j,\ g_0}$ 该方程可以表示为函数 J,

$$\boldsymbol{\Phi}_g^{(k+1)} = \mathrm{J}\left(\boldsymbol{\Phi}_g^{(k)}, (\boldsymbol{A}_g^{0,0})^{\odot -1}, \boldsymbol{s}_g, \boldsymbol{D}_g \right), \tag{27}$$

该函数计算一次雅可比迭代后的更新解。图 2 展示了该函数的架构，即用于求解方程（26）离散化后的中子输运问题的一次雅可比迭代的神经网络。绿色方框包含输入；蓝色方框是卷积层；橙色方框是作为层的数学函数；灰色方框是网络输出。每个方框中的第二行给出了该方框的输出维度。

到目前为止，我们已经介绍了如何求出卷积层滤波器的权重，这些滤波器对应于采用雅可比方法求解的中子扩散方程的有限体积离散。不过，本文所述方法并不限于有限体积法。作为比较，我们还采用了一种基于新型卷积有限元法（ConvFEM） [23] 的扩散算子离散化。使用二次九节点矩形单元时，该离散化对应的 5×5 滤波器为

$$\boldsymbol{w} = \frac{1}{900} \begin{bmatrix} -5 & 50 & -15 & 50 & -5 \\ 50 & -320 & -660 & -320 & 50 \\ -15 & -660 & 3600 & -660 & -15 \\ 50 & -320 & -660 & -320 & 50 \\ -5 & 50 & -15 & 50 & -5 \end{bmatrix} \tag{28}$$

$$D_{\mathrm{left}} = D_{\mathrm{right}} = D_{\mathrm{top}} = D_{\mathrm{bottom}}, \tag{29}$$

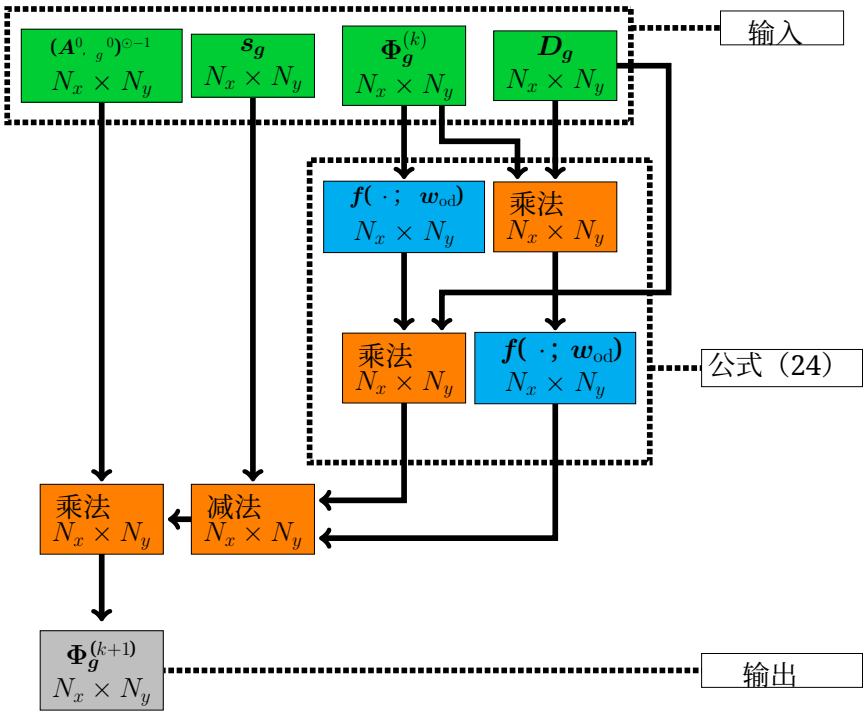


图 2：用于执行单次雅可比迭代的神经网络示意图，该迭代在公式 (26) 中记作 J(·)。该网络对单个能群的通量执行一次雅可比迭代。输入包括通量（$\boldsymbol{\Phi}_g^{(k)}$）、代表性源项（$\boldsymbol{s}_g$）、扩散系数（$\boldsymbol{D}_g$）和严格对角系数（$\boldsymbol{A}^{\mathbf{0},\ \boldsymbol{g}\ \mathbf{0}})^{\odot-1}$）（绿色方框）。网络执行若干层操作，数学运算以橙色显示，卷积运算以青色显示。输出是下一次雅可比迭代的通量（$\boldsymbol{\Phi}_g^{(k+1)}$）。箭头从数据来源的层引出，箭头末端表示接收该数据作为输入的层。层的维度标注在每个方框的第二行。

左侧为:

$$D_{\text{left}} = D_{i,j,g} \quad \forall i \in \{3,4\} \quad \forall j \in \{3,4,\ldots,N_y-2\}, \tag{30}$$

并对右侧、顶部和底部施加相应的约束。如果方程（29）成立，且方程（30）对所有边均成立，则左侧的边界条件 $i = 1$ 可通过以下方式实现:

$$\phi_{1,j,g} = 0, \quad \phi_{2,j,g} = 0, \quad D_{\text{left}} = -D_{1,j,g} = -D_{2,j,g}, \tag{31}$$

其他各边也有类似条件。如果边界旁的两个单元取值不同，方程（30）可能不成立，因此这种情况下可能无法采用此方法，也可以使用某种平均值。另一种适用于所有滤波器尺寸的方法是，直接将晕区中的扩散系数和通量设为零，这样就无需为边界条件调整吸收截面。这实际上实现了使用方程（9）、（10）和（11）得到的 $2\Delta x$ 外推边界条件。

2.4. 多重网格

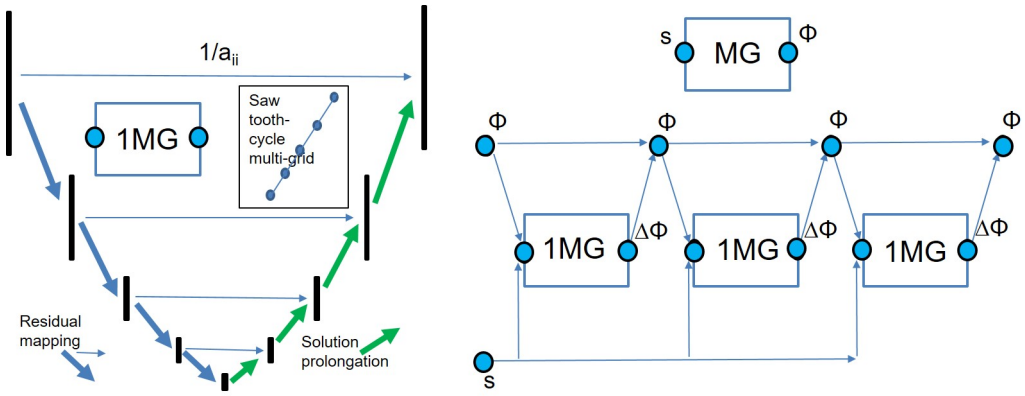


图3：示意图展示了用于构成一个多重网格锯齿循环的 U-Net架构（左），右侧则展示了如何将多个循环组合起来，形成整体求解方法。

图3展示了如何重新设计 U-Net [24]架构，使其构成锯齿多重网格方法。图4展示了使用 U-Net 并进行两次限制的一次多重网格迭代。请注意

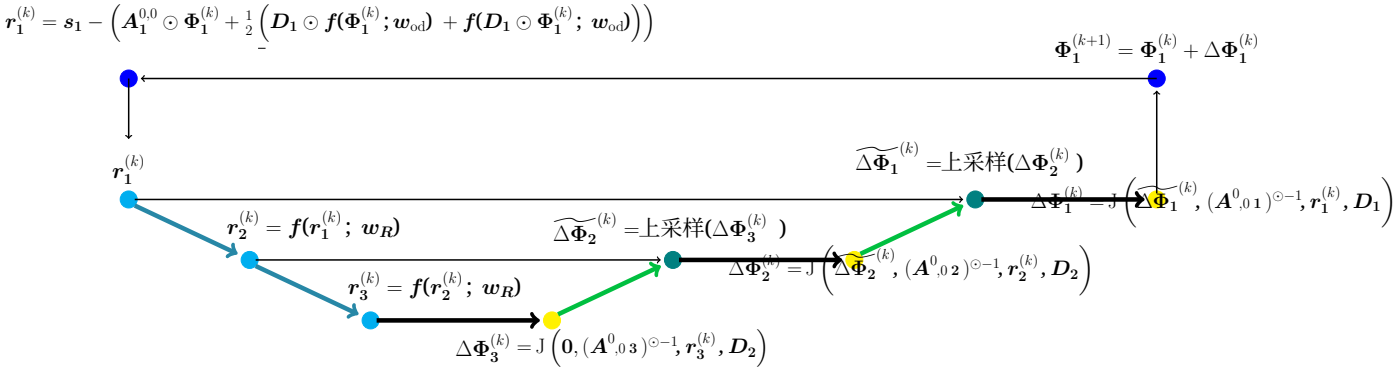


图4：多重网格迭代，其中下标表示分辨率，上标表示多重网格迭代次数，J(·)表示一次雅可比迭代。计算残差并进行两次限制操作，由青色节点表示。这些残差与雅可比平滑结合使用，由黄色节点表示。平滑后，使用上采样层执行延拓，由蓝绿色节点表示。达到最细层级后，更新通量，并重复该过程。

表示能群的下标不再显示，粗体下标现在表示网格的粗糙度，其中 1 表示最精细的网格。残差（$\boldsymbol{r^k}$）按以下方式计算：

$$\boldsymbol{r}_1^{(k)} = \boldsymbol{s}_1 - \left(\boldsymbol{A}_1^{0,0} \odot \boldsymbol{\Phi}_1^{(k)} + \frac{1}{2}\left(\boldsymbol{D}_1 \odot \boldsymbol{f}(\boldsymbol{\Phi}_1^{(k)}; \boldsymbol{w}_{\mathrm{od}}) + \boldsymbol{f}(\boldsymbol{D}_1 \odot \boldsymbol{\Phi}_1^{(k)}; \boldsymbol{w}_{\mathrm{od}})\right)\right) \tag{32}$$

然后将其限制两次，得到 $(\boldsymbol{r}_2^{(k)})$ 和 $(\boldsymbol{r}_3^{(k)})$。在最粗层级（粗体下标 3）上执行一次雅可比迭代，以确定 $\boldsymbol{\Delta\Phi_3}^{(k)}$，初始值为零数组。随后将其延拓，以估计$\widetilde{\boldsymbol{\Delta\Phi_2}}^{(k)}$，再使用更高一层级的残差进行另一次雅可比迭代，将其平滑为 $\boldsymbol{\Delta\Phi_2}^{(k)}$。此过程不断重复，直至达到最细层级（粗体下标 1），此时更新通量（$k+1$），并重复该过程若干次多重网格迭代。限制操作可以通过

