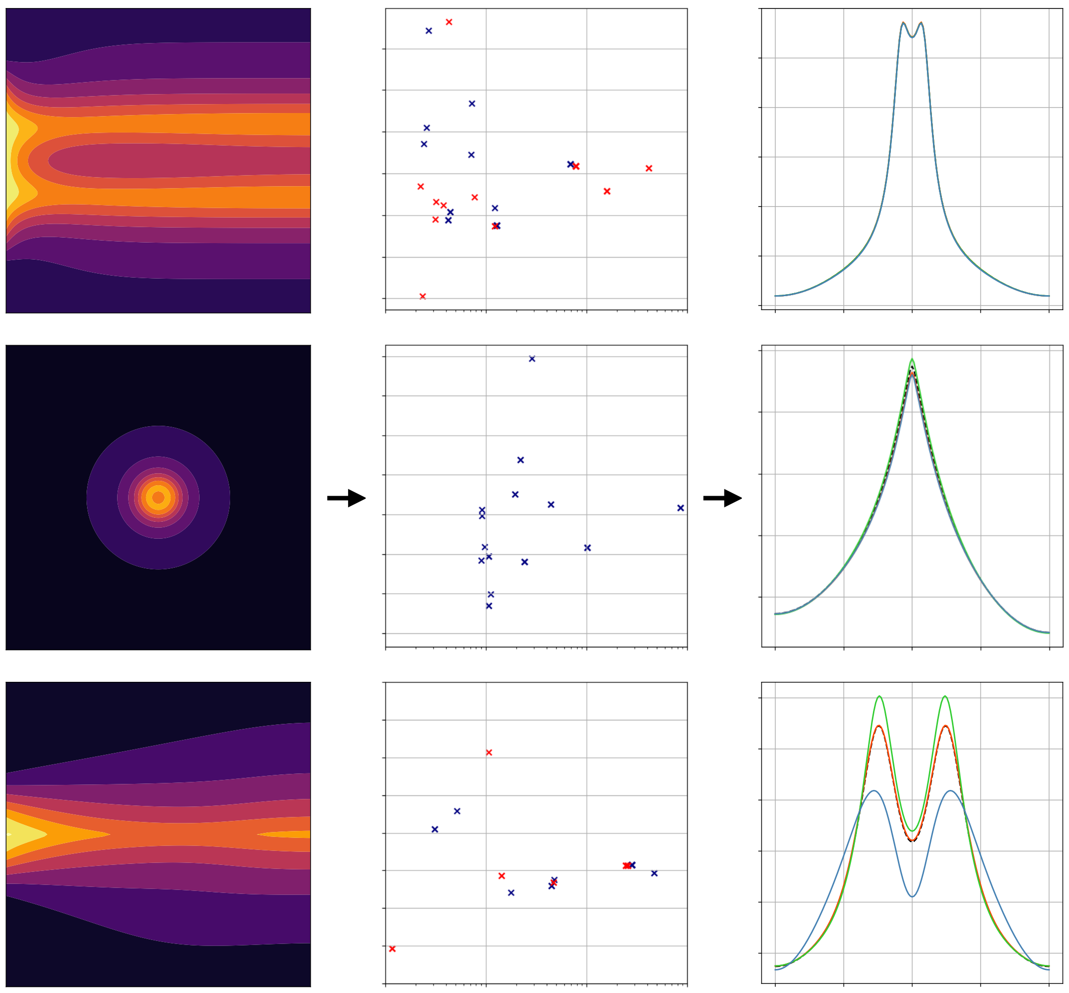
Threshold Ridge Regression for the Purpose of Turbulence Closure Modelling

Testing sparse regression for the possible creation of transport equations for RANS turbulence modelling

J.V. Weijers



Threshold Ridge Regression for the Purpose of Turbulence Closure Modelling

Testing sparse regression for the possible creation of transport equations for RANS turbulence modelling

by

J.V. Weijers - 4540050

A thesis presented for the degree of Aerospace Engineering
To be defended publicly on September 6, 2024 at 13:00 at AE hall G

Thesis Duration: November 13, 2023 - September 6, 2024

Thesis Committee:	Dr. R. P. Dwight,	TU Delft,	Thesis supervisor
	Dr. N. A. K. Doan,	TU Delft,	Chair
	Dr. D. Modesti,	TU Delft,	External member

Version 1.0

Aerospace Engineering
Delft University of Technology
The Netherlands
August 28, 2024

Abstract

Reynolds averaged Navier Stokes (RANS) models are the industry standard when it comes to computational fluid dynamics (CFD). RANS models have notable shortcomings as the Reynolds stress is modelled locally and does not include temporal behaviour. Therefore, current RANS models fail to make accurate predictions in areas of out-of-plane straining, high streamline curvature and significant anisotropic Reynolds stresses.

Data augmented RANS models have shown promise to improve accuracy, with sparse regression and machine learning algorithms being able to train models between high fidelity data and RANS. The current data driven models have problems, as they lack generalisability between data-sets, are difficult to interpret and only provide a local, non-temporal correction or prediction for the Reynolds stresses.

The current work will explore sparse regression routines, in particular PDE FIND, for the purpose of creating a non-local, temporal transport equation from data. Rather than using high-fidelity turbulence data, a self-developed model problem, dubbed the $\psi - \phi$ system, is created to replicate the uncertainties related to turbulence modelling. Multiple candidate libraries for PDE FIND are tested, with the greatest emphasis on libraries lacking a part of the true functional form, known as incomplete candidate libraries.

Incomplete candidate libraries are found to benefit from using highly correlated replacement functions for the missing function with an exception being a candidate library lacking the correct production term. Even if models were trained poorly, PDE FIND was able to identify the most important functions with an accurate coefficient.

PDE FIND could potentially create an universal transport equation to model the residual between high fidelity data and RANS, but only if there are enough highly correlated candidate functions and the production terms are well known or well represented in the candidate library.

Preface

The current work is written as a MSc thesis for a degree in Aerospace Engineering and was conducted from November 2023 till September 2024. It will represent my final chapter at the Delft University of Technology (TU Delft) for now and I am more than glad to do it in the field of turbulence closure modelling.

Since following the course 'Turbulent Flows' during my bachelor's minor at the ETH Zürich as an SEMP (Swiss Erasmus) student, I have had a keen interest in turbulence modelling and I was always eager to explore the subject more. Turbulence modelling was one of my main motivations for choosing the master's degree in Aerodynamics at the Delft University of Technology (TUDelft) and was my main driving motivation for finding the current thesis topic.

There are plenty of people I would like to thank for my time at the TU Delft. My main friends in Delft: Tom, Amber, Reiny, Josh and Herbert gave me plenty of good evenings and made my time one of a kind, while my friends in my hometown of Eindhoven, in particular from 'Cone and Friends', provided me with an unparalleled friendship that has already exceeded more than ten years. In a similar trend, I would like to thank my parents and sister for providing me with their unwavering support and trust in my abilities to study. Moreover, I would like to thank Formula Student Team Delft, one of my main hobbies during my studies. A student society I have been a part of since 2017, first as a part-time engineer in the 18 and 19 year, then as a full-time aerodynamics manager in the 21 year and more recently as an advisor on the development of new cars. Without Formula Student Team Delft, I wouldn't be where I am now and I truly wish them the best in the future.

Regarding my thesis, I would like to thank Dr. Richard Dwight for being my supervisor. His expertise, advice and feedback during the meetings were crucial for my thesis and he challenged me to think critically about my own work while being clear and patient. Furthermore, I would like to thank PhD students Andrea Bettini and Kherlen Jigjid for reading my draft thesis and providing me with concrete and useful feedback. At last, I would like to thank Dr. Anh Khoa Doan and Dr. Davide Modesti for being a part of my Thesis Committee.

Contents

List of Abbreviations

Abbreviation	**Explanation**
ARSM	Algebraic Reynolds Stress Model
CFD	Computational Fluid Dynamics
DHC	Dimensional Homogeneity Constrained
DNS	Direct Numerical Simulation
DSW	Different Search Window
EARSM	Explicit Algebraic Reynolds Stress Model
FIML	Field Inversion Machine Learning
GEP	Gene expression programming
HW	Hasegawa & Mima
KS	Kuramoto-Sivashinsky
LES	Large Eddy Simulation
LSQ	Least Squares
NAN	Not a Number
NASA	National Aeronautics and Space Administration
N/D	Not Determined
NS	Navier-Stokes
ODE	Ordinary Differential Equation
PDDO	Peridynamic Differential Operator
PDE	Partial Differential Equation
PiResNet	Physics Informed Residual Network
PINN	Physics Informed Neural Network
PIML	Physics Informed Machine Learning
RANS	Reynolds Averaged Navier Stokes
RSM	Reynolds Stress Models
RSS	Residual Sum of Squares
SA	Spalart Allmaras
SBL	Sparse Bayesian Learning
SGS	Subgrid-Scale
SpaRTA	Sparse Regression of Turbulent Stress Anisotropy
SST	Shear Stress Transport
STRidge	Sequential Threshold Ridge
TBNN	Tensor Basis Neural Network
TBRF	Tensor Basis Random Forest

List of Symbols

Symbol	Explanation	Unit
Greek symbols		
α_n	Coefficients for tensor products	–
β_ν	Correction in Spalart-Allmaras	–
β_i	Coefficients for regression	–
Γ_ϵ	The destruction term of ϵ	m^2/s^4
Γ_ω	The destruction term of ω	$1/s^2$
δ_{ij}	Dirac delta	–
ϵ	The turbulent dissipation	m^2/s^3
ϵ_ϕ	Dissipation for ϕ	1/s
ϵ_ψ	Dissipation for ψ	1/s
ϵ_{ij}	The turbulence dissipation tensor	m^2/s^3
ϵ_k	The turbulent dissipation for k	m^2/s^3
ϵ_{RSM}	Dissipation term for Reynolds stress models	m^2/s^3
ϵ_ϕ^C	Cross-destruction term for ϕ	1/s
η	Correction in Spalart-Allmaras	–
Θ_{ij}	Matrix consisting of candidate functions	–
θ	Momentum thickness	–
λ	Hyper parameter used in regression	–
μ_f	Potential extra candidate function	–
μ	Hyper parameter for PDE FIND	–
ν	The kinematic viscosity	m^2/s
ξ	Correction in Spalart-Allmaras model	–
ξ_{best}	Best error in regression	–
ξ_{new}	New error prediction	–
ξ_{prio}	A priori error for regression	–
$\xi_{post,\phi}$	A posteriori ψ for regression	%
$\xi_{post,\psi}$	A posteriori ψ error for regression	%
$\tilde{\nu}$	Spalart-Allmaras working variable	m^2/s
ν_T	The eddy viscosity	m^2/s
Π_{ij}	Pressure-strain correlation tensor	m^2/s^3
Π_{RSM}	Pressure-strain correlation tensor for RSM	m^2/s^3
ρ	Density	kg/m^3
ρ_{cor}	Pearson correlation coefficient	–
ρ_R	Hyper parameter in elastic net-regression	–
σ	Turbulent Prandtl number	–
σ_ϵ	Coefficient in $k-\epsilon$ model	–

σ_ω, σ_d	Coefficient in $k-\omega$ model	–
σ_k	Coefficient in the k turbulent transport term	–
σ_y	Standard deviation of the target data	1/s
ϕ	Model function for k	–
ϕ_i	Basis functions for weak form	–
ψ	Model function for the Reynolds stress	–
Ω	Domain	–
Ω_{ij}	Velocity rotation tensor	–
ω	Specific rate of dissipation	1/s

Latin symbols

a_{ij}	Anisotropic Reynolds stress tensor	m^2/s^2
a	Bilinear form in weak formulation	–
B	Source term correction for Spalart-Allmaras model	–
b_{ij}	Non dimensional anisotropic Reynolds stress tensor	–
b_{ij}^Δ	Anisotropic Reynolds Stress correction	–
b_{ij}^R	Correction for R based on Reynolds stress tensor	–
$C_{\nu\epsilon}$	Coefficient for prediction ν_t in the $k-\epsilon$ model	–
$C_{\nu\omega}$	Coefficient for prediction ν_t in the $k-\omega$ model	–
C_D	Coefficient for ϵ in the Prandtl one-equation model	–
C_{RSM}	Turbulent transport tensor for RSM	m^3/s^3
$c_1\, cdots\, c_3$	Dimensional coefficient in ψ equation	s, m^2/s, 1/s
$c_{\epsilon 1}$, $c_{\epsilon 2}$	Coefficients for the $k-\epsilon$ model	–
$c_{\omega 1}$, $c_{\omega 2}$	Coefficients for the $k-\omega$ model	–
c_i	Coefficients for basis function expansion	–
$c_{i,model}$	Coefficients of the model from regression	–
$c_{i,true}$	Coefficients of the true form	–
c_{b1}, c_{b2}, c_{w1}	Coefficients for the Spalart-Allmaras model	–
D_ϵ	Diffusion term for ϵ	m^2/s^4
D_ν	Diffusion term for Spalart Allmaras model	m^2/s^2
D_ϕ	Diffusion term for ϕ	1/s
D_ψ	Diffusion term for ψ	1/s
D_ω	Diffusion term for ω	$1/s^2$
D_k	Diffusion term for k	m^2/s^3
D_{RSM}	Diffusion term for Reynolds stress models	m^2/s^3
D_ν^C	Cross-diffusion term for Spalart-Allmaras	m^2/s^2
D_ϕ^C	Cross-diffusion term for ϕ	1/s
D_ω^C	Cross-diffusion term for ω	$1/s^2$
$d_1 \cdots d_8$	Dimensional coefficient in ϕ equation	s, m^2/s, 1/s
d_tol	Tolerance step in PDE FIND	–
e	Internal energy per unit mass	m^2/s^2
F_h	Heat source	kg/ms^3
F_m	Momentum source	kg/m^2s^2
f_i	Correction in Spalart-Allmaras	–
$f^0 \cdots f^8$	Candidate function used for PDE FIND	–, $1/s^2$, $1/s^4$, $1/m^2$
f_w	Empirical model for Spalart-Allmaras model	–
$\tilde{f}_{\tilde{\nu}}$	Correction in Spalart-Allmaras	–
g	A function related to the derivative of u	–
$\tilde{g}_{\tilde{\nu}}$	Correction in Spalart-Allmaras	–

H^1_Ω	Sobolev space	–
h	Enthalpy per unit mass	m^2/s^2
$I_1 \ldots I_5$	Invariants for the tensor bases	1/s^2, 1/s^3, 1/s^4
k	Turbulent kinetic energy	m^2/s^2
L	Linear form in weak formulation	–
l	Turbulent length scale	m
l_{kol}	The Kolmogorov length scale	m
m	Marker function	
n	Normal direction vector	–
p	Pressure	kg/ms^2
P_ϵ	Production term for ϵ	m^2/s^4
P_ν	Production term for Spalart Allmaras model	m^2/s^2
P_ϕ	Production term for ϕ	1/s
$P_{\phi,1}$	Shear rate driven production term for ϕ	1/s
$P_{\phi,2}$	Squared production term for ϕ	1/s
P_ψ	Production term for ψ	1/s
P_ω	Production term for ω	1/s^2
P_k	Production term for k	m^2/s^3
P_{RSM}	Production term for Reynolds stress models	m^2/s^3
Q	Extra information Candidate functions	–
Q_ϕ	Quartic term for ϕ	1/m^2
q_i	Heat flux in direction i	kg/s^3
R_ϕ	Residual in the ϕ equation	–
Re_L	Integral Reynolds number	–
R	Residual in k-frozen RANS	m^2/s^3
$R_{\tilde{\nu}}$	Correction in Spalart-Allmaras	–
R_i	Residual in the weak form	–
r	Radius	m
S_{ij}	The shear strain tensor	1/s
s	Circumference	m
$\tilde{S}$	Spalart-Allmaras measure for deformation tensor	1/s
T_ϵ	Turbulent transport term for ϵ	m^2/s^4
T_ν	Turbulent transport term for $\tilde{\nu}$	m^2/s^2
T_ω	Turbulent transport term for ω	1/s^2
T_k	Turbulent transport term for k	m^2/s^3
T^n_{ij}	Tensor products for the eddy viscosity	m^2/s^2
t	Time	s
t_{fin}	Final time step	s
t_{kol}	The Kolmogorov time scale	s
tol	Hyper parameter for PDE FIND	–
u_i	Velocity in direction i	m/s
u_{kol}	The Kolmogorov velocity scale	m/s
v	Weak form test function	–
X_{ij}	Matrix consisting of input variables	–
x	Spatial direction	m
x_i	Spatial direction in dimension i	m
y	Spatial direction	m
y_i	Vector consisting of output variables for regression	–
z_i	Z-score per coefficient β_i	–

Math operators and notation

$\bar{\cdot}$	Mean of variable
$\cdot'$	Fluctuation of variable
$\hat{\cdot}$	Regressed variable
$\cdot^s$	Subset of selection
$f(\cdot)$	Function related to variable $\cdot$
$\mathcal{N}(\cdot)$	operator for the momentum equation of the Navier-Stokes

1. Introduction

High fidelity aerodynamic simulations, used in the aerospace, wind, automotive and more hi-tech industries are of importance for making accurate and reliable designs. The current industry standard is Reynolds Averaged Navier Stokes (RANS), a method that is computationally cheap, but inaccurate. For most models, the main cause is the Boussinesq assumption, Boussinesq [1], that creates an analytical, local and non-temporal equation for the Reynolds stresses. The Reynolds stresses are inherently non-local and memory driven with most deficiencies found in the areas that have out-of-plane straining, significant streamline curvature and significant anisotropic Reynolds stresses, Spalart [2], Wilcox [3].

Data driven turbulence has the potential to remedy the inaccuracies of RANS. Tracey et al [4] is one of the first to apply high-fidelity data for RANS modelling by creating corrections for the spatial field. The method was expanded by the group of Duraisamy to form the field inversion machine learning (FIML) method [5, 6, 7], transforming the spatial field discrepancies in terms of the mean flow parameters by using a source to the production term for the Spalart-Allmaras model [8]. Ferrero et al [9] used FIML beyond the original application on airfoils, by using FIML for low pressure gas turbine cascades, and used a sparsity constraint to make more interpretable models.

Franceschni et al [10] used a similar method to FIML, by analysing a source term in the RANS momentum equation and in the Spalart-Allmaras equation, arguing that both corrections work. The group of Volpiani [11, 12, 13, 14] expanded upon the work of Franceschni et al by adding multiple different sources and corrections to the momentum equation and the Spalart-Allmaras-neg model [15], with the main goal of finding the most effective location for data driven corrections. All terms were found to improve the solution, with particular impressive performance found for corrections in the momentum equation, acting as an effective correction to the Boussinesq assumption.

Corrections for the Boussinesq assumption and the Reynolds stress anistropy has seen quite some success in data driven turbulence closure modelling, with the tensor bases neural network (TBNN) of Ling et al [16] being the foundation. TBNN is a neural network driven method using the tensor bases and invariants of the more general effective viscosity hypothesis by Pope [17] (the non-linear Boussinesq assumption), improving the prediction of the Reynolds stress by creating an algebraic Reynolds stress (EARSM) model.

TBNN has seen multiple variations and expansions, with extensions by Cai at al [18] and Fang et al [19] improving the algorithm and discussing the applicability of TBNN. Kaandorp and Dwight [20] used a random forest algorithm rather than a neural network with the same tensor bases of Pope [17], calling it the tensor bases random forest (TBRF) while Xiao et al [21, 22, 23] used a random forest algorithm in their physics informed machine learning (PIML) algorithm, focusing on predictions in the Reynolds stress in the barycentric map, a visualisation concept introduced by Banjeree et al [24]. PiResNet by Jiang et al [25, 26] forced sparsity in the solution by using an L_1 and L_2 penalty in the cost function and Sáez de Ocáriz Borde et al [27] improved interpretability by increasing the traceability of the activation of the neurons.

genetic expression programming (GEP) by Ferreira et al [28] was implemented by Weatherhitt et al [29, 30] for creating a data driven EARSM. An extension is made by Zhao et al [31], where RANS calculations are put in the GEP loop, improving predictions but for a penalty in computational power. Bleh and Geiser [32] used the GEP algorithm by Ma et al [33] being Dimensional Homogeneity Constrained GEP (DHC-GEP), selecting functional forms for the correct dimensions in the augmentations. The DHC-GEP algorithm is improved by Bleh and Geiser [32] by making the variables non-dimensional.

One of the leading methods for data driven corrections in the RANS formulation is the sparse regression of turbulent stress anisotropy (SpaRTA) by Schmelzer et al [34], where a regression method is used to obtain corrections for both the Reynolds stress anisotropy and the k-equation. SpaRTA has seen success in modelling of wind turbine wakes, with Steiner et al [35, 36, 37] expanding the candidate library beyond the Pope tensor bases and Stöcker et al [38] using SpaRTA for particle laden flow, implementing a correction only for the production term. Cherroud et al [39] expanded SpaRTA by using sparse Bayesian learning (SBL) over linear regression. SBL introduces uncertainty windows for the solution that can be used as a tool for the engineer to analyse the uncertainty of a result, while also eliminating hyper parameters associated with regularised linear regression in the process.

In SpaRTA, a predetermined set of candidate functions are used to create a sparse correction using elastic-net regression. Sparse regression techniques have the advantage of being relatively easy to implement for a low computational cost, with PDE FIND by Rudy et al [40, 41] and SINDy by Brunton et al [42] being the earliest baseline works. Multiple studies have shown the advantages of sparse linear regression, such as obtaining the governing equations for drift-wave turbulence equation for plasma by Abramovic et al [43], or finding the macroscopic governing equations of granular flow by Zhao et al [44]. PDE FIND has been extended by Rudy et al to include time varying coefficients [45] and a similar method to PDE FIND is proposed by Schaeffer [46], where an L_1 lasso regularisation is used to enforce sparsity.

Problems with PDE FIND arise with the tuning of hyper parameters and noise, with noise worsening the predictions surrounding derivatives of the data set. Sparse Bayesian learning (SBL) methods, like the method of Zhang et al [47], the S^3d of Yuan et al [48], SubTSBR of Zhang and Ling [49] or the algorithm of Chen et al [50] improves the prediction for noisy data sets and eliminate hyper parameter tuning, but carry a penalty in computational efficiency. The group of Grigoriev [51, 52, 53] and weak-SINDy by Messenger and Bortz [54, 55] uses a weak form formulation for the derivatives and have proofed a high resistance to noise, but the method is still subject to hyper parameter tuning.

More complex methods to find governing equation exist, with the peridynamic differential operators (PDDO) used by Bekar et al [56] from Madenci [57]. A PDDO reduces the effect of noise for sparse regression by using a Taylor expansion on a point, being able to represent it as a function of multiple points. The $\psi-$PDE method by Zhang et al [58, 59] eliminates noise by using high frequency filters and machine learning [58, 59] while the correct governing equation is found using brute force. A kernel filter method is introduced by Joemon et al [60] and found a prediction improvement for noisy data, even though it is not recommended by Rudy et al [41].

Other methods for finding the governing equations, include machine learning methods used in the form of deep learning with convolution kernels used in partial differential equation-net (PDE-net) by Long et al [61], deep learning with physics-informed neural networks (PINNs) by Raissi et al [62] and GEP with the works of Vaddireddy et al [63], Xing et al [64] and Ma et al [33], all able to obtain the governing system from data.

Data driven RANS models lack generalisability between test datasets. Strides have been made to improve generalisability, but full confidence is still lacking. Most of the models, such as SBL-SpaRTA [39], SpaRTA [34], TBNN [16], TBRF [20] and more, are an EARSM model, using the tensor bases of Pope [17] to create a correction for RANS models. The problem with an EARSM is that the Reynolds stress equation is local and does not include temporal effects.

Based on the locality of the correction, an idea is to have the residual of k-frozen RANS from Schmelzer et al [34] to be a transport equation. A transport equation provides information of a variable over a streamline, providing non-locality and includes memory effects. Due to the unknown behaviour of the residual between DNS and RANS, it is unknown what the exact equations need to be to represent DNS in RANS.

The lack of understanding between the true function of DNS and RANS motivates the objective in the current work to analyse sparse regression routines to deduce a transport equation from data. A simplified model problem is made, where the complete functional form is known and is analogous to turbulence modelling and the residual of k-frozen RANS and DNS. The research objective is as follows:

The research objective is to analyse sparse regression routines to obtain a non local expression for a known residual in a partial differential equation by creating a model problem analogous to turbulence modelling

Based on the research objective, a research question is made that focuses on sparse regression and the ability for future work to find a transport equation for the residual with the k-frozen RANS. The research question is as follows:

How can sparse regression provide a trained model for non-local temporal behaviour in a model problem to represent the residual in k-frozen RANS?

The research question can be subdivided in sub questions, as follows:

- *To what extent can a residual be represented by a model problem?*
- *How can sparse regression be used for the purpose of a transport equation?*
- *How reliable is sparse regression for obtaining the complete functional form of a model problem?*
- *To what extent can sparse regression approximate a model with an incomplete set of candidate functions?*
- *Could sparse regression provide insight even if the trained model is incorrect?*

A model problem, called the $\psi - \phi$ model, is created that is analogous to turbulence modelling. Three distinct steady state velocities are tested on a two dimensional square domain, being a vortex case, an exponential case and a shear case. The equations are simulated using FEniCSx and PDE FIND is the regression routine to obtain trained models that are cross validated among one another.

The experiments with PDE FIND will be subject to tests with a complete candidate library and multiple incomplete candidate libraries. The incomplete library seeks to analyse if it is possible to train a reliable, consistent, low error model with the aim to see the effect of unknowns between the RANS and DNS functional form. There are four main incomplete libraries, being a library where a small contribution to the overall solution is lacking, a library where a larger part is lacking, a library where a larger part is lacking with additional functions and a library where the production terms are replaced by highly correlated different production terms.

The current work has shown that PDE FIND is able to return the full ground truth for a set of transport equations for a complete candidate library. For models that do not have the full functional form returned, a near perfect performing model is still found.

For an incomplete candidate library lacking a small contribution to the solution, models perform well on the test data, with PDE FIND being able to find suitable models able to reflect the ground truth. For an incomplete library lacking a larger contribution to the solution, a sparse candidate library lacked the ability to find decent models. With the addition of functions, only one satisfactory model was obtained, where an addition of search windows and highly correlated functions proofed to be crucial. Replacing the production terms by different production terms provided poor results and while the trained models performed decent, the cross validated models performed poorly and no suitable model was found.

Throughout the tests, PDE FIND has been successful in identifying the most important functions of the data, functions with a large contribution to the overall solution and functions with a large amplitude. The coefficients of these functions were mostly accurate, even for poor performing models. The z-score proofed to be a good tool, able to identify the likelihood of terms in the solution and always able to detect the most important functions.

For turbulence modelling, it means that a large, highly correlated set of candidate functions, with an complete as possible invariant candidate collection could potentially result in a viable transport equation for the residual in k-frozen RANS.

The work is structured as follows, Chapter 2 discusses the background of turbulence modelling, system identification techniques and data driven turbulence while Chapter 3 discusses the methodologies used to perform the experiment. Chapters 4, 5 and 6 discusses the results, focusing on data generation, data regression and data validation. The conclusions and recommendations are given in Chapter 7.

2. Background Information and Previous Works

The current chapter is divided in three main sections related to turbulence modelling and data driven models. Section 2.1 provides the basics behind turbulence modelling, describing concepts such as the Boussinesq approximation and describing well-known turbulence models. Section 2.2 describes techniques related to finding the governing equations from data with a specific focus on sparse regression algorithms. At last, Section 2.3 describes data driven turbulence model, and the role data has for creating augmented models.

2.1 Introduction to Turbulence Modelling

The section 'Introduction to Turbulence Modelling' discusses the classic turbulence methods with a deep-dive into Reynolds averaged Navier-Stokes (RANS). The aim of the section is to discuss the historical development of turbulence modelling up to the current day.

In Subsection 2.1.1, the full Navier-Stokes equations are discussed. Subsection 2.1.2 discusses Navier-Stokes modelling and Subsection 2.1.3 discusses RANS in more depth. All work in the section is based on the books of Wilcox [3] and Pope [65]. If other works are used, it will be cited as such.

2.1.1 The Navier-Stokes Equations

The Navier-Stokes equations are the governing equations for aerodynamics and fluid dynamics and for three dimensional flows consists of five equations. These equations consisting of the continuity equation, momentum equation in the x, y and z directions and the energy equation, given as

$$\frac{\partial \rho}{\partial t} + \frac{\partial \rho u_i}{\partial x_i} = 0 \tag{2.1}$$

$$\frac{\partial \rho u_i}{\partial t} + \frac{\partial \rho u_j u_i}{\partial x_j} = -\frac{\partial p}{\partial x_i} + \rho\nu \frac{\partial^2 u_i}{\partial x_j \partial x_j} + F_m \tag{2.2}$$

$$\frac{\partial \rho(e + \frac{1}{2}u_i u_i)}{\partial t} + \frac{\partial \rho u_j (h + \frac{1}{2}u_i u_i)}{\partial x_j} = \frac{\partial}{\partial x_j}\left[\rho\nu\left(\frac{\partial u_i}{\partial x_j} + \frac{\partial u_j}{\partial x_i}\right)\right] - \frac{\partial q_j}{\partial x_j} + F_h \tag{2.3}$$

In (2.1), (2.2) and (2.3), x_i is the spatial direction, u_i is the velocity, ρ is the density, p the pressure, ν the kinematic viscosity , F_m a momentum source, F_h a heat source, e the energy, h the enthalpy and q_i the heat flux.

