

**Machine learning non-local closures for turbulent  
anisotropic multiphase flows**

by

Alexis-Tzianni Charalampopoulos

Diploma of Naval Architecture & Marine Engineering, NTUA (2016)

Submitted to the Department of Mechanical Engineering  
in partial fulfillment of the requirements for the degree of

Master of Science in Mechanical Engineering

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

September 2019

© Massachusetts Institute of Technology 2019. All rights reserved.

**Signature redacted**

Author .....

Department of Mechanical Engineering  
August 15, 2019

**Signature redacted**

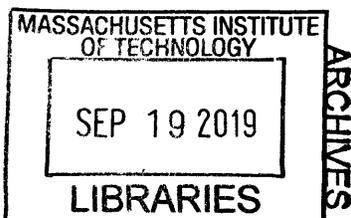
Certified by .....

~~Themistoklis P. Sapsis~~  
Doherty Associate Professor in Ocean Utilization  
Thesis Supervisor

**Signature redacted**

Accepted by .....

Nicolas Hadjiconstantinou  
Chairman, Committee on Graduate Students





# Machine learning non-local closures for turbulent anisotropic multiphase flows

by

Alexis-Tzianni Charalampopoulos

Submitted to the Department of Mechanical Engineering  
on August 15, 2019, in partial fulfillment of the  
requirements for the degree of  
Master of Science in Mechanical Engineering

## Abstract

In this thesis we formulate a data-driven, nonlocal closure scheme for turbulent multiphase fluid flows. In more detail, we use the predictions of neural nets to actively integrate in time the evolution of a turbulent anisotropic fluid flow on which bubbles that act as passive inertial tracers are being transported. The first step of our method requires the introduction of a filter on the initial 2D dynamical system that reduces it to a 1D problem. Then we model the appearing closure terms using recurrent neural networks and convolutions in space. We avoid the use of fully-connected layers due to their large computational overhead, tendency to over-fit on training data and non-physical implications. We choose to work with recurrent neural networks as recent works have shown memory effects can enhance data-driven predictions in applications in turbulence. We first test our method on unimodal jets and then proceed to bimodal profiles. Our model appears to accurately learn the statistical steady state profile of both the fluid flow velocity field as well as the profile of the distribution of bubbles in space. Finally, we use neural networks to learn the statistics of bubble deformation so that we can incorporate the effects of the bubble motion back to the fluid flow itself, on a later stage.

Thesis Supervisor: Themistoklis P. Sapsis

Title: Doherty Associate Professor in Ocean Utilization



## Acknowledgments

I would like to express my gratitude first of all, to Professor Themis Sapsis, my advisor, for his support throughout these two years. The work presented here would not be possible without his guidance and mentorship.

I would also like to thank Zhong Yi Wan, for the stimulating discussions and helpful recommendations upon the topics presented in this thesis. It was of great help to me to have a colleague in the lab that was very familiar with neural networks as I was just starting studying this topic.

I want to thank Dr. Spencer Bryngelson for the collaboration on the work presented in chapter 7. His knowledge on the topic of bubble deformation and was crucial for the project to yield interesting result within a month.

I would like to thank my family and friends for their support throughout the completion of my degree.

Finally, I would also like to thank ONR for sponsoring the MURI program "High-Fidelity Simulation Methodologies for Multi-Phase Flows", part of which is the research presented in this thesis.



# Contents

<b>1</b>	<b>Introduction</b>	<b>15</b>
1.1	Turbulence closure models . . . . .	15
1.1.1	RANS models . . . . .	16
1.1.2	LES models . . . . .	17
1.2	Machine learning in fluid dynamics . . . . .	18
1.2.1	Neural networks in turbulence closure schemes . . . . .	19
1.3	Contributions . . . . .	20
<b>2</b>	<b>Theoretical setup of multiphase flow</b>	<b>23</b>
2.1	Fluid flow setup . . . . .	23
2.1.1	Description of spectral method solver . . . . .	25
2.2	Bubble flow setup . . . . .	28
2.3	Filtering equations of motion . . . . .	30
<b>3</b>	<b>Implementation of ML closure model</b>	<b>33</b>
3.1	ML architecture . . . . .	33
3.2	Takens Theorem . . . . .	37
3.3	Training method . . . . .	37
3.4	Feature selection . . . . .	38
3.5	Imitation learning . . . . .	40
<b>4</b>	<b>Numerical Results</b>	<b>43</b>
4.1	Validation on uni-modal jets . . . . .	43

4.2	Testing generalizability on bi-modal jets . . . . .	47
4.3	Limits of predictability . . . . .	50
<b>5</b>	<b>Machine learning statistics of the deformation of cavitating bubbles</b>	<b>51</b>
5.1	Problem Statement . . . . .	53
5.2	Machine Learning approach . . . . .	55
5.2.1	Evolution of low-order moments . . . . .	55
5.2.2	Evolution of high-order moments . . . . .	59
<b>6</b>	<b>Conclusions and Future work</b>	<b>63</b>
6.1	Conclusions . . . . .	63
6.2	Future work . . . . .	64

# List of Figures

3-1	Schematic illustration of the spatio-temporal area around point from which we take information for our predictions. . . . .	33
3-2	Architecture of the neural net we use for predicting closure term $\partial_y(\overline{u'_1 u'_2})$ . 36	
3-3	Architecture of the neural net we use for predicting closure term $\partial_y(\overline{v'_2 \rho'})$ . 36	
3-4	Mean square training-error (solid line) and validation error (dashed line) for $\mathbb{D}_2$ . . . . .	39
4-1	Time-averaged profile of $\bar{u}_1$ for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line. . . . .	44
4-2	Time-averaged profile of $\bar{\rho}$ for the ML closure model (blue line) and the DNS (black line). Initial bubble distribution is depicted with dashed line. . . . .	45
4-3	Forward FTLE simulations for different resolutions, for unimodal validation case for $t - t_0 = 10$ . Yellow corresponds to repelling areas. . . . .	46
4-4	Illustration of the main grid (black dots) and the auxiliary grid (blue dots) used for the calculation of the Cauchy-Green strain tensor. . . . .	47
4-5	Time-averaged profile of $\bar{u}_1$ for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line. . . . .	47

4-6	Time-averaged profile of $\bar{u}_1$ for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line. . . . .	48
4-7	Time-averaged profile of $\bar{p}$ for the ML closure model (blue line), LES model (red line) and the DNS (black line). Initial bubble distribution is depicted with dashed line. . . . .	49
4-8	Time-averaged profile of $\bar{u}_1$ for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line. . . . .	50
5-1	Time-series of moment $\mu_{10}$ as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).	56
5-2	Time-series of moment $\mu_{01}$ as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).	57
5-3	Time-series of moment $\mu_{20}$ as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).	57
5-4	Time-series of moment $\mu_{02}$ as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).	58
5-5	Time-series of moment $\mu_{11}$ as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).	58
5-6	Time-history of $\mu_{30}$ moment, as computed by Monte-Carlo data (blue), non-Gaussian closure (orange) and ML predictions (green). . . . .	60
5-7	Time-history of $\mu_{12}$ moment, as computed by Monte-Carlo data (blue), non-Gaussian closure (orange) and ML predictions (green). . . . .	61

5-8	Time-history of $\mu_{32}$ moment, as computed by Monte-Carlo data (blue), non-Gaussian closure (orange) and ML predictions (green). . . . .	61
-----	----------------------------------------------------------------------------------------------------------------------------------------------	----



# List of Tables

3.1	Feature selection for closure of Navier-Stokes . . . . .	38
3.2	Feature selection for closure of bubble transport equation . . . . .	39



# Chapter 1

## Introduction

Recent advancements in the field of machine learning has inspired a great interest in utilizing neural networks to enhance predictions of numerical solvers for turbulent flows. The work presented here is motivated by this trend. In particular, we model turbulent multiphase flows using recurrent neural networks. While a lot of data-based approaches have been formulated for turbulent fluid flows little work has been done on the topic of multiphase flows. In this thesis we tackle exactly this problem using nonlocal representations for the closure terms both in time and space. Before we begin discussing our method, we present a short overview on the topic of turbulence closure models. Furthermore, we discuss the history of machine learning and turbulent fluid flows while also listing recent notable works in this rapidly expanding field.

### 1.1 Turbulence closure models

Solving turbulent flows numerically, requires the resolution of a very broad number of spatio-temporal scales. As a result, the number of scales that are relevant for fully resolving many physically important problems is enormous. Our current computational capabilities do not allow for solving such complex problems. Hence, scientists resort to modifying the original problem by means of turbulence closure schemes for unresolved, but physically relevant, quantities.

Since we are not able to fully resolve the original physical system, we have to

limit ourselves to resolving only the spatiotemporal scales we can afford to. This approximation, from a theoretical point of view, is effectively equivalent to applying a filter  $\overline{(\cdot)}$  to the original equations of motion. Usually, this filter can be thought of as some kind of low-pass filtering or averaging. Hence, we move from the original flow variables  $\xi$  to their filtered version  $\bar{\xi}$ . Therefore, if the operator characterising the original equations of motion is  $\mathbb{L}(\xi)$ , then we work with its filtered version  $\overline{\mathbb{L}(\xi)}$ . Ideally we would have  $\overline{\mathbb{L}(\xi)} = \mathbb{L}(\bar{\xi})$ . Yet, such an assumption can only hold true for linear operators. Hence, for nonlinear operators we have the relationship between the filtered version of the original operator and the original operator acting on the filtered variables to be

$$\overline{\mathbb{L}(\xi)} = \mathbb{L}(\bar{\xi}) + \mathbb{D}(\bar{\xi}; \bar{c}), \quad (1.1)$$

where  $\mathbb{D}$  corresponds to additional terms arising from applying the filter. For example, in the case of incompressible flow of a Newtonian fluid, the additional terms are simply the Reynolds stresses  $\nabla \cdot \boldsymbol{\tau}$ , where  $\tau_{i,j} = \overline{u_i u_j}$  and  $u_i$  are the velocity components of the fluid flow.

However, the real problem from this representation arises by the terms  $\bar{c}$  which are new flow variables that need to be found in order to solve the system. These terms are usually called closure terms. Hence, we end up with an under-determined system, unless we introduce modelling assumptions for the closure terms  $\bar{c}$ . The field of turbulence closure models deals with exactly that problem. The main two families of closure models that are widely used in fluid mechanics are the so-called Reynolds-averaged Navier-Stokes (RANS) family of models [43] and the so-called large eddy simulations (LES) family of models [40], [10]. We present a brief overview for both of these families of closure models.

### 1.1.1 RANS models

This family of models stems from applying an averaging filter to the exact equations of motion. RANS models represent turbulent terms completely through modelling

assumptions (i.e. the entirety of operator  $\mathbb{D}$  in Eq. (1.1)). The general form of such an averaging filter can be written as

$$\bar{\xi} = \int_{\Omega} G(\mathbf{x}', t') \xi(\mathbf{x} - \mathbf{x}', t - t') d\mathbf{x}' dt, \quad (1.2)$$

where  $G$  is a spatio-temporal kernel and  $\Omega$  is the spatio-temporal domain where our physical problem lies.

As a result, they require substantially lower computational resources for simulations, compared to a direct numerical simulation of the full problem (DNS). Out of this family of models, the subcategory called single-point RANS is the most widely used technique. In such models a locally defined constitutive equation is established between the closure terms and the filtered (averaged) flow variables. Hence, the unknown closure terms are expressed locally in terms of the averaged flow variables rendering the system determined and thus solvable. If we go one step further and assume that the closure terms are a local function of only the filter velocity gradient tensor, Cayley-Hamilton theorem is applicable and an exact expansion can be derived for the closure terms [13]. Linear eddy viscosity models and algebraic stress models are examples of such closure schemes.

### 1.1.2 LES models

LES models introduce a filtering operator in the equations the same way RANS models do, with the exception that in most cases contrary to RANS models averaging in time is not enforced. This means that in many LES models we have a filter of the form

$$\bar{\xi} = \int_{\Omega'} G(\mathbf{x}') \xi(\mathbf{x} - \mathbf{x}') d\mathbf{x}', \quad (1.3)$$

where  $G$  is a spatial kernel and  $\Omega'$  is the spatial extent of the domain where our physical problem lies.

The main difference between the LES family of methods and RANS, is that in LES schemes we fully resolve the parts of the closure terms that correspond to the resolved

spatio-temporal scales. For example, for the case of incompressible Newtonian fluid, we expand the Reynolds stresses as  $\tau_{ij} = \overline{u_i u_j} = \overline{u_i} \overline{u_j} + \tau'_{ij}$ . We then fully resolve the stresses corresponding to  $\overline{u_i} \overline{u_j}$  and use a constitutive equation for the stresses  $\tau'_{ij}$  which correspond to unresolved scales. For example, for the case of a simple linear eddy viscosity model, we model the residual Reynolds stresses as

$$\tau'_{ij} = \nu_{sgs} \left( \frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i} \right), \quad (1.4)$$

where  $\nu_{sgs}$  is a free parameter that needs to be tuned using data from the flow we try to approximate.

## 1.2 Machine learning in fluid dynamics

Machine learning (ML) and turbulence go a long way back. The problem of turbulence was coined as a possible problem for statistical learning theory in the 1940's by Kolmogorov. However, for the next 30 years very few innovations will be accomplished in the field of turbulence due to data-driven approaches. The lack of results will lead to the 1973 Lighthill report and its subsequent debate (which is free on youtube for everyone to watch). In this debate Lighthill attacks the ML community for making high promises it has failed to accomplish. The main gripe of Lighthill was that due to combinatorial increase in the complexity of the state space of physical problems (as we introduce more degrees of freedom) ML will never be able to be used outside of toy examples. This will lead to the so called AI winter of the 70's and early 80's. This limited interest in ML will stop with the introduction of the back-propagating algorithm, [34], [35], that allows for training of deep neural networks.

The first application of deep neural networks to turbulent flows appears to be the work of Milano & Koumoutsakos [32]. They utilized nonlinear autoencoders to develop a nonlinear embedding for a turbulent flow near a wall. Their work showed that they were able to learn the flow just by using information on the wall only. Their approach can be thought of as a nonlinear version of the widely used in

turbulence POD decomposition [25],[26], [39]. Their use of neural networks for Principal Components Analysis (PCA) [2] improved the predictions possible by traditional POD.

Recurrent neural networks and in particular Long-Short-Term memory (LSTM) layers [19] have been utilized for the purpose of enhancing the predictive capabilities of reduced-order models in fluid dynamics. Their incorporation of time delays to such systems has showcased superior prediction in both kinematics of bubbles in turbulent flows [47] and prediction of extreme events [46].

Furthermore, researchers started using generative adversarial networks (GANs), introduced by [15], to extract physically important flow features [49].