卷积来执行：

$$\boldsymbol{r}_{\boldsymbol{2}}^{(k)} = \boldsymbol{f}(\boldsymbol{r}_{\boldsymbol{1}}^{(k)}; \boldsymbol{w}_{\boldsymbol{R}}), \tag{33}$$

滤波器权重：

$$\boldsymbol{w}_{\boldsymbol{R}} = \begin{bmatrix} 0.25 & 0.25 \\ 0.25 & 0.25 \end{bmatrix}. \tag{34}$$

上采样层可以起到将解延拓到更高层级的作用。上采样操作只需将较粗网格单元中的值复制到较细网格上对应的单元中，从而增加数据的维度 [42]，并得到较细网格上数据的近似：

$$\widetilde{\boldsymbol{\Phi}_{\boldsymbol{1}}}^{(k)} = \mathrm{UpSamp}(\boldsymbol{\Phi}_{\boldsymbol{2}}^{(k)}). \tag{35}$$

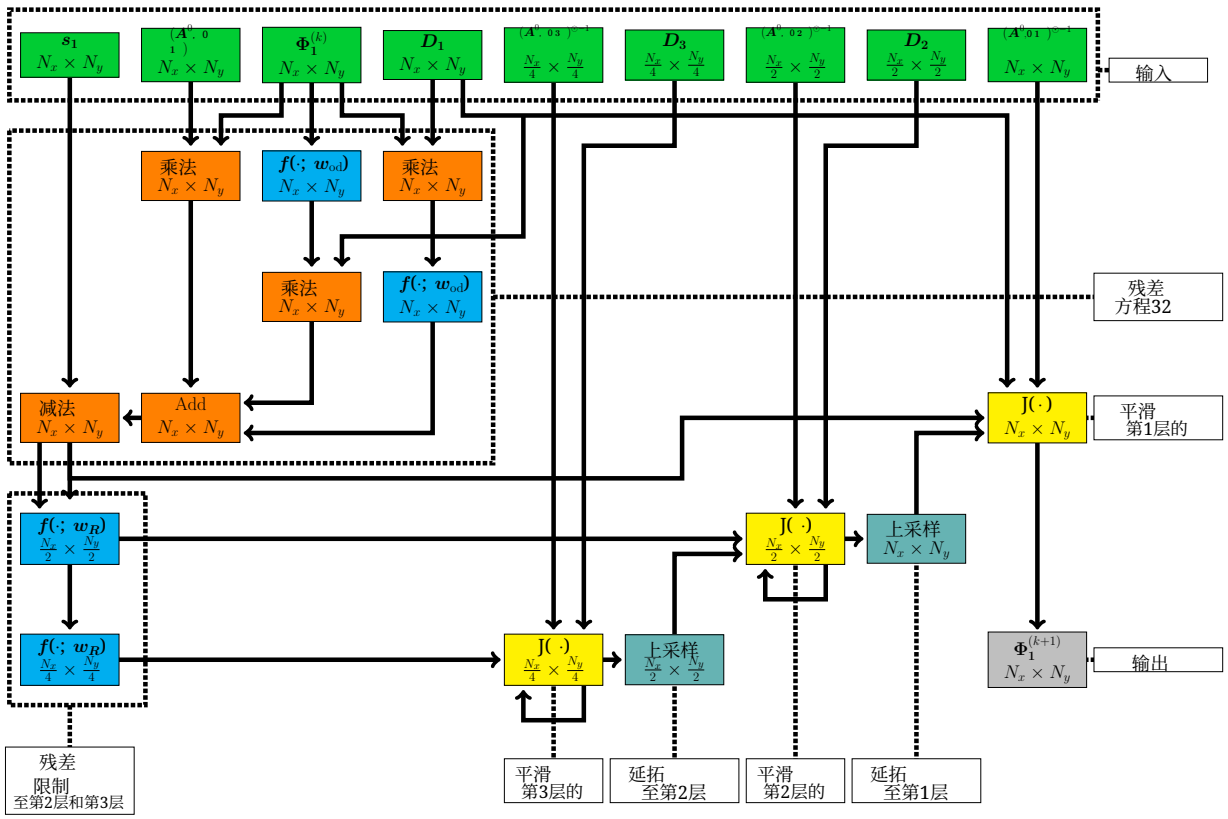


图5：多重网格网络 MG(·)，表示一次多重网格迭代。该网络对单个能群的通量执行一次多重网格迭代。它将通量（$\boldsymbol{\Phi}_{\boldsymbol{1}}^{(k)}$）、代表性源项（$\boldsymbol{s}_{\boldsymbol{1}}$）、扩散系数（$\boldsymbol{D}_{\boldsymbol{1}}$）和严格对角系数（$\boldsymbol{A}^{0}{}_{0\,1}$），以及较粗分辨率系数作为输入（绿色方框）。网络执行多种层操作：数学运算以橙色表示，卷积过程以青色表示，子模型操作以黄色表示，上采样以青绿色表示。子模型可以多次迭代。最后，它输出下一次多重网格迭代的通量（$\boldsymbol{\Phi}_{\boldsymbol{1}}^{(k+1)}$）。箭头起点显示数据源自哪个层，箭头末端显示哪个层将该数据作为输入。层的维度标注在每个方框的第二行。

图5展示了如何将多重网格方法表示为神经网络。绿色方框中包含

输入，蓝色方框表示卷积层，橙色方框表示作为层的数学函数，黄色方框表示子网络，青绿色方框表示上采样层，灰色方框表示网络输出。每个方框中的第二行表示输出维度。可写为:

$$\boldsymbol{\Phi}_{\mathbf{1}}^{(k+1)} = \mathrm{MG}\left(\boldsymbol{\Phi}_{\mathbf{1}}^{(k)}, \boldsymbol{s}_{\mathbf{1}}, (\boldsymbol{A}_{\mathbf{1}}^{0,0}), (\boldsymbol{A}_{\mathbf{1}}^{0,0})^{\odot -1}, (\boldsymbol{A}_{\mathbf{2}}^{0,0})^{\odot -1}, (\boldsymbol{A}_{\mathbf{3}}^{0,0})^{\odot -1}, \boldsymbol{D}_{\mathbf{1}}, \boldsymbol{D}_{\mathbf{2}}, \boldsymbol{D}_{\mathbf{3}}\right), \tag{36}$$

对于单个能群 g:

$$\boldsymbol{\Phi}_{\mathbf{1}g}^{(k+1)} = \mathrm{MG}_g\left(\boldsymbol{\Phi}_{\mathbf{1}g}^{(k)}, \boldsymbol{s}_{\mathbf{1}g}, (\boldsymbol{A}_{\mathbf{1}g}^{0,0}), (\boldsymbol{A}_{\mathbf{1}g}^{0,0})^{\odot -1}, (\boldsymbol{A}_{\mathbf{2}\boldsymbol{g}}^{0,0})^{\odot -1}, (\boldsymbol{A}_{\mathbf{3}g}^{0,0})^{\odot -1}, \boldsymbol{D}_{\mathbf{1}g}, \boldsymbol{D}_{\mathbf{2}g}, \boldsymbol{D}_{\mathbf{3}g}\right), \tag{37}$$

其中，MG(·) 是一个用于计算一次锯齿多重网格迭代结果的函数；将多次此类迭代串联起来即可得到最终解，见图3。MG$_g$(·) 表示应用于能群 g 的多重网格迭代，下标用于标示该能群。

在粗网格上形成离散化之前，应注意，扩散系数和其他材料属性会通过调和平均映射到粗网格上。每个多重网格层级使用相同的离散化，但单元尺寸不同。

2.5. 多群网络

方程（36）和图5分别给出的多重网格函数和网络，展示了如何将一次多重网格迭代应用于单个能群 g。多群问题必须使散射项达到平衡，这可通过不断迭代直至各项平衡来实现。方程（16）展示了在构造 $s_{i,\,j,\,g}$ 时如何使用块高斯-赛德尔方法。散射项 Σ^s 使用最新的通量信息构造，这通过依次求解各个能群来实现。

图 6 展示了如何在神经网络中使用块高斯-赛德尔方法求解能群。绿色方框表示输入，黄色方框表示子网络，灰色方框表示输出。为清晰起见，图中仅显示绿色输出与第一个能群相连，但实际上它们会与后续所有能群相连。 $\boldsymbol{s}_g$ 是一个包含 Σ^s、 $\boldsymbol{s}_{\mathrm{fiss}}$ 和 $\boldsymbol{\Phi}^{(k)}$ 的向量，并根据方程（16）构成。随后使用 $\boldsymbol{s}_g$ 在多重网格子模型中求解 $\boldsymbol{\Phi}_g$，并重复进行多重网格迭代，直到:

$$\boldsymbol{\Phi}_g^{(k+1)} \approx \boldsymbol{\Phi}_g^{(k)}. \tag{38}$$

$\boldsymbol{\Phi}_g$ 随后用于$\boldsymbol{s}_{g'}$ ，其中 $g' > g$。所有能群求解完毕后，可将它们拼接起来形成 $\boldsymbol{\Phi}^{(k+1)}$。重复此过程，直到:

$$\boldsymbol{\Phi}^{(k+1)} \approx \boldsymbol{\Phi}^{(k)}. \tag{39}$$

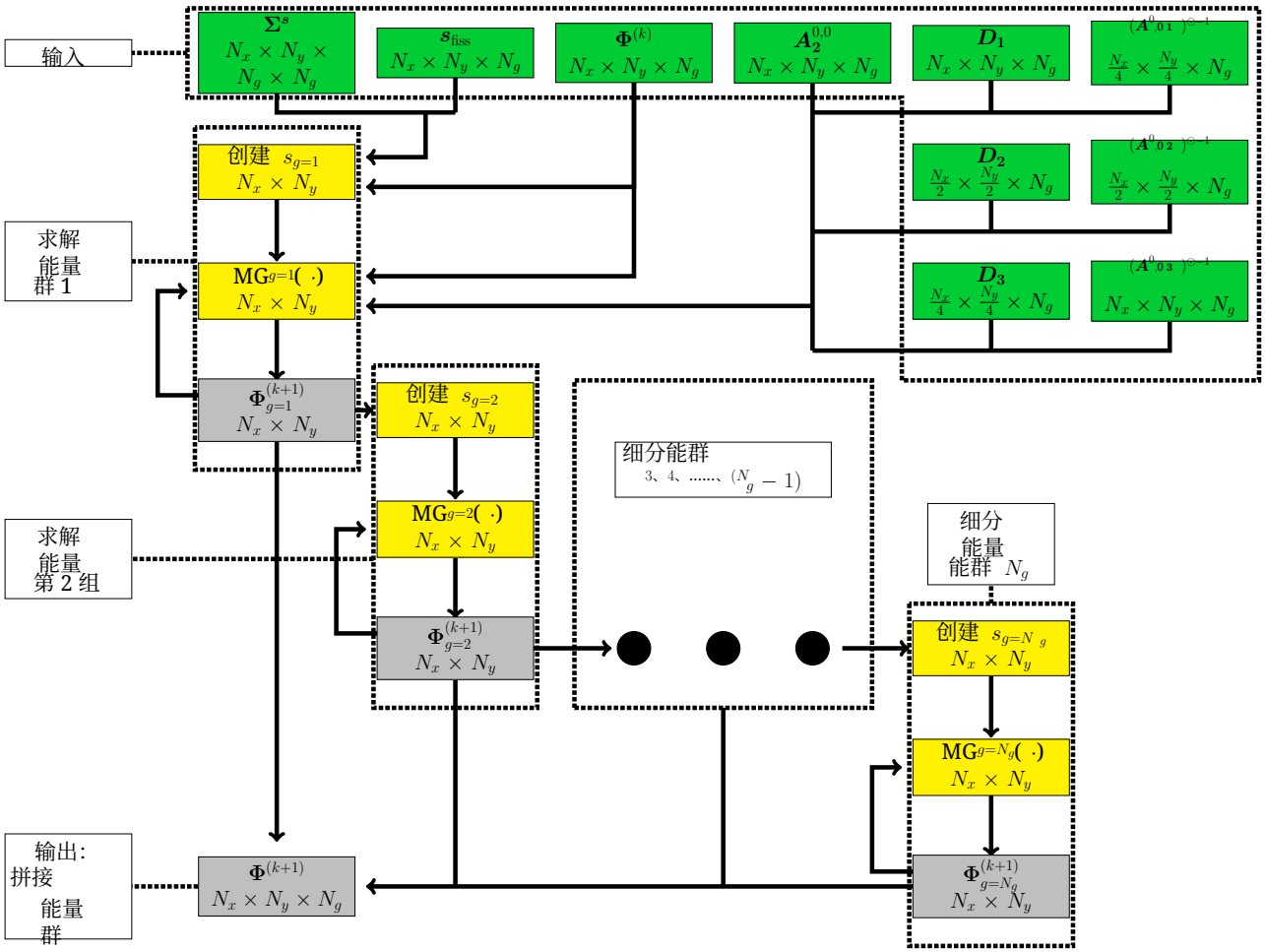


图 6：表示单次多群迭代的多群网络。该网络对所有能群的通量执行一次多群迭代。它将通量（$\boldsymbol{\Phi}_1^{(k)}$）、散射截面 Σ_s、裂变源项（$\boldsymbol{s}_{\text{fiss}}$）、扩散系数（$\boldsymbol{D}_1$）和严格对角系数（$\boldsymbol{A}^0{}_{,01}$），以及较粗分辨率系数，作为输入（绿色方框）。每个能群依次更新：先更新特定能群的源项，再执行若干次多重网格子模型迭代。某个能群更新后的通量随后传递给后续能群。为清晰起见，图中省略了连接关系；所有输入（绿色方框）都连接到后续子模型（黄色方框）。最后，网络输出下一次多群迭代的通量（$\boldsymbol{\Phi}_1^{(k+1)}$）。箭头起点表示数据源自哪一层，箭头末端表示哪一层将该数据作为输入。各层的维度标注在每个方框的第二行。

高斯-赛德尔方法的另一种替代方案是使用雅可比方法同时求解所有能群，这可以仅通过多重网格网络来实现，如第 2.4 节所述。具体做法是同时将所有 N_g 能群传入 MG 网络，仅在此步骤之外更新 $\boldsymbol{s}$。方程（16）中的源项则改为：

$$s_{i,j,g} = \sum_{g'=1}^{N_g} \Sigma^s_{i,j,g' \to i,j,g} \phi^{(k)}_{i,j,g'} + \lambda \chi_g \sum_{g'=1}^{N_g} \nu_{g'} \Sigma^f_{i,j,g'} \phi^{(k)}_{i,j,g'} . \tag{40}$$

公式（12）是一个特征值问题，因此需要确定 λ。通常采用近似

$\lambda = 1$），并将裂变项传递给多群网络，其中

$$\boldsymbol{s}_{\text{fiss},g} = \lambda \chi_g \sum_{g'=1}^{N_g} \nu_{g'} \boldsymbol{\Sigma}_{g'}^{\boldsymbol{f}} \boldsymbol{\Phi}_{g'}^{(k)}, \tag{41}$$

对于每个能群， g 和 $\boldsymbol{s}_{\text{fiss}}$ 是一个数组，其中包含$\boldsymbol{s}_{\text{fiss}}$的全部 g ， g。幂法 [43]是本文选用来确定该问题主特征值的方法。本文使用的幂法实现与 [37] 相同，并在多群网络之外运行。

3. 结果

本文所述方法在两个测试案例上进行了展示：一个燃料组件和一个反应堆堆芯，二者均基于 KAIST 基准算例 [44]。对于燃料组件，研究了两种构型（控制棒完全拔出和完全插入）。二维中子扩散方程有限体积离散的结果由具有预先确定权重的神经网络生成，并与传统 Fortran 实现的结果进行比较。此外，还给出了基于二次有限元法的离散化的神经网络解，即 ConvFEM [23], 。对于反应堆堆芯，截面取自 KAIST 基准算例，并使用 3×3 个燃料组件组成堆芯的四分之一。结果来自采用神经网络的中子扩散方程有限体积离散。���本节中的所有神经网络均使用 Keras [42] 在以 TensorFlow 为后端的 Python 中实现 [1]。

3.1. 燃料组件——几何结构与配置

基于 KAIST 基准算例 [44]的 UOX燃料组件几何结构如图 7 所示。它由一个 17×17晶格组成，其中包含 264 根 UOX 燃料棒，其余 25 个晶格单元中设有导向管，这些单元可填充慢化剂或控制棒。我们考虑该组件的两种构型。第一种构型中，这 25 个晶格单元全部填充慢化剂，代表控制棒完全拔出的系统。第二种构型中，其余 25 个晶格单元全部为控制棒，代表控制棒完全插入的系统。我们使用两种计算网格，每个晶格单元内分别包含 20×20 个单元或 10×10 个单元。较高分辨率的网格用于燃料组件测试案例，粗网格用于对整个反应堆建模（见第 3.6 节）。对于 20×20案例，晶格中共有 115， 600 个计算单元，其中 1， 364 个构成边界（即作为光晕单元或虚单元）。能量离散为七个群，这意味着该燃料组件具有 818,720 个自由度。燃料组件每一侧的长度为

21.42 cm，因此每个计算单元的尺寸为 0.063 cm $\times$ 0.063cm。燃料组件每一侧均施加真空边界条件。

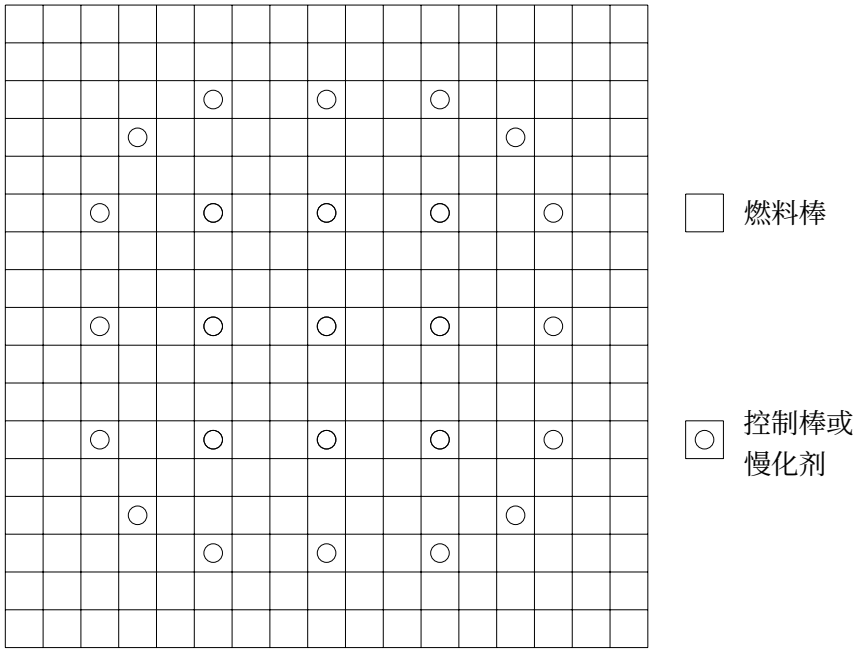


图 7：具有晶格单元的 UOX燃料组件几何结构，其中晶格单元包含燃料棒或导向管，导向管内填充慢化剂或控制棒。

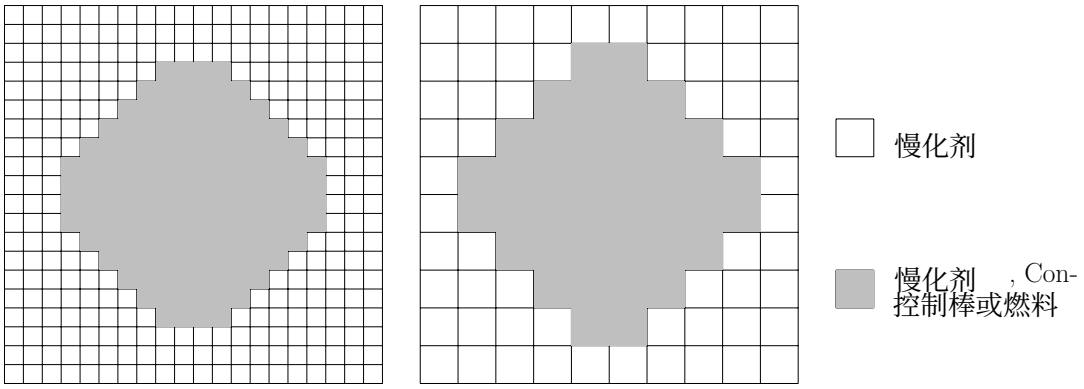


图 8：此处显示了单个晶格单元的计算网格，包括精细网格（20×20，用于第 3.2 节的组件计算）和粗略网格（10×10，用于第 3.6 节的反应堆堆芯计算）。

燃料组件中的所有晶格单元几何形状相同，每个晶格单元的外部区域均由慢化剂占据，内部区域则由燃料、控制棒或慢化剂占据。如图 8 所示。模型未对导向管进行建模。所需的材料参数包括燃料棒的 UOX 截面，以及控制棒和慢化剂（与冷却剂相同）的截面，均取自 KAIST 基准算例。

3.2. 燃料组件 - 有限体积离散

针对一个二维燃料组件求解中子扩散方程，该组件的几何结构和截面取自 KAIST 基准算例 [44]。我们执行两次雅可比迭代、100 次多重网格迭代和 100 次多群迭代，以获得由权重预先通过有限体积离散化确定的多群神经网络的解。最后一次多群迭代后，解收敛至有效容差 10^{-14}。我们将神经网络得到的解与采用高斯-赛德尔迭代法（容差为 10^{-15}）的 Fortran 有限体积离散化传统实现所得的解进行比较。