2.1.2 Navier-Stokes and Turbulence

The non-linear convective terms of (2.2) drive the chaotic nature of turbulence in the momentum equation and provides modelling issues. Multiple methods exist for the simulation of the Navier-Stokes equations, ranging from low-fidelity methods such as panel methods to high fidelity methods such as the direct numerical simulations (DNS). Problems in modelling arise when turbulence is considered. For computational fluid dynamics (CFD), the main three types are DNS, large eddy simulations (LES) and Reynolds averaged Navier-Stokes (RANS), where DNS is the most expensive and RANS is the least expensive method.

The Kolmogorov Equilibrium Theorem

Turbulence can be described as an irregular, unsteady motion that considers a wide range of scales. Rotational motion with varying vorticity is often found in turbulent flows and are described as eddies. The turbulent eddies exist throughout the flow in different (length) scales and can be described to encompass a turbulent motion. In mathematical terms, it can be seen as a collection of different wavelengths of the flow. The cascade process describes the nature of the eddies. When turbulence decays, large eddies transfer their kinetic energy to smaller eddies where the smallest eddies will dissipate into heat through viscosity. The rate of dissipation of the smallest eddies are bounded by the energy transfer of the larger size eddies.

Kolmogorov introduced the universal equilibrium theorem [66], stating that the energy transfer from the larger eddies to the smaller eddies must be equal to the rate of dissipation. Kolmogorov relates the kinematic viscosity ν to the dissipation ϵ, establishing equations for the smallest length, time and velocity scales,

$$l_{kol} \equiv (\nu^3/\epsilon)^{1/4} \tag{2.4}$$

$$t_{kol} \equiv (\nu/\epsilon)^{1/2} \tag{2.5}$$

$$u_{kol} \equiv (\nu/\epsilon)^{1/4} \tag{2.6}$$

In (2.4), (2.5) and (2.6), ϵ is the dissipation and l_{kol}, t_{kol} and u_{kol} are the Kolmogorov length, time and velocity scales, related to the smallest scales in the flow.

Direct Numerical Simulation

DNS is the direct discretisation of the Navier-Stokes equations. For DNS simulations, the smallest discretisation length, is driven by the smallest time and length scales of the flow. A relation can be found with the Kolmogorov scales of (2.4) and (2.5), resulting in a relation for the integral Reynolds number Re_L as

$$\frac{l_I}{l_{kol}} \propto \mathrm{Re}_L^{3/4} \tag{2.7}$$

In (2.7), L is the integral length scale, describing the largest eddie size (most often related to the size of the object of interest). From (2.7), it can be seen that the smallest length scale scales with the Reynolds number as the smallest scale decreases in size with increasing Reynolds number. For DNS to work effectively, the full range of scales, and therefore the smallest eddies, have to be captured, meaning that the amount of mesh cells will increase with increasing Reynolds number. For 3D time dependent DNS simulations, DNS will scale with Re_L^3. An increase in mesh cells, is an increase in computing power and for a 1 % equivalent airspeed increase of a transport aircraft (assuming Re_L=10^7) an increase in the amount of cells of 10^{15} is found.

Some DNS data sets are openly available for use. The national aeronautics and space administration (NASA) has multiple datasets for both compressible and incompressible aerodynamics [1], such as the DNS periodic hill case (Xiao [67]) or the converging-diverging duct (Marquillie [68]). The highest Reynolds number for the DNS of the periodic hill case is around 5600, what is several orders lower than aerospace applications. A simple search allows for more datasets, such as but not limited to, data provided by the KTH royal institute of technology [2] or the John Hopkins institute [3], with a Reynolds number of $O(10^4)$ at most. Figure 2.1 shows an example of a periodic hill LES simulation by Balakumar and Park [69]. The periodic hill is often used in data driven turbulence modelling applications

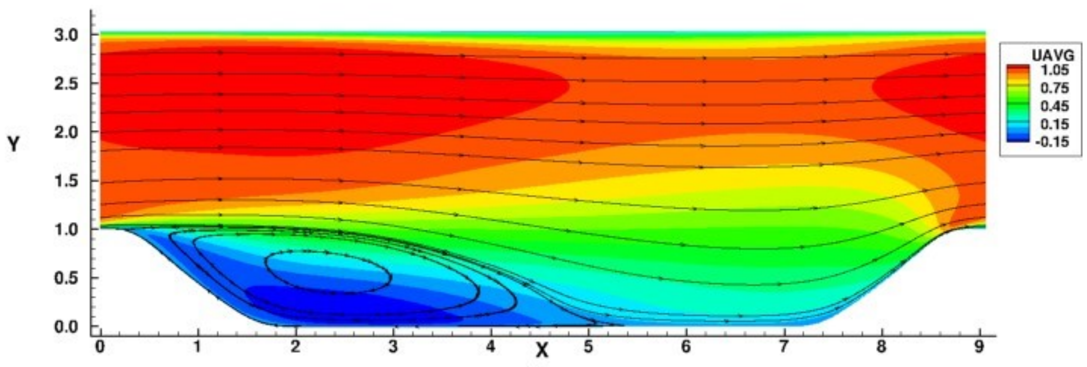


Figure 2.1: Example of the periodic hill case often used in data driven turbulence modelling, with in the current case streamlines over averaged velocity, provided by Balakumar and Park [69]

The use of DNS for high Reynolds purposes in the imminent future is highly unlikely. Moore's law, stating that the amount of computational power doubles every two years, is halting as computer components are reaching size limits related to quantum-and thermodynamics (Waldrop [70], Theis and Wong [71]). Compromises have to be made for aerodynamic simulations.

Large Eddy Simulations

A compromise for accurate simulations is LES. As DNS resolves all scales, LES only fully resolves larger scales, where most of the anisotropy and energy is associated in the turbulence. The smaller scales are modelled by simpler models. For LES, a filtering procedure is performed to split larger scale and smaller scale motions. For the larger

[1]Found at `https://turbmodels.larc.nasa.gov/other_dns.html`
[2]Found at `https://www.flow.kth.se/flow-database/simulation-data-1.791810`
[3]Found at `https://turbulence.pha.jhu.edu/datasets.aspx`

scales, a subgrid-scale (SGS) tensor is obtained that arises from the residual moments. Closure is reached by modelling the smaller scale motions by an eddy-viscosity model, modelling the SGS tensor and coupling it to the larger scale movements in the flow.

Similar to DNS, LES data is widely available as well. Accurate high fidelity LES data is available for data driven turbulence cases, such as the periodic hill by Cinella [72] where the highest case Re is 37000. For industrial purpose, the LES discretisation is still considered too significant and current challenges with LES are found in the setup of LES simulations, needing proper background study and proper research to setup such simulation.

2.1.3 Introduction to Reynolds Averaged Navier Stokes

RANS is computationally the cheapest model of CFD and is considered the industry standard. Due to the cheap nature of RANS, it is able to provide results for a fraction of the time and computational resources compared to DNS. RANS does have a problem with accuracy, as for most models (in particular the Bousinessq assumption driven models) RANS performs poorly in highly transient, highly anisotropic flows.

The Reynolds Averaged Navier Stokes Equations

RANS models are based on Reynolds averaging, where a variable (such as the velocity u_i) is decomposed in a mean $\bar{u}_i$ component and a fluctuating component u_i' and subsequently is averaged. For incompressible aerodynamics, three different types of averaging exist:

- **Time averaging** Averaging a statistically stationary flow at a certain location x_i, n measurements are made at n different times t.
- **Ensemble averaging** Suitable for flows with unsteady behaviour. For n different experiments, measurements are made at a station x_i at the same t per experiment.
- **Phase averaging** Able to capture averages of periodically repeating flow, where n measurements are made for the same phase lag of periodic behaviour.

The averaging procedure is applied to the Navier Stokes equations from Section 2.1.1. For incompressible flows, the Reynolds averaged equations for the continuity equation and the momentum equation are

$$\frac{\partial \overline{u}_i}{\partial x_i} = 0 \tag{2.8}$$

and

$$\frac{\partial \overline{u}_i}{\partial t} + \overline{u}_j \frac{\partial \overline{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \nu \frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j} + \frac{\partial}{\partial x_j}\left(\overline{u_i' u_j'}\right) \tag{2.9}$$

The Reynolds stress $\overline{u_i' u_j'}$ is introduced in (2.9) and is a symmetric tensor with six unknowns for a three dimensional system, resulting in a total of 10 unknowns for the RANS equations (2.8) and (2.9). With five equations, the system is undetermined and needs closure modelling to be able to solve for (2.8) and (2.9).

The Boussinesq Eddy-Viscosity Approximation

The Boussinesq eddy-viscosity assumption (Boussinesq [1]) provides closure modelling for the RANS equations and is a well used assumption to create a relation for the Reynolds stress tensor from (2.9). The basis of the Boussinesq assumptions can be obtained from continuum mechanics and states that the molecular momentum transport fluctuation bare similar behaviour to turbulence, resulting in a mathematically analogous relation to the stress-rate-of-strain relation for a Newtonian Fluid. The Boussinesq assumption follows

$$-\overline{u_i'u_j'} \approx \nu_T \left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right) - \frac{2}{3}k\delta_{ij} = 2\nu_\tau S_{ij} - \frac{2}{3}k\delta_{ij} = a_{ij} - \frac{2}{3}k\delta_{ij} \tag{2.10}$$

In (2.10), ν_t is the eddy viscosity, k the turbulent kinetic energy, δ_{ij} a dirac operator that is one for i equal to j and zero otherwise and a_{ij} is the anisotropic part of the Reynolds stress tensor. The main assumption in the Boussinesq assumption is that the anisotropy of the Reynolds stress tensor is driven by the mean rate-of-strain of the flow S_{ij},

$$S_{ij} = \frac{1}{2}\left(\frac{\partial \overline{u_i}}{\partial x_j} + \frac{\partial \overline{u_j}}{\partial x_i}\right) \tag{2.11}$$

The Turbulent Kinetic Energy Equation

The equation for the turbulent kinetic energy, the k-equation, is an important transport equation often used for closure in turbulence models. The equation is obtained by obtaining a transport equation for the Reynolds stresses. Such a transport equation is obtained, following an averaging and multiplication procedure of the momentum equation of the Navier Stokes (2.2), as

$$\overline{u_i'\mathcal{N}(u_j) + u_j'\mathcal{N}(u_i)} = 0 \tag{2.12}$$

In (2.12), $\mathcal{N}$ is an operator for the momentum equation of the Navier-Stokes. After simplifications, a transport equation for the Reynolds stress tensor can be obtained, given as

$$\begin{aligned}\frac{\partial \overline{u_i'u_j'}}{\partial t} + \overline{u}_k \frac{\partial \overline{u_i'u_j'}}{\partial x_k} &= \overline{u_i'u_k'}\frac{\partial \overline{u}_j}{\partial x_k} + \overline{u_j'u_k'}\frac{\partial \overline{u}_i}{\partial x_k} - 2\nu \overline{\frac{\partial u_i'}{\partial x_k}\frac{\partial u_j'}{\partial x_k}} \\ &+ \overline{\frac{p'}{\rho}\left(\frac{\partial u_i'}{\partial x_j} - \frac{\partial u_j'}{\partial x_i}\right)} + \frac{\partial}{\partial x_k}\left[\nu \frac{\partial \overline{u_i'u_j'}}{x_k} + \overline{u_i'u_j'u_k'} + \overline{\frac{p'}{\rho}u_i'}\delta_{jk} + \overline{\frac{p'}{\rho}u_j}'\,\delta_{ik}\right]\end{aligned} \tag{2.13}$$

Equation (2.13) can be expressed in a term-by-term basis as

$$\frac{D\overline{u_i'u_j'}}{Dt} = P_{RSM} + \epsilon_{RSM} + \Pi_{RSM} + D_{RSM} + \frac{\partial}{\partial x_k}C_{RSM} \tag{2.14}$$

where the terms are defines as

$$P_{RSM} = \overline{u_i'u_k'}\frac{\partial \overline{u}_j}{\partial x_k} + \overline{u_j'u_k'}\frac{\partial \overline{u}_i}{\partial x_k} \tag{2.15}$$

$$D_{RSM} = \frac{\partial}{\partial x_k}\left[\nu \frac{\partial \overline{u_i'u_j'}}{x_k}\right] \tag{2.16}$$

$$\epsilon_{RSM} = -2\nu \overline{\frac{\partial u_i'}{\partial x_k}\frac{\partial u_j'}{\partial x_k}} \tag{2.17}$$

$$\Pi_{RSM} = \overline{\frac{p'}{\rho}\left(\frac{\partial u_i'}{\partial x_j} + \frac{\partial u_j'}{\partial x_i}\right)} \tag{2.18}$$

and

$$C_{RSM} = \overline{u_i'u_j'u_k'} + \overline{\frac{p'}{\rho}u_i'}\delta_{jk} + \overline{\frac{p'}{\rho}u_j}'\,\delta_{ik} \tag{2.19}$$

Equations (2.13) and (2.14) consists out of 22 unknowns, C_{RST} has 16 unknowns and ϵ_{RST} has 6 unknowns. The turbulent kinetic energy k is obtained by taking half of the trace of the Reynolds stress tensor,

$$k = \frac{1}{2}\overline{u_i'u_i'} \tag{2.20}$$

A subsequent transport equation for k is found by taking the trace of the Reynolds stress transport equation (2.13), resulting in

$$\frac{\overline{D}k}{\overline{D}t} = -\overline{u_i'u_j'}\frac{\partial \overline{u}_i}{\partial x_j} - \epsilon + \frac{\partial}{\partial x_j}\left[\nu\frac{\partial k}{\partial x_j} - \frac{1}{2}\overline{u_j'u_i'u_i'} - \frac{1}{\rho}\overline{p'u_j'}\right] = P_k + \epsilon_k + D_k + T_k \tag{2.21}$$

Similar to the Reynolds stress equations, the k-equation (2.21) can be divided in a production term P_k, a dissipation term ϵ, a viscous transport D_k term and a turbulent transport T_k term. The turbulent transport term T_k consists of the triple correlation $\overline{u_j'u_i'u_i'}$ and the pressure fluctuation term $\overline{p'u_j'}$. These correlations are difficult to measure in experiments and are therefore modelled to provide closure for the k-equation. For the Boussinesq assumption, an approximation for the turbulent transport term results in the relation

$$T_k = -\frac{\partial}{\partial x_j}\left[\frac{1}{2}\overline{u_j'u_i'u_i'} + \frac{1}{\rho}\overline{p'u_j'}\right] \cong \frac{\partial}{\partial x_j}\left[\frac{\nu_t}{\sigma_k}\frac{\partial k}{\partial x_j}\right] \tag{2.22}$$

In (2.22), σ_k is a constant based on measurements and the full k-equation, with the Boussinesq assumption is

$$\frac{\overline{D}k}{\overline{D}t} = P_k + D_k + T_k - \epsilon = -\overline{u_i'u_j'}\frac{\partial \overline{u}_i}{\partial x_j} + \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_k}\right)\frac{\partial k}{\partial x_j}\right] - \epsilon \tag{2.23}$$

2.1.4 Closure Modelling in Reynolds Averaged Navier Stokes

The RANS equations from Subsection 2.1.3, (2.8) and (2.9) are not closed, having more unknowns than equations. Multiple different RANS models exist for closure, such as the one- and two-equation models, Reynolds stress models, explicit algebraic Reynolds stress models and more.

One-Equation Models

One-equation models are models with one transport equation. The two main models are the Prandtl one equation model [73] and the Spalart-Allmaras model [8]. The Prandtl one equation model states that the dissipation of the k-equation (Equation (2.23)) becomes a function of the turbulent length scale l, given as

$$\epsilon = C_D\frac{k^{3/2}}{l} \tag{2.24}$$

In (2.24), C_D is a constant obtained from experiments.

The Spalart-Allmaras model is a transport equation for a type of eddy viscosity (stated in literature often as the turbulent field variable), denoted as $\tilde{\nu}$, and still satisfies the Boussinesq assumption (2.10) to predict the Reynolds stresses. The Spalart-Allmaras model transport equation is given as

$$\begin{gathered}\frac{D\tilde{\nu}}{Dt} = P_{\tilde{\nu}} - \epsilon_{\tilde{\nu}} + D_{\tilde{\nu}} + T_{\tilde{\nu}} + D^C_{\tilde{\nu}} = \\ c_{b1}\tilde{S}\tilde{\nu} - c_{w1}f_w\left(\frac{\tilde{\nu}}{d}\right)^2 + \frac{1}{\theta}\frac{\partial}{\partial x_k}\left[(\nu+\tilde{\nu})\frac{\partial\tilde{\nu}}{\partial x_k}\right] + \frac{c_{b2}}{\sigma}\frac{\partial\tilde{\nu}}{\partial x_k}\frac{\partial\tilde{\nu}}{\partial x_k}\end{gathered} \tag{2.25}$$

In (2.25), a production term, destruction term, diffusion term, transport term and a non-conservative diffusion term $D_{\tilde{\nu}C}$ are found and c_{b1}, c_{w1}, c_{b2} are coefficients, while θ is the momentum thickness and σ the turbulent Prandtl number. the function f_w is an empirical function while $\tilde{S}$ is the result of an (empirical) function of the real shear strain rate tensor. For the relations defining f_w and $\tilde{S}$, the reader is suggested to look at the original paper of Spalart and Allmaras [8]. In (2.25), d is the distance to the wall, a measure to make the Spalart-Allmaras model non local.

Two-Equation Models

Two-equation models mostly expands on the k-equation (2.23), by having a second transport equation. The most basic models are the $k-\epsilon$ model by Jones and Launder [74] and the $k-\omega$ model by Wilcox [75], where ω is the specific dissipation rate. Those models provide values for ν_T to use in the Boussinesq relation (2.10), where the relation is obtained for such models using dimensional analyses.

The $k-\epsilon$ model is a two equation model that relates the eddy viscosity ν_T to the turbulent kinetic energy k and the dissipation ϵ following

$$\nu_T = C_{\nu\epsilon}\frac{k^2}{\epsilon} \tag{2.26}$$

In (2.26), $C_{\nu\epsilon}$ is a coefficient. The result for ν_t can subsequently be used to provide a prediction for the Reynolds stress in the Boussinesq assumption (2.10).

The ϵ in the $k-\epsilon$ equation is not the same as the exact ϵ (the dissipation) of the flow. In the $k-\epsilon$ model, empirical argumentation is used, where ϵ is considered as the energy flow throughout the energy cascade rather than the process through the dissipative range. Jones and Launder [74] created the empirical transport equation for ϵ, building it from a production P_ϵ, transport T_ϵ and a destruction Γ_ϵ, resulting in

$$\frac{\overline{D}\epsilon}{\overline{D}t} = D_\epsilon + T_\epsilon + P_\epsilon + \Gamma_\epsilon = \frac{\partial}{\partial x_j}\left[\left(\nu + \frac{\nu_t}{\sigma_\epsilon}\right)\frac{\partial\epsilon}{\partial x_j}\right] + c_{\epsilon 1}\frac{P_k\epsilon}{k} - c_{\epsilon 2}\frac{\epsilon^2}{k} \tag{2.27}$$

For the $k-\epsilon$ model, to reach closure, values should be found for the coefficients: $c_{\epsilon 1}$ $c_{\epsilon 2}$, $C_{\nu\epsilon}$, σ_k and σ_ϵ.

The $k-\omega$ model, is a two equation model where the eddy viscosity is related to the turbulent kinetic energy k and the dissipation per unit turbulent kinetic energy ω. The relation is given as

$$\nu_t = C_{\nu\omega}\frac{k}{\omega} \tag{2.28}$$

In (2.28), $C_{\nu\omega}$ is a constant based on measurements (Wilcox [75]). Similar to the $k - \epsilon$ model, the $k - \omega$ model has a description for ν_t in Equation (2.28) that can be used to find the Reynolds stress with the Boussinesq assumption (2.10). The ω equation uses a similar physics build-up analogy as the ϵ equation, creating an equation with a production term P_ω, a transport term T_ω, a cross diffusion term D^C_ω, a viscous transport term D_ω and a destruction term Γ_ω, combined as

$$\begin{gathered}\frac{\overline{D}\omega}{\overline{D}t} = P_\omega + D^C_\omega + D_\omega + T_\omega - \Gamma_\omega = \\ c_{\omega 1}\frac{\omega}{k}P_k + \frac{\sigma_d}{\omega}\frac{\partial k}{\partial x_i}\frac{\partial \omega}{\partial x_i} + \frac{\partial}{\partial x_j}\left[\left(\nu + \sigma_\omega\frac{k}{\omega}\right)\frac{\partial \omega}{\partial x_j}\right] - c_{\omega 2}\omega^2\end{gathered} \tag{2.29}$$

For the $k - \omega$ model, to reach closure, values should be found for the coefficients: $c_{\omega 1}$ $c_{\omega 2}$, $C_{\nu\omega}$, σ_k, σ_d and σ_ω. Besides, the $k - \epsilon$ and the $k - \omega$ model, a lot more industry applicable two equations models exist, including (but not limited to) Mentor's [76] $k - \omega$ shear stress transport (SST) model or Speziale, Abid and Anderson $k - \tau$ model [77].

Reynolds Stress Models

Equation (2.13) is a direct transport equation for the Reynolds stress. A subset of RANS models, called Reynolds Stress models (RSM) utilise the transport equation form of the Reynolds stress. Compared to one and two equation models, RSMs do not use the Boussinesq hypothesis and correct some of the Boussinesq shortcomings. For the Reynolds stress, diffusion and convection are modelled completely and together with the time-history and non-local transport behaviour, Reynolds stresses should be better represented than with one- or two equation models. RSMs also provides production and convection terms that are able to represent streamlines and curvature problems more naturally and RSMs do not require an a priori assumption of the mean stress rates, resulting in a more realistic behaviour of the flow for sudden rates of changes in the mean strain rate. The reason RSMs are not the industry standard, is that RSMs are difficult to convergence and still need modelling of some terms.

As stated in Section 2.1.3, the Reynolds Stress equation has 22 unknowns, creating an undetermined system that still needs modelling. Three specific terms need to be modelled, being the dissipation tensor e_{RST} (2.17), the pressure strain correlation Π_{RST} (2.18) and the turbulent transport tensor C_{RST} (2.19).

An example of an RSM, is the Launder, Reece and Rodi LRR model [78]. The LRR model uses the Kolmogorov [66] hypothesis for local isotropy, finding a closure approximation for ϵ_{RST}. For the Π_{RST}, the pressure strain term, the pressure fluctuations are split into rapid fluctuations and slow fluctuations and are modelled appropriately. For the turbulent transport term C_{RST}, the pressure correlation term is neglected and a gradient transport process is assumed that is rotationally invariant. For more information please look at the LRR introduction by Launder, Reece and Rodi LRR model [78] and the books of Wilcox [3] and Pope [65].

Nonlinear Constitutive Relation

Algebraic Reynolds Stress Models (ARSM) are non-linear algebraic relations for the Reynolds stresses and eliminate a transport equation for the Reynolds Stress. The classic ARSM equation is obtained by approximating the convective and turbulent transport terms of (2.13) as proportions to the considered Reynolds Stress, giving

$$\frac{D\overline{u_i'u_j'}}{Dt} \approx \frac{\overline{u_i'u_j'}}{k}\frac{Dk}{Dt} \tag{2.30}$$

Equation (2.30) shows the initial assumption of Rodi [79], where the structural form of the Reynolds stresses are self-similar in space and time and the anisotropic rate of change is zero for homogeneous flows. The result is an algebraic nonlinear relation that is able to provide a prediction for the Reynolds stress,

$$\frac{-\overline{u_i'u_j'}}{k}\left[-\overline{u_m'u_n'}\frac{\partial U_m}{\partial x_n} - \epsilon\right] = \overline{u_i'u_k'}\frac{\partial U_j}{\partial x_k} + \overline{u_j'u_k'}\frac{\partial U_i}{\partial x_k} + \epsilon_{ij} - \Pi_{ij} \tag{2.31}$$

Equation (2.31), is implicit as the Reynolds stress occurs both on the left and the right hand side of the equation. Explicit algebraic Reynolds stress models (EARSM) are models that circumvent the implicit nature.

A generalisation by Pope [17] for EARSM is made by creating it as a non-linear expansion for the Boussinesq assumption, resulting in

$$\overline{u_i'u_j'} = \frac{2}{3}k\delta_{ij} + k\sum_{\lambda}\alpha_n T_{ij}^n \tag{2.32}$$

In Equation (2.32), also known as the effective viscosity hypothesis or the non-linear generalisation of the eddy viscosity assumption, T_{ij}^n are tensor products and α_n are coefficients. The tensor products are polynomial combinations, build up from the strain rate tensor (2.11) and the rotation tensor

$$\Omega_{ij} = \frac{1}{2}\left(\frac{\partial\overline{u_i}}{\partial x_j} - \frac{\partial\overline{u_j}}{\partial x_i}\right) \tag{2.33}$$

The coefficients in (2.32) can be functions of the invariants. The invariants are, similar too the tensors as a function of the strain rate tensor (2.11) and the rotation tensor (2.33). Due to the Cayley-Hamilton theorem, (2.32) is not an infinite summation, being able to express the infinite summation by a finite set of tensor bases and invariants. For two dimensional cases, three tensor products and two invariants are enough to represent the summation and for three dimensional cases, ten tensor products and five invariants are enough. The full set of tensor products and invariants can be found in Appendix A.

2.1.5 Deficiencies of Current RANS Models

Most practical models are driven by the Boussinesq assumption, as explained in previous sections. For the Boussinesq assumption, problems are found with flows that have sudden changes in the mean strain rate. The Boussinesq assumption starts to fail for flow cases where out of plane straining, significant streamline curvature and significant anisotropic Reynolds stresses are present.

In 2015, Spalart [2] published an article, stating the current state of RANS turbulence modelling, noting current challenges and developments. Spalart argues that the Reynolds stress in the current RANS frameworks are modelled locally, evaluating the mean values and derivatives in one point even if along a streamline, a particle can be influenced by preceding eddies of finite size. In time averaging, a sample can be averaged that includes highly rotational flow and irrotational flow, averaging multiple states together and losing information. Highly virtual quantities of the Reynolds stress that are affected by non-local values and phenomena are treated locally by Reynolds averaging and the Boussinesq assumption. Current models such as the $k-\epsilon$ model do have a region of influence and show some level of non-locality, as the diffusion equation affects multiple values in the flow field, resulting in a local equation where the mature solution field is not local.

Spalart [2] states that two strategic decisions can work against the locality of models. The Spalart Allmaras model [8] and the Mentor $k-\omega$ SST model [80] use a wall distance in the model to obtain the relative location of a point while another strategy is to use higher order derivatives. Higher order derivatives do not occur naturally in the Reynolds stress equation (2.13), but can be used empirically to force non-local effects. Spalart believes that a promising direction in the future are detached eddy simulations (DES), also known as hybrid RANS-LES methods, where the more mature smaller turbulent structures that have more repeatable behaviour are in line with the Boussinesq assumption and are modelled by RANS, but RANS-LES methods have their own challenges, such as the communication at the interface between RANS and LES, better know as gray-zoning (Spalart et al [81]). In theory, a Transport equation, such as the RSM models, could provide non-local behaviour due to the time and advection properties of the equation. As seen with RSM models, it would need proper modelling to reduce the inaccuracies.

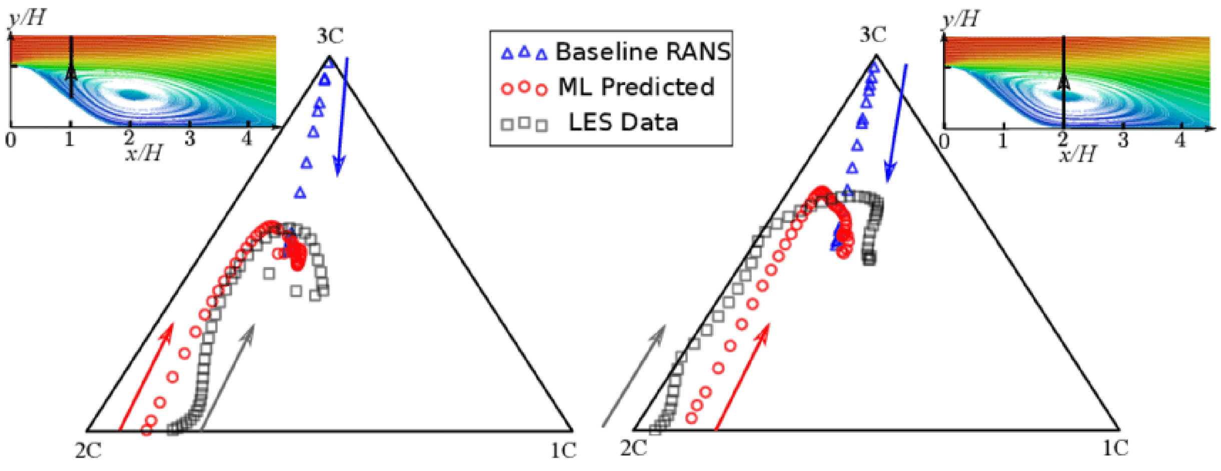


Figure 2.2: Barycentric map comparing $k-\epsilon$ RANS, Machine leaning augmented RANS and high fidelity LES at two specific locations of the periodic hill case. Picture taken from Duraisamy et al and Wang et al [82, 22]

The Barycentric map, a method of visualisation introduced by Banjeree et al [24], is a triangle that at the edges show one-, two- or three component turbulence at a certain spatial location, resulting from the eigenvalues from the Reynolds stress tensor. In Figure

2.2, the deficiencies of RANS are clear. RANS (the $k - \epsilon$ model) and the high fidelity LES data have a significantly different response in the Barycentric map. Machine learned work from Wang et al [22] provides an augmentation to RANS.

