



上海大学

SHANGHAI UNIVERSITY

毕业设计（论文）

UNDERGRADUATE PROJECT (THESIS)

题目： 基于物理深度学习的微气泡系统动力学预测

学 院： 力学与工程科学学院

专 业： 理论与应用力学

学 号： 17121632

学生姓名： 翟晗锋

指导教师： 胡国辉

起讫日期： 2021.1.18 - 2021.6.18

基于物理深度学习的微气泡系统动力学预测

摘要

微气泡及气泡群在微管道中的流动在生物医学、化工、工程热物理领域有着极其广泛的应用和研究。微环境下气泡运动涉及到单气泡流动变形，气泡间的相互融合及剥离以及两相流作用等复杂力学问题，是学术界研究的前沿问题。为研究微流体环境下气泡运动，本文设定两算例进行探究：(1) 微管道中单气泡流动变形，(2) 微气泡群流动过程。相关参数基于生物学应用背景给出。仿真结果显示单气泡流动变形形态为前段先外凸呈抛物线状，尾端内凹，直至附着气泡剥离气泡主体；微气泡群会随着时间推进出现融合及剥离现象。我们计算得到的单气泡形态与微管道实验中血红细胞变形形貌相似，管壁尾端在液相流部分符合泊肃叶流动基本速度分布。微气泡群算例整体流体速度分布符合泊肃叶流动规律。对这两个算例，我们通过对两相组分随时间的变化进行分析，印证算例中各组分质量守恒，并对产生误差的原因进行了分析。

多相流的数值模拟对计算精度需求很高，若网格精度不够则会导致计算过程中某一相流体出现组分丢失。因此，用数据驱动的方法重构流场很有必要。随着机器学习快速发展，物理深度学习在近些年增长迅猛。基于物理神经网络（PINN）的精神，我们提出了一种全新的、可用于预测气泡动力学的神经网络结构 BubbleNet。该网络由两部分构成：基本的深度神经网络（DNN）以及包含连续不可压条件的内置方程（物理信息）。同时，我们还引入了时间离散监督器，以对流场数据进行时间离散化的归一化。我们分别用 DNN 和 BubbleNet 预测了单气泡在 $2000\mu\text{s}$ 和多气泡系统在 $1500\mu\text{s}$ 时刻物理场 (u, v, p, ϕ) 的分布。结果表明，BubbleNet 结构相较于传统的 DNN 可以以更小的迭代步数获得更高的预测精度。因为对整个时空域上不同时间步流场数值的较大差异可能会导致特定时间步特征被“剥夺”，使得训练神经网络时让网络被“欺骗”，所以时间离散归一器使网络预测时空数据更准确。这些算法和成果不仅可应用于微气泡流，更可以广泛应用于电化学、电磁、燃烧等具有工程应用背景的数学物理问题。

关键词：机器学习；物理神经网络；多相流；气泡动力学；微流体.

Predicting micro-bubble system dynamics with physics-informed deep learning

Abstract

Micro-bubbles and bubble systems are widely applied in biophysics, medical engineering, chemical physics, etc. Microfluidic bubble movements involve the deformation of single bubbles, collision, mixture, and rupture of bubbles, etc. These complex problems pose a real challenge for studying such. To investigate the mechanism of micro-bubbles dynamics, we here set up two cases: (1) single bubble movement confined in micropipes and (2) micro-bubbles group movement in microfluidics. The related parameters are given referred to their biological counterparts. Simulation results indicate that a single bubble front interface will flow outwards as an increasing parabolic shape while the tale gets flow inwards till two attached smaller bubbles get ruptured from the main bubble. Grouped bubbles collide to be fused or ruptured. The single bubble deformation agrees with the red blood cell deformation shape confined in the pipe experiment. The fluid phase velocity distribution agrees with the general distribution of the Poiseuille flow. For both cases, we analyze the gas-liquid component to ensure the correct calculation of simulations and estimate the possible errors.

Multiphase flow simulation requires high computation accuracy due to possible component losses that may be caused by sparse meshing during iterations. Hence, data-driven methods can be adopted as a useful tool. Based on physics-informed neural networks (PINN), we proposed a novel deep learning architecture BubbleNet, which entails two main parts: a deep neural net (DNN) and an inner-contained continuum equation as physics information of incompressible continuum fluid equations. We also elicit Time Discretized Normalizer (TDN), an algorithm to normalize field data per time step before training. We applied and DNN and BubbleNet algorithms, respectively, to predict the physical distribution (u , v , p , ϕ) of the bubble movement. For the two cases, we aim to predict the distribution at $2000\mu\text{s}$ for a single bubble case and $1500\mu\text{s}$ for multi

bubbles case. Our BubbleNet algorithms exhibit higher accuracy for predictions and require fewer iterations. The TDN can effectively improve the training accuracy due to the data scale variance among the time-space domain that may 'confuse' the training of the NN. The proposed algorithms can be applied to various fields including electrochemistry, electromagnetic, combustion, and related mathematical physics problems with wide engineering backgrounds.

Keywords: Machine learning, physics-informed neural network, multiphase flow, bubble dynamics, microfluids.

目 录

摘要	I
Abstract	II
第1章 绪论	1
1.1 本文研究背景与意义	1
1.2 机器学习与人工智能	1
1.3 机器学习在计算流体力学领域的应用	2
1.4 多相流与气泡动力学	4
1.5 本文的主要工作	4
第2章 理论背景	5
2.1 机器学习	5
2.1.1 神经网络	5
2.1.2 物理深度学习	8
2.2 多相流与气泡流动	10
2.2.1 流体力学基本理论	10
2.2.2 多相流理论	11
2.3 本章小结	13
第3章 微气泡流动	14
3.1 模型构建	14
3.1.1 Level set 数值模拟算法	14
3.1.2 单气泡模型	17
3.1.3 复杂多气泡系统	19
3.2 动力学分析	20
3.2.1 单气泡计算结果	20
3.2.2 多气泡计算结果	25
3.3 本章小结	28
第4章 数据驱动的两相流预测与学习方法	29
4.1 优化函数选取	29
4.1.1 Adam 优化器	29

上海大学本科生毕业设计（论文）

4.1.2 L-BFGS-B 优化器	31
4.2 DNN: 深度人工神经网络	33
4.3 BubbleNet: 气泡物理神经网络	35
4.3.1 TDN 的使用与验证	37
4.3.2 数据粗化的定性分析	37
4.4 训练过程与计算结果	41
4.5 本章小结	54
结 论	57
附 录	67
附录 A N-S 方程的推导	67
附录 B 多相流密度公式的推导	68
致 谢	70

第1章 绪论

1.1 本文研究背景与意义

近年来，机器学习和深度学习因为其在学术界、工业界甚至娱乐界的强大表现，以及其针对特定问题的精度和解决问题的广度，越来越受到人们的关注以及资本的投资。同时，在计算物理领域，机器学习也在快速地革新着传统研究的方法与范式。但是机器学习在该领域内却并没有互联网、社交软件中使用的那么广泛和深入。大量的工业科学技术应用涉及到计算科学领域，包括计算流体力学的算法。其中，多相流，特别是气泡流，在生物、医学、化学工程等领域有着大量的应用价值，同时也具有相对完善的理论框架。本文就微观尺度下的气泡流动进行研究，通过物理深度学习的方法对其进行学习预测。

机器学习是一个通过经验和数据，可以自动提高程序性能的计算机算法^[1]。近年来，随着科技不断进步，计算资源和硬件的不断提升，以及大量数据的易于获取，机器学习研究领域取得飞速进展和进步。同时，随着 AlphaGo 的横空出世^[2, 3]，又随着机器学习和深度学习以其巨大的潜能和其在医疗^[4-6]、公共卫生^[7, 8]、商业广告^[9, 10]等领域取得的巨大成功，针对不同问题提出的机器学习算法不断更新出现，并推动着机器学习的进步。

经过专家、学者的大量研究和努力，机器学习在各领域都取得了成功。在医疗领域中，AlphaFold 可以自动识别蛋白质结构的能力，更是颠覆传统医学的限制^[11]；同时，对于新型冠状病毒（SARS-CoV-2）的 CT 感染预测也使得机器学习快速、通用性强、应用广泛等优点得以充分体现^[12, 13]。在公共卫生领域，随着 COVID-19 疫情的爆发，基于机器学习的整体疫情传染预测更是为疫情发展提供重要参考^[14, 15]。在商业广告领域，机器学习基于用户大量数据的训练和学习使其可以精准、有效给特定用户群体推送广告，极大地促进商业发展^[16-18]。同时，基于 Google 开发的深度学习开源平台 TensorFlow^[19] 已经大量的被国内外知名公司机构使用。

1.2 机器学习与人工智能

随着互联网的蓬勃发展和智能电子设备的普及化，数据正在以前所未有的速度增长^[20, 21]。基于海量数据下，机器学习算法得以发挥其预测能力。基于大数据的深度

学习(Deep Learning, DL)广受关注^[22]。人工神经网络(Artificial Neural Networks, ANN)作为最成熟，最有效的学习方法之一，更是被大量地研究^[23, 24]，其中最为简单且得到广泛应用的人工神经网络框架是前向传播神经网络(Feed-forward Neural Networks, FNN)^[25]。同时，循环神经网络(Recurrent Neural Networks, RNN)因其可以提取序列中随时间变化的特征等优势^[26]，被应用在自然语言处理(Natural Language Processing, NLP)^[27]、计算机视觉(Computer Vision, CV)^[28]、和计算生物学(Computational Biology, CB)^[29]等领域。针对图像识别和图像处理问题，卷积神经网络(Convolutional Neural Networks, CNN)得到了更加广泛的认可^[30]。更为重要的是，基于CNN开发的AlexNet因其精准和有效使得CNN被更多计算机视觉领域的研究人员青睐^[31]。针对不同的图像处理方法，基于对图片特征进行识别和分割的U-Net也成为计算机视觉领域得力工具之一^[32]。基于以上深度学习算法和工具，Google公司于2015年发布开源深度学习计算平台TensorFlow^[19]，其被广泛应用于工业和学术界。此后的2016年，Facebook AI实验室也紧接着发布了深度学习开源平台PyTorch^[33]。现如今，这两大平台已成为计算机机器学习领域内研究人员广泛使用和开发的工具。

1.3 机器学习在计算流体力学领域的应用

流体力学最早起源于古希腊时期亚里士多德和阿基米德对浮力和流体静力学的研究^[34, 35]。经过长时间的发展，经历从欧拉到 Claude-Louis Navier 和 Sir George Gabriel Stokes 提出 Navier-Stokes (N-S) 方程，流体力学逐渐具有相对完备的理论基础。但是，因为 N-S 方程的强非线性，运用解析求解的方法只适用于特定问题。所以科学家们提出了各种离散方法对其进行数值求解。上世纪，大量的计算方法，如有限差分(Finite Difference Method, FDM)^[36, 37]、有限体积(Finite Volume Method, FVM)^[38-42]、有限元(Finite Element Method, FEM)^[43]、光滑粒子(Smoothed Particle Hydrodynamics, SPH)^[44, 45]等方法得到快速发展。这些方法用不同格式和算法离散 N-S 方程，其优势是物理信息准确，方程确定。但是其缺点是计算格式复杂，对于工程实际中的非线性问题往往难以同时兼顾计算速度和精度。

近年来，对于不同的物理问题，计算科学研究人员开发了各种基于机器学习和数据驱动的计算物理算法。2016年，Rudy et al.^[46]提出了通过数据驱动拟合物理控制方程的算法(PDE-FIND)，颠覆了传统通过实验、经验和推理寻找物理控制方程的思路。2019年，Raissi et al.^[47-49]提出了基于物理场数据驱动的神经网络(Physics-

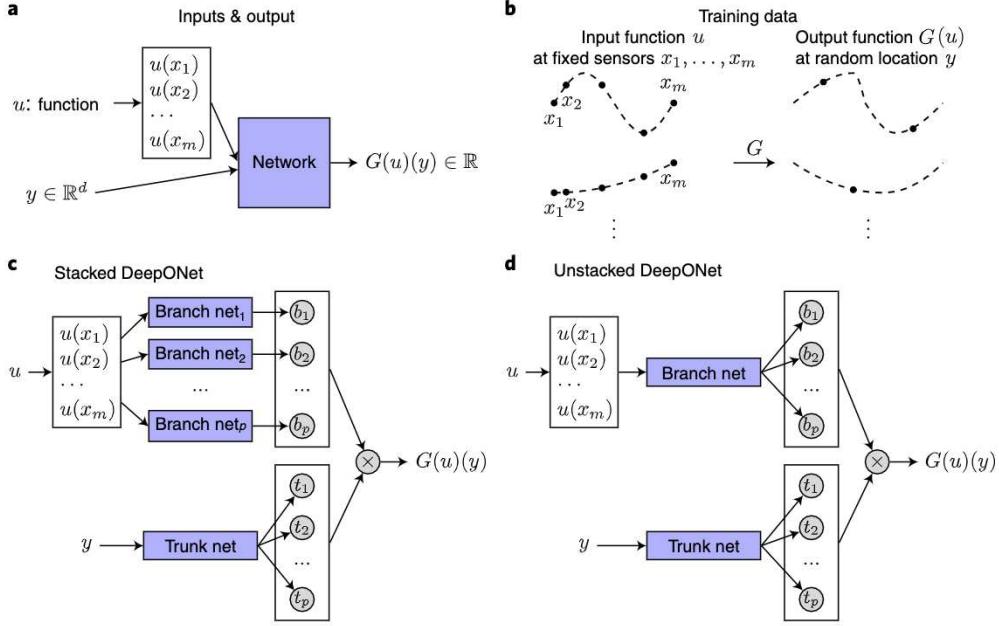


图 1-1 DeepONet 的结构示意图^[51]

Informed Neural Network, PINN)。2020 年, 基于 PINN, Lu et al. 又提出了新的基于物理数据驱动的深度学习模型 DeepXDE, 用以求解偏微分方程控制的物理问题^[50]。仿效 PDE-FIND 的思路, 并且基于 DeepXDE 的研究成果之上, 2020 年, Lu et al.^[51] 提出利用深度神经网络可学习得到物理系统的控制方程的框架 DeepONet。DeepONet 基于单隐层神经网络可以逼近非线性算子的基本思路上开展。它主要分为两种, 针对每个物理输出变量都设置子神经网络的 Stacked DeepONet 和所有物理输出公用一个神经网络参数的 Unstacked DeepONet, 如图1-1所示。同时, DeepONet 已被应用于生物学, 物理学等工程问题^[52-54]。因 PINN 局限于特定物理问题, 2020 年, Li et al.^[55] 提出了通过傅立叶神经算子 (Fourier Neural Operator, FNO), 即对空间映射进行学习的深度学习方法, 且具有快速高效和低误差的特点。基于前文提到的 CNN, 利用计算机视觉技术的物理深度学习模型也在近些年逐渐流行。2020 年, Jiang et al.^[56] 提出了通过对三维场信息 (二维物理场随时间变化) 进行卷积学习, 可以实现模糊物理场的加密功能 (MeshfreeFlowNet)。实际上, 相关的物理深度学习模型近些年快速增加, 本节仅能简述其概貌。

1.4 多相流与气泡动力学

多相流，特别是气泡流是一个经典流体力学问题^[57]。它具有非常广泛的应用背景，其中也涵盖大量的工程实际问题，包括悬浮的谷物粉尘或煤粉，液滴和喷雾，推进剂燃烧，炭化、烟灰、烟雾形成，淤浆，液体中的气泡，雨水和沉降等^[58]。在物理上，我们经常通过质量平均混合速度和扩散通量来表示混合流动速度和单相速度的区别^[59]。在大量多相流物理现象中，气泡流动是实际应用中常常出现的现象。因为气泡流具有大量的生物学和物理化学的应用背景，其已经成为流体力学乃至生物物理领域中主要问题之一，其中针对管道气泡流研究众多。毛细管道内气泡的“剥离”现象（Bubble Pinch-Off）的理论与实验分析是该领域内最主流的问题之一^[60-64]。微气泡在管道中的运动可以广泛地应用于靶向输药^[65, 66]，特别是血脑屏障药物输送的研究中^[67-69]。同时，因为多气泡系统的复杂性以及广泛的生物应用，多气泡群体动力学在不同物理场下气泡的相互作用表现出复杂性与随机性^[70, 71]。因此，基于上述认识，在本文中作者针对两种情形下的管道流动进行研究，一种为微管道内部单气泡流动，其中计算模型参考了 Hosseinkhah et al.^[69] 的工作。另一种为微管道内多气泡的系统动力学，计算模型参考了 Lea et al.^[72] 提出的思路，即管道几乎完全充满液体，游离气体以小气泡的形式存在，并在液体中运动。

1.5 本文的主要工作

本文中，作者在第一章简要概述了人工智能和机器学习领域的现状和研究成果以及论文启发。在第二章中简要概述了本文中涉及的理论背景和数学基础，其中包括神经网络的基本构建，物理神经网络的基本思路和实现，流体力学和两相流的基本理论和数学推导等。在第三章中作者介绍了本文中用到的两个气泡流算例：单气泡流动和气泡群流动的建模，相关参数给定方法、计算方法和仿真模拟计算结果分析以及结果验证。在第四章中作者介绍了利用普通深度神经网络预测气泡流的方法以及本文中提出的全新的可预测气泡流动的物理神经网络结 BubbleNet，以及两种神经网络预测结果分析。最后作者给出相关结论，并在附录中补充了一些相关数学模型的理论推导。

第2章 理论背景

2.1 机器学习

2.1.1 神经网络

在前文中，我们简要概括了神经网络的历史和概况。本章我们着重介绍神经网络数学基础。人工神经网络是基于模拟生物学神经元之间传播原理启发构建的计算系统^[73]，本质上是一个多重非线性回归模型。

一个基本的人工神经单元结构如图2-1所示：其中 x_1, x_2, \dots, x_n 为输入量， w_1, w_2, \dots, w_n 为拟合输入量的权重，经过阈值 b ，神经元输出为 y 。对于神经网络来说，基本的神经元模型是基于线性多重回归模型所建立的：

$$y_w(x) = w_0 + w_1x_1 + w_2x_2 + w_3x_3 + \dots + w_nx_n + b \quad (2-1)$$

其中 w_n 为回归权重， b 为阈值。

基于回归模型，每个神经元经过激活函数被“激活”传入下一层神经元中：

$$\hat{y}(x) = \sigma(y_w) = \sigma(w^T \mathbf{x} + b) \quad (2-2)$$

其中 $\hat{y}_i = (\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n)$ 可看作网络产生的预测值， $\sigma = \sigma(z)$ 为激活函数。

激活函数作为将输入信息通过非线性变换传输给另一个神经元的工具，选取和使用主要依靠经验。在实际应用中，常用的三种激活函数有 sigmoid，ReLU 和线性激活（图2-2）：

$$\sigma_{sigmoid}(z) = \frac{1}{1 + e^{-z}}; \quad \sigma_{ReLU}(z) = max(0, z); \quad \sigma_{linear}(z) = z \quad (2-3)$$

在实际运算中（如 TensorFlow, PyTorch, Keras 等主流深度学习平台），神经网络的初始权重是随机生成的，而后应用梯度下降方法，进行优化、拟合训练集。权重的

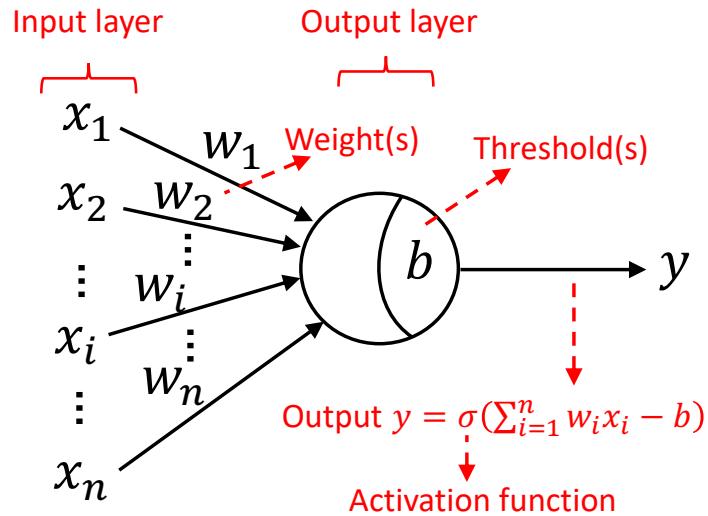


图 2-1 基本神经元的结构图^[74]。

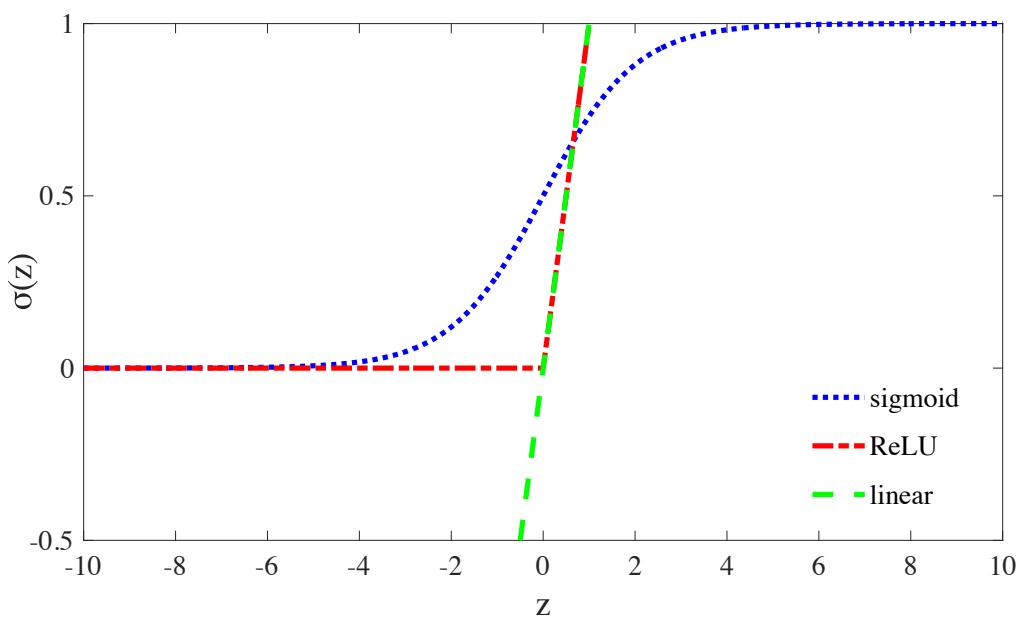


图 2-2 三种激活函数的示意图。

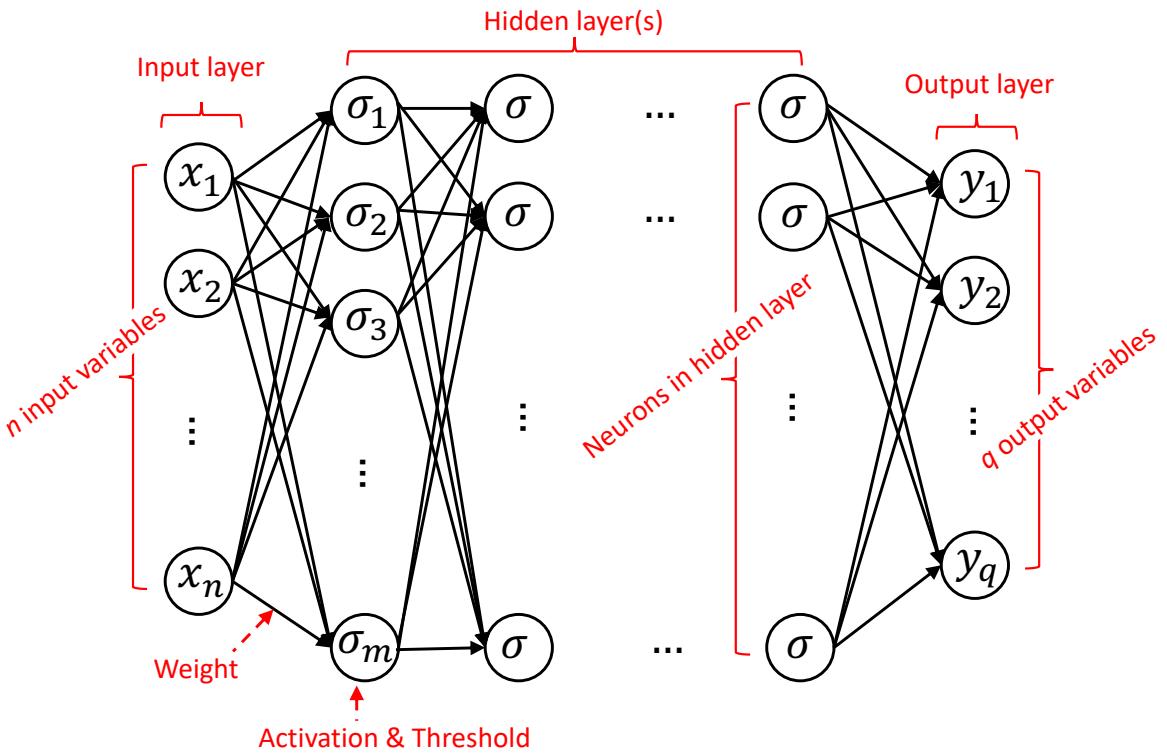


图 2-3 深度神经网络的结构图。

迭代过程可以写作：

$$w_i \leftarrow w_i + \Delta w_i; \quad \Delta w_i = l_r(y - \hat{y})x_i \quad (2-4)$$

其中 l_r 为网络的学习率。基于方程2-4，我们通过设置学习率，可以改变权重的更新迭代过程，以此控制回归模型训练的快慢和准确度。

在神经网络训练中，我们通常采用均方根误差（MSE）作为损失函数来训练；而训练迭代神经网络的过程就是通过不断减小 MSE 的过程实现的。

$$MSE = \frac{1}{M} \sum_{i=1}^M (y_i - \hat{y}_i)^2$$

在训练中，MSE 在前后迭代的不断减小过程会产生误差，而误差则是通过反向传播（back-propagation）算法传递给上一层网络进行迭代，迭代后会更新权重 w_n 促使网络再次进行迭代。

• 训练过程

现设置一深度神经网络 DNN，由前文定义， $\hat{y}_i = (\hat{y}_1, \hat{y}_2, \dots, \hat{y}_n)$ 可看作为网络的输出值（预测值）。在此我们将其写为：

$$\hat{y}_{nj} = \sigma(\beta_j - b_j) \quad (2-5)$$

其中 β_j 是第 j 个神经元的输入值， b_j 为阈值。

在本章中我们采取梯度下降方法为例，在 DNN 中连接第 h 和 j 隐层的权重因此可以写作：

$$\Delta w_{hj} = -l_r \frac{\partial MSE}{\partial w_{hj}} \quad (2-6)$$

而方程2-6 可以进一步展开为：

$$\frac{\partial MSE}{\partial w_{hj}} = \frac{\partial MSE}{\partial \hat{y}_j} \frac{\partial \hat{y}_j}{\partial \beta_j} \frac{\partial \beta_j}{\partial w_{hj}} \quad (2-7)$$

由 β 的定义，我们可以写出： $\frac{\partial \beta_j}{\partial w_{hj}} = b_h$ 。因此，根据方程2-5以及均方根误差的定义，我们可以写出：

$$\begin{aligned} g_j &= -\frac{\partial MSE}{\partial \hat{y}_j} \frac{\partial \hat{y}_j}{\partial \beta_j} \\ &= -(\hat{y}_j - y_j) \dot{\sigma}(\beta_j - b_j) \\ &= \hat{y}_j(1 - \hat{y}_j)(y_j - \hat{y}_j) \end{aligned}$$