图9展示了控制棒完全拔出时燃料组件中三个能群的通量分布。第7群标量通量的高值表明了导向管内慢化剂的位置。可以观察到，神经网络解与采用高斯-赛德尔迭代的 Fortran 解之间的逐点差异很小，$\mathcal{O}(10^{-10})$，且在求解器设定的容差范围内。

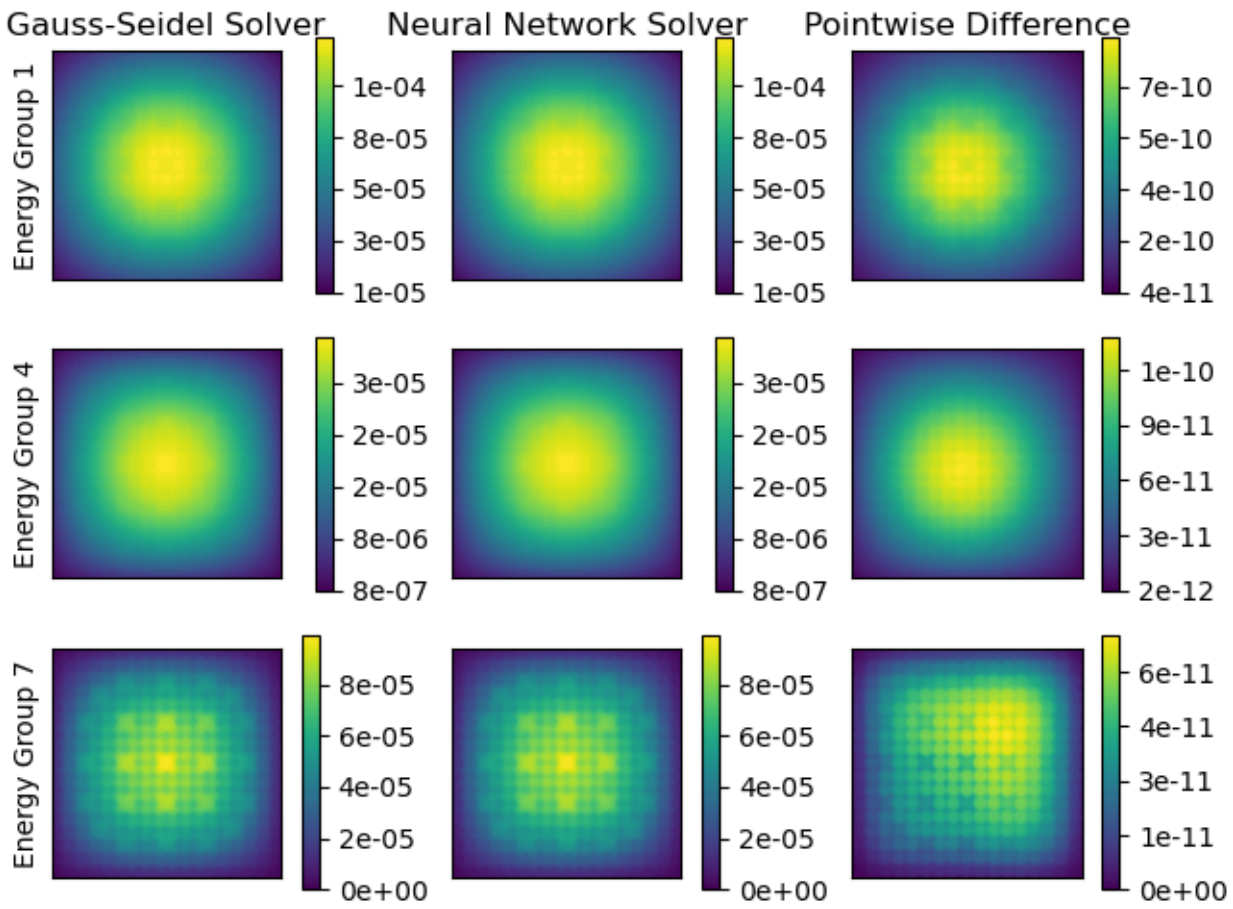


图 9：使用多群网络生成的控制棒完全拔出的燃料组件中三个能群的标量通量（中子 $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$）。

图10展示了控制棒完全插入时燃料组件中三个能群的通量分布。可以观察到，控制棒位于燃料棒之间，此处通量急剧下降。可以观察到，神经网络解与 Fortran 解之间的逐点差异很小， $\mathcal{O}(10^{-10})$ ，且在求解器设定的容差范围内。

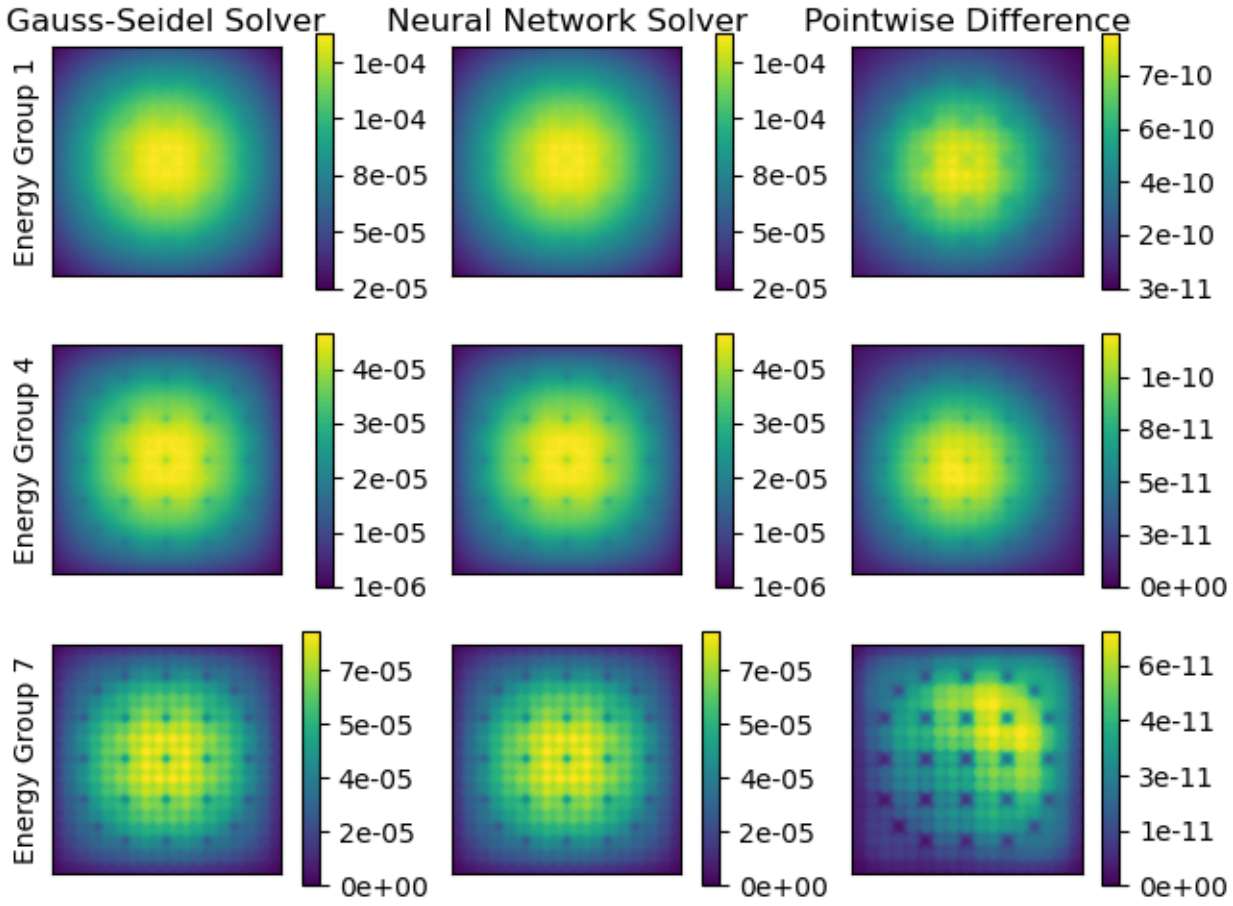
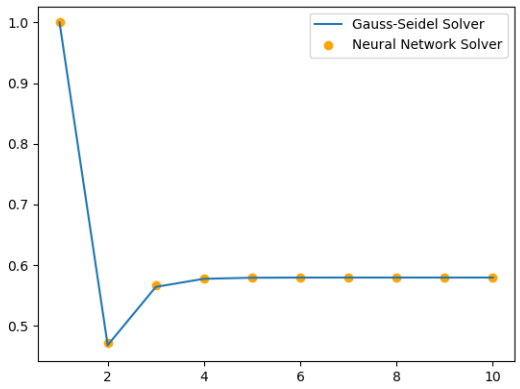
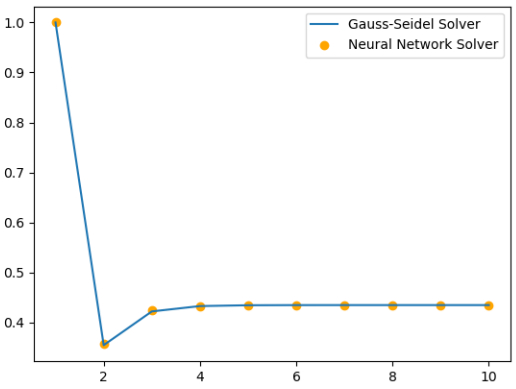


图10：使用多群网络生成的控制棒完全插入时燃料组件中三个能群的标量通量（中子 $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$）。

图11给出了控制棒完全拔出的燃料组件（图11(a)）和完全插入的燃料组件（图11(b)）的 k_{eff} 收敛速率。可以观察到，插入控制棒时， k_{eff} 较低，这符合预期。对于两种构型（控制棒完全拔出和完全插入），神经网络求解器所得解与 Fortran 实现所得解的收敛情况相同。 k_{eff}的收敛值对比见表1。