In general, Figure 2.3 shows the fidelity versus computational cost of all methods. In CFD, RANS is currently the industrial standard but has quite a low fidelity. DNS has the highest fidelity, but has the greatest costs. A lot of development is performed in Hybrid RANS/LES (DES) methods, not discussed in the current report, and have some promise. Another field that is interesting is the data driven RANS models, where high fidelity LES or DNS data is augmented in the Reynolds stress.

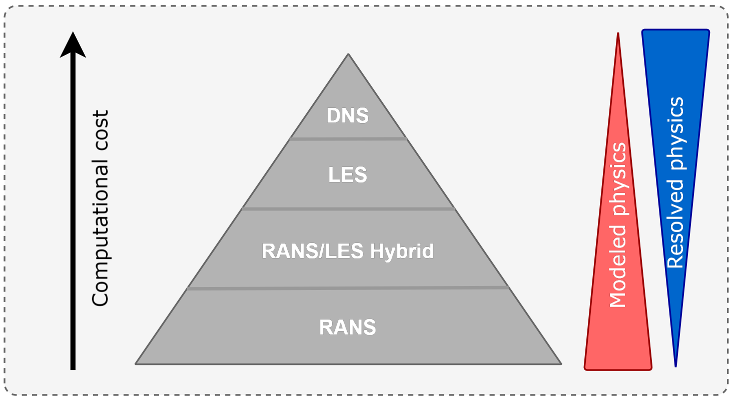


Figure 2.3: Summary of the accuracy and computational time of the most important simulations for the Navier-Stokes equation. Picture is inspired by Xiao and Cinella [83]

2.2 Identification of Dynamical Systems

The governing equations of dynamical systems, like the Navier-Stokes equations, are derived via conservation and physics laws. Due to the increase in data over the past decade, new techniques have arose for obtaining governing equations. The current chapter discusses (regression) techniques used for discovering partial differential equations.

In Subsection 2.2.1, the basics of regression are discussed, introducing least squares regression and shrinkage methods for least squares regression. Subsection 2.2.2 discusses the sparse linear regression for PDE discovery while Subsection 2.2.3 discusses different methods for the discovery of governing equations.

2.2.1 Data Regression and Techniques

The subsection 'Data regression and techniques' is based on the book by Hastie [84] and provides the basics behind least squares regression and several shrinkage methods. Regression enables finding suitable models from data, using training data to train models that can be tested for performance on testing data.

Basics of Least Squares Regression

The most classic regression algorithm is the least-squares algorithm. Least-squares regression aims to minimise the residual sum of squares (RSS) between a predictor and the regressor. For a data set of outputs y_i with input x_i, for all i, a β coefficient is select to minimise the residual sum as

$$\text{RSS}(\beta) = \sum_{i=1}^{N} (y_i - X_{ij}^T \beta_j)^2 \tag{2.34}$$

From (2.34), a unique solution for the β coefficients can be found, given as

$$\hat{\beta}_i = (X_{km} X_{ki})^{-1} X_{jm} y_j \tag{2.35}$$

Least Squares regression creates a linear model to approximate regression as

$$\hat{y} = x\hat{\beta} \tag{2.36}$$

where an output $\hat{y}$ is formed based on a data point x and a regressed weight $\hat{\beta}$.

The problem of least squares regression is that by minimising the residual sum of squares for a certain training set, any candidate function in X_{ij} passing through the solution can become a part of the solution. The result is that a poor predictor can be established that works well for a specific set of training data, but poorly on test data not included in the training data.

Shrinkage Methods for Least Squares

Shrinkage methods can help the problem of least squares regression, by adding a regulator (penalisation term). The result is a more sparse or less correlated result that can improve prediction on test data not included in the training set.

One such shrinkage method is ridge regression (introduced by Hoerl and Kennard [85]). Ridge regression adds an L_2 constraint on (2.34), resulting in the form

$$\hat{\beta}_i = \underset{\beta}{\text{argmin}} \left\{ ||(y_j - X_{ij}\beta_i)||_2^2 + \lambda ||\beta_i||_2^2 \right\} \tag{2.37}$$

In (2.37), λ is a hyper parameter that controls the amount of shrinkage of the values, where the greater the value of λ, the greater the amount of shrinkage. Ridge regression will still result in a unique solution and can be rewritten in similar form like (2.35) as

$$\hat{\beta}_i = (X_{km} X_{ki} - \lambda \delta_{im})^{-1} X_{jm} y_j \tag{2.38}$$

In (2.35), it is clear that λ functions as a scaling parameter, resulting in β for ridge regression being a scaled version of β from least squares.

Lasso is similar to ridge as it provides a constraint to the RSS least squares (2.35). Lasso, introduced formally by Tibshirani [86], constraints the equation by providing an L_1 constraint, resulting in

$$\hat{\beta}_i = \underset{\beta}{\text{argmin}} \left\{ ||y_j - X_{ij}^T \beta_i||_2^2 + \lambda ||\beta_i||_1 \right\} \tag{2.39}$$

The L_1 constraint differs to ridge regression in that Lasso forces values of β to 0 for a defined λ. Lasso effectively shifts the least squares β solution, truncating the solution at 0. For that reason, no closed form similar to (2.38) and (2.35) is possible.

Best subset regression (following Hastie [84]) finds for each set of coefficients, the subset of coefficients that gives the smallest residual sum of squares. Effectively, best subset selection will force certain coefficients to zero as a hard threshold. For a large set of coefficients, best subset selection is costly, and more effective procedures exist such as forward- and backward stepwise selection. In forward stepwise selection, the model starts with the intercept (constant value y_0) and adds progressively with the predictor that improves the fit, while backward stepwise selection starts with the full model and progressively eliminates the worst predictor to the model.

The effect of ridge, lasso and best subset selection on a certain coefficient β is best visualised in Figure 2.4. In Figure 2.4, best subset is seen to act as a hard threshold, where the coefficients are forced to zero if the coefficient becomes to small. For ridge, the slope of the coefficient becomes less, resulting in lower regressed coefficients while for lasso a translation of the curve is seen, forcing an earlier onset to zero of the regressed coefficient.

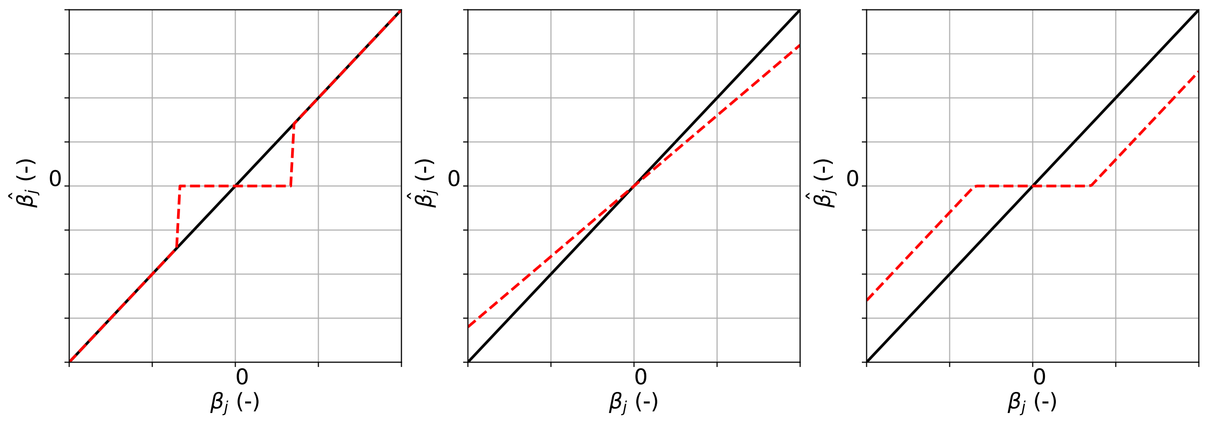


Figure 2.4: Picture visualising the procedure of shrinkage methods with on the left the best subset method, in the middle the ridge regression and on the right lasso regression, inspired by Hastie [84]

Besides best subset, lasso and ridge regression, Hastie [84] summarises a lot of different shrinkage methods to improve the predictive capabilities of regression, such as forward-stagewise regression, least angle regression, principal component regression and partial least squares.

2.2.2 Regression and Dynamical System Discovery

The regression routines in the previous section, results in different predictors for β based on the type of regularisation and hyper parameters used. In recent years, regression and regularised regression have been used for the act of system identification, using data to obtain the governing equations for physical systems.

Sparse Regression for Data Driven Discovery

The group of Brunton introduced the Sparse Identification for Non-Linear Dynamics (SINDy) algorithm [42]. SINDy is aimed at obtaining the important terms from data of ordinary differential equations (ODE), in the form of

$$\frac{d}{dt}x_i(t) = f(x_i(t)) \tag{2.40}$$

In (2.40), $f(x)$ is a specific function. A sequential threshold least-squares algorithm drives parsimonious solutions, where per iteration a least squares step is performed and small coefficients lower than a certain threshold are set to zero. After an initial selection, a new least squares operation is performed until a convergence of the non-zero values are found. Brunton et al claims that the sequential threshold least-squares performs better for sparse solutions than L_1 regularisation (such as the Lasso Subsection 2.2.1) as L_1 regularisation might become expensive for large data sets.

SINDy uses a predetermined set of candidate functions that allows for possible solutions for the governing equation. Candidate functions are all the possible functions that an equation can become and for regression based purposes have to be defined by the user before hand. Candidate functions can be any type of function, a constant,a polynomial and more, resulting in the system

$$\frac{d}{dt}x_i(t) = \Theta_{ij}(x)\beta_j \tag{2.41}$$

In (2.41), $\Theta_{ij}(x)$ is the matrix consisting of the candidate functions for x and β is the (sparse) coefficient vector. The sparse regression routines allow for the candidate functions to span beyond the ground truth functional form as the sparsity is aimed to limit the bias related to a specific training data point, reducing the amount of used functional forms.

Rudy at al (Same group as Brunton) introduced the PDE FIND algorithm [40, 41]. Rudy et al expands upon SINDy by making it able to obtain the (general form) of PDEs from data of the form

$$\frac{d}{dt}u(t) = f(u, \frac{\partial u}{\partial x_i}, \frac{\partial^2 u}{\partial x_i \partial x_i}, \dots, x, \mu_f) \tag{2.42}$$

In (2.42), μ_f is an additional function, not driven by u. Similar to SINDy, PDE FIND uses a selection of pre-determined (non-linear) candidate functions. The candidate functions are populated in a linear system given as

$$\frac{d}{dt}u_i(t) = \Theta_{ij}(u, Q)\beta_j \tag{2.43}$$

In Equation (2.43), Θ_{ij} consists of candidate functions of the variable of interest u and Q. Q are candidate functions that provide additional information but are not a function of u. An example of a vector system with candidate functions with an additional function not related to the variable of interest in the form of x is given as

$$\boldsymbol{u_t} = \left[\mathbf{1}\ \boldsymbol{u}\ \frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\ \frac{\partial^2 \boldsymbol{u}}{\partial \boldsymbol{x}^2} \dots \boldsymbol{u^2}\frac{\partial \boldsymbol{u}}{\partial \boldsymbol{x}}\ \boldsymbol{x}\ \boldsymbol{x^2}\right]\boldsymbol{\beta} \tag{2.44}$$

In the PDE FIND algorithm by Rudy et al, an L_0 penalty is introduced that adds error to a non sparse solution, seen in Equation (2.45). The idea is that the addition of an additional term, should carry a greater decrease in prediction error than the L_0 penalty adds to the error. Before the error analyses, a sequential threshold ridge regression step (STRidge in the works of Rudy et al) is performed. STRidge is similar to the threshold least squares of SINDy, with ridge regression used instead of least-squares regression to reduce the correlation between variables and force coefficients towards lower values before cut-off. The term β is evaluated with STRidge and the resulting error is estimated using the L_0 penalty. A new updated threshold tolerance for STRidge is found and the same sequence is performed until convergence. The effective relation for PDE FIND is given in

$$\hat{\beta}_i = \underset{\beta}{\operatorname{argmin}} \left\{ \left|\left| \Theta_{ij}(u,Q)\beta_i - \frac{d}{dt}u_j(t) \right|\right|_2^2 + \mu \left|\left|\beta_i\right|\right|_0 \right\} \tag{2.45}$$

In the paper of Rudy et al [40], multiple equations like the Korteweg-De Vries and the Navier-Stokes equations are discovered from data. The PDE FIND algorithm provides good correlation for no noise. For situations with noise, the results are varying. For the non-linear Schrödinger equation, PDE-FIND is able to find the underlying partial differential equation with reasonable error while for the Kuramoto-Sivashinsky equation there is a significant error of more than 50%. In the case of the Navier-Stokes equation, PDE FIND is able to return all coefficients associated with the PDE within 0.2-1% of the original value while for 1% noise (defined as 1% of the standard deviation of the solution function), an offset of 6-7% is present. Rudy relates the sensitivity to noise for the sparse regression routine due to the discretisation error imposed for derivatives.

Rudy et al created an improved algorithm where time dependent constants can be classified with the sparse regression algorithm. In [45], the aim is to find the underlying PDEs in the form of

$$\frac{d}{dt}u(t) = f\left(u, \frac{\partial u}{\partial x_i}, \frac{\partial^2 u}{\partial x_i \partial x_i}, \ldots, x, \mu_f(t)\right) \tag{2.46}$$

The result is that the sparse regression coefficients β will have a time dependency, creating $\beta(t)$. The data is distributed in 'bins' for specific time stamps and for each subset a PDE prediction is made.

At a similar time to the group of Brunton, Schaeffer [46] found a different numerical algorithm based on sparse regression to recover PDEs from data. Schaeffer approaches the problem similar to Rudy et al, creating and populating candidate functions, aiming to solve a system of Equations of form (2.43). An L_1 regularised least-squares minimisation is chosen similar to Lasso as presented in Section 2.2.1 that forces sparsity. Similar to Rudy et al [40], the method is able to capture the governing partial differential equations based on data and is stated to be robust to noise and data size.

PDE FIND and SINDy has been successful beyond [40] and [42]. A few examples, but not limited to, include Jin et al [87], who used numerical data to obtain the governing PDEs behind the heat conduction equation and the conductive–convective heat transfer equation. Abramovic et al [43] was able to identify the drift-wave turbulence equations based on the Hasegawa & Mima (HW) and modified HW models for tokamak plasma. Zhao et al [44] used SINDy and STRidge to find ordinary differential equations successfully for the macroscopic governing equations of granular flow. Zhang et al [88] was able to obtain macroscopic PDEs from molecular simulation using discrete simulation Monte Carlo. In the work of Zhang et al [88], the data-set is independent of the governing equations a priori and PDE-FIND was able to return the macroscopic PDEs and provide accurate predictions of the transport coefficients.

Noise Mitigation for Sparse Regression

A problem with the PDE-FIND algorithm by Rudy et al [40] is the robustness to noise. The evaluated candidate functions in (2.44) consists of derivatives that are deduced from data. In the additional information to PDE FIND, Rudy et al [41] uses finite difference or polynomial interpolation for higher order terms, methods using adjacent points. As the adjacent points change with noise, the effect is translated on the calculated derivatives and the derivative becomes more inaccurate with noise. Rudy et al [41] stated a few techniques that are aimed to increase the robustness to noise. A Gaussian smoothing Kernel was found to remove noise from the derivatives, but it added a bias that created problems with the identified dynamics, and is subsequently not recommend by Rudy et al. A similar conclusion is found for Tikhonov differentation. Rudy et al[41] found that a polynomial interpolation of multiple data-points improves the accuracy related to noise, but does that it is not perfect.

On the contrary to the work of Rudy et al, the work by Schaeffer [46] concludes a decent robustness to noise. For multiple cases, derivatives are constructed with a spectral method and pre-processing and regularisation is used for finite differences and finite elements to ensure smooth derivatives. Inaccurate results are found after 3% of noise. It can be argued, however, that the robustness to noise levels of 3 %, is still on the low side.

The group of Grigoriev [51, 52, 53] uses the weak formulation of the PDE for the sparse regression problem,using the weak form for the candidate terms. The weak form is obtained by multiplying each term with a test function and integrating it over a domain, meaning that a certain function should hold over a domain rather than at a certain point (also known as the hard form). The regression algorithm is a thresholding algoritihm similar to SINDy [42], where all small values in a least squares approach are set to zero under a certain tolerance. The weak formulation has an advantage that it can reduce the order of the derivatives using integrating by parts. The noise constraint is then moved from the derivative of the variable of interest (the trail function) to the test function, reducing the dependencies on accurate derivatives. For the test function, exponential test functions are chosen and a specific statement is made that carefull considerations should be made for which test function to choose. The weak formulation, in the form presented by the group of Grigoriev, has given improvement in finding the governing equations of dynamics with noise, increasing the robustness to noise up to levels of 30%.

Messenger and Bortz [54, 55] introduced the weak SINDy method. Similar to the work of Grigoriev, the weak SINDy method is able to obtain high accuracy for high levels of noise, by eliminating the pointwise derivative. A sparse solution is found using a modified sequential-thresholding least-squares algorithm, where an additional constraint on the thresholding is present to provide a better balancing of the candidate terms. Different to the work of Grigoriev, weak SINDy uses the convolutional weak form of the PDE, where a set of test functions are used that are a translation of another test function. For a greater understanding on the convolutional weak form, the reader is recommended to read the original work by Messenger and Bortz [54, 55].

Bekar at al [56] improves robustness to noise for sparse regression by a peridynamic differential operator (PDDO). Peridynamics presents a non-local representation of the scalar field by accounting for the effect and the interaction with other points. a PDDO is constructed by means of a Taylor series expansion around the scalar point of interest, allowing to represent a derivative of with a certain dimension at a certain point x in terms of multiple points in a space (for more on PDDO, please read the book of Madenci et al [57]). Bekar at al found that the implementation of the PDDO increased the resistance against noise, being able to find the Kuramoto-Sivashinsky equation within 0.3% at a noise level of 50%.

Joemon et al [60] proposes a smoothing approach using a 2D Gaussian kernel filter. The filter computes a smoothed value at each point as a weighted average of the surrounding points. After smoothing, an L0BnB method by Hazimeh et al [89] is implemented to perform sparse regression. Results from [60] shows improved performance under noise of both the heat equation and the Burgers equation.

The work by Joeman et al is in conflict with what is stated by Rudy et al [41], where Rudy states that caution should be applied for smoothing differential values with kernel functions. It should also be noted that the PDE FIND algorithm in [60] is not able to fully return the Heat equation. The work of [40] is able to return the full governing equations for multiple (similar and more complex) PDEs and no reasoning is provided by Joeman et al [60] why there is a discrepancy present. Some reasons might be that numerical noise due to discretisation doesn't favour the PDE FIND algorithm or that the incorrect selection of hyperparameters are used.

2.2.3 Different Techniques for System Indentification

The previous sections discussed data driven discovery techniques in the field of regression. Multiple techniques beyond sparse regression exist to obtain the governing dynamics from data, such as sparse Bayesian learning (SBL), neural networks, gene expression programming (GEP) and more.

Sparse Bayesian Learning for Data Driven Discovery

Sparse Bayesian learning (SBL), introduced by Zhang et al [47] for system identification, is a different technique than sparse regression for the parsimonious identification of the governing equations. SBL is focused around assigning coefficients in (2.41) with independent Gaussian priors around a mean of zero. A posterior Gaussian distribution and a most probable point estimate is found by maximising the log-likelihood. Most of the coefficients will go towards infinity at zero, hence promoting sparsity.

Zhang et al [47] introduces sparse Bayesian regression for model identification. SBL has the advantage of providing error bars and quantify uncertainties in the data-driven discovery process. In the article, Bayesian regression is used for ODE's and the claim is made that the SBL method by Zhang et al [47] is able to outperform symbolic regression methods such as Lasso and STRidge (Rudy et al [40]) with tests performed for the shallow water equation and the Navier Stokes equations. Nayek et al [90] used spike-and-slab priors in comparison to the relevant vector machine priors of [47] to promote sparsity. The spike-and-slab priors were tested on ODEs and found competitive results compared to relevant vector machine models. An overall less in-depth study by Fuentes et al. [91] (from the same group as Nayek et al, published around the same time as [90]) is performed for non linear first order differential equations. A comparison is made with real life experimental data and similar to Zhang et al [47], encouraging results are found. One of the main advantages of the work of Zhang et al [47], is the tuning of hyperparameters. In the SBL method, the hyperparameters are tuned automatically, while for Rudy et al PDE FIND [40] a trail and error approach must be employed that can lead to an inaccurate result.

One of the problems with the SBL method of Zhang et al [47], is the efficiency of the cost function. Yuan et al [48] proposes the S^{3}d model that has a more efficient cost function. The S^{3}d model is able to deduce the governing PDE from spatiotemporal data from high-quality experimental data and has the ability to provide adequate results for data with noise, by applying polynomial interpolation for derivatives. The method is also used with real data, for the Complex Ginzburg-Landau Equation (travelling wave experiment) where a good representation of the governing equations to recreate the experiment are found.

Zhang and ling [49] expand on Zhang et al [47] by introducing the SubTSBR model. The SubTSBR is designed to tackle high noise and outliers, by using sub sampling methods. The model was tested on four models, being the predator-prey model with noise, shallow water equations with outliers, heat-diffusion equation with a random initial and boundary condition and fish-harvesting problem with bifurcations. SubTSBR is able to provide an improvement on the previous SBL methods, but is subject to proper tuning of the sub sampling method, what is arguably against one of the purposes of Zhang et al [47] as the article advocates against the tuning of (hyper) parameters.

Chen et al. [50] created a SBL algorithm that is able to discover variable coefficients, for both spatial and temporal dependency, as a competitor to the sparse regression algorithm for varying coefficients by Rudy et al [45]. In the work, a Lasso operator with spike-and-slab priors is used (for more information please see [92]) and improved computations by implementing a threshold for the coefficients. The uncertainty, key to Bayesian processes, is implemented as a selection criteria, which is said to outperform selection procedures. The algorithm in [50] is compared to PDE FIND and has shown to outperform for large noise and provided a lower false discovery.

Neural networks for Data Driven Discovery

Physics-informed neural networks (PINNs) are a popular method for physics related data science. In early works leading up to PINNs, physical laws are described in Gaussian process priors for Bayesian regression with an aim on predicting future time predictions [93]. The process got expanded by incorporating non-linear partial differential equations, introducing the capability to infer the governing equations from data, Raisse et al [94], resulting in the paper of PINNs by [62]. In PINNs, neural networks are used that contain physics information (such as conservation laws) to obtain governing equations. In the original paper, PINNs are used to identify the governing equations from data and are able to identify different governing equations, even tested with 1 % noise.

PDE-Net is introduced by long et al [61], a feed-forward deep network that has the goal of accurately predicting dynamics of complex systems and uncover underlying PDEs. PDE-Net uses convolution Kernels to learn linear differential operators of a system and uses Machine learning methods to approximate non-linear responses. PDE-Net only needs minor information on the nonlinear response function, to create an accurate representation. PDE-Net is able to find both linear and non-linear governing equations of observed dynamics, even in a noisy environment.

Berg and Nyström [95] used deep learning practices to discover PDEs from complex data sets from measurement data. In [95], it is stated that a second order non-linear PDE can be approximated by an ODE where neural networks work well. The ODE is able to accurately represent the training data and predict the future (within a certain time domain), for a fraction of the cost. Berg and Nyström states the importance of data quality, and that data transformations, such as coordinate transformations are of great importance for machine learning.

The ψ-PDE and it's Derivatives

A combination of previous methods, Zhang et al [58] proposes the ψ-PDE method in an effort to mitigate the effects of noise and the use of hyperparameters, one of the limiting factors in sparse regression techniques such as PDE FIND [40, 45]. ψ-PDE starts with a neural network that filters noise and the filtered values are processed based on the candidate functions. With a discrete Fourier transform, the values are transformed to the frequency domain where additional values are cut to only preserve low frequency components. Sparse regression is performed in the frequency domain in a progressive manner using a sort of best subset selection, where all possible combinations are compared for the lowest error. The best subset selection algorithm is expensive due to the search requirement and an updated version is presented from the same group [59], where the best subset selection sparse regression routine is replaced by an SBL method, with the added benefit that uncertainties are quantified for each discovered term.

The cut-off and the filtering of high frequency responses could poise challenges in the works of Zhang et al [58, 59]. From Rudy et al [41], one of the issues with filtering, is that higher frequency terms could be important for the solution and could lead to a loss of information, leading that the ψ-PDE method might have issues where the true response is a high frequency response.

Gene Expression Programming for physical law discovery

Gene expression programming (GEP) is a popular form of regression, where the forms of functions and their corresponding coefficients are learned concurrently. Introduced by Ferreira [28], the essential part of GEP is similar to natural evolution, where GEP is a tree structure that learns and adapts by changing the sizes, shapes and composition of the tree structure based on the available data.

Vaddireddy et al [63] used GEP to find physical laws represented by linear and non-linear PDEs. GEP is proven to retrieve the model from data, but with less accurate coefficients than Rudy et al [40] PDE FIND. Compared to PDE FIND, GEP was also more computationally expensive and has more hyper parameters to tune. The advantage of GEP is that a candidate library is not needed and explores past that limitation. Xing et al [64] used GEP for molecular simulations (simulations not based on the original PDE) to find the physical system and was able to retrieve well known relations.

Ma et al [33] introduced the dimensional homogeneity constrained GEP (DHC-GEP). The dimensional homogeneity is added via a verification process per step in the GEP algorithm by checking the dimension of the terms, leaving the major algorithm the same and keeping the advantage of the original GEP algorithm. DHC-GEP is able to recover physical principles from data for cases including the diffusion equation and the voriticity equation and has three improvements over the original GEP, being the robustness to the size and noise of data sets, less sensitivity to hyperparameters and lowering the computational costs. Similar to Xing et al [64], DHC-GEP is able to recover the correct equations for molecular simulations.

2.3 Turbulence and Data Driven Methods

Turbulence modelling has always been data driven to a certain extent. RANS Boussinesq models such as $k-\omega$ and $k-\epsilon$ from Section 2.1.4 have been developed empirically, using data to obtain coefficients needed for closure and making analogies for transport functions based on observations.

The section 'Data Driven Turbulence' expands on that notion, by discussing RANS augmentations created using high-fidelity data of DNS and/or LES simulations, starting with the inadequacies of RANS and the problems and challenges of data augmented models in Subsection 2.3.1. Subsection 2.3.2 discusses machine learning and data driven turbulence while Subsection 2.3.3 discusses symbolic regression and turbulence modelling. The chapter ends with a comparison and summary between the most important methods in Subsection 2.3.4

2.3.1 Data Driven RANS Strategies and Problems

The earliest uptick in data driven turbulence research started 18 years ago (Duraisamy [96]) and has since seen a significant amount of contributions. Data driven techniques aim to reduce the modelling errors and uncertainties and provide potential solutions, but challenges are present and difficult.

Modelling Errors Found in RANS

The review paper by Duraisamy et al [82], describes four levels of uncertainties and potential modelling errors found in RANS. The modelling errors can be used as the stepping stone for data driven turbulence and the error can be corrected with such models. The following four uncertainties are found:

- **Level 1** Uncertainty that comes from averaging. With averaging, time local dynamical behaviour, such as specific shedding patterns, are lost resulting in a general loss of information.
- **Level 2** Uncertainty that comes from approximations, such as the Boussinesq assumption for RANS models that reduces the Reynolds stress from it's non-local behaviour to a local expression.
- **Level 3** Uncertainty in the functional form of the equation. The functional form approaches and satisfies physics driven process but is not necessarily the exact equation. An example is the real dissipation versus the modelled dissipation in RANS models such as the $k - \epsilon$ model.
- **Level 4** Uncertainty in the coefficients of the RANS models. The coefficients in the models are based on experimental data and use specific conditions depending on the tests that might infer biases.

In the review paper of Duraisamy [96], the developments in the field of data-driven/machine are provided with the main recommendation to use physics informed models to infer data. Physics informed models reduce the search space and the general guideline is to use non-dimensional values, consider invariance, use non-local quantities and have an understanding of the used data. The physics informed terms can also be constraint to result in satisfying results, by for instance using conservation laws.

The current report will only discuss the level two and level three approaches. For level four errors, the author recommends the reader to look at the review paper of Duraisamy et al [82] for a concise but informative summary.

Challenges and Problems with Data Augmented RANS models

The papers by the group of Duraisamy [7, 82, 96] have list challenges and problems related to data turbulence modelling and are meant as guidelines. Most of the turbulence modelling methods take into account the guidelines, but do not always adhere to it. For instance, Singh et al [7] promotes the use of propagating uncertainties to the output, but a significant amount of work in the field of data driven turbulence only relies on deterministic results. The following challenges and problems are listed:

1. **Usage of data** There is only a limited set of high fidelity data sets and turbulence models augmented with data show bias towards a certain data set. Small discrepancies in data can also (potentially) drive to incorrect models if not taken care off properly. Due to the high computational cost of high Re number high fidelity data, all high fidelity data have low Re numbers.

2. **Consistency of data between the RANS and the high fidelity model** The description of parameters of the high fidelity model might not be the same as for RANS models, as for example the dissipation ϵ.