### 1.2.1 Neural networks in turbulence closure schemes

Attempts to augment the predictive capabilities of RANS closure models have also been of great interest [23], [24]. Similar results have been published for LES closure models [12], [20], [29].

Notable works include the introduction of the so-called "field inversion and machine learning" framework by [33]. This framework was later utilized by [38] to formulate a neural-net based enhancement of RANS predictions for flows around airfoils.

The authors in [24] and [23], showcase a simple, yet effective way to embed Galilean invariance to the neural net predictions for an anisotropic Reynolds stress tensor. Their innovation allows for the neural nets architecture to respect physical symmetries and invariances of the underlying dynamical system.

Neural nets have also been recently used in geophysical fluid dynamics (GFD). Specifically, [5] used convolutional neural nets to predict unresolved Reynolds stresses from degraded high-resolution data. The high-resolution data came from a quasi-geostrophic model for oceans and even when subsampling the data up to 10% the model was able to accurately predict the turbulent Reynolds stresses.

Finally, the authors of [27] and [28] utilized neural networks and RANS models to model the interphase mass and momentum fluxes in turbulent multiphase flows.

## 1.3 Contributions

Before we begin presenting our approach, we will list the original contributions of this thesis. The main contribution of this thesis is that it describes a data-driven methodology for turbulent closure of multiphase fluid flows, which allows for the incorporation of spatio-temporal nonlocality. To the best of our knowledge, recurrent neural networks have not been used for enhancing the predictive capabilities of turbulent closure schemes. As spatial nonlocality has been shown to affect favourably prediction of Reynolds stresses [5], our inclusion of temporal nonlocality, through LSTM layers [19], appears to further increase the effectiveness of data-driven approaches.

We use this model to formulate a data-driven closure for the problem of a turbulent and anisotropic flow of bubbles, that act as passive inertial tracers in an incompressible Newtonian fluid. We study this problem from an Eulerian point of view. For the velocity field of the bubble flow we utilized the slow manifold approximation [18] of the Maxey-Riley equation [30]. The predictive abilities of this model allow us to actively couple it with the filtered evolution equations of the multiphase system in discussion. To the best of our knowledge, it is the first time where neural networks were utilized for creating a functional closure scheme for a multiphase flow where bubbles are assumed to have inertia (and thus do not follow fluid elements). Our novel approach yields accurate predictions with regards to the statistical steady state of the turbulent and anisotropic flow reaches (statistics of both the fluid and the bubbles) for both unimodal and bimodal jet-flows.

Finally, with the help of Dr. Spencer Bryngelson from the group of Prof. Tim Colonius at Caltech, we are showcasing a novel data-driven approach to model effects due to deformation of bubbles back to the fluid flow. In more detail, instead of calculating the deformation of each bubble independently, we compute the statistics of the kinematics of the population of bubbles. This approach is drastically less time-consuming compared to evolving the deformation of each bubble in the flow. Furthermore, it compares favourably with a Gaussian closure scheme for this system. Adding non-Gaussian statistics to the model allows us to capture physical effects not

observed in the predictions of Gaussian statistics, such the decay (in time) of the motion of bubbles as they pulsate.



# Chapter 2

## Theoretical setup of multiphase flow

The goal of this chapter (and the major goal of this thesis) is to derive a data-informed nonlocal closure scheme for multiphase anisotropic turbulent flows. In more detail, the system we work with is that of an incompressible Newtonian fluid with bubbles being transported inside it. The data-informed part of the closure stems from the fact that we use neural nets to approximate the appearing closure terms in the system of equations we derive below. These neural nets make pointwise predictions in the Eulerian frame work, yet they incorporate information from a neighbourhood of each point. Hence, it is apparent that our approach is nonlocal in space. Such a nonlocality in space can be easily justified for such a problem (for its closure terms), for a example by looking at the nonlocal incompressibility constraint for the fluid flow. In addition, our approach is nonlocal in time as well, since we implement our method using recurrent neural nets. Hence, we not only use information at the current time-instant but we utilize a certain portion of the (Eulerian) time-history of the input data. Such an approach can be justified by making use of Takens theorem [42].

### 2.1 Fluid flow setup

In this section we focus on the physical assumptions driving the fluid flow of the problem. Our main goal is to simulate the transport of bubbles in a turbulent fluid flow. For the bubbles, we assume that they behave as passive inertial tracers. This

means that their velocity field is different from that of the underlying fluid flow (due to inertia effects) but their motion does not affect in any way the underlying fluid flow. For the fluid flow itself we can then assume that the incompressible Navier-Stokes equations describe its dynamics. In its dimensionless form this system of equations can be written as

$$\frac{D\mathbf{u}}{Dt} = \nabla p + \frac{1}{Re}\Delta\mathbf{u} + \mathbf{F}, \quad (2.1)$$

$$\nabla \cdot \mathbf{u} = 0, \quad (2.2)$$

where  $u$  is the velocity field of the fluid,  $p$  its pressure,  $Re$  is the Reynolds number of the flow,  $\frac{D}{Dt}$  is the material derivative operator and  $F$  denotes an external forcing term (we describe below its purpose). In this thesis we focus our attention to flows that are assumed to be 2D and doubly periodic in space. The domain is also assumed to be rectangular for simplicity. However, the method we describe below can work equally well on 3D flows in space. For space we use the notation  $\mathbf{x} = (x, y)$  and for the velocity field  $\mathbf{u} = (u_1, u_2)$  with  $u_1$  corresponding to the velocity component along the  $x$ -direction and  $u_2$  to the velocity component along the  $y$ -direction. For initial conditions, since we desire anisotropy in our flow, we use Gaussian jet structures of the general form

$$u_{1,jet} = \sum_i A_i \exp \left[ -c_i(y - \beta_i)^2 \right], \quad u_{2,jet} = 0, \quad (2.3)$$

where  $A_i, c_i, \beta_i$  are parameters. With regards to the external forcing term  $\mathbf{F}$  its purpose is twofold. First, since we want an anisotropic flow we need a large-scale external forcing term to maintain the jet structure. For maintaining the jet, we assume a forcing term that cancels the dissipation term due to the jet profile. Second, we add a small-scale and small-amplitude forcing term to stir the flow and instigate instabilities to the flow (Rayleigh instability) so that we enter a turbulent regime. To achieve turbulence we choose a forcing term that acts only on some wave numbers

with  $6 \leq \|\mathbf{k}\| \leq 7$ . Stirring a flow with a forcing localized only at a particular wavenumber interval is a practice widely utilized to achieve turbulence [7], [4], [31], [6]. Hence, we assume a form  $\mathbf{F} = -\frac{1}{R}\Delta\mathbf{u}_{jet} + f$ , with  $f$  being

$$f(\mathbf{x}, t) = \sum_i A_i(t) \cos(\mathbf{k} \cdot \mathbf{x} + \omega_i), \quad (2.4)$$

where  $6 \leq \|\mathbf{k}\| \leq 7$ ,  $A_i(t)$  are vectors sampled by white Gaussian noise (each one independently from the other) and  $\omega_i$  are phases sampled from a uniform distribution over  $[0, 2\pi]$ . The standard deviation for these amplitudes is set to 0.03. This ensures that the energy and enstrophy inputs are localized in Fourier space and only a limited range of scales around the forcing is affected by the details of the forcing statistics. Furthermore, such a forcing ensures that the system reach a jet-like statistical steady state after a transient phase. Due to the small-scale forcing being essentially homogeneous in space we can deduce that the statistical steady state profile be essentially only dependent on  $y$  (since our large-scale forcing and initial conditions depend only on  $y$ ). This fact hints that the statistics of such a flow can be approximated by a 1-dimensional approximation.

We solve this flow using spectral method and  $256^2$  modes. These results act later on as our reference solutions.

### 2.1.1 Description of spectral method solver

In this subsection we discuss in more detail the numerical implementation of a spectral method solver for the incompressible Navier-Stokes equation for this problem. Our implementation draws from the guide provided by [44] and the Matlab implementation of [11]. As we stated, we want to solve the 2D periodic Navier-Stokes equations in the presence of the background jet. Hence, we not only impose a jet as the initial condition for the problem, but introduce large-scale forcing to preserve the anisotropy for the duration of the simulation. The large-scale forcing term is such that in the absence of any other external forcing this jet structure is preserved. Hence, we expect that at each time-instant the deviation of the velocity field from the initial condition

to be an order-lower than the amplitude of the initial jet.

Due to the nature of the forcing, we expect that the flow reaches a statistical steady state, with its steady state being a jet-like formation (slightly deviating from the initial condition). For that reason, we write the velocity field as  $\mathbf{u} = \bar{\mathbf{u}} + \mathbf{u}'$ , where  $\bar{\mathbf{u}} = (\bar{u}_1(x_2), 0)$  is the background, a priori known, velocity field of the jet. The term  $\mathbf{u}'$  denotes the interactions between the background jet, the forcing and dissipation. Since, the jet acts only in the direction of the  $x$  axis and varies only along the  $y$  axis, we can easily see that the pressure generated by the undisturbed jet is  $\bar{p} = \text{const}$ , and thus can be ignored in the computations. Hence, we have  $p = p'$ . Replacing these representations in the classical momentum equations of the Navier-Stokes equation, we get

$$\begin{aligned} \partial_t(\bar{u}_1 + u'_1) &= -(\bar{u}_1 + u'_1)\partial_x(\bar{u}_1 + u'_1) - (\bar{u}_2 + u'_2)\partial_y(\bar{u}_1 + u'_1) \\ &\quad - \frac{1}{Re} \left[ \partial_x^2(u'_1 + \bar{u}_1) + \partial_y^2(u'_1 + \bar{u}_1) \right] - \partial_x p + F_1 \end{aligned} \quad (2.5)$$

$$\begin{aligned} \partial_t(\bar{u}_2 + u'_2) &= -(\bar{u}_1 + u'_1)\partial_x(\bar{u}_2 + u'_2) - (\bar{u}_2 + u'_2)\partial_y(\bar{u}_2 + u'_2) \\ &\quad - \frac{1}{Re} \left[ \partial_x^2(u'_2 + \bar{u}_2) + \partial_y^2(u'_2 + \bar{u}_2) \right] - \partial_y p + F_2. \end{aligned} \quad (2.6)$$

By rearranging the terms in the above equations and by remembering that  $\bar{u}$  is only a function of  $y$ , we get (after some algebraic manipulations)

$$-(\bar{u}_1 + u'_1)\partial_x u'_1 - u'_2\partial_y(\bar{u}_1 + u'_1) = u'_2\omega - \frac{1}{2}\partial_x[(u'_1 + \bar{u}_1)^2 + (u'_2)^2], \quad (2.7)$$

$$-(\bar{u}_1 + u'_1)\partial_x u'_2 - u'_2\partial_y u'_2 = -u'_1\omega - \frac{1}{2}\partial_y[(u'_1 + \bar{u}_1)^2 + (u'_2)^2], \quad (2.8)$$

where  $\omega$  is the vorticity of the velocity field. Note that in the 2D case vorticity is essentially a scalar field and not a vector. Taking the Fourier transform of this evolutionary equations, and neglecting the pressure term we have

$$F[\partial_t u'_1] = F[u'_2 \omega] - \frac{1}{2} i k_1 F[(u'_1 + \bar{u}_1)^2 + (u'_2)^2] - \frac{1}{R} (k_1^2 + k_2^2) F[u'_1 + \bar{u}_1] \quad (2.9)$$

$$F[\partial_t u'_2] = -F[u'_1 \omega] - \frac{1}{2} i k_2 F[(u'_1 + \bar{u}_1)^2 + (u'_2)^2] - \frac{1}{R} (k_1^2 + k_2^2) F[u'_2]. \quad (2.10)$$

We then need to project this solution into the divergence-free subspace of our space of functions. We note that

$$\int_{\Omega} \nabla \cdot (\mathbf{u}p) d\mathbf{x} = \int_{\partial\Omega} (\mathbf{u}p) \cdot \hat{n} ds = 0, \quad (2.11)$$

where the boundary terms vanish due to periodicity. Therefore we get

$$\int_{\Omega} \mathbf{u} \cdot \nabla p d\mathbf{x} = - \int_{\Omega} p \nabla \cdot \mathbf{u} d\mathbf{x} = 0, \quad (2.12)$$

due to incompressibility of the flow. It is clear then that,  $\nabla p$  is orthogonal (in the  $L^2$ -sense) on the space of divergence free vector fields  $U = \{\mathbf{u} \in L^2 : \nabla \cdot \mathbf{u} = 0\}$ . Using the Fourier space ( $\mathbf{k}$ -space) to rewrite this vector field, we have  $\hat{u}_k \in U_k = \{\hat{u}_k \in \mathbb{C}^2 : k \cdot \hat{u}_k = 0\}$ . Hence, the representation of  $F_k[\nabla p] = i \hat{p}_k k$  in  $k$ -space (with  $\hat{p}_k \in \mathbb{C}$ ) is orthogonal to  $U_k$ . Therefore, as presented in [44], if we solve only the dynamic equations of the system for a certain time-instant, obtaining a compressible velocity field  $\mathbf{f}$ , we can map the compressible flow field  $\mathbf{f}$  we calculated, to the incompressible flow field  $\mathbf{u}$  through  $\nabla p$ . Thus, we consider  $\mathbf{u} + \nabla p = \mathbf{f}$ . Assuming  $2\pi$ -periodic boundary conditions and the representations

$$u_j(x) = \sum_{k \in \mathbb{Z}^2} \hat{u}_{j,k} e^{ik \cdot x}, \quad p(x) = \sum_{k \in \mathbb{Z}^2} \hat{p}_k e^{ik \cdot x}, \quad f_j(x) = \sum_{k \in \mathbb{Z}^2} \hat{f}_{j,k} e^{ik \cdot x}. \quad (2.13)$$

Then, using the incompressibility condition and the connection between  $\mathbf{f}$  and  $\mathbf{u}$ , we project our solution to the divergence-free subspace by using the transformation (for the cases  $|k| \neq 0$ ):

$$\begin{bmatrix} \hat{u}_{1,k} \\ \hat{u}_{2,k} \\ \hat{p}_k \end{bmatrix} = \frac{1}{k_1^2 + k_2^2} \begin{bmatrix} k_2^2 & -k_1 k_2 \\ -k_1 k_2 & k_1^2 \\ -ik_1 & -ik_2 \end{bmatrix} \begin{bmatrix} \hat{f}_{1,k} \\ \hat{f}_{2,k} \end{bmatrix} \quad (2.14)$$

For the case  $k_1 = k_2 = 0$  we require that the mean flow forces vanish, i.e.  $\hat{u}_{1,0} = \hat{u}_{2,0} = 0$ . This conditions is physically valid in our case, since we assume that the fluctuations about the statistical steady state are zero-mean.