因此，由上述方程，我们可以获得由隐层 h 传至隐层 j 的权重的更新方程：

$$\Delta w_{hj} = l_r g_j b_h \quad (2-8)$$

这便是神经网络训练更新过程的数学描述。

2. 1. 2 物理深度学习

2019 年，基于神经网络的梯度下降拟合参数原理，Raissi et al.^[47-49] 提出了基于物理信息的深度神经网络模型。在本文中，作者简称物理神经网络。其思路为：构造一个含有物理方程的损失函数 $Loss_{Phys}$ ，并将该函数代入神经网络整体的损失

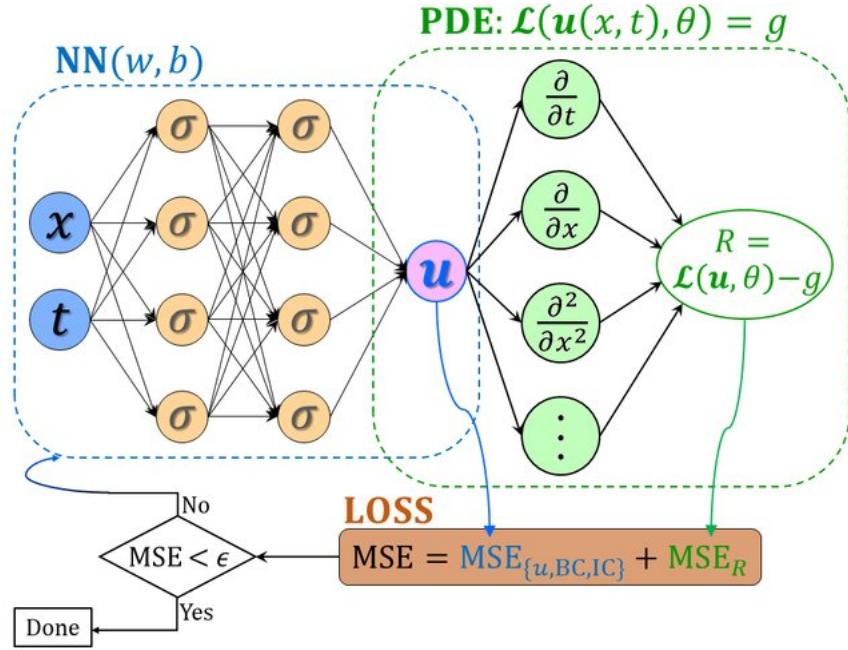


图 2-4 物理神经网络的示意图，来自 Meng et al., 2020^[75]。

函数中: $\text{Loss} = \text{Loss}_{NN} + \text{Loss}_{Phys}$ 。举例来说, 若我们采取均方根误差作为梯度下降指标, 采用 N-S 方程作为引入的物理方程, 那么该物理神经网络形式可写作 $\text{Loss} = MSE_{NN} + \rho D_t \mathbf{u} - \mathbf{f} + \nabla p - \mu \nabla^2 \mathbf{u}$ 。其中, MSE_{NN} 是普通神经网络迭代过程中的均方根误差, ρ 为流体的密度, D_t 为物质导数 (或随体导数), \mathbf{f} 为流体所受外力, p 为流体所受压强, μ 为流体粘性系数。因此在神经网络进行梯度下降训练时, 物理方程被作为一个“弱限制”加入了训练过程, 使得最后训练出来的结果逐渐逼近真实物理方程。

首先, 根据偏微分方程 (PDE) 的定义, 我们可以将一般偏微分方程写成如下形式:

$$u_t + \mathcal{N}[u; \mu] = 0, \quad x \in \omega, \quad t \in [0, T]$$

因此, 我们可以构建一含有 PDE 的函数 f , 并将其引入神经网络的迭代中:

$$f := u_t + \mathcal{N}[u]$$

在实际神经网络训练迭代过程中, 若采用均方根误差作为迭代标准, 损失函数

可写作^[47-49]:

$$MSE = MSE_u + MSE_f$$

其中

$$MSE_u = \frac{1}{N_u} \sum_{i=1}^{N_u} |u(t_u^i, x_u^i) - u^i|^2$$

$$MSE_f = \frac{1}{N_f} \sum_{i=1}^{N_f} |f(t_f^i, x_f^i)|^2$$

在本文中，我们要研究二维微气泡流动，因此采用物理神经网络的思路，将二维情况下的连续性方程 $u_x + v_y = 0$ 引入深度神经网络中，用于“物理限制”气泡流的训练过程。我们将对应的深度物理神经网络命名为 BubbleNet。

2.2 多相流与气泡流动

2.2.1 流体力学基本理论

流体力学是力学的一个分支，涉及流体在各种状态下的性质以及它们对作用在其上的力的反应^[76]。流体力学广泛应用涉及航空航天、车辆工业、生物医学等众多领域；在前文中我们已经对流体力学进行了简概介绍，在此不过多赘述。

流体力学的控制方程 Navier-Stokes (N-S) 方程可由 Lagrange 和 Euler 坐标系下分别推出^[77]。N-S 方程是由动量守恒推导而得出，具体推导过程详见附录 A。一般形式下的 N-S 方程写为：

$$\rho \left(\frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \mathbf{u} \right) = \mathbf{f} - \nabla p + \mu \nabla^2 \mathbf{u} \quad (2-9)$$

在本文中，我们主要考虑二维情况下微气泡的流动。在二维情况下，N-S 方程可写作：

$$\rho \left(\frac{\partial u}{\partial t} + u \frac{\partial u}{\partial x} + v \frac{\partial u}{\partial y} \right) = f_x - \frac{\partial p}{\partial x} + \mu \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

$$\rho \left(\frac{\partial v}{\partial t} + u \frac{\partial v}{\partial x} + v \frac{\partial v}{\partial y} \right) = f_y - \frac{\partial p}{\partial y} + \mu \left(\frac{\partial^2 v}{\partial x^2} + \frac{\partial^2 v}{\partial y^2} \right)$$

其中， x, y 为二维坐标系的坐标轴， t 为时间， ρ 为流体的密度， u, v 为流体的速度分量， f_x, f_y 为在两个方向上的外力， p 为压强， μ 为流体的粘性。

在实际描述流体运动时，除了动量方程，为满足基本物理定律，流体还应满足质量守恒（连续性），能量守恒以及状态方程。连续性方程可写为：

$$\frac{\partial \rho}{\partial t} + \nabla \cdot (\rho \mathbf{u}) = 0 \quad (2-10)$$

在二维不可压情况下，该方程可写作 $u_x + v_y = 0$.

能量方程写为：

$$\frac{\partial}{\partial t} (\rho E_{tot}) + \nabla \cdot [\mathbf{u} (\rho E_{tot} + p)] = 0 \quad (2-11)$$

其中 E_{tot} 为系统内的总能量。流体的状态方程写为：

$$p = \mathbb{S}(\rho, T)$$

其构建了流体中压强与密度和温度之间的关系。

上述方程构建了控制流体运动的基本物理规律。流体既包括水、油、空气等物质，也包括微颗粒物等广义的流体介质。但是值得注意的是，要利用流体理论建模，物质要满足连续性假设，以便在进行整体分析时，忽略基本单元（分子，原子等）的随机热运动。

2.2.2 多相流理论

在流体力学中，多相流是指具有两个或多个热力学相的材料同时流动^[78]。两相流是一种常见的多相流系统，其中包括气-液两相流、气-固两相流和液-固两相流。几种典型的气泡流，随着气-液组分占比不同形式如图2-5所示。

在本文中，我们主要考虑气液两相流中的气泡流（bubble flow），即气体占比较小的气-液两相流。

气泡流动中，气体和液体均满足上述流体力学基本方程，而气液界面则由表面

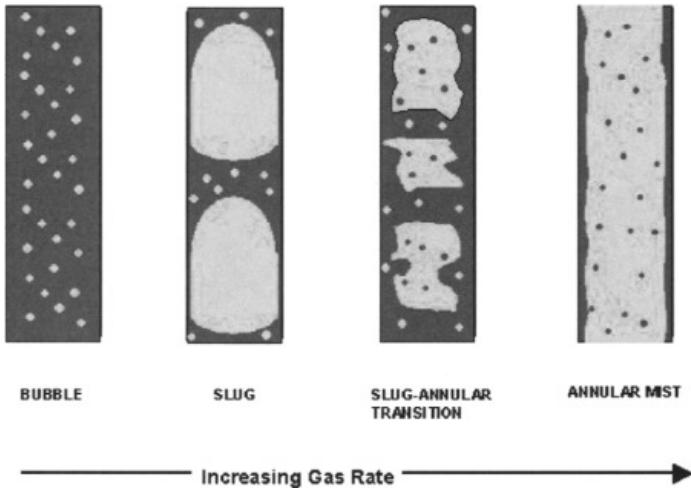


图 2-5 不同气-液两相流动示意图^[72]。

张力控制。在模拟仿真时，我们引入相函数 ϕ 描述两相运动，N-S 方程因此写为：

$$\frac{\partial \phi}{\partial t} + \nabla \phi \cdot \mathbf{u} = F$$

现设气、液的密度和粘性分别为 ρ_1, ρ_2 和 μ_1, μ_2 ，则气-液混合流体的密度和粘性可由相函数 ϕ 表示为：

$$\rho = \rho_1 + \phi(\rho_2 - \rho_1)$$

$$\mu = \mu_1 + \phi(\mu_2 - \mu_1)$$

对于气泡流，我们采取 COMSOL Multiphysics® 中的 Level Set 模型进行数值模拟，其中时间相关的控制方程写为：

$$\frac{\partial \phi}{\partial t} + \mathbf{u} \cdot \nabla \phi = \gamma \nabla \cdot \left(\epsilon_{ls} \nabla \phi - \phi(1-\phi) \frac{\nabla \phi}{|\nabla \phi|} \right)$$

其中 γ 为计算参数，为保证相函数梯度 $\nabla \phi$ 在时间迭代中始终保持在气液界面区域；而 ϵ_{ls} 则控制两相交界界面的区域厚度。在此处我们省略关于多相流的基本理论的内容，对于两相流组分关系数学模型的基本推导详见附录 B。

本章节中，我们简要介绍了物理神经网络的构建和数学基础以及多相流理论。下一章节中，我们将通过两个微气泡流动算例数值模拟的结果，分析微管道中气泡动力学的流动性质。

2.3 本章小结

在本章节中，作者首先简要介绍了在机器学习领域中最常用的数学逼近方法之一，神经网络。作者分别从其基本原理，构建方法和数学基础介绍讨论了神经网络的构建及训练方法。之后，作者又介绍了时下最流行和受关注的针对物理问题而产生的物理激发的神经网络的由来，基本构建思路及其训练方法。而后，作者先简单介绍流体力学基本理论及控制方程，N-S 方程的基本组成。之后，作者由流体力学理论出发，介绍了多相流理论的基本构架及我们在商业软件中计算气泡流使用的控制方程和相关参数物理意义。

第3章 微气泡流动

3.1 模型构建

为了探究微流体环境中气泡的运动形式，以及考虑到实际应用场景，我们本次共设置了两个算例：单气泡微管道流动和多气泡管道流。其中，我们研究单气泡管道流动的主要背景是（但不限于）血脑屏障中血管内气泡运动的研究^[67-69] 和红细胞的变形^[79, 80]；计算模型主要参考 Hosseinkhah et al.^[69] 的工作：单气泡被限制在管壁（血管壁）内运动。而多气泡模型则主要考虑（但不限于）组织液内多细胞运动^[81, 82] 以及药物输运^[66]，计算模型参考 Talu et al.^[66] 的实验以及 Lea et al.^[72] 的工作（在第一章节已介绍）。

3.1.1 Level set 数值模拟算法

在本文所涉及到的模拟中，我们使用了 COMSOL Multiphysics® 中用于计算多相流的时间依赖的 level set 算法。现我们给出 level set 算法简要介绍和其数学推导。

我们首先给定一空间函数 ϕ ，在三维空间上分布，如图3-1所示；假设有一点 $x = (x, y)$ 具有对时间依赖性，即 $x = x(t)$ ；在任意时间 t ，在设定的水平表面上 $x(t)$ 的高度 ϕ 应该为零：

$$\phi(x(t), t) = 0$$

在 level set 的众多应用中，函数 ϕ 可以代表很多含义，只要满足在我们给定水

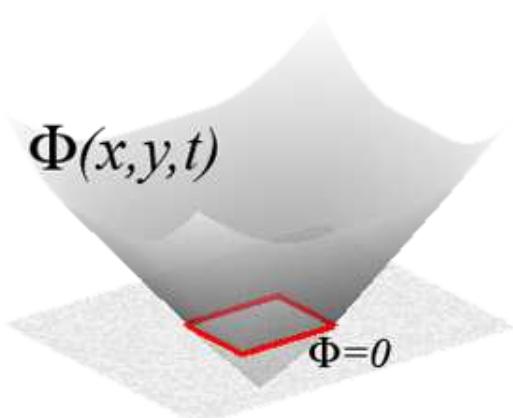


图 3-1 相函数 ϕ 在三维坐标分布示意图。

平面上其数值为零，就符合 level set 算法的设定。在这里， ϕ 特指描述两相流的相函数；其中 $\phi = 0$ 代表纯液体（给定定义液体水平面）， $\phi = 1$ 代表纯气体。现我们假定在时刻 $t = 0$ 存在初始值 ϕ ，通过运动方程 $\partial_t \phi$ 我们可以得到任意时刻的 ϕ 值；通过使用链式法则：

$$\begin{aligned}\frac{\partial \phi(x(t), t)}{\partial t} &= 0 \\ \frac{\partial \phi}{\partial x(t)} \frac{\partial x(t)}{\partial t} + \frac{\partial \phi}{\partial t} \frac{\partial t}{\partial t} &= 0 \\ \frac{\partial \phi}{\partial x(t)} x_t + \phi_t &= 0\end{aligned}$$

现在，我们考虑到 $\partial_x \phi = \nabla \phi$ ，速度变量 x_t 可以通过垂直于表面的力 F 计算得到。我们定义 $\mathbf{n} = \frac{\nabla \phi}{|\nabla \phi|}$ ，因此可以得到 $x_t = F(x(t)) \mathbf{n}$ 。上述方程可被重写为：

$$\phi_t + \nabla \phi x_t = 0$$

$$\phi_t + \nabla \phi F \mathbf{n} = 0$$

$$\phi_t + F \nabla \phi \frac{\nabla \phi}{|\nabla \phi|} = 0$$

$$\phi_t + F |\nabla \phi| = 0$$

上述最后一个方程定义了相函数 ϕ 的运动。现若我们已经有初始条件 $\phi(x, y, t = 0)$ ，由运动方程，我们可以得到全计算域 ϕ 的分布。

根据相函数 ϕ ，若从三维视角考虑 ϕ 函数的空间分布，我们可以得到表面曲率：

$$\kappa_{ls} = \nabla \frac{\nabla \phi}{|\nabla \phi|} = \frac{\phi_{xx} \phi_y^2 - 2\phi_{xy} \phi_x \phi_y + \phi_{yy} \phi_x^2}{(\phi_x^2 + \phi_y^2)^{\frac{1}{2}}}$$

• 离散化算法

现我们基于前向差分算法，给出简单的离散化 ϕ 的方法。对 ϕ_t 在坐标点 (i, j) 进行时间离散，可以得到： $\frac{\phi(i, j, t + \Delta t) - \phi(i, j, t)}{\Delta t}$ 。利用有限差分格式，梯度可写为：

$$\nabla^{+x}(i, j) = \max [0, \Delta^{-x} \phi(i, j)]^2 + \min [0, \Delta^{+x} \phi(i, j)]^2, \text{ when } F > 0, \text{ or}$$

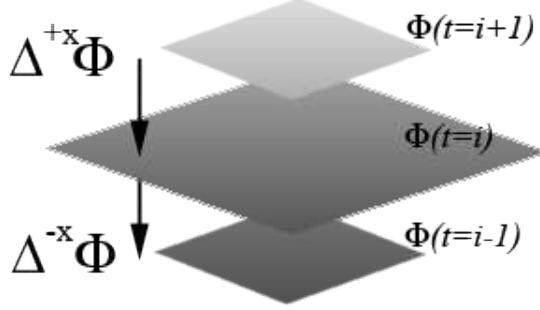


图 3-2 对空间函数 ϕ 离散化的过程示意图。

$$\nabla^{-x}(i, j) = \max [0, \Delta^{+x}\phi(i, j)]^2 + \min [0, \Delta^{-x}\phi(i, j)]^2, \text{ when } F < 0$$

其中, $\nabla^{+x}\phi$ 和 $\nabla^{-x}\phi$ 是我们给定点的左侧或右侧的差分。差分的具体形式详见图3-2。

前述的运动方程在差分格式下可写作:

$$\frac{\phi(i, j, t + \Delta t) - \phi(x, y, t)}{\Delta t} + \max[F, 0]\Delta^{+x}(i, j) + \min\Delta^{-x}(i, j) = 0$$

因此, 基于离散格式的运动方程, 相函数 ϕ 可以用一下形式进行更新迭代:

$$\phi(i, j, t + \Delta t) = \phi(x, y, t) - \Delta t [\max[F, 0]\Delta^{+x}(i, j) + \min\Delta^{-x}(i, j)]$$

通过 ϕ 计算结果, 我们因此可以计算空间曲率, 采用中心差分算法, ϕ 的导数可以写为:

$$\phi_{xx}(i, j) = (\phi(i + 1, j) - \phi(i, j)) - (\phi(i, j) - \phi(i - 1, j))$$

$$\phi_{yy}(i, j) = (\phi(i, j + 1) - \phi(i, j)) - (\phi(i, j) - \phi(i, j - 1))$$

$$\phi_{xy}(i, j) = \frac{1}{4} (\phi(i + 1, j + 1) - \phi(i - 1, j + 1)) - (\phi(i + 1, j - 1) - \phi(i - 1, j - 1))$$

$$\phi_x(i, j) = \frac{1}{2} (\phi(i + 1, j) - \phi(i - 1, j))$$

$$\phi_y(i, j) = \frac{1}{2} (\phi(i, j + 1) - \phi(i, j - 1))$$

基于前文给出的曲率计算公式，我们可以数值计算得到曲率：

$$\kappa_{ls} = \frac{\phi_{xx}\phi_y^2 - 2\phi_y\phi_x\phi_{xy} + \phi_{yy}\phi_x^2}{(\phi_x^2 + \phi_y^2)^{\frac{3}{2}}}$$

$$\kappa_{ls}|\nabla\phi(i,j)| = \frac{\phi_{xx}\phi_y^2 - 2\phi_y\phi_x\phi_{xy} + \phi_{yy}\phi_x^2}{\phi_x^2 + \phi_y^2}$$

为防止曲率过大，使得差分过程引入较大误差，Lombaert 推荐在更新 ϕ 时引入曲率：

$$\phi(x, y, t + \Delta t) = \phi(i, j, t) + \Delta t [max(F, 0)\nabla^{+x}(i, j) + min(F, 0)\nabla^{-x}(i, j)] + \Delta t[\kappa_{ls}(i, j)]$$

上述是 level set 算法的数学基础。其基本推导主要基于加拿大 École de technologie supérieure ÉTS 副教授 Hervé Lombaert 对于 level set 算法的网站^[92]。

3.1.2 单气泡模型

单气泡微管道建模方法如图3-3所示：坐标原点定位微管道左下角处；管道左侧以(2.5, 2.5)为圆心，做一直径为4μm的圆，为初始气泡。柱状管道设置长度为15μm，直径为5μm。基于前述对于血红细胞、组织细胞以及药物输运中油性气泡等的研究^[65-71, 79-81]，我们发现，细胞与气泡大多直径处在几个微米量级，因此此处我们给定气泡直径 $d = 4\mu\text{m}$ ，以符合应用背景。轴向给定周期性边界条件，且管道进出两端压强差为 $\Delta p = 10\text{Pa}$ 。初始边界条件给定为：初始速度为0，初始压强给定为组织液压强^[83]：基于实验数据与过往研究，我们将初始压强设定为6mmHg，转换为国际标准单位后，可给定： $p_0 = 799.932\text{Pa}$ 。同时，我们将液体流出段两端设定两压强限制点，压强依旧选为组织液压强 $p = 799.932\text{Pa}$ 。上下两端为管壁，在多相流系统内设定为润湿壁，液相流体与固壁的接触角设置为 $\pi/2$ 。气泡（气相材料）设为空气（Air），流体（液相材料）设置为水（H₂O）。温度为室温（293.15K）。

给定模型后，针对多相流我们采用 Level Set 算法^[84]（相关算法已在第2章给出介绍）。我们采用时间相关（Time-Dependent）的求解器，总共计算5000μs内单气泡的运动，其中每20μs输出一次结果。

因多相流的两相接触界面需要高精度计算，否则可能造成其中一相的成分损失，致使迭代后时间步计算误差增大，使结果不可信。因此，在本算例中我们采用系统

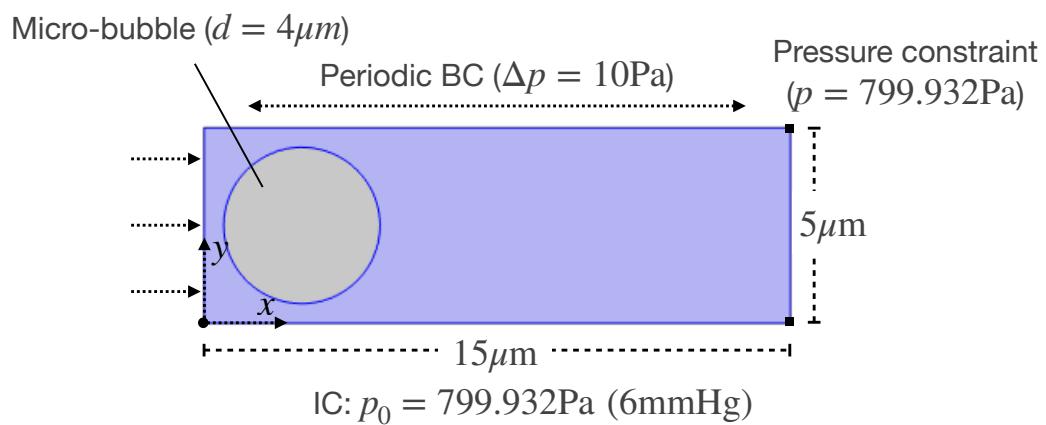


图 3-3 单气泡模型的建模。

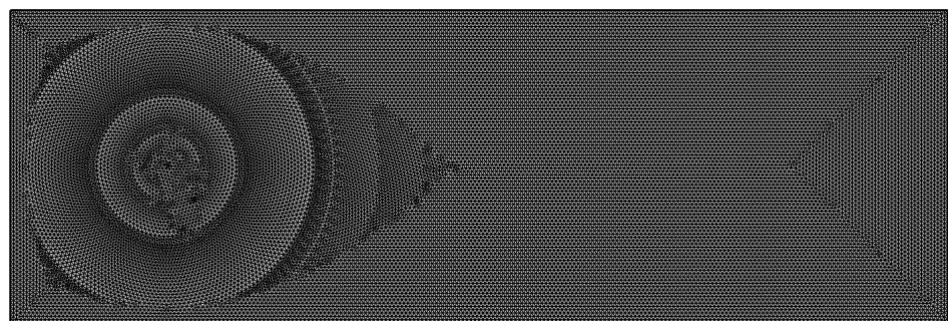


图 3-4 单气泡模型的初始网格构建。

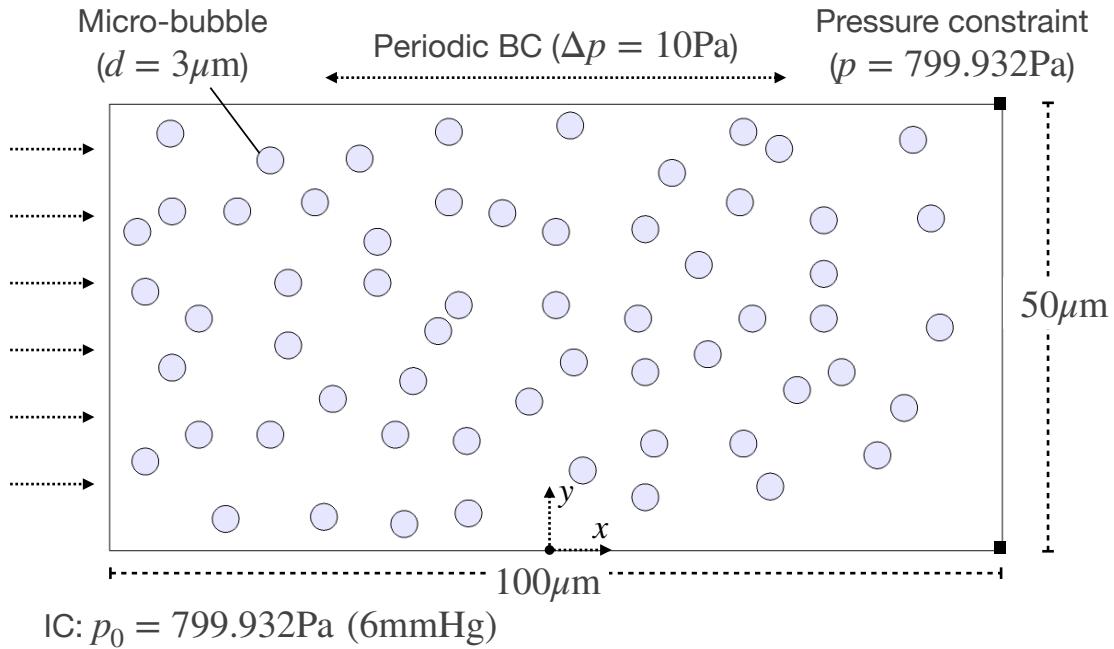


图 3-5 多气泡模型的建模。

定义的加密网格，选用”Extra Fine” 网格，网格数为 24812，如图3-4所示。

3.1.3 复杂多气泡系统

多气泡系统的二维建模方法如图3-5所示。参考前一小节的应用设定，我们在此也将气泡直径设定在 $1 \sim 5\mu\text{m}$ 内，选定 $d = 3\mu\text{m}$ 。由前述工作^[70, 71]，我们知道，生物体内气泡分布较为均匀和密集，参考 Wang and Hu^[85] 对于细胞内流体建模的工作。在生物力学环境中，当气泡分布均匀时可以采用单向流处理多相流问题。因此，在对多气泡系统进行建模时，我们假设气泡密集地，并且较为均匀地分布在流域内。设定管道长度为 $100\mu\text{m}$ ，管壁直径为 $50\mu\text{m}$ 。坐标原点定为下管壁中点处。60 个直径为 $3\mu\text{m}$ 的微气泡均匀分布在管道内部。参考上一小节，初始条件给定 $p_0 = 799.932\text{Pa}$ ，管道出口压力限制点 $p = 799.932\text{Pa}$ 。轴向给定周期性边界条件，且压差 $\Delta p = 10\text{Pa}$ 。温度为室温（ 293.15K ）。

对于该算例，我们设定多气泡系统计算 $3000\mu\text{s}$ ，其中每 $1\mu\text{s}$ 输出一次结果。数值模拟采用 Level Set 和 Time-Dependent 计算方法。由于多气泡系统中每个气泡相对较小，在整个算例中容易造成长时间计算致使组分丢失问题，因此我们在该算例中给定网格为最密集等级（Extremely Fine），网格数为 75302，网格如图3-6所示。

