PODTherm-GP：一种基于物理原理的数据驱动方法，用于实现多核 CPU 的高效架构级热仿真

Lin Jiang、Anthony Dowling、Ming-C. Cheng、Yu Liu

arXiv:2305.01911v1 [cs.CE] 3 May 2023

摘要—本文基于本征正交分解（POD）和伽辽金投影（GP）提出了一种热仿真方法，以下简称 PODTherm-GP，并评估了该方法在多核 CPU 上的效率和准确性。GP 将传热方程投影到一个数学空间中，该空间的基函数由学习算法 POD 基于热数据生成。热解数据通过 FEniCS 使用有限元法（FEM）采集，并考虑了适当的参数变化。GP 将热传递的物理原理融入该方法，从而实现高准确性和高效率。FEM 热仿真中 CPU 的动态功率图由 gem5 和 McPACT 生成，仿真负载采用 SPLASH-2 基准测试。结果表明，PODTherm-GP 能够以与 FEM 同样精细的分辨率准确预测 CPU 的温度。此外，结果还证明，即使超出训练条件，PODTherm-GP 也能够准确预测芯片的动态温度分布。与 FEM 相比，该方法可将自由度减少 5 个数量级以上，并实现 4 个数量级的加速。

索引词—**热仿真、正交分解、数据驱动学习方法、多核 CPU**。

一、引言

在过去几十年中，由于半导体技术节点的大幅微缩以及多核架构的引入，高性能微处理器的性能提升一直受到热问题的制约，这些变化导致处理器的功率密度显著增加 [1]。较大的温度梯度和热点不仅会损害处理器的性能，还会降低其可靠性 [2], [3]。对于高性能处理器而言，热管理和面向热特性的设计空间探索 [4], [5] 一直是缓解这些热问题、提高处理器性能和可靠性的有效方法。例如，研究发现[6], , 与基于 ARM 动态电压频率调节（DVFS）的智能功率分配方法相比，通过自适应热管理框架，异构多核处理器的平均性能进一步提升了 8.9%。然而，这些有效的热管理方法仍然.....

Lin Jiang、Anthony Dowling、Ming-C. Cheng 和 Yu Liu 任职于美国纽约州波茨坦市克拉克森大学电气与计算机工程系，邮编 13699-5720。

电子邮件： {jiangl2, dowlinah, mcheng, yuliu}@clarkson.ed u.

本研究得到美国国家科学基金会项目资助，项目编号为 ECCS-2003307 和 OAC-2118079。

需要一种有效的热仿真工具。对于实时应用，例如运行时热感知任务调度，人们希望拥有一种效率极高且精度很高的热仿真工具。

对于半导体芯片的热仿真，针对不同应用已经开发了许多方法。其中一些方法着重于热仿真的准确性，以捕捉热点，例如基于有限差分法、有限体积法或有限元法（分别为 FDM、FVM 或 FEM）的直接数值模拟（直接数值模拟）。然而，这些直接数值模拟方法计算量巨大，通常难以用于架构级热仿真。因此，在效率发挥重要作用的情况下，还提出了其他几种方法，包括热电路模型、格林函数法、基于机器学习的方法等。所有这些方法都通过采用近似方法牺牲准确性和/或分辨率，从而实现比直接数值模拟更高的效率，但这些近似也带来了严重限制。例如，高效的热电路是以极低的分辨率和不准确的解为代价实现的。当使用热电路模型或基于机器学习的方法时，如果希望保持与直接数值模拟一样精细的分辨率，计算开销就会变得与直接数值模拟相当。此外，格林函数法难以应用于三维动态热仿真。近年来，基于本征正交分解（POD）的数据驱动方法，在物理原理的指导下，凭借能够同时实现高准确性、高分辨率和高效率的优势，逐渐成为半导体芯片热仿真中越来越受关注的方法 [7]–[9]。这些热仿真方法将在第 II 节中进一步讨论。

在本研究中，我们基于本征正交分解-伽辽金（PODTherm-GP）方法开发了一种架构级热仿真器，用于对多核 CPU 进行三维动态热仿真。该方法的早期概念已在 [7] 中进行了简要说明。这种本征正交分解-伽辽金建模技术无需先验假设，即可准确、高效地预测多核处理器的温度分布。本征正交分解将多核 CPU 物理域中的热问题投影到函数空间中，该函数空间的基函数（或 POD 模态）由多核 CPU 的直接数值模拟所生成的热数据训练得到。在我们的研究中，热数据采集自 FEniCS；FEniCS 是一个使用有限元法（FEM）求解偏微分方程（PDE）的开源计算平台 [10]。为了在 FEniCS-FEM 热仿真中提供真实的热源

每个单元中的功率轨迹由 gem5 [11] 和 McPAT [12] 结合选定的基准测试生成。在验证 PODTherm-GP 时，我们为每个单元使用了不同于热数据采集阶段所用的功率轨迹。

POD-Galerkin 方法的关键概念和贡献总结如下：

1) 本征正交分解过程会训练一组最优模态，以提取所收集数据中蕴含的关键信息。若数据质量足够高，该方法便能以最少数量的模态（即自由度）实现高精度预测。2) Galerkin 投影依据传热方程，在预测过程中施加物理原理，并以良好精度提供外推能力。3) 因此，PODTherm-GP 能够高效且准确地进行三维动态热仿真，其热分辨率可达到通过直接数值模拟收集的训练数据的精细程度。4) 与 FEniCS-FEM 相比，PODTherm-GP 在保持高精度的同时，将数值自由度减少了 5 个数量级以上，仿真速度提升了 4 个数量级。5) POD 模态的质量在很大程度上取决于所收集训练数据中热梯度的准确性。

本文其余部分组织如下：首先，第 II 节介绍半导体芯片和处理器热仿真的相关研究。第 III 节从数学角度介绍 PODTherm-GP 方法论。第 IV 节阐述该方法论的评估过程。随后，第 V 节采用 PODTherm-GP 对选定的多核 CPU 进行动态热仿真，并在第 VI 节从准确性和效率两个方面讨论仿真结果。最后，第 VII 节总结本文的研究成果。

II. 半导体芯片热仿真相关研究

热仿真是热[6], [13] 和功耗 [14], [15] 管理的重要组成部分。热管理和功耗管理是两种主要策略，旨在最大限度地减小芯片温度与功耗之间的耦合效应，并防止半导体芯片过热。通过精确的热仿真，可以预测半导体芯片中的热点，随后可通过动态电压频率调节 [6]等功耗管理技术加以抑制。温度降低反过来又能提高半导体芯片的功耗效率。此外，热仿真预测的芯片温度分布是热管理所必需的，可通过探索冷却技术 [13], [15]或热感知任务调度 [3]来提升芯片性能并降低功耗。以下简要讨论现有的半导体芯片热建模方法。

A. 直接数值模拟（*DNS*）

为了获得半导体芯片中详细的温度分布，通常采用有限元法、有限体积法或有限差分法。例如，Wang 等人基于有限差分法开发了三维热交替方向隐式法 [16]。Xu [17] 使用基于有限元法的 ANSYS 对四核芯片进行热仿真。Vaddina 等人 [18] 使用 COMSOL 对三维堆叠结构进行热建模与分析。随着多核处理器 [19], 的快速发展，局部功率密度显著提高，因此热预测需要达到足够高的分辨率，以捕捉小尺寸热点。尽管直接数值模拟具有较高的热分辨率和精度，但由于其计算量巨大，不适用于芯片架构级的仿真。

B. 热电路模型

由于其简单高效，集总元件热路已在架构级得到广泛应用 [20]。例如，HotSpot [21] 和 3D-ICE [13]是一些广受欢迎的架构级热路模拟器。Glocker 等人 [22]使用 HotSpot 研究了 16 核系统中的温度分布，Florea 等人 [23] 则使用 HotSpot 为增强 Sniper 多核模拟器提供热分析。HotSpot 和 3D-ICE 已集成到性能-功耗-热仿真工具链中，例如 HotSniper [24], 、HotGauge [25]和 CoMeT [26]。然而，热电路模型的效率是通过使用大型集总元件实现的，这些元件无法充分考虑分布式热传递，可能导致预测不准确 [27]。为了提高准确性，需要大幅减小 RC 元件的尺寸，正如 HotSpot 的网格模型 [28] 所采用的方法；该模型随后类似于计算密集型的有限差分法。