## 2.2 Bubble flow setup

We now move to the modeling of the dynamics of the bubbles. The bubbles that are transported by the fluid flow have inertia. Hence, their trajectories differ from these of fluid particles. Therefore, the need for a model that includes inertial effects arises (the bubbles cannot be model to be ideal tracers).

We assume that the bubbles behave as small rigid spheres. Since no deformation is allowed for the bubbles, it is enough to study only the motion of the center of mass of each bubble. We also note that possible rigid body rotations are neglected as they have a much less significant effect to their linear motion compared to the velocity of their respective center of mass. Under the aforementioned assumptions, and assuming that the Stokes number corresponding to the relative motion of the bubble and the fluid, Maxey & Riley derived the following model [30] (from a Lagrangian point of view), known as the Maxey-Riley equation

$$\begin{aligned}
\rho_p \frac{d\mathbf{v}}{dt} = & \rho_f \frac{D\mathbf{u}}{Dt} \\
& - \frac{9\nu\rho_f}{2a^2} \left( \mathbf{v} - \mathbf{u} - \frac{a^2}{6} \Delta\mathbf{u} \right) \\
& - \frac{9\rho_f}{2a} \sqrt{\frac{\nu}{\pi}} \int_0^t \frac{1}{\sqrt{t-\zeta}} \frac{d}{d\zeta} \left( \mathbf{v} - \mathbf{u} - \frac{a^2}{6} \Delta\mathbf{u} \right) d\zeta \\
& - \frac{\rho_f}{2} \left[ \frac{d\mathbf{v}}{dt} - \frac{D}{Dt} \left( \mathbf{u} + \frac{a^2}{10} \Delta\mathbf{u} \right) \right],
\end{aligned} \tag{2.15}$$

where  $\mathbf{v} = (v_1, v_2)$ ,  $\rho_p$  and  $\rho_f$  are the density of the bubble and of the fluid respectively,  $a$  is the radius of the bubble and  $\nu$  the kinematic viscosity of the fluid. Looking at the right-hand-side of the the equation, we see that the first line corresponds to the force by the undisturbed flow, the second line corresponds to the Stokes' force, the third line is the so-called Basset-Boussinesq memory term that models the effect of the wake of the bubble to its motion and the last line corresponds to the added mass term. Terms of the form  $\Delta\mathbf{u}$  correspond to the Faxen corrections.

While the Maxey-Riley equation is a widely used model for the motion of bubbles and particles, it introduces a difficulty when we want to study the problem from an Eulerian point of view. That is, we do not have direct access to an Eulerian velocity field of the bubbles. To derive such a field we have to run numerous Lagrangian simulations of bubbles with different initial conditions and then interpolate from these Lagrangian results an Eulerian velocity field. For this reason, we choose to use an alternative approximation of the Maxey-Riley for the velocity field of the bubbles. Specifically, under the assumption that the inertia of the bubble goes to zero, we can derive an asymptotic approximation of the Maxey-Riley equation, that is known as the slow manifold kinematic model [18]. In more detail, Haller & Sapsis showed that the velocity for the bubbles is given by

$$\mathbf{v} = \mathbf{u} + \epsilon \left( \frac{3R}{2} - 1 \right) \frac{D\mathbf{u}}{Dt} + O(\epsilon^2), \quad \epsilon = \frac{St}{R} \ll 1, \quad R = \frac{2\rho_f}{\rho_f + 2\rho_p}, \tag{2.16}$$

where  $\epsilon = \frac{St}{R} \ll 1$  is a parameter representing the importance of inertial effects and  $R = \frac{2\rho_f}{\rho_f + 2\rho_p}$ . This approximation can be derived from the Maxey-Riley equation, using singular asymptotic expansion with respect to  $\epsilon \rightarrow 0$ , after neglecting the Faxen corrections and the Basset-Boussinesq memory term. For small values of  $\epsilon$  this model produces results almost identical to the Maxey-Riley equation and thus has recently found ground for applications [36], [47].

Since now we have an approximation for the Eulerian velocity field of the bubbles at each time-instant, we can formulate an Eulerian transport equation for them. In more detail, by introducing  $\rho$  as the distribution of bubbles at a particular point (number of bubbles per area) we can write the following transport equation

$$\partial_t \rho + \nabla \cdot (\mathbf{v}\rho) = \nu \Delta^n \rho. \quad (2.17)$$

The right-hand-side of the transport equation represents a hyperviscosity term that we add to allow us to simulate this problem without the need for too high resolution. Hence, we are essentially dealing with the problem of bubble transportation as one strictly driven by advection only.

We solve Eq. (2.17) using spectral method on a 2D doubly periodic rectangular domain. We use  $256^2$  modes and use these results as reference solutions to compare our approximations to the closure terms. For the simulations we show, we use  $\epsilon = 0.05$  and  $R = 2$ .

## 2.3 Filtering equations of motion

As we stated previously, our goal is to derive accurate closure schemes for the problem comprised of the Eqs. (2.1), (2.16) and (2.17). We should note, that since we are using spectral method for our solution, the incompressibility condition for the fluid flow is not explicitly solved and thus we do not present it in our filtered equations. Hence, the unknowns of our reference system are  $\mathbf{u}$  and  $\rho$ . For the reference solutions, we first solve Eq. (2.1) with a spatial resolution of  $256 \times 256$  to compute  $\mathbf{u}$ . Then we use Eq. 2.16 to compute the bubble velocity field  $\mathbf{v}$  without the need to solve some

ordinary or partial differential equation. Finally, we proceed to solve Eq. (2.17) again with a spatial resolution of  $256 \times 256$  and compute  $\rho$  at the next time-step.

We now introduce a filter to the aforementioned dynamical system, to derive its averaged form. For filtering we use averaging along the  $x$ -direction and a low-pass filter along the  $y$ -direction. We therefore completely remove any  $x$ -dependency from our system rendering it effectively a one dimensional problem in space. The decision to do such a drastic reduction of the system stems from the fact that the statistical steady state of such a problem should be essentially only  $y$ -dependent. We then proceed to decompose our variables into  $\bar{\mathbf{u}}, \bar{\mathbf{v}}, \bar{\rho}$  (large-scale components) and  $\mathbf{u}', \mathbf{v}', \rho'$  (small-scale perturbations). Note that applying the filter to the small-scale perturbations yield zero. Since the fluid flow is almost homogeneous in the  $x$ -direction, we can expect that  $\bar{u}_2 \approx 0$ . Hence, we can neglect  $\bar{u}_2$  from our system. Notice, however, that even though  $\bar{v}_2$  is expected to be smaller than  $\bar{v}_1$ , we cannot neglect it completely from our study. Keeping this in mind we can derive the following filter equations of motion

$$\partial_t \bar{u}_1 + \partial_y(\overline{u'_1 u'_2}) = \frac{1}{Re} \partial_y^2 \bar{u}_1 + \bar{F} - \partial_y(\overline{u'_1 u'_2}), \quad (2.18)$$

$$\partial_t \bar{\rho} = \nu \partial_y^4 \bar{\rho} - \partial_y(\bar{v}_2 \bar{\rho} - \overline{v'_2 \rho'}). \quad (2.19)$$

Note that we have completely neglected the Navier-Stokes component along the  $x$ -direction since we assume  $\bar{u}_2 = 0$ . This assumption can be further justified by noting that if initial we have  $\bar{u}_2 = 0$  we get the evolution equation for  $\bar{u}_2$

$$\partial_t \bar{u}_2 = -\partial_y(\overline{u'_2 u'_2}). \quad (2.20)$$

Assuming that  $u'_2 \ll \bar{u}_1$  at all times, we can assume that  $\bar{u}_2$  even though non-zero remains order of magnitudes smaller than  $\bar{u}_1$ . To make clearer our spatial reduction, we have transformed our initial system from a spatial resolution of  $256 \times 256$  to a  $1 \times 80$ . With such a dramatic decrease in resolution we obviously do not attempt to predict every time-step of the reference system but be able to predict its statistical steady state.

Finally, we see that our system reduction although allows to solve a much cheaper problem, it introduces additional unknowns, rendering the system underdetermined. While we still have the unknowns  $\bar{u}_1$  and  $\bar{\rho}$  (remember that  $\bar{v}_1$  and  $\bar{v}_2$  can be directly computed through the filtered version of Eq. (2.16)) we have the additional closure terms  $\partial_y(\overline{u'_1 u'_2})$  and  $\partial_y(\overline{v'_2 \rho'})$ . Hence, to solve our system we need to introduce some modelling assumptions for the newly added closure terms. This complicating of course is to be expected in any kind of closure scheme where the operator of the equations of motion is not linear. The modelling of these terms is the topic of the next chapter.

# Chapter 3

## Implementation of ML closure model

### 3.1 ML architecture

In the previous chapter, we explained the need to derive some models for the unknown closure terms  $\partial_y(\overline{u'_1 u'_2})$  of Eq. (2.18) and  $\partial_y(\overline{v'_2 \rho'})$  of Eq. (2.19). In this chapter, we describe our methodology of using recurrent neural networks for the approximation of these two terms as well as their implementation. In more detail, we utilize LSTM recurrent neural networks [19]. The specific RNN implementation was picked based on its tested ability to incorporate long-term memory effects of hundreds of time-delays, while simpler RNN models suffer from vanishing or exploding gradients [3].

However, apart from the way we choose the temporal nonlocality we have to choose how we introduce the spatial nonlocality as well.

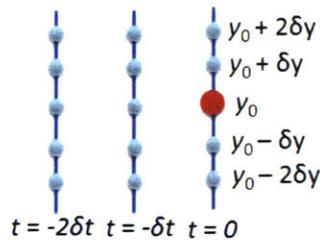


Figure 3-1: Schematic illustration of the spatio-temporal area around point from which we take information for our predictions.

That is, given a point in space  $y_0$ , we only want to use information from points that lie in a small neighbourhood of  $y_0$ . By information we meet flow features that are used as input for the neural nets. However, the optimal size of this neighbourhood must be determined heuristically. We strive to restrict spatial nonlocality to a determined neighborhood around each point, because incorporating information from the entirety of the domain is not only computationally unfeasible but also physically counter-intuitive as it means that a small perturbation far away from  $y_0$  causes a significant disturbance on it immediately. For this reason we are using convolutions in space to make sure that we incorporate information only from a region around each point and not from the entire domain. Hence, we choose to utilize a Stacked LSTM architecture [16]. It uses multiple LSTM layers with the detail that all input and recurrent transformations are convolutional. As a result, the closure terms are modelled in the following form

$$\begin{aligned}\partial_y(\overline{u'_1 u'_2}) &= \mathbb{D}_1[\theta_1; \{\bar{\xi}(y - i\delta y, t - j\delta t)\}_{i=-d, j=-k}^{i=d, j=0}] \\ \partial_y(\overline{v'_2 \rho'}) &= \mathbb{D}_2[\theta_2; \{\bar{\zeta}(y - i\delta y, t - j\delta t)\}_{i=-d, j=-k}^{i=d, j=0}],\end{aligned}\tag{3.1}$$

where  $\xi$  and  $\zeta$  are (filtered) flow features to be selected.

To decide on the specific hyperparameters that we choose for each model, bayesian optimization packages like Spearmint [41] can be used. However, in this work the optimization of the hyperparameters was carried out through trial and error. For the closure term  $\mathbb{D}_1$ , corresponding to the fluid flow dynamics, 3 convolutional LSTM layers were utilized while for the closure term  $\mathbb{D}_2$ , corresponding to the bubble flow dynamics, 4 convolutional LSTM layers were selected. Drawings of the aforementioned architectures can be found in the end of this section. A spatial window of 9 points (out of the 80 that the entire domain is comprised of) was selected for both models. This number was selected after seeing that further increase of the window did not reduce the training error any more.

Note however, that the number of points in space that have to be considered is dependent on the discretization of the domain. From a physical point of view,

there is an optimal neighbourhood around any point  $y_0$  from which we must include information. Hence, if we increase the resolution of our model the spatial window should increase respectively. With regards to the time-delays, we choose 16 time-delays for  $\mathbb{D}_1$  and 12 for  $\mathbb{D}_2$ .

At this point we point out that what we seek to do is inline prediction. This means that the neural nets described by Eq. (3.1) must be coupled with the evolution Eqs. (2.18)-(2.19). For a simple forward Euler scheme for temporal integration, this would imply that by knowing the values of  $\bar{u}_1, \bar{\rho}$  at time  $t$  we can predict the closure terms at time  $t$  using Eq. (3.1) and use their values to integrate Eqs. (2.18)-(2.19) by one time-step  $\delta t$  so that we compute  $\bar{u}_1, \bar{\rho}$  at time  $t + 1$ . However, if we want to use a higher-order integration scheme like a 4th-order explicit Runge-Kutta, we need to evaluate the closure terms at time  $t + 1/2$  as well. Since, we do not have access to the required time-history for such a prediction, we instead integrate in time not by  $\delta t$  but by  $2\delta t$  and thus get a time-integration error of the for  $O[(2\delta t)^4]$ .

In terms of the training process itself, we normalize input and output data as usually suggested [37]. The loss function for this problem is chosen to be the single-step prediction mean square error, which can be written as

$$L(\Theta, X) = \frac{1}{N} \frac{1}{T} \sum_{i=1}^N \sum_{t=1}^T \left\| D_1(\Theta; \{\bar{\xi}\}) - \partial_y(\overline{u'_1 u'_2}) \right\|^2 \quad (3.2)$$

$$L(\Theta, X) = \frac{1}{N} \frac{1}{T} \sum_{i=1}^N \sum_{t=1}^T \left\| D_2(\Theta; \{\bar{\xi}\}) - \partial_y(\overline{v'_2 \rho'}) \right\|^2 \quad (3.3)$$

Furthermore, we achieve learning by using the Adam method [21], which is a widely popular first-order gradient based method. Finally, we must investigate which flow features are important as input for each of the two models. As we mentioned previously we use their values at time  $t$ , together with some time-delays to predict the closure terms at time  $t$ . However, some of the flow features may not be known at

time  $t$ . In this case, we can set this value of the input-vector to zero, or shift their time-history by one time-step (effectively starting from  $t - 1$ ).