(a) 控制棒完全拔出。 k_{eff} 神经网络求解器收敛至 0.5797，Fortran 实现（使用高斯-赛德尔求解器）收敛至 0.5797。

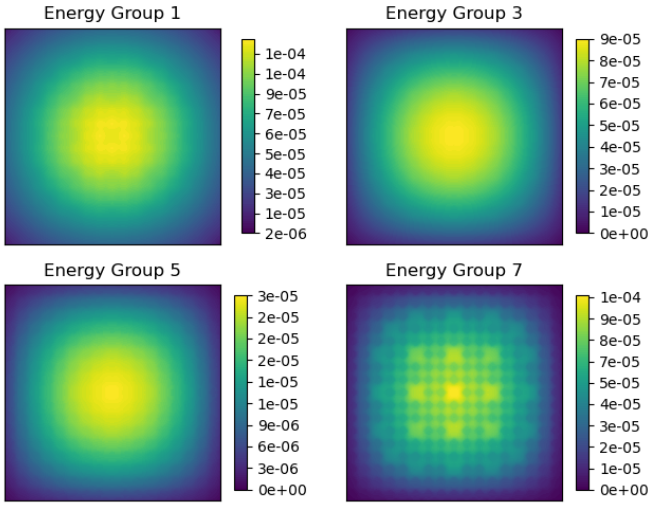


(b) 控制棒完全插入。 k_{eff} 神经网络求解器收敛至 0.4347，Fortran 实现（使用高斯-赛德尔求解器）收敛至 0.4347。

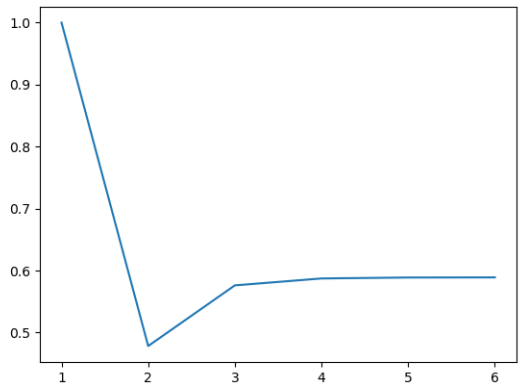
图11：使用多群网络和有限体积离散化生成的燃料组件幂迭代收敛曲线。 k_{eff}

3.3. 燃料组件 - 有限元离散

图12和图13分别给出了控制棒完全拔出和完全插入的燃料组件的标量通量解。两种解均采用神经网络实现的二次卷积有限元（ConvFEM）生成。滤波器中使用的权重见公式（28）。在两种情况下，通量分布均与第3.2节中使用有限体积离散生成的解呈现相似的分布。使用二次有限元得到的 k_{eff} 收敛值均略大于有限体积离散得到的值，见表1。

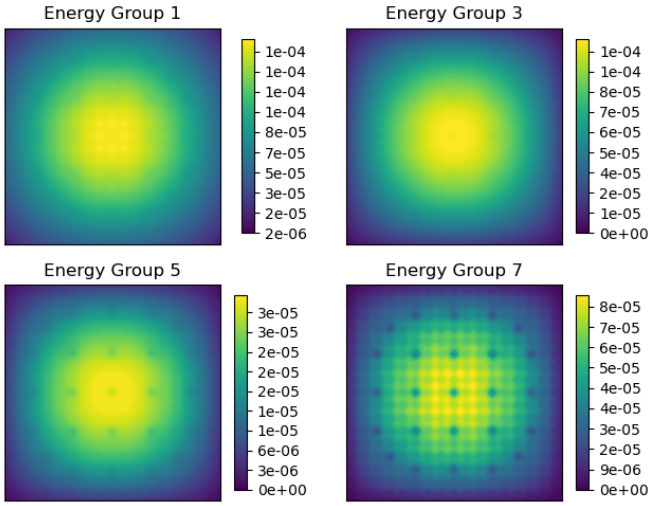


(a) 燃料组件中四个能群的标量通量（中子 $\text{cm}^{-2}\,\text{s}^{-1}$）能群 .

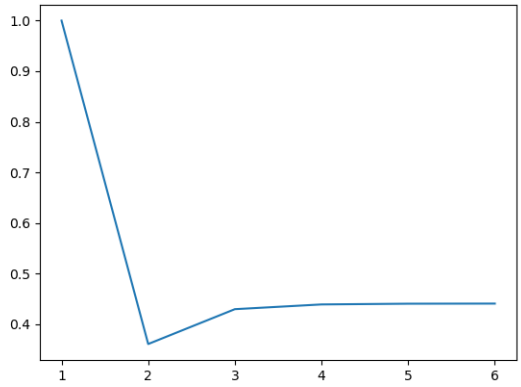


（b） k_{eff} 相对于幂迭代的收敛性。 k_{eff} 收敛至 0.5838，而有限体积离散的结果为 0.5797。

图12：使用二次卷积有限元的多群网络（采用 ConvFEM）生成的控制棒完全拔出时的燃料组件结果。



(a) 燃料组件中四个能群的标量通量（中子 $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$）。



(b) k_{eff} 与幂迭代的对比。 k_{eff} 收敛至 0.4370，而有限体积离散的结果为 0.4347。

图13：控制棒完全插入时燃料组件的结果，使用二次卷积有限元的多群网络（借助 ConvFEM）生成。

离散化	实现	求解器	拔出	插入
有限体积	神经网络	带有雅可比迭代的多重网格	0.5797	0.4347
有限体积	Fortran	高斯-赛德尔	0.5797	0.4347
有限元	神经网络	带有雅可比迭代的多重网格	0.5838	0.4370

表1：燃料组件测试案例的 k_{eff} 值

3.4. 燃料组件 - 时间比较

表 2 展示了在燃料组件测试案例上执行 100 次雅可比迭代的时间比较。使用图形处理器的神经网络实现采用了多群网络（见图 6），但将多重网格网络（见图5）替换为雅可比网络（见图 2）。使用中央处理器，以 Fortran 串行执行了等效操作。可以看出，神经网络求解器的平均耗时不到 Fortran 求解器的三分之一。神经网络求解器的耗时也更加稳定，最短与最长耗时相差 0.0636 秒。Fortran 求解器的耗时波动则大得多，最短与最长耗时相差 1.3400 秒。

实现	硬件	最长时间（秒）	最短时间（秒）	平均时间（秒）
神经网络	NVIDIA RTX 6000 图形处理器	1.3412	1.2776	1.2819
Fortran	AMD EPYC 7742 中央处理器	5.2568	3.9168	4.3681

表 2：使用图形处理器运行神经网络求解器、使用中央处理器运行 Fortran 代码，在燃料组件测试案例上执行 100 次雅可比迭代的时间比较。共执行了 400 组、每组 100 次迭代，因此下文列出了这些运行时间的最大值、最小值和平均值。

3.5. 反应堆堆芯——几何结构与配置

现在，我们使用 KAIST 基准算例的截面来模拟反应堆堆芯 [44]。与该基准算例不同，我们的堆芯是由 UOX 型燃料组件构成的 3×3 网格。我们对计算域的四分之一进行建模，并采用反射边界条件来代表堆芯的其余部分，见图 14。燃料组件周围的反射层使用慢化剂材料。堆芯外部边界采用真空边界条件。反射层和每个燃料组件的宽度均为 21.42 cm，因此图 14 所示计算域的每条边长为 85.68 cm。每个组件的晶格单元都具有由 10×10 单元构成的计算网格（见图 8）。整个计算域内的网格均匀，因此反射层包含 202,300 个单元，全部九个燃料组件共包含 260,100 个单元，另有 $2,724$ 个单元用作应用边界条件所需的光晕单元。能量再次离散为七个群，得到 $3{,}236{,}800$ 个自由度。

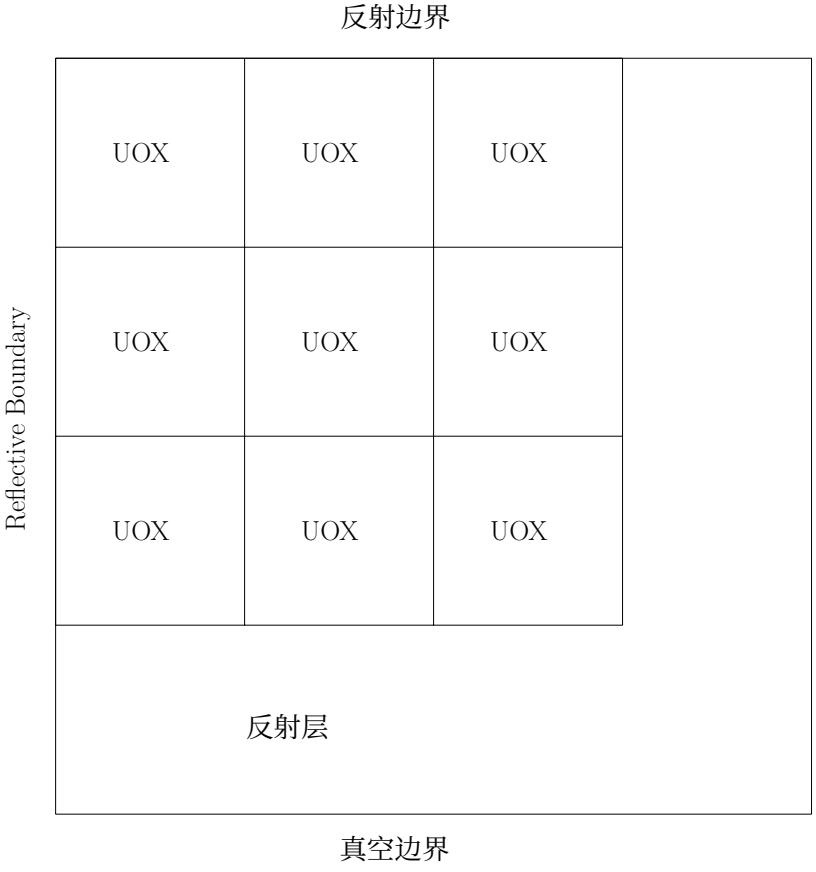


图14：KAIST 基准算例简化版本的反应堆堆芯几何结构 [44]。

每个燃料组件的控制棒可以完全插入或完全拔出。本研究考察的堆芯两种构型如图15所示。构型一有五个燃料组件的控制棒完全拔出，四个燃料组件的控制棒完全插入。构型二有六个燃料组件的控制棒完全拔出，三个燃料组件的控制棒完全插入

控制棒。这些构型是随机选取的。

插入	拔出	拔出
拔出	插入	拔出
拔出	插入	插入

（a）反应堆配置一。

拔出	拔出	拔出
拔出	插入	拔出
插入	插入	拔出

(b) 反应堆配置二。

图15：反应堆堆芯构型，其中“拔出”表示控制棒已完全拔出，“插入”表示控制棒已完全插入。

3.6. 反应堆堆芯 - 有限体积离散化

采用权重由有限体积离散确定的神经网络求解二维中子扩散方程，并得到了上一节所述反应堆堆芯的解。本节所有解均进行了 5 次雅可比迭代、100 次多重网格迭代和 100 次多群迭代。图16展示了反应堆配置一中四个能群的通量分布。可以观察到，控制棒完全拔出的区域通量较高；而在左上角控制棒插入的区域，所有能群的通量均显著下降。最低能群（第 7 群）的通量分布清楚地显示了控制棒和慢化剂（位于导向管内）的位置，通量分别在这些位置急剧下降或上升。