C. 格林函数法

在传统的格林函数方法中，假设不存在边界影响，以大芯片中心的单位点热源为响应来计算单个格林函数。格林函数是芯片的空间脉冲响应，而热解则通过对不同位置点源的脉冲响应进行叠加构建。因此，在有限域的热仿真中纳入边界条件（BCs） [29], [30] ，或在功率源靠近芯片边缘时使用格林函数 [31], 本质上都很困难。将传统方法应用于瞬态热仿真也很困难 [29], [32]。不过，与有限元法、有限体积法或有限差分法相比，该方法效率显著更高，因为它只考虑生成热源的单层。为克服上述局限，人们采用了不同技术并进行了重大改进，但这不可避免地降低了计算效率。例如，基于绝热边界条件 [33]的镜像法，利用格林函数修正角部和边缘效应。然而，这种方法不适用于其他类型的边界条件。使用

功率模糊方法并结合时空脉冲后，便可在格林函数框架内进行瞬态热仿真 [33]。不过，这些方法仍只能提供功耗层上的二维温度。已有研究尝试通过多层格林函数纳入三维温度分布 [34]；但该方法仅适用于稳态仿真。

D. 机器学习技术

近年来，机器学习方法已被应用于研究半导体芯片的各个方面，包括多核芯片， [35]–[37]。在各种机器学习技术中，由于神经网络方法简单且高效，因此已成为半导体芯片热仿真中最常用的方法 [35], [36]。采用神经网络时，需要使用热数据训练计算域中的每个热单元（或神经元），使其能够对热激励（热源和边界条件）以及相邻神经元作出响应。因此，热仿真所需的内存空间和计算时间取决于所选神经元的数量以及各个神经元之间紧密连接所采用的距离。尽管这些方法比直接数值模拟具有显著更高的效率，但通常仅限于对少量节点进行热解，而无法获得芯片的详细温度分布。此外，这些方法基于数据统计，缺乏基本物理原理的指导，因此在进行外推时通常会产生错误的解。

E. 基于物理的学习方法

过去已有研究基于POD，探讨了一种源自基于投影的降阶模型的不同学习算法 [38], [39]。[40] 中包含了一份POD教程，以直观且易于理解的方式介绍了基本概念和方法。POD将热问题从物理域投影到函数空间，以减少自由度。不同于许多其他基于投影的方法（如傅里叶级数、勒让德多项式、贝塞尔函数等）预先假设基函数，构成POD空间的基函数（或POD 模态）通过所关注问题的热解数据生成（或训练）得到。热数据采集自精确的直接数值模拟，用于训练POD 模态，以捕获由参数变化（包括加热功率和边界条件）引起的关键热信息。POD提供了一组最优基函数，使最小二乘（LS）误差达到最小，而该误差受热数据质量限制。为将热传递的物理原理纳入POD仿真，还需将仿真域的动态热传递方程进一步投影到POD空间。这一严格过程引导POD热解遵循动态热传递方程。Cheng团队已应用POD仿真方法，为绝缘体上硅和FinFET器件、门级FinFET电路以及互连线开发了高效且精确的热仿真模型 [8]。

POD仿真方法在基本物理原理指导下的主要优势，是在保持与直接数值模拟相当的精度和等效分辨率的同时，利用少量模态将大维度问题降至极小维度。然而，还需要通过后处理在物理域中重构动态温度，这成为主要的计算瓶颈。不同于直接数值模拟需要将整个仿真时间和仿真域纳入仿真，针对实际情况，POD方法中的后处理计算只需在每个选定的时间间隔内，对芯片上的少量高温区域进行处理。这将显著减少后处理所需的计算时间和内存空间。

III. POD热仿真方法

如上所述，基于本征正交分解的模拟器PODTherm-GP 能够利用少量 POD 模态 [8], [9]对大规模动态热问题进行建模。这些模态通过物理域的热解数据进行训练，而这些数据来自在一系列参数变化（包括边界条件和动态功率图）下执行的直接数值模拟。因此，POD模态专门适用于目标问题中边界条件和功率变化的训练范围，并能大幅降低传热问题的自由度。正如第 V 节将展示的那样，由于将传热方程投影到 POD空间中，只要纳入更多模态，PODTherm-GP 即使超出训练范围，也能提供准确的预测。投影到 POD 模态显然融入了传热物理原理，使本征正交分解方法即使在外推情形下，也能充分预测空间中的动态热评估。

A. POD 模态的生成

通过最大化整个区域内热解数据与模态的均方内积，对 POD 模态进行优化，其中该区域受到动态或静态参数变化 [38], [39], subjectedto 的影响。在我们的研究中，通过 DNS 在每个仿真时间步收集空间中的热数据。该最大化过程最终形成一个由弗雷德霍姆方程描述的特征值问题，

$$\int_{\Omega} R(\vec{r},\vec{r}')\varphi(\vec{r}')d\vec{r}' = \lambda\varphi(\vec{r}), \tag{1}$$

其中， λ 是与 POD 模态（即特征函数 φ）对应的特征值，而 $R(\vec{r},\vec{r}')$ 是一个两点相关张量，其定义为

$$R(\vec{r},\vec{r}') = \langle T(\vec{r},t) \otimes T(\vec{r}',t)\rangle, \tag{2}$$

其中， $\otimes$ 为张量算子，尖括号 $\langle\rangle$表示对多个热数据集取平均。求解式（1）中的特征值问题后，温度可表示为

$$T(\vec{r},t) = \sum_{i=1}^{M} a_i(t)\varphi_i(\vec{r}), \tag{3}$$

其中， a_i 为 φ_i 的加权系数， M 为用于重构温度解的 POD 模态数量。如果数据的特征值随模态阶数增加而迅速降低，则只需少量自由度（或 M）即可实现准确的热预测。

B. 将传热方程投影到 *POD*空间

为了预测域结构中式（3）所给出的动态热分布，需要确定系数 $a_i(t)$。为此，使用 Galerkin 投影将传热方程投影到 POD空间，

$$\int_\Omega (\varphi_i(\vec{r})\frac{\partial \rho CT}{\partial t} + \nabla\varphi_i \cdot k\nabla T)d\Omega = \int_\Omega \varphi_i(\vec{r})P_d(\vec{r},t)d\Omega - \int_S \varphi_i(\vec{r})(-k\nabla T\cdot\vec{n})dS, \tag{4}$$

其中， k、 ρ 和 C 分别为热导率、密度和比热， $P_d(\vec{r},t)$ 为内部功率密度， S 为边界表面， $\vec{n}$ 为边界表面上的外法向量。利用所选的 POD 模态，式（4）可改写为关于 $a_i(t)$ 的 M 维常微分方程（ODEs），

$$\sum_{i=1}^{M} c_{i,j}\frac{da_i(t)}{dt} + \sum_{i=1}^{M} g_{i,j}a_i(t) = p_j,\ j = 1 \text{ to } M, \tag{5}$$

其中，$c_{i,j}$ 是 POD空间中热容矩阵的元素，其定义为

$$c_{i,j} = \int_\Omega \rho C\varphi_i\varphi_j d\Omega. \tag{6}$$

$g_{i,j}$ 和 p_j 分别是 POD空间中热导矩阵和功率向量的元素。在式（5）中，$g_{i,j}$ 和 p_j 也可能随边界条件变化。本文在芯片的边界表面上采用了两种边界条件：绝热边界条件和对流边界条件。对于绝热表面，热通量为零，$g_{i,j}$ 和 p_j 可写为

$$g_{i,j} = \int_\Omega k\nabla\varphi_i\cdot\nabla\varphi_j d\Omega,\ p_j = \int_\Omega \varphi_j P_d(\vec{r},t)d\Omega. \tag{7}$$

对于对流边界，表面上的热通量表示为

$$-k\nabla T\cdot\vec{n} = -k\frac{\partial T}{\partial n} = h(T - T_{amb}), \tag{8}$$

其中， h 是传热系数， T_{amb} 是环境温度。将（8）代入（4），$g_{i,\ j}$ 对流边界条件为

$$g_{i,j} = \int_\Omega k\nabla\varphi_i\cdot\nabla\varphi_j d\Omega + \int_S h\varphi_i\cdot\varphi_j dS, \tag{9}$$

and p_j 变为

$$p_j = \int_\Omega \varphi_j P_d(\vec{r},t)d\Omega + \int_S h\varphi_j T_{amb}dS. \tag{10}$$

从 gem5 和 McPAT 中获得动态功耗后（详见第 IV 节），即可预先计算式（7）和（10）中 POD 空间内的内部功率源强度。衬底底部的边界条件建模为式（8）给出的对流换热，其传热系数由均热板和散热器的结构尺寸及材料属性决定，温度为 T_{amb} = 45 °C。其他所有边界均为绝热。确定模态后，即可预先计算式（5）中的所有系数。在 POD 仿真中由式（5）求得 $a_i(t)$ 后，即可根据式（ 3 ）预测温度解。