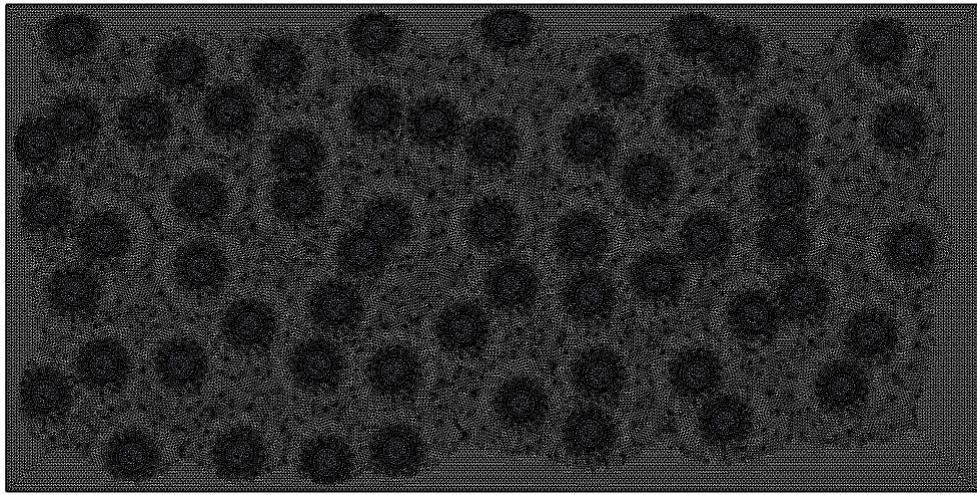


图 3-6 多气泡模型的初始网格构建。

3.2 动力学分析

3.2.1 单气泡计算结果

单气泡计算后气泡运动形态如图3-7所示。图中从 $400\mu\text{s}$ 开始显示单气泡运动，每隔 $400\mu\text{s}$ 记录气泡形态，共记录九个气泡形态直至 $3600\mu\text{s}$ 。Tomaiuolo et al.^[80] 曾研究了流速为 1.12cm/s ，红细胞在 $6.6\mu\text{m}$ 的微管道中运动的情况。在我们的仿真中，由定性分析可观测到，在 $3600\mu\text{s}$ 时的单气泡变形与 Tomaiuolo et al.^[80] 中 Fig. 2 的单红细胞变形形貌基本一致。这在一定程度上验证了我们计算的准确性。

从图3-7中，我们可以观测到单气泡运动变化形式。气泡左端背流处最先向内凹，右端受微流动推动向右凸起，呈抛物线状前进。随着时间推进，左端界面受到流体剪切的作用，内凹至形成一定空间形成两个尾端泡张力面。

在微流体环境中，整个气泡前进造成的尾端两个附着“小气泡”的产生原因主要有二：一是随着流体速度前进致使气泡产生的惯性；二为微流体环境中流体的粘性和表面张力起到明显作用。二者共同作用下：单气泡在微管道中前进时，因气泡本身速度，以及气液接触界面对气泡表面产生的“拖曳”效应，使得单气泡变形的尾迹被拖出两个附着气泡。由前文所述，单气泡该变形模式和 Tomaiuolo et al.^[80] 实验中单血红细胞变形相似。但是因为气泡外表面仅为气-液界面表表面张力支持的，所以很容易被“撕裂”，使得附着小气泡被“剥离”。而 Tomaiuolo et al. 实验中红细胞却在流动过程中没有产生这一现象，因为血红细胞因为外表面有生物膜，生物膜本

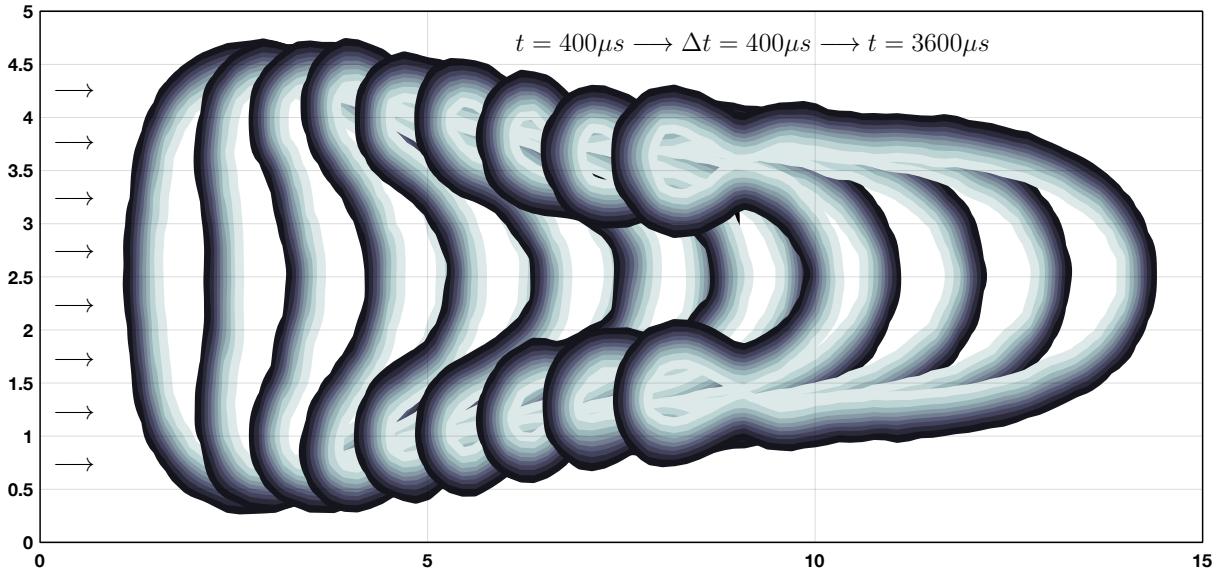


图 3-7 单气泡在管道中的运动形态。

身具有弹性、韧性等力学性质，故可以使得红细胞在微管道流动中保持完整形态。

为研究及验证气泡管道流动的规律及机理，图3-8展示单气泡管道流管道尾端（管道出口端）在前 $3600\mu s$ 的速度分布。时间选至 $3600\mu s$ 的原因是单气泡在 $3600\mu s$ 之后会流经管道尾端（如图3-7），致使速度分布与一般管道流的速度型不完全一致。

由图3-8，管道右段速度在 $40\mu s$ 后便符合抛物线的速度分布，之后直至 $3600\mu s$ 基本上都符合该分布规律，即一般泊肃叶流动的流动特征^[86, 87]。抛物线在 $40 \sim 1000\mu s$ 内基本符合同一抛物线参数，而后在 $2000 \sim 3600\mu s$ 内进入另一抛物线分布：抛物线顶端斜率导数减小速度降低，靠近管壁测速度分量减小，管道中心处速度分量增大。

为了更好地分析气泡流经时（气泡运动过程）捕捉气液界面变化过程的物理信息，我们将初始气泡位置设定为距离起始位置 $5\mu m$ 处设为“观察轴”，记录此处的流体速度分布，如图3-9所示。我们可观察到，初始 $t = 20\mu s$ 时，气泡还未穿越观测轴，直至 $t = 600\mu s$ 时，气泡右侧顶端开始穿过我们设定的观测轴，为中间外凸的抛物线；后至 $t = 1000\mu s$ 时，气泡一大部分都穿过“观察轴”，使得抛物线中间外凸部分数值增大，外凸趋势更加明显；至 $t = 1400\mu s$ 时，抛物线外凸趋势消失，此时气泡主体部分开始流经观测轴，抛物线的外凸顶点下凹；至 $t = 2000\mu s$ 时，气泡主体部分正在记录数轴上，抛物线的外凸呈“宽大状”，气泡的尾端（左侧）也即将流经观测轴；而后至 $t = 2600\mu s$ 到 $t = 5000\mu s$ 时，气泡已经流过观测数轴，管道速度分布

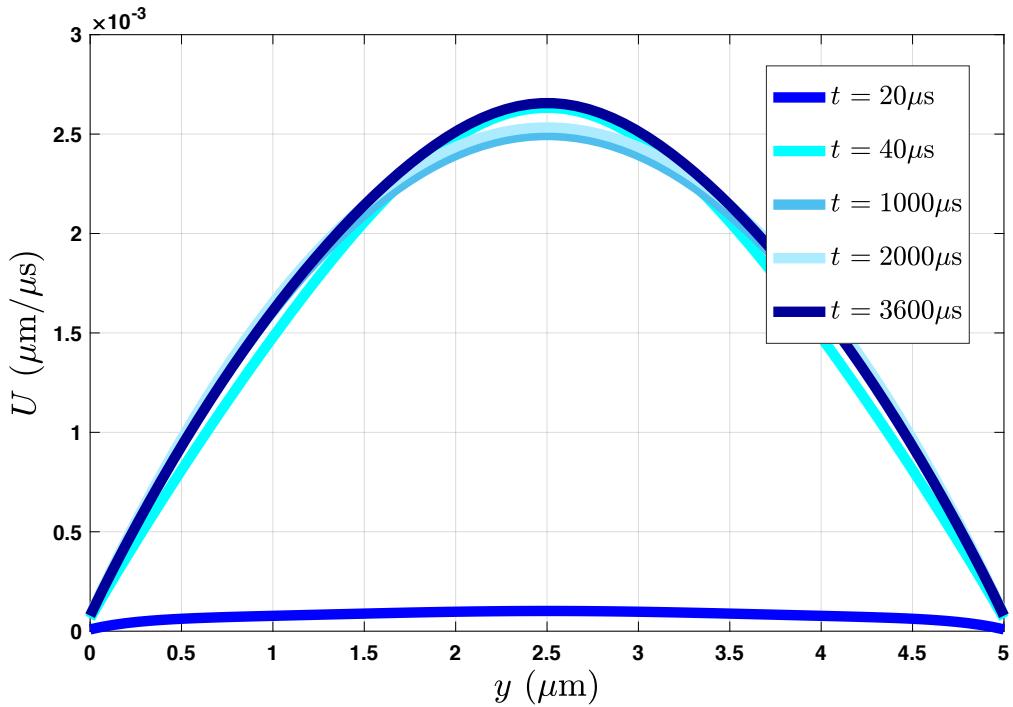


图 3-8 单气泡模型流场末端的速度分布。

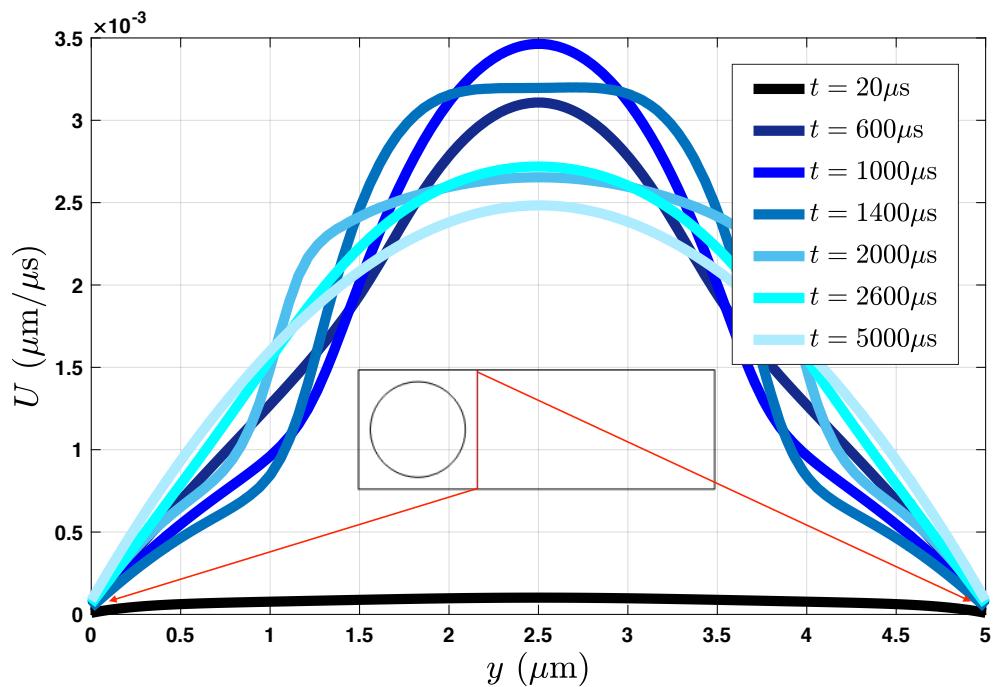


图 3-9 单气泡模型气泡右端观测轴上的速度分布。

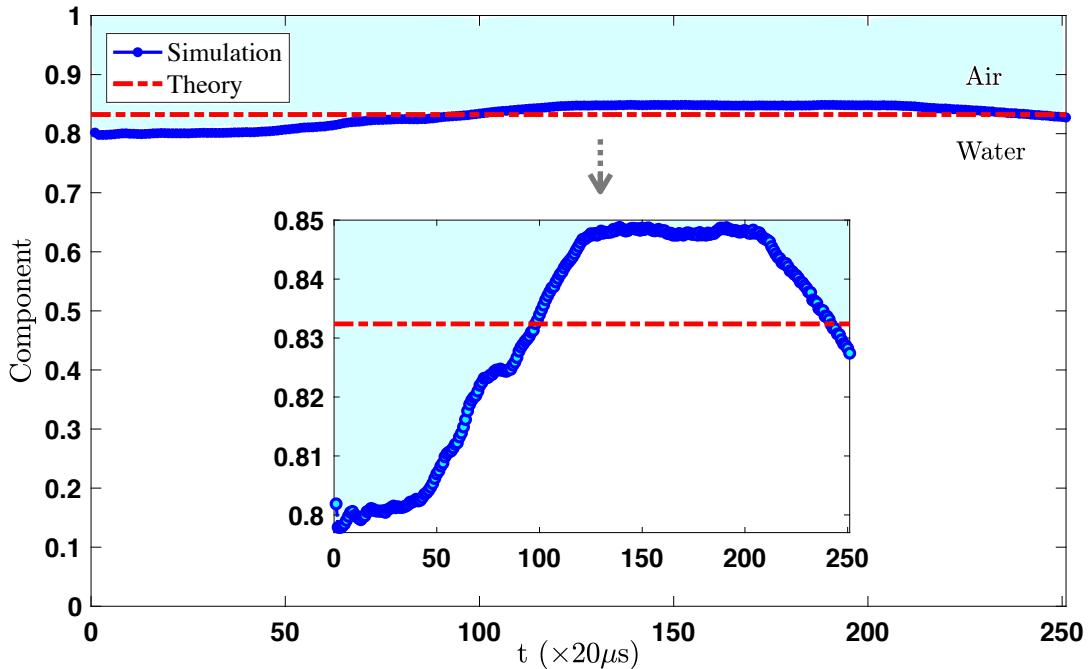


图 3-10 单气泡模型运算过程中气-液组分分布。

已经与图3-8纯液相管道流分布相似，满足抛物线基本特征。其中，每一时刻气泡形态可参考图3-7气泡整体运动形态变化。

多相流计算中对网格需求较高，可能因稀疏网格导致在计算迭代过程中相流体组分丢失，致使误差增大，使结果不可信。因此，我们选择较密网格进行数值模拟。但在结果分析时，我们仍需验证两相组分分布，以确保结果精准。图3-10展示了对单气泡算例气-液两相的组分在整个计算过程的变化情况，其中浅蓝色区域为空气（气泡），白色区域为水（液体）。红色直线为根据气液面积（二维体积）比计算理论值。举例来说，空气相组分面积占比计算公式可写为：

$$Component_{Air} = \frac{Area_{Air}}{Area_{Air} + Area_{Water}}$$

在实际计算中，由第2章节介绍，我们用相函数 $\phi \in [0, 1]$ 来描述气-液两相占比。在实际计算得出的场分布中，相函数分布在 $(0, 1)$ 之间，极少存在数值为 0 或 1 的情况。这也是符合真实物理规律的：在实际的空气-水混合体系中，水中会混合一定量的空气或气体分子微团，而空气（气泡）中也会混合一定量的水分子微团（游离液态水），既纯粹的空气或水（或纯气态 & 液态）几乎不存在，即界面一般是具有一定

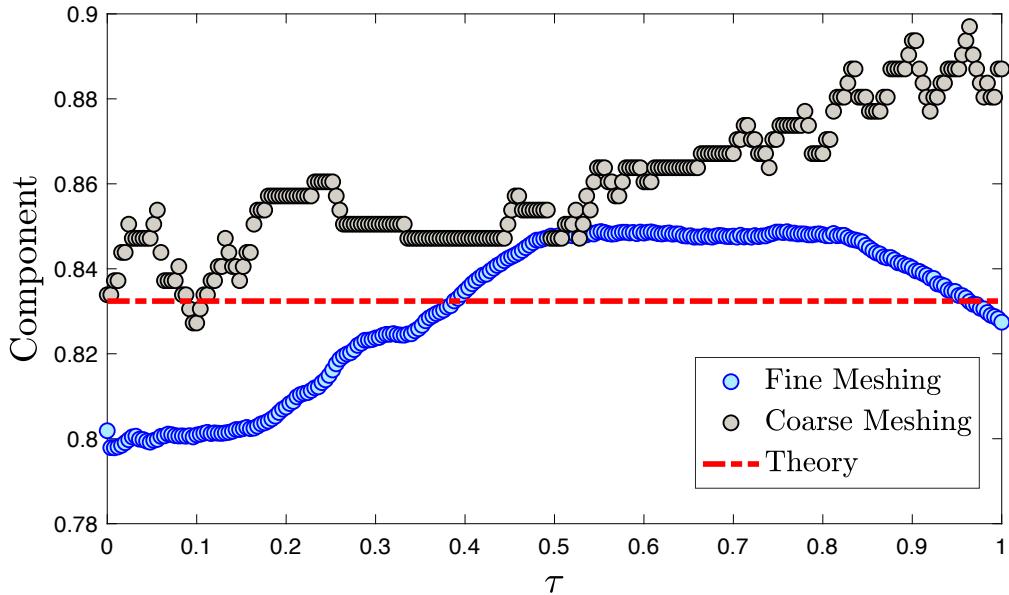


图 3-11 针对单气泡算例的稀疏和密集网格对比下组分函数随时间分布变化。

厚度的。实际操作中，对于 $\phi \in [0, 1]$ ，我们取 $\phi \in [0.5, 1]$ 属于气相，取 $\phi \in [0, 0.5]$ 属于液相，记录组分随时间变化，计算所得两相比为图中所示蓝色曲线。由图我们可以得出在整个气泡运动过程中气-液相分布基本符合预期，在理论值上下波动。理论计算结果为 $Component_{Water}/Component_{Air} = 0.8324$ ，仿真计算结果最大值小于 0.85，最小值约在 0.8，均在接受误差范围内。

针对前文中我们提到的网格稀疏程度会影响计算精度和结果的问题，我们在此处针对单气泡流动算例给出分析。对于单气泡流动，我们分别使用了我们上文介绍的 Extra Fine 和粗化网格 Extremely Coarse 进行计算。其中精密网格数为 24182，粗化网格数为 301，组分分布随时间变化曲线如图3-11所示。由图我们可以得出粗化网格的组分在初始阶段还比较符合理论值，到后期组分明显偏离理论值。这说明了在计算多相流过程中网格不够密集会导致某一相组分的损失，为计算过程的误差。

除了前文中我们提到的计算迭代致使某一相流体含量被忽略丢失导致的计算误差外，在计算气泡运动过程中气泡运动本身也会造成组分比产生误差。因为真实物理运行在三维空间，在进行二维计算时，二维物理变量的数值在二维空间内体现在第三维度上。例如，假如将气-液组分作为分析量，在第三维度上产生 $[0, 1]$ 的数值取样，接近 1 为气相，接近 0 为液相；在涉及到气泡融合 & 分离时，两气泡主体间会产生介于 $(0, 1)$ 间的气-液界面；对于本小节单气泡算例，该界面在运动过程中可

能因为数值趋向于 0 而被算作液相部分被损失掉。对于涉及到气泡融合情况（多气泡算例），当两气泡融合瞬间，在气泡间的碰撞、气液两相的混合以及小气泡的剥离损耗都可能是计算误差的来源。

3.2.2 多气泡计算结果

分析多气泡算例时，我们采用和上一小节相同的分析方法。整个多气泡系统运动趋势如图3-12所示：初始状态下 60 个微气泡均匀分布在流场各点，至 $t = 900\mu s$ 时，可观测到极个别气泡已经融合，少量呈现融合趋势；至 $t = 1500 \sim 1800\mu s$ 时，可以看到已经有少部分气泡已经融合，且分别对应 $t = 1500 \& 1800\mu s$ 两时刻，我们也可观测到气泡的融合过程，并且融合气泡还会产生更小的微气泡剥离现象。至 $t = 2100 \& 3000\mu s$ 时，大部分气泡已经发生了气泡间的碰撞和融合；在 $3000\mu s$ 我们观察到多气泡融合后的“大气泡”更易与周围微气泡产生新的融合。

图3-13是我们在微管道尾端（右侧）设置观察数轴，并记录微气泡群流经该轴时流体速度分布。由图我们可以得出，在微气泡群管道流情况下，尽管有气-液两相混合，但是与单气泡算例不同，管道速度分布基本符合泊肃叶流的抛物线速度分布；这是因为当气体含量占比较小并均匀混合在液体中时，该种流动被定义为传统的气泡流（bubble flow）^[72]。这也与 Wang and Hu 工作中对于生物细胞环境下当微气泡较小时，可以将两相流用单向流进行建模的思路吻合^[85]。在计算至 $1000\mu s$ 后，我们观测到流体速度分布几乎不发生改变，既说明气泡群管道流在 $1000\mu s$ 后流动稳定，满足相似速度分布规律，直至在 $t = 3000\mu s$ 算例计算完成。

同样，采用上一小节对单气泡分析方法，我们也验证了气泡群动力学在计算过程中组分变化是否符合物理规律。图3-14展示了在整个气泡群运动过程中气-液组分比和理论标准值的对比。与上一小节采用同样方法，对计算所得相函数 ϕ ，取 $\phi \in [0.5, 1]$ 属于气相，取 $\phi \in [0, 0.5]$ 属于液相，记录所得组分随时间变化规律如图3-14中蓝色曲线所示，其中浅蓝色区域为空气，白色区域为水。理论值对应初始情况下液相在整体占比： $Component_{Water} = 0.915$ ，由图3-14可得 $Component_{Water} \in (\sim 0.9, \sim 0.92)$ ，符合误差范围。

本章节中，我们分别介绍了单气泡和微气泡群的算例应用背景、建模及数值实现过程；而后对气泡演化形态、流场速度分布和相体积变化等计算结果进行了多方面的分析。下一章节，我们将介绍我们对多相流问题提出的深度学习算法以及预测

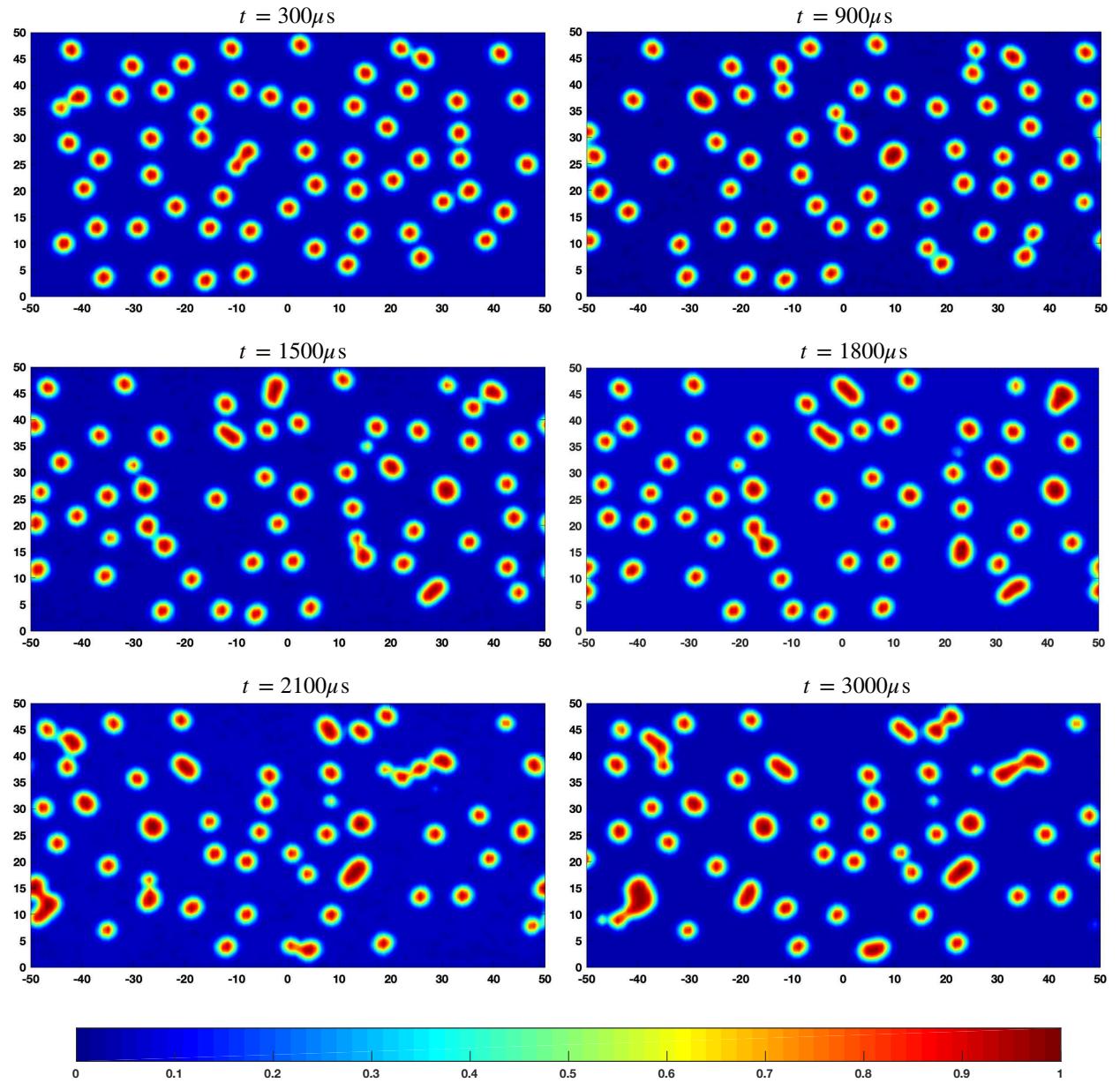


图 3-12 微管道中多气泡系统运动过程中气液组分的变化趋势。

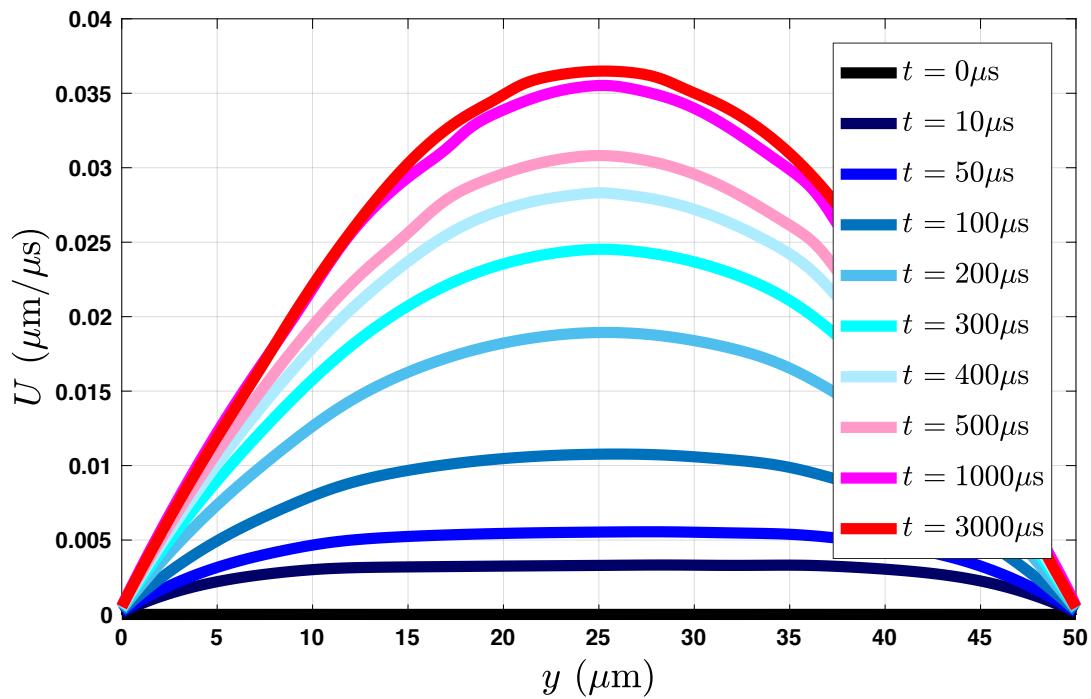


图 3-13 多气泡模型流场末端的速度分布。

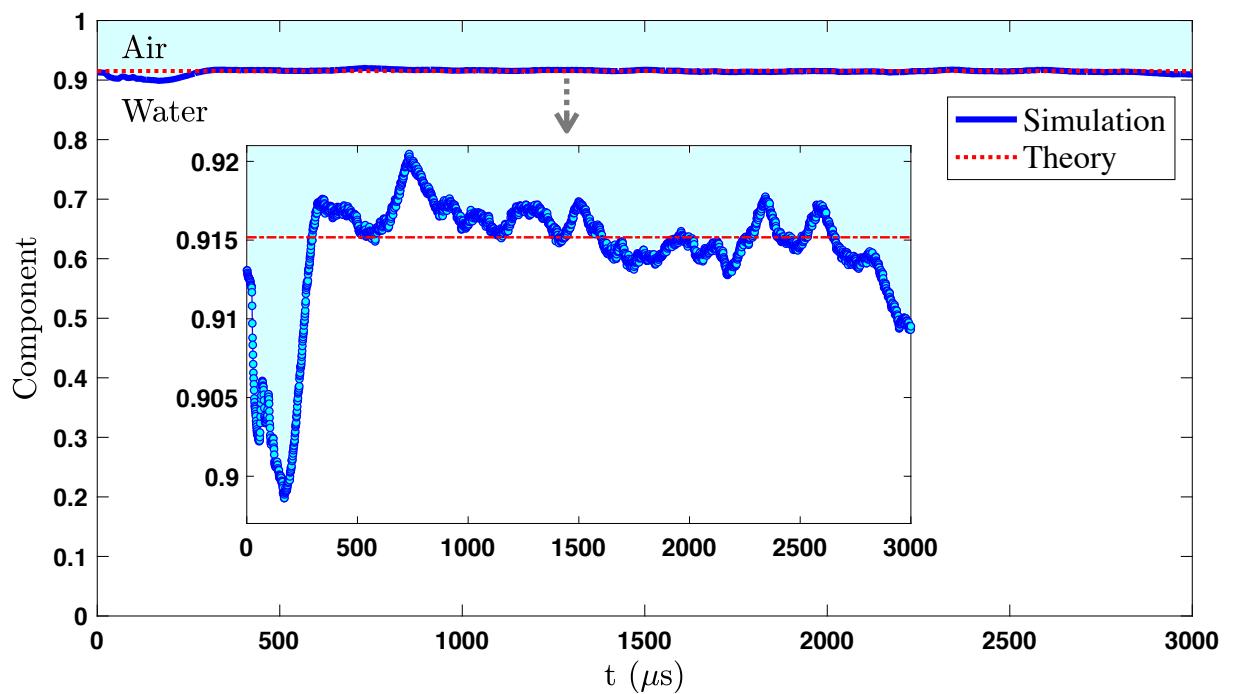


图 3-14 多气泡模型运算过程中气-液组分随时间的变化。

结果。

3.3 本章小结

在本章节中，作者首先简要介绍了在我们数值计算气泡流动中所使用到的 level set 算法的基本原理。之后给出了本文中主要用到的两个算例，单气泡和多气泡流动，的基本建模方法以及相关的参数给定的应用背景。在建模中我们主要关注这两个模型在生物化学的应用，同时侧重于力学原理分析。我们分别分析了单气泡和多气泡流动的管道速度场分布以及气泡形貌变化及其背后的力学机理。同时，为了验证我们做的数值模拟符合力学规律，我们计算了两相系统中各项组分之比符合理论预测。