3. **Correct form of the data augmentation for turbulence models** Due to a limited data set and the unknown nature of the problem, the true form of the turbulence model is not known and the designer of the turbulence model should take care in which forms are applicable.

4. **Generalisability of the data across data sets** Learning models from specific data sets can create a bias for that model and cannot guarantee prediction accuracy between training and testing data.

5. **Balance between model and data** Data driven techniques have been developed to produce models, but the researcher should take care to what extent data driven equations drive the model result compared to physics driven analogies.

6. **Loss of data due to the averaged field** For RANS, the process of averaging can lead to a loss of information as during the averaging process multiple different mechanism at a certain locations of x can be lost (Level 1 error from previous section).

7. **Impact of discretisation and numerical filtering** When LES models are used as the high fidelity data set, care should be taken that numerical errors due to filtering should not be propagated through the data inference.

8. **Interpretability of models** More interpretable models, by for instance a reduction of the amount of model terms, allow for a better understanding and less implementation errors.

9. **Data driven models shouldn't influence correct flow patterns** The correctly predicted flow patterns by RANS models shouldn't be influenced in a negative manner by the data driven corrections.

10. **Uncertainties should be accounted for** Uncertainties in the model should be accounted for if possible and be propagated to the output. The result is that an engineer has the possibility to know how the prediction influences the result and where modelling might provide inadequate answers.

2.3.2 Data Driven Turbulence and Machine Learning

The modelling errors of RANS and the challenges of data driven RANS are implemented in multiple cases using machine learning. The main methods are related to field inversion and machine learning, tensor bases machine learning, physics informed machine learning, sparse implementation of machine learning and corrections to the Spalart-Allmaras turbulence model.

Field Inversion and Machine Learning

As far as the author can find, the use of high fidelity data for Reynolds stress modelling was first introduced by Tracey et al. (part of the group of Duraisamy) [4]. The work is described as a proof-of-concept and Kernel regression is used to train the true anisotropy given a set of flow features from a low-fidelity model. Four inputs where given: the x and y coordinates of the barycentric triangle, a marker function m that provides a measure of the local flow condition for parallel shear flow and the ratio of turbulence production to dissipation. From the trained kernel regression model, new predictions were made for the anisotropy of turbulence. An improvement of the turbulence anisotropy is found compared to RANS, with the problem that the correction is a function of spatial coordinates.

Continuations on the work of Tracey et al, resulted in the Field Inversion Machine Learning (FIML) technique introduced by the group of Duraisamy [5, 6, 97, 7]. In FIML, the discrepancy between RANS and the high-fidelity data is inferred in terms of spatial coordinates. The inversion is done using Bayesian inversion, able to show uncertainty in predictions of sections of the flow what can act as a tool for the engineer. After the inversion, a machine learning algorithm is used to find a corrections in terms of the mean flow parameters. Most of these works followed on adding a source term $B(x)$ to the production term in the Spalart-Allmaras model like

$$\frac{D\tilde{\nu}}{Dt} == B(x)P_{\tilde{\nu}} - \epsilon_{\tilde{\nu}} + D_{\tilde{\nu}} + T_{\tilde{\nu}} + D^{C}_{\tilde{\nu}} \tag{2.47}$$

In Singh et al [7], 1D channel flow was shown to be improved by only using a sparse set of data. Another test, showed improvements in the prediction of the lift coefficient, drag coefficient and stall onset angles for turbulent flow separation over airfoils. Generalisability was found between data sets, where improvement was found for airfoils and angles of attack that were not a part of the training set.

Ferrero et al [9] used the FIML for improving RANS predictions for low pressure gas turbine cascades. A similar improvement to Singh et al [7] is found compared to original Spalart-Allmaras RANS model. Unbiased programmed principles improved prediction favouritism and a sparsity constraint is added to create a more simple model, where only four terms are considered.

Tensor Bases Neural Network and Variations

Ling et al [16] used a neural network to represent the anisotropy of the Reynolds stress tensor, calling it the Tensor Basis Neural Network (TBNN). A special neural network is proposed to promote Galilean invariance based on previous work by Ling [98]. The neural network creates an EARSM (in the form of (2.32)) using the integrity bases by Pope [17] (As seen in Appendix A) that is able to promote the Galilean invariance for the Reynolds stress anisotropy. The result is a Boussinesq assumption with a non-linear correction term

$$b_{ij} = -\frac{\nu_t}{k} S_{ij} + b_{ij}^{\Delta} \tag{2.48}$$

In (2.48), b_{ij}^{Δ} is the additional anisotropy correction that have the form of the bases and invariants of Pope [17] (See Chapter 2 for EARSM models). The goal of the neural network is to find coefficients based on the invariants for the tensor bases. The predicted eddy viscosity of the neural network were shown to be significantly more accurate than for linear and non-linear Boussinesq RANS models. The model was able to show separation and corner vortices in duct flow compared to RANS.

Fang et al [19] did a deep dive in the terms that the TBNN predicts and found that a TBNN is able to make an effort to overcome the Boussinesq deficiencies. Non-locality of the EARSM is still noted to be an issue for such models, as an EARSM is still inherently local.

Cai at al [18] revisited the work by Ling et al [16] and proposed an augmented TBNN. In the work, non-local quantities (such as wall distance and the friction Reynolds number) are added to EARSM structure. Transfer learning is used to mitigate the issue of a lack of data, using datasets of different Reynolds numbers to cross correlate between cases. Strong agreement was found between the reference data and the prediction data.

Kaandorp & Dwight [20] created a random forest algorithm called Tensor Basis Random Forest (TBRF), expanding upon the work off Ling et al [16]. An EARSM model, similar to Ling et al [16] with the tensor bases and invariants of Pope [17] (As seen in Appendix A), are created. The random forest algorithm is less demanding to implement than the TBNN, less demanding to train, needs less hyperparameters and has a better natural protection against overfitting. Different training and testing sets are used such that the testing set is not the same as the training set. For all cases, TBRF showed improvement over the baseline $k - \omega$ simulations.

Physics Informed Machine Learning for Reynolds Stress Anisotropy

The research group of Xiao [21, 22, 23] has developed a different strategy for the usage of neural networks for turbulence modelling, calling it the physics informed machine learning (PIML). A Reynolds anisotropy discrepancy field is obtained from high fidelity data where a neural network is trained to learn the discrepancy. In the earliest work, Wu et al [21] made a discrepancy correction based on the spatial field that was expanded in Wang et al [22] to create corrections based on the mean flow field parameters. A random forest regression routine is used to create a correction for the Reynolds stress anisotropy. Improvement of the Reynolds Stress anisotropy is found over the RANS simulation in the barycentric map.

In later work, Xiao et al [23] made an improvement for the prediction of the main flow parameters, rather than only the Reynolds stress anisotropy. For the input features, the strain rate, rotation rate, gradients of pressure and gradients of the turbulent kinetic energy and an integrity basis of the invariants of Pope [17] are used. The result are 57 potential candidate functions that the random forest regression can explore. Improvements are found in the predictions of the mean flow parameters, but transforming Reynolds stress anisotropy improvements to quantities of interest is still challenging. At the time of publishing the article, the state of the art machine learning models had problems with physical system properties (fixed by machine learning models such as PINNs, Raissi et al [62]), as the machine learning algorithms are not well suited to learn physical system properties, such as conservation laws. Another problem are numerical issues of neural networks, where non-smoothness provides problems with velocity predictions. The work of the group of Xiao also states problems with generalisability, where neural networks add an improved correction for statistically similar flows but cannot guarantee improvements for different cases.

Sparse Machine Learning Anisotropy Corrections

Jiang et al [26, 25] developed a deep neural network, the physics-informed residual network (PiResNet), to compute the Reynolds stress anisotropy. A universally interpretable machine learning framework is created where a physics informed residual network is made. The machine learning model uses the tensor basis and invariants of Pope [17] (As seen in Appendix A), similar to TBRF and TBNN and adds three more candidates, being the dissipation, turbulent kinetic energy and the kinematic viscosity. The cost function for the machine learning algorithm consists of an L_1 and L_2 penalty, promoting sparsity. The result are simple and interpretable models with an requirement of four tensor bases at most are found with the sparsity adding a benefit in the reduction in sensitivity of the input parameters.

Jiang at al [26] state that TBNN models provide a non-unique input to output mapping with how the coefficients are build-up. A remedy to enforce uniqueness is a time-scale measure, with the result being a less biased result, more robust to noise and general input uncertainty.

Two main deficiencies are noted in PiResNet, being that the corrections are local and the corrections are lacking memory (do not have temporal effects). Turbulence is inherently non local and memory driven but the nature of the EARSM model creates local and non-memory corrections. Jiang et al [25] state that higher order derivatives have the potential to model non-local behaviour and might be a future solution, similar to an explanation by Spalart [2], stated in Subsection 2.1.5.

PiResNet did show satisfactory outcomes compared to RANS and claims that the deep neural network has an advantage over a random forest regression in extracting informative data features, capturing invariances and improved approximation capabilities. Jiang et al [26] urges the development of a benchmark dataset containing diverse typical flow phenomena for the development and evaluation of data-driven turbulence for the turbulence community.

Sáez de Ocáriz Borde et al [27] proposed a convolutional neural network with the aim of improving the interpretability of the model. Current models by Jiang et al [25, 26], TBNN [16] and more neural network driven works are claimed to be a 'black-box' and lack interpretability. A convolutional neural network is a form of deep learning where each neuron is connected to a small, nearby amount of neurons, whereby the same set of weights are shared by every neuron. For an in-depth look into convolutional neural networks, please look at the book of Millstein [99]. An improvement of the Reynolds stress anisotropy is found, based on the input of the mean velocity gradient with the output being the Reynolds anisotropy in one direction. The work is compared with the work of Fang et al [19] and a good fit is claimed to be found. The interpretability improvement is mostly reached by visualising the activation of each neuron and the sensitivity of the results.

Corrections Inferred for the Spalart Allmaras Model

A similar principle to Subsection 2.3.2, Franceschini et al [10] introduced a source term in the momentum equation and a source term in the turbulent equations of an Spalart Allmaras RANS model. Using a gradient descent based operator, a correction field is obtained. The correction field is transformed with machine learning to a correction field based on the main flow parameters. Franceschini et al [10] proofed that both corrections can work but have different constraints, a correction within the Spalart Allmaras model provides more control for less precision while a correction factor in the momentum equation had better results, but was more difficult to find an appropriate model for.

Volpiani et al [12] performed a similar exercise by adding a vectorial source correction to the momentum equation and using a similar data assimilation technique to describe the correction with machine learning correcting the terms in the local mean-flow quantities. Different to [10], Volpiani et al [12] proofed that for (small) differences in geometry, the machine learned frame work can still provide high quality corrected results.

Volpiani extended on the work in [13], by adding a correction term to the turbulent eddy viscosity with an additional contribution in [14] by analysing the capabilities of neural networks versus random forest for correction of the turbulent eddy viscosity. Neural networks are stated to perform better in interpolating and extrapolating the output quantity compared to random forest. For the random forest, oscillatory behaviour was better captured but at a cost of predictive behaviour. A compromise is proposed between a neural network and random forest algorithm. In the later work ([14]), the results from the machine learning algorithm were made sparse by choosing the most important terms.

From the same group as Volpiani, Cato et al [11] performed a more encompassing analyses. Data assimilation strategies for multiple corrections are compared using the Spalart-Allmaras-neg RANS model of Crivellini et al [15] with the main goal of finding the location of the most effective correction. The corrections have been added to the Reynolds Averaged momentum equations and the transport equation of the Spalart Allmaras model, resulting in

$$\overline{u}_j \frac{\partial \overline{u}_i}{\partial x_j} = -\frac{1}{\rho}\frac{\partial \overline{p}}{\partial x_i} + \nu \frac{\partial^2 \overline{u}_i}{\partial x_j \partial x_j} + \frac{\partial}{\partial x_j}\left((1+\xi)\nu_T S_{ij} - \frac{2}{3}k\delta_{ij} + R_{\tilde{\nu}} \right) + f_i \tag{2.49}$$

and

$$u_i \frac{\partial \tilde{\nu}}{\partial x_i} - \frac{\partial}{\partial x_i}\left(\eta \frac{\partial \tilde{\nu}}{\partial x_i} \right) = (1+\beta_\nu)P_{\tilde{\nu}} - \epsilon_{\tilde{\nu}} + D^C_{\tilde{\nu}} + \tilde{f}_{\tilde{\nu}} + \tilde{g}_{\tilde{\nu}}\tilde{\nu} \tag{2.50}$$

In (2.49) and (2.50), η multiplies the eddy viscosity and is added to directly correct for the eddy viscosity. A trace free symmetric tensor $R_{\tilde{\nu}}$ is a correction that is a conservative force within the eddy viscosity assumption while f_i is not a conservative force. The multiplication β_ν is a correction to the production, $\tilde{f}_{\tilde{\nu}}$ and $\tilde{g}_{\tilde{\nu}}\tilde{\nu}$ and acts as source term corrections to the general Spalart Allmaras model.

Cato et al [11], concluded that all terms responded differently to data-assimilated techniques. In general, the terms in the momentum equation ($R_{\tilde{\nu}}$ and f_i) performed the best, except for the scalar multiplier η in the Boussinesq assumption. The vector corrections in the momentum equation are an effective EARSM model and an expansion of the Boussinesq assumption. The source corrections ($\tilde{f}_{\tilde{\nu}}$ and $\tilde{g}_{\tilde{\nu}}\tilde{\nu}$) in the Spalart Allmaras transport equation also functioned well and showed promise considering the Reynolds stresses. The study proofed that despite using the Boussinesq assumption, accurate mean field predictions are able to be generated and for some cases even correct Reynolds stress values are found.

2.3.3 Symbolic Regression for Turbulence Modelling

The last subsection discusses machine learning, but symbolic regression can also provide data driven turbulence models. Symbollic regression techniques such as GEP and the Sparse Regression of Turbulent Stress Anisotropy (SpaRTA, Schmelzer et al. [34]) are some of the methods that tackles some of the unknown black box behaviour related to machine learning.

Symbollic Regression and Data Driven Turbulence

Weatheritt et al [29, 30] implement the original GEP of Ferreira et al [28] (explained in Section 2.2.3). The GEP created an EARSM, using the 2D bases and invariants of Pope [17] (For the full equations, see Appendix A). The study is split into an a priori study, demonstrating the reconstruction of the Reynolds stress and a prosteriori study by running a full implementation of the correction. The trained models using the technique are not sparse but are able to improve on the standard Boussinesq assumption models but robustness among other cases is not guaranteed.

An extension on the work is performed by Zhao et al [31] (Same group as Weatherhitt), where an effort is made to improve the RANS environment calculations to high fidelity data using GEP. RANS calculations are integrated in the GEP training loop in the form of a non-linear Boussinesq model, with an aim of improving a posteriori predictive accuracy and stable solutions. The framework was applied to wake mixing in turbomachines and showed good agreement to high fidelity data. The main disadvantage is that the added CFD loop provides greater computational cost.

Bleh and Geiser [32] expanded on the DHC-GEP algorithm of Ma et al [33], explained in Subsection 2.2.3, by normalising physical dimensional input features to be non-dimensional, reducing the likelihood of expressions with non-matching physical dimensions. The algorithm was used to find transition models for two turbine profiles with transition induced separation. A correction is applied in a $k-\omega$ production term, similar to (2.47) and LES data is used as the high fidelity reference. The GEP algorithm was able to accurately predict separation on the turbine profiles and a recommendation is given to check the method with more test cases.

Sparse Regression of Turbulent Stress Anisotropy

As a continuation on the work by Weatherhitt and all, Schmezler et al [34] introduced the Sparse Regression of Turbulent Stress Anisotropy (SpaRTA). SpaRTA promotes sparsity on the solutions by creating an EARSM from a pre determined set of candidate functions. The implementation of the corrections follow (2.48) and

$$\frac{\overline{D}k}{\overline{D}t} = P_k + R + D_k + T_k - \epsilon \tag{2.51}$$

In (2.51), the R is a correction to the k-equation. Similar to TBNN and TBRF, the corrections for b_{ij}^{Δ} and R are based on the integrity bases from Pope [17] (full set found in Appendix A), resulting in an EARSM model following Equation (2.32). The build-up of R is shown as

$$R = 2kb_{ij}^{R}\frac{\partial u_j}{\partial x_i} \tag{2.52}$$

The model discovery part in SpaRTA is subdivided in two parts: Model selection and model inference and a solution is sought to satisfy (2.36). For the model discovery part, an elastic net formulation is used, given as

$$\hat{\beta}_i = \underset{\beta}{\text{argmin}}\left\{||\Theta_{ij}\beta_i - y_j||_2^2 + \lambda\rho_r\,||\beta_i||_1 + 0.5\lambda(1-\rho_r)\,||\beta_i||_2^2\right\} \tag{2.53}$$

In (2.53), λ and ρ_r are hyper parameters (being a regularisation weight and mixing parameter), while y_j is the data. The equation is able to filter between candidate functions and create an initial estimation for b_{ij}^{Δ} and R. L_1 regularisation is used to force sparsity on the solution while L_2 regularisation is used to identify correlated candidate functions and reduce the dependency among them. The hyper parameters are chosen to avoid overfitting by means of cross validation and a wide range of values of λ and ρ are analysed to find the most optimum model.

For the model inference step, a L_2 ridge regression is used,

$$\hat{\beta^s}_i = \underset{\beta^s}{argmin}\left\{\left|\left|\Theta_{ij}^s\beta_i^s - y_j\right|\right|_2^2 + \lambda\,||\beta_i^s||_2^2\right\} \tag{2.54}$$

The model inference step removes the dependency of the model coefficients from the model selection step. An L_2 penalty is used to force the coefficients to smaller values as experience by Schmelzer et al has shown that large coefficients increase the convergence sensitivity. In (2.54), the superscript s is used to note the subset of the selection from Equation (2.53). The model inference step does not change the inactive zero values as a result from (2.53).

The test cases analysed by Schmelzer et al [34], are a periodic hill, a converging-diverging channel and a curved backward-facing step. Schmelzer et al [34] found a sparse set of correction terms for each case. Cross-correlation has been performed between cases to find better results. It is found that a model trained for one flow can perform well outside the training range with similar features. Corrections of R are found to be sufficient for providing a better representation of the ground truth, rather than the Boussinesq correction.

The effect of SpaRTA can be seen in Figure 2.5, where three resultant models of SpaRTA are compared with high fidelity experimental data at a higher Re number than the models are trained at and with a $k-\omega$ SST 'baseline' computation. Figure 2.5 shows a considerable improvement in prediction of the bulk velocity for all three obtained models and shows superior prediction to the $k-\omega$ SST simulation, even if the trained Re number is different.

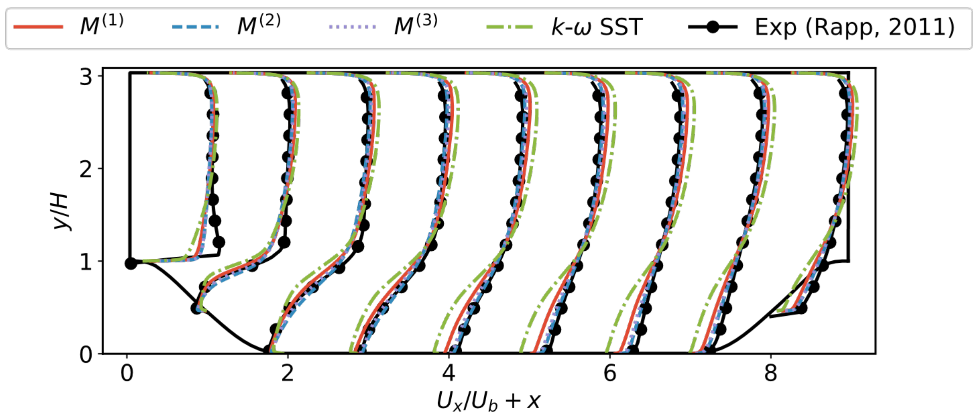


Figure 2.5: Predicted streamwise velocities of the three models found by SpaRTA, the $k - \omega$ SST baseline and high fidelity data of Rapp [100] for the periodic hill at $Re = 37000$, courtesy of Schmelzer et al [34].

SpaRTA uses a procedure called k-frozen RANS, that is able to extract the error of high-fidelity data sources efficiently. The high-fidelity value for k, u_i and b_{ij} are from the high-fidelity data and for a $k - \omega$ SST model, only the ω equation is computed. For the unknowns, the Residual R and the correction for the Boussinesq assumption b^{Δ}_{ij} are obtained at every iteration.

Developments beyond SpaRTA

Around a similar time as SpaRTA, Beetham et al [101] explored sparse regression for turbulence closure modelling. Contrary to the work of Schmelzer et al [34], Beetham et al [101] used a single sparse regression loop with an L_1 Lasso style penalty to force sparsity. No second loop of tuning the coefficients were present. Two cases are considered with different correction methods: A homogeneous free shear turbulence case where corrections are made to a RSM and a periodic hill case where corrections are added to the Boussinesq assumption, creating an EARSM (Similar to Schmelzer et al [34], without the correction term to the k-equation). Instability of the correction terms are found with the value of the coefficients, similar to SpaRTA. SpaRTA solves the instability by adding an L_2 penalty to the regression routine. The conclusion of Beetham et al [101] is similar to the conclusion of Schmelzer et al [34], where there is a significant improvement found in the performance by adding a sparse set of algebraic terms to the main equations.

Cherroud et al [39] improved upon the original SpaRTA scheme of Schmelzer et al [34], creating the SBL-SpaRTA (Sparse Bayesian Learning - SpaRTA) method. SBL-SpaRTA treats the parameters as random variables with associated prior normal distributions, using the SBL algorithm of Tipping [102]. The goal is to provide uncertainty intervals for the parameters such that a stochastic prediction is made, rather than a deterministic 'stiff' result. An added benefit with stochastic predictions is that flow regions with a greater sensitivity to turbulence modelling can be visualised and can be used as a tool for the engineer. The same non linear EARSM tensor bases and invariants from Pope [17] are used, like SpaRTA, TBNN, TBRF and more. Similar to SpaRTA, SBL-SpaRTA is shown to have better results compared to a standard Boussinesq model and provides good agreement with high fidelity data. A similar conclusion is made compared to a physics based EARSM, where SBL-SpaRTA is shown to provide better agreement to high-fidelity data while being a less complex model.

SpaRTA has been incorporated for the prediction of wakes of wind turbine farms by

Steiner et al [35, 36], where a two turbine constellation is used as the training data that is cross validated with a three turbine constellation. A $k - \epsilon$ RANS model is used, where similar to Schmelzer et al [34], terms are found for the production term in the k-equation and the anisotropy term in the Bousinessq assumption.

The work by Steiner et al [35, 36] is at a high Reynolds number and three dimensional, expanding upon the original SpaRTA where only 2D considerations are made. With the original SpaRTA, only the shear strain and rotation strain tensors are taken into account for the invariants. Steiner et al expands upon the invariants by adding pressure- and k-gradients and a set of additional scalar variables as invariants. The result of the study is that the SpaRTA procedure is able to create corrections for RANS models that are able to approach the LES ground truth for specific cases related to wind turbine fields. The same group continued this work in [37] by adding classifiers to the areas of interest where a correction is needed, allowing the result to increase in simplicity.

The SpaRTA algorithm has also been incorporated by Stöcker et al [38] for particle laden flow. In that research, a correction for the Boussinesq model is not implemented. Rather, only the production term is corrected and an additional correction for the dissipation is added. Although a better agreement is found with the high-fidelity data, it is less great than in the work presented by Schmelzer et al [34].

2.3.4 Comparison of Methods for Data Driven Turbulence

The current section acts as a brief summary for all (relevant) models presented in the works, comparing them together on sparsity, generalisability, uncertainity and correction (Based on the challenges and problems of Section 2.3.1 of the works of Duraisamy [82, 96, 7]). Table 2.1 shows a summary of the most important and impact full methods.

Sparsity determines if the correction form of the method is sparse and easy to understand. Generalisability analyses the ability for a model from a training set to accurately predict a model from a test set. Uncertainty checks if the model has uncertainty measures, like standard deviation to show probability bands. Correction is based on the work of Duraisamy [82] and states which level RANS correction is applied. Both sparsity and generalisability have two checkmarks or two crossmarks, depending on how well the model relatively performs. Uncertainty is just a checkmark or a crossmark (boolean decision), correction states which level based on Section 2.3.1 is corrrected with the model. Note that Table 2.1 is not a trade-off table and only functions to summarise all the methods and that not all methods of the previous sections are summarised in the table, only the ones deemed most relevant.

Table 2.1: Comparison of different data driven turbulence techniques for different categories, based on the Duraisamy [82, 96, 7].

	Sparsity	Generalisability	Uncertainty	Correction	Works
TBNN	✗	✓[4]	✗	Level 2	[16, 98, 19, 18]
TBRF	✗	✓✓	✗	Level 2	[20]
PiResNet	✓	✓✓	✗	Level 2	[25, 26]
FIML	✗[5]	✓	✓✓	Level 2, 3	[5, 6, 97, 7, 9]
Volpiani[6]	✗[7]	✓	✗	Level 2, 3	[12, 13, 14, 11]
PIML	✗	✓	✗	Level 2	[21, 22, 23]
SpaRTA	✓✓	✓✓	✗	Level 2, 3	[34, 35, 36, 37, 38]
SBL-SpaRTA	✓✓	✓✓	✓	Level 2, 3	[39]

From Table 2.1, it can be seen that all methods tackle the level two error as defined in Section 2.3.1, where the Boussinesq assumption is the source of the error. Few methods tackle error three, being the error of the model form. The work by the group of Volpiani provides a Level two correction explicitly with the work of Cato et al [11] by providing a source term and a multiplication term in the momentum equation, mimicking a Level two error correction. All of the level two corrections (except for the work by the group of Volpiani, FIML and PIML) create an effective EARSM, mostly following the tensor bases and invariants of Pope [17]. Some models, such as PiResNet, expand the EARSM, by providing invariant values such as the dissipation and turbulent kinetic energy.

The most sparse methods are PiResNet, SpaRTA and SBL-SpaRTA. In table 2.1, PiResNet is considered less sparse with the reason that there is more control in the candidate library correction terms of SpaRTA and SBL-SpaRTA. All three methods behave differently in achieving sparsity, where for PiResNet an L_1 lasso relation is considered, forcing terms in the machine learning algorithm to be zero. For SpaRTA an elastic-net approach is used and for SBL-SpaRTA an SBL routine from Tipping [102] is applied. In a later work of Volpiani, sparsity is reached by employing a method choosing the most important terms, not reflected in Table 2.1. A different work from Ferrero et al [9] for FIML also uses enforces sparsity.

All models show generalisability to a certain extent. A particular good generalisability is found for models that enforce sparsity, what might be related to a reduction in potential overfitting. Models such as TBNN, the work of Volpiani and PIML show promising behaviour for case-specific work, where the Reynolds number is different for the same geometry. In the TBNN [16] it is stated that the performance of the model deteriorates significantly for different geometries, different to SpaRTA where multiple models deduced from multiple test-cases provide a decent correction. Compared to all works, the FIML method (Singh et al[7]) states that generalisabilty is reached between different shape of airfoils at different angles of incidences, using experimental pressure measurements. For most works, an increase in generalisability is seeked, with Jiang et al (from PiResNet [25, 26]) arguing for a standardised test condition set across works. Xiao et al [23] has an interesting statement regarding generalisability, where statistically similar flow are easily corrected in their method.

FIML and SBL-SpaRTA are the only methods that provide a measure of uncertainty, stating where predictions of the solutions are most likely to be inaccurate. FIML reaches it in the inversion step, using a Bayesian inversion step while SBL-Sparta uses an SBL method.

[5]Only from the work of [18], where higher Re numbers show similar behaviour

[6]Doesn't have a particular name for the work and is considered to the full group of work by Volpiani

[7]In later works (Volpiani et al [14]) sparsity is considered

3. Methodologies

The experiment can be sub-divided in three main stages, being the data generation stage, model training stage and the training validation stage. A $\psi - \phi$ model is developed, discussed in Section 3.2, that is analogous to RANS turbulence modelling and is analogous to the residual from k-frozen RANS. The data generated is subject to the sparse regression routine, discussed in Section 3.3, where new models are trained. From the model training phase, errors are analysed and a subset of trained models are discussed and tested in depth in Section 3.4. A list of assumptions is given in Section 3.1 and an overview of the complete process is given in Figure 3.1 with a Table of the most important settings provided in Table 3.1.

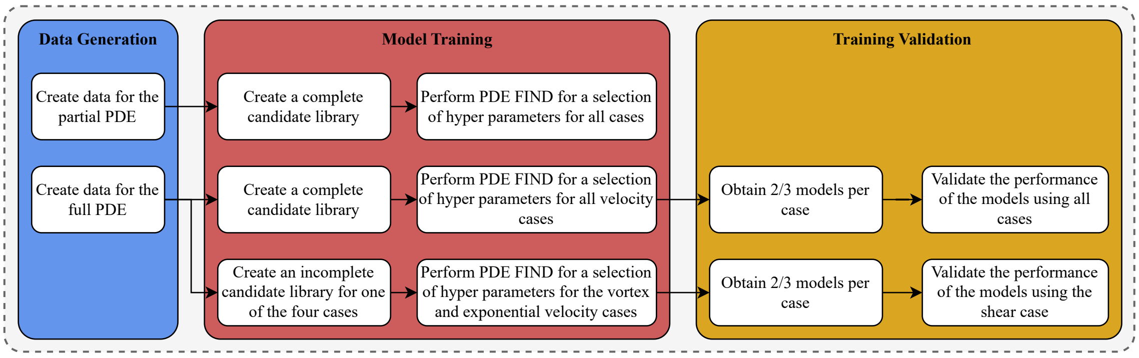

Data Generation
Create data for the partial PDE
Create data for the full PDE
Model Training
Create a complete candidate library
Perform PDE FIND for a selection of hyper parameters for all cases
Create a complete candidate library
Perform PDE FIND for a selection of hyper parameters for all velocity cases
Create an incomplete candidate library for one of the four cases
Perform PDE FIND for a selection of hyper parameters for the vortex and exponential velocity cases
Training Validation
Obtain 2/3 models per case
Validate the performance of the models using all cases
Obtain 2/3 models per case
Validate the performance of the models using the shear case

Figure 3.1: Compact work breakdown structure of the planned work.