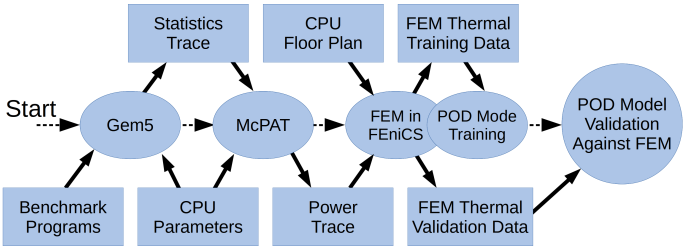

图 1. 使用 PODTherm-GP 对架构级多核 CPU 进行热仿真的工作流程。

表 I Intel Xeon E5-2699 v3 的 GEM5 配置参数

参数	值
时钟频率	2.3 GHz
核心数	18
$L1_i$ 大小	32 kB
$L1_d$ 尺寸	32 千字节
二级缓存大小	256 千字节
三级缓存大小	45 MB
缓存行大小	64
内存容量	16 GB
内存类型	DDR3 1600 8x8
发射宽度	8
ROB容量	192
整数单元数量	4
浮点单元数量	2
整数乘法延迟	3
整数除法延迟	18
浮点乘法延迟	5
浮点除法延迟	6
DRAM带宽	68 GiB/s
DRAM 请求到响应延迟	30ns

IV. 评估方法

图 1 展示了通过 PODTherm-GP 对多核 CPU 进行架构级热仿真的完整工作流程。本研究选用一款具有 18 个核心的 Intel Xeon E5-2699 v3 处理器，其布局图如图 2 [41]，所示，以展示 PODTherm-GP 的有效性。使用下文所述的基准测试，通过两个开源模拟器 gem5 和 McPAT 生成 CPU 各单元的功耗轨迹，供 FEniCS-FEM 执行热仿真。选择 gem5 对 CPU 执行周期级仿真，并生成 CPU 的性能统计数据 [11]。用于通过 gem5 仿真 Intel Xeon E5-2699v3 处理器的配置参数如表 I 所示。这些参数基于 Intel 提供的该处理器规格，以及 Jongerius 等人在其研究中使用的参数；他们在 gem5 [42] 中对一款类似的 Intel Xeon 处理器进行了建模。对于未知参数，均采用 gem5 配置的默认值。

表 II 选定的 SPLASH-2 基准测试

基准测试	描述	分配的核心索引
连续海洋	模拟大规模海洋洋流，使用连续分区	2、8、14
非连续海洋	模拟大规模流动海洋洋流，使用非连续分区	3, 9, 15
辐射度	计算光线的分布在场景中使用分层漫反射辐射度方法	1, 7, 13
水体空间	模拟水流在……中的流动使用三维空间数据结构	5、11、17
Water N-Squared	模拟水在空间中的流动使用不同的算法比水-空间	4、10、16
Barnes	使用 Barnes-Hut 方法来模拟一组天体的相互作用	0, 6, 12

选用 SPLASH-2基准测试套件作为 gem5 [43] 的仿真工作负载。SPLASH-2 是一套以 C/C++ 编写的开源基准测试，在计算机体系结构研究领域中得到广泛应用 [44]。本文从 SPLASH-2 中选取了部分可用基准测试，如表 II 所示。为生成逼真的动态功率图，本文按照图 2 中标记的核心索引分配这些基准测试，如表 II 所示，以模拟真实场景。这四个基准测试中，有两个包含不同变体：Ocean 和 Water。Ocean 的变体为连续分区和非连续分区。这些变体会改变算法划分仿真空间的方式。Water 的变体为 spatial 和 N-squared。这些变体会改变仿真所用的算法。凡是需要从输入文件加载数据的基准测试，均经过修改，将基准测试所提供的数据直接包含在源代码中。虽然这增加了源代码的大小，但此修改避免了 gem5 从磁盘读取文件的需求。gem5 输出的 CPU 事件跟踪包含 CPU 不同硬件组件的性能计数器，例如各功能单元的使用情况、缓存访问次数等。模拟器会在每个仿真时间步捕获并存储尽可能多的 CPU 使用信息。本文中的仿真以 4.35 μs 为采样间隔，训练和演示 PODTherm-GP 的持续时间分别为 4.35 ms 和 6.09 ms 。

解析 gem5 针对所选基准测试输出的统计跟踪数据，为 McPAT 创建输入，以模拟 CPU 运行期间的功耗。这些输入文件采用 XML文件 格式，其中包含 McPAT 运行所需的信息。McPAT 还需要体系结构信息，例如功能单元的数量、缓存的大小等。结合体系结构信息和性能统计数据，McPAT 会估算各组件的面积，并计算每个时间步 [12] 各组件的动态功耗和静态功耗。随后解析 McPAT 输出的功耗值，以生成动态

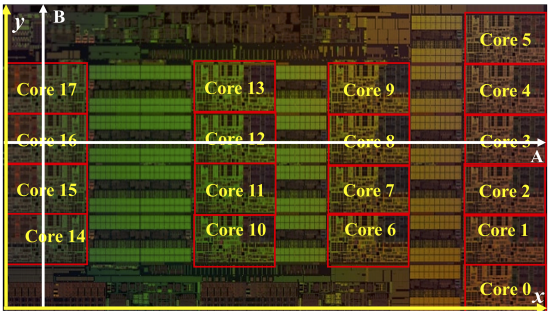

图 2。面积为 31.0 $mm \times 21.5\ mm$ 的 IntelXeon E5-2699v3 CPU [41] 布局图。A 和 B 表示演示的绘图路径。

功率图文件。

利用生成的动态功率图，使用 FEniCS-FEM 进行热仿真并收集数据，以生成本征正交分解模态，如图 1 所示。本征正交分解模态的生成过程和 POD 模型的构建过程将在下一节进一步介绍。为验证 POD 模型，针对所选多核 CPU，使用不同的动态功率图进行单独的 FEniCS-FEM 热仿真。随后，将本征正交分解结果与采用相同数值设置和动态功率图的 FEniCS-FEM 仿真结果进行比较。

V. 本征正交分解模型的实现与验证

A. POD 模态的训练

所选多核 CPU 的仿真域（其布局图如图 2 [41], 所示）包括一个厚度为 55.8 um 的顶部加热层，以及一个厚度为 241.8 um 的衬底。加热层覆盖功率耗散所在的顶部厚度区域（例如器件层和互连线层）。将式（8）中给出的传热系数所描述的对流换热应用于衬底底部。传热系数根据芯片尺寸和材料属性，通过均热板、热界面材料和散热器的热阻进行建模，与其他研究类似 [21], [28]。其余边界表面假定为绝热。由 gem5 和 McPAT 生成的各功能单元动态功耗按 2.3 GHz 下的 10k 个 CPU周期取平均值。为研究训练数据质量对 POD 模型鲁棒性的影响，使用相同的动态功率图，从 FEniCS-有限元法中分别采集两组热数据，其网格为 $256 \times 256 \times 14$ 和 $1104 \times 620 \times 14$，或空间分辨率为 $0.121 \times 0.083 \times 0.023\ mm^3$ 和 $0.028 \times 0.035 \times 0.023\ mm^3$，分别对应 x、y 和 z。基于这两组分辨率较粗和较细的数据集构建的 POD模型分别称为 POD Model-A 和 Model-B。需要注意的是，本研究使用的较粗网格仍比大多数芯片级热仿真研究所用的网格更精细，除非需要极高的分辨率 [31], [33]。

POD 模态和特征值通过快照法根据 (1) 确定， [8], [45] 使用从 FEniCS-FEM 仿真中收集的训练数据。POD 模态是

因此，该方法能够针对训练数据中包含的关键信息进行定制，以考虑边界条件和动态功率图的变化。每个特征值代表其对应模态所捕获的温度变化均方值，因此能够反映该模态的重要性。随后，可以根据.....估算公式 (3) 中实现准确预测所需的 POD 模态数量。

$$Err_{Theo} = \sqrt{\frac{\sum_{i=M+1}^{N_s} \lambda_i}{\sum_{i=1}^{N_s} \lambda_i}}, \tag{11}$$

其中， Err_{Theo} 为理论最小二乘误差， N_s 为快照或采样数据集的数量。(11)给出了理想的最小二乘误差，但由于数值精度和计算机精度的限制，PODTherm-GP 得出的最小二乘误差通常大于 Err_{Theo}。

图3按降序描述了Model-A和Model-B所收集数据的特征值。由于所收集数据集的分辨率足够高，这两个POD模型的特征值几乎相同。特征值在最初几个模态中显著下降，而在更高阶模态中的下降速率较慢。由于计算机精度的限制，特征值降低了16个数量级后，在170个模态之后最终趋于近似平坦。插图中的放大频谱显示，从第一模态到第二模态，特征值下降了两个以上数量级，而到第三模态时几乎下降了四个数量级。根据(11)，如果由FEniCS-FEM收集的训练数据质量足够高，则预期Model-A或Model-B仅使用3个或更多模态即可提供准确的热预测。