第 4 章 数据驱动的两相流预测与学习方法

由第 1 章和第 2 章提到，Raissi et al.^[47] 于 2019 年提出了基于引入物理方程损失函数的深度神经网络 PINN。在本章节中，我们将会效仿他的思路，提出针对微管道中气泡流的神经网络结构。我们首先介绍利用普通深度神经网络预测气泡流的算法以及结果，随后介绍本文提出的网络框架 BubbleNet 结构与算法和结果。

由第 3 章节中我们介绍的单气泡与多气泡系统两个算例，我们分别针对这两个算例使用了传统深度神经网络（DNN）与我们提出的基于气泡动力学神经网络（BubbleNet）进行计算预测。对于单气泡算例，我们的目标是预测 $t = 2000\mu\text{s}$ 时气泡的运动状态；对于多气泡算例，我们的目标是预测 $t = 1500\mu\text{s}$ 时气泡的运动状态。运动状态包括当时刻物理场的速度、压力与相函数分布 (u, v, p, ϕ) 。

为了从算例中获取数据用作神经网络训练数据 $(x_{train}, y_{train}, t_{train})$ ，同时为了减小计算资源损耗，我们对于两不同算例采用相同的策略：将场数据分布 (x, y, t) 数据进行稀疏化。对于单气泡算例，我们在 (x, y) 方向对场数据进行每隔 10 个数据点取样一次；对于多气泡算例，我们对场数据进行每隔 20 个数据点取样一次。对于时间步，单气泡算例每隔 2 个数据点取样一次；单气泡算例每隔 30 个数据点取样一次。以上稀疏化方法可表述为

$$(x_{train}, y_{train}, t_{train}) = (x(1 : \Delta^{space} : end), y(1 : \Delta^{space} : end), t(1 : \Delta^{time} : end))$$

式中 Δ 为抽样间隔（interpolate），其中单气泡算例场数据 (x, y) 取样如图4-1所示，多气泡算例我们也做了同样的处理。

4. 1 优化函数选取

在我们使用的深度神经网络中，我们使用了优化函数 L-BFGS-B 和 Adam：先设定固定次数的 Adam 优化器的迭代次数，完成给定的迭代次数后，利用 L-BFGS-B 优化函数完成优化^[91]。

4. 1. 1 Adam 优化器

ADAM（或 Adam）优化器是自适应动量分析的（ADAptive Moment estimation）的简称。其基本思路为每一步保存梯度下降前一步梯度的第一和第二动量来更新参

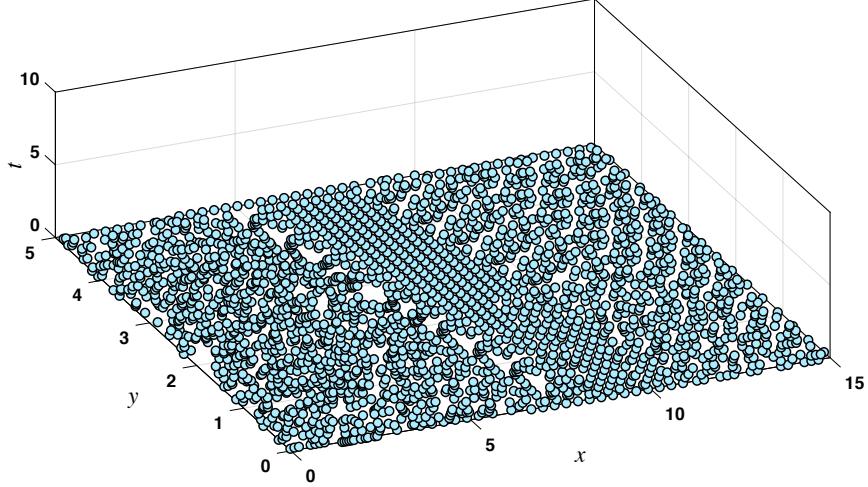


图 4-1 对单气泡模型的场数据取样。

数。

首先，我们先定义参数：由第二章， w 为神经网络线性模型中的权重，定义拟合优度函数 $\mathcal{Q}(w) = \frac{1}{n} \sum_i^n \text{Loss}(y_i, (w, x_i))$ ， Loss 为损失函数；定义动量的指数衰减率 $\mathcal{B}_1, \mathcal{B}_2 \in [0, 1)$ ，一般默认数值 $\mathcal{B}_1 = 0.9, \mathcal{B}_2 = 0.999$ ；数值常量 \mathcal{E} ，一般默认数值为 $\mathcal{E} = 10^{-8}$ 。

令 \mathcal{W}_t 是迭代中上一步梯度的指数衰减平均值：

$$\mathcal{W}_t = \mathcal{B}_1 \mathcal{W}_{t-1} + (1 - \mathcal{B}_1) \nabla \mathcal{Q}(w^{(t)})$$

同样的，令 ν_t 是过去平方梯度的指数衰减平均值

$$\nu_t = \mathcal{B}_1 \nu_{t-1} + (1 - \mathcal{B}_1) \nabla (\mathcal{Q}(w^{(t)}))^2$$

在迭代之前，我们给定初始值 $\mathcal{W}_0 = \nu_0 = 0$ 。基于该初始化方法，我们预估 \mathcal{W}_t 和 ν_t 在早期步骤中偏向于零梯度下降。

因此我们可以写出 Adam 优化器的最终方程

$$\widetilde{\mathcal{W}}_t = \frac{\mathcal{W}_t}{1 - \mathcal{B}_1^t}, \quad \widetilde{\nu}_t = \frac{\nu_t}{1 - \mathcal{B}_2^t}$$

$$w^{(k+1)} = w^{(k)} - \frac{\eta}{\sqrt{\tilde{\nu}_t} + \mathcal{E}} \widetilde{\mathcal{W}}_t$$

基于已经给出的方程，我们可以给出下述 Adam 算法的一般流程。

初始化过程： 初始化第一动量向量 $\mathcal{W}_0 = 0$ ， 初始化第二动量向量 $\nu_0 = 0$ ， 初始化权重矩阵 w_0 。

给定参数： 迭代步长 η （一般默认值 $\eta = 0.001$ ），给定动量的指数衰减率 β_1, β_2 （上述介绍），给定数值常数 \mathcal{E} 。

For $k = 1, 2, \dots$, 在收敛前执行

- 计算第一动量向量和第二动量向量，

$$\mathcal{W}^{(k)} = \mathcal{B}_1 \mathcal{W}^{(k-1)} + (1 - \mathcal{B}_1) \nabla \mathcal{Q}(w^{(k)})$$

$$\nu^{(k)} = \mathcal{B}_1 \nu^{(k-1)} + (1 - \beta_1) \nabla (\mathcal{Q}(w^{(k)}))^2$$

- 计算相应的更正量，

$$\widetilde{\mathcal{W}}_t^{(k)} = \frac{\mathcal{W}^{(k)}}{1 - \mathcal{B}_1^{(k)}}, \quad \widetilde{\nu}^{(k)} = \frac{\nu^{(k)}}{1 - \mathcal{B}_2^{(k)}}$$

- 更新相应的权重，

$$w^{(k+1)} = w^{(k)} - \frac{\eta}{\sqrt{\widetilde{\nu}^{(k)}} + \mathcal{E}} \widetilde{\mathcal{W}}_t^{(k)}$$

Return $w^{(k)}$

上述推导过程主要来源于 École Polytechnique 应用数学中心助理教授 Erwan Scornet 的深度学习课件 [91]。

4.1.2 L-BFGS-B 优化器

L-BFGS-B 优化器是有限内存的 BFGS 算法（Limited-memory BFGS）的延伸，其扩展了 L-BFGS 并使其可以处理简单框约束的变量。通俗讲就是更节约内存的

BFGS。BFGS 是 Broyden–Fletcher–Goldfarb–Shanno 算法的简称。而 Broyden–Fletcher–Goldfarb–Shanno 则是一种半牛顿优化算法。要给出 L-BFGS 的算法基本推导，我们首先给出拟牛顿方法（Quasi-Newton method）的计算流程。

首先，对于无外加约束的优化问题，我们都可以简写为

$$\min_{x \in \mathbb{R}^n} \mathcal{P}(x)$$

其中，我们的目的就是寻找能使对象函数 \mathcal{P} 达到最小的 $x^* \in \mathbb{R}^n$ 。如果仅用解析方法找到 x^* 是几乎不可能的，所以我们采用数值迭代方法：生成关于 x 的一串离散点 $\{x^{(j)}\}_{j \in \mathbb{N}}$ 并使他们能够在 $j \rightarrow \infty$ 使 \mathcal{P} 函数达到局部最小值：

$$\mathcal{P}(\lambda x + (1 - \lambda)y) \leq \lambda \mathcal{P}(x) + (1 - \lambda)\mathcal{P}(y)$$

对于所有的 $x, y \in \mathbb{R}$ ，以及所有 $\lambda \in [0, 1]$ 若上式均成立，则 \mathcal{P} 函数的局部最小即是全局最小。

拟牛顿法

我们首先假设函数 \mathcal{P} 连续且可微。在第 j 次迭代上采用线性搜索方法我们可以计算搜索方向 $\xi^{(j)}$ 以及步长 α_j ，则下一次迭代的 $x^{(j+1)}$ 可以被计算出来：

$$x^{(j+1)} = x^{(j)} + \alpha_j \xi^{(j)}$$

对于不同的线性搜索方法，采用不同的 $\xi^{(j)}$ 和 α_j 。

对于拟牛顿法，搜索方向可以写作：

$$\xi^{(j)} = -C_j \nabla \mathcal{P}(x^{(j)})$$

其中 C_j 被称为校正矩阵，又称海森矩阵（Hessian matrix）。每迭代一次， C_j 就更新迭代一次。

只要 C_j 满足正定流形，我们有 $(\xi^{(j)})^T \nabla \mathcal{P}(x^{(j)}) < 0$ ，此时 $\xi^{(j)}$ 为下降方向；如果我们采用 $C_j = I$ ，该方法则简化为最速下降法。

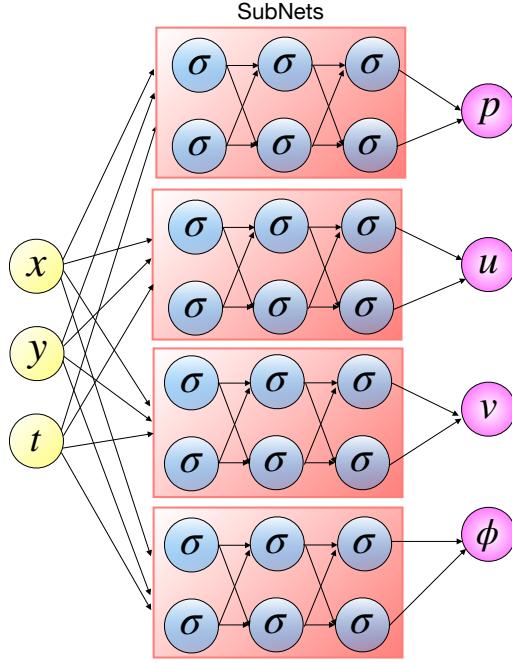


图 4-2 人工深度神经网络的示意图。其中我们用 $[3 \times 2]$ 网络结构做示意图；实际训练网络结构采用 $[9 \times 30]$ 网络结构。

基于拟牛顿法，BFGS 算法中，校正矩阵更新方法为

$$C_{j+1}^{\text{BFGS}} = (I - \rho_j s_{(j)}(y^{(j)})^T) C_j (I - \rho_j y_{(j)}(s^{(j)})^T) + \rho_j s_{(j)}(s^{(j)})^T$$

其中 $s^{(j)} = C_{j+1} y^{(j)}$, $\rho_j = (s_{(j)}(s^{(j)})^T)^{-1}$ 。

而 L-BFGS 算法基于 BFGS 算法，其省略了计算海森矩阵的步骤，只计算全迭代过程中每一步上 m 步的 s_j 和 y_j 的值。既只储存 $s_j, s_{j-1}, \dots, s_{j-m-1}$ 和 $y_j, y_{j-1}, \dots, y_{j-m-1}$ 。并用这些数值去计算近似的 ξ_j 。而我们使用的 L-BFGS-B 算法扩展了 L-BFGS 以处理变量上的简单框约束 [96]。

上述推导主要 Anders Skajaa 博士在纽约大学 Courant 数学研究所的硕士论文 [93]。

4. 2 DNN: 深度人工神经网络

由第 2 章节介绍，人工神经网络可以看作一种模仿人脑中神经元结构的非线性数据拟合器。深度人工神经网络则是由多个隐层以及多个神经元构成的结构。本文中，为了学习并预测气泡的运动，我们采用了有 9 个隐层，每隐层具有 30 个神经

元的深度神经网络结构，如图4-2所示。为了能够预测气泡流场所具有的基本物理量，我们令需要被预测的4个物理量 $[u, v, p, \phi]$ 为神经网络的输出量作为监督，离散化后全时间步长的场数据 $[x, y, t]$ 为训练数据。此处，我们采用四个子神经网络分别预测对应物理场是基于Lu et al. 在DeepONet^[51]中利用Branch Net和Trunk Net两个子网络分别逼近空间域与时间域上的非线性算子思路启发。我们认为分别对物理参数进行逼近可以达到更高的准确度和效果。

Algorithm 1 DNN for predicting bubble dynamics

```

1: function DEEPNEURALNET(self, x, y, t, u, v, p, ϕ, layers)
2:    $(\hat{x}, \hat{y}, \hat{t}, \hat{u}, \hat{v}, \hat{p}, \hat{\phi}) = \text{UPDATE}(x, y, t, u, v, p, \phi)$ 
3:    $(weights, biases, layers) = self.\text{INITIALIZENN}(weights, biases, layers)$ 
4:   self.Loss = MSE $[(u - u_{pred}) + (v - v_{pred}) + (p - p_{pred}) + (\phi - \phi_{pred})]$ 
5:    $u_{pred} = self.\text{Net}_u(x, y, t)$ 
6:    $v_{pred} = self.\text{Net}_v(x, y, t)$ 
7:    $p_{pred} = self.\text{Net}_p(x, y, t)$ 
8:    $\phi_{pred} = self.\text{Net}_{\phi}(x, y, t)$ 
9:   Optimization method 'L-BFGS-B' & Optimizer: Adam
10:  def INITIALIZENN(self, layers)
11:    Initialize all the weights & biases for  $\text{Net}_u$ ,  $\text{Net}_v$ ,  $\text{Net}_p$ ,  $\text{Net}_{\phi}$ .
12:  def NEURALNET(self, weights, biases)
13:    Build NN for u, v, p, ϕ with four sets of weights & biases.
14:  def { $\text{Net}_u, \text{Net}_v, \text{Net}_p, \text{Net}_{\phi}$ } (self, x, y, t)
15:     $\{u, v, p, \phi\} = self.\text{NEURALNET}(x, y, t, weights, biases)$ 
16:  def TRAIN(self, iterations)
17:    Obtain training time & Losses; train the NN with Adam optimizer.
18:  def PREDICT  $\{u, v, p, \phi\}$  (self, iterations)
19:     $\{u_{pred}, v_{pred}, p_{pred}, \phi_{pred}\} = self.\text{sess.run}(x, y, t)$ 
20: end function
21: Input =  $\{x, y, t\}$ , Output =  $\{u, v, p, \phi\}$ 
22: Hidden layers = [30 neurons  $\times$  9 layers]
23: Load fields data of micro-bubble system dynamics simulation.
24: Set training sets =  $\{x_{train}, y_{train}, t_{train}, u_{train}, v_{train}, p_{train}, \phi_{train}, layers\}$ 
   = MaxMinScaler(Simulation Data)
25: model = DEEPNEURALNET(training sets)
26: model.TRAIN(#Iterations)
27: Set target prediction time as  $t_{pred}$ 
28: Obtain  $\{u_{pred}, v_{pred}, p_{pred}, \phi_{pred}\} = \text{model.PREDICT}(x, y, t)$  at  $t_{pred}$ .
29: Save all the data & post-processing.
  
```

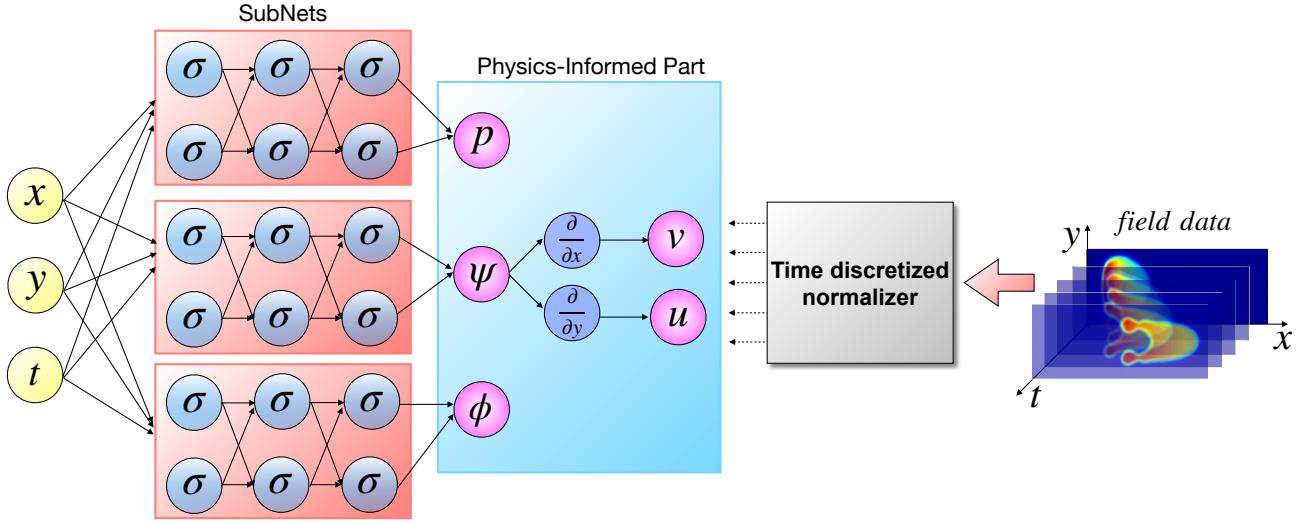


图 4-3 我们提出的物理神经网络 BubbleNet 示意图；结构同图4-2，训练采用 $[9 \times 30]$ 结构。

该神经网络程序流程如 **Algorithm 1** 所示。基本思路可以简单介绍为：定义深度神经网络函数；首先更新网络内部运算参数；设定损失函数为预测值和监督数据的差值；给定网络基本参数和优化器；开始网络计算：初始化网络，训练数据，预测数值。给定网络数据与迭代次数等基本参数。对于本文的两算例，我们设定单气泡算例迭代次数为 10000，多气泡算例迭代次数为 200000。对于给定优化器 L-BFGS-B，给定最大迭代次数为 500000。

4.3 BubbleNet: 气泡物理神经网络

根据第 1 章介绍，深度神经网络已经在工业，学术界取得大量成果并证明其应用的广泛性和可靠性。我们基于普通深度神经网络结构，提出了一种包含物理信息，并可以更精准预测多相流的神经网络 BubbleNet，其基本结构如图4-3所示。该种结构与前文中普通深度神经网络主要有两点不同：其一是，网络的输出不再是 4 个物理量，而是三个场函数 (p, ψ, ϕ) ，其中 (p, ϕ) 同上，而我们通过对流函数 ψ 进行求导，求得两个速度分量： $u = \partial_y \psi, v = -\partial_x \psi$ 。这样，我们通过预测流函数 ψ ，在获得速度场的过程中，流体的连续性方程 $u_x + v_y = 0$ 就自动满足了。

BubbleNet 的另一创新点在于我们引入了时间离散归一器（Time Discretized Normalizer, TDN）。和传统神经网络归一化的思路不同，因为针对微管道气泡流速度、压力会随时间发生很大变化，若直接对数据整体进行传统的归一化（MaxMinScaler），则会造成一部分时间步上物理场特征会被那些相应物理量数值较

大的时刻所剥夺，致使在该时刻的数据被归一化后，原本流场的特征消失。举例来讲，若 $t = t_1$ 时刻流场中压力分布的数量级为 10^5 (SI)，而 $t = t_2$ 时刻流场中压力分布数量级 10^2 ；则对整体数据进行归一化后 t_1 时刻流场压力分布可能几乎不受影响，但是 t_2 时刻数值却几乎接近为同一数值。这在训练时会“欺骗”神经网络，使得优化器认为已经完成拟合，导致最终预测偏差较大。

Algorithm 2 BubbleNet: physics-informed neural network for bubble dynamics

```

1: function BUBBLENET(self, x, y, t, u, v, p,  $\phi$ , layers)
2:    $(\hat{x}, \hat{y}, \hat{t}, \hat{u}, \hat{v}, \hat{p}, \hat{\phi}) = \text{UPDATE}(x, y, t, u, v, p, \phi)$ 
3:    $(weights, biases, layers) = self.\text{INITIALIZENN}(weights, biases, layers)$ 
4:   self.Loss = MSE[ $(u - u_{pred}) + (v - v_{pred}) + (p - p_{pred}) + (\phi - \phi_{pred})$ ]
5:    $\{u_{pred}, v_{pred}, p_{pred}, \phi_{pred}\} = self.\{\text{Net}_\psi, \text{Net}_p, \text{Net}_\phi\}(x, y, t)$ 
6:   Optimization method 'L-BFGS-B' & Optimizer: Adam
7:   def INITIALIZENN(self, layers)
8:     Initialize all the weights & biases for  $\text{Net}_\psi$ ,  $\text{Net}_p$ ,  $\text{Net}_\phi$ .
9:   def NEURALNET(self, weights, biases)
10:    Build NN for  $\psi$ , p,  $\phi$  with four sets of weights & biases.
11:   def  $\{\text{Net}_\psi, \text{Net}_p, \text{Net}_\phi\}(self, x, y, t)$ 
12:      $\{\psi, p, \phi\} = self.\text{NEURALNET}(x, y, t, weights, biases)$ 
13:      $u = \partial_y \psi \quad \& \quad v = -\partial_x \psi$ 
14:   def TRAIN(self, iterations)
15:     Obtain training time & Losses; train the NN with Adam optimizer.
16:   def PREDICT  $\{u, v, p, \phi\}(self, iterations)$ 
17:      $\{u_{pred}, v_{pred}, p_{pred}, \phi_{pred}\} = self.\text{sess.run}(x, y, t)$ 
18:   end function
19: Set training sets =  $\{x_{train}, y_{train}, t_{train}, u_{train}, v_{train}, p_{train}, \phi_{train}, layers\}$ 
= TimeDiscretizedNormalization(Simulation Data, timestep)
20: model = BUBBLENET(training sets)
21: model.TRAIN(#Iterations)
22: Rest procedures same as Algorithm 1

```

BubbleNet 的相应程序算法如 **Algorithm 2** 所示。基本思路与 **Algorithm 1** 类似，我们先给定网络基本函数，更新网络参量 & 设定网络基本参数和优化器。不同点在于我们预测数值时先通过 Net_p , Net_ψ , Net_ϕ 预测得 p , ψ , ϕ ，再通过求导数得到 u , v 。值得注意的是，网络的损失函数依然为 u , v , p , ϕ 训练数据与预测数据的差值，这也说明我们依然用网络逼近四个基本物理量，只是在网络结构“蕴含”了 ψ ，使得连续条件“藏”于其中。另一点不同的是我们在给定训练数据时使用了时间离散归一化器 TDN。