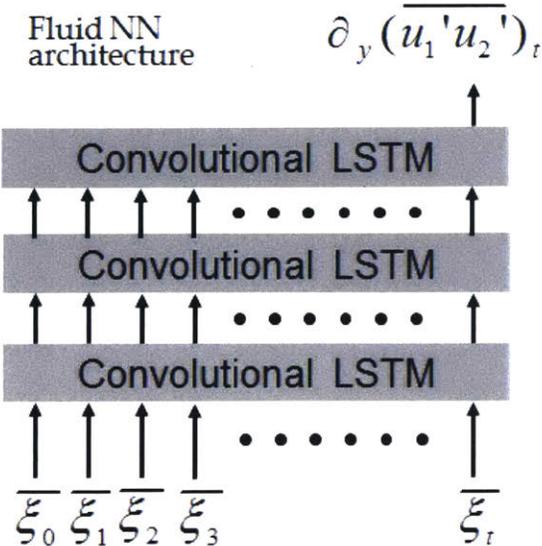


Figure 3-2: Architecture of the neural net we use for predicting closure term  $\partial_y \overline{(u'_1 u'_2)}$ .

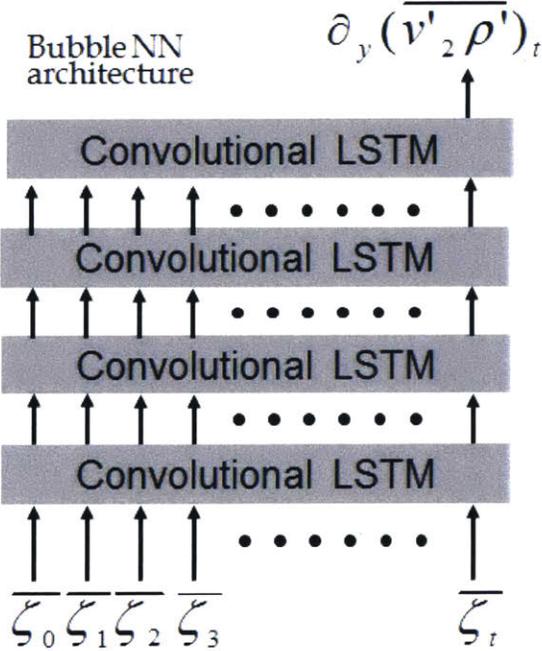


Figure 3-3: Architecture of the neural net we use for predicting closure term  $\partial_y \overline{(v'_2 \rho')}$ .

## 3.2 Takens Theorem

With respect to the justification of our modeling choice, the use of nonlocal description in space, as stated previously, can be justified for example by the incompressibility condition, which is an inherently nonlocal kinematic constraint. Yet, the nonlocality in time may seem counter-intuitive for a Markovian system. However, we should remember that in this problem we do not have access to all the degrees of freedom needed to fully describe this turbulent system. Since we are implementing a closure scheme we are a priori omitting some variables that are essential for describing the dynamics of the system (i.e. small-scale effects). Hence, we make use of Takens theorem [42] which informally states that if we know only a limited number of the state variables of a system, we can gain the attractor of the full system by using delay embedding of a state variable. In more detail, it allows for the construction of an attractor that is diffeomorphic to the original chaotic attractor from a time series of a single measurement. Under certain conditions, the full dynamics of a system as complicated as a turbulent fluid may be uncovered from a time-series of a single point measurement. Although the theorem itself is quite old, it was not used until recently because the theorem itself does not provide a way to compute said diffeomorphism. However, with the introduction of machine learning neural nets can actually learn this diffeomorphism as part of their training process [8]. It has been shown to enhance accuracy of predictions in [46], [48].

## 3.3 Training method

Before we begin presenting training and testing results it is required that we describe how we pick our training and testing data. To showcase that our method can be applied to a variety of flows and not only to flows that it was trained on (i.e. the neural net generalizes well) we trained on unimodal jets and we test on bimodal ones. This means that we trained on unimodal jets for flows with Reynolds number in the range [1000, 2000] and for inertial parameter of bubbles  $\epsilon = 0.05$ . We mention again

Table 3.1: Feature selection for closure of Navier-Stokes

$\xi$ feature selection			
Features	Dim	Train-MSE	Val-MSE
$\bar{u}_1$	1	0.0935	1.6194
$\partial_t \bar{u}_1$	1	0.0385	0.5330
$\partial_y (\overline{u'_1 u'_2})$	1	0.0232	0.1785
$\bar{u}_1, \partial_t \bar{u}_1$	2	0.0485	0.4964
$\bar{u}_1, \partial_y (\overline{u'_1 u'_2})$	2	0.0248	0.1574
$\partial_t \bar{u}_1, \partial_y (\overline{u'_1 u'_2})$	2	0.0252	0.1161
$\bar{u}_1, \partial_t \bar{u}_1, \partial_y (\overline{u'_1 u'_2})$	3	0.0282	0.0340

that the filtered model is a  $1D$  problem in space and we use 80 points in space to simulate it. We compare the results of the averaged model with the predictions of the  $2D$  reference solutions that we derived using spectral method and  $256^2$  models.

### 3.4 Feature selection

As we previously stated, we have to decide what flow features to use as inputs for the neural networks. At this time, this decision is taken through a heuristic approach. We choose some possibly important features and train using different combinations of them. We then look at both the training and the testing error. We finally deem as optimal the combination that minimizes both the training and validation error. We note that if we only looked at the training error itself we run the risk of overfitting. We carry out this process individually for each of the closure terms for the Navier-Stokes equation and the transport equation. For closure term  $\mathbb{D}_1$  (corresponding to the fluid flow) we try as possible flow features the quantities  $\bar{u}_1, \partial_t \bar{u}_1, \partial_y (\overline{u'_1 u'_2})$ . We then obtain the following table with training and validation errors:

From Table 1, we observe that the single most important feature is the material derivative of the fluid velocity  $D\bar{u}/Dt$ . Furthermore, we see that the optimal combination of features is to select all the used features. We used 3 convolutional-LSTM layers to model this closure term. Finally, we also mention that 12 time-delays were found to be adequate at minimizing the mean-square error to an acceptable value.

For the closure of the transport equation we carry out the same process in Table

2. We observe that the single most important features seems to be  $\bar{\rho}$ . For an acceptable mean-square error (both training and validation) we choose the combination  $\bar{u}, \bar{\rho}, \partial_t \bar{u}, \partial_y(\overline{\rho'v'})$ . 4 convolutional-LSTM layers and 16 time-delays where used for the architecture of this neural net.

Table 3.2: Feature selection for closure of bubble transport equation

$\zeta$ feature selection			
Features	Dim	Train-MSE	Val-MSE
$\bar{\rho}$	1	0.1086	0.1502
$\bar{v}$	2	0.6036	0.6730
$\bar{v}, \bar{\rho}$	3	0.0806	0.0895
$\bar{v}, \bar{\rho}, \partial_t \bar{v}, \partial_t \bar{\rho}$	6	0.0579	0.0601
$\bar{v}, \bar{\rho}, \partial_t \bar{v}, \partial_y(\overline{\rho'v'_2})$	6	0.0253	0.0307
$\bar{v}, \bar{\rho}, \partial_t \bar{\rho}, \partial_y(\overline{\rho'v'_2})$	5	0.0258	0.0356
$\bar{v}, \bar{\rho}, \partial_t \bar{v}, \partial_t \bar{\rho}, \partial_y(\overline{\rho'v'_2})$	7	0.0247	0.0310

Having chosen the quantities  $\bar{\rho}, \bar{v}, \partial_t \bar{v}, \partial_y(\overline{u'_1}, v'_2)$  as input arguments for  $\mathbb{D}_2$ , we present the training and validation error as we train over 100 periods. We can see that the value of both errors is quite similar (hinting towards generalizability of the predictions) and plateaus after about 60 epochs.

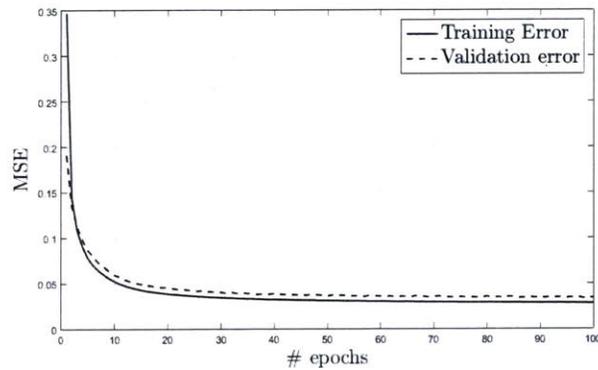


Figure 3-4: Mean square training-error (solid line) and validation error (dashed line) for  $\mathbb{D}_2$ .

### 3.5 Imitation learning

Before we begin showing results, we mention a final subtlety of the approach. While we use the single-step prediction for training, we want to use these models for multi-step prediction. The feedback effect caused by refeeding the prediction error of the neural net to itself invalidates the iid assumption between training and testing data. Under these circumstances, the error can be shown to grow exponentially [45]. This effect was observed in this problem as well, with the filtered dynamical system becoming unstable after a few hundred time-steps.

To alleviate this problem we use a version of imitation learning presented in [45]. This meta-algorithm is called Data as Demonstrator (DAD). That is after we undergo a round of initial training we make predictions and demand that after a certain number of time-steps the prediction should again match the training data. Assuming an error tolerance  $\delta$ , for the predictions of the closure term  $\partial_y(\overline{u'_1 u'_2})$ , the approach consists of the following steps

- (0) Let  $s = t$ .
- (1) From training data  $\overline{u}_1|_s$  predict  $\partial_y(\overline{u'_1 u'_2})^*|_s$ .
- (2) Evolve Eq. (2.18) to compute  $\overline{u}_1^*|_{s+1}$ .
- (3) Predict  $\partial_y(\overline{u'_1 u'_2})^*|_{s+1}$ .
- (4) Replace  $s$  with  $s + 1$  and repeat steps (1)-(3) until  $\left\| \partial_y(\overline{u'_1 u'_2})^*|_s - \partial_y(\overline{u'_1 u'_2})|_s \right\| > \delta$ .
- (5) Compute  $\partial_y(\overline{u'_1 u'_2})^*|_{s+1}$  so that  $\overline{u}_1^*|_{s+2} = \overline{u}_1|_{s+2}$ .

In the above pseudo-algorithm, quantities denoted with  $*$  correspond to predictions of the neural net while quantities without the star correspond to training data obtained from the reference solutions. As we can see, we demand that once our error tolerance is exceeded the predictions should go back to the training data exactly. This demand creates an additional training point  $(\overline{u}_1^*, \partial_y(\overline{u'_1 u'_2})^*|_{s+2})$ . By repeating this process, we create additional training data which we use together with the initial "actual" training data to continue training the model. Of course, we follow the exact same approach for the closure term  $\partial_y(\overline{\rho' v'_2})$  as well. We repeated this process 20 times, in order to achieve the robustness of our models. Training took approximately

20 hours on an i5-4210U CPU 1.70GHZ model, running on a single core.



# Chapter 4

## Numerical Results

We now present numerical results regarding our closure scheme. As stated in the previous chapter, we train on unimodal jets for  $Re \in [1000, 2000]$  and amplitude  $A \in [0.8, 1.2]$ . For now, we set the inertia parameter to  $\epsilon = 0.05$ . Each training case contains data inside the time-interval  $t \in [0, 600]$ . The spatial domain is always assumed to be  $[0, 2\pi]^2$  with doubly periodic boundary conditions.

### 4.1 Validation on uni-modal jets

In this section we present results for a validation case of a unimodal jet. The initial data for this case is

$$u_{1,jet} = \exp \left[ -2(y - \pi)^2 \right], \quad u_{2,jet} = 0. \quad (4.1)$$

We present results both for the fluid velocity statistics as well as for the bubble distribution statistics. We note here that comparisons are made between the time-averaged results that the (1D) closure scheme produces and the statistics of the (2D) reference solution. For the fluid velocity field, from the reference solution we compute the time-averaged jet-profile of the statistical steady state to be

$$\langle \bar{u}_1 \rangle = \frac{1}{400} \int_{t=200}^{t=600} \bar{u}_1^{ref} dt = \frac{1}{400} \frac{1}{2\pi} \int_{t=200}^{t=600} \int_0^{2\pi} u_1^{ref}(x, y, t) dx dt, \quad (4.2)$$

where  $u_1^{ref}$  is the time-series of  $u_1$  we obtain from the 2D reference solution. Note that we omit the first transient part of our simulations. We can carry out the same calculation using the results from our 1D closure scheme by using

$$\langle \bar{u}_1 \rangle = \frac{1}{400} \int_{t=200}^{t=600} \bar{u}_1^{ML}(y, t) dt, \quad (4.3)$$

where  $\bar{u}_1^{ML}$  is the time-series of  $\bar{u}_1$ , which corresponds to the filtered version of  $u_1$ , as predicted by our closure scheme. We mention that due to the homogeneity of the forcing term with respect to  $x$ , we expect that the statistical steady state of the flow should be independent of  $x$ . Comparison between the prediction of the statistics for the fluid flow between the reference solution and our closure scheme is depicted below.

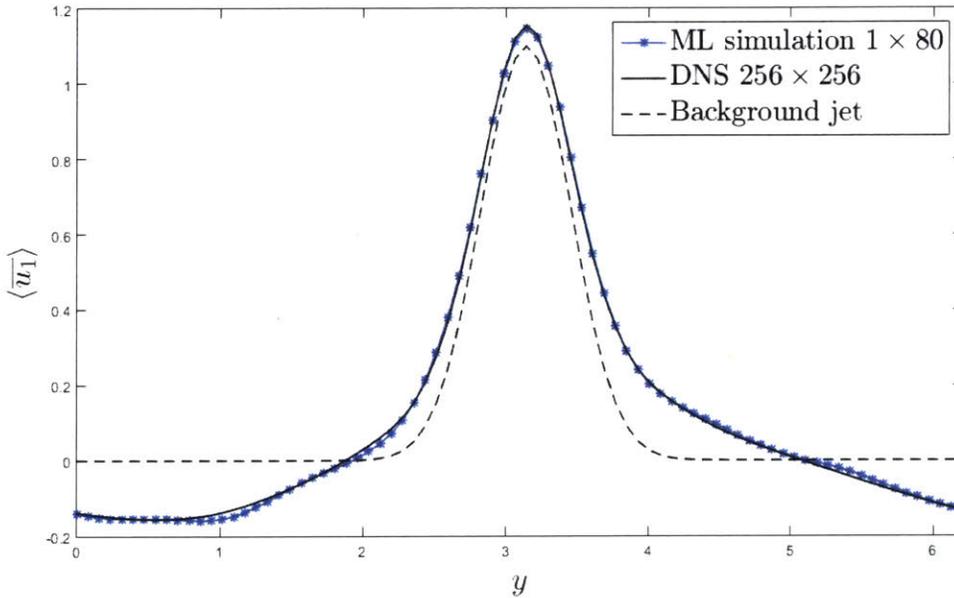


Figure 4-1: Time-averaged profile of  $\bar{u}_1$  for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line.