为构建 POD 模型，需要使用训练数据的 POD 模态 $\varphi_i(\vec{r})$，按照式 (6)–(10) 预先计算式 (5) 中投影常微分方程的模型参数。因此，PODTherm-GP 会求解具有选定数量模态 M 的常微分方程 $\vec{a}(t)$，随后还需通过式 (3) 后处理，以计算 CPU 中的动态热分布。以下演示采用两种不同分辨率的 POD 模态，重点验证 POD 仿真技术的准确性和效率，并考察分辨率如何影响训练数据的质量、生成的模态以及模型准确性。此外，还会在训练范围之外进行验证，以考察 PODTherm-GP 在外推情况下的稳健性。

*B. POD*仿真方法演示

在这些演示中，分别使用 Model-A 和 Model-B 对选定的多核 CPU 进行热仿真，并与空间分辨率和其 POD 模态相同的 FEniCS-FEM 仿真进行比较。Model-A 和 Model-B 均在 4.35 ms 的仿真时间内完成训练，并针对 4.35 ms 和 6.09 ms 的仿真时间，使用 FEniCS-FEM 对 POD 模型进行验证。在 POD 模型与 FEniCS-FEM 仿真的任何比较中，边界条件和动态功率图均保持一致。不过，演示所用的动态功率图不同于训练数据集所用的动态功率图，详见第 IV 节。

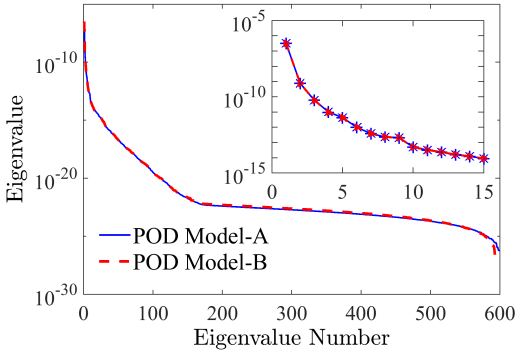

图 3. POD Model-A 和 POD Model-B 热数据的特征值谱。局部放大图显示了前 15 个特征值。

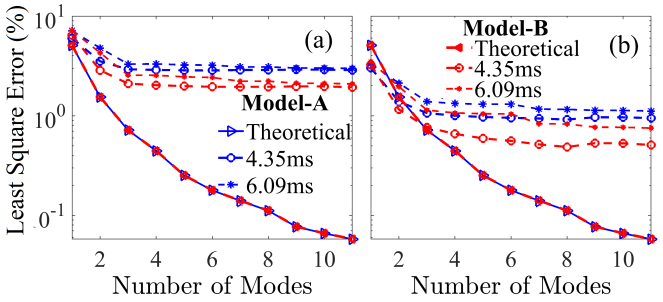

图4. 多核 CPU 在 (a) 整个芯片和 (b) 加热层上的热仿真的最小二乘误差。

图4展示了 POD Model-A 和 Model-B 对选定的多核 CPU 进行热仿真时，相对于 FEniCS-FEM 仿真的最小二乘误差。POD 模型的数值最小二乘误差是在整个仿真时间内估算的，计算方式为

$$Err_{Num} = \sqrt{\frac{\sum_{i=1}^{N_t} \int_\Omega e_i^2(\vec{r}) d\Omega}{\sum_{i=1}^{N_t} \int_\Omega (T_i(\vec{r}) - T_{amb})^2 d\Omega}}, \tag{12}$$

其中， $T_i(\vec{r})$ 和 $e_i(\vec{r})$ 分别表示 FEniCS-FEM 给出的温度，以及在第 i 个时间步，FEniCS-FEM 与 POD 模型之间的温差； N_t 表示时间步总数。

可以看出，由于图3所示的特征值谱几乎相同，Model-A 和 Model-B 的理论最小二乘误差基本重合。理论最小二乘误差从使用1个模态时的5.1%迅速下降至使用3个模态时的0.71%。然而，在使用3个模态时，图4(a)所示整个芯片的数值最小二乘误差，在4.35 ms 仿真中，Model-A（分辨率较粗）达到2.9%，Model-B（分辨率较细）达到2.1%。超过3个模态后，较粗网格模型的误差基本保持不变，而较细网格模型的误差则继续缓慢下降至1.93%。对于图4(b)中的加热层，在4.35ms 仿真中使用3个模态时，Model-A 和 Model-B 的最小二乘误差分别为1.1%和0.76%。Model-A 的误差继续下降，在使用8个模态时达到0.91%；Model-B 的误差则下降至0.49%，并在超过8个模态后保持在0.51%左右。整个芯片和♀♀♀♀♀♀

当使用更精细网格的训练数据生成 POD 模态时，加热层的表现明显得到改善。

对于仿真时间超出训练范围的 6.09 ms 情况，不出所料，POD 模型的最小二乘误差会增大。然而，图4(a)和(b)清楚地表明，在这种外推情况下，随着纳入更多模态，6.09 ms 仿真的最小二乘误差会下降，并趋近于 4.35 ms 情况下的误差。例如，使用 3–11 个模态时，Model-B 对整个芯片的最小二乘误差在 4.35 ms 情况下保持在 1.93%–2.1% 左右；在 6.09 ms 情况下，其误差在使用 3 个模态时为 2.5%，使用 7 个和 11 个模态时则分别降至 2.2% 和 2.09%。图4(a)中 Model-A 的曲线也呈现出类似趋势：6.09 ms 曲线随模态数增加而下降，并且这两条最小二乘误差曲线在使用 10 个或更多模态时几乎重合。图4(b)中加热层也呈现出类似现象：使用更多模态时，比较这两个 POD 模型产生的最小二乘误差，结果同样如此。

如 4.35 ms 案例所示，对于 Model-A，使用 3 个模态并与 FEniCS-FEM 相比，自由度可减少 5 个数量级以上（$256 \times 256 \times 14$ 与 3 相比），整个芯片的最小二乘误差为 2.9%，加热层的最小二乘误差为 1.1%。对于 Model-B，使用 3 个模态时，自由度可减少 6 个数量级以上（$1104 \times 620 \times 14$ 与 3 相比），整个芯片或加热层的最小二乘误差分别可达到 2.1% 或 0.76%。在此 4.35 ms 案例中，如果使用 7 个模态，整个芯片或加热层的最小二乘误差分别可低至 1.95% 或 0.52%，而自由度仍可减少多达 6 个数量级。

为理解图4所示的最小二乘误差，下面将详细考察训练数据的质量如何影响 POD 模态的准确性，以及由此对动态温度分布产生的影响。

在仿真时间为 6.09 ms 的范围内，图 5(a) 和 (b) 分别比较了 POD Model-A 和 Model-B 预测的路径 A 和路径 B 交点处的温度演化与 FEniCS-FEM 的结果。在这一特定位置，即使超出训练时间（4.35 ms），采用 3 个或更多模态的两种 POD-Galerkin 模型均与 FEniCS-FEM 结果高度吻合。当使用 7 个模态时，$t > 2.2\ ms$ 的绝对百分比误差趋于平缓，Model-A 和 Model-B 的误差分别保持在接近或低于 1.62% 和 0.85% 的水平，其中分辨率更高的模型具有更高的精度。由于 T_{amb} 被用作初始芯片温度，因此相对于 T_{amb} 而言，$t < 0.55\ ms$ 的百分比误差看起来明显较大，尽管在 $t \approx 0.55\ ms$ 处的绝对误差小于 0.1℃，且在 $t = 0^+\ ms$ 附近远小于 0.01℃。

Model-A 的路径 A 和路径 B 上的温度分布如图 6 所示，Model B 的温度分布如图 7 所示。根据图 2 中的 CPU 布局图，图 6 和图 7 表明，由于根据所选基准测试及其计算强度进行核心分配，多核 CPU 的核心 12、16 和 17 的温度似乎高于其他功能单元。总体而言，如图 6 所示，采用 3 个模态的 POD Model-A 能够准确预测芯片中路径 A 和路径 B 上的热分布。然而，在高温区域，即核心 16 和 17 中，与 FEniCS-FEM 结果相比，存在 2%-3% 的偏差。当采用更精细的网格时，