Table 3.1: Summary of all setting used in the report.

Setting	Value	Setting	Value
Regression samples	1000	Discretisation in y	150
Different velocity cases	3	Discretisation in x	150
Tuneable hyperparamters	λ, tol	Time step	0.1
Max iterations TrainSTRidge	25	Amount of time steps	300
μ	0.001	min-max Domain x and y	-2 to 2

3.1 Assumptions

The following assumptions are present in the current work:

1. **All values are dimensionless** As the system is analogous to turbulence modelling, no physical units are taken into account for ψ and ϕ. Coefficients in RANS models carry dimensions to ensure a correct dimension in the transport equation. Therefore,

results from the regression routine can always be considered to have the correct dimension as the coefficient can reflect the dimension to correct the result. To make the analyses easier, the dimensions of ϕ and ψ are dimensionless throughout the report.

2. **The Reynolds stress can be analogous as a scalar value** The current work can be seen as a primer for the viability to create a transport equation for the Reynolds stress using sparse linear regression routines. As the current study plays with the linear regression routine, a simple scalar Reynolds stress is justified to analyse the performance and generate simple results that are understandable. It does create an inaccurate representation of the Reynolds stress tensor but for the purpose of the current report it is acceptable.

3. **Steady state velocity over the complete flow field** There is a steady state flow field driving the solution, meaning that there is no chaotic behaviour in the flow, as would be expected for turbulence. It also means that no velocity equation has to be solved and the complete focus can be on the $\psi - \phi$ system. An error can be considered and care should be taken with the performance of the linear regression routine for chaotic flows.

4. **The turbulent transport term does not have to be modelled** The turbulence transport is neglected for the ϕ and ψ equations as the turbulent transport term is driven by a triple correlation. The triple correlation is difficult to measure in general and for the k-equation is modelled by means of the eddy viscosity assumption. With the steady state velocity field over the complete system, the error should be small and for the purpose of the report, to analyse sparse linear regression routines for turbulence modelling, neglecting the term shouldn't adjust the conclusion of the research.

5. **The pressure term does not have to be modelled** The pressure is not considered in the current report as it would mean an additional equation that would need to be solved, and therefore the pressure strain tensor can be neglected. The pressure strain tensor is modelled in other works and one might see the additional terms R_ϕ in the ϕ equation as modelling of the pressure term. The result is that some analogy is lost between turbulence and the current system.

6. **The strain rate tensor can be squared for production terms** To ensure rotational invariance, the strain rate tensor is squared. The analogy compared to turbulence modelling is off, but justifiable as the Boussinesq assumption for the Reynolds stress contains the strain rate tensor, effectively always providing a squared strain rate tensor in the production term. Not squaring the strain rate, would give non-physical and non-rotational invariant results.

7. **The model is two dimensional** For the purposes of ease, understanding and computational cost, a two dimensional model is chosen over a three dimensional model. As the system is not chaotic, it is believed that a two dimensional system will represent the turbulence analogy well.

3.2 The Data Generation Phase

In the data generation phase, a model problem analogous to turbulence modelling is generated for the purpose to obtain models from sparse regression. The model problem is a two equation model, dubbed the $\psi - \phi$ system, on a two dimensional square grid and the software used to generate the data is FEniCSx.

3.2.1 The $\psi - \phi$ System of Equations

The current work is centred around a model problem with equations analogous to turbulence modelling. The model equations will follow similar principles to RANS as it is meant as an extension of RANS. The model problem is a two equation system, the $\psi - \phi$ system, reflecting the k-equation (2.20) as ψ and a scalar value for the Reynolds stress transport equation (2.14) as ϕ. The equations follow certain requirements and has the goal to be analogous to turbulence modelling. The transport equation for ψ is given as

$$\frac{D\psi}{Dt} = \frac{\partial \psi}{\partial t} + u_i \frac{\partial \psi}{\partial x_i} = c_1 \phi S_{ij} S_{ij} + c_2 \frac{\partial^2 \psi}{\partial x_i \partial x_i} - c_3 \psi^2 = P_\psi + D_\psi + \epsilon_\psi \tag{3.1}$$

and for ϕ

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + u_i \frac{\partial \phi}{\partial x_i} = d_1 \phi S_{ij} S_{ij} + d_2 \frac{\partial^2 \phi}{\partial x_i \partial x_i} - d_3 \phi^2 + R_\phi = P_\phi + D_\phi + \epsilon_\phi + R_\phi \tag{3.2}$$

The R_ϕ term is the residual term and aims in modelling the uncertainty between RANS and DNS. The current work will have two versions of the R_ϕ equation, the partial system and the full system. The partial system is given as

$$R_\phi = d_4 S_{ij} S_{ij} + d_5 \frac{\partial \phi}{\partial x_i} \frac{\partial \psi}{\partial x_i} - d_8 \phi^2 \frac{\partial \phi}{\partial x_i} \frac{\partial \phi}{\partial x_i} = P_{\phi,1} + D_\phi^C + Q_\phi \tag{3.3}$$

while the full system is

$$\begin{gathered} R_\phi = d_4 S_{ij} S_{ij} + d_5 \frac{\partial \phi}{\partial x_i} \frac{\partial \psi}{\partial x_i} + d_6 \phi^2 S_{ij} S_{ij} - d_7 \psi \phi - d_8 \phi^2 \frac{\partial \phi}{\partial x_i} \frac{\partial \phi}{\partial x_i} = \\ P_{\phi,1} + D_\phi^C + P_{\phi,2} + \epsilon_\phi^C + Q_\phi \end{gathered} \tag{3.4}$$

The equations should conform to the following requirements to fully represent turbulence modelling:

1. **Rotational invariant** Ensures that certain terms, such as the production term, are not related to the coordinate system.
2. **Finite steady state** The values of ϕ and ψ should be bounded for t going towards infinity.
3. **Transport equations** ϕ and ψ should be transport equations to represent the non-locality and temporal effects.
4. **Analogy with Turbulence modelling** There has to be a clear analogy between the model equations and turbulence modelling.
5. **Coefficients shouldn't be too large or too small** To have a fair distributed contribution of terms, the coefficients shouldn't be too large or too small for specific terms.

In the work of Schmelzer et al [34], the most effective correction was found in the R term of the k-equation, where the correction was based on the Reynolds stress tensor and therefore the production term. Work of the group of Duraisamy [82, 96, 7] state, to a certain extent, that discrepancies to the Reynolds stress tensor are one of the main contributors to the error in the solution. The barycentric map in Figure (2.2) (based on the work of Duraisamy et al [82] and Wang et al[22]) show the differences in the Reynolds stress between DNS and RANS, where the discrepancies are large. A transport equation for the Reynolds stress is able to mitigate such an error and provide a better prediction.

The $\psi - \phi$ equations therefore, assumes that the residual found in k-frozen RANS is a function of the Reynolds stress tensor. As the Reynolds stress tensor, is a 3D dimensional tensor, the complexity for modelling is rather significant. As the purpose of the work is to analyse and quantify sparse regression routines for turbulence modelling, the Reynolds stress is modelled and simplified as a scalar quantity. Based on the work from Schmelzer et al. [34], the full discrepancy is considered in the k-equation as an incorrect modelling of the Reynolds stress tensor.

As stated, the ψ equation (3.1) is a direct measure for the k-equation and is based on (2.20), having a production term, diffusion term and a destruction term. The turbulent transport term is not considered in the equation (as stated in Subsection 3.1), as the analogy of a scalar value for the Reynolds stress and the steady state velocity does not suit the triple correlation and is therefore neglected.

The production term is based on (2.20), but to force a scalar value as well as rotational invariance of the shear strain rate tensor, it is squared (as explained in Subsection 3.1). The dissipation term is based on the destruction terms in both the $k - \epsilon$ and $k - \omega$ equations (By Jones et al [74] and Wilcox [75]), what is driven by a negative squared term of the variable of interest. The dissipation term in the k-equation is driven by (for two equation models) a transport equation. A third equation is considered not relevant for the current work as the goal of the dissipation is to limit the amount ψ in the system, what is reached with the current analogy.

The ϕ equation is a scalar measure for the Reynolds stress and is a transport equation to visualise non-local temporal behaviour. The ϕ equation is based on the RSM of Equation (2.14) and as the k-equation is the trace of the RSM equation (2.14), meaning that the ϕ equation is similar to the ψ equation. With similar arguments as the ψ equation, the ϕ equation has a production, diffusion and destruction term. Different to ψ equation is the R_ϕ term that represents the residual in turbulence modelling and the uncertainties related to modelling.

The ϕ equation is different to an RSM, not representing the pressure strain correlation tensor Π_{RST} and the turbulent transport tensor C_{RST}. In the current work, pressure is not simulated and can be assumed to have no impact on the current equations, meaning that the pressure strain correlation can be neglected. One can see the residual terms in (3.2) as a model for the pressure strain, but note that the terms are not meant to represent it. Similar argument can be made for the turbulent transport tensor, where the pressure fluctuations can be neglected and similar to the ψ equation, the triple correlation can be neglected.

Having ψ as a measure for the k-equation and ϕ for the Reynolds stress can be an issue when relating it to turbulence modelling. The k-equation is used in (non-linear) Boussinesq models in the form of one- or two- equation models (See Section 2.1 of Chapter 2 for more). The Boussinesq models create a prediction for the Reynolds stress and having it as a transport equation for the Reynolds stress makes the use of the Boussinesq assumption redundant. As the Boussinesq assumption is redundant, so is the k-equation as no measure for ν_t has to be given. As the work is meant to augment Boussinesq models in potential future work, the ϕ equation can be seen more as an additional, non-local memory driven correction term based on the Reynolds stress discrepancies found in the residual of k-frozen RANS of Schmelzer et al [34]. Regardless, the current analogy could work on multiple levels and the analogy for ϕ to a RSM is meant as a basis for the modelling of ϕ.

The R_ϕ term (3.3) and (3.4), short for additional, aims in modelling the uncertainty between RANS and DNS. The current work will have two versions of the additional equation, the partial system and the full system. The partial system is meant as a lower entry functional form for the regression routine to discover. The partial system has three additional terms, shown in (3.3), being an additional strain rate driven production term a cross diffusion term and a quartic term. The strain rate driven production term is the 'kick starter' for the process and converts the velocity field to quantities of ϕ, without the term, there won't be any value of ϕ generated in the domain. The cross diffusion term is a non conserving diffusion term between ϕ and ψ and is inspired by the cross diffusion term in the $k-\omega$ equation of Wilcox [75]. The quartic term has the form of a non-conservative diffusion term, with ϕ squared in front of it. It acts as an unpredictable term and has no direct physical meaning with the main idea of it representing an uncertainty addition to an equation.

The full equation is an expansion on the partial equation by adding two additional terms, shown in Equation (3.4). The additional terms are an extra production term and an extra destruction term. Both the terms are highly correlated with the terms in the equation and provide an additional challenge for the regression procedure (The correlation of terms can be found in Appendix F).

The ψ equation is linked to the ϕ equation through the production term as the production term is dominated by ϕ. Note that in the k-equation (2.20), the Reynolds stress drives the production and as ϕ is a measure for the Reynolds stress and the ϕ and ψ equations are similar, they both have the same production term.

The ϕ equation is linked to ψ through the cross diffusion and the cross destruction term. The cross diffusion term is a non-conservative diffusion term, inspired by the Wilcox $k-\omega$ model [75] and states that a deficit of concentration of ψ can drive a diffusion of ϕ to the region of low ψ. The cross destruction term is a dissipation like term that limits the generation of ϕ by means of ψ. A high amount of ψ can act as a limitation on the production of ϕ and limit the amount of ϕ.

Table 3.2 show the coefficients of the cases. The coefficients are the same between the partial and full case. Note that for all equations ((3.1), (3.2), (3.4) and (3.3)) the term $d_4 S_{ij} S_{ij}$ is crucial to start the solution from an initial condition of 0 with an active flow field.

Table 3.2: Coefficients related to the ψ-ϕ system used for testing. Both the partial case and the full case have these values.

Coefficient	Value	Coefficient	Value
d_1	1	c_1	1
d_2	0.1	c_2	0.035
d_3	0.75	c_3	1.5
d_4	1.5		
d_5	1		
d_6	0.5		
d_7	3		
d_8	0.35		

3.2.2 The Three Velocity Cases

Three different velocity cases are considered for the $\psi - \phi$ model that are all steady state. The steady velocity profiles eliminates the chaotic nature of turbulence, as well as the feedback loop between turbulent kinetic energy and velocity. For the current study, however, it is considered fine as the purpose is to analyse the performance of the regression routine for future purposes.

The computations are done on a square two-dimensional domain, with the velocity cases being a vortex case, an exponential case and a shear case. Figure 3.2 show the velocity profiles for each specific case.

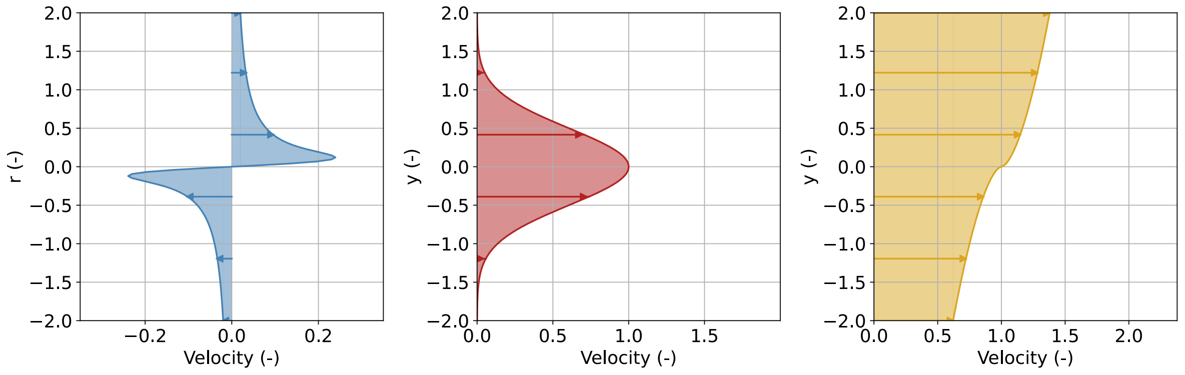


Figure 3.2: The velocity of the flow field for (from left to right) the vortex case, exponential case and the shear case. Note that for the vortex case the flow field is in the radial direction while in y for the exponential and shear case.

The vortex case is simulated as a Lamb-Oseen (Oseen [103]) vortex, the exponential case as an exponential equation and the shear as a polynomial equation. The velocities are given as

$$u(r) = 0.5\frac{\left(1 - e^{-(r/b)^2}\right)}{2\pi r} \qquad (3.5) \qquad\qquad u(y) = e^{-2y^2} \qquad (3.6)$$

for the vortex and exponential case and for the shear case as

$$u(y) = 0.25y^{3/5} + 1 \qquad (3.7)$$

For the shear case, the derivative with respect to y will go to infinity when y approaches 0, providing a unique challenge for the sparse regression algorithm.

For the square domain, the boundary conditions are found in Table 3.3. The exponential and shear case both have Dirichlet boundary conditions for ϕ and ψ on the left domain as the streamlines enter the domain from the left. The vortex case only has Neumann boundary conditions.

Table 3.3: Boundary conditions related for each case on a square domain, with x and y ranging from -2 to 2.

Side	**Left**	**Top**	**Right**	**Bottom**
Vortex	Neumann	Neumann	Neumann	Neumann
Exponential	Dirichlet	Neumann	Neumann	Neumann
Shear	Dirichlet	Neumann	Neumann	Neumann

For the exponential case, the Dirichlet boundary conditions are given as

$$\psi|_{x=0} = 0.8e^{-0.7y^4} \qquad (3.8) \qquad\qquad \phi|_{x=0} = 0.56e^{-0.7y^4} \qquad (3.9)$$

and for the shear case the Dirichlet boundary conditions are given as

$$\psi|_{x=0} = 0.12e^{-5y^2} \qquad (3.10) \qquad\qquad \phi|_{x=0} = 0.12e^{-3y^2} \qquad (3.11)$$

All other boundaries are given as Neumann boundaries where the conditions are given as 0 ($\frac{\partial \psi}{\partial x_i} = 0$ and $\frac{\partial \phi}{\partial x_i} = 0$). The aim of the Dirichlet boundary conditions is to simulate some sort of upstream phenomena, such as the influence of a jet or cylinder wake that comes in the domain at a later stage. The values for (3.8), (3.9), (3.10) and (3.11) are arbitrary as the author has used brute force to find suitable boundary conditions.

3.2.3 Discretisation of ψ and ϕ

The current subsection discusses the discretisation in the spatial and the temporal field. As there is no ground truth, the error is compared to a preset value that acts as a reference. The preset values are 0.13, 0.02 and 0.016 for the vortex, exponential and shear velocity cases for ψ and 0.25, 0.041 and 0.061 for the vortex, exponential and shear velocity cases for ϕ. The only difference between the preset and the ground truth is a scaling of the system. The error is taking as the average value of ψ or ϕ over the complete temporal and spatial range. Figure 3.3 shows a log-log plot for the spatial discretisation.

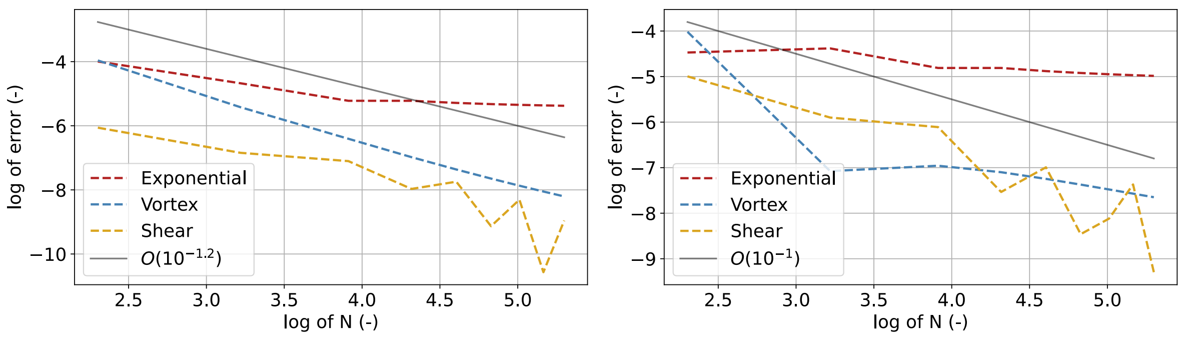


Figure 3.3: Log-log plot looking at the spatial discretisation error with on the left ψ and on the right ϕ.

From Figure 3.3, a rather poor performance can be found, with the ψ equation being of $O(10^{-1.2})$ approximately for all cases while for ϕ being of $O(10^{-1})$. A reason might be that the complex PDE equation is interfering with each others term and an accumulation of errors might be happening. Another explanation could be the use of low accuracy finite element functions to represent the solution.

The time discretisation for ψ and ϕ can be found in Appendix B. An forward-backward Euler time integration scheme is used, where all linear terms in the transport equation are implicit, while the non-linear terms will be explicit, ensuring a bi-linear form $a(u, v)$.

The simulations are all ran with a time step of 0.1 for a total of 300 steps, resulting in a time window of 0 to 3 s. The spatial discretisation both in x and y are equal to 150, ranging from a minimum value of -2 to a maximum value of 2. The discretisation is rather high due to the poor performance in Figure 3.3. For the ease of the report, the x and y values are dimensionless. There is no noise considered as the work is deemed a primer for future work. A table with data is provided in Table 3.1.

3.2.4 Data Generation with FEniCSx

The python package FEniCSx is used to generate the data for the training stage with sparse regression. FEniCSx is an open-source computing platform that solves PDEs with the finite element method. It allows for a simple representation of partial differential equations where the user defines the mesh and variables that are desired with the required settings. FEniCSx removes the numerical discretisation part for the user and is able to fully describe the PDE in a few lines of code [1]. FEniCSx has multiple examples, ranging from the 1D heat equation to the 3D Navier-Stokes. The code in the current work is based on the heat equation of the FEniCSx tutorial [2].

FEniCSx makes use of finite elements by means of the weak formulation. The weak formulation states that a partial differential equation should hold over a section of the domain rather than any point in the domain. The form used in the current work, the strong form, states that the differential equation should hold at every point. A PDE is in the weak form by multiplying the equation with a test function v and integrating it over a domain Ω, resulting in the example for a Poisson equation in

[1]For more information on FEniCSx please look at `https://fenicsproject.org`

[2]Tutorial for the heat equation can be found in `https://jsdokken.com/dolfinx-tutorial/chapter2/heat_equation.html`

$$\int_{\Omega} v\left(\frac{\partial^2 u}{\partial x_i \partial x_i} - f\right) = \int_0^1 vR_{weak} = 0 \tag{3.12}$$

In (3.12), u is said to be the trial function. The weak form is sought to be stated as a bilinear form $a(u,v)$ and a linear form $L(v)$ as

$$L(v) = a(u,v) \tag{3.13}$$

The trail function u can be described by a series of basis functions, as

$$u = \sum_{i=0}^{N} c_i \phi_i \tag{3.14}$$

In the current work, a one dimensional Lagrange family of elements using the Galerkin method is used, meaning that $v = \phi_j$. The one dimensional Lagrange family of elements are localised piecewise hat functions (linear increasing to a point followed by a linear decrease), meaning that they are in the Sobolev space, defined as

$$H^1_{\Omega} = \left\{ f : \int_{\Omega} |f|^2 + \left|\frac{\partial f}{\partial x_i}\right|^2 \mathrm{d}x < \infty \right\} \tag{3.15}$$

The Sobolev space states that a function and it is derivative are both bounded integrals in Ω. The second derivative of u will be equal to zero and the order of the to be evaluated derivative cannot be greater than one. For the second derivative, differentiation by parts is used to eliminate the second order derivative and create an equivalent form in terms of the first derivative of both the test and trail functions.

Based on the FEniCSx tutorial by Langtangen and Logg [104], a step-by-step derivation of the $\psi - \phi$ system weak form is given in Appendix B. The interested reader is recommend to read more on finite elements in the book of Brenner and Scott [105] and the implementation of finite elements in FEniCSx in Langtangen and Logg [104]

3.3 The Model Training Phase

The sparse regression algorithm used is the PDE FIND algorithm by Rudy et al [40] [3]. The algorithm is well-tested and has shown great promise in other literature works (Abramovic et al [43], Zhao et al [44] and more). PDE FIND is easy to implement and understand, and has a low computational cost. As the report won't tackle noise, the ordinary PDE FIND algorithm should be suited. Sparse linear regression is chosen over SBL and Neural networks routines as it is a more efficient and a more controlled option. The PDE FIND algorithm is not based on an expensive cost function (as is the case for SBL) and is controlled by the user rather than the black box neural networks.

PDE FIND will find coefficients related to a pre-selected set of candidates for the target. The candidates, also used as candidate library in the current work, are functions augmented with data to represent the possible solutions for a certain equation. The target

[3]The code for the PDE FIND is found on `https://github.com/snagcliffs/PDE-FIND`

is the data where PDE FIND will regress too. More on PDE FIND can be found in the literature section, Section 2.2.2 in Chapter 2.2.

A flow diagram of the PDE FIND algorithm by Rudy et al [40] can be found in Figure 3.4. The original work of Rudy et al [40] is split into TrainSTRidge and STRidge. STRidge performs the ridge regression and cuts off the coefficients smaller than a certain tolerance. Ridge regression, as explained in Section 2.2.1 of Chapter 2.2, shrinks the coefficients of the regression, effectively reducing the correlation between the coefficients. A hard tolerance is used to eliminate coefficients that are lower than the tolerance. The important hyper parameter for ridge regression is λ and for the hard cut off tol.

TrainSTRidge predicts an error using an L_0 norm and a different hyper parameter, being μ. In the work, μ is set at 0.001, as default by the PDE FIND algorithm. The L_0 increases the error for non-sparse results and effectively a model must be found to overcome the non sparsity error. A search algorithm is applied in TrainSTRidge for the tol, where a different tol is calculated at each iteration to obtain a different sparse result from STRidge.

To start the process, the data is split into train and test data and a baseline predictor for the coefficients is created with a least squares (LSQ) prediction. The algorithm has a hard coded limit, meaning that no convergence error is sought but rather a maximum amount of iterations are applied. The reason is that for a certain set of hyper parameters, a convergence criteria might not be met, while a result can still be usefull. The amount of iterations in TrainSTRidge are kept at 25 and only two hyper parameters are adjusted, being the λ and tol.

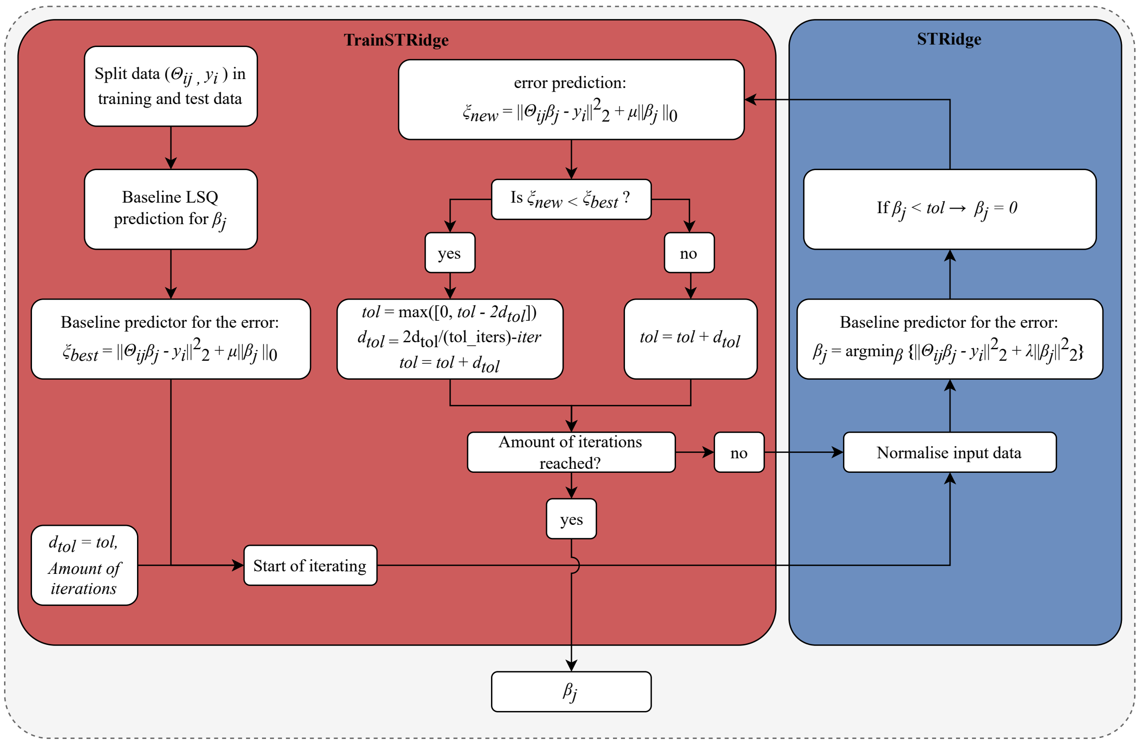

TrainSTRidge
STRidge
Split data (Θ_{ij}, y_i) in training and test data
Baseline LSQ prediction for β_j
Baseline predictor for the error: $\xi_{best} = \|\Theta_{ij}\beta_j - y_i\|^2_2 + \mu\|\beta_j\|_0$
$d_{tol} = tol$, Amount of iterations
Start of iterating
error prediction: $\xi_{new} = \|\Theta_{ij}\beta_j - y_i\|^2_2 + \mu\|\beta_j\|_0$
Is $\xi_{new} < \xi_{best}$?
yes
no
$tol = \max([0, tol - 2d_{tol}])$ $d_{tol} = 2d_{tol}/(\text{tol\_iters})\text{-}iter$ $tol = tol + d_{tol}$
$tol = tol + d_{tol}$
Amount of iterations reached?
no
yes
If $\beta_j < tol \rightarrow \beta_j = 0$
Baseline predictor for the error: $\beta_j = \text{argmin}_\beta \{\|\Theta_{ij}\beta_j - y_i\|^2_2 + \lambda\|\beta_j\|^2_2\}$
Normalise input data
β_j

Figure 3.4: Compact description of the PDE FIND algorithm of Rudy et al [40].