The results appear to be in very good agreement showcasing that our closure scheme is able to predict the statistical steady state of the flow. We can also observe the Rayleigh instability the initial jet profile (dashed line) undergoes due to the excitation by the external forcing. Our model appears able to predict this new

statistical steady state which clearly differs from the initial conditions. Finally, we should note that the slight asymmetry the velocity profile exhibits is due to a minor inhomogeneity of the forcing term along the  $y$  direction.

We now turn our attention to the statistics of the bubble distribution. We apply the same operation describe above to compute  $\langle \bar{\rho} \rangle$  from both the reference solution and the data-based closure scheme. Comparison between the two model is presented below.

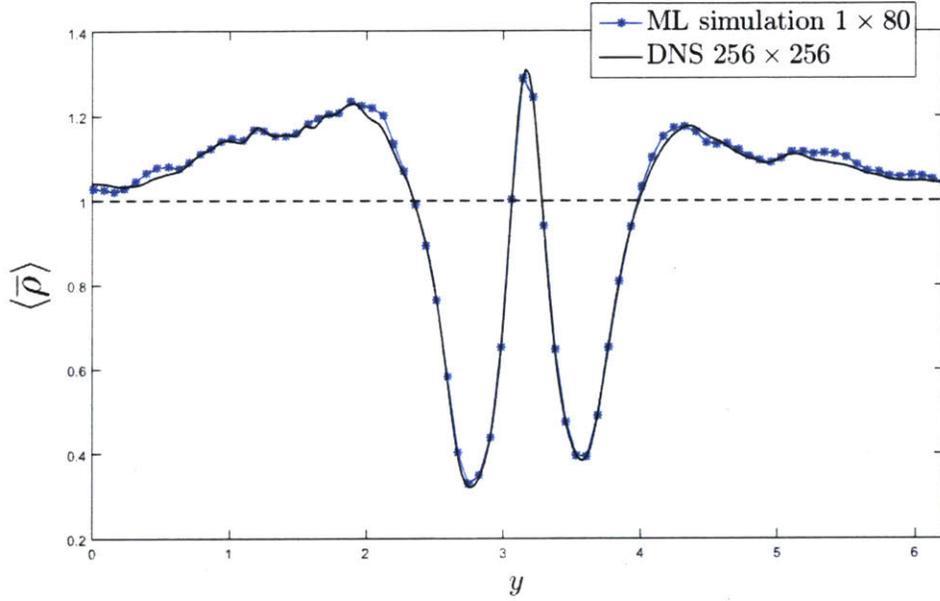


Figure 4-2: Time-averaged profile of  $\bar{\rho}$  for the ML closure model (blue line) and the DNS (black line). Initial bubble distribution is depicted with dashed line.

We again see very good agreement between the machine learning approach and the reference solution, for the statistics of the distribution of the bubbles. By observing the initial distribution of bubbles (dashed line) we note that the bubbles seem to cluster in at the core of the jet and be repelled from the adjacent areas of the jet core. To look more into this behaviour of our simulation, using the reference solution data, we plot the Forward FTLE (Finite-time Lyapunov exponent) [17] results for the bubble velocity field. To do so, we utilize the flow map  $F_{t_0}^t(\mathbf{x}_0) = \mathbf{x}(t; t_0, \mathbf{x}_0)$ . From this quantity we can compute the Cauchy-Green strain tensor to be

$$C(\mathbf{x}_0) = \nabla F_{t_0}^t(\mathbf{x}_0) \nabla F_{t_0}^t(\mathbf{x}_0). \quad (4.4)$$

From this quantity we can then compute its eigenvalues at each point in space and deduce how much these lines act as repellents. To compute the Cauchy-Green strain tensor we use a new grid that where we interpolate the velocity field of the bubbles. We then add an additional local auxiliary grid around each point of the main grid. A schematic of this configuration is depicted in Figure 6-4. In Figure 6-3 we display our Forward FTLE results. We can clearly see that the area around the core of the jet is a strongly repelling one and thus justifies the statistical steady state of the bubble distribution displayed in Figure 6-2.

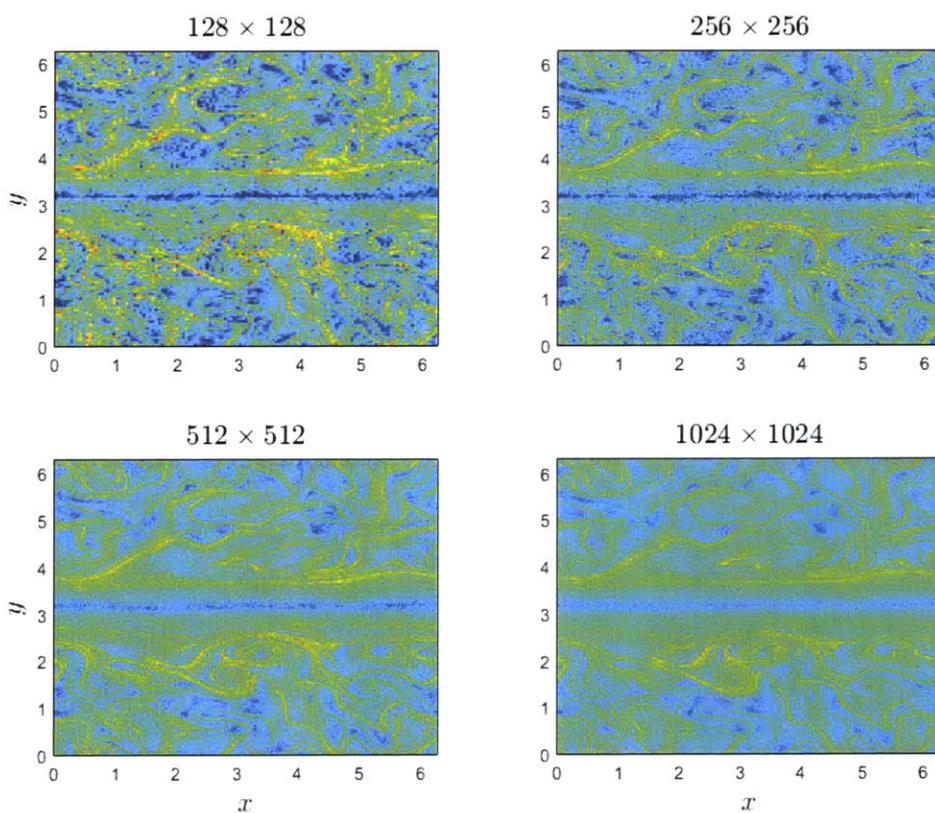


Figure 4-3: Forward FTLE simulations for different resolutions, for unimodal validation case for  $t - t_0 = 10$ . Yellow corresponds to repelling areas.

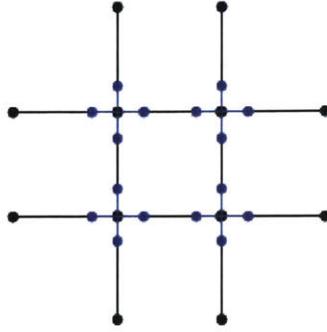


Figure 4-4: Illustration of the main grid (black dots) and the auxiliary grid (blue dots) used for the calculation of the Cauchy-Green strain tensor.

## 4.2 Testing generalizability on bi-modal jets

For test cases, we first present results for a bimodal symmetric jet for  $Re = 1500$  and  $\epsilon = 0.05$ . The initial jet-structure we impose on the flow is  $u_1 = \exp[-2(y - \pi)^2]$ . We compare the statistical profile of the fluid velocity field between the two models.

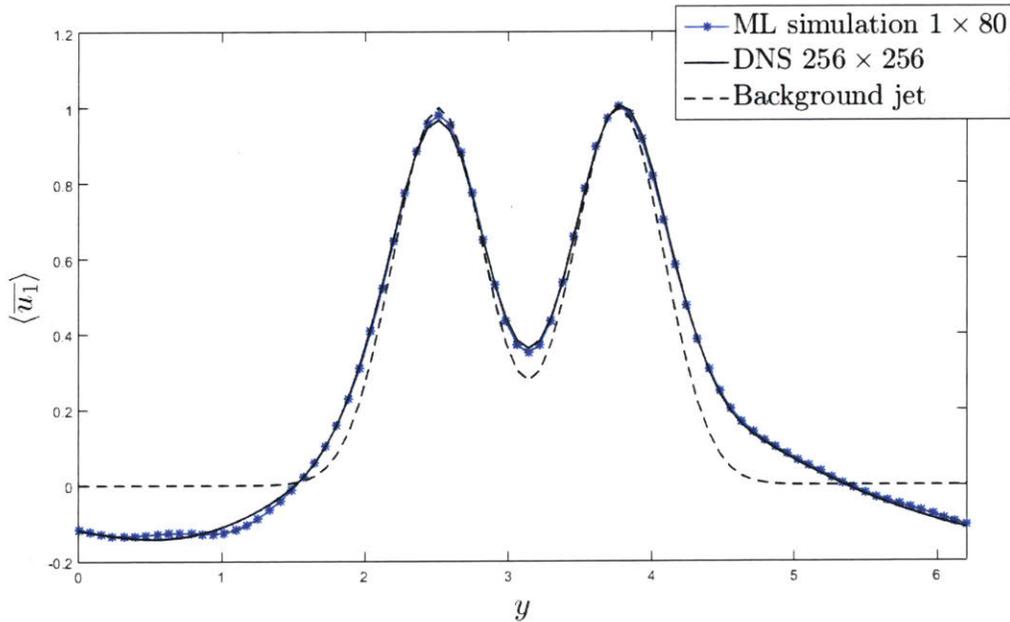


Figure 4-5: Time-averaged profile of  $\bar{u}_1$  for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line.

We observe very good agreement between the two models. Once again, we see that the closure scheme is able to predict the correct statistical profile for  $u_1$  after the system exhibited a Rayleigh instability (shifting from the initial jet profile depicted in dashed line).

We now attempt to test our model with a more complicated form. That is, we choose a bimodal asymmetric jet for  $Re = 1500$  and  $\epsilon = 0.05$ . The initial jet-structure we impose on the flow is  $u_1 = 0.5 \exp[-3(y - 0.8\pi)^2] + \exp[-3(y - 1.2\pi)^2]$ . Results regarding the time-averaged profile of  $\bar{u}_1$  are depicted below.

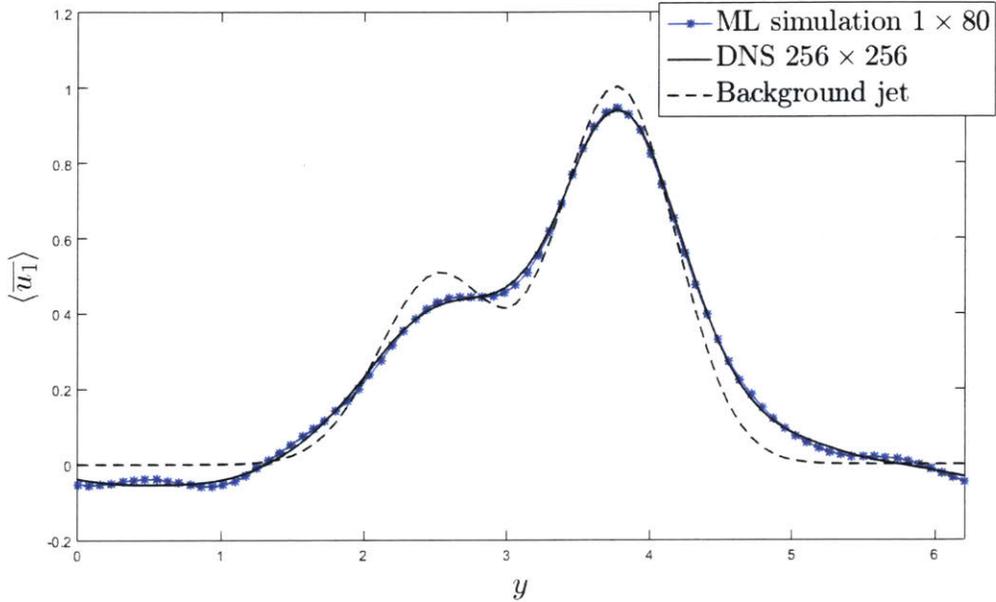


Figure 4-6: Time-averaged profile of  $\bar{u}_1$  for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line.

Once again, we see very good agreement between the two models. These results, encourage us that our model indeed is able to makes accurate predictions for flows it has not been trained on. The closure model is again able to predict the correct statistical steady state after the occurrence of the instability. We now proceed to compare the reference solution and the closure model for the time-averaged distribution of bubbles. This time, we compare our predictions with these of a simple LES model in

a  $80 \times 80$  discretization, where we replace the closure term for the flow with a linear eddy-viscosity model (whose parameters are tuned through trial and error) and then we use the transport Eq. (2.17) to compute our distribution of bubbles.

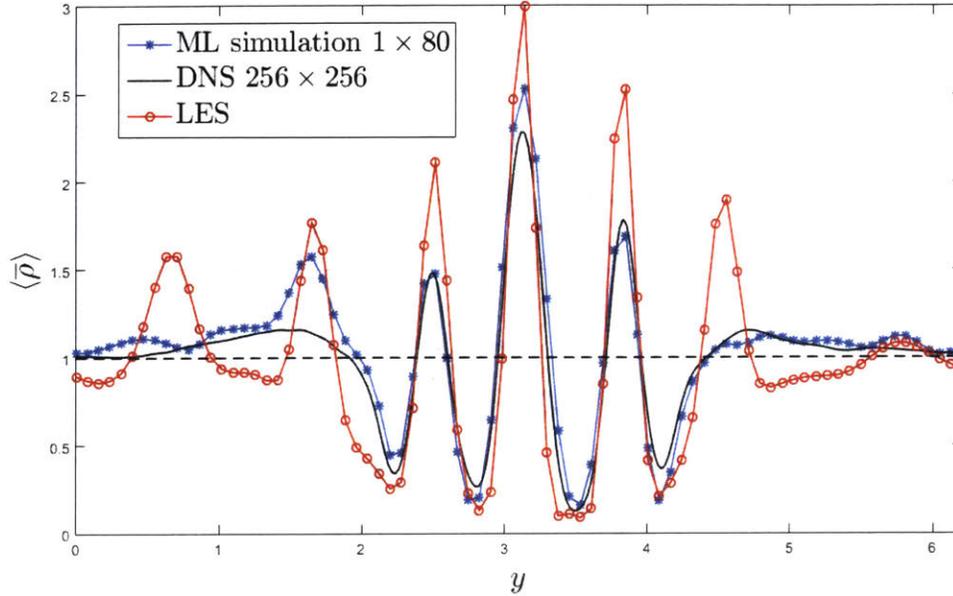


Figure 4-7: Time-averaged profile of  $\bar{\rho}$  for the ML closure model (blue line), LES model (red line) and the DNS (black line). Initial bubble distribution is depicted with dashed line.