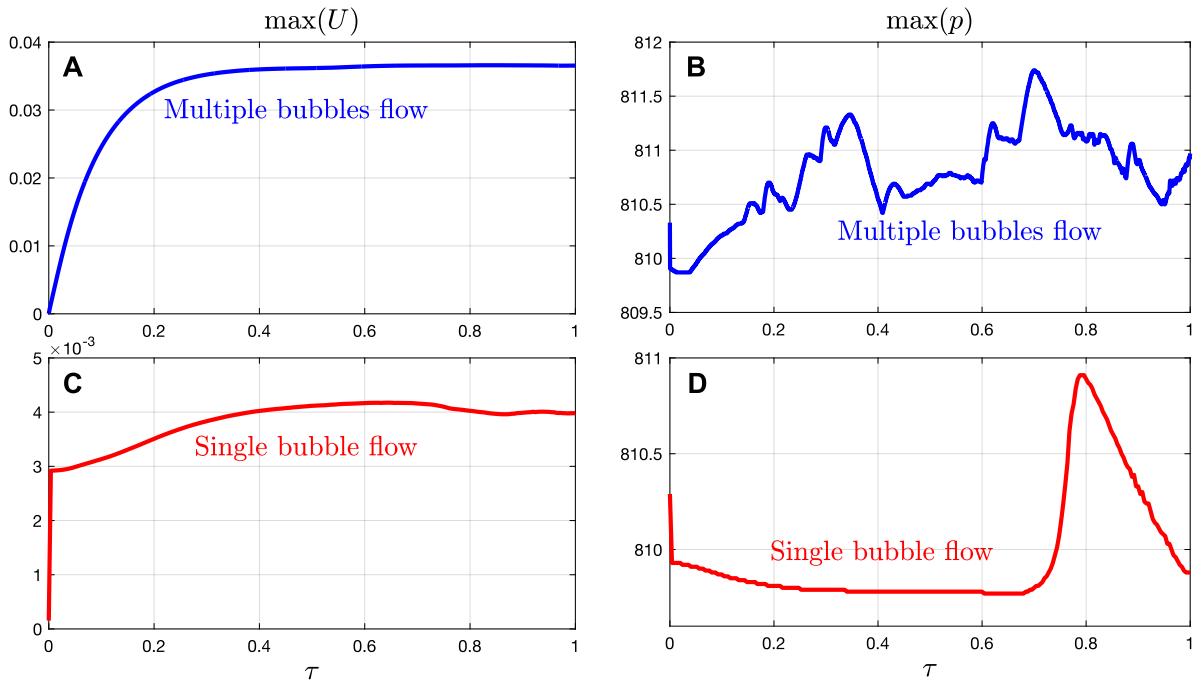


图 4-4 单气泡和多气泡算例物理场最大值随时间变化曲线。A. 多气泡流速度大小 $U = \sqrt{u^2 + v^2}$ 。B. 多气泡流压力大小 p 。C. 单气泡流速度大小 $U = \sqrt{u^2 + v^2}$ 。D. 单气泡流压力大小 p 。

4.3.1 TDN 的使用与验证

为了更好的说明为什么要使用 TDN 以及为什么 TDN 有效，我们要先说明气泡流仿真物理场每一个时刻物理量的尺度变化很大。因此，基于 TDN 针对每个时间步分别进行归一化的原理，我们给出整个仿真过程每一时间步物理量（i.e., 速度大小 $U = \sqrt{u^2 + v^2}$ 和压力 p ）的最大值，如图4-4所示。我们可以看出，物理场最大值随时间波动明显，这也间接说明使用 TDN 可以有效规避因该问题导致的数据拟合不准确问题。

4.3.2 数据粗化的定性分析

前文中我们提到，为了节省计算资源，我们在对神经网络进行训练的时候将数据进行粗化，对于多气泡算例我们采用每 20 个点取一值，对单气泡算例我们采用每 10 点取一值得到训练数据。为了验证我们的思路是可行的，且并没有因为此操作给计算带来明显误差，我们再次定型对该方法进行验证讨论。

如图4-5和4-6所示，我们分别对于单气泡和多气泡流动，对应原始数值计算数据 (CFD)，粗化后的训练数据 (Training)，深度神经网络预测结果 (DNN) 以及我们提出的 BubbleNet 预测结果，分别展示了物理场 $[u, v, p, \phi]$ 的数据分布。由两图可以

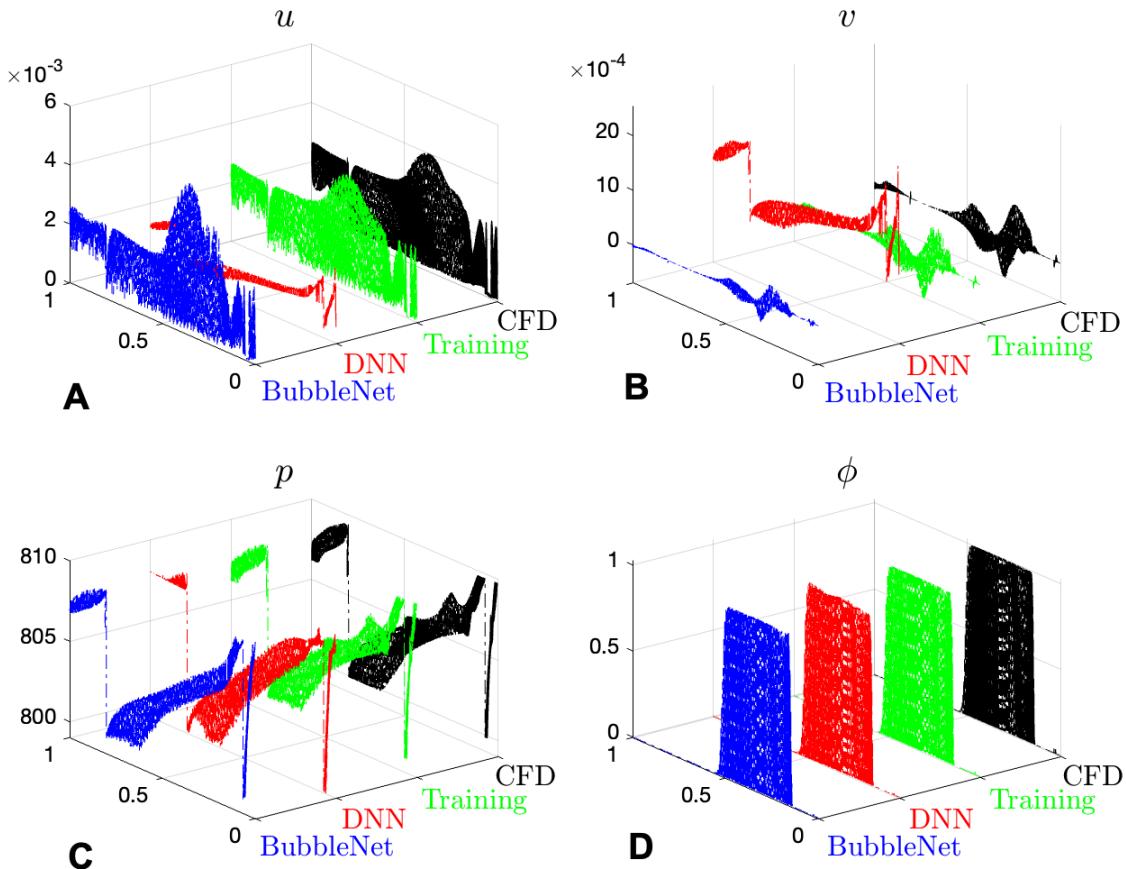


图 4-5 针对单气泡流动的原始 CFD 计算数据、粗化后的训练数据、利用普通深度神经网络预测数据以及 BubbleNet 预测数据对比结果。

看出 BubbleNet 比 DNN 在数据拟合上有更准确的结果；仅除去图4-6A 中多气泡流速度场略有小偏差，整体而言 BubbleNet 都是更加强大的数据逼近器。尤其是针对图4-5中两速度场 BubbleNet 的预测效果无论是与原场还是粗化后的场对比均明显强于传统 DNN。从两图我们也可以得出数据粗化后并没有导致明显的数据特征损失，整体数据的趋势（保有的物理信息）都依然保存。但是粗化后的数据却可以明显减少计算资源的占用（两算例粗化后的数据分别为原始数据的 $1/10$ 和 $1/20$ ）这也是用粗化数据训练的主要原因之一。

现同时会产生另一个问题，除了节约计算资源，数据粗化有没有其他用途？抑或是数据粗化有无其他目的。这便要引入计算流体力学的基本概念，一个表示物理场的数据集的大小是由什么决定的。在计算机图像领域，这是由一幅图片的像素点决定的。而在计算流体力学中，这是由一个物理场的网格数决定的；既网格密度部分决定了最后耗费的计算资源。因此，为了说明我们进行数据粗化的目的，我们在

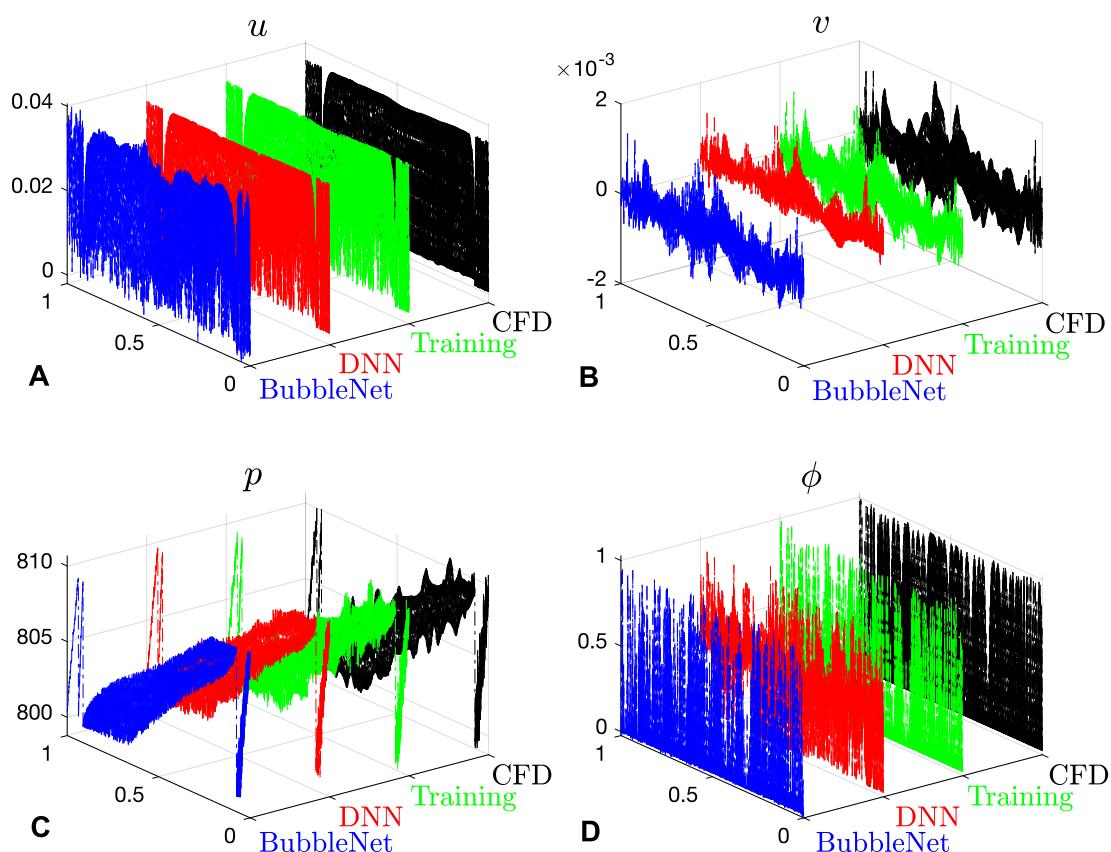


图 4-6 针对多气泡流动的原始 CFD 计算数据、粗化后的训练数据、利用普通深度神经网络预测数据以及 BubbleNet 预测数据对比结果。

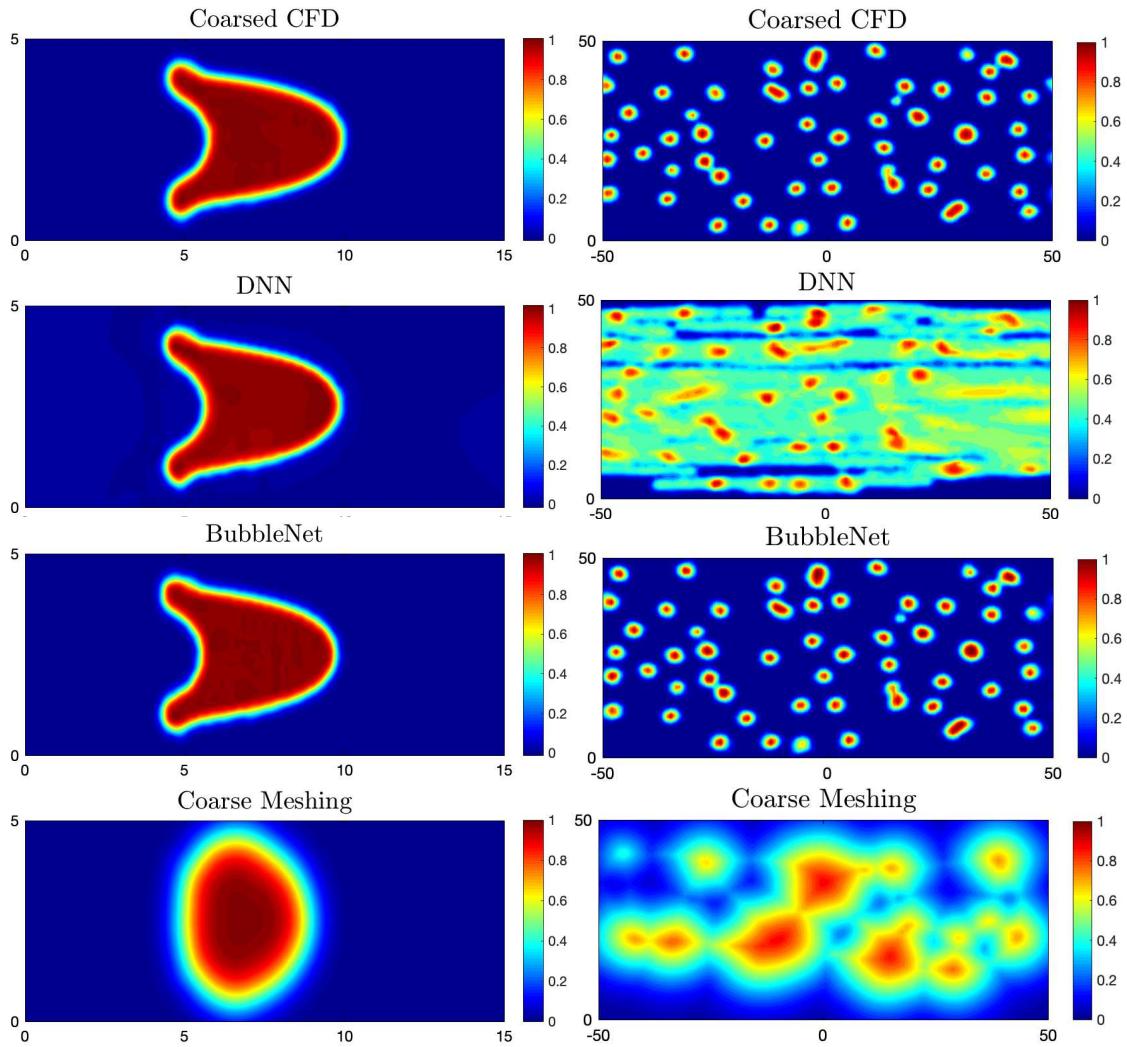


图 4-7 针对两个气泡流算例分别用粗化后的精细网格 CFD 计算 (Coarsed CFD) 与深度神经网络预测结果 (DNN), BubbleNet, 粗化网格 (Coarse Meshing) 的对比效果图。其中左侧为单气泡算例, 右侧为多气泡算例。

此给出采用与我们粗化后的数据相似大小的数据量对应的网格数进行 CFD 计算两算例所计算出来的结果对比, 以此来说明我们数据粗化效果以及采用神经网络方法的原因。

对于单气泡和多气泡算例我们使用粗化网格进行了 CFD 计算, 并把相关结果和粗化后的原始精细网格的 CFD 计算结果、以及使用 DNN 和 BubbleNet 的预测结果进行了对比, 如图4-7所示。对于单气泡算例原始 CFD 计算精密网格数为 24182, 粗化后的神经网络训练数据为 2419, 而采用粗化网格计算的网格数为 2102, 基本维持在同一数量级, 以便进行对比。我们可以明显看出, 粗化网格计算结果不准确且难以

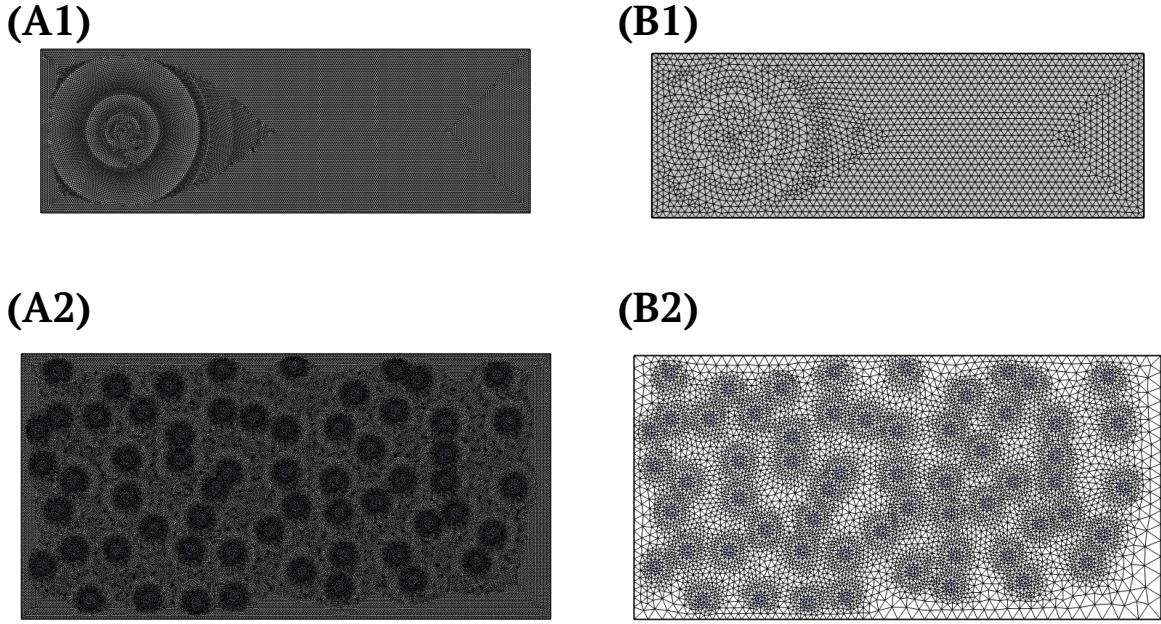


图 4-8 针对两个气泡流算例采用 CFD 计算的精细化网格和粗化网格。(A1) 单气泡的精细网格。(A2) 多气泡的精细网格。(B1) 单气泡的粗化网格。(B2) 多气泡的粗化网格。

捕捉气泡运动的形貌细节；而采用粗化数据训练的神经网络且可以预测出气泡运动趋势。对于多气泡算例，原始 CFD 计算精密网格数为 75302，粗化后的神经网络训练数据为 3766，而采用粗化网格计算的网格数为 7833；粗化网格数之所以依然为训练数据的两倍之多，因为在软件操作中我们已经采用了最粗的网格，无法进行更少的网格数生成。而多气泡算例中机器学习的优势体现更加明显：尽管采用 DNN 预测结果已经相较 BubbleNet 不准确，但是依然比采用粗化网格进行 CFD 计算拥有更精确的趋势捕捉。CFD 网格划分如图4-8所示。总而言之，神经网络对于粗化训练数据的预测方法相较直接在粗网格进行 CFD 计算有更高的精度，而我们提出的 BubbleNet 则比传统 DNN 预测更准。

4. 4 训练过程与计算结果

基于以上两种神经网络结构，我们分别使用并对单气泡和多气泡算例进行了训练和预测。我们首先讨论单气泡算例计算结果。两种神经网络训练结果如图4-9所示。图4-9显示在训练过程中损失函数出现了“梯度爆炸点”(exploding gradients)，但是很快梯度继续正常下降。这主要是由于我们使用的优化器导致的。在训练深度神经网络时，随着隐层增加，权重的误差会呈指数级放大，致使训练梯度发生急剧增大^[88]。

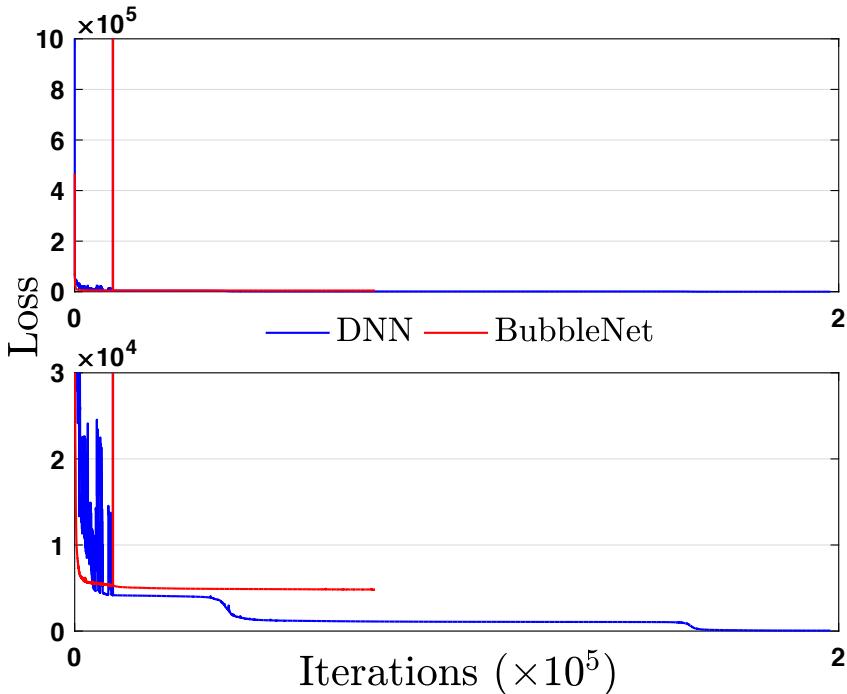


图 4-9 两种网络结构训练的损失函数。

而在本文中，因为使用了‘L-BFGS-B’ 优化算法以及 Adam 优化器，神经网络的训练步骤可以理解为：先进行设定次数的迭代次数（单气泡算例 10000，多气泡 200000），在迭代完成后进行 L-BFGS-B 优化器的迭代。在优化器迭代至一定次数后损失函数很难减小，此时便认为训练完成。而我们算例中的梯度爆炸正是发生在迭代次数完成后，采用 L-BFGS-B 优化的交替步骤，所以之后损失函数下降依然符合正常趋势。

由图4-9我们得到损失函数下降的趋势为：DNN 迭代次数更长久，损失函数数值下降至更小；而 BubbleNet 梯度下降至一固定数值后便结束迭代完成训练，最终损失函数数值明显大于传统 DNN。在此我们提供两点可能解释：第一，利用两种归一化器计算后的训练数据特征不同，网络对其适应性也不同；第二，因为我们在 BubbleNet 的误差函数中引入了物理信息（通过预测 ψ 算出 u, v ），所以该结构误差总额较大，使得 ψ 不能同时精准拟合 u, v 的训练数据。

两种网络结构对于单气泡算例速度场 u 预测结果如图4-10所示。图中，我们可以观察到 BubbleNet 预测流场趋势与实际计算结果基本相符，而传统的 DNN 预测结果却相差较大。在前文我们提到，造成该现象的主要原因是在对整个时空域的数据进行归一化后，由于在特定时间步某些物理量数值较大，从而“剥夺”了其他时间

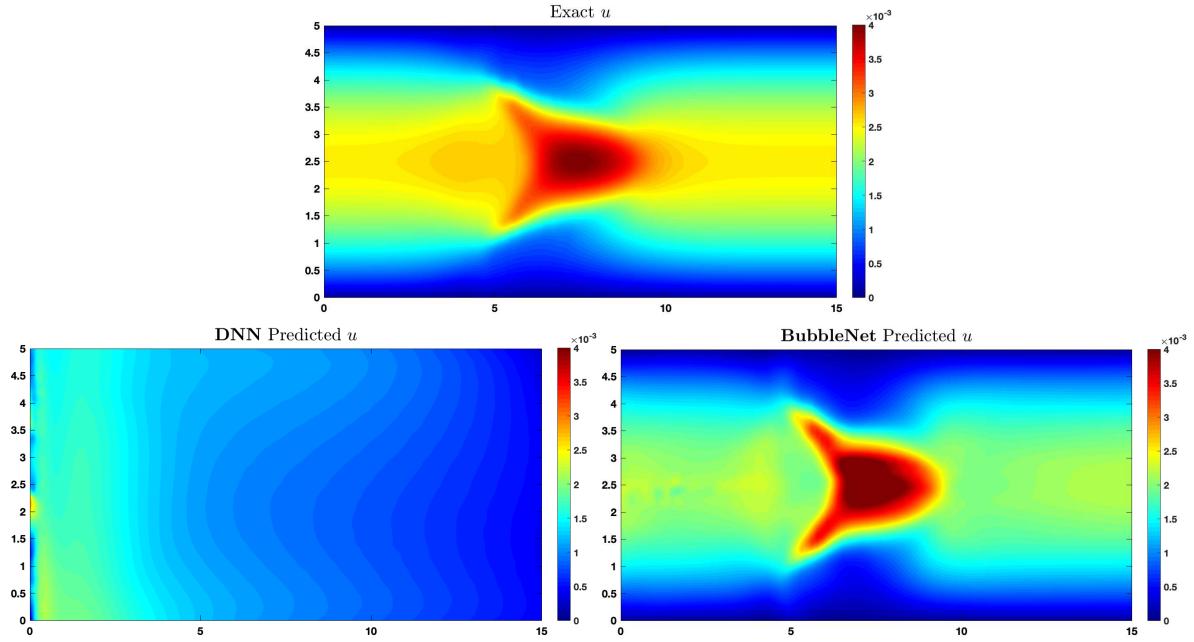


图 4-10 基于物理仿真结果与普通深度神经网络 DNN 和物理神经网络 BubbleNet 预测的速度场 u 分布。

步该物理量的特征，以致进行整体归一化后在该时间步上整体数值都趋于 0（或量级上非常小），该数据特征因此“欺骗”神经网络使其误以为对该时间步的预测为准确的。而从图4-10的特征，我们可以基本判断速度场 u 分布满足这种特征。

如图4-11, 4-12, 4-12则分别展示了两种神经网络对单气泡算例的速度场 v , 压力场 p , 以及相函数 ϕ 的预测。通过四个物理场预测结果我们可以判断无论从数值范围还是运动趋势上 BubbleNet 预测结果均优于传统的 DNN；并且速度场 $u & v$ 都符合前文描述的由于整体归一化导致的“特征被剥夺”。因此 BubbleNet 预测效果优势更为突出。但是，从压力场 p 分布我们也可以看出两种神经网络在预测时都会造成细节遗失，即便是 BubbleNet 在预测压力场也遗失气泡部分压场变化。对此，我们给出两个解释：首先是压力场左端与右端压差较大，神经网络在训练时主要考虑了两侧压差这个的明显特征，以至于忽略了气泡的细节特征；同时我们对于神经网络训练迭代次数仅有 10000，使得网络训练不充分不完全。而对于描述气泡构型的相函数，传统 DNN 已经表现出彩，可以基本预测出整体气泡形态；对比图4-13左下图场中有部分浅蓝色区域，我们发现 BubbleNet 的突出点在于对于场背景预测也更加准确了。