The current work can be seen as a continuation of SpaRTA. In SpaRTA, the residual was analysed and a correction was made to the Boussinesq assumption, making an effective

EARSM, and a correction was made to the production term in the k-equation. Both corrections are local, and in the current work sparse regression is analysed for non-local corrections. In particular, an additional transport equation will be analysed and PDE FIND will be assessed for the performance to find transport equations. PDE FIND uses as a the target the time derivative data and in the current work the target is the material derivative, resulting in the PDE FIND system as

$$\frac{D\phi}{Dt} = \frac{\partial \phi}{\partial t} + u_i \frac{\partial \phi}{\partial x_i} = f\left(\phi^0 f^0, \dots \phi^n f^0, \phi^0 f^0, \dots \phi^n f^1 \dots \phi^0 f^m, \dots \phi^n f^m\right) \tag{3.16}$$

In (3.16), f is a potential candidate function. The candidate function is always given as rotational invariant in the current work, as the inherent characteristic of turbulence is that it is rotational invariant.

The PDE FIND algorithm will be subject to six different test cases, a summation can be found below.

1. **Partial PDE** The partial PDE consistes of the simple R_ϕ from (3.3), with the main goal of ensuring PDE FIND works as expected.
2. **Full PDE** The full PDE has more correlated R_ϕ function from (3.4) and is intended as an analogous system to turbulence modelling.
3. **First incomplete library case** The first case lacks the quartic term and is a small term contributing to the solution. The aim of the experiment is to analyse the performance of PDE FIND where most of the governing terms are known.
4. **Second incomplete library case** The second case lacks the cross diffusion term that has a greater share in the overall solution of the $\psi - \phi$ system, intending to analyse the performance of PDE FIND where a large part of the governing terms of the result are unknown.
5. **Third incomplete library case** The third case is similar to the second case, with the addition of extra terms that are highly correlated with the missing cross diffusion term. The aim is to see if the replacement terms are a good alternative selection for PDE FIND to create a decent model.
6. **Fourth incomplete library case** The fourth case replaces the strain rate driven candidates (the production candidates) by a different strain rate candidate. The goal is to mimic not having the full picture of the invariants and analyse what the effect might be if the wrong invariants are part of the candidate library.

While the cases 'Partial PDE' and 'Full PDE' analyse more the performance of the PDE FIND for a highly correlated systems, the other four cases relate it to turbulence modelling and the unknowns between RANS and DNS. Each case has increasing difficulty, with cases two and three being similar. The additional candidates in case three are

$$\phi^n \frac{\partial \psi}{\partial x_i}\frac{\partial \psi}{\partial x_i} \tag{3.17}$$

$$\phi^n \frac{\partial^2 \psi}{\partial x_i \partial x_i} \tag{3.18}$$

while the replacement candidate for the fourth case is

$$\phi^n S_{ij} S_{ij} S_{ij} S_{ij} \tag{3.19}$$

A full overview of the candidate library are found in Appendix D.

Each case has different criteria, therefore the PDE FIND procedure is performed for a selection of hyper parameters λ and tol. A list of all the ran cases is provided in Appendix E. For the partial and full case, all three velocity cases are regressed. For the incomplete library cases, only the vortex and exponential velocity cases are regressed. From the six regression cases, a few models (2/3) per velocity case are chosen for further analyses.

For all cases, the amount of samples is equal to 1000, distributed randomly over a search window with the most important information. The search windows per case can be found in Appendix C. For some cases, multiple search spaces are used to improve search performance as the author found an improvement in PDE FIND performance for different search windows. For the shear case and the exponential case, a search window is employed in a non significant gradient area and in a significant gradient area. The vortex case always has the same search window.

3.4 The Training Validation Phase

The models chosen in the model training step are determined by the a priori and a posteriori error, given in the report in an a priori-a posteriori diagram. The a priori error is defined as the error that the functional form obtains from the linear regression routine compared to the true functional form (being Equations (3.2), (3.1), (3.4) or (3.3)).

The a priori error is built up in two parts: The identifying part and the correcting part. The correcting part will give the a priori error an increase by how much the coefficients differ compared to the true form (for the normalised coefficient) while the identifying part will give the a priori error an additional error of one per term not included or lacking compared to the true functional form. An a priori error lower than one will correspond to the return of the true model while an a priori error greater than one corresponds to a model lacking a function or having functions too many compared to the true functional form. The a priori is given as

$$\xi_{prio} = \sum_{i=0}\left(\sum_{j=0}\left\|\frac{c_{i,model} - c_{j,true}}{c_{j,true}}\right\|_2^2 \delta_{ij} + \|c_{i,model}\|_0 - \|c_{i,true}\|_0\right) \tag{3.20}$$

In (3.20), $c_{i,model}$ are the coefficients resulting from the PDE FIND algorithm, $c_{j,true}$ the coefficients of the ground truth, δ_{ij} is the Dirac delta that equals 1 when i is equal to j and 0 else. With how the incomplete library cases are defined in Subsection 3.3, the a priori will always exceed one. For the first, second and third incomplete case, it will be at least greater than one while for the fourth incomplete case it will be greater than three. It is chosen not to correct for the a priori error (making it able to be less than one for each test case), as an a priori error lower than one corresponds to the true form and the true equation. The true functional form without the term in the candidate library might perform worse than an incorrect addition of a certain function and therefore the a priori error might give a wrong impression if corrected for the incorrect library.

The a posteriori error is given as the average error of the ground truth data versus the simulated trained model from the PDE FIND algorithm averaged for every time step. Per velocity case, trained models are selected with the a priori-a posteriori diagram. Note that not always the lowest a priori model is chosen for analyses, as sometimes a low a priori error might mean a high a posteriori error.

The trained models are cross checked with other velocity test cases. The partial models are not cross checked as the purpose of the partial model is to check if the PDE FIND algorithm works as intended. For the full cases, all cases are cross checked with one-another while for the incomplete library cases, only the shear case is cross checked by the trained exponential and vortex models. The best performing model from each trained velocity case will have further analyses in the model form and the regression performance.

4. Data Generation

The chapter 'Data Generation' discusses the (future) train and test data for both the partial and full PDE cases. All three velocity cases for both the partial and full PDE system didn't convergence, but tend to go to convergence and would reach convergence if more numerical steps would have been included.

For the budget, the R_ϕ term drives the initial solution, and for all cases becomes less important with increasing time. The ψ values lack behind on the ϕ values for both production, dissipation and the material derivative. Depending on the velocity case, R_ϕ varies if it has a positive or negative contribution with increasing time. For the shear cases, the convection on the left hand-side boundary hasn't reached the right hand side for the value of ψ. For both the vortex and exponential cases, ϕ and ψ are developed fully.

The functional forms related to f^4, with in particular f^4, have the greatest contribution to the overall solution in terms of field averaged performance. The single f^4 term has a contribution as high as 57 % to the overall solution in the full PDE shear velocity case and 63 % in the partial PDE shear velocity case. In terms of the greatest amplitude, the f^2 term has the greatest contribution in the shear and exponential cases for both the partial and full PDE. A more muted response is present in the vortex cases where f^4 has the greatest amplitude response.

The chapter reads as follows, a note on the notation is given in Section 4.1, while the results for the cases are discussed in Section 4.2. Section 4.3 discusses the specific contribution of the terms while Section 4.4 will briefly discuss some important parameters obtained from the data generation.

4.1 A Note on Notation

In the previous chapters, the partial differential equations have been written in index notation with complete and written out derivatives. For ease, in the current section, a different style will be used for the differential part of the equations. Every possible function used throughout the work will have a function form as f^x, leading to the forms needed to represent the true equations as

$$f^0 = 1 \quad (4.1) \qquad f^1 = \psi \quad (4.2) \qquad f^2 = \frac{\partial^2 \phi}{\partial x_i \partial x_i} \quad (4.3)$$

$$f^3 = \frac{\partial \phi}{\partial x_i}\frac{\partial \psi}{\partial x_i} \quad (4.4) \qquad f^4 = S_{ij}S_{ij} \quad (4.5) \qquad f^5 = \frac{\partial \phi}{\partial x_i}\frac{\partial \phi}{\partial x_i} \quad (4.6)$$

These forms are used in combination with the material derivative to make the ψ equation become

$$\frac{D\psi}{Dt} = -1.5\psi f^1 + 0.035 f^7 + \phi f^4 \tag{4.7}$$

and ϕ become

$$\frac{D\phi}{Dt} = -0.75\phi^2 f^0 - 3\phi f^1 + 0.1 f^2 + f^3 + (1.5 + \phi + 0.5\phi^2) f^4 + 0.35 f^5 \tag{4.8}$$

Similar to the full PDE equation of ϕ (4.8), the partial PDE equation of ϕ is rewritten as

$$\frac{D\phi}{Dt} = -0.75\phi^2 f^0 + 0.1 f^2 + f^3 + (1.5 + \phi) f^4 + 0.35 f^5 \tag{4.9}$$

The additional and replacement candidate functions, discussed in Section 3.3 have a similar change in notation, making the additional functions

$$f^6 = \frac{\partial^2 \psi}{\partial x_i \partial x_i} \tag{4.10}$$

$$f^7 = \frac{\partial \psi}{\partial x_i}\frac{\partial \psi}{\partial x_i} \tag{4.11}$$

and the replacement function

$$f^8 = S_{ij} S_{ij} S_{ij} S_{ij} \tag{4.12}$$

4.2 Results of the Velocity Cases

The current section will show the data used for the regression experiment for three cases, being the vortex case, exponential case and shear case. For each case, different results are shown for the partial case and the full case. Subsection 4.2.1 discusses the results related to the vortex case, Subsection 4.2.2 the exponential case and Subsection 4.2.3 the shear case.

In the current section, budget graphs are given for ψ and ϕ to provide a clear development of the terms for the $\psi - \phi$ system. The area of which the budget graphs are taken are ten cells away from the border. Due to the boundary conditions and a zero initial condition for the shear and exponential velocity cases, a sharp gradient is present in some sub functions that are subject to numerical noise. For example, at $t = 0$ s, a step is present between the boundary and the field, resulting in an oscillating value of the diffusion term. Note that the change has barely an effect for the result of the budget graphs. R_ϕ is taken as a combination of terms, such as given in Subsection 3.2.1 of Chapter 3. So, when analysing the R_ϕ term, one should keep in mind that R_ϕ consists of five terms for the full PDE and three terms for the partial PDE.

4.2.1 The Vortex Velocity Case

Figures 4.2 and 4.1 shows the propagation of ϕ and ψ for different moments of time for the partial case. A clear core can be seen around the centre, where the gradient of velocity is the greatest as per Figure 3.2. ψ also changes behaviour, going from an initial more donut structure to a peak like structure and has a smaller area of influence than ϕ.

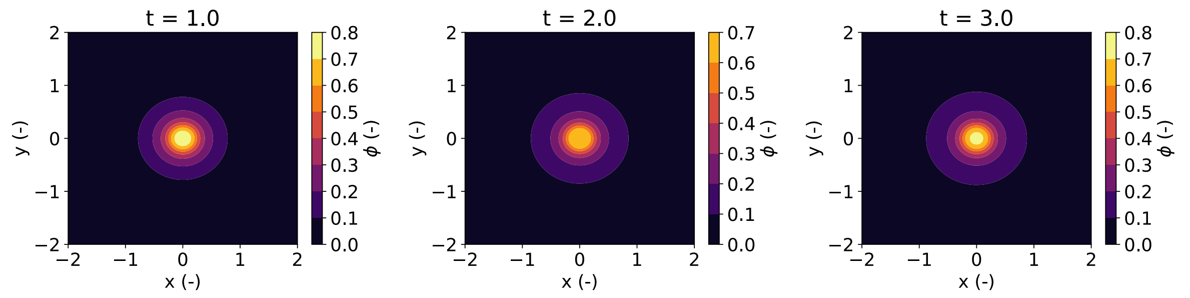


Figure 4.1: The time development of the ϕ field with for the partial PDE vortex case with $t = 3.0$ s being equal to the final time moment.

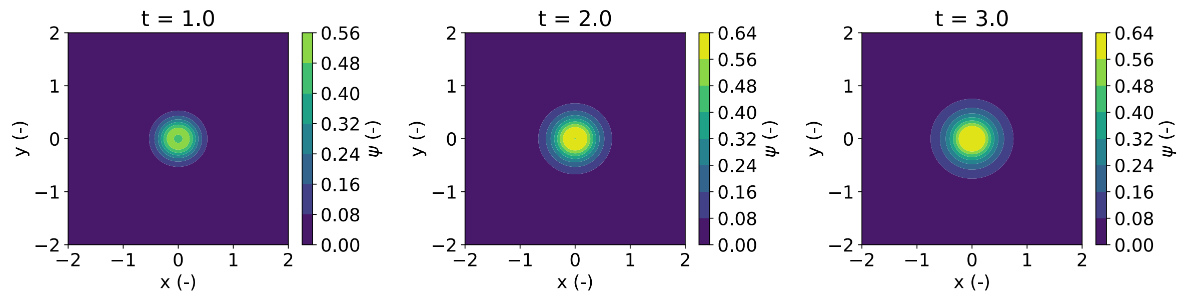


Figure 4.2: The time development of the ψ field with for the partial PDE vortex case with $t = 3.0$ s being equal to the final time moment.

Figure 4.3 shows the ϕ and ψ budget for the partial vortex case in time. Both ϕ and ψ do not reach steady state but are close, in particular for ϕ. R_ϕ is initially dominant, due to the f^4 contribution that is driven from the velocity, but becomes less important as the cross diffusion term in R_ϕ, f^3, becomes more powerful. A similar effect is found for the destruction in ϵ_ϕ, that increases strongly till a t of 0.5 s, but then stabilises. At the final t, P_ϕ is the most dominant process, from the given four processes. For ψ, the destruction is constantly increasing in time and while the production has stabilised, the destruction makes the solution not yet converged and in general ψ is lacking behind on ϕ. In both ψ and ϕ, the diffusion does not provide a contribution. Diffusion reduces the peak of the solution and spreads the solution over the domain and as the solution is axis symmetric and not losing significant values of ϕ and ψ over the border, a zero average is found.

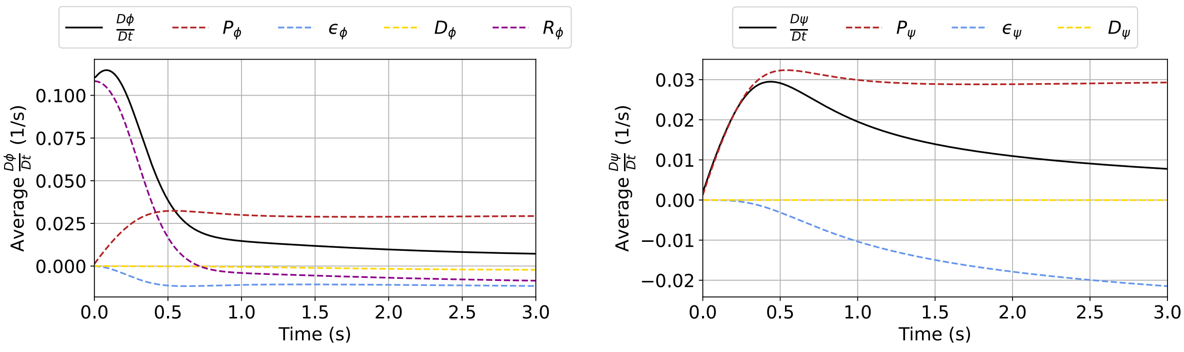


Figure 4.3: The budgets of ϕ and ψ for the partial PDE system for the vortex velocity case over time, with on the left ϕ and on the right ψ.

For the full PDE system, Figures 4.4 and 4.5 show the development of ϕ and ψ for the vortex velocity case. Similar behaviour is present compared to the partial case, but for increasing time, the peak is not at the centre but rather at the location of the velocity peak. The addition of the cross destruction term ϕf^0 to R_ϕ could provide an explanation, as the term damps the highest values the most.

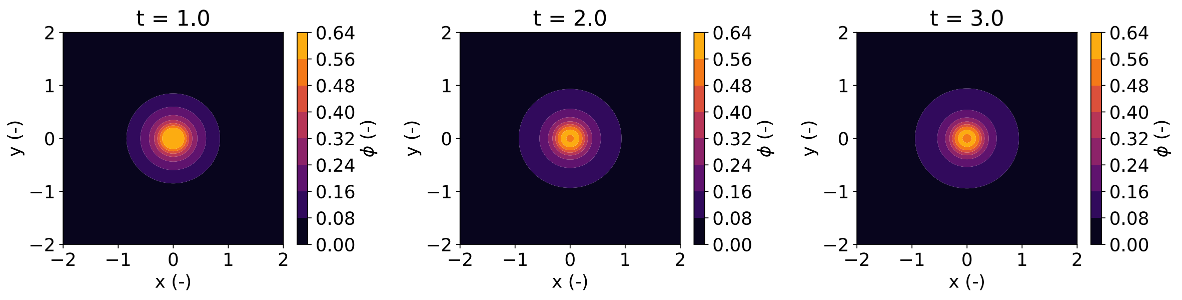


Figure 4.4: The time development of the ϕ field with for the full PDE vortex case with t = 3.0 s being equal to the final time moment.

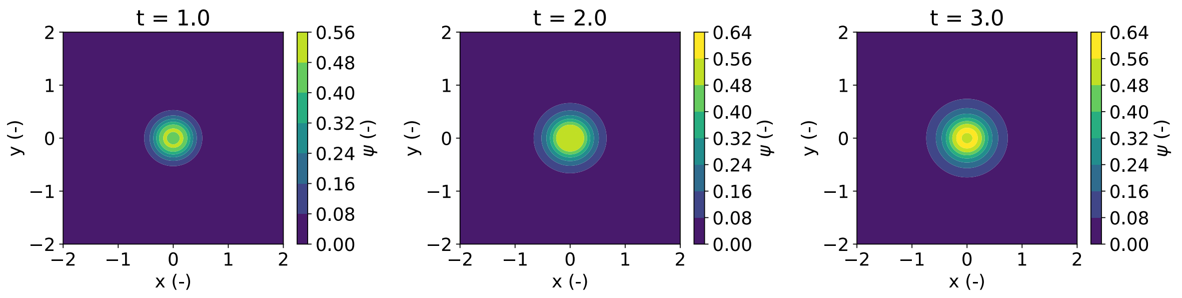


Figure 4.5: The time development of the ψ field with for the full PDE vortex case with t = 3.0 s being equal to the final time moment.

Figure 4.6 shows the ϕ and ψ budget for the full vortex case in time. The solutions do not reach steady state but are close, similar to the partial PDE. The effect of R_ϕ is less pronounced, what might be due to the addition of the destruction term ϕf^0 to the R_ϕ term. When steady state is reached, the dominant contribution is still P_ϕ. While R_ϕ is a negative contribution for the partial PDE at the latest time step, R_ϕ becomes a positive contribution in the full PDE case.

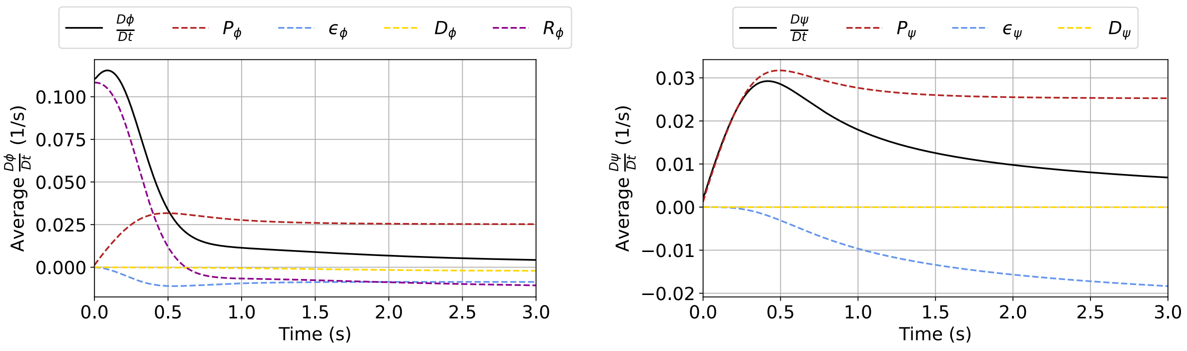


Figure 4.6: The budgets of ϕ and ψ for the full PDE system for the vortex velocity case over time, with on the left ϕ and on the right ψ.

4.2.2 The Exponential Velocity Case

Figures 4.8 and 4.7 shows the propagation of ϕ and ψ for different moments of time for the partial exponential case. For ϕ, at the location of the greatest velocity gradient, the value of ϕ is greater than the boundary condition on the left and keeps increasing within the given time frame in the domain until steady state. The same is happening for ψ, where the production term keeps on increasing the value of ψ. The main reason is that the equilibrium between the terms is greater than the value of the boundary condition.

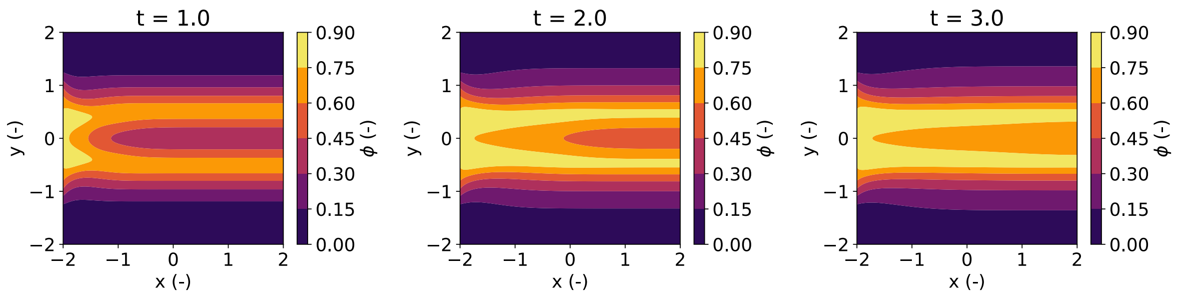


Figure 4.7: The time development of the ϕ field with for the partial PDE exponential case with $t = 3.0$ s being equal to the final time moment.

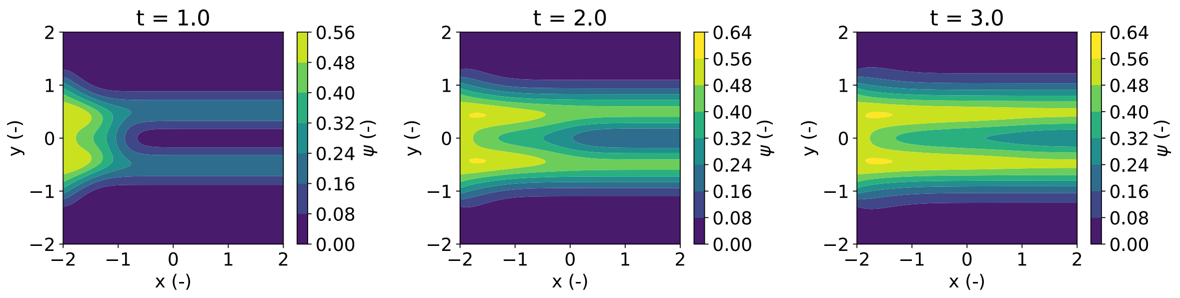


Figure 4.8: The time development of the ψ field with for the partial PDE exponential case with $t = 3.0$ being equal to the final time moment.

In Figure 4.9, the ϕ and ψ budget for the partial exponential case in time can be found. Similar to the vortex case, R_ϕ is initially dominant, due to the f^4 contribution that is driven by the velocity. R_ϕ becomes less important in time, similar to the vortex case due to the cross diffusion, f^3. Contrary to the vortex case, ϵ_ϕ is more dominant than P_ϕ and at the final t, R_ϕ is positive and ϕ has reached steady state solution while ψ is still varying. As ϕ has reached steady state, the production of ψ P_ψ has reached steady state as well and only the destruction ϵ_ψ influences the main solution. The diffusion is more present in the current case than the vortex case for the average plot, but doesn't provide a large contribution to the overall solution.

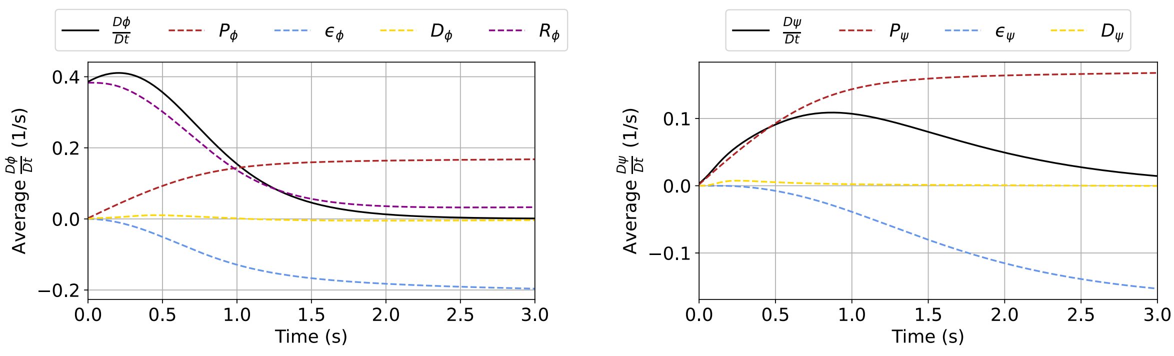


Figure 4.9: The budgets of ϕ and ψ for the partial PDE system for the exponential velocity case over time, with on the left ϕ and on the right ψ.

For the full PDE, Figures 4.10 and 4.11 show the development of ϕ and ψ for the exponential velocity case. For ϕ, a build up happens away from the left border (where the Dirichlet boundary condition is enforced) and moves along the streamline to a lower steady state solution, creating a small dip in the solution, clear in $t = 2.0$ s. The ψ solution does not have such behaviour, rather progressing the value of ψ along with the convection of the flow and having a lower production of ψ compared to the partial case. The cross destruction ϕf^1 could be leading, reducing the production of ϕ at the point of the greatest gradient.

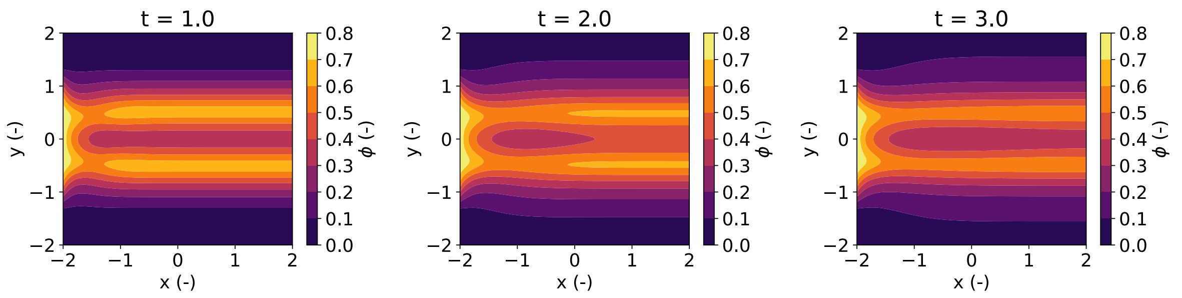


Figure 4.10: The time development of the ϕ field with for the full PDE exponential case with $t = 3.0$ s being equal to the final time moment.

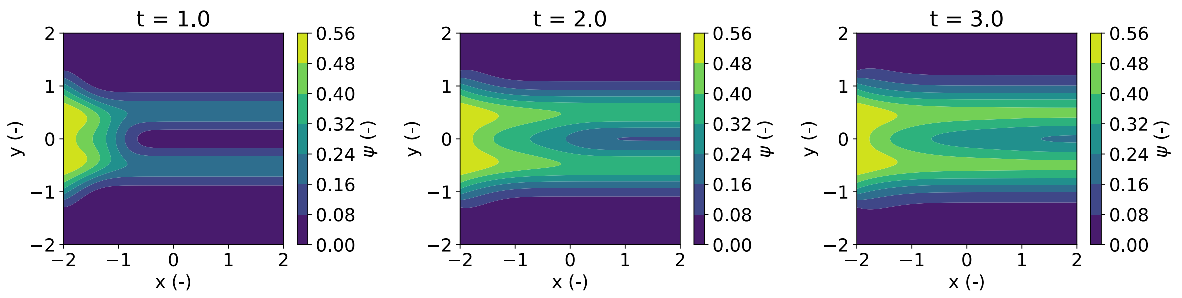


Figure 4.11: The time development of the ψ field with for the full PDE exponential case with $t = 3.0$ s being equal to the final time moment.

Figure 4.12, shows the ϕ and ψ budget for the full PDE exponential velocity case in time. Similar to the partial PDE result, R_ϕ is initially dominant, but loses importance quicker in the full PDE case compared to the partial PDE case. Similar to the vortex case, the addition of ϕf^1 in R_ϕ could explain such behaviour as that is a destruction term that becomes more important with an increase in ϕ and ψ. For the full case, an increase in ϕ is present that later decreases, resulting in $\frac{D\phi}{Dt}$ crossing the zero y-axis and an overall peak of ϕ being found before convergence. A peak is found in the production term in ψ P_ψ at that location and becomes less after that peak, but it is not large and the behaviour is not found for ψ. It is also reflected in the destruction term of ϕ ϵ_ϕ, that becomes less important with time. In general, the solution will reach steady state if ran long enough.