We see that our approach predicts much better the time-averaged distribution of the bubbles compared to the LES model. We can also see that the error in our predictions in this case seems to be larger than that for the velocity field of the flow. This result can be attributed to two factors. First, the predictions for this closure term rely on the predictions of the previous closure term (hence error accumulates) and second, the flow of the bubbles is a highly compressible flow that is harder to capture than that of the incompressible fluid itself.

### 4.3 Limits of predictability

As a final case we want to demonstrate when this model would be expected to not perform well. Neural networks are essentially a way of interpolating data. If performed correctly, we can hope that this interpolation has a physical interpretation and not just overfit on the training data. Hence, it is very unlikely that a model would be able to perform well for a flow that is well outside the data it is trained on. For such a case we present a unimodal jet for which the Reynolds number is set to  $Re = 3000$ .

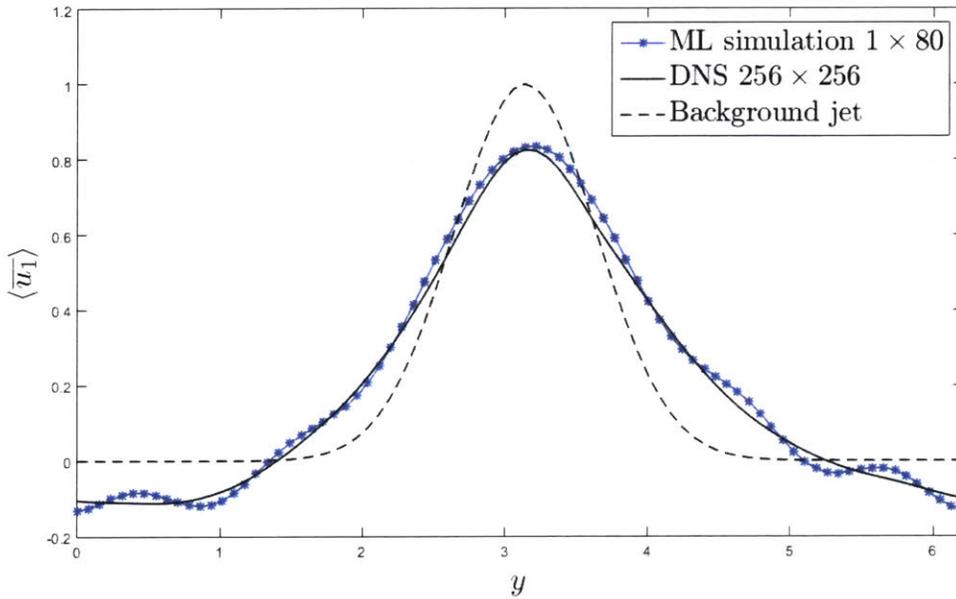


Figure 4-8: Time-averaged profile of  $\bar{u}_1$  for the ML closure model (blue line) and the DNS (black line). The shape of the jet that is imposed by the large-scale forcing is depicted with dashed line.

In this case our model performs demonstrably worse for the statistics of the velocity field of the problem. We should also note that for this case the evolution of the distribution of the bubbles causes a blow up (evolved using the closure scheme). Hence, it is intuitively reasonable to assume that a data-based approach for closure schemes would be not good for a flow that is more turbulent than any flow it is trained on.

# Chapter 5

## Machine learning statistics of the deformation of cavitating bubbles

In the previous chapters we dealt with approximating the transport of bubbles under the influence of the underlying fluid flow. To this end, we assumed that in no way do the the bubbles affect the underlying fluid. However, it is intuitive that such an assumption is non-physical and in general will not be true in a realistic application, unless the population of bubbles is very small. Yet, the difficulty of modelling such effects can be particularly cumbersome, which is why we neglected studying this effect.

In this chapter, we carry out some preliminary work towards fully coupling the bubble flow and the fluid flow. In a somewhat simplified framework, we can point to two main mechanisms with which the bubbles affect the surrounding fluid. The first is the motion of the bubble as a rigid body, while the second is the deformation of the bubble as it cruises through the fluid. It is obvious that these two reasons are not independent of one another, however, in most generic applications the time-scales at which the bubble (i) moves as a rigid body and (ii) deforms, are distinct enough to study the two phenomena separately. In this chapter we be interested in modelling the effects of the deformation of the bubbles to the flow.

To do so, we must first choose a model that describes the deformation of the bubbles. We use the Rayleigh-Plesset equation, for a single isolated bubble, which is a simplified model for such effects. To look now at how the deformation of the bubbles

affects the mixture, we can adopt an ensemble point of view at each location in space. This can be done by introducing the ensemble averaging operator  $\overline{(\cdot)}$ . Under these assumptions, and defining  $R$  as the radius of each bubble and  $\dot{R}$  its time derivative, it can be shown [50], [9] that, the mixture pressure takes the form

$$p = (1 - \alpha_b)p_l + \alpha_b \left( \frac{\overline{R^3 p_{bw}}}{\overline{R^3}} - \rho_b \frac{\overline{R^3 \dot{R}^2}}{\overline{R^3}} \right), \quad (5.1)$$

where  $p_{bw}$  is the bubble-wall pressure. We also note that the transport equation for the bubble void fraction, has to change from the one used in previous chapters so the compressibility effects are taken into account. These effects can be shown [50] to yield the following equation

$$\frac{\partial \alpha_b}{\partial t} + u \cdot \nabla \alpha_b = 3\alpha_b \frac{\overline{R^2 \dot{R}}}{\overline{R^3}}. \quad (5.2)$$

It is therefore apparent that we must compute the high-order moments  $\overline{R^3}$ ,  $\overline{R^2 \dot{R}}$ ,  $\overline{R^3 \dot{R}^2}$  and  $\overline{R^3 p_{bw}}$ . In this chapter we focus on the first three moments (since they only require  $R$ ,  $\dot{R}$  for their computation) and discuss at the end an approximation for the last required moment. Using this equation as a reference, we perform Monte-Carlo simulations and compute the time-history of the required moments. However, such an approach is computationally demanding. Our alternative here relies on deriving evolution equations for the required moments using Machine Learning. However, the required moments are up to 5th order. Instead of evolving all the required moments up to 5th order, we actually evolve only the moments up to 2nd order. Note that this is essentially the same computational cost a Gaussian closure scheme requires. After that, we utilize recurrent neural nets to predict the time-history of the higher-order moments using these time-delays. We utilize neural networks with memory for the predictions of the higher-order moments based on the related research that states that such moments can be approximated by knowing the time-history of the low-order moments [14].

## 5.1 Problem Statement

We assume the bubbles to be pulsating perfect spheres in an infinite medium. That is, we assume that each bubble changes only in volume, always remaining a perfect sphere. The center of mass of each bubble remains motionless. We also assume that a time-independent pressure force is applied to the bubble with a wavelength that is much larger than the radius of the bubble. We parametrize this forcing by the ratio of the vapour pressure inside the bubble under equilibrium conditions,  $p_0$ , and of the fluid far-field pressure around the bubble  $p_\infty$ . We assume the bubble to be filled uniformly with gas (of uniform temperature) whose density is much smaller than that of the surrounding liquid. Finally, we neglect interactions between bubbles (we assume that the number of bubbles inside the liquid is small enough for this to be valid). We also neglect body forces (we also neglected gravity in the previous chapter) and surface tension. Hence, under these assumptions we can use the simplified Rayleigh-Plesset equation to model the evolution of the radius  $R$  of each bubble as

$$R\ddot{R} + \frac{3}{2}\dot{R}^2 + \frac{4}{Re} \frac{\dot{R}}{R} = C_a \left[ \left( \frac{R_0}{R} \right)^{3\gamma} - 1 \right] - C_p, \quad (5.3)$$

where  $\gamma = 1.4$  is the polytropic index (assuming adiabatic process) and  $R_0$  is the equilibrium radius of the particular bubble under study. This equation can be derived by equilibrating the kinetic energy of the pulsating bubble with the energy provided by the fluid through the constant pressure forcing [22]. The dimensionless parameters used above are the Reynolds number, cavitation number and the pressure forcing which are defined respectively as

$$Re = \sqrt{\frac{p_0}{\rho_0}} \frac{R_0}{\nu_0}, \quad C_a = \frac{p_0 - p_v}{p_0}, \quad \text{and} \quad C_p = \frac{p_\infty - p_0}{p_0}, \quad (5.4)$$

where we used the reference quantities  $\nu_0$  as the kinematic viscosity,  $\rho_0$  as the liquid density as well as the ambient pressure  $p_v$ .

In general, for different bubbles the equilibrium radius is going to be different. This variation adds polydispersity to the problem. As a result, the variables of the

Rayleigh-Plesset equation that can be though to be stochastic are  $\{R, \dot{R}, R_0\}$ . Yet, in this study we simplify our problem by focusing on monodisperse clusters of bubbles. Hence, the only variables that are treated as stochastic are  $\{R, \dot{R}\}$  and we consider  $R_0$  the same for all bubbles.

We could try and solve the Rayleigh-Plesset equation for a variety of different initial conditions  $R, \dot{R}$ , yet solving numerically this equation is particularly cumbersome. Hence we choose to model this monodisperse problem as the evolution of a multivariate probability density function. This approach seems more natural, especially when only certain moments of this probability density function are needed for us to model the effects of the oscillations of bubbles back to the fluid flow. Another reason for avoiding Monte-Carlo simulations for this problem is its computationally forbidding nature as bubbles that are close to collapsing render the Rayleigh-Plesset equation singular and thus really hard to approximate numerically. Instead, we try to machine learn the evolution of such moments. Assuming that the random variables that govern the problem are  $x = \{R, \dot{R}\}$  we can define a probability density function that governs the problem

$$p = p(x, \theta, t), \quad (5.5)$$

where  $\theta$  is the probabilistic parameter. The moments corresponding to this pdf are defined as

$$\mu_{lm} = \int p R^l \dot{R}^m dx. \quad (5.6)$$

The dynamics of  $p$  are restricted by the equation

$$\frac{dp}{dt} = \frac{\partial p}{\partial t} + \frac{\partial}{\partial R}(p\dot{R}) + \frac{\partial}{\partial \dot{R}}(p\ddot{R}) = 0. \quad (5.7)$$

Note that we make use of the fact that  $\dot{R}_0 = 0$ . To evolve these moments in time we simply have to compute  $\frac{\partial \mu_{l,m}}{\partial t}$ . Reminding ourselves that  $\frac{dp}{dt} = 0$  and that we integrate over the entire 2D plane, we have

$$\frac{\partial \mu_{lm}}{\partial t} = l\mu_{l-1,m+1} + m \int R^l \dot{R}^{m-1} \ddot{R} p dx. \quad (5.8)$$

As we saw in the beginning of this chapter, the moments that need to be computed to model the effects of the bubbles back to the flow can be up to 5th order. Instead of evolving all moments up to 5th order we implement the following approach. We evolve the first and second order moments using recurrent neural networks. Then, we predict the need higher-order moments using as input only the time-history of the first and second order moments.

## 5.2 Machine Learning approach

In this section we seek to model the evolution of the first and second order moments of the problem using only Machine Learning. To this end we incorporate recurrent neural networks, in particular LSTM nets. We then use the derived time history to predict the evolution of the needed higher-order moments. For the evolution of the higher-order moments we once again use LSTM layers. As input for all these neural nets we have the (normalized) time-history of the first and second order moments.

### 5.2.1 Evolution of low-order moments

We model the evolution equations of the low-order moments as

$$\frac{\partial \mu_{lm}}{\partial t} = f_{lm}^{ML}(\{\mu_{lm}\}), \quad (l, m) = \{(1, 0), (0, 1), (2, 0), (0, 2), (1, 1)\}, \quad (5.9)$$

where  $f_{lm}^{ML}$  are the predictions of the right-hand-side of the evolution equations of the moments, as predicted by the neural nets we use. We model the right-hand-side of these evolution equations using a single LSTM layer followed by a fully-connected neural net. Each evolution equation corresponds to a different neural net. For input we use the time-history of the five low-order moments. We use as input the 32 previous time-steps of each of the moments. We train on cases  $p_0/p_\infty = 0.25, 0.35, 0.45, 0.55, 0.65, 0.75$ . We test on cases  $p_0/p_\infty = 0.20, 0.30, 0.3250.375, 0.40, 0.475$ .

We test on these values because cases where the pressure ratio is low are particularly hard for the Gaussian closure scheme to approximate and even so the estimates yield large errors. For reference solutions we use data derived from Monte-Carlo simulations. These Monte-Carlo simulations of the Rayleigh-Plesset equations were implemented and performed by Dr. Spencer Bryngelson in California Institute of Technology.

We are interested in the statistics of a cluster of bubbles with initial conditions sampled from a reference probability distribution. Specifically we study the oscillations in the size of the bubbles under a constant pressure ratio (initial pressure inside the bubble and applied pressure outside the bubble). We compare our Machine Learning approach with a Gaussian closure model for the low-order moments. The modelling as well as the numerical implementation of the Gaussian closure were carried out by Dr. Spencer Bryngelson from prof. Colonius' group at Caltech.

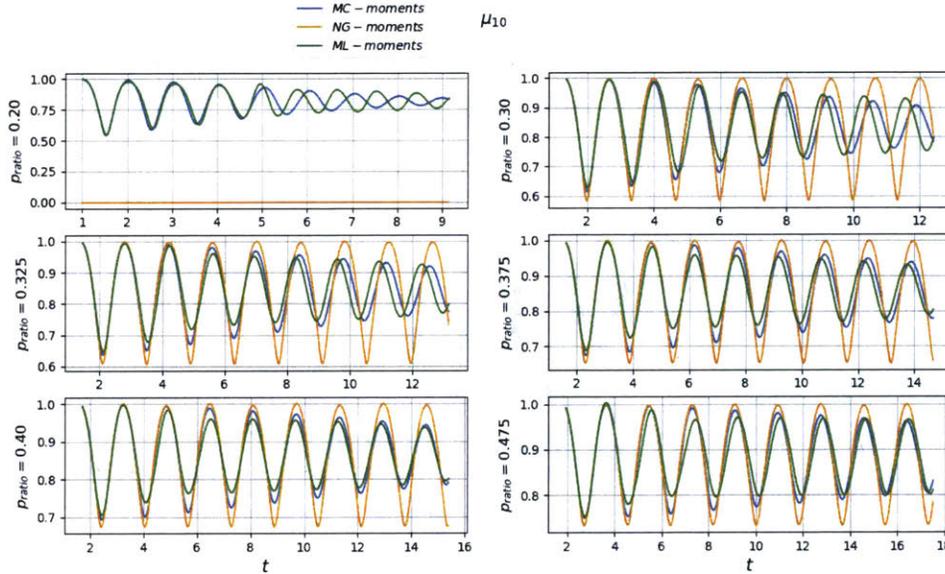


Figure 5-1: Time-series of moment  $\mu_{10}$  as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).