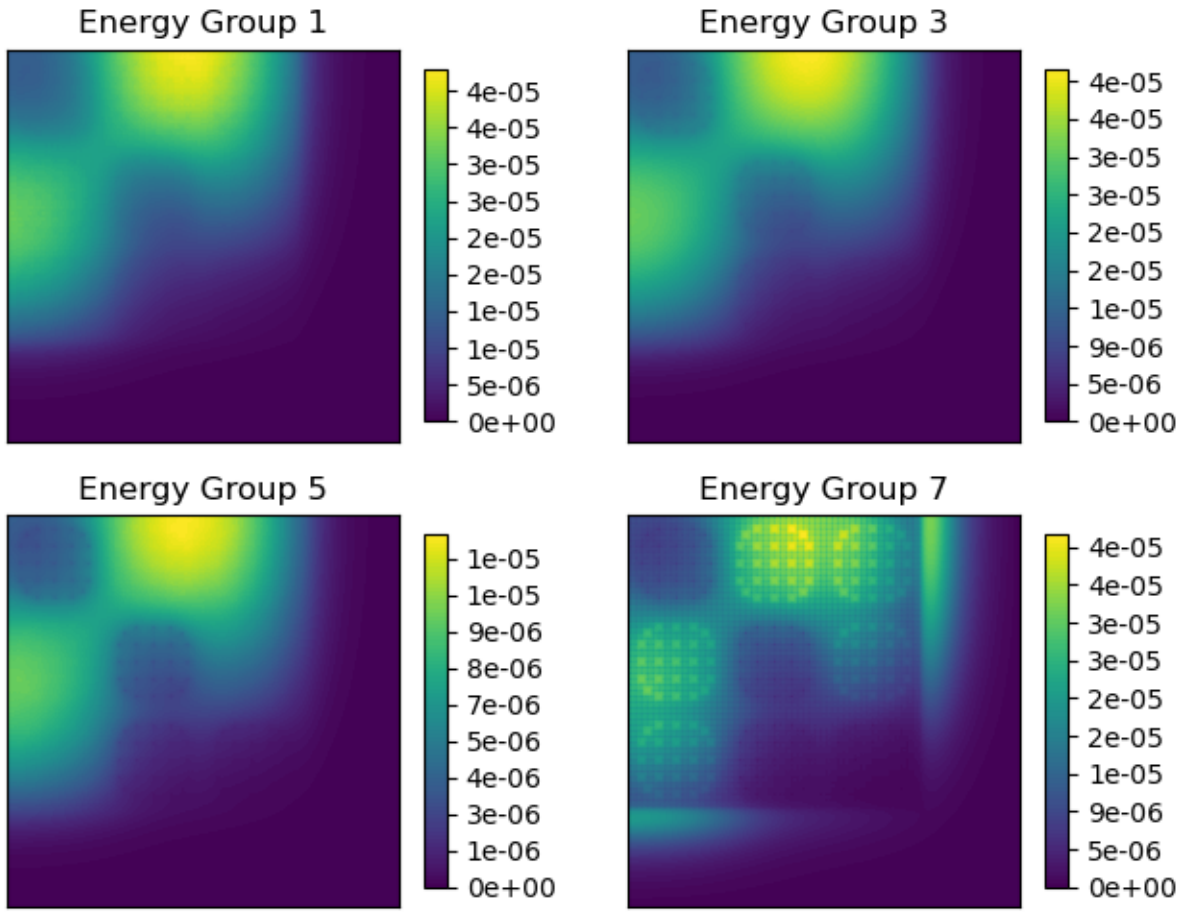


图16：反应堆配置一中，使用多群网络和有限体积离散化生成的四个能群下燃料组件的标量通量（中子 $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$）。

图17展示了反应堆配置二中四个能群的通量分布。同样，控制棒插入处的通量会急剧下降，而最高通量值出现在靠近反射边界的左上角。最低能群（第 7 群）的通量分布清楚地显示了导向管内控制棒和慢化剂的位置。

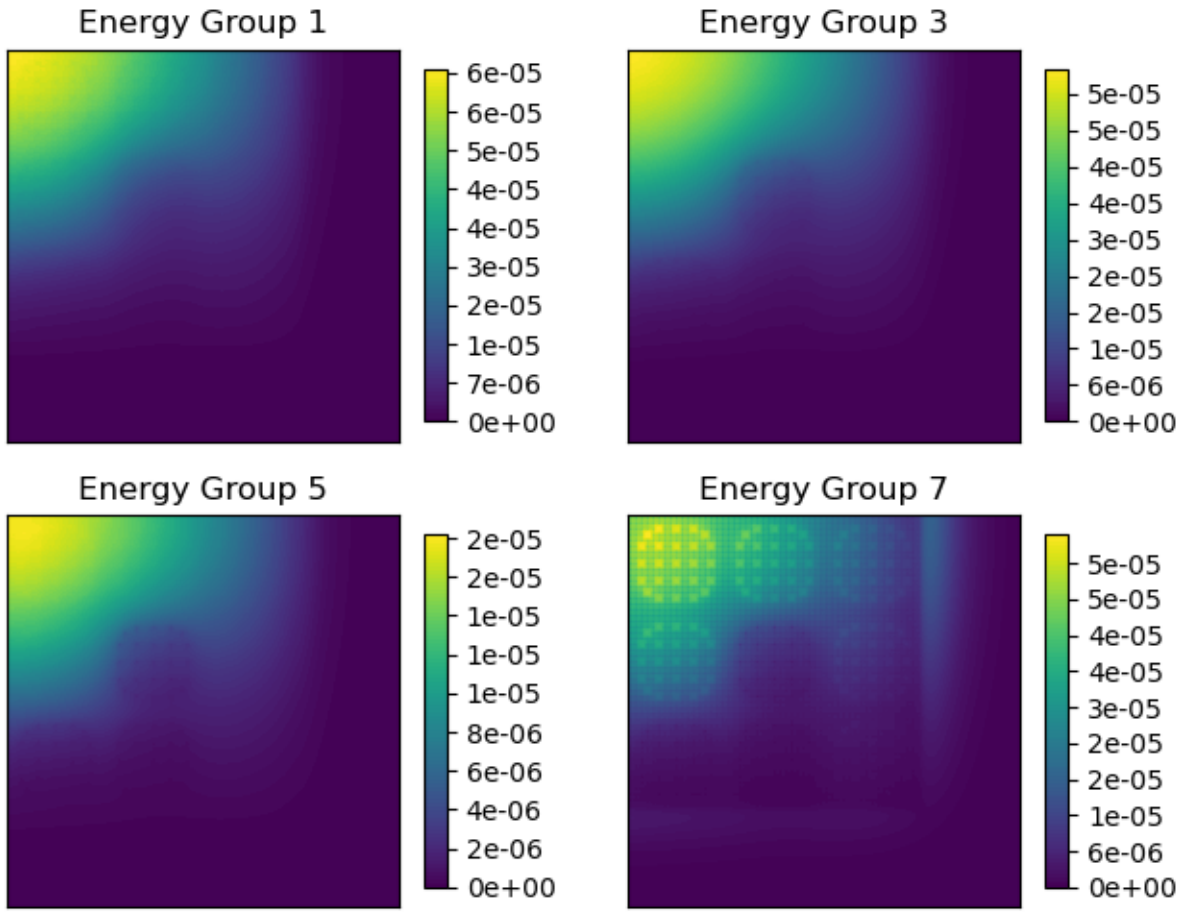
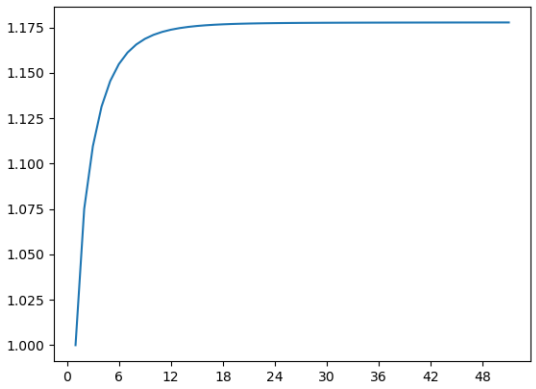
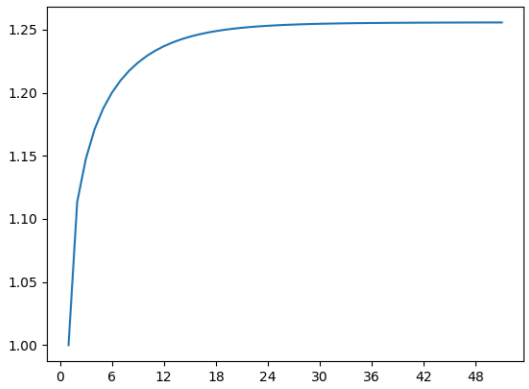


图17：反应堆配置二中，采用多群网络进行有限体积离散化得到的燃料组件内四个能群的标量通量（中子 $\mathrm{cm}^{-2}\,\mathrm{s}^{-1}$）。

图18展示了两种反应堆构型下 k_{eff} 的收敛情况。可以观察到，构型二的 k_{eff} 略高于构型一，这是意料之中的，因为构型二插入的控制棒较少。



(a) k_{eff} 与反应堆配置一的幂迭代对比，收敛至 1.1777。



(b) k_{eff} 与反应堆配置二的幂迭代对比，收敛至 1.2557。

图18：采用多群网络和有限体积离散化计算两种构型的反应堆堆芯 k_{eff} 与幂迭代对比。

4. 结论与未来工作

本文提出了一种新方法，利用人工智能（AI）软件库中的工具，复现通过标准数值方法格式离散化的偏微分方程的求解过程。尽管该方法普遍适用于偏微分方程（PDEs），本文重点关注核反应堆物理领域，并求解由扩散理论描述的中子输运所产生的特征值问题。此外，尽管底层离散化方法可以任意选择，本文的演示重点在于使用卷积神经网络复现有限体积法的求解过程。这里采用的方法不是训练网络，而是定义卷积神经网络的权重，以精确复现离散化过程。迭代求解器也在网络中得到复现。本文研究了一种基于U-Net架构、内部采用雅可比迭代的锯齿多重网格方法。随后，将多重网格网络与另一个跨所有能群运行的网络结合使用，后者充当多群求解器。

本文采用两个测试案例来展示该方法：一个燃料组件和一个反应堆堆芯。对于燃料组件测试案例，将神经网络解与采用高斯-赛德尔方法求解、并以标准方式用 Fortran 实现的相同有限体积离散解进行比较。两个解之间的绝对逐点误差为 $\mathcal{O}(10^{-10})$。燃料组件测试案例表明，该方法能够得到与标准方法相同的解（考虑求解器容差），并且对于 k_{eff}，能够达到相同的收敛速率。该测试案例还用于展示如何在卷积