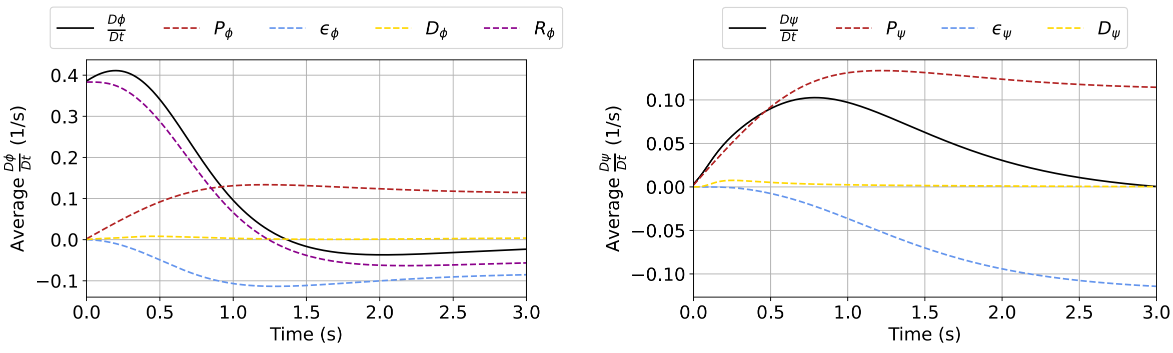


Figure 4.12: The budgets of ϕ and ψ for the full PDE system for the exponential velocity case over time, with on the left ϕ and on the right ψ.

4.2.3 The Shear Velocity Case

Figures 4.14 and 4.13 shows the propagation of ϕ and ψ for different moments of time for the partial PDE shear velocity case. For ψ, the main core has not convected to the end of the domain yet and is not fully developed while ϕ has convected and is more developed. The asymmetry is clear, as is expected with the asymmetric velocity case. The peak value of ψ and ϕ at $y = 0$ is visible due to the gradient of the velocity approaching infinity (From Section 3.2.1 of Chapter 3).

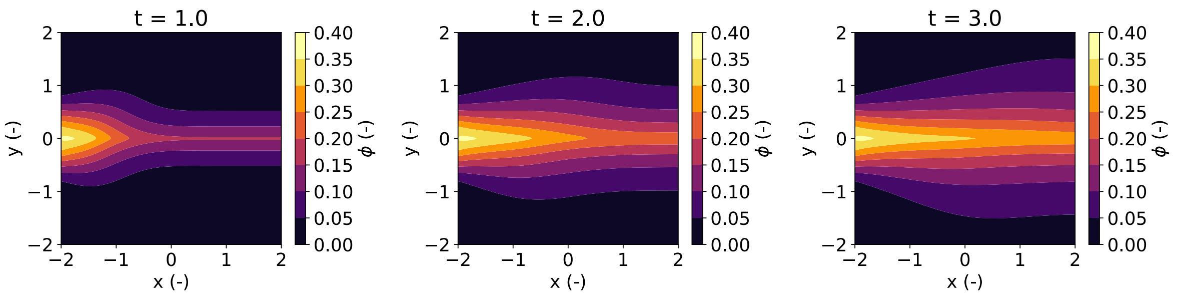


Figure 4.13: The time development of the ϕ field with for the partial shear PDE case with $t = 3.0$ s being equal to the final time moment.

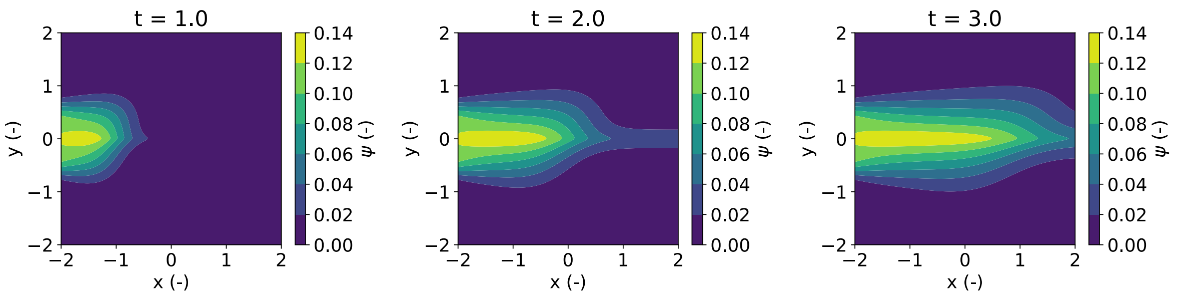


Figure 4.14: The time development of the ψ field with for the partial shear PDE case with $t = 3.0$ s being equal to the final time moment.

Figure 4.15 shows the ϕ and ψ budget for the partial shear case in time. The curves are different than the vortex and exponential velocity cases, but R_ϕ is still dominant from the onset due to the f^4 term. Different to the other cases, is that R_ϕ is significantly more dominant than the production P_ϕ and destruction ϵ_ϕ. An initial bump in the diffusion is also found at a low time value. The diffusion spreads the solution from the boundary in the main field and is greater with a substantial gradient, having a greater contribution earlier in the solution. As the contribution of ϵ_ϕ and P_ϕ are significantly lower, the effect of the diffusion is more pronounced in the current case than in the exponential case. The solutions will go to steady state if ran long enough, but note that in the current case it is not close, with ψ in particular . In ψ, the destruction ϵ_ψ is far from steady state and has yet to reach steady state. In general, the build up of the processes in the shear case is slower than the vortex and exponential case.

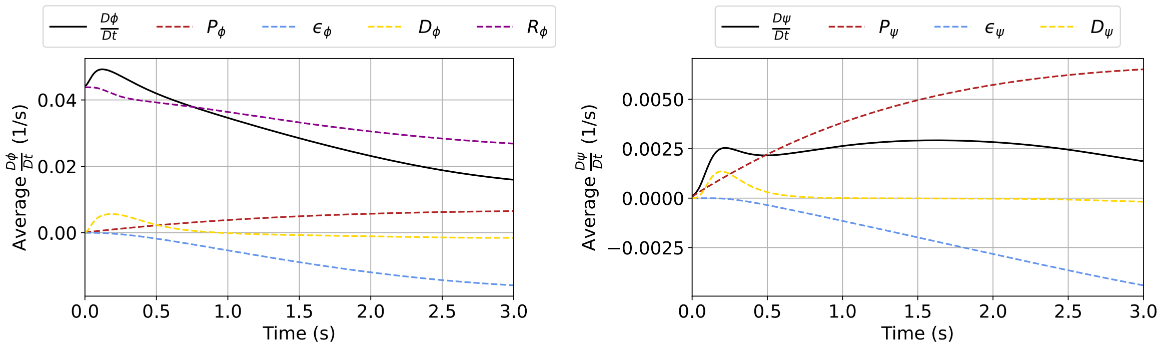


Figure 4.15: The budgets of ϕ and ψ for the partial PDE system for the shear velocity case over time, with on the left ϕ and on the right ψ.

For the full PDE, Figures 4.16 and 4.17 show the development of ϕ and ψ for the shear velocity. Similar behaviour is found to the vortex and exponential velocity cases, where the location at the highest velocity gradient has a less great value of ϕ and ψ between the full and partial PDE systems. As with the partial PDE, the convection of ψ does not reach the end while for ϕ, at the right border, an increase in ϕ is present at $y = 0$. The rise can be related due to ψ not yet being fully convected to the right border having a lower influence for the cross destruction term ϕf^1 on the part of the solution, inflating the peak.

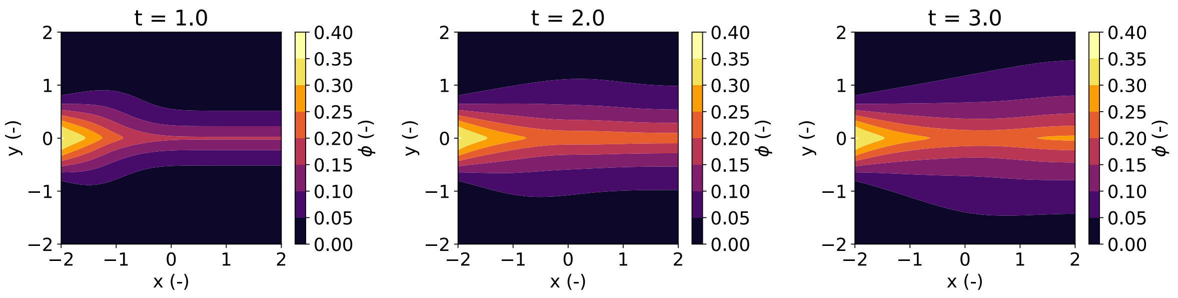


Figure 4.16: The time development of the ϕ field with for the partial PDE shear case with $t = 3.0$ s being equal to the final time moment.

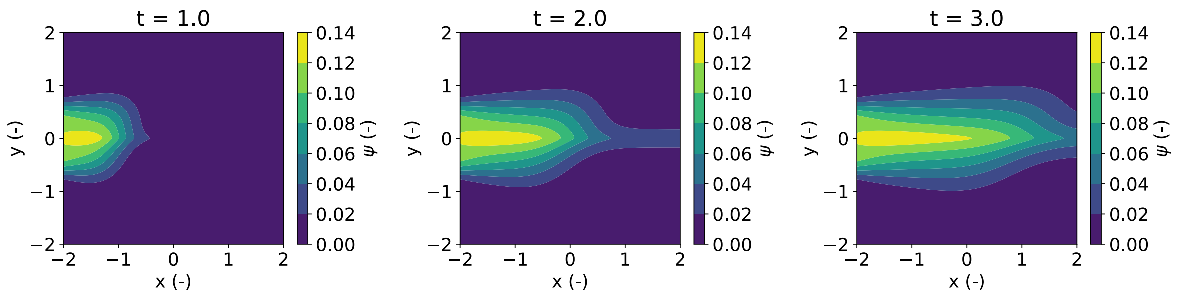


Figure 4.17: The time development of the ψ field with for the partial PDE shear case with $t = 3.0$ s being equal to the final time moment.

Figure 4.18 shows the ϕ and ψ budget for the full PDE shear velocity case in time. The trajectory is similar to the partial PDE with a less great influence on R_ϕ. Similar to the vortex and the exponential velocity cases, it is believed that it is due to the cross destruction term ϕf^1. There is a slightly earlier convergence than the partial case, but still the ψ value is far from converging. In ϕ, rather than R_ϕ being the leading term, the destruction is the leading term. Note that for both the partial as well as the full PDE, the shear case will convergence if given enough time.

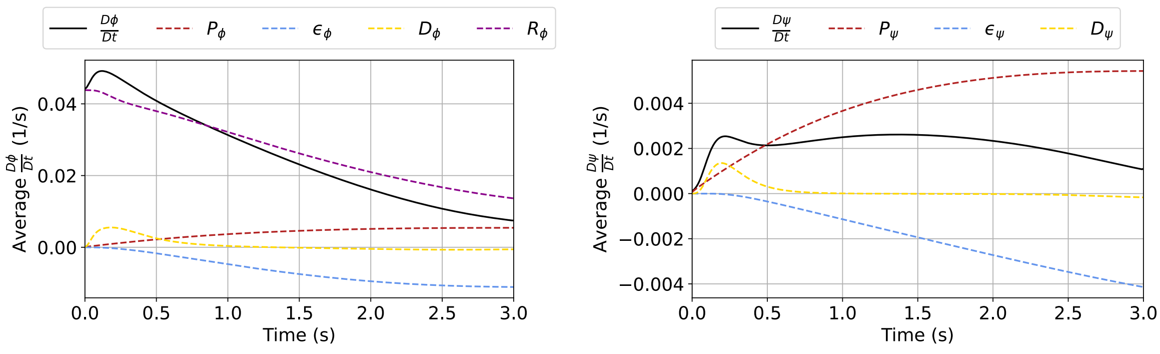


Figure 4.18: The budgets of ϕ and ψ for the full PDE system for the shear velocity case over time, with on the left ϕ and on the right ψ.

4.3 Analyses of Terms

The current section will look into the contribution of each separate function of the PDE to the overall solution. Subsection 4.3.1 discusses the contributions for the partial PDE velocity cases while Subsection 4.3.2 for the full PDE velocity cases.

4.3.1 The Partial PDE Analyses

Figure 4.19 show the averaged contributions for all time steps for the partial PDE solutions of the vortex, exponential and shear velocity cases (left to right). The greatest contribution is found by the production processes f^4, while the cross diffusion process f^3 also has a significant contribution. For the shear case, the contribution of the production driven processes f^4 are the most prevalent, with f^4 having a share of 63%. For all cases, the smallest contribution is the diffusion term f^2 and the quartic term $\phi^2 f^5$, having shares as low as 0 % for f^2 in the vortex case and as low as 0.6 % for $\phi^2 f^5$ in the shear case

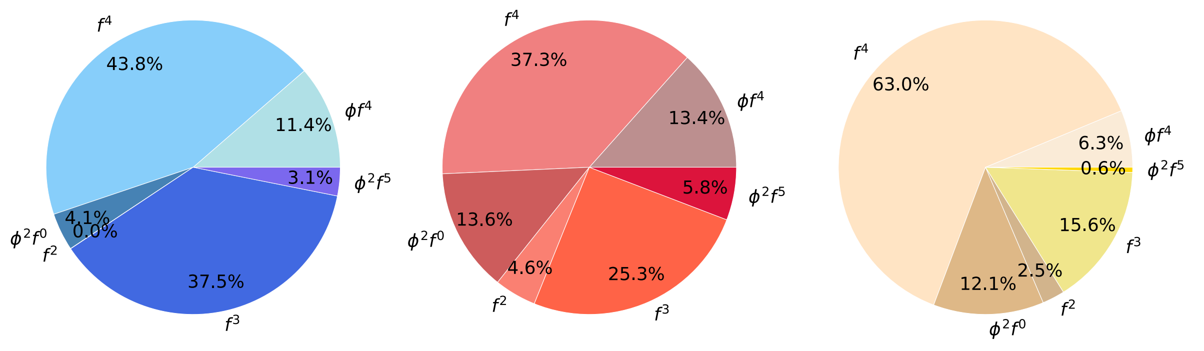


Figure 4.19: Pie chart of the averaged contribution of each step for all time steps for the partial PDE cases. From left to right, the vortex case, the exponential case and the shear case.

Figure 4.20 show the maximum amplitude of all of the functional contributions. In Figure 4.20, it can be seen that f^2 has a significant contribution for the shear and the exponential velocity cases and a more muted response for the vortex case. The f^3 and f^4 terms have quite a large contribution for the vortex case, but are more muted for other cases. The f^4 term still has the second greatest contribution for the shear case while $\phi^2 f^5$ has close to no response in the shear case and has one of the lowest responses in other cases.

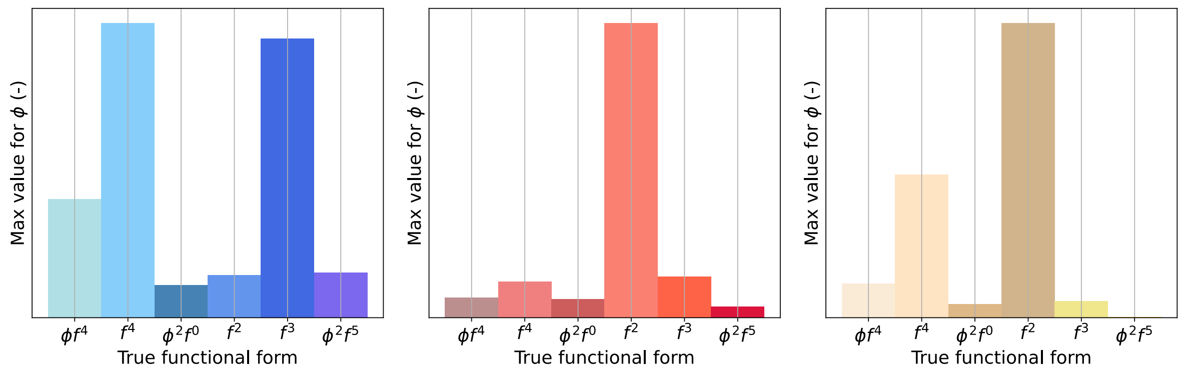


Figure 4.20: Maximum amplitude for all the separate functions in the partial PDE with from left to right the vortex case, the exponential case and the shear case.

4.3.2 The Full PDE Analyses

Figure 4.21 show the averaged contributions for all time steps for the full case of the vortex, exponential and shear velocity cases(left to right). The most dominant terms are all the f^4 related terms, with the greatest contribution being the single f^4 term with a peak of 57 % for the shear velocity case. The $\phi^2 f^5$ term is on average the lowest contributing case, having the highest average contribution in the exponential velocity case of 3.7 % and the lowest contribution in the shear velocity case of 0.5%.

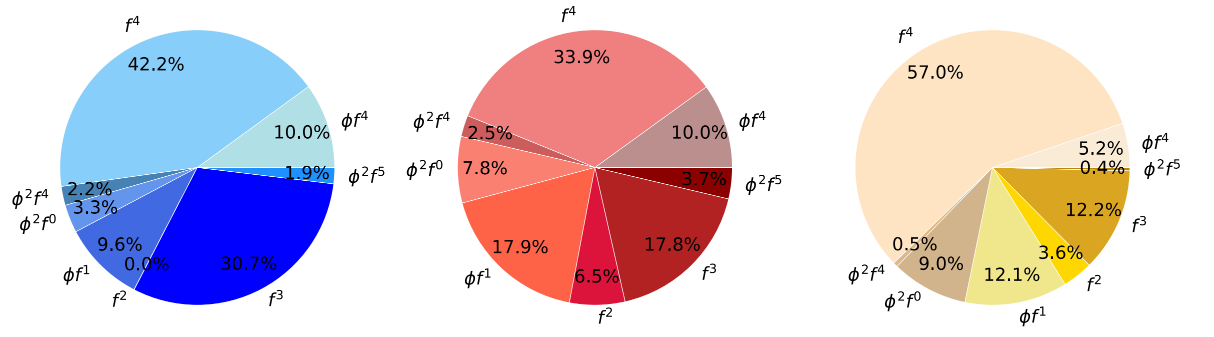


Figure 4.21: Pie chart of the averaged contribution of each step for all time steps for the full PDE case. From left to right, the vortex case, the exponential case and the shear case

Figure 4.22 shows the maximum amplitude for each case, with f^4 and f^2 having the greatest amplitudes. For the vortex case, other functions such as f^3 and ϕf^4 have a significant amplitude while for the exponential case by far the largest contribution is due to f^2. For the shear case, f^4 and f^2 have the most significant amplitude, in line with the partial PDE solutions. The terms of $\phi^2 f^4$ and $\phi^2 f^0$ have one of the lowest amplitudes for all the cases.

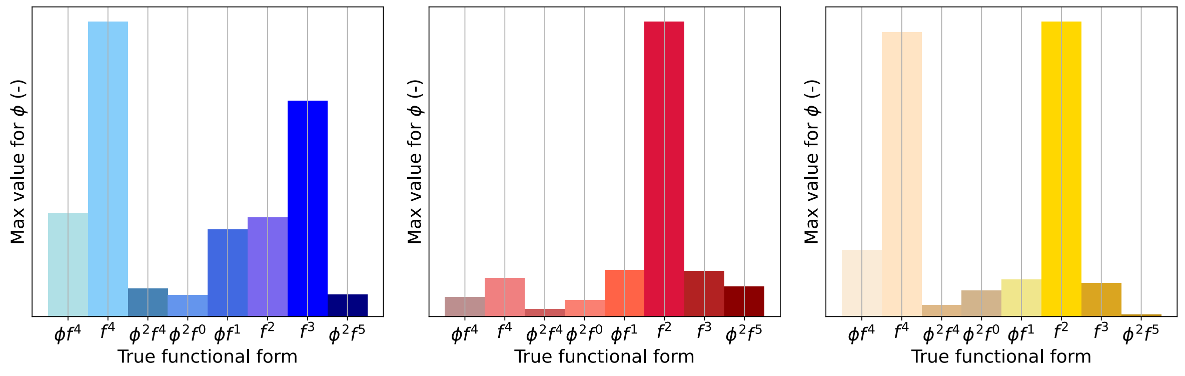


Figure 4.22: Maximum amplitude for all the separate functions in the PDE with from left to right the vortex case, the exponential case and the shear case.

For f^2 (the diffusion term) in the vortex velocity case in both PDE systems (Figure 4.21 and 4.19), the value has a contribution of 0.0 % what is in contrast to Figure 4.22. An explanation might be, that diffusion is inherently related to conservation laws and has no streamlines crossing the borders in the vortex case, no information about the diffusion is propagated over the boundary, resulting in an averaged zero contribution as the diffusion aims at evening the solution over the domain. The same analogy holds for the partial PDE system.

4.4 Statistics of the Simulations

Appendix F show the correlation coefficient and the variance for all the functions of the full PDE velocity solutions for all possible candidates (See appendix D for the candidates). The variance is the diagonal value of the covariance matrix, and the covariance matrix is given as

$$\mathrm{COV}(x_i, x_k) = \sigma_y^2 (X_{ij} X_{kj})^{-1} \tag{4.13}$$

In (4.13), COV is the covariance matrix. The standard deviation is given as σ_y and is related to the target data, the material derivative in the current work. The standard deviation is calculated for the whole target dataset. A major assumption is made that the target data is uncorrelated, what is inherently incorrect. The target data consists of multiple samples of different t and when the result changes at an earlier t, the future t is affected and hence the data is correlated. For the purpose of the covariance matrix, it is not a problem as the standard deviation functions as a scaling parameter so the resulting error should not affect the analyses.

The correlation coefficient relates each function with one another using the Pearson correlation coefficient (obtained from Hastie [84]) and is defined as

$$\rho_{cor} = \frac{\mathrm{E}[f^x f^y] - \mathrm{E}[f^x] E[f^y]}{\sqrt{\mathrm{VAR}[f^x]\mathrm{VAR}[f^y]}} \tag{4.14}$$

In (4.14), $\mathrm{E}[f^x]$ is the expectation of function f^x while $\mathrm{VAR}[f^x]$ is the variance of function f^x. In Appendix F, heat maps are given for correlation coefficients for each velocity case for the full PDE.

5. Data Regression

The chapter 'Data Regression' obtains models using the PDE FIND sparse regression algorithm for all velocity and regression cases. A selection of all the run simulations for the specific hyper parameters can be found in Appendix E with all the result found in Appendix G and a subset of models found in Appendix H. In the current chapter, the focus shifts away from the partial PDE and more towards the full PDE as the partial PDE is used as a verification process for PDE FIND.

For the partial PDE system, for both the exponential and vortex velocity cases, the complete functional form was able to be retrieved from the regression algorithm. For the full PDE system, only the exponential velocity case was able to retrieve the full functional form, for only one specific set of λ and *tol* in a specific search window. The selected models that do not have the correct model returned, showed good performance on the training data, in access of a posteriori errors lower than 1 %.

For the incomplete library cases, the case where a small contribution was lacking (the first incomplete library case) provided low a posteriori and low a priori error models. The cases where a larger contribution was lacking (the second and third incomplete library case) proved training to be more difficult. Unreliable models were found for a library with limited candidate terms, but training improved by including highly correlated candidate functions for the missing candidate. Replacing the production term (the fourth incomplete library case) with highly correlated different production terms provided poor a posteriori error trained models for the vortex trained case but decent models for the exponential trained case.

In all tests, the most recognised functions where the f^2 and the f^4 terms, being the terms with the highest amplitude response and the greatest average contribution given in Section 4.3 from Chapter 4. Even for models with a poor a posteriori error, the coefficients of these functions were reasonable. The terms with a small average contribution, such as $\phi^2 f^5$ and $\phi^2 f^4$, are often not recognised.

The data regression for the partial PDE systems with a full set of candidate functions can be found in Section 5.1 and for the the full PDE system it can be found in Section 5.2. For the incomplete candidate functions, the regression results can be found in Section 5.3.

5.1 Data Regression for the Partial PDE

For the sparse data regression, a full library of candidate functions are used, resulting in

$$\frac{D\phi}{Dt} = \sum_{n=0}^{4} \sum_{m=0}^{5} \beta_{nm} \phi^n f^m \tag{5.1}$$

In (5.1), ϕ varies from $n = 0$ to $n = 4$ resulting in a total candidate library consisting of 30 functions.

Figure 5.1 shows the log of the a posteriori error versus the a priori error for the partial case of the vortex. PDE FIND is able to return the complete functional form for ϕ from the data with an a priori error of 0.032, leading to an average absolute coefficient error of 0.53 %. The model can be found in Appendix H. Multiple other models with a higher a priori error are found with a low (less than 1 %) a posteriori error. The complete functional form still has the lowest a posteriori error for all cases. The lowest a priori error is found at a λ of 10^{-5} for a *tol* of 2.5, 5 and 7.5, as can be seen in Appendix G.

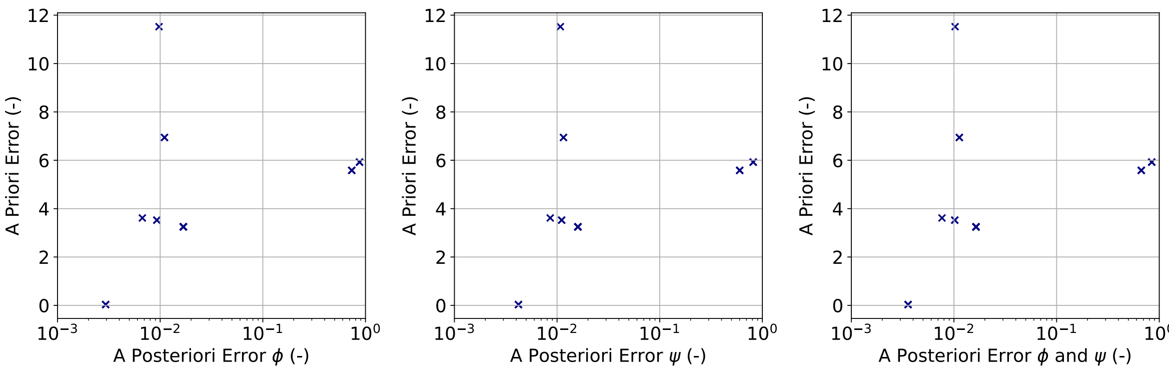


Figure 5.1: A comparison of the errors for different models resulting from the PDE FIND procedure with a complete library for the partial PDE vortex case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ.

Figure 5.2 shows the log of the a posteriori error versus the a priori error for the exponential case for the partial PDE. Similar to the vortex case, the exponential case is able to return the complete model, with an a priori error of 0.058, with an average absolute coefficient error of 0.96 %. There is a model, with a high a priori error that has a lower a posteriori error than the return of the complete model, suggesting a potential overfit, although only cross checking would confirm that. Multiple models are found with an a priori error between 2 and 5 with still decent performance (around a few %) error. The lowest a priori error that returned the complete functional form is found for multiple hyper parameters, at a λ of 10^{-4} and 10^{-5} for a *tol* of 1.5, 2.5, 5 and 7.5 and for λ of 10^{-6} only at a *tol* of 1.5, as seen in Appendix G.

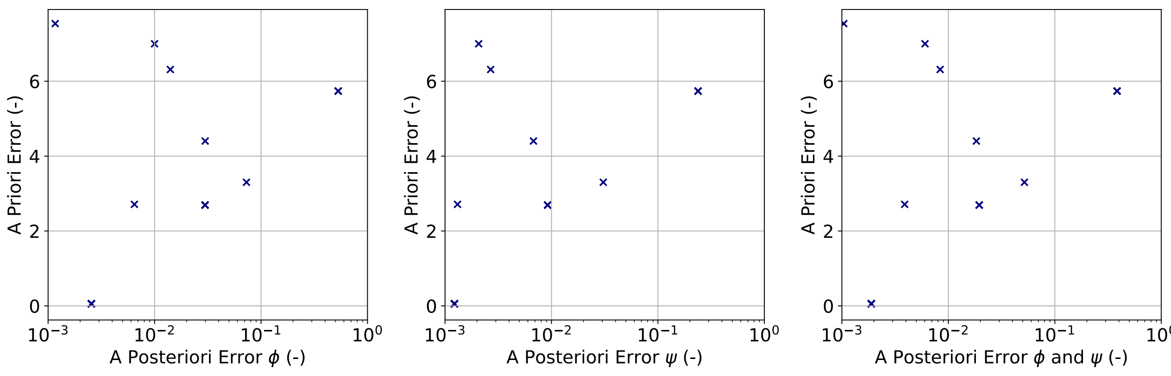


Figure 5.2: A comparison of the errors for different models resulting from the PDE FIND procedure with a complete library for the partial PDE exponential case. On the left the error relate to the a posteriori error of ϕ, in the middle the error relate to the a posteriori error of ψ and on the right the average error related to the a posteriori error of ϕ and ψ.

Figure 5.3 shows the log of the a posteriori error versus the a priori error for the shear case of the partial PDE. Compared to the previous cases, the shear case is not able to return the complete model, having the lowest a priori error of 1.16. From all cases, the lowest a posteriori error is found in the shear velocity case, being an order of magnitude smaller than the previous cases. A different search window is used in Figure 5.3 (marked red), but significantly different models are not found. A lower a posteriori error is found for the same functional form between the two search windows, as different coefficients are tuned.