在网络训练中，针对不同的归一化方法产生的不同数据，网络训练目标也不同。网络基于损失函数评估训练过程，而损失函数在此可以理解为网络的训练误差，此

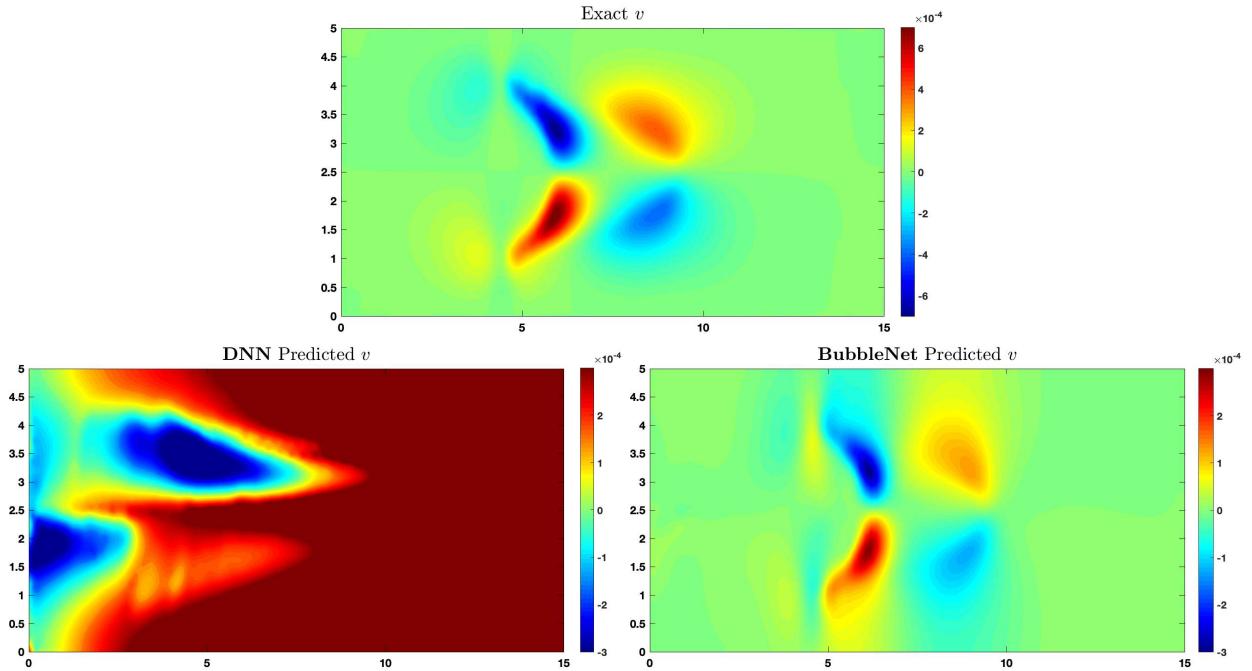


图 4-11 基于物理仿真结果与普通深度神经网络 DNN 和物理神经网络 BubbleNet 预测的速度场 v 分布。

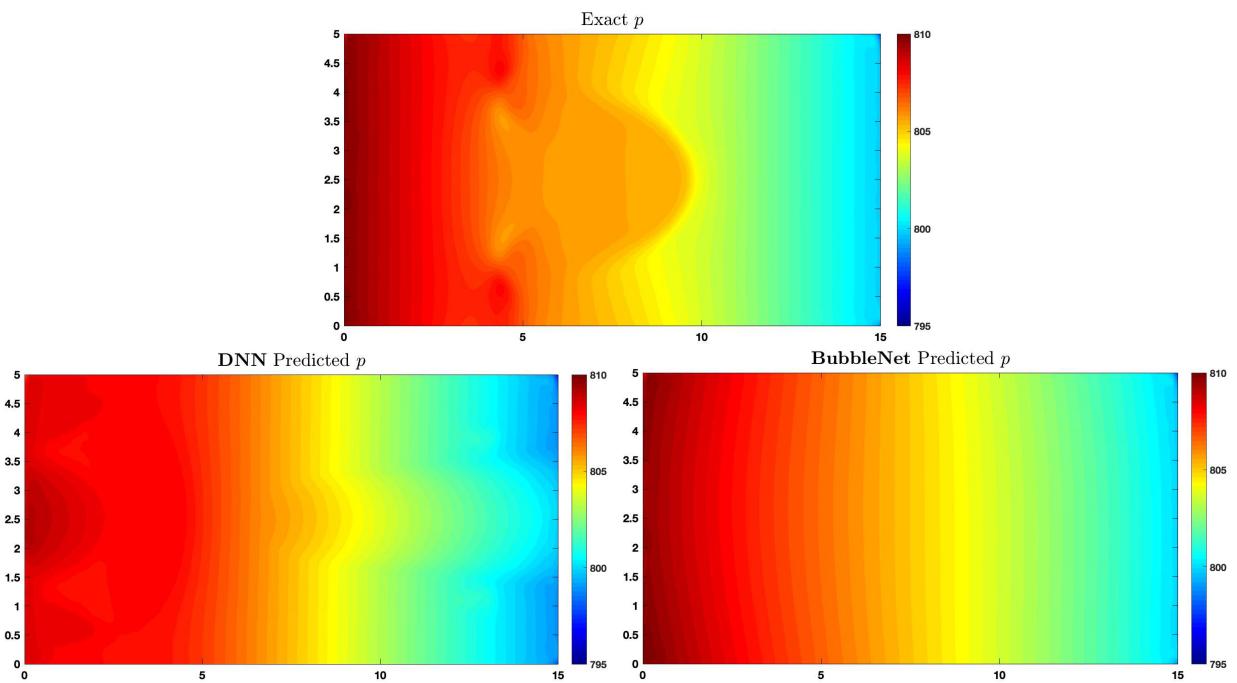


图 4-12 基于物理仿真结果与普通深度神经网络 DNN 和物理神经网络 BubbleNet 预测的压力场 p 分布。

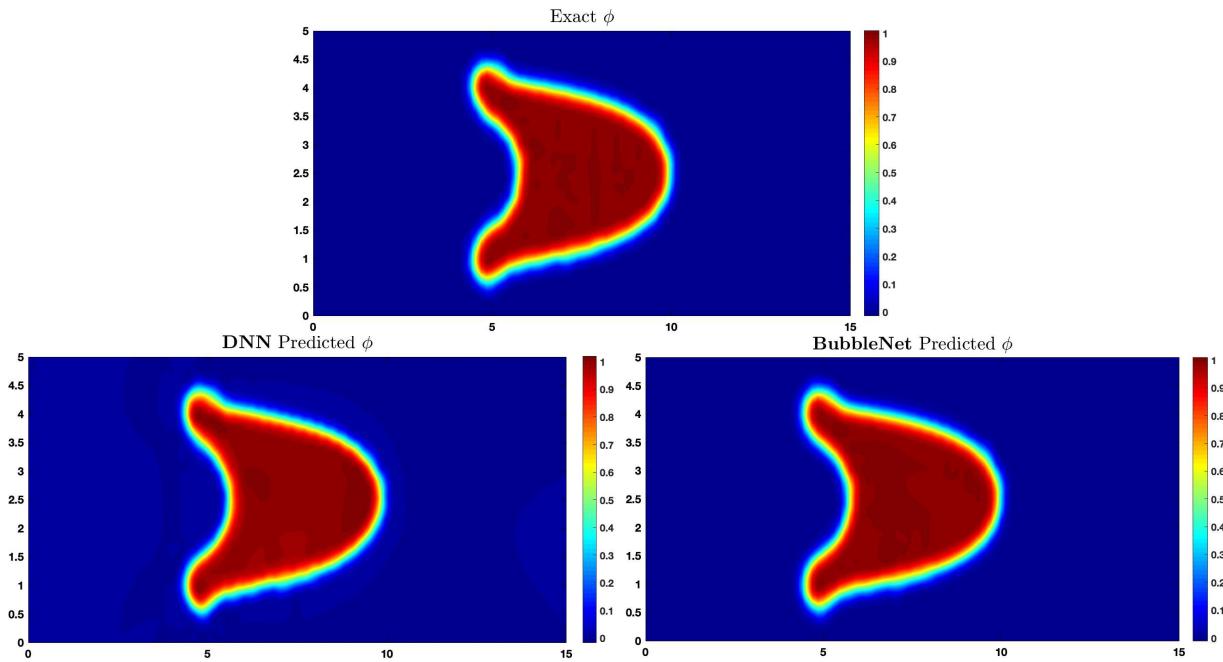


图 4-13 基于物理仿真结果与普通深度神经网络 DNN 和物理神经网络 BubbleNet 预测的相函数 ϕ 分布。

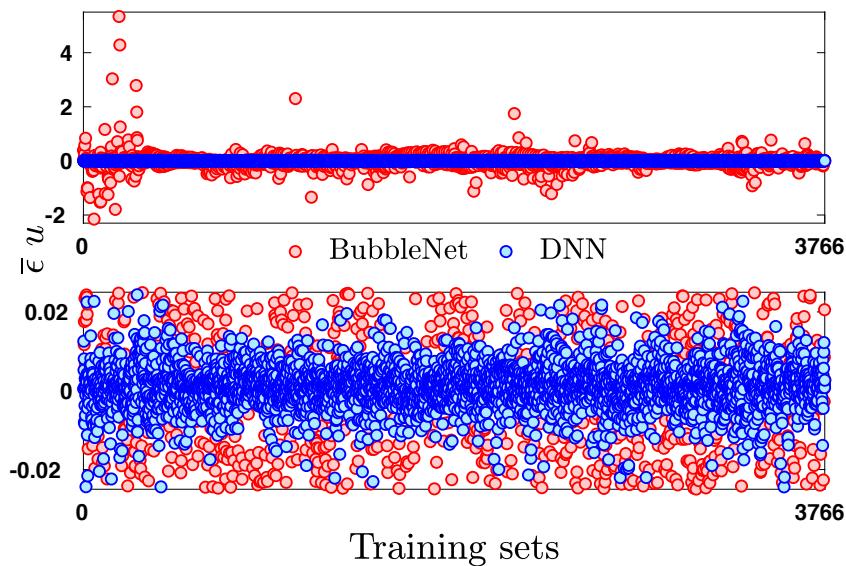


图 4-14 针对速度场 u 的 DNN & BubbleNet 神经网络训练相对误差 $\bar{\epsilon}$ 。上图为适应 BubbleNet 数值尺度的误差图，下图为适应 DNN 数值尺度的误差图。

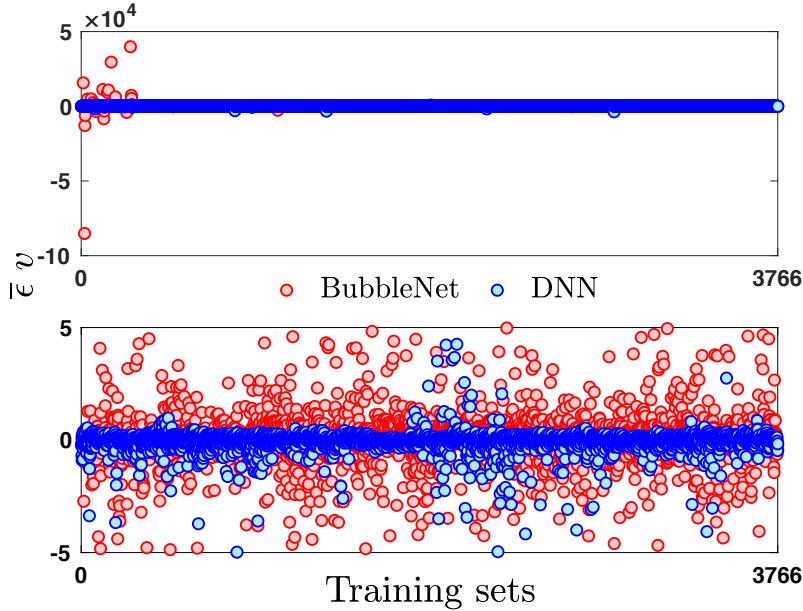


图 4-15 针对速度场 v 的 DNN & BubbleNet 神经网络训练相对误差 $\bar{\epsilon}$ 。上图为适应 BubbleNet 数值尺度的误差图，下图为适应 DNN 数值尺度的误差图。

处用 $\bar{\epsilon}$ 指代。既网络针对输入训练数据所产生预测数值相对误差，可写作

$$\bar{\epsilon} = \frac{|\mathbf{W}_{train} - \mathbf{W}_{pred}|}{|\mathbf{W}_{train}|}$$

其中 \mathbf{W} 指代物理系数，此处代表 u, v, p, ϕ 。

速度场 u 的训练误差 $\bar{\epsilon}$ 如图4-14所示。由图我们可以明显看出 BubbleNet 训练误差高于传统 DNN。

由图4-15我们也观察到同样的趋势：BubbleNet 训练误差高于 DNN。对此，我们根据前文给出的 $\bar{\epsilon}$ 定义给出如下解释：相对误差计算中，当分母为输入数据 \mathbf{W}_{train} 太小时，例如为 10^{-9} 量级，即使神经网络训练的绝对误差仅在 10^{-3} 量级，计算相对误差也会达到 10^6 量级，造成我们看到的 BubbleNet 训练误差尺度较大。

同理，压力场 p ，组分分布函数 ϕ 的两种神经网络计算相对误差如图4-16, 4-17所示。我们可以用上述理论解释相函数误差较大的原因。特殊情况为训练所得压力 p 的相对训练误差在此处更小；可以说明在训练上我们提出 BubbleNet 的拟合效果也更好。

在分析神经网络训练效果时，仅考虑训练相对误差是不够的，因为这反映了

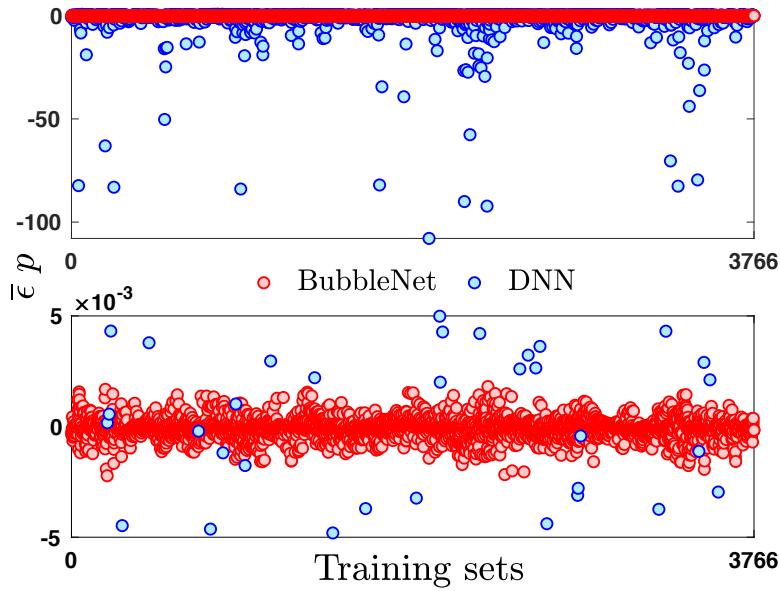


图 4-16 针对压力场 p 的 DNN & BubbleNet 神经网络训练相对误差 $\bar{\epsilon}$ 。上图为适应 DNN 数值尺度的误差图，下图为适应 BubbleNet 数值尺度的误差图。

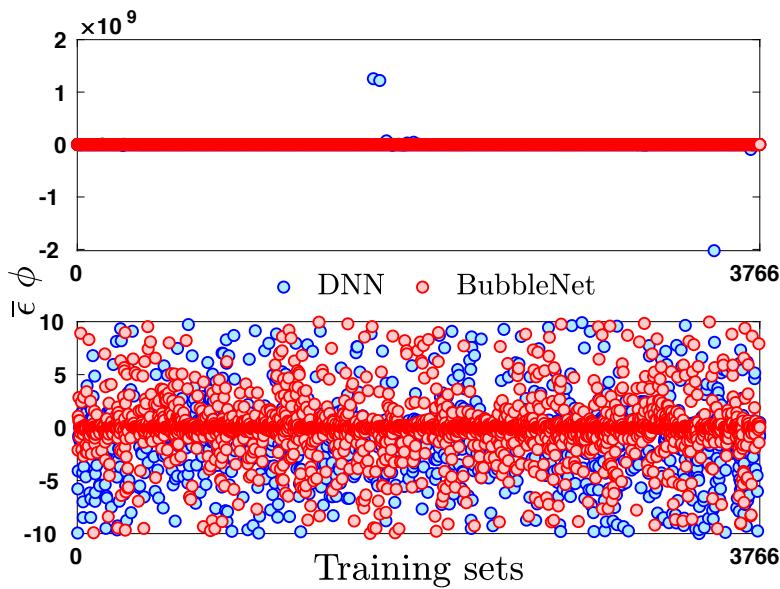


图 4-17 针对相函数 ϕ 的 DNN & BubbleNet 神经网络训练相对误差 $\bar{\epsilon}$ 。上图为适应 DNN 数值尺度的误差图，下图为适应 BubbleNet 数值尺度的误差图。

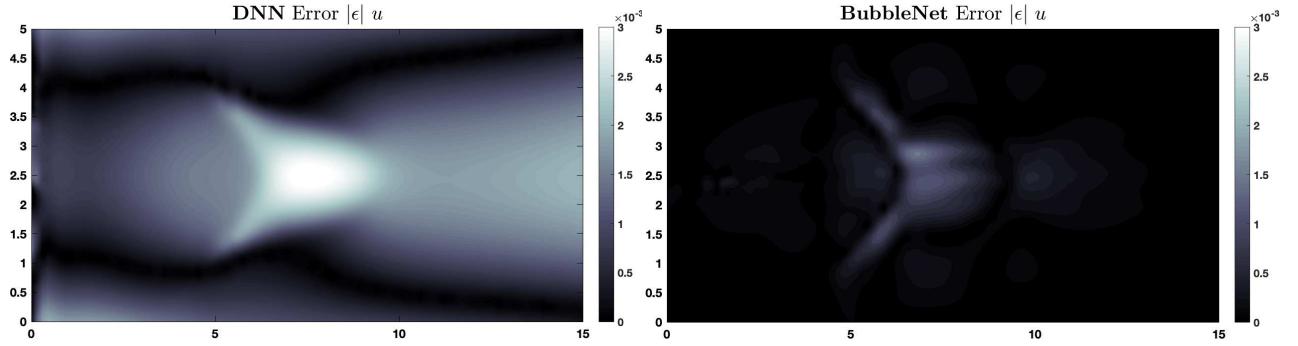


图 4-18 针对速度场 u 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

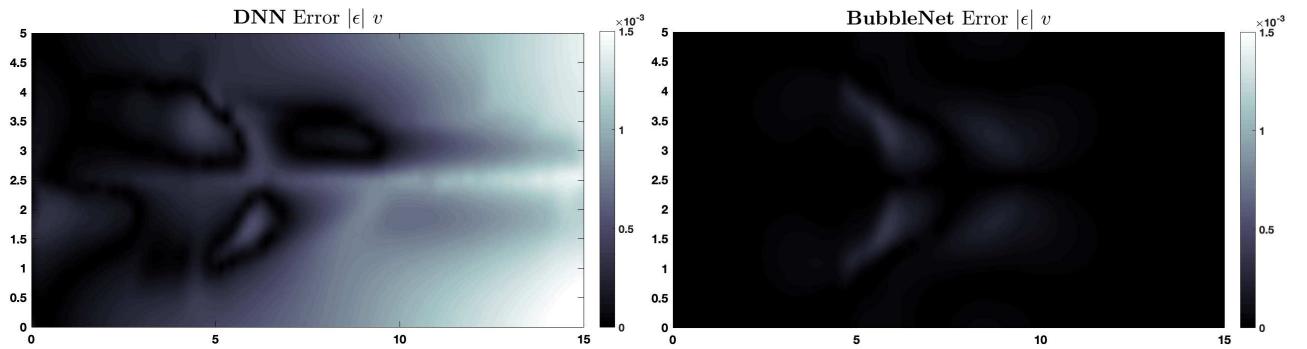


图 4-19 针对速度场 v 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

网络拟合效果的准确度。对于实际物理问题，我们需要权衡物理场的绝对误差。DNN 和 BubbleNet 对物理场 u, v, p, ϕ 预测的绝对误差 $|\epsilon|$ ，如图4-18, 4-19, 4-20, 4-21所示。

由图4-18, 4-19, 4-20我们可看出 BubbleNet 的绝对误差均明显小于传统 DNN，但是两者误差大体分布趋势相同。仅是描述气泡运动形态的组分分布 ϕ 传统 DNN 误差略小于我们提出的 BubbleNet (图4-21)，这可能是由于相函数本身分布区间为 $[0, 1]$ ，我们的离散时间归一器以及物理限制在此处不起明显作用。但对于其他所有的物理场来看，我们的 BubbleNet 所表现出的绝对误差均更小。总体来看，对于单气泡算例，我们提出的深度学习框架在误差上显著高于 DNN，并迭代次数更少 (图4-9)。

对于多气泡算例，分析思路同单气泡算例。理论部分此处不再赘述。首先，两种不同网络针对不同算例训练损失函数下降过程如图4-22所示：我们提出的 BubbleNet 明显以更快的速度进行梯度下降并且误差更小。传统深度神经网络在训练时梯度处在相对剧烈的波动中，没有稳定下降。此处我们给出的解释为：I、多气泡流动场物理数据有很多明显变化，即很多个小气泡与周围场的组分分布的数值间断难以被神经网络捕捉；II、根据我们给出的周期性边界条件，压力、速度等物理量在时间域上变

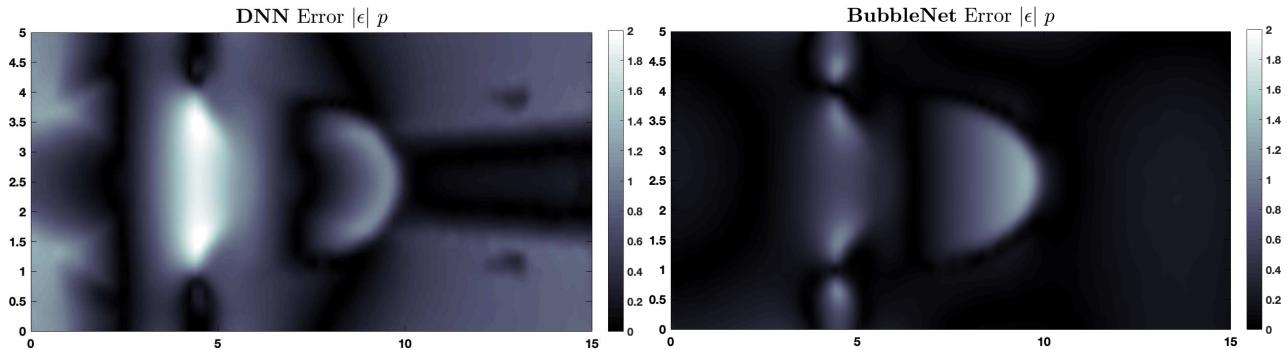


图 4-20 针对压力场 p 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

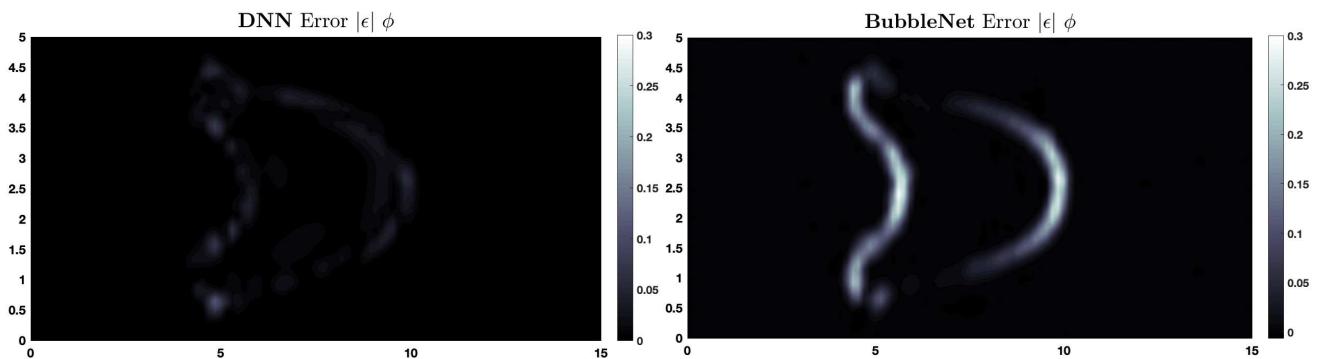


图 4-21 针对组分分布函数 ϕ 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

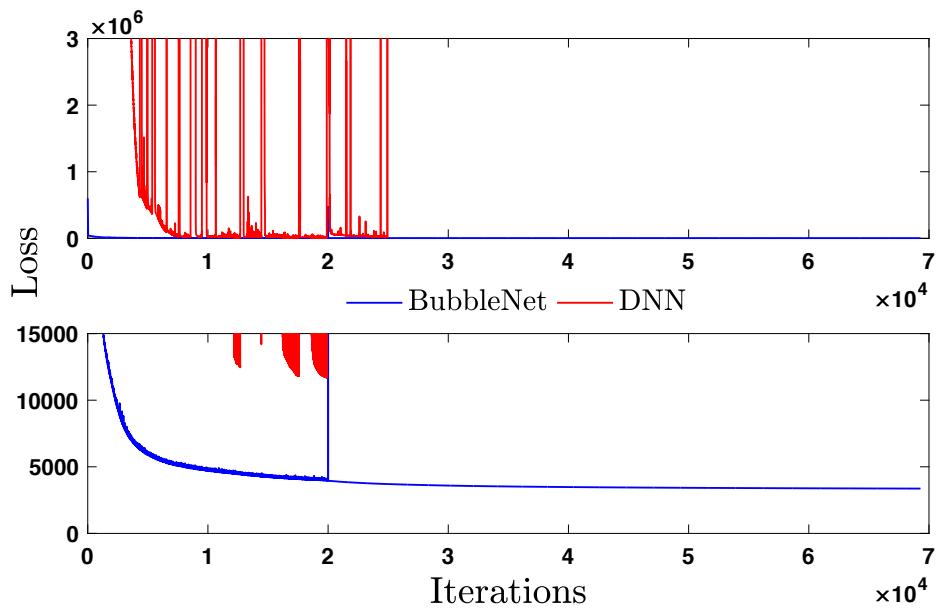


图 4-22 DNN & BubbleNet 针对多气泡算例训练的损失函数下降过程。

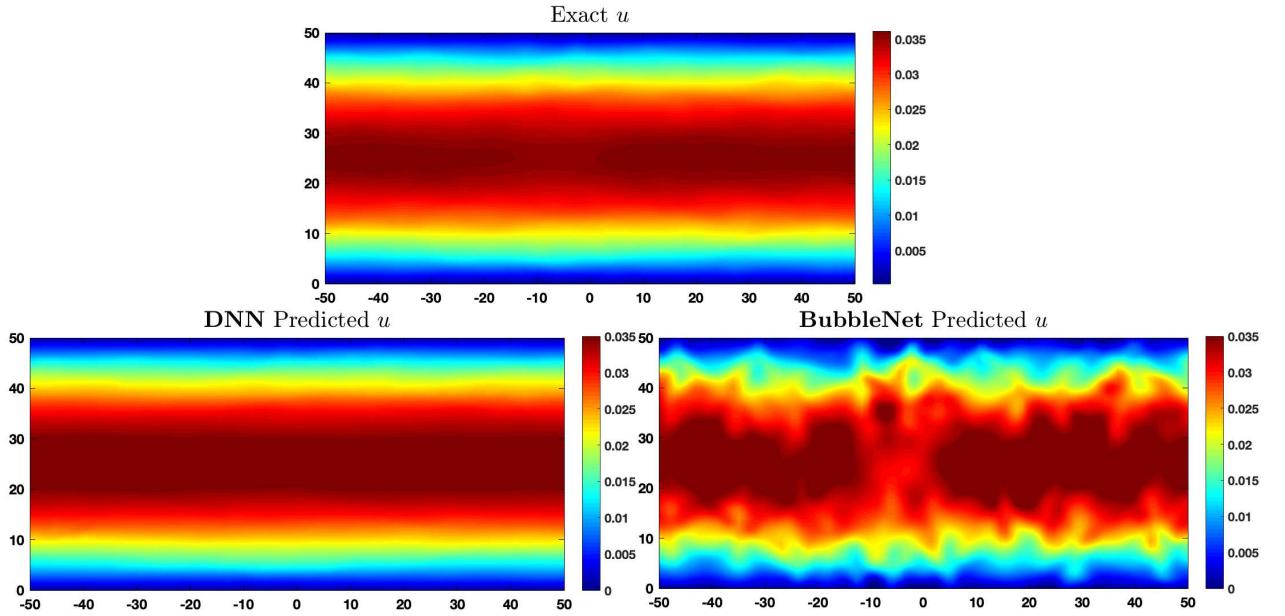


图 4-23 针对速度场 u 的仿真计算与 DNN & BubbleNet 的预测结果和数值模拟结果对比。

化较大，在每个时间步上又存在间断，因此使得神经网络在拟合这样的数值场时难以兼顾多特征，因此上一次迭代产生的权重在下一次训练验证就产生较大误差，以至于损失函数波动较大。但是时间离散归一器使得每个时间步的数据处在同一尺度上，因此拟合误差相对较小。