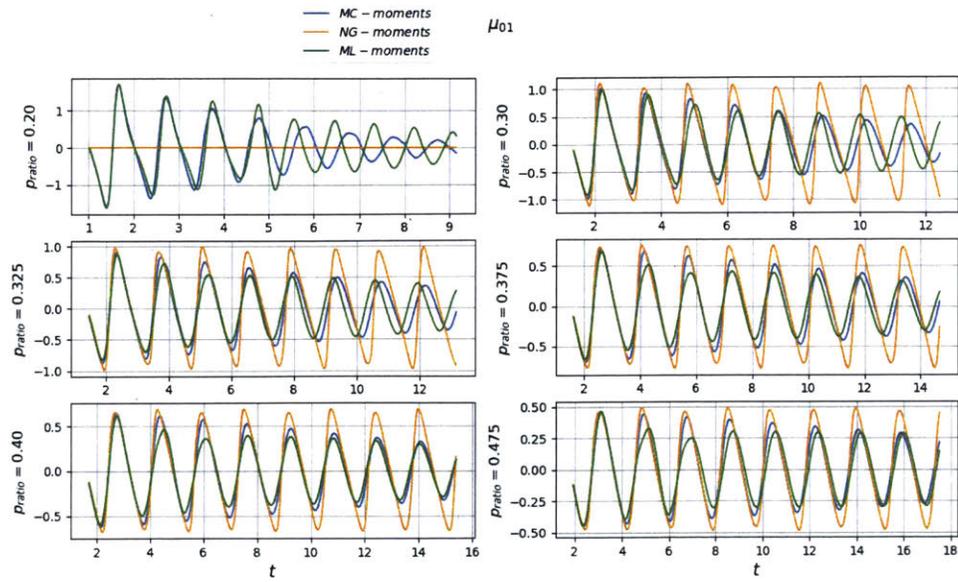


Figure 5-2: Time-series of moment  $\mu_{01}$  as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).

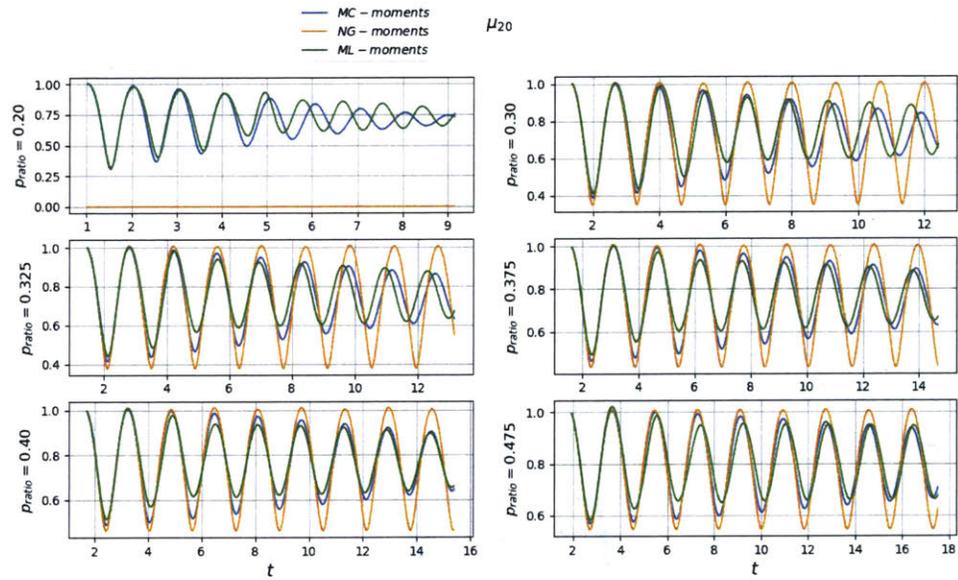


Figure 5-3: Time-series of moment  $\mu_{20}$  as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).

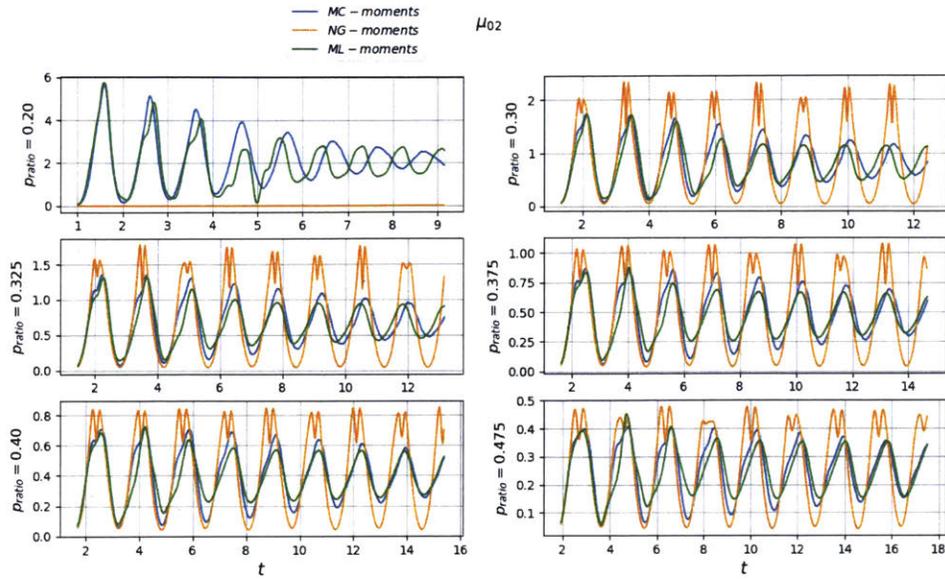


Figure 5-4: Time-series of moment  $\mu_{02}$  as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).

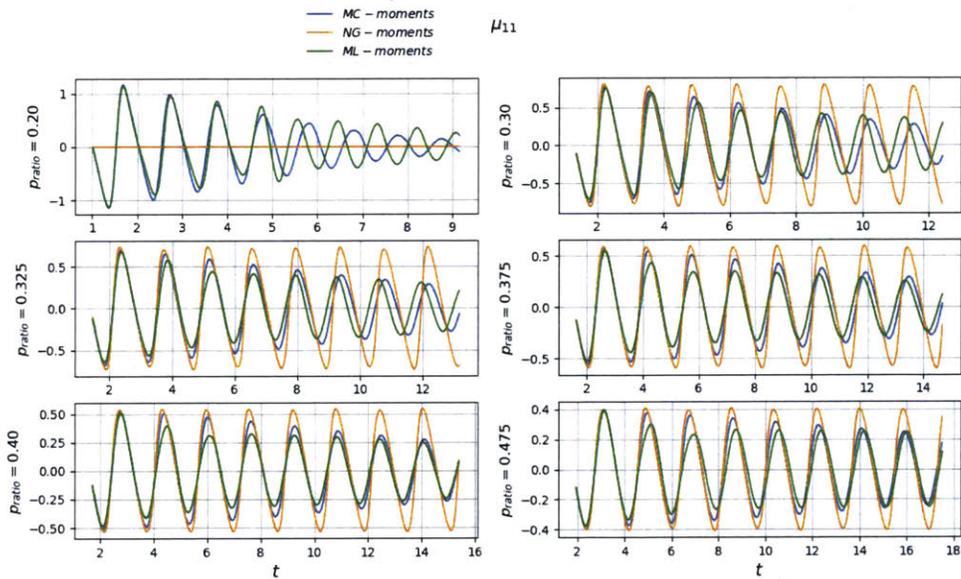


Figure 5-5: Time-series of moment  $\mu_{11}$  as predicted by Monte-Carlo simulation (blue), nonlinear Gaussian closure (orange) and Machine Learning (green).

We observe that in all cases the machine learning results outperform the Gaussian closure results. For  $p_0/p_\infty = 0.30$  and below we observe that the Machine Learning

results start being out of phase with the Monte Carlo results yet still outperforming the Gaussian closure results. We can notice that the Gaussian closure results cannot model the progressive reduction of the amplitude of oscillations of the bubble population, a physical characteristic captured by our neural network architecture. We observe that the neural net can perform adequately in cases it has never seen, allowing us to believe that it is not simply overfitting on the training data but actually learning the dynamics of the problem.

### 5.2.2 Evolution of high-order moments

As we stated previously, in order to model the effects of the oscillation of the bubbles back to the fluid flow, we must compute high-order moments (higher than 2nd-order). Trying to evolve the coupled system of all moments of up to 5th order, would be computationally expensive. Our approach in this section is to use the results of evolving the low-order moments and use this data to predict the required high-order moments.

As stated in the beginning of the chapter, the driving force behind our belief that a system with memory can give better predictions for the high-order moments, with knowledge of only the low-order moments stems from works such as [14]. In this work, the authors showcase that high-order moments of arbitrary probability measures can be approximated by knowledge of the time-history of the low-order moments.

For this purpose, we use recurrent neural networks to predict the higher-order moments in a non-Gaussian framework. These results can be compared both to the results derived from Monte-Carlo simulations of the Rayleigh-Plesset equation as well as the analytic predictions of a Gaussian closure scheme for the higher-order moments. In more detail, if we assume that the radius and its velocity for a bubble can be written as

$$R = \mu_{1,0} + r, \quad \dot{R} = \mu_{0,1} + \dot{r}, \quad (5.10)$$

where we assume that  $\overline{r^{2n+1}} = \overline{\dot{r}^{2n+1}} = 0$ . Under these assumptions we can then compute the required moments in a Gaussian framework to be

$$\mu_{3,0} = \mu_{1,0}^3 + 3\mu_{1,0}(\mu_{2,0} - \mu_{1,0}^2), \quad (5.11)$$

$$\mu_{1,2} = \mu_{1,0}\mu_{0,1}^2 + \mu_{1,0}(\mu_{0,2} - \mu_{0,1}^2) + 2\mu_{0,1}(\mu_{1,1} - \mu_{1,0}\mu_{0,1}), \quad (5.12)$$

$$\begin{aligned} \mu_{3,2} = & \mu_{1,0}^3\mu_{0,1}^2 + \mu_{1,0}^3(\mu_{0,2} - \mu_{0,1}^2) + 6\mu_{1,0}^2\mu_{0,1}(\mu_{1,1} - \mu_{1,0}\mu_{0,1}) + 3\mu_{1,0}\mu_{0,1}^2(\mu_{2,0} - \mu_{1,0}^2) \\ & + 9\mu_{1,0}(\mu_{2,0} - \mu_{1,0}^2) + 6\mu_{0,1}(\mu_{2,0} - \mu_{1,0}^2)(\mu_{0,2} - \mu_{0,1}^2). \end{aligned} \quad (5.13)$$

As a reference solution we use the Monte-Carlo predictions of the high-order moments. As input for the neural nets we use the time-series of the first and second-order moments as predicted by the Monte-Carlo simulations. However, during testing we use the low-order moments predicted by the ML approach we described in the previous section. We compare these results to the reference Monte-Carlo predictions as well as those derived by nonlinear Gaussian closure.

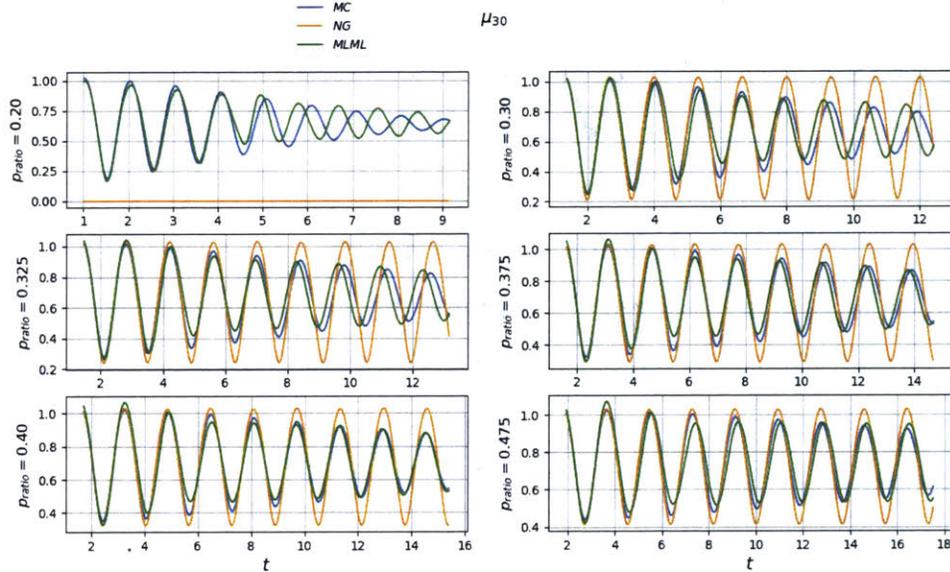


Figure 5-6: Time-history of  $\mu_{30}$  moment, as computed by Monte-Carlo data (blue), non-Gaussian closure (orange) and ML predictions (green).

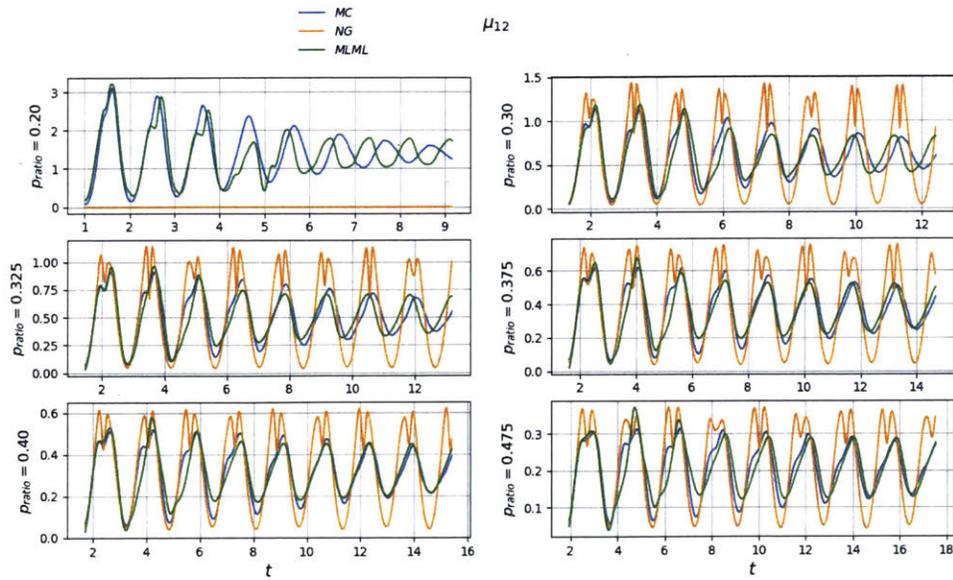


Figure 5-7: Time-history of  $\mu_{12}$  moment, as computed by Monte-Carlo data (blue), non-Gaussian closure (orange) and ML predictions (green).