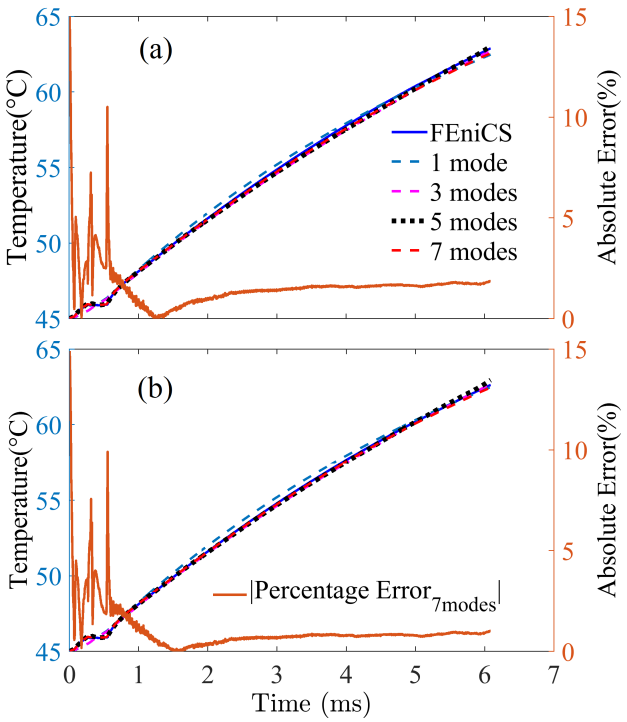

图 5. FEniCS-FEM 与 (a) Model-A 和 (b) Model-B 在 (2.3 mm, 11.8 mm) 处的温度演化对比，该点是图 2 所示路径 A 和路径 B 的交点。还给出了每个采用 7 个模态的 POD 模型的绝对百分比误差。

在 FEniCS-FEM 中用于收集训练数据以生成 POD 模态时，具有 3 个或更多模态的 Model-B 在路径 A 和路径 B 上均能提供出色的热预测，如图 7 所示。图中还显示，沿路径 A 和路径 B 几乎所有位置上，Model-B（图 7 中更细的网格）产生的误差都比 Model-A（图 6 中更粗的网格）低 50%。

为了解网格尺寸如何影响 POD 模型精度，图8考察了 Model-A 和 Model-B 的 POD 模态分布及其梯度，并展示了沿路径 A 和路径 B 的两种模型的模态梯度差异。由于这两个 POD 模型的特征值谱几乎相同（如图3所示），其模态也非常接近。然而，仔细观察各个方向上的模态斜率，就会发现基于较粗网格构建的模型存在的问题。如图8所示，在热梯度较大的某些位置，Model-A 与 Model-B 的 POD 模态梯度差异相对较大。例如，如图8(a)–8(c)所示，在图6(a)和7(a)所示 x 热梯度最高的位置附近，模态1、2或3在 x 约25%、75%或45%处分别出现了 POD 模态斜率差异。其他斜率较大的位置附近也可观察到明显的模态斜率差异，见 x，分别位于 $x = 12.4\ mm$ 和 $16.15\ mm$ 附近。沿路径 B，图8(d)–8(f)所示 y 处的斜率差异较小，但仍可在 $y = 10.4\ mm$ 和 $17.9\ mm$ 处的前 2 或 3 个模态中清楚观察到；而 $13.4\ mm$ 处的差异仅出现在第三个模态中。

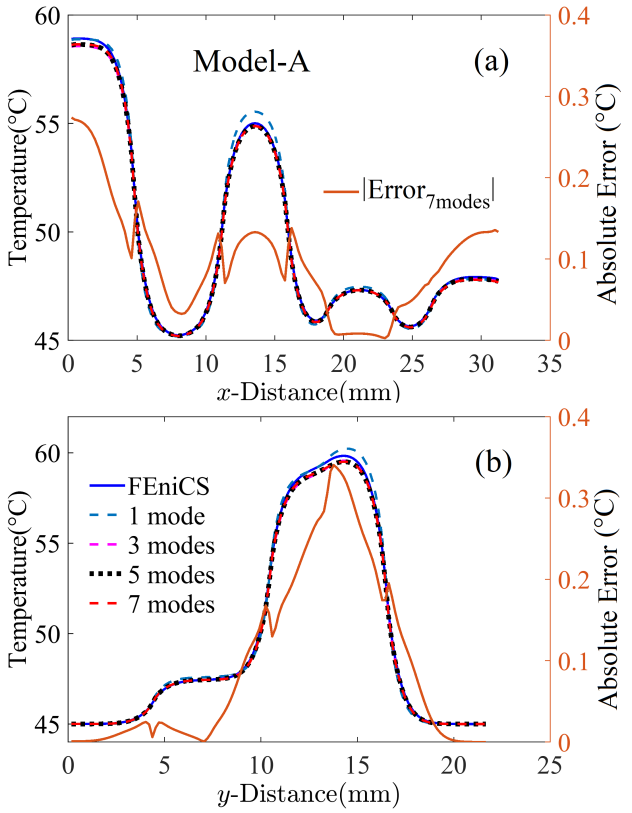

图 6. 与 FEniCS-FEM 相比，沿 (a) 路径A 和 (b) 路径B 在 4.35 ms 时由 POD Model-A 得出的温度分布。还给出了使用 7 个模态时的绝对误差。

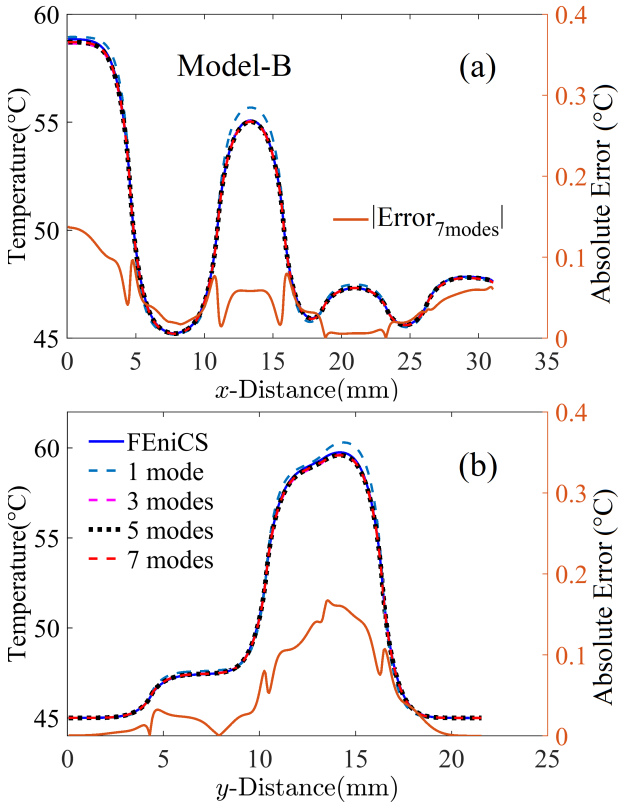

图 7. 与 FEniCS-FEM 相比，沿 (a) 路径A 和 (b) 路径B 在 4.35 ms 时由 POD Model-B 得出的温度分布。还给出了使用 7 个模态时的绝对误差。

式(5)中的 POD 模型精度取决于式(6)–(10)中各系数的质量。POD空间中的热导系数项$g_{i,\ j}$ 由热通量（$-k\nabla T$）投影到 POD 模态上得到，因此强烈依赖于模态梯度，具体定义见式(7)和(9)。因此，POD 模型的精度会受到所生成 POD 模态梯度的显著影响，而这些梯度直接受训练数据质量影响。这也解释了为何基于采用较细网格的 FEniCS 有限元模拟所采集热数据构建的 Model-B，其最小二乘误差远小于 Model-A，如上文图4所述。

为进一步考察 PODTherm-GP 在外推情形下的鲁棒性，图9和图10分别展示了 Model-A 和 Model-B 预测的 6.09 ms 时刻的温度分布，并与 FEniCS-FEM 结果进行了对比。对于图9中网格较粗的情形，采用 3 或 5 个模态的 POD Model-A 在路径 A 和路径 B 上的温度分布与 FEniCS-FEM 的结果基本吻合，但 $x\ <\ 3\ mm, 19\ mm\ <\ x\ <\ 23\ mm$ 附近的位置以及图9(a)中的 $x > 27\ mm$ 和图9(b)中的 $5\ mm < y < 10\ mm$ 除外。图4(b)表明，采用 7 个或更多模态时，Model-A 开始降低训练期之外仿真的最小二乘误差，并且图9(a)和(b)中观察到其与 FEniCS-FEM 的结果吻合得更好。由于施加在核心3和核心8上的功率显著更高，且仿真时间长于训练条件，因此需要 19 个模态才能显著改进了 Model-A 的预测。图 9 中包含的绝对误差也表明，与使用 7 个模态相比，使用 19 个模态时性能得到了显著提升，尽管在 Core 17 的 $13\ mm < x < 15\ mm$ 附近仍存在低至 1.5%-2% 的误差（类似于图 6 (b)）。对于图 10 中更精细的网格情况，在使用更多模态时，训练期之后的 Model- B 仿真也取得了类似的改进，并且在 $x < 3\ mm$ 和 $13\ mm < y < 15\ mm$ 中观察到明显更好的预测结果，其中与 FEniCS-FEM 达到了极佳的一致性。