对多气泡流场 DNN 和 BubbleNet 预测的物理场 u, v, p, ϕ 如图4-23, 4-24, 4-25, 4-26所示。图中显示，DNN 仅在速度场 u 上预测效果好于我们提出的 BubbleNet。此处我们给出的解释是，由于速度两分量 u, v 是由潜在函数 ψ 耦合在一起的（为满足连续条件），且 BubbleNet 在预测时侧重拟合速度场 v 的趋势，因此导致 u 出现了很多不必要的细节，但是预测速度场整体上无论在数值大小还是在趋势上都基本符合原场分布。图4-24, 4-25, 4-26验证了 BubbleNet 对于整体物理场在预测效果上均好于 DNN，无论是趋势还是数值拟合都非常好地符合原场分布。

为了印证神经网络训练效果的好坏，由前面提到的相对误差定义 $\bar{\epsilon}$ ，此处图4-27, 4-28, 4-29, 4-30分别展示了 DNN 和 BubbleNet 训练相对误差分布。由上图可得出，在速度场 u, v 的预测上，传统的 DNN 的训练相对误差低于 BubbleNet；在压力场 p 和组分分布函数 ϕ 上 BubbleNet 高于 DNN。同时对于组分分布函数 ϕ 两个网络总体训练效果相差不大。以此可以说明 BubbleNet 在训练上并没有体现出明显优势，主要优势在于实际预测的场数据更符合原物理分布，这也和前文中对单气泡算例的分析

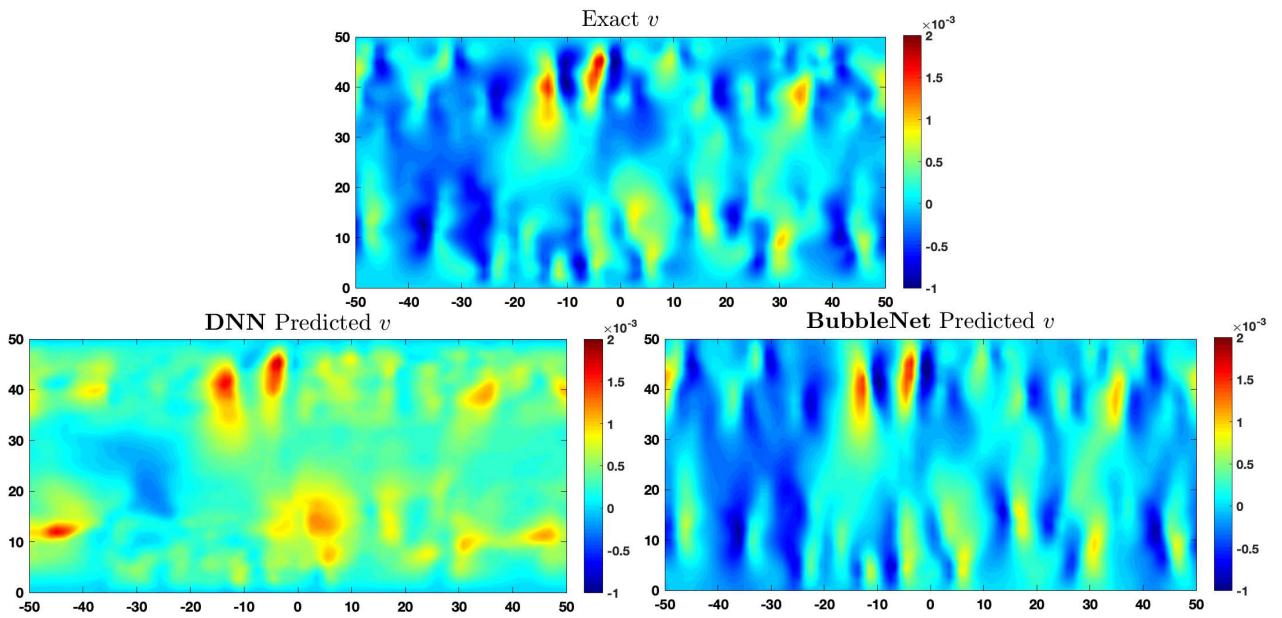


图 4-24 针对速度场 v 的仿真计算与 DNN & BubbleNet 的预测结果和数值模拟结果对比。

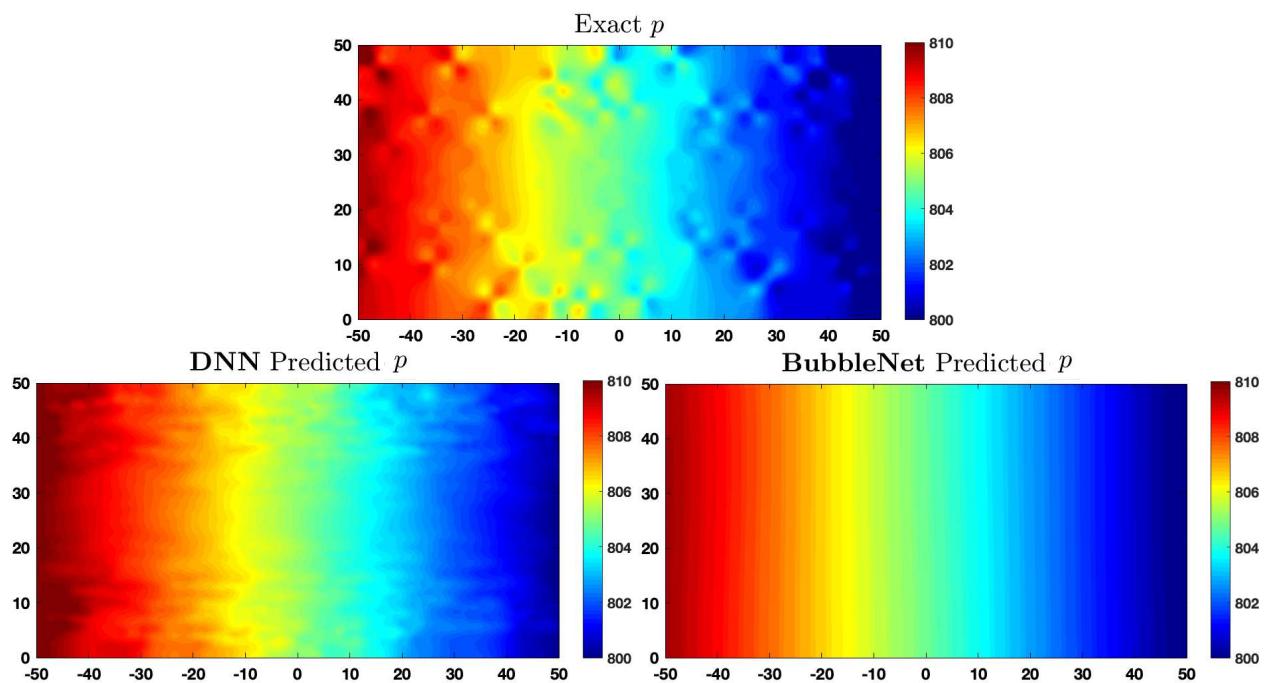


图 4-25 针对压力场 p 的仿真计算与 DNN & BubbleNet 的预测结果和数值模拟结果对比。

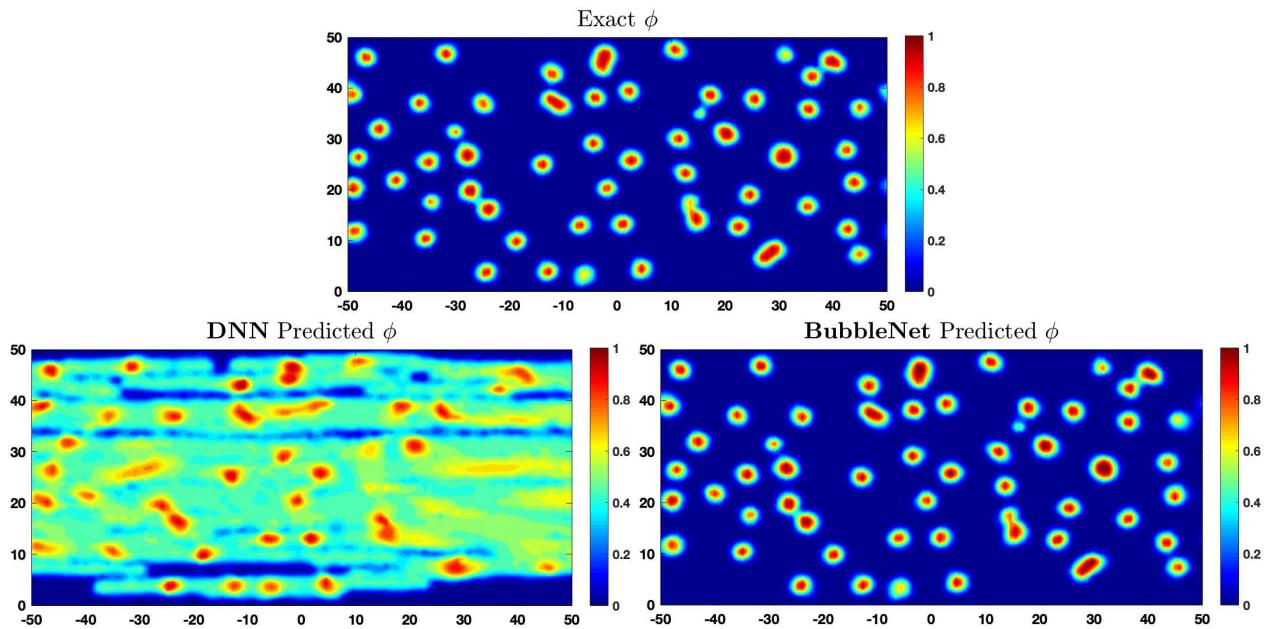


图 4-26 针对组分分布函数 ϕ 的仿真计算与 DNN & BubbleNet 的预测结果和数值模拟结果对比。

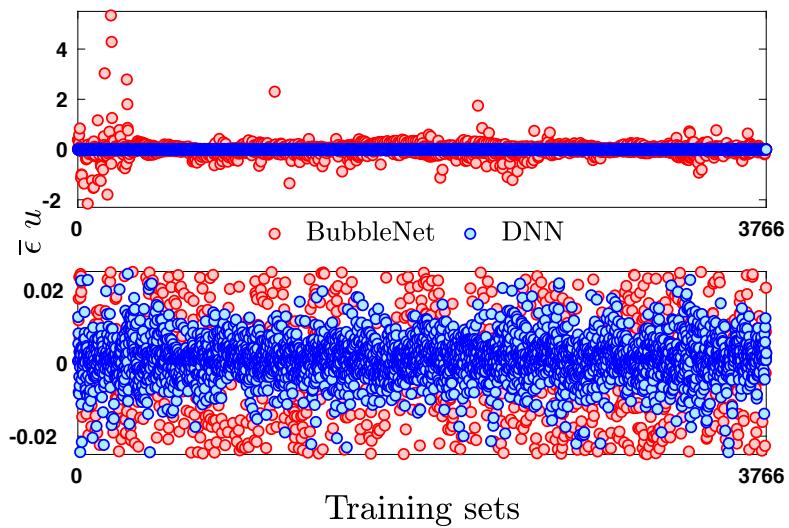


图 4-27 针对速度场 u 的仿真计算与 DNN & BubbleNet 的相对误差 $\bar{\epsilon}$ 。上图为适应 BubbleNet 数值尺度的误差图，下图为适应 DNN 数值尺度的误差图。

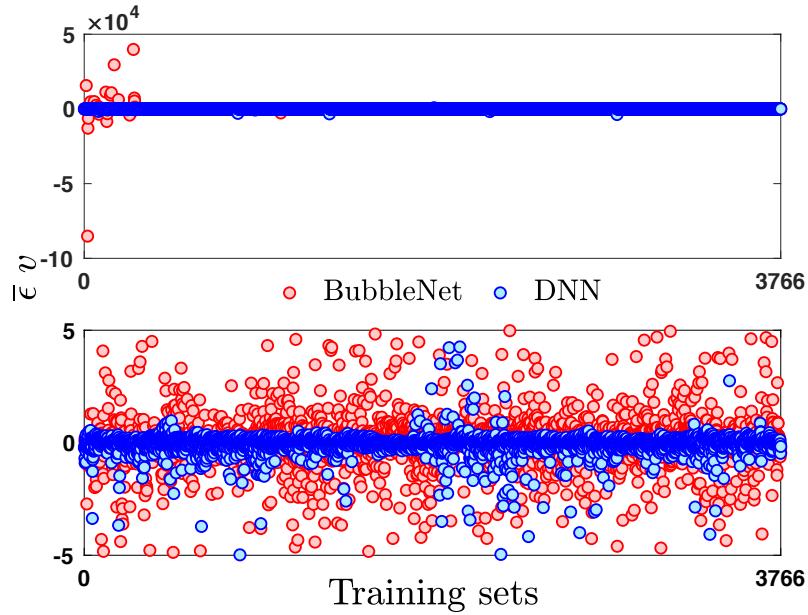


图 4-28 针对速度场 v 的仿真计算与 DNN & BubbleNet 的相对误差 $\bar{\epsilon}$ 。上图为适应 BubbleNet 数值尺度的误差图，下图为适应 DNN 数值尺度的误差图。

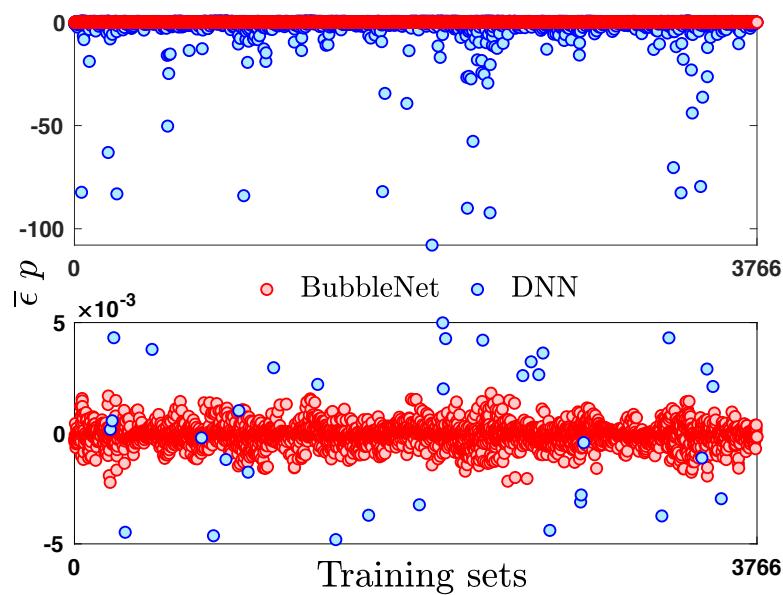


图 4-29 针对压力场 p 的仿真计算与 DNN & BubbleNet 的相对误差 $\bar{\epsilon}$ 。上图为适应 DNN 数值尺度的误差图，下图为适应 BubbleNet 数值尺度的误差图。

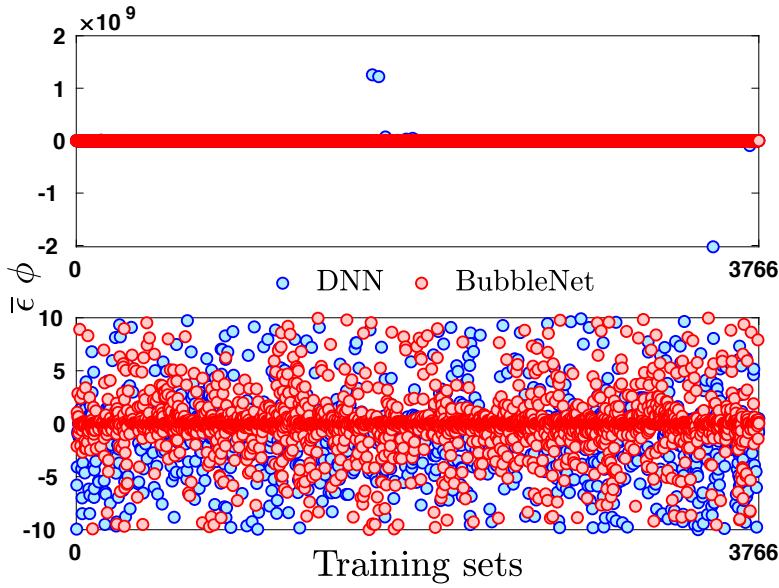


图 4-30 针对组分分布函数 ϕ 的仿真计算与 DNN & BubbleNet 的相对误差 $\bar{\epsilon}$ 。上图为适应 DNN 数值尺度的误差图，下图为适应 BubbleNet 数值尺度的误差图。

结论是相同的。

最后我们分析 DNN 和 BubbleNet 对于真实物理分布的绝对误差 $|\epsilon|$ ，对物理场 u, v, p, ϕ 的绝对误差分布如图4-31, 4-32, 4-33, 4-34所示。基本分布同我们在前文分析的相同：DNN 对于速度场 u 的绝对误差更小，预测更准确；但是 BubbleNet 在其他物理场 v, p, ϕ 预测时绝对误差均明显小于 DNN，预测精度更高。

对于单气泡和多气泡两个算例下 DNN 和 BubbleNet 的预测过程和结果分析我们均可以得出，BubbleNet 相较于 DNN 的优势主要体现对于物理场分布精度更高，更准确（通过绝对误差 $|\epsilon|$ 得出）。但是在网络训练本身，我们很难明确 DNN 和 BubbleNet 的训练效果好坏，这和数据本身的趋势和结构有关。总而言之，针对微气泡流动的两个算例 BubbleNet 的预测效果均更好，我们因此认为该结构是有效且具有鲁棒性的。

4.5 本章小结

在本章节中，作者首先概括了本章节以及本文的灵感和动机：设计一种针对气泡流动的物理激发的神经网络用于繁复科学计算的替代品。而后，作者首先简单介绍了其深度学习框架中用到的两种优化器 Adam 和 L-BFGS-B 的基本原理和数学基础。而后，作者分别介绍了利用传统的深度神经网络（DNN）和提出的基于物理的气泡流动神经网络（BubbleNet）对气泡流动的物理问题如何建模，如何训练和计算。

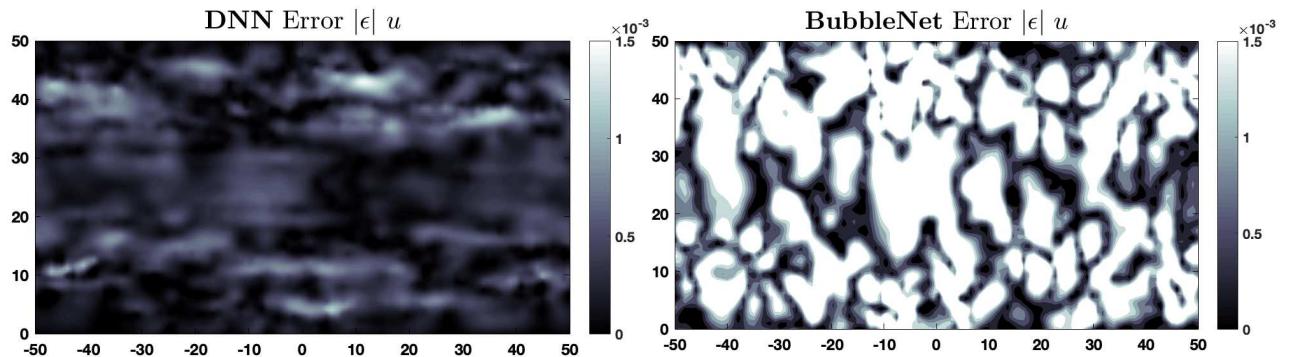


图 4-31 针对速度场 u 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

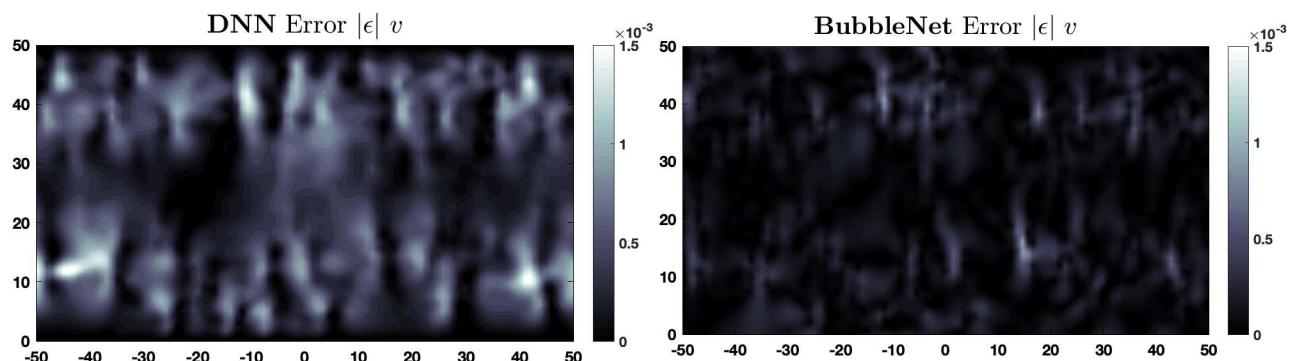


图 4-32 针对速度场 v 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

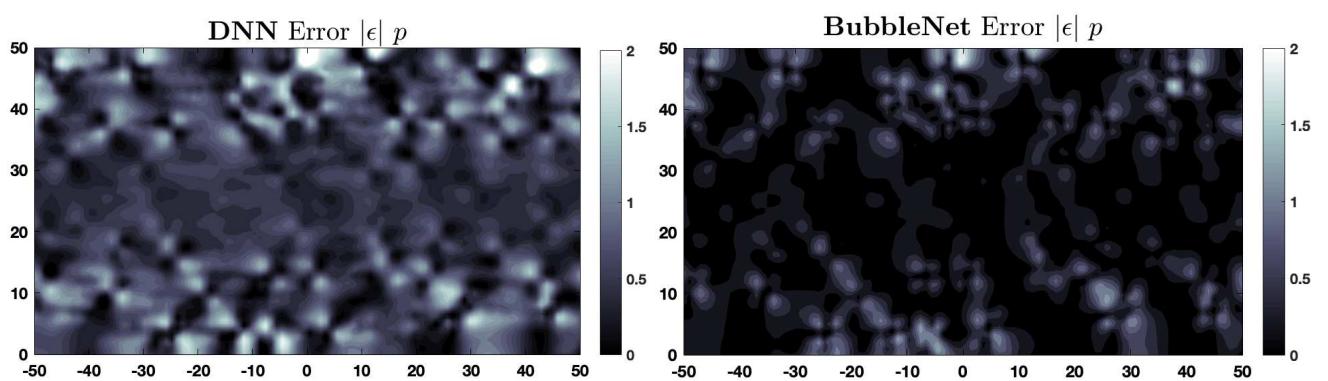


图 4-33 针对压力场 p 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

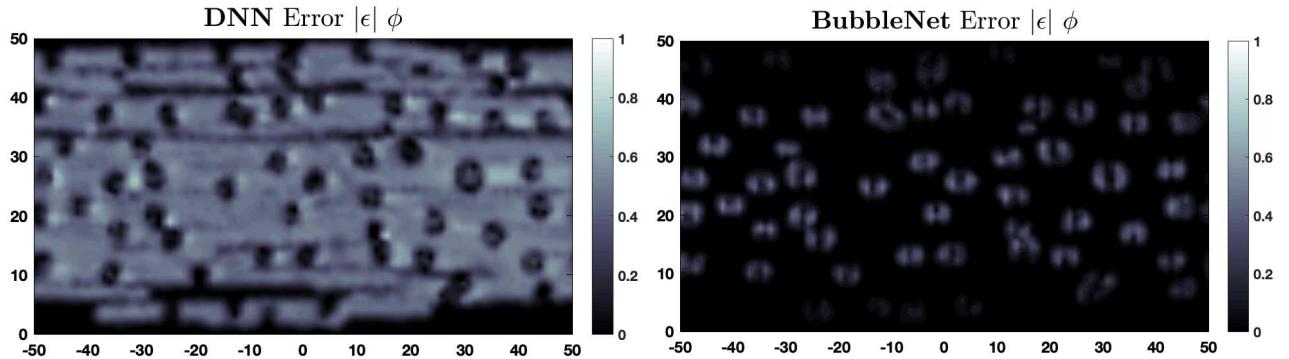


图 4-34 针对组分分布函数 ϕ 的仿真计算与 DNN & BubbleNet 的绝对误差 $|\epsilon|$ 。

同时，作者分析时间归一化器 TDN 的使用原理的效果；以及在本文中为什么要使用粗化后的计算数据和分析。最后，作者给出不同深度学习框架的计算结果并对误差进行了分析，证明了 BubbleNet 的优异性能。

结 论

为了解决在化工、生物领域常见的多相流问题，特别是微管道中气泡流动问题，我们使用了近些年来发展迅速的数据驱动物理深度学习方法来预测微气泡流。首先我们基于数值模拟，分别计算了在微米尺度下微管道中单气泡与多气泡系统的运动规律。模拟仿真中计算参数基于生物学应用背景给出。单气泡算例计算了微管道中单气泡运动 $5000\mu\text{s}$ 的过程；多气泡算例则计算微管道中气泡群运动 $3000\mu\text{s}$ 的过程。仿真结果显示单气泡流动变形形态为前段先外凸呈抛物线状，尾端内凹直至附着气泡剥离气泡主体；微气泡群中气泡会随着时间推进发生融合及剥离。单气泡算例与现有文献中微管道实验中血红细胞变形形貌相似，管壁尾端在液相流部分符合泊肃叶流动基本速度分布。微气泡群算例中，整体流体速度分布符合泊肃叶流动规律。对本文的两个算例，我们通过对组分进行分析印证算例中流体组分守恒，验证了计算的精确度。

基于物理神经网络^[47]，我们提出了一种内部嵌构了连续性方程的针对气泡运动的神经网络 BubbleNet。我们的网络由两部分构成：构成基本的深度神经网络（DNN）以及包含连续不可压条件的内置方程（物理信息）。同时，我们还引入了时间离散监督器：即对于场数据进行时间离散化的归一化。我们分别用 DNN 和 BubbleNet 预测了单气泡在 $2000\mu\text{s}$ 和多气泡系统在 $1500\mu\text{s}$ 时刻物理场 (u, v, p, ϕ) 的分布。我们的 BubbleNet 结构相较于传统的 DNN 可以以更小的迭代步数获得更高的预测精度。因为对整个时空域上不同时间步流场数值的较大差异可能会导致特定时间步特征被“剥夺”，使得训练神经网络时网络被“欺骗”，所以时间离散归一器使网络预测时空数据更准确。对于我们给出的两个算例：微管道中的单气泡流动和气泡群流动中，与数值模拟得到的结果相比（根据与数值模拟的绝对误差 $|\epsilon|$ 对比），BubbleNet 在预测精度上均高于传统的 DNN；但是对于网络本身的训练效果来讲（神经网络训练的相对误差 $\bar{\epsilon}$ 判定），BubbleNet 的性能相较 DNN 并没有明显提高。本文提出的相关算法成果不仅限于微气泡流，更可以广泛应用与电化学、电磁、燃烧等具有广泛工程应用背景的数学物理问题。

参考文献

- [1] Mitchell, T. (1997). Machine Learning. New York: McGraw Hill. ISBN 0-07-042807-7.
- [2] Silver, D., Huang, A., Maddison, C. et al. (2016) Mastering the game of Go with deep neural networks and tree search. *Nature* **529**, 484–489.
- [3] Silver, D., Schrittwieser, J., Simonyan, K. et al. (2017) Mastering the game of Go without human knowledge. *Nature* **550**, 354–359.
- [4] Jo, T., Nho, K., and Saykin, A.J. (2019) Deep Learning in Alzheimer’s Disease: Diagnostic Classification and Prognostic Prediction Using Neuroimaging Data. *Front. Aging Neurosci.* **11**, 40, 220.
- [5] Waring, J., Lindvall, J., and Umeton, R. (2020) Automated machine learning: Review of the state-of-the-art and opportunities for healthcare. *Artificial Intelligence in Medicine* **104**, 101822.
- [6] Woldaregay, A.Z., Årsand, E., Walderhaug, S., et al. (2019) Data-driven modeling and prediction of blood glucose dynamics: Machine learning applications in type 1 diabetes. *Artificial Intelligence in Medicine* **98**, 109-134.
- [7] Benke, K., and Benke, G. (2018) Artificial Intelligence and Big Data in Public Health. *International Journal of Environmental Research and Public Health*. **15**(12):2796.
- [8] Panch, T., Pearson-Stuttard, and J., Greaves, F., et al. (2019) Artificial intelligence: opportunities and risks for public health. *The Lancet Digital Health* **1**, 1, 13-14.
- [9] Jarek, K., and Mazurek, G. (2019) Marketing and Artificial Intelligence. *Central European Business Review*, **8**(2), 46-55.
- [10] Davenport, T., Guha, A., and Grewal, D. (2020) How artificial intelligence will change the future of marketing. *Journal of the Academy of Marketing Science* **48**, 24–42.
- [11] Senior, A.W., Evans, R., Jumper, J., et al. (2020) Improved protein structure prediction using potentials from deep learning. *Nature* **577**, 706–710.