层中使用二次有限元离散。该方法还扩展应用于计算量更大的问题，即反应堆堆芯。

采用这种方法的一个好处是，可以充分利用人工智能库及其内置技术。例如，这些库的执行已针对不同的计算机架构进行了优化，无论是中央处理器、图形处理器，还是新一代人工智能处理器。这种灵活性让用户能够轻松地在多个平台上运行代码，而无需修改代码。另一个好处是简化了代码开发，因为人工智能库将与平台相关的代码抽象出来，用户只需专注于编程任务。除了充分利用人工智能库已有的重大进展外，将数值离散表示为神经网络中的卷积层，也意味着这些代码已可在最新的人工智能处理器上运行。

未来的工作将包括把幂特征值迭代纳入神经网络。这样，神经网络便能够利用自身的反向传播算法，自动计算特征值对材料属性的灵敏度。一个重要的后续步骤是进一步优化代码和方法（例如，考虑由最粗网格导致的多重网格瓶颈），以便能够在图形处理器或新型人工智能计算机上运行大型问题。

CRediT 作者贡献声明

TRFP: 方法学、软件、写作（初稿撰写、审阅与编辑）。**CEH:** 方法学、写作（初稿撰写、审阅与编辑）、指导。**BC:** 软件、写作（审阅与编辑）。**AGB:**软件、写作（初稿撰写、审阅与编辑）。
CCP: 概念化、方法学、软件、写作（初稿撰写、审阅与编辑）、指导、经费获取。

致谢

作者谨此感谢以下英国工程与自然科学研究委员会（EPSRC）资助项目：RELIANT，COVID-19快速智能风险评估工具（EP/V036777/1）；CO-TRACE，COVID-19传播风险评估案例研究——教育机构（EP/W001411/1）；INHALE，跨生物学尺度的健康评估（EP/T003189/1）；PREMIERE 计划资助项目（EP/T000414/1）；MAGIC（EP/N010221/1）；以及 MUFFINS（EP/P033180/1）。

参考文献

[1] M. Abadi、P. Agarwal、Aand Barham、E. Brevdo、Z. Chen、C. Citro、G. S. Corrado、A. Davis、J.Dean、M. Devin、S. Ghemawat、I. Goodfellow、A. Harp、G. Irving、M. Isard、Y. Jia、R. Jozefowicz、L. Kaiser、

M. Kudlur、J. Levenberg、D. Man´e、R. Monga、S. Moore、D. Murray、C. Olah、M. Schuster、J. Shlens、B. Steiner、I. Sutskever、K. Talwar、P. Tucker、V. Vanhoucke、V. Vasudevan、F. Vi´egas、O. Vinyals、P. Warden、M. Wattenberg、M. Wicke、Y. Yu、X. Zheng，《TensorFlow：异构系统上的大规模机器学习》，2015年。软件可从 www.tensorflow.org 获取。

[2] A. Paszke, S. Gross, F. Massa, A.Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N.Gimelshein, L. Antiga, A. Desmaison, A. Kopf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chil- amkurthy, B. Steiner, L. Fang, J. Bai, S. Chintala，《PyTorch：一种命令式风格的高性能深度学习库》，载于：神经信息处理系统进展第32卷，Curran Associates, Inc.，2019年，第8024–8035页。网址：http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf。

[3] B. Vermeire, F. Witherden, P. Vincent，《GPU加速高阶方法在非定常流动模拟中的效用：与行业标准工具的比较》，《计算物理学杂志》334（2017年）497–521。

[4]J. Chan, Z. Wang, A. Modave, J.-F. Remacle, T. Warburton，混合网格上的 GPU 加速间断伽辽金方法，《计算物理学杂志》318（2016年）142–168。

[5] R. M. Bergmann, K. L. Rowland, N. Radnovi´c, R. N. Slaybaugh, J. L. Vuji´c，使用 WARP（一种面向图形处理器上通用三维几何结构的连续能量蒙特卡洛中子输运框架）进行临界计算的性能与精度，《核能年鉴》103（2017年）334–349。

[6] R. N. Slaybaugh, M. Ramirez-Zweiger, T. Pandya, S. Hamilton, T. M. Evans，领导级计算机上核反应堆建模的特征值求解器，《核科学与工程》190（2018年）31–44。

[7] C. Cecka、A. J. Lew、E. Darve，《图形处理器上的有限元方法组装》，《国际工程数值方法杂志》85（2011年）640–669。

[8] F. Mossaiby、R. Rossi、P. Dadvand、S. Idelsohn，《基于 OpenCL 在图形硬件上实现非结构边基有限元对流-扩散求解器》，《国际工程数值方法杂志》89（2012）1635–1651。

[9] A. Dziekonski, P. Sypek, A. Lamecki, M. Mrozowski，《在多个图形处理器上生成大型有限元矩阵》，《国际工程数值方法杂志》94 (2013) 204–220。

[10] S. Sanfui, D. Sharma, 《一种面向非结构网格的三阶段图形处理器有限元分析矩阵生成策略》, 《国际工程数值方法杂志》121 (2020) 3824–3848。[11] A. Modave, A. Atle, J. Chan, T. Warburton, 《一种具有高阶吸收边界条件和角点/边缘兼容性的 GPU 加速节点间断伽辽金方法》, 《国际工程数值方法杂志》112 (2017) 1659–1686。

[12] A. G. M. Lewis、J. Beall、M. Ganahl、M. Hauru、S. B. Mallick、G. Vidal，借助张量处理单元进行大规模分布式线性代数，《美国国家科学院院刊》119（2022年）e2122762119。

[13] Graphcore，智能处理单元，https://www.graphcore.ai/products/ipu，2022年。访问日期：2022年12月16日。

[14] Cerebras，CS-2：人工智能基础设施的一场革命，https://www.cerebras.net/product-system/，2022年。访问日期：2022年10月12日。

[15] T. Lu、T. Marin、Y. Zhuo、Y.-F. Chen、C. Ma，在张量处理单元上加速磁共振成像重建，载于：2020 IEEE高性能极限计算会议（HPEC），2020年，第1–9页。doi:10.1109/HPEC43674.2020.9286192。

[16] T. Lu、T. Marin、Y. Zhuo、Y.-F. Chen、C. Ma，《张量处理单元上的非均匀快速傅里叶变换》，载于：2021年第18届IEEE生物医学成像国际研讨会（ISBI），2021年，第783–787页。doi:10.1109/ISBI48211.2021.9434068。

[17] F. Belletti、D. King、K. Yang、R. Nelet、Y. Shafi、Y.-F. Shen、J. Anderson，《用于金融蒙特卡罗的张量处理单元》，载于：2020年SIAM科学计算并行处理会议论文集，2020年，第12–23页。doi:10.1137/1.9781611976137.2。

[18] A. Morningstar、M. Hauru、J. Beall、M. Ganahl、A. G. Lewis、V. Khemani、G. Vidal，《利用张量处理单元模拟量子多体动力学：Floquet预热化》，PRX Quantum 3 (2022年) 020331。

[19] R. Pederson、J. Kozlowski、R. Song、J. Beall、M. Ganahl、M. Hauru、A. G. M. Lewis、S. B. Mallick、V. Blum、G. Vidal，《作为量子化学超级计算机的张量处理单元》，arXiv预印本（2022年）2202.01255。[20] X.-Z. Zhao、T.-W. Xu、Z.-T. Ye、W.-J. Liu，基于TensorFlow的用于计算流体力学的新型高性能计算框架，《水动力学研究与进展》32（2020年）735–746。

[21] Q. Wang、M. Ihme、Y.-F. Chen、J. Anderson，用于在张量处理单元上模拟流体流动科学计算的TensorFlow框架，《计算机物理通讯》274（2022年）108292。[22] B. Chen、C. E. Heaney、C. C. Pain，使用人工智能库进行不可压缩计算流体力学研究，准备中（2023年）。[23] T. R. Phillips、C. E. Heaney、B. Chen、A. G. Buchan、C. C. Pain，使用神经网络求解离散玻尔兹曼输运方程：中子输运中的应用，准备中（2023年）。[24] O. Ronneberger、P. Fischer、T. Brox，U-Net：用于生物医学图像分割的卷积网络，载于：《医学图像计算与计算机辅助干预》（MICCAI），计算机科学讲义第9351卷，Springer，2015年，第234–241页。doi:10.48550/arXiv.1505.04597。[25] T.-W. Ke、M. Maire、S. X. Yu，多重网格神经网络架构，载于：2017年IEEE计算机视觉与模式识别会议（CVPR），2017年，第4067–4075页。doi:10.1109/CVPR.2017. 433。

[26] J. He, J. Xu，《MgNet：多重网格与卷积神经网络的统一框架》，《中国科学：数学》62（2019年）1331–1354。

[27] N. Thuerey, K. Weißenow, L. Prantl, X. Hu，《翼型流动雷诺平均纳维-斯托克斯模拟的深度学习方法》，《AIAA期刊》58（2020年）25–36。

[28] Q. T. Le、C. Ooi，采用受多重网格启发的神经网络架构对流体动力学进行代理建模，《机器学习及其应用》6（2021年）100176。[29] N. Margenberg、D. Hartmann、C. Lessig、T. Richter，纳维–斯托克斯方程的神经网络多重网格求解器，《计算物理学杂志》460（2022年）110983。[30] E. Cervi、X. Lu、A. Cammi、F. Di Maio、E. Zio，面向熔盐堆控制的灵敏度分析驱动的代理模型，《核工程杂志》3（2022年）277–294。[31] H. D. Kabir、A. Khosravi、M. A. Hosen、S. Nahavandi，基于神经网络的不确定性量化：方法与应用综述，《IEEE开放获取期刊》6（2018年）36218–36234。[32] R. K. Tripathy、I. Bilionis，Deep UQ：学习用于高维不确定性量化的深度神经网络代理模型，《计算物理学杂志》375（2018年）565–588。

[33] H. Gong、S. Cheng、Z. Chen、Q. Li，数据赋能的物理信息机器学习用于降阶建模数字孪生：在核反应堆物理中的应用，《核科学与工程》196（2022年）668–693。

[34] E. Schiassi, M. De Florio, B. D. Ganapol, P. Picca, R. Furfaro，应用于核反应堆动力学点堆动力学方程的物理信息神经网络，《核能年鉴》167（2022年）108833。

[35] J. Wang, X. Peng, Z. Chen, B. Zhou, Y. Zhou, N. Zhou，基于施加边界条件的守恒型物理信息神经网络的中子扩散问题代理建模，《核能年鉴》176（2022年）109234。

[36] B. Foad, R. Elzohery, D. R. Novog，展示结合降阶模型与深度神经网络对时变反应堆瞬态进行仿真的方法，《核能年鉴》171（2022年）109017。

[37] T. R. F. Phillips, C. E. Heaney, P. N. Smith, C. C. Pain，应用于中子扩散的特征值问题基于自编码器的降阶模型，《国际工程数值方法杂志》122（2021年）3780–3811。