图4、图9和图10所示结果表明，即使核心功率和仿真时间超出训练条件，PODTherm-GP 仅使用 7 个模态也能提供具有相当良好精度的预测，而要达到非常高的预测精度则需要更多模态。这种外推能力源于对传热方程进行的 Galerkin 投影。不同于大多数机器学习方法 —— 当仿真设置超出训练条件的范围时，这些方法通常会产生不可预测的解——Galerkin 投影强制预测结果遵循公式 (5) 中嵌入的物理原理。这些结果还证实，在外推情况下，数据质量（本例中由网格尺寸决定）会显著影响 PODTherm-GP 的预测精度。

VI. 讨论

将 POD Model-A 和 Model-B 的仿真结果与

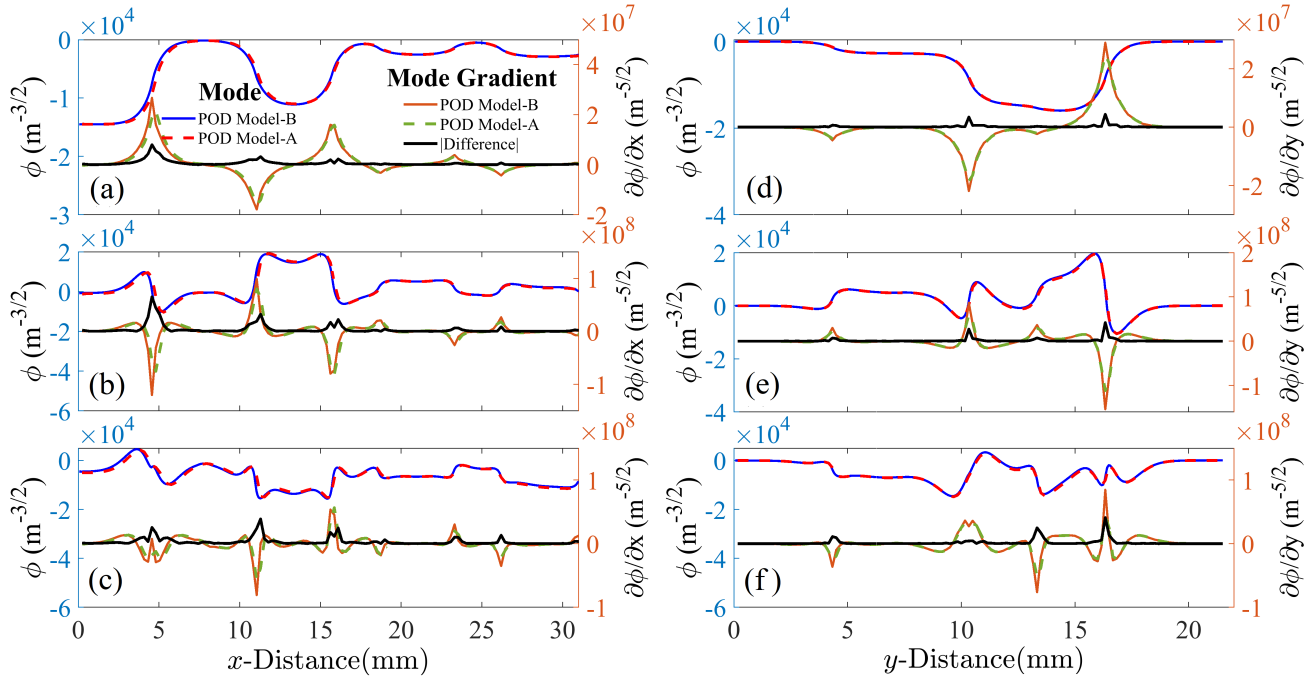

图8. 路径A上的本征正交分解模态及其梯度：(a) 第一模态、(b) 第二模态、(c) 第三模态；路径B上的本征正交分解模态及其梯度：(d) 第一模态、(e) 第二模态、(f) 第三模态。

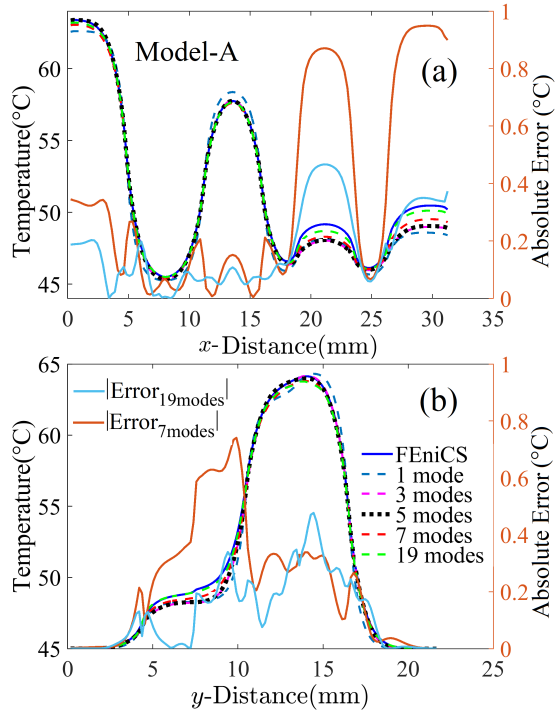

Fig. 9. 在由本征正交分解模型得出的温度分布，与 FEniCS-FEM 的结果进行比较。图中还包含使用 7 个和 19 个模态时的绝对误差。

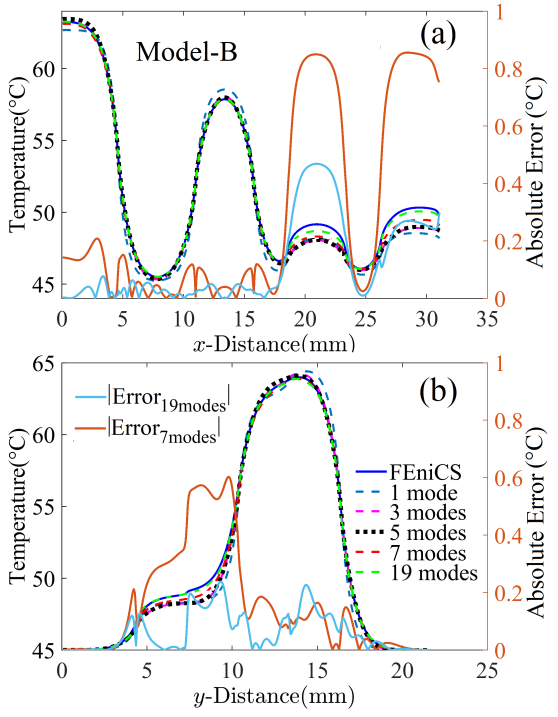

Fig. 10. 在由本征正交分解模型得出的温度分布，与 FEniCS-FEM 的结果进行比较。图中还包含使用 7 个和 19 个模态时的绝对误差。

训练时间内及超出训练时间范围的 FEniCS-FEM 仿真结果进行比较后，观察到了许多有趣且令人鼓舞的发现。当 POD 模态由更细网格训练数据生成时，整个芯片和加热层的最小二乘误差明显降低。在训练时间范围内，仅使用 3

如图4所示，Model-A和Model-B分别采用相应模态时，整个芯片的最小二乘误差约为2.9%和2.1%。更细网格模型（Model-B）采用7个模态时，误差可达到约1.95%。如果只关注加热层中的温度，

那么，采用3个模态的Model-A和Model-B的最小二乘误差分别为1.1%和0.76%，而Model-B采用7个模态时，误差最低可达0.52%。对于仿真时间6.09 ms 的外推，图4显示，POD模型的最小二乘误差如预期般增大；然而，在仿真中纳入更多模态，可以使最小二乘误差降低至更接近插值情形的水平。图4、图9和图10以及表 III 中均可观察到这一现象。表 III 中的最小二乘误差是相对于细网格 FEniCS仿真的结果估算得出的。

结果表明，用于 POD Model-A 的 FEniCS-FEM 仿真所采用的较粗网格足以生成质量较好的数据，因此其前 3 个模态的特征值和 POD 模态分布与 Model-B 的结果几乎完全相同，如图3和图8所示。然而，粗网格有限元仿真无法准确捕捉温差较大的功能单元之间某些界面处的梯度，导致训练数据不足以生成质量合格的 POD 模态。显然，如图8所示，这是 Model-A 的最小二乘误差高于 Model-B 的关键原因。这也是表 III 所示 CPU 粗网格与细网格有限元热仿真结果存在明显偏差的原因。与细网格有限元法相比，粗网格有限元法的最小二乘误差在加热层和整个芯片上分别高达 0.8% 和 1.63%。