- [12] Harmon, S.A., Sanford, T.H., Xu, S., et al. (2020) Artificial intelligence for the detection of COVID-19 pneumonia on chest CT using multinational datasets. *Nat. Commun.* **11**, 4080.
- [13] Chassagnon, G., Vakalopoulou, M., Battistella, E., et al. (2021) AI-driven quantification, staging and outcome prediction of COVID-19 pneumonia. *Medical Image Analysis* **67**, 101860.
- [14] Li, Y., Shang, K., Bian, W., et al. (2020) Prediction of disease progression in patients with COVID-19 by artificial intelligence assisted lesion quantification. *Sci. Rep.* **10**, 22083.
- [15] Punn, N.S., Sonbhadra, S.K., and Agarwal, S. (2020) COVID-19 Epidemic Analysis using Machine Learning and Deep Learning Algorithms. *medRxiv*:10.1101/2020.04.08.20057679.
- [16] Curry, B., and Moutinho, L. (1993). Neural Networks in Marketing: Modelling Consumer Responses to Advertising Stimuli. *European Journal of Marketing* **27**, 5-20.
- [17] Kaefer, F., Heilman, C.M., and Ramenofsky, S.D. (2005) A neural network application to consumer classification to improve the timing of direct marketing activities. *Computers & Operations Research* **32**, 10, 2595-2615.
- [18] Zhou, G., Zhu, X., Song, C., et al. (2018) Deep Interest Network for Click-Through Rate Prediction. In Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining (KDD '18). Association for Computing Machinery, New York, NY, USA, 1059–1068.
- [19] Abadi, M., Agarwal, A., Barham, P., et al. (2016) TensorFlow: Large-Scale Machine Learning on Heterogeneous Distributed Systems. *arXiv*. *arXiv*:1603.04467
- [20] Tsai, CW., Lai, CF., Chao, HC., et al. (2015) Big data analytics: a survey. *Journal of Big Data* **2**, 21.

- [21] Pääkkönen, P., and Pakkala, D. (2015) Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems. *Big Data Research* **2**, 4, 166-186.
- [22] Zhang, W. J., Yang, G., Lin, Y., et al. (2018) On Definition of Deep Learning. 2018 World Automation Congress (WAC) Stevenson, WA, pp. 1-5.
- [23] Mishra, M., and Srivastava, M. (2014) A view of Artificial Neural Network. 2014 International Conference on Advances in Engineering & Technology Research (ICAETR - 2014), Unnao, pp. 1-3.
- [24] Mehlig, B. (2019) Artificial Neural Networks. *arXiv*. arXiv:1901.05639
- [25] Bebis, G., and Georgopoulos, M. (1994) Feed-forward neural networks. *IEEE Potentials*, **13**, 4, 27-31, Oct.-Nov.
- [26] Sherstinsky, A. (2020) Fundamentals of Recurrent Neural Network (RNN) and Long Short-Term Memory (LSTM) network. *Physica D: Nonlinear Phenomena* **404**, 132306.
- [27] Mikolov, T., Karafiat, M., Burget, L., et al. (2010). Recurrent neural network based language model. Proceedings of the 11th Annual Conference of the International Speech Communication Association, INTERSPEECH 2010. 2. 1045-1048.
- [28] Yang, Y., Krompass, and D., Tresp. V. (2017) Tensor-Train Recurrent Neural Networks for Video Classification. *arXiv*. arXiv:1707.01786
- [29] Hill, S.T., Kuintzle, R., Teegarden, A., et al. (2018) A deep recurrent neural network discovers complex biological rules to decipher RNA protein-coding potential. *Nucleic Acids Research* **46**, 16, 8105–8113.
- [30] Albawi, S., Mohammed, T.A., and Al-Zawi, S. (2017) Understanding of a convolutional neural network. 2017 International Conference on Engineering and Technology (ICET), Antalya, 2017, pp. 1-6.
- [31] Krizhevsky, A., Sutskever, I., Hinton, G.E. (2017) ImageNet classification with deep convolutional neural networks. *Communications of the ACM*. **60**, 6. June 2017 pp 84–90.

- [32] Ronneberger,O., Fischer P., and Brox, T. (2015) U-Net: Convolutional Networks for Biomedical Image Segmentation. International Conference on Medical Image Computing and Computer-Assisted Intervention MICCAI 2015: Medical Image Computing and Computer-Assisted Intervention –MICCAI 2015 pp 234-241.
- [33] Paszke, A., Gross, S., Massa, F., et al. (2019) PyTorch: An Imperative Style, High-Performance Deep Learning Library. Advances in Neural Information Processing Systems 32 (NeurIPS 2019), pp 8026-8037.
- [34] Rosentrater, K.A., and Balamuralikrishna, R. (2005) Essential Highlights of the History of Fluid Mechanics. *ASEE PEER Document Repository*. Session 2661.
- [35] Anderson Jr., J.D. (2010) Brief History of the Early Development of Theoretical and Experimental Fluid Dynamics. *Encyclopedia of Aerospace Engineering*. ISBN: 978-0-470-68665-2.
- [36] Bhanduvula, S. (2012) Finite Difference Method in Computational Fluid Dynamics. IJEAR Vol. 2, Issue 2, ISSN: 2348-0033.
- [37] Dlamini, P.G., Motsa, S.S., and Khumalo, M. (2013) Higher Order Compact Finite Difference Schemes for Unsteady Boundary Layer Flow Problems. *Nonlinear Fluid Flow and Heat Transfer* 2013, 941096.
- [38] Bertram, V. (2012) Practical Ship Hydrodynamics. Butterworth-Heinemann. ISBN 978-0-08-097150-6.
- [39] Ahmadian, A.S. (2016) Numerical Models for Submerged Breakwaters. Butterworth-Heinemann. ISBN 978-0-12-802413-3.
- [40] Mazumder, S. (2016) Numerical Methods for Partial Differential Equations. Academic Press. ISBN 978-0-12-849894-1.
- [41] Rapp, B.E. (2017) Microfluidics: Modeling, Mechanics and Mathematics. Elsevier. ISBN 978-1-4557-3141-1.
- [42] Neill, S.P., and Hashemi, M.R. (2018) Fundamentals of Ocean Renewable Energy. Academic Press. ISBN 978-0-12-810448-4.

- [43] Logan, D.L. (2011). A first course in the finite element method. Cengage Learning. ISBN 978-0495668251.
- [44] Gingold, R.A., and Monaghan, J.J. (1977) Smoothed particle hydrodynamics: theory and application to non-spherical stars. *Mon. Not. R. Astron. Soc.* **181** (3): 375–89.
- [45] Lucy, L.B. (1977). A numerical approach to the testing of the fission hypothesis. *Astron. J.* **82**: 1013–1024.
- [46] Rudy, S.H., Brunton, S.L., and Proctor, J.L., et al. (2017) Data-driven discovery of partial differential equations. *Science Advances* **3**, 4.
- [47] Raissi, M., Perdikaris, P., and Karniadakis, G.E. (2019) Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics* **378**, 686-707.
- [48] Raissi, M., Perdikaris, P., and Karniadakis, G.E. (2019) Physics Informed Deep Learning (Part I): Data-driven Solutions of Nonlinear Partial Differential Equations. *arXiv*. arXiv:1711.10561
- [49] Raissi, M., Perdikaris, P., and Karniadakis, G.E. (2019) Physics Informed Deep Learning (Part II): Data-driven Discovery of Nonlinear Partial Differential Equations. *arXiv*. arXiv:1711.10566
- [50] Lu, L., Meng, X., Mao, Z., et al. (2020) DeepXDE: A deep learning library for solving differential equations. *arXiv*. arXiv:1907.04502.
- [51] Lu, L., Jin, P., and Karniadakis, G.E. (2019) DeepONet: Learning nonlinear operators for identifying differential equations based on the universal approximation theorem of operators. *arXiv*. arXiv:1910.03193.
- [52] Cai, S., Wang, Z., Lu, L., et al. (2020) DeepM&Mnet: Inferring the electroconvection multiphysics fields based on operator approximation by neural networks. *arXiv*. arXiv:2009.12935.

- [53] Mao, Z., Lu, L., Marxen, O., et al. (2020) DeepM&Mnet for hypersonics: Predicting the coupled flow and finite-rate chemistry behind a normal shock using neural-network approximation of operators. *arXiv*. arXiv:2011.03349.
- [54] Lin, C., Li, Z., Lu, L., et al. (2020) Operator learning for predicting multiscale bubble growth dynamics. *J. Chem. Phys.* **154**, 104118.
- [55] Li, Z., Kovachki, N., Azizzadenesheli, K., et al. (2020) Fourier Neural Operator for Parametric Partial Differential Equations. *arXiv*. arXiv:2010.08895.
- [56] Jiang, C., Esmaeilzadeh, S., Azizzadenesheli, K., et al. (2020) MeshfreeFlowNet: A Physics-Constrained Deep Continuous Space-Time Super-Resolution Framework. *arXiv*. arXiv:2005.01463.
- [57] Bai, Y., and Bai, Q. (2018) Subsea Engineering Handbook. ISBN 978-0-12-812622-6.
- [58] Meyers, R.A. (2001) Encyclopedia of Physical Science and Technology. ISBN 978-0-12-227410-7.
- [59] Wang, C.Y. (2007) MODELING MULTIPHASE FLOW AND TRANSPORT IN POROUS MEDIA. *Transport Phenomena in Porous Media*, 383-410.
- [60] Dollet, B., van Hoeve, W., Raven, J.-P., et al. (2008) Role of the Channel Geometry on the Bubble Pinch-Off in Flow-Focusing Devices. *Physical Review Letters*. **100**, 034504.
- [61] Herrada, M.A., Montanero, J.M., Ferrera, C., et al. (2010) Analysis of the dripping-jetting transition in compound capillary jets. *J. Fluid Mech.* **649**, 523–536.
- [62] van Hoeve, W., Dollet, B., Versluis, M., et al. (2011) Microbubble formation and pinch-off scaling exponent in flow-focusing devices. *Physics of Fluids* **23**, 092001.
- [63] Vega, E. J., Acero, A. J., Montanero, J. M., et al. (2014) Production of microbubbles from axisymmetric flow focusing in the jetting regime for moderate Reynolds numbers. *Physical Review E* **89**, 063012.
- [64] Zhao, B., Pahlavan, A.A., Cueto-Felgueroso, L., et al. (2014) Forced Wetting Transition and Bubble Pinch-Off in a Capillary Tube. *Physical Review Letters* **120**, 084501.

- [65] Peyman, S.A., Abou-Saleh, R.H., McLaughlan, J.R., et al. (2012) Expanding 3D geometry for enhanced on-chip microbubble production and single step formation of liposome modified microbubbles. *Lab Chip* **12**, 4544–4552.
- [66] Papadopoulou, V., Tang, M.-X., Balestra, C., et al. (2014) Circulatory bubble dynamics: From physical to biological aspects. *Advances in Colloid and Interface Science* **206**, 239–249.
- [67] Miao, H., Gracewski, S.M., and Dalecki, D. (2008) Ultrasonic excitation of a bubble inside a deformable tube: Implications for ultrasonically induced hemorrhage. *J. Acoust. Soc. Am.* **124**, 2374–2384.
- [68] Hosseinkhah, N., Chen, H., Matula, T.J., et al. (2013) Mechanisms of microbubble–vessel interactions and induced stresses: A numerical study. *J. Acoust. Soc. Am.* **134**, 1875–1885.
- [69] Hosseinkhah, N., Goertz, D.E., and Hynynen, K. (2015) Microbubbles and Blood Brain Barrier Opening: A Numerical Study on Acoustic Emissions and Wall Stress Predictions. *IEEE Transactions on Biomedical Engineering*. **62**(5): 1293–1304.
- [70] Lea, J.F., Nickens, H.V., and Wells, M.R. (2008) Gas Well Deliquification. ISBN 978-0-7506-8280-0.
- [71] Talu, E., Hettiarachchi, K., Powell, R.L., et al. (2008) Maintaining Monodispersity in a Microbubble Population Formed by Flow-Focusing. *Langmuir*. **24**(5): 1745–1749.
- [72] Tenjimbayashi, M., Doi, K., and Naito, M. (2019) Microbubble flows in superwettable fluidic channels. *RSC Advances*. **9** 21220.
- [73] Chen, Y., Liu, Y., Kung, C., et al. (2019) Design and Implementation of Cloud Analytics-Assisted Smart Power Meters Considering Advanced Artificial Intelligence as Edge Analytics in Demand-Side Management for Smart Homes. *Sensors* **2047**, 19(9).
- [74] Zhai, H. Mathematical Model of Neural Network. URL: <https://hanfengzhai.net/note/NN.pdf>

- [75] Meng, X., Li, Z., Zhang, D., et al. (2020) PPINN: Parareal physics-informed neural network for time-dependent PDEs. *Computer Methods in Applied Mechanics and Engineering.* **370** 113250.
- [76] Wikipedia. Fluid Mechanics. URL: https://en.wikipedia.org/wiki/Fluid_mechanics
- [77] Gibiansky, A. Fluid Dynamics: The Navier-Stokes Equations. URL: <https://andrew.gibiansky.com/downloads/pdf/Fluid%20Dynamics:%20The%20Navier-Stokes%20Equations.pdf>
- [78] Wikipedia. Multiphase Flow. URL: https://en.wikipedia.org/wiki/Multiphase_flow
- [79] Fedosov, D.A., Peltomäkia, M., and Gomppera, G. (2014) Deformation and dynamics of red blood cells in flow through cylindrical microchannels. *Soft Matter.* **10**, 4258.
- [80] Tomaiuolo, G., Simeone, M., Martinelli, V., et al. (2009) Red blood cell deformation in microconfined flow. *Soft Matter.* **5**, 3736–3740.
- [81] Alizadehrad, D., Imai, Y., Nakaaki, K., et al. (2009) Quantification of red blood cell deformation at high-hematocrit blood flow in microvessels. *Journal of Biomechanics.* **45**(15), 2684-2689.
- [82] Matuła, K., Rivello, F., and Huck, W.T.S. (2019) Single-Cell Analysis Using Droplet Microfluidics. *Advanced Biomaterials.* **4**(1), 1900188.
- [83] Science Direct. Interstitial Pressure. URL: <https://www.sciencedirect.com/topics/engineering/interstitial-pressure>
- [84] COMSOL Multiphysics ®. The Level Set Method. URL: <https://www.comsol.com/forum/thread/attachment/37361/The-level-set-methodfrom-MEMS-Module-5198.pdf>
- [85] Wang, H., and Guo, H. (2017) P granules phase transition induced by cytoplasmic streaming in *Caenorhabditis elegans* embryo . *Sci. China-Phys. Mech. Astron.* **60**, 1.
- [86] Sutera, S.P., and Skalak, R. (1993) THE HISTORY OF POISEUILLE'S LAW. *Annu. Rev. Fluid Mech.* **25**, 1-20.

- [87] Shankar, B.M., and Shivakumara, I.S. (2020) Stability of Poiseuille flow in an anisotropic porous layer with oblique principal axes: More accurate solution. *Journal of Applied Mathematics and Mechanics.* **101**, 2.
- [88] Ng, A. Deep Learning. *Coursera*. URL: https://www.coursera.org/specializations/deep-learning?utm_source=deeplearningai&utm_medium=institutions&utm_campaign=SocialYoutubeDLSC2W1L10.
- [89] Lima, G.C. The Growth of AI and Machine Learning in Computer Science Publications. URL: <https://medium.com/@thegcamilo/the-growth-of-ai-and-machine-learning-in-computer-science-publications-603d75467c38>
- [90] Zhai, H., and Hu, G. (2021) Inferring micro-bubble dynamics with physics-informed deep learning. *arXiv preprint*. arXiv:2105.07179.
- [91] Scornet, E. Deep Learning - Optimization. URL: <https://erwanscornet.github.io/teaching/Optimization.pdf>.
- [92] Lombaert, H. Level set method: Explanation. URL: <https://profs.etsmtl.ca/hlombaert/levelset/#fig:levelset-square>.
- [93] Skajaa, A. Limited Memory BFGS for Nonsmooth Optimization. Master Thesis. New York University.
- [94] Gaspar Elsas, J.H., Koide, T., and Kodama, T. (2014) Noether Theorem of Relativistic-Electromagnetic Ideal Hydrodynamics. *Brazilian Journal of Physics*. **45** 334–339.
- [95] Darby, R. Two-Phase Gas/Liquid Pipe Flow. <https://www.aiche.org/sites/default/files/docs/webinars/DarbyR-TwoPhaseGasLiquidPipeFlow.pdf>.
- [96] Wikipedia. Limited-memory BFGS. https://en.wikipedia.org/wiki/Limited-memory_BFGS.

附录

附录 A N-S 方程的推导

第2章节中我们给出并直接使用了一般形式下的N-S方程。该方程可由 Eulerian 及 Lagrangian 坐标系下分别推导出。现我们给出基于 Lagrangian 坐标系的推导过程^[77]。

Eulerian & Lagrangian 视角

在描述流体运动时，Eulerian 坐标系（视角）是基于固定坐标系对以固定空间内的流体运动进行描述；而 Lagrangian 坐标系（视角）则是跟随但流体微团（粒子），追踪其运动轨迹^[94]。两视角下描述流体运动示意图如图 A.1 所示。

Lagrangian 与 Eulerian 速度描述可写作关系式：

$$\mathbf{u}_L = \mathbf{u}_E(t, x_p, y_p, \delta_p)$$

若考虑时间相关情形，则坐标变量均与时间存在函数关系：

$$\mathbf{u}_L = \mathbf{u}_E(t, x_p(t), y_p(t), \delta_p(t))$$

因此，基于 Lagrangian 视角的加速度可写作：

$$\mathbf{a}_L = \frac{d\mathbf{u}_L}{dt} = \frac{\partial \mathbf{u}_E}{\partial \mathbf{x}} \cdot \frac{d\mathbf{x}}{dt}$$

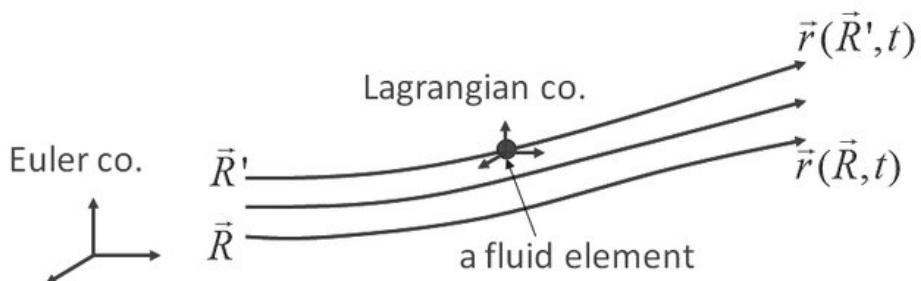


图 A.1 Eulerian 和 Lagrangian 坐标系的示意图，图源于 J. H. Gaspar Elsas et al.^[94]。

其中, \mathbf{a}_L 为 Lagrangian 视角下流体加速度; 在本文主要考虑的二维情况下, $\mathbf{u} = (u, v)$, $\mathbf{x} = (x, y)$ 。

我们因此给出物质导数 (material derivative) D 的定义:

$$\mathbf{a} = \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \cdot \mathbf{u} = \frac{D\mathbf{u}}{Dt}$$

现在, 我们给出牛顿第二定律并可展开为:

$$m\mathbf{a} = \mathbf{F} \rightarrow \rho V \mathbf{a} = \mathbf{F}$$

现将前文中给出的物质导数定义带入牛顿第二定律, 若只考虑流体所受重力情况下, 在二维情况下方向向量为 (\mathbf{i}, \mathbf{j}) , 我们得出:

$$\rho V \frac{D\mathbf{u}}{Dt} = V(-\nabla p - \rho g \mathbf{j})$$

其中 p 为流体所受压强。

我们可以进一步把上述方程化简为:

$$\rho \frac{D\mathbf{u}}{Dt} = (-\nabla p - \rho g \mathbf{j})$$

如果将物质导数的定义带入展开的话, 我们可以得到如下形式:

$$\rho \frac{\partial \mathbf{u}}{\partial t} + (\mathbf{u} \cdot \nabla) \cdot \mathbf{u} = (-\nabla p - \rho g \mathbf{j})$$

这便是动量方程的一般形式。流体力学方程组还包括能量守恒、质量守恒 (连续性条件) 等 (第 2 章已给出), 在此不过多赘述。

附录 B 多相流密度公式的推导

在第 2 章中我们简要地给出多相流体在计算中的控制方程。为更严谨的表述多相流运动, 特别是本文中涉及的气-液两相运动, 我们基于 Darby^[95] 关于多相流的课件, 给出描述涉及两相运动的基本方程和数学关系。

现对于一两相流系统, 有液-气两相 (liquid-gas, L-G), 系统的总质量为 \mathcal{M} , 质

量导数可以写作：

$$\dot{\mathcal{M}} = \dot{\mathcal{M}}_L + \dot{\mathcal{M}}_G = \rho_L Q_L + \rho_G Q_G$$

其中 Q 为体积流动率 (volume flow rate)。

我们因此可以写出多相流系统的质量通量 G (mass flux)：

$$G = \frac{\dot{\mathcal{M}}}{A} = \frac{\dot{\mathcal{M}}_L}{A} + \frac{\dot{\mathcal{M}}_G}{A} = G_L + G_G$$

基于质量通量 G , 我们可以写出多相流系统的体积通量 J (volume flux)：

$$J = J_L + J_G = \frac{G}{\rho} = \frac{G_L}{\rho_L} + \frac{G_G}{\rho_G} = \frac{Q_L + Q_G}{A}$$

现假设两相流系统中气相占比为 \mathcal{V} 。多相流系统的整体体积 $V = J$, 则气相和液相的体积可以写作：

$$V_G = \frac{J_G}{\mathcal{V}}, \quad V_L = \frac{J_L}{1 - \mathcal{V}}$$

为方便后续推导, 我们引入

$$S = \frac{V_G}{V_L}, \quad \kappa = \frac{\dot{\mathcal{M}}_G}{\dot{\mathcal{M}}_G + \dot{\mathcal{M}}_L}$$

根据 κ 和 S , 气液两相质量导数的比值可以写为：

$$\frac{\dot{\mathcal{M}}_G}{\dot{\mathcal{M}}_L} = \frac{\kappa_{LG}}{1 - \kappa_{LG}} = S \left(\frac{\rho_G}{\rho_L} \right) \left(\frac{\mathcal{V}}{1 - \mathcal{V}} \right)$$

多相流系统的密度可写作

$$\rho = \mathcal{V}\rho_G + (1 - \mathcal{V})\rho_L$$

我们因此可以将前文中引入的气相体积分数 \mathcal{V} 展开为

$$\mathcal{V} = \frac{\kappa_{LG}}{\kappa_{LG} + S(1 - \kappa_{LG})\rho_G / \rho_L}$$

致 谢

值此论文完成之际，作者首先感谢导师胡国辉教授，与胡老师多次的交流与讨论奠定了后期的选题和成果。胡老师就方向上的指导和建议推动了算法上的相应的创新与优化。胡老师在资源和经费上的大力支持才保证了本文论文计算的完成。胡老师追求严谨的学术精神更促进了我对于力学问题的思考和研究方式和思路的形成。

感谢我的父母、亲友以及家人们，他们对我的无私关怀，支持和鼓励推动了我在学术的道路上追求创新，勇攀高峰。感谢实验室师兄师姐们提出相关建议，为我答疑解惑，特别是陆钰师兄的帮助。还要感谢力学系的同学们，正是力学同学们浓厚的学术氛围，融洽的同学关系促进了我们更好地学习力学，使用力学；更帮助大家更自由、开放地选择自己的研究方向。最后，还要感谢力学系的各个老师和教职员们，他们的专心研究和辛勤奉献奠定了我们四年本科学习的理论基础，更促进了我们走进力学学科，开创属于自己的方向。

文中部分计算资源由上海索辰信息科技股份有限公司支持。文中主要计算基于北京超级云计算中心计算资源运行。文章中提出的 BubbleNet 源代码已经开源在 GitHub：<https://github.com/hanfengzhai/BubbleNet>，相关内容的技术细节和使用方法详见作者个人网站：<https://hanfengzhai.net/BubbleNet>

作者还要感谢力学、计算机科学、计算物理领域内一路以来的先驱者和大师们，正是他们的不懈努力，开创了计算力学和深度学习的今天。同时，作者也感谢开源社区（Open Source Community）提供的公开、透明的文化。若没有大量优秀代码开源在 GitHub、GitLab、Zenedo 等开源平台上并供同行使用，计算力学和机器学习领域也不会有今日辉煌的成就。前人的努力铺垫了后续学者在大厦上建立新的理论。