[38] S. Qin, Q. Zhang, J. Zhang, L. Liang, Q. Zhao, H. Wu, L. Cao，深度神经网络在生成共振自屏蔽截面中的应用，《核能年鉴》149（2020年）107785。[39] A. Beck, M. Kurz，机器学习方法在湍流建模中的应用展望，GAMM-Mitteilungen 44（2021年）e202100002。[40] F. S. Acton，《通常有效的数值方法》，美国数学协会，华盛顿特区，1990年。[41] R. Reams，近似半定矩阵的 Hadamard 逆、平方根与乘积，《线性代数及其应用》288（1999年）35–43。[42] F. Chollet 等，Keras，2015年。https://keras.io。[43] G. H. Golub, C. F. Loan，《矩阵计算》，约翰·霍普金斯大学出版社，1996年。[44] Z. Cho，KAIST 基准问题 2A：MOX 燃料装载小型压水堆堆芯，http://nurapt.kaist.ac.kr/基准测试/KAIST_ben1a.pdf，2000。

附录 A. 扩散算子

对于两个标量 ϕ_g 和 D_g，以下结论成立

$$\begin{aligned}\nabla^2(D_g\phi_g) &= \nabla\cdot\nabla(D_g\phi_g) &\text{(A.1)}\\ &= \nabla\cdot(\phi_g\nabla D_g + D_g\nabla\phi_g) &\text{(A.2)}\\ &= \phi_g\nabla^2 D_g + D_g\nabla^2\phi_g + 2\nabla D_g\cdot\nabla\phi_g\,, &\text{(A.3)}\end{aligned}$$

由此得到以下恒等式

$$2\nabla D_g\cdot\nabla\phi_g = \nabla^2(D_g\phi_g) - \phi_g\nabla^2 D_g - D_g\nabla^2\phi_g\,. \tag{A.4}$$

展开方程 (1) 中的扩散项，然后代入方程 (A.4) 中的表达式，可得

$$\begin{aligned}\nabla\cdot(D_g\nabla\phi_g) &= D_g\nabla^2\phi_g + \nabla\phi_g\cdot\nabla D_g &\text{(A.5)}\\ &= D_g\nabla^2\phi_g + \frac{1}{2}\left(\nabla^2(D_g\phi_g) - \phi_g\nabla^2 D_g - D_g\nabla^2\phi_g\right) &\text{(A.6)}\\ &= \frac{1}{2}\left(\nabla^2(D_g\phi_g) - \phi_g\nabla^2 D_g + D_g\nabla^2\phi_g\right)\,. &\text{(A.7)}\end{aligned}$$

方程 (A.7) 用于方程 (21)。

附录 B. 用标准记法表示的有限体积离散与用卷积表示的有限体积离散的等价性

首先，回顾一下，两个 N 乘 M 矩阵的 Hadamard 积为

$$\boldsymbol{A}\odot\boldsymbol{B}\big|_{k\ell} = A_{k\ell}B_{k\ell} \quad \forall k\in\{1,2,\ldots,N\},\,\ell\in\{1,2,\ldots,M\} \tag{B.1}$$

和符号 $\sum$ 对矩阵的所有元素求和元素

$$\sum_{\text{entries}}\boldsymbol{A} \equiv \sum_{k=1}^{M}\sum_{\ell=1}^{N} A_{k\ell}\,. \tag{B.2}$$

在本节中，我们将展示方程（6）（亦见方程 (13) 和 (14)）中给出的扩散算子的有限体积离散与采用预定义权重表示为卷积

层的相同离散化方法之间的等价性，如公式（20）和（22）所述。考虑方程（22）右侧的每一项，我们从第二项的一部分开始，并在第 i, j 个单元中对其进行计算：

$$\boldsymbol{f}(\boldsymbol{\Phi_g}; \boldsymbol{w})\Big|_{i,\,j} = \sum_{\text{entries}} \begin{bmatrix} 0 & \frac{-1}{\Delta y^2} & 0 \\ \frac{-1}{\Delta x^2} & \frac{2}{\Delta x^2} + \frac{2}{\Delta y^2} & \frac{-1}{\Delta x^2} \\ 0 & \frac{-1}{\Delta y^2} & 0 \end{bmatrix} \odot \begin{bmatrix} \phi_{i-1,j+1,g} & \phi_{i,j+1,g} & \phi_{i+1,j+1,g} \\ \phi_{i-1,j,g} & \phi_{i,j,g} & \phi_{i+1,j,g} \\ \phi_{i-1,j-1,g} & \phi_{i,j-1,g} & \phi_{i+1,j-1,g} \end{bmatrix} \tag{B.3}$$

$$= \frac{-\left(\phi_{i-1,j,g} + \phi_{i+1,j,g}\right)}{\Delta x^2} + \frac{-\left(\phi_{i,j-1,g} + \phi_{i,j+1,g}\right)}{\Delta y^2} + \left(\frac{2}{\Delta x^2} + \frac{2}{\Delta y^2}\right)\phi_{i,j,g}\,. \tag{B.4}$$

现在，考虑第二项的整体，

$$\left(\boldsymbol{D_g} \odot \boldsymbol{f}(\boldsymbol{\Phi_g}; \boldsymbol{w})\right)\Big|_{i,\,j} = \frac{-D_{i,j,g}\left(\phi_{i-1,j,g} + \phi_{i+1,j,g}\right)}{\Delta x^2} + \frac{-D_{i,j,g}\left(\phi_{i,j-1,g} + \phi_{i,j+1,g}\right)}{\Delta y^2} + \left(\frac{2}{\Delta x^2} + \frac{2}{\Delta y^2}\right) D_{i,j,g}\phi_{i,j,g}\,. \tag{B.5}$$

类似地，对于方程（22）右侧的第三项

$$\left(\boldsymbol{\Phi_g} \odot \boldsymbol{f}(\boldsymbol{D_g}; \boldsymbol{w})\right)\Big|_{i,\,j} = \frac{-\phi_{i,j,g}\left(D_{i-1,j,g} + D_{i+1,j,g}\right)}{\Delta x^2} + \frac{-\phi_{i,j,g}\left(D_{i,j-1,g} + D_{i,j+1,g}\right)}{\Delta y^2} + \left(\frac{2}{\Delta x^2} + \frac{2}{\Delta y^2}\right) D_{i,j,g}\phi_{i,j,g}\,. \tag{B.6}$$

The 方程（22）右侧的第一项可展开如下

$$\boldsymbol{f}\left(\boldsymbol{D_g} \odot \boldsymbol{\Phi_g}; \boldsymbol{w}\right)\Big|_{i,\,j} = \frac{-\left(D_{i-1,j,g}\phi_{i-1,j,g} + D_{i+1,j,g}\phi_{i+1,j,g}\right)}{\Delta x^2} + \frac{-\left(D_{i,j-1,g}\phi_{i,j-1,g} + D_{i,j+1,g}\phi_{i,j+1,g}\right)}{\Delta y^2} + \left(\frac{2}{\Delta x^2} + \frac{2}{\Delta y^2}\right) D_{i,j,g}\phi_{i,j,g}\,. \tag{B.7}$$

根据扩散的定义，将公式（B.5）、（B.6）和（B.7）中的表达式合并

算子由方程（22）给出，并将乘以各个标量通量项的项进行汇总，得到

$$\begin{aligned}
\boldsymbol{f}^{\mathrm{Diff}}(\boldsymbol{\Phi}_g, \boldsymbol{D}_g; \boldsymbol{w}) &= \frac{1}{2}\left(\boldsymbol{f}(\boldsymbol{D}_g \odot \boldsymbol{\Phi}_g; \boldsymbol{w}) + \boldsymbol{D}_g \odot \boldsymbol{f}(\boldsymbol{\Phi}_g; \boldsymbol{w}) - \boldsymbol{\Phi}_g \odot \boldsymbol{f}(\boldsymbol{D}_g; \boldsymbol{w})\right) && \text{(B.8)} \\
&= \frac{-\left(D_{i-1,j,g}\phi_{i-1,j,g} + D_{i+1,j,g}\phi_{i+1,j,g}\right)}{2\Delta x^2} + \frac{-\left(D_{i,j-1,g}\phi_{i,j-1,g} + D_{i,j+1,g}\phi_{i,j+1,g}\right)}{2\Delta y^2} \\
&\quad + \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right) D_{i,j,g}\phi_{i,j,g} && \text{(B.9)} \\
&\quad + \frac{-D_{i,j,g}\left(\phi_{i-1,j,g} + \phi_{i+1,j,g}\right)}{2\Delta x^2} + \frac{-D_{i,j,g}\left(\phi_{i,j-1,g} + \phi_{i,j+1,g}\right)}{2\Delta y^2} + \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right) D_{i,j,g}\phi_{i,j,g} \\
&\quad - \frac{-\phi_{i,j,g}\left(D_{i-1,j,g} + D_{i+1,j,g}\right)}{2\Delta x^2} - \frac{-\phi_{i,j,g}\left(D_{i,j-1,g} + D_{i,j+1,g}\right)}{2\Delta y^2} - \left(\frac{1}{\Delta x^2} + \frac{1}{\Delta y^2}\right) D_{i,j,g}\phi_{i,j,g} \\
&= -\left(\frac{D_{i-1,j,g} + D_{i,j,g}}{2\Delta x^2}\right)\phi_{i-1,j,g} - \left(\frac{D_{i,j,g} + D_{i+1,j,g}}{2\Delta x^2}\right)\phi_{i+1,j,g} \\
&\quad - \left(\frac{D_{i,j-1,g} + D_{i,j,g}}{2\Delta y^2}\right)\phi_{i,j-1,g} - \left(\frac{D_{i,j,g} + D_{i,j+1,g}}{2\Delta y^2}\right)\phi_{i,j+1,g} && \text{(B.10)} \\
&\quad + \left(\frac{D_{i-1,j,g} + 2D_{i,j,g} + D_{i+1,j,g}}{2\Delta x^2} + \frac{D_{i,j-1,g} + 2D_{i,j,g} + D_{i,j+1,g}}{2\Delta y^2}\right)\phi_{i,j,g}\,.
\end{aligned}$$

由此可见，方程（B.10）等价于方程（6）中的离散扩散算子。换言之，这种特定的有限体积离散可以表示为神经网络中的一个卷积层，其卷积核或滤波器大小为 3×3，权重为

$$\boldsymbol{w} = \begin{bmatrix} 0 & \frac{-1}{\Delta y^2} & 0 \\ \frac{-1}{\Delta x^2} & \frac{2}{\Delta x^2} + \frac{2}{\Delta y^2} & \frac{-1}{\Delta x^2} \\ 0 & \frac{-1}{\Delta y^2} & 0 \end{bmatrix} \qquad \text{(B.11)}$$