表 III 还列出了 Model-B 在训练期内及训练期外的误差。随着纳入的模态增多，Model-B 在 $4.35\ ms$ 和 $6.09\ ms$ 情况下的最小二乘误差均有所下降，而 $6.09\ ms$ 情况下的误差也逐渐接近 $4.35\ ms$ 情况下的误差。在整个芯片上，Model-B 得出的最小二乘误差略高于采用较粗网格的有限元法仿真。然而，在加热层中，仅使用 3 个模态的 Model-B 所产生的误差就小于采用较粗网格的 FEniCS-FEM；使用 13 个模态时，误差更低至 0.51%，而较粗网格有限元法产生的误差为 0.8%。对于 $6.09\ ms$ 情况，Model-B 使用 13 个模态可将最小二乘误差降至 0.61%，低于较粗网格有限元法仿真的 0.72%。如前所述，本研究所用的较粗网格尺寸实际上比架构级热仿真中通常采用的网格更细 [31]–[33]。即使在超出训练时间的外推情况下，若使用更多模态，Model-B 的精度实际上也高于网格尺寸比芯片级热仿真多数研究通常采用的网格更细的有限元法仿真。

如第 V 节所估算，自由度可减少 5 至 6 个数量级。为评估计算效率，表 IV 列出了采用各方法对选定多核 CPU 进行热仿真的计算时间。所有实验均在 Dell Precision Tower T7910 工作站上进行，该工作站配备两颗 Intel Xeon E5-2697A v4 处理器、512G 内存，并运行 Ubuntu 20.04 操作系统。对于 PODTherm-GP，计算时间包括求解式 (5) 中的$a_i(t)$所需的时间（表 IV 中记为常微分方程），以及使用式 (3) 在物理空间中恢复温度的后处理时间。结果表明，由于 POD 模态具有较高的空间分辨率，每个时间步都要在物理空间中恢复动态温度，这一后处理过程所需的计算量明显高于求解常微分方程中的 $a_i(t)$所需的计算量。不过，后处理计算具有很强的灵活性。如前所述，实际应用中只需获取计算负载较重的某些单元的温度。这将显著减少后处理所需的计算时间和内存空间。Post1 和 Post2 分别表示恢复加热层和整个 CPU 动态温度分布的后处理过程。与分辨率和 Model-A 相同的 FEniCS-FEM 相比，采用 3 个模态的 Model-A 在加热层和整个芯片中分别实现了超过 19,000 倍和 5,500 倍的计算加速（即 2,2880/(0.082+1.098) 和 2,2880/(0.082+4.034)）。采用 5 个模态时，相较于 FEniCS-FEM，速度提升仍分别高达 12,900 倍和 3,200 倍。与采用更精细网格的 FEniCS-FEM 仿真相比，POD Model-B 仿真节省的计算时间是 Model-A 的 3.5 至 4 倍。

与采用较粗网格的 FEniCS-有限元法相比，Model-B（更精细的网格）在整个芯片上的计算时间分别节省了超过 2,100 倍和 1,200 倍（即使用 3 个和 5 个模态时分别为 2,2880/(10.604+0.081) 和 2,2880/(18.0+0.101)）。仅考虑加热层时，与采用较粗网格的 FEniCS-有限元法相比，使用 3 个和 5 个模态时，Model-B 的加速比分别超过 7,700 倍和 5,000 倍；然而，在加热层中，Model-B 仅使用 3 个模态就具有更高的精度（见表 III）。

值得一提的是，在实际的动态热管理中，没有必要获取每个时刻的温度分布。因此，PODTherm-GP 根据式（3）恢复温度所需的计算时间（每隔十个时间步恢复一次）将与求解式（5）中的常微分方程所需的时间处于同一数量级。与需要将整个时间域和空间域中的温度都纳入仿真的直接数值模拟不同，如果只需要在某一时间段内获取高温区域中少数几个时刻的温度，则 PODTherm-GP 的计算速度至少还会比上述讨论的结果再快一个数量级。

本研究的发现强烈表明，如果 POD 模型基于采用更精细网格的 DNS 构建，那么它的热预测精度可能高于采用精细网格的 DNS。尽管训练时间会更长，但在计算速度至少提升 3 个数量级的同时，POD 模型的精度也将优于 DNS。在动态热管理的实际应用中，POD 模型至少可节省 4 个数量级的计算时间，因为只需预测仿真时间跨度和空间域中很小一部分范围内的温度。

与其他用于解决复杂问题的学习方法类似，当将 PODTherm-GP 应用于具有大量核心的 CPU 或 GPU 的热仿真时，POD-Galerkin 方法会因收集用于 POD 模态训练的数据而面临高昂的计算开销。为解决具有数百或数千个核心的处理器的训练问题，可以采用区域分解，将处理器划分为许多较小的子区域，并对每个子区域

表 III 相对于更细网格 FEniCS 有限元仿真的最小二乘误差

Case	**4.35** ms						**6.09** ms					
模拟器	FEniCS 粗网格	POD Model-B（模态数）					FEniCS 粗网格	POD Model-B（模态数）				
		3	5	7	11	13		3	5	7	11	13
加热层（%）	0.80	0.76	0.59	0.52	0.51	0.51	0.72	1.14	1.05	0.83	0.75	0.61
整个芯片（%）	1.63	2.10	1.98	1.95	1.93	1.93	1.51	2.55	2.46	2.22	2.09	1.86

表 IV CPU 热仿真的计算时间

Mesh	**计算时间**（s）								
$256 \times 256 \times 14$	POD Model-A	POD 模态数								
		1			3			5		
		ODEs	Post1	Post2	ODEs	Post1	Post2	ODEs	Post1	Post2
		0.075	0.572	1.795	0.082	1.098	4.034	0.089	1.677	7.011
	FEniCS-FEM	2.288×10^4								
$1104 \times 620 \times 14$	POD Model-B	POD 模态数量								
		1			3			5		
		ODEs	Post1	Post2	ODEs	Post1	Post2	ODEs	Post1	Post2
		0.079	1.474	4.688	0.081	2.885	10.604	0.101	4.406	18.400
	FEniCS-FEM	2.247×10^5								

分别进行训练，从而大幅减少所需的工作量。未来将进一步研究这种方法以及不同的训练策略。

VII. 结论

本文针对一款18核CPU——Intel Xeon E5-2699 v3，研究了PODTherm-GP这一架构级热仿真方法的准确性、效率和鲁棒性。该方法基于本征正交分解（POD），并将传热方程通过 Galerkin 投影映射到POD模态上；这些模态由准确的 FEniCS-FEM 仿真所采集的热数据训练得到。所选CPU在 FEniCS-FEM 热仿真中使用的动态功率图由 gem5 和 McPAT 结合 SPLASH-2基准测试套件生成。研究表明，训练热数据的质量，尤其是训练热数据的热梯度所产生的已训练POD模态梯度的准确性，是决定PODTherm-GP准确性的关键。这种数据驱动学习方法只需极少数量的POD模态（因而自由度也极少），即可捕捉训练数据中蕴含的关键时空热信息。训练数据质量足够时，与有限元仿真相比，PODTherm-GP能够非常准确地预测动态温度分布，同时将自由度至少减少5个数量级、计算速度至少提升3个数量级。由于实时热管理只需获取特定时间步高温区域的温度，PODTherm-GP可将计算时间缩短超过4个数量级，因此具有实际应用价值。

此外，本研究展示了 PODTherm-GP 在训练范围之外的能力。结果表明，如果纳入更多模态，PODTherm-GP 能够准确预测所选 18核CPU 的动态热分布，即使仿真时间比训练时间长 40%。最后，一项有趣的发现表明，如果训练数据由更高分辨率的直接数值模拟生成，PODTherm-GP 可能比高分辨率直接数值模拟提供更准确的预测。如果能够承担训练所需的计算开销，并且大规模全芯片热仿真的准确性和效率是主要考虑因素，那么值得通过具有极高空间分辨率的直接数值模拟来收集训练热数据。