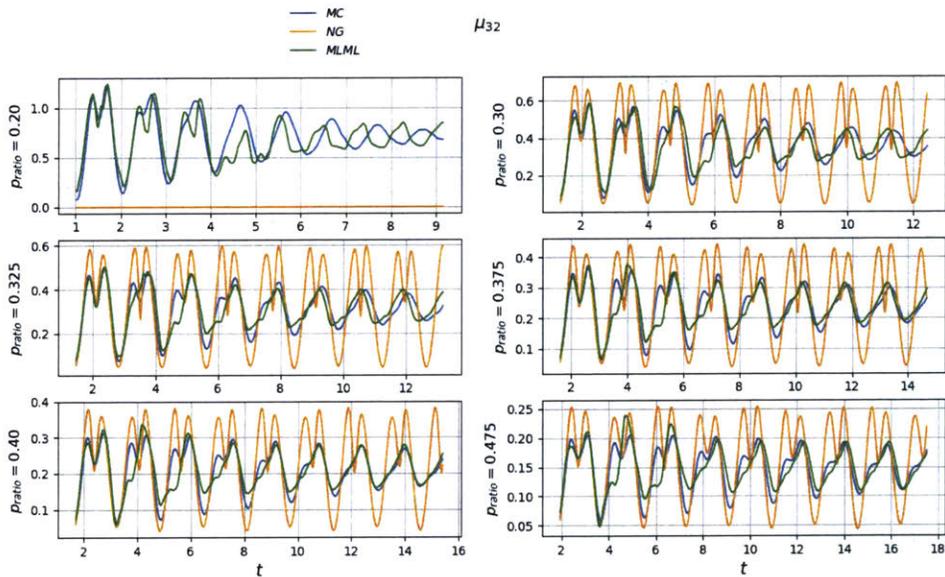


Figure 5-8: Time-history of  $\mu_{32}$  moment, as computed by Monte-Carlo data (blue), non-Gaussian closure (orange) and ML predictions (green).

we observe that our Machine Learning approach clearly outperforms the Gaussian closure results. We should emphasize that our results were taking by using the

Machine Learning predictions for the low-order moments and not the reference Monte-Carlo predictions. Our model seems to not be able to capture correctly the phase for the case  $p_0/p_\infty$ , however we should state that the Gaussian closure scheme could not run for such strong forcing. Once again our model is able to capture the apparent slow-down of the bubble dynamics as time progresses, a phenomenon missing from the Gaussian closure results.

Finally, we discuss some additional work that is currently being done in this problem. If we look at Eqs. (5.1), (5.2), we see that we still haven't computed the higher-order moment  $\overline{R^3 p_{bw}}$ . An easy way to carry out this calculation is to use a model of the form  $p_{bw} = p_{bw}(R, \dot{R})$ , i.e. expressing the pressure of the bubble at the wall as a function of  $R$  and  $\dot{R}$ . This modelling assumption renders the approach presented here perfectly applicable to this moment as well. Another approach for this problem that is currently jointly developed by the author of this thesis and Dr. Spencer Bryngelson, is to enhance the predictive capabilities of the Gaussian closure scheme for the moments problem with recurrent neural networks. Hence, instead of using a purely data-driven approach we employ a hybrid one. This approach is not discussed in this thesis.

# Chapter 6

## Conclusions and Future work

### 6.1 Conclusions

We have derived and demonstrated the implementation of a spatio-temporally non-local, data-driven closure scheme for anisotropic multiphase fluid flows. We focused on applying our idea to a 2D, doubly periodic domain, where bubbles that behave as passive inertial tracers are transported by an incompressible newtonian fluid. To this problem, we introduced a spatial filter that averaged the system along the one spatial direction  $x$  and applied a low-pass filter along the other direction  $y$ , reducing the original problem to a 1D problem in space. We then model the newly emerged closure terms using Stacked LSTM layers [16]. To introduce stability to the active integration of the neural net predictions to the time evolution of the filtered equations of motion, we utilize an imitation learning algorithm called DAD [45]. Using this approach, we integrate in time the multiphase flow and reliably reproduce its statistics.

For testing purposes we first trained our model using data from unimodal jets with Reynolds number  $Re \in [1000, 2000]$ . Our first test case was the validation of our method on unimodal jets not included in the training examples. The algorithm was able to correctly predict the statistical average of both the velocity field of the fluid and of the distribution of bubbles in space. We then proceeded to test our model on bimodal jets with the model capturing reliably the statistical average of the velocity

field of the fluid. For the distribution of bubbles in space the errors were more significant, yet it adequately captured the physical characteristics of the solution. That is, areas where bubbles are concentrated were correctly displayed as well as areas where bubbles are repelled. Our model also was shown to compare favourably with a simple linear eddy-viscosity closure model for this case.

## 6.2 Future work

For future work we plan to generalize the implementation of our closure scheme for 2D or 3D filters. We also want to replace the Maxey-Riley assumption for the velocity fields of the bubbles to a machine learning approach developed by Zhong Yi Wan from our group, that utilizes velocity data from the fully-resolved multiphase flow. In addition we would like to try causal convolutional networks [1] for this problem, as they seem to some times over-perform with respect to LSTM neural nets. Finally, we want to incorporate the effects of the bubble flow back to the fluid flow. Towards this goal, we have engaged in the work presented in Chapter 7.

# Bibliography

- [1] Shaojie Bai, J Zico Kolter, and Vladlen Koltun. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. *arXiv preprint arXiv:1803.01271*, 2018.
- [2] Pierre Baldi and Kurt Hornik. Neural networks and principal component analysis: Learning from examples without local minima. *Neural networks*, 2(1):53–58, 1989.
- [3] Yoshua Bengio, Patrice Simard, Paolo Frasconi, et al. Learning long-term dependencies with gradient descent is difficult. *IEEE transactions on neural networks*, 5(2):157–166, 1994.
- [4] Guido Boffetta and S Musacchio. Evidence for the double cascade scenario in two-dimensional turbulence. *Physical Review E*, 82(1):016307, 2010.
- [5] Thomas Bolton and Laure Zanna. Applications of deep learning to ocean data inference and subgrid parameterization. *Journal of Advances in Modeling Earth Systems*, 11(1):376–399, 2019.
- [6] Vadim Borue. Spectral exponents of enstrophy cascade in stationary two-dimensional homogeneous turbulence. *Physical review letters*, 71(24):3967, 1993.
- [7] Annalisa Bracco and James C McWilliams. Reynolds-number dependency in homogeneous, stationary two-dimensional turbulence. *Journal of Fluid Mechanics*, 646:517–526, 2010.
- [8] Steven L Brunton, Bingni W Brunton, Joshua L Proctor, Eureka Kaiser, and J Nathan Kutz. Chaos as an intermittently forced linear system. *Nature communications*, 8(1):19, 2017.
- [9] Spencer H Bryngelson, Kevin Schmidmayer, and Tim Colonius. A quantitative comparison of phase-averaged models for bubbly, cavitating flows. *International Journal of Multiphase Flow*, 115:137–143, 2019.
- [10] James W Deardorff. A numerical study of three-dimensional turbulent channel flow at large reynolds numbers. *Journal of Fluid Mechanics*, 41(2):453–480, 1970.

- [11] Mohammad Farazmand and George Haller. Computing lagrangian coherent structures from their variational theory. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, 22(1):013128, 2012.
- [12] Masataka Gamahara and Yuji Hattori. Searching for turbulence models by artificial neural network. *Physical Review Fluids*, 2(5):054604, 2017.
- [13] TB Gatski and T Jongen. Nonlinear eddy viscosity and algebraic stress models for solving complex turbulent flows. *Progress in Aerospace Sciences*, 36(8):655–682, 2000.
- [14] P. N. Gavriiadis. *Theoretical and numerical exploration of the moment problem with applications to the probabilistic predictions of stochastic responses of dynamical systems*. PhD thesis, National Technical University of Athens, 2005.
- [15] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *Advances in neural information processing systems*, pages 2672–2680, 2014.
- [16] Alex Graves, Abdel-rahman Mohamed, and Geoffrey Hinton. Speech recognition with deep recurrent neural networks. In *2013 IEEE international conference on acoustics, speech and signal processing*, pages 6645–6649. IEEE, 2013.
- [17] George Haller. Lagrangian coherent structures. *Annual Review of Fluid Mechanics*, 47:137–162, 2015.
- [18] George Haller and Themistoklis Sapsis. Where do inertial particles go in fluid flows? *Physica D: Nonlinear Phenomena*, 237(5):573–583, 2008.
- [19] Sepp Hochreiter and Jürgen Schmidhuber. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.
- [20] Lluís Jofre, Stefan P Domino, and Gianluca Iaccarino. A framework for characterizing structural uncertainty in large-eddy simulation closures. *Flow, Turbulence and Combustion*, 100(2):341–363, 2018.
- [21] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*, 2014.
- [22] TG Leighton. Derivation of the rayleigh-plesset equation in terms of volume. 2007.
- [23] Julia Ling, Reese Jones, and Jeremy Templeton. Machine learning strategies for systems with invariance properties. *Journal of Computational Physics*, 318:22–35, 2016.
- [24] Julia Ling, Andrew Kurzawski, and Jeremy Templeton. Reynolds averaged turbulence modelling using deep neural networks with embedded invariance. *Journal of Fluid Mechanics*, 807:155–166, 2016.

- [25] JL Lumley. Similarity and the turbulent energy spectrum. *The physics of fluids*, 10(4):855–858, 1967.
- [26] John L Lumley. Coherent structures in turbulence. In *Transition and turbulence*, pages 215–242. Elsevier, 1981.
- [27] Ming Ma, Jiakai Lu, and Gretar Tryggvason. Using statistical learning to close two-fluid multiphase flow equations for a simple bubbly system. *Physics of Fluids*, 27(9):092101, 2015.
- [28] Ming Ma, Jiakai Lu, and Gretar Tryggvason. Using statistical learning to close two-fluid multiphase flow equations for bubbly flows in vertical channels. *International Journal of Multiphase Flow*, 85:336–347, 2016.
- [29] Romit Maulik, Omer San, Adil Rasheed, and Prakash Vedula. Subgrid modelling for two-dimensional turbulence using neural networks. *Journal of Fluid Mechanics*, 858:122–144, 2019.
- [30] Martin R Maxey and James J Riley. Equation of motion for a small rigid sphere in a nonuniform flow. *The Physics of Fluids*, 26(4):883–889, 1983.
- [31] Andrea Mazzino, Paolo Muratore-Ginanneschi, and Stefano Musacchio. Scaling properties of the two-dimensional randomly stirred navier-stokes equation. *Physical review letters*, 99(14):144502, 2007.
- [32] Michele Milano and Petros Koumoutsakos. Neural network modeling for near wall turbulent flow. *Journal of Computational Physics*, 182(1):1–26, 2002.
- [33] Eric J Parish and Karthik Duraisamy. A paradigm for data-driven predictive modeling using field inversion and machine learning. *Journal of Computational Physics*, 305:758–774, 2016.
- [34] David E Rumelhart, Geoffrey E Hinton, and Ronald J Williams. Learning internal representations by error propagation. Technical report, California Univ San Diego La Jolla Inst for Cognitive Science, 1985.
- [35] David E Rumelhart, Geoffrey E Hinton, Ronald J Williams, et al. Learning representations by back-propagating errors. *Cognitive modeling*, 5(3):1, 1988.
- [36] Themistoklis Sapsis and George Haller. Inertial particle dynamics in a hurricane. *Journal of the Atmospheric Sciences*, 66(8):2481–2492, 2009.
- [37] Shai Shalev-Shwartz and Shai Ben-David. *Understanding machine learning: From theory to algorithms*. Cambridge university press, 2014.
- [38] Anand Pratap Singh, Shivaji Medida, and Karthik Duraisamy. Machine-learning-augmented predictive modeling of turbulent separated flows over airfoils. *AIAA Journal*, pages 2215–2227, 2017.

- [39] Lawrence Sirovich. Turbulence and the dynamics of coherent structures. i. coherent structures. *Quarterly of applied mathematics*, 45(3):561–571, 1987.
- [40] Joseph Smagorinsky. General circulation experiments with the primitive equations: I. the basic experiment. *Monthly weather review*, 91(3):99–164, 1963.
- [41] Jasper Snoek, Hugo Larochelle, and Ryan P Adams. Practical bayesian optimization of machine learning algorithms. In *Advances in neural information processing systems*, pages 2951–2959, 2012.
- [42] Floris Takens. Detecting strange attractors in turbulence. In *Dynamical systems and turbulence, Warwick 1980*, pages 366–381. Springer, 1981.
- [43] Hendrik Tennekes, John Leask Lumley, JL Lumley, et al. *A first course in turbulence*. MIT press, 1972.
- [44] Hannes Uecker. A short ad hoc introduction to spectral methods for parabolic pde and the navier-stokes equations. *Summer School Modern Computational Science*, pages 169–209, 2009.
- [45] Arun Venkatraman, Martial Hebert, and J Andrew Bagnell. Improving multi-step prediction of learned time series models. In *Twenty-Ninth AAAI Conference on Artificial Intelligence*, 2015.
- [46] Pantelis R Vlachas, Wonmin Byeon, Zhong Y Wan, Themistoklis P Sapsis, and Petros Koumoutsakos. Data-driven forecasting of high-dimensional chaotic systems with long short-term memory networks. *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, 474(2213):20170844, 2018.
- [47] Zhong Yi Wan and Themistoklis P Sapsis. Machine learning the kinematics of spherical particles in fluid flows. *Journal of Fluid Mechanics*, 857, 2018.
- [48] Zhong Yi Wan, Pantelis Vlachas, Petros Koumoutsakos, and Themistoklis Sapsis. Data-assisted reduced-order modeling of extreme events in complex dynamical systems. *PloS one*, 13(5):e0197704, 2018.
- [49] Jin-Long Wu, Heng Xiao, and Eric Paterson. Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework. *Physical Review Fluids*, 3(7):074602, 2018.
- [50] DZ Zhang and A Prosperetti. Ensemble phase-averaged equations for bubbly flows. *Physics of Fluids*, 6(9):2956–2970, 1994.