参考文献

[1] S. I. Guggari，“热性能指标分析——在高性能计算服务器 CPU 冷却中的应用”，*IEEE*计算机组件、封装与制造技术汇刊，第11卷，第2期，第222–232页，2021年。[2] A. Heinig、R. Fischbach 和 M. Dittrich，“采用宽I/O存储器的2.5D和3D集成系统的热分析与优化”，载于*ITHERM*会议论文集，第86–91页，IEEE，2014年。[3] J. Zhou、J. Yan、等，“异构多处理器片上系统中面向热感知的实时任务相关双层调度及处理器能耗降低”，系统架构期刊，第82卷，第1–11页，2018年。[4] A. Silva、I. Weber、等，“利用抽象建模实现众核系统的动态热管理”，载于*ISCAS*会议论文集，第1–5页，2021年。[5] N. Revathi 和 G. Sumathi，“面向芯片多处理器主动热管理的多步温度预测”，超级计算期刊，第77卷，第8期，第8967–8994页，2021年。[6] Y. G. Kim 、 M. Kim 、 J. Kong 和 S. W. Chung，“ 一种面向异构多核处理器的自适应热管理框架”，*IEEE*计算机汇刊，第69卷，第6期，第894–906页，2020年。[7] L. Jiang、A. Dowling、Y. Liu 和 M.-C. Cheng，“探索一种高效的多核 CPU 架构级热仿真方法”，载于*ISCAS*会议论文集，第1–5页，2022年。[8] W. Jia 和 M.-C. Cheng，“一种通过结合材料属性变化的模型降阶实现互连线热仿真的方法”，计算机科学期刊，第61卷，第101665页，2022年。[9] W. Jia、B. T. Helenbrook 和 M.-C. Cheng，“利用适当正交分解对多鳍场效应晶体管结构进行热建模”，*IEEE*电子器件汇刊，第61卷，第2759页，2014年，第8期，第2752页。[10] “FEniCS项目。” https://fenicsproject.org/。访问日期：2023年4月9日。[11] N. Binkert、B. Beckmann、等，“Gem5模拟器”，*SIGARCH*计算机体系结构新闻，第39卷，第2期，第1–7页，2011年。[12] S. Li、J. H. Ahn、等，“McPAT：面向多核和众核架构的集成功耗、面积和时序建模框架”，载于*MICRO*会议论文集，第469–480页，2009年。[13] A. Sridhar、A. Vincenzi、M. Ruggiero、等，“3D-ICE：采用层间液冷的3D集成电路快速紧凑瞬态热建模”，载于*ICCAD*会议论文集，第463–470页，IEEE，2010年。

[14] P. Bogdan、R. Marculescu 和 S. Jain，“多域片上系统平台的动态功耗管理：一种最优控制方法”，ACM《设计自动化与电子系统汇刊》，第18卷，20，2013年4，1第。页码 – 。[15] A. K. Cos.kun、J. L. Ayala、D. Atienza 和 T. S. Rosing，“液冷3D堆叠架构的热建模与管理”，载于《VLSI-SoC：系统集成技术》，第34–55页，Springer，2011年。[16] T.-Y. Wang 和 C. C.-P. Chen，“3-D thermal-ADI：一种线性时间芯片级瞬态热仿真器”，《IEEE计算机辅助设计与集成电路系统汇刊》，第21卷，第12期，第1434–1445页，2002年。[17] G. Xu，“多核处理器的热建模”，载于 ITHERM 会议论文集，第96–100页，IEEE，2006年。[18] K. R. Vaddina、A.-M. Rahmani 等，“先进3 257、2012 30、248 d堆叠结构的热建模与分析”，《Procedia Eng.》，第卷。页码 – 。[19] “AMD。” https://www.amd.com/en/products/professional-graphics/amd-radeon-pro-w6800 product-specs。访问日期：2021-05-21。[20] Y.-W. Wu、C.-L. Yang 等，“综合考虑热因素的架构与物理设计空间联合探索”，载于 ISLPED 会议论文集，第123–126页，2005年。[21] “HotSpot 6.0 温度建模工具。” http://lava.cs.virginia.edu/HotSpot。访问日期：2023年4月9日。[22] E. Glocker 和 D. Schmitt-Landsiedel，“多核处理器系统中的温度场景建模”，《Adv. Radio Sci.》，第11卷，第D. 1期，第219–225页，2013年。[23] A. Florea、C. Buduleci、R. Chis. 等，“通过热测量增强 Sniper 仿真器”，载于 ICSTCC 会议论文集，第31–36页，IEEE，2014年。[24] A. Pathania 和 J. Henkel，“HotSniper：用于开放系统中多核热仿真的基于Sniper的工具链”，《IEEE Trans. Embedded Syst. Lett.》，第11卷，第2期，第54–57页，2018年。[25] A. Hankin、D. Werner、M. Amiraski、J. Sebot、K. Vaidyanathan 和 M. Hempstead，“HotGauge：一种用于表征现代及下一代处理器中先进热点的方法”，载于 IISWC 会议论文集，第163–175页，IEEE，2021年。[26] L. Siddhu、R. Kedia、S. Pandey、M. Rapp、A. Pathania、J. Henkel 和 P. R. Panda，“CoMeT：适用于2D、2.5D和3D处理器-内存系统的集成区间热仿真工具链”，《ACM Trans. Archit. Code Optim.》，第19卷，第3期，第1–25页，2022年。[27] D. Fetis 和 P. Michaud，“HotSpot-3.0基于块的温度模型评估”，载于 WDDD 会议论文集，2006年。[28] W. Huang、K. Sankaranarayanan 等，“HotSpot 4.0中考虑粒度的改进型基于块的热模型”，《Proc. Int. Cond. WDDD》，第7卷，2007年。[29] H. Sultan、A. Chauhan 和 S. R. Sarangi，“芯片级热仿真器综述”，《ACM Comput. Surv.》，第52卷，第2期，第1–35页，2019年。[30] Y. Zhan 和 S. S. Sapatnekar，“高效的基于格林函数的热仿真算法”，《IEEE计算机辅助设计与集成电路系统汇刊》，第26卷，第9期，第1661–1675页，2007年。[31] H. Sultan 和 S. R. Sarangi，“一种适用于3D芯片的快速、感知漏电的基于格林函数的热仿真器”，《IEEE Trans. Very Large Scale Integr. (VLSI) Syst.》，第28卷，第11期，第2342–2355页，2020年。[32] S. Varshney、H. Sultan、P. Jain 等，“Nanotherm：一种用于全芯片热仿真的解析傅里叶-玻尔兹曼框架”，载于 ICCAD 会议论文集，第1–8页，IEEE，2019年。[33] A. Ziabari、J.-H. Park 等，“功率模糊：适用于封装集成电路和功率器件的快速静态与瞬态热分析方法”，《IEEE Trans. Very Large Scale Integr. (VLSI) Syst.》，第22卷，第11期，第2366–2379页，2014年。[34] Y. Zhan 和 S. S. Sapatnekar，“一种高效的全芯片热仿真算法”，载于 ICCAD 会议论文集，第635–638页，IEEE，2005年。[35] K. Zhang、A. Guliani 等，“基于机器学习的温度预测，用于系统组件的运行时热管理”，《IEEE Trans. Parallel Distrib. Syst.》，第29卷，第2期，第405–419页，2017年。[36] A. Sridhar、A. Vincenzi 等，“基于神经网络的GPU集成电路热仿真”，《IEEE计算机辅助设计与集成电路系统汇刊》，第31卷，第1期，第23–36页，2012年。[37] Z.-L. Qian、D.-C. Juan、P. Bogdan、C.-Y. Tsui、D. Marculescu 和 R. Marculescu，“适用于片上网络（NoC）架构的基于支持向量回归（SVR）的延迟模型”，《IEEE计算机辅助设计与集成电路系统汇刊》，第35卷，第3期，第471–484页，2015年。[38] J. L. Lumley，“非均匀湍流的结构”，《大气湍流与无线电波传播》，1967年。[39] G. Berkooz、P. Holmes 和 J. L. Lumley，“本征正交分解在湍流分析中的应用”，《流体力学年度评论》，第25卷，第1期，第539–575页，1993年。[40] J. Weiss，《本征正交分解教程》，载于*AIAA* 航空 *2019* 论坛，第 3333 页，2019 年。[41] H. Mujtaba，《Intel Xeon E5-2600 V3 “Haswell-EP” 工作站和服务器处理器，为高性能计算蓄势待发》。 https://wccftech.com/intel-xeon-e52600-v3-haswellep-workstation-server-processors-unleashed-highperformance-computing。访问日期：2021-05-21。[42] R. Jongerius、A. Anghel、等，《用于快速设计空间探索的多核处理器解析模型》，*IEEE*计算机汇刊，第 67 卷，第 6 期，第 755–770 页，2017年。[43] S. C. Woo、M. Ohara、等，《SPLASH-2 程序：特征分析与方法学考量》，*SIGARCH*计算机体系结构新闻，第 23 卷，第 2 期，第 24–36 页，1995 年。[44] G. Oxman 和 S. Weiss，《支持偏转路由、GPU/CPU集成和协同仿真的片上网络模拟器》，*IEEE*计算机辅助设计与集成电路系统汇刊，第 35 卷，第 10 期，第 1667–1680 页，2016 年。[45] L. Sirovich，《湍流与相干结构的动力学：I. 相干结构》，应用数学季刊，第 45 卷，571，1987 3，第 561 页。