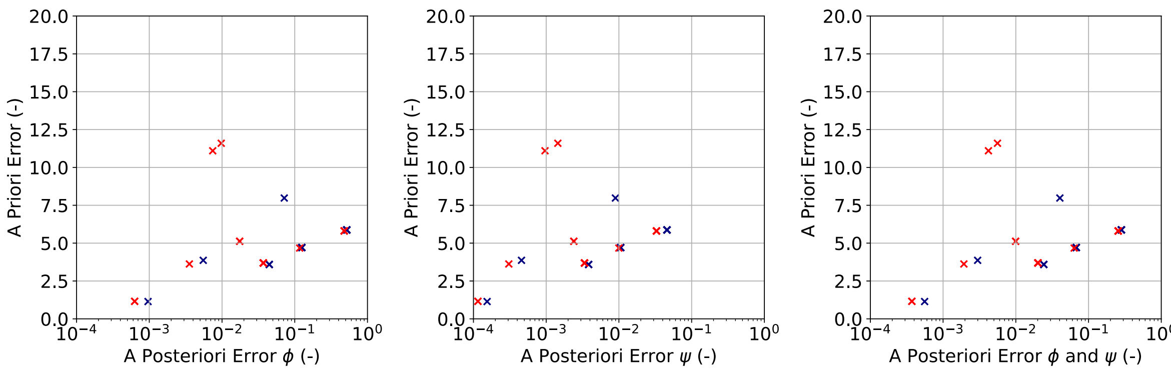


Figure 5.3: A comparison of the errors for different models resulting from the PDE FIND procedure with a complete library for the partial PDE shear case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ. Compared to all other a priori-a posteriori plots, the current plot has a different x-axis scale.

The lowest a priori model for the partial PDE shear velocity case can be found in Appendix H. The a priori error is build-up from the lack of $\phi^2 f^5$ term, the quatric term. Looking at the pie chart and amplitude chart of Figures 4.19 and 4.20, the contribution of the term in the solution is rather small. For the correct terms that are recognised, a higher absolute coefficient error is found, being 3.32 %.

PDE FIND is able to find the full model for both the exponential and vortex velocity cases. For the shear case it is not able to retrieve one functional form. PDE FIND functions well and other models have a low a posteriori error. For the fully returned functional forms, the average absolute error of the coefficients is within 1 %.

5.2 Data Regression for the Full PDE

The candidate functions used for the full PDE regression can be found in Equation (5.1) and are the same as for the partial PDE. For all three cases, the model training procedure can be found in Subsection 5.2.1 and Subsection 5.2.2 discusses a subset of viable models per trained velocity case.

5.2.1 Training of Different Models

Figure 5.4 shows the a priori-a posteriori plot for the full PDE vortex velocity case. The true model is not able to be returned, but low a priori models are found, with an a priori error of 1.37. The lowest a priori error is not the lowest a posteriori error and an lower than 1 % a posteriori error model is found with various levels of a priori errors, ranging from approximately 4.0 to 8.0.

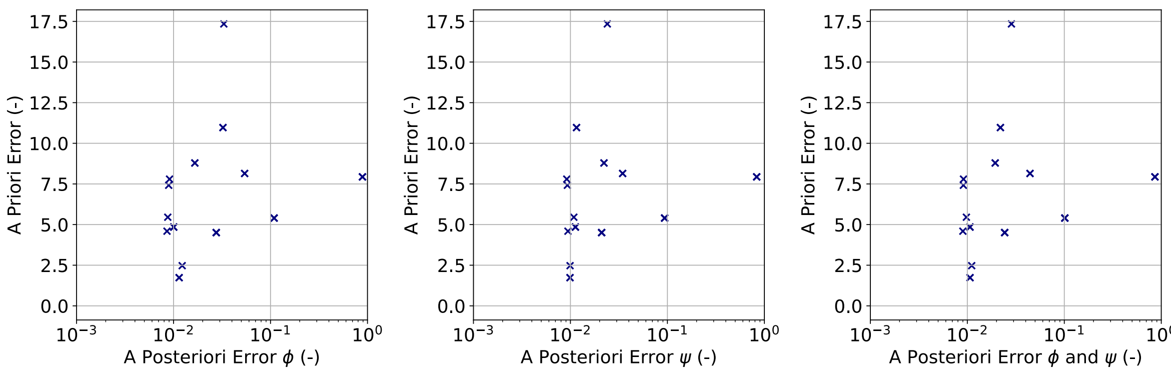


Figure 5.4: A comparison of the errors for different models resulting from the PDE FIND procedure with a complete library for the full PDE vortex case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ.

Figure 5.5 shows the exponential case. For the full PDE case, only one specific λ, *tol* and search window configuration is able to return the full PDE functional form, being at a λ of 10^{-5} for a *tol* of 1 for a large search window. For the size of the different search windows, one is advised to look at Appendix C. The fully returned model has an a priori error of 0.15 with an average absolute coefficient error of 1.8 %, what is greater than the complete returned models of the partial PDE system, from Section 5.1. Similar to the vortex case, multiple models are found over a large a priori range that have a low a posteriori training error (lower than 1 %, and even some lower than 0.5 %).

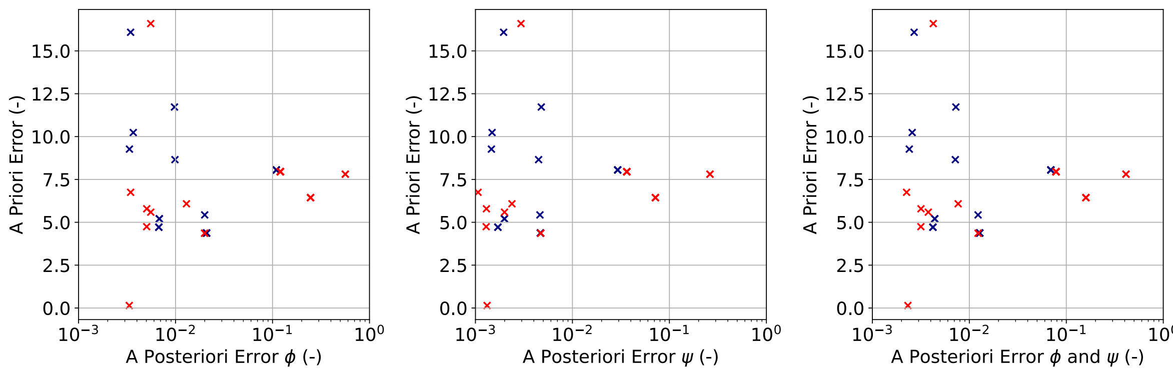


Figure 5.5: A comparison of the errors for different models resulting from the PDE FIND procedure with a complete library for the full PDE exponential case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ.

Figure 5.6 shows the a priori-a posteriori graph for the full PDE system of the shear case. The amount of models are less that have both a low a posteriori error and a low a priori error compared to the vortex and the exponential training cases. One specific models has an a priori error of 2.34, what is also the lowest a posteriori error of all the full PDE velocity cases. The results for ψ are noticeably better than for ϕ.

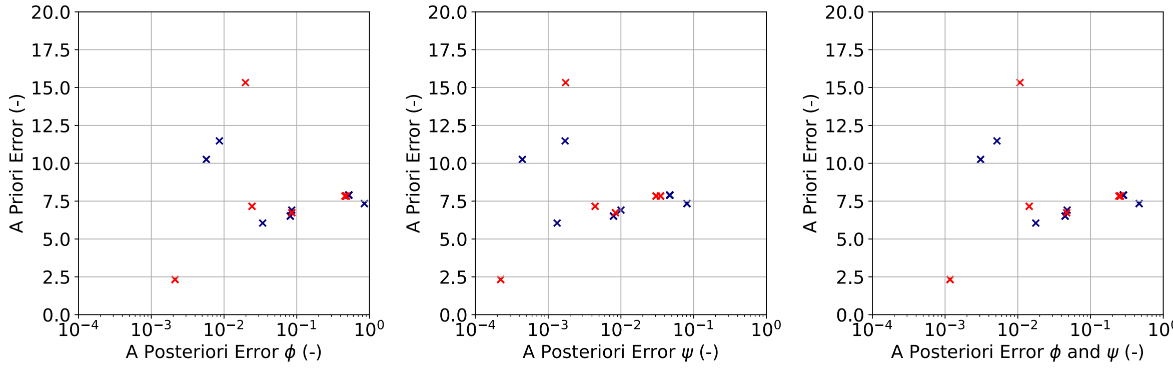


Figure 5.6: A comparison of the errors for different models resulting from the PDE FIND procedure with a complete library for the full PDE shear case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ. Note that there is a different x axis compared to others cases.

PDE FIND is able to return the full PDE for one specific set of search criteria, proofing that even between highly correlated functions, PDE FIND is able to return the full functional form. In an a posteriori sense, PDE FIND is able to return models with a low, and sometimes lower than true functional form, a posteriori error.

From Figures 5.4, 5.5 and 5.6, three models per velocity case are chosen for further analyses. The complete functional form of these models can be found in Appendix H and the a priori and a posteriori errors can be seen in Table 5.1. All chosen models show good training behaviour, being able to return the model within an a posteriori error of 1 % for a large range of a priori errors, with an exception for $M_{3,F,3}$.

Table 5.1: Candidate models from the trained cases with each trained case being specific to their own trained trained dataset with the a priori error, a posteriori error and a note what it represents in the a priori-a posteriori plot.

Model	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	**note**
$M_{1,F,1}$	4.59	0.9	0.9	Lowest a posteriori error
$M_{1,F,2}$	1.73	1.1	1.0	Lowest a priori error
$M_{1,F,3}$	7.80	0.9	0.9	Low a posteriori error at a high a priori error
$M_{2,F,1}$	6.75	0.3	0.1	Lowest a posteriori error
$M_{2,F,2}$	0.15	0.3	0.1	Lowest a priori error
$M_{2,F,3}$	9.27	0.4	0.1	lowest a posteriori error in a different search window
$M_{3,F,1}$	2.32	0.2	0.02	lowest a posteriori error and lowest a priori error
$M_{3,F,2}$	10.25	0.6	0.04	Low a posteriori error at a high a priori error
$M_{3,F,3}$	6.05	3.4	0.1	Lowest a priori error in a different search window

The simulated performance of the trained models from Table 5.1 are shown in Figures 5.7, 5.8, 5.9. All cases show near identical behaviour to the ground truth throughout time. Only $M_{3,F,3}$ has a slight offset, but as seen in Table 5.8, the error is still manageable.

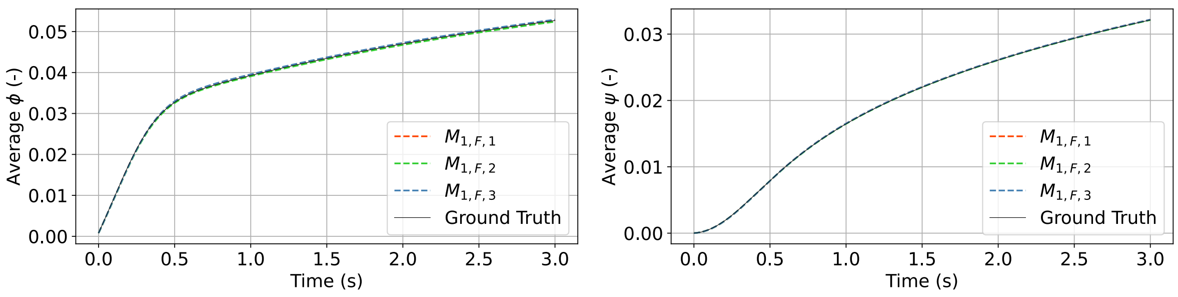


Figure 5.7: The development of the average in time of three models for the full PDE vortex case, obtained from the PDE FIND. The models are compared to the ground truth, with ϕ on the left and ψ on the right

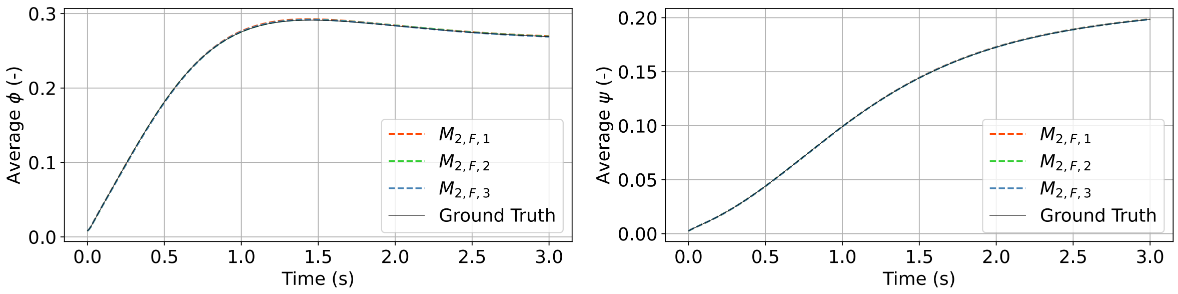


Figure 5.8: The development of the average in time of three models for the full PDE exponential case, obtained from the PDE FIND procedure compared to the ground truth, with ϕ on the left and ψ on the right

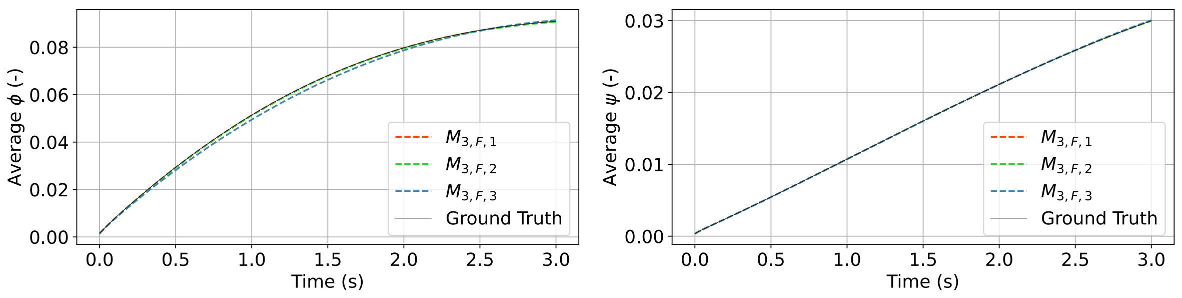


Figure 5.9: The development of the average in time of three models for the full PDE shear case, obtained from the PDE FIND procedure compared to the ground truth, with ϕ on the left and ψ on the right

Figures 5.10, 5.11 and 5.12 show the final time step of the selected models. The final time step is meant as a qualitative measure, to see if the behaviour is similar to the ground truth. As expected on the bases of Table 5.1 and Figures 5.7, 5.8, 5.9, all models perform well compared to the ground truth, Figures 4.4, 4.10 and 4.16. The model that has the largest a posteriori error in Table 5.1, model $M_{3,F,3}$, still has a good qualitative performance.

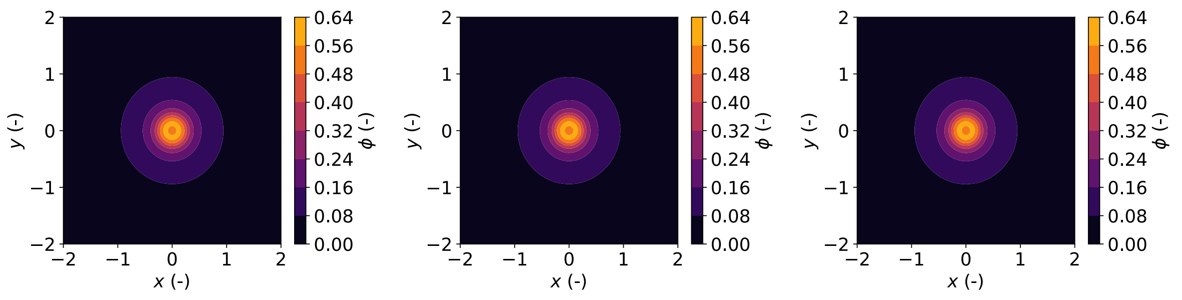


Figure 5.10: The final step for all three model for the full PDE vortex case, with on the left model $M_{1,F,1}$, in the middle $M_{1,F,2}$ and on the right $M_{1,F,3}$.

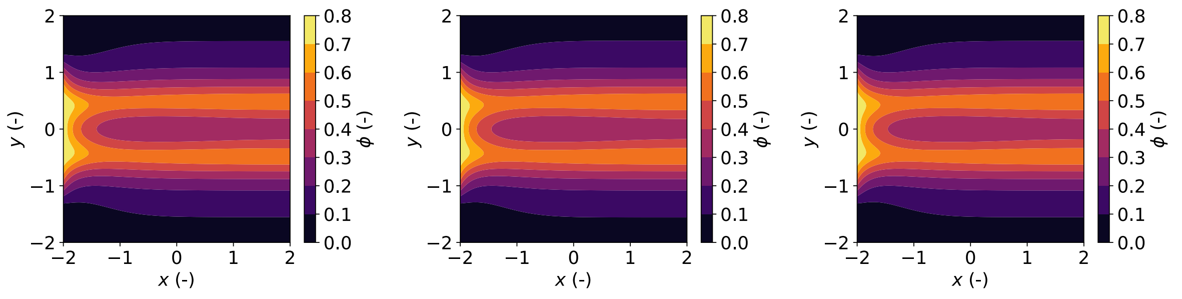


Figure 5.11: The final step for all three model for the full PDE exponential case, with on the left model $M_{2,F,1}$, in the middle $M_{2,F,2}$ and on the right $M_{2,F,3}$.

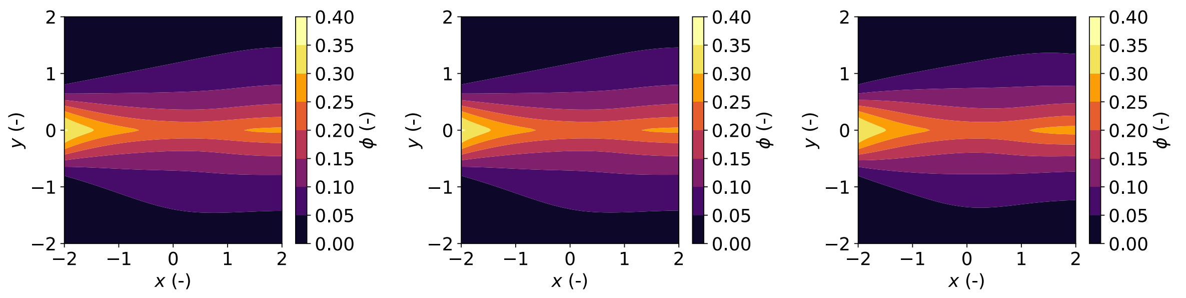


Figure 5.12: The final step for all three model for the full PDE shear case, with on the left model $M_{3,F,1}$, in the middle $M_{3,F,2}$ and on the right $M_{3,F,3}$.

5.2.2 Model Performance of the Full Cases

The difference in coefficients of the model compared to the true model is seen in Figure 5.13. All models are able to (relatively) accurately predicted the f^2 and f^4 terms. For all models except $M_{3,F,3}$, f^3 and $\phi^2 f^0$ are recognised with a respectable difference in coefficient. The ϕf^4 term is also often recognised but with a less accurate coefficient. The PDE FIND algorithm struggles to find the $\phi^2 f^5$ term, with all the models trained on the shear case data not even returning the value at all.

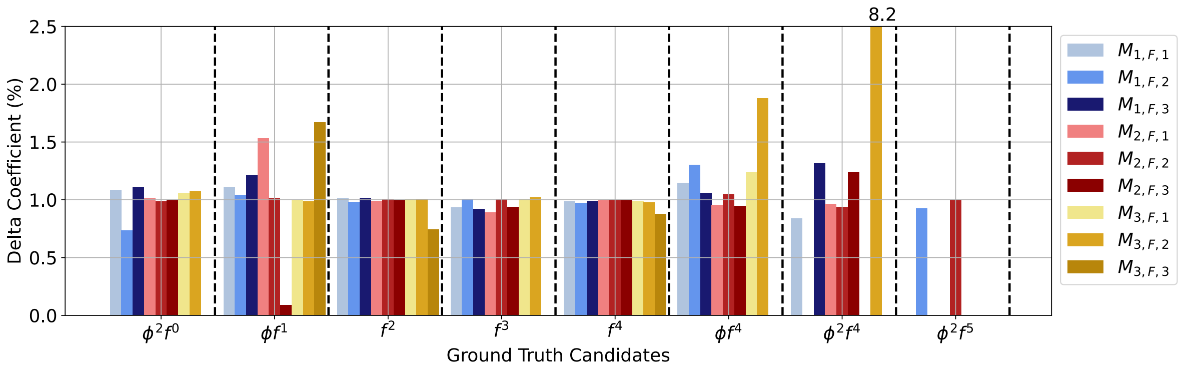


Figure 5.13: Bar plot, showing the predicted coefficient in terms of percentages compared to the true values of the true coefficients of the training data. The data shown here is for the full PDE case.

The two terms that are most accurately recognised in Figure 5.13 are f^2 and f^4, those relate as the functions with one of the greatest amplitude responses and average contributions of Figures 4.21 and 4.22. In those figures, $\phi^2 f^5$ is not well represented and that shows in the returning data of Figure 5.13. The f^3 term is also well recognised, but with a slightly worse prediction in coefficient. Other terms, such as ϕf^1 and $\phi^2 f^4$ have mixed results in discovery. The importance of the terms to the average contribution and amplitude are less and depending on the hyper parameters, one model has a more accurate return than others.

5.3 Data Regression for the Incomplete Candidate Libraries

The section 'Data Regression for the Incomplete Candidate Libraries' analyses the sparse data regression results for the incomplete libraries. The section is split in four subsections, reflecting the four different incomplete library cases. Subsection 5.3.1 show the results for leaving out the $\phi^2 f^5$ term, Subsection 5.3.2 for leaving out the f^3 term, Subsection 5.3.3 for leaving out the f^3 term with additional highly correlated candidates and Subsection 5.3.4 for replacing the f^4 terms with the highly correlated f^8 terms.

5.3.1 The First Case Incomplete Candidate Library

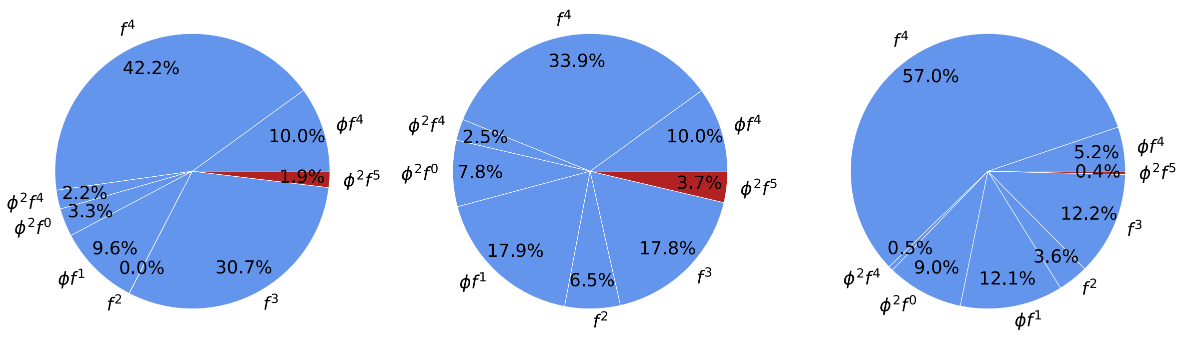


Figure 5.14: Pie chart for the averaged contribution at each step, with from left to right the vortex, exponential and shear case. In red is the function left out from the candidate library set for regression, the quatric term.

In the current section, analyses is performed for a lack of the quartic term $\phi^2 f^5$. Figure 5.14 shows the share of the full averaged ϕ contribution that is not considered by the candidate library in red. The resultant incomplete candidate library creates all possible solutions

$$\frac{D\phi}{Dt} = \sum_{n=0}^{4}\sum_{m=0}^{4} \beta_{nm}\phi^n f^m \tag{5.2}$$

In (5.2), n and m vary from 0 to 4, resulting in a total of 25 possible candidate equations.

Figures 5.15 and 5.16 show the trained models with PDE FIND for the vortex and exponential case. Note that for the incomplete section, only the vortex and exponential velocity cases are considered for training and the shear velocity case only for cross validation. Both the vortex and exponential case show good training performance, with the exponential case resulting in models that both have a low a posteriori and a low a priori error. For the exponential case, the different search window allows for more models. Both the vortex and exponential case are able to return trained a posteriori errors of less than 1 %.

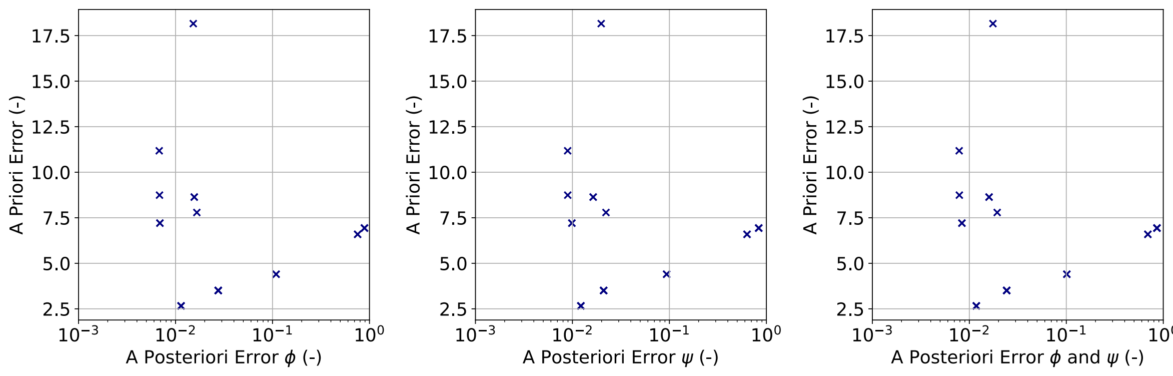


Figure 5.15: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the $\phi^2 f^5$ term for the full PDE vortex case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ.

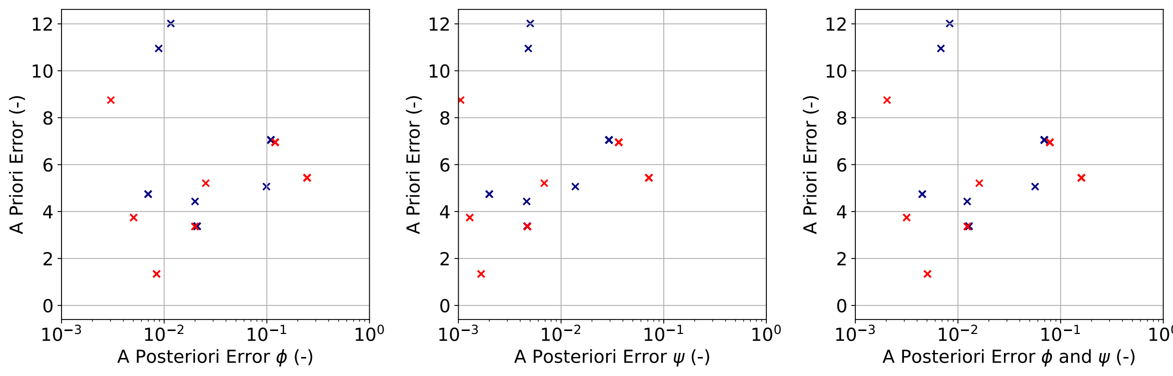


Figure 5.16: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the $\phi^2 f^5$ term for the full PDE exponential case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ. Note that blue and red represent different search windows.

Based on Figures 5.15 and 5.16, three models per case have been elected and summarised in Table 5.2. All models in the table have an a posteriori error lower than 1 % for a range of a priori errors, with the lowest a priori error being 1.34 (Model $M_{2,U1,2}$). The errors are similar to the error of the full candidate library case of Section 5.2. The lowest a priori error is the full functional form, only lacking the $\phi^2 f^5$ term that is excluded from the candidate library set. The functional form of the models can be found in Appendix H, where U_1 is the incomplete library related to the missing $\phi^2 f^5$ term.

Table 5.2: Candidate models missing the f^5 term for the vortex and exponential cases with the a priori error, a posteriori error and a note what it represents in the a priori-a posteriori plot.

Model	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	**note**
$M_{1,U1,1}$	8.74	0.7	0.9	Lowest a posteriori error
$M_{1,U1,2}$	2.66	1.1	1.2	Lowest a priori error
$M_{1,U1,3}$	7.21	0.7	1.0	Combination low a posteriori error and low a priori error
$M_{2,U1,1}$	8.75	0.3	0.1	Lowest a posteriori error
$M_{2,U1,2}$	1.34	0.8	0.2	Lowest a priori error
$M_{2,U1,3}$	3.74	0.5	0.1	Combination lowest a posteriori error and a priori error

Figures 5.17 and 5.18 shows the models of Table 5.2 to the ground truth. As expected from the low a posteriori error, the models have near perfect behaviour compared to the ground truth.

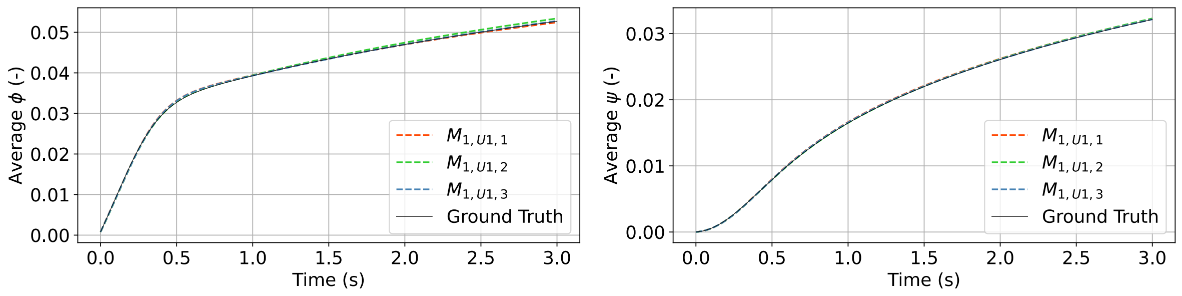


Figure 5.17: The error per time step of the three models for the full PDE exponential case, obtained from the PDE FIND procedure, with ϕ on the left and ψ on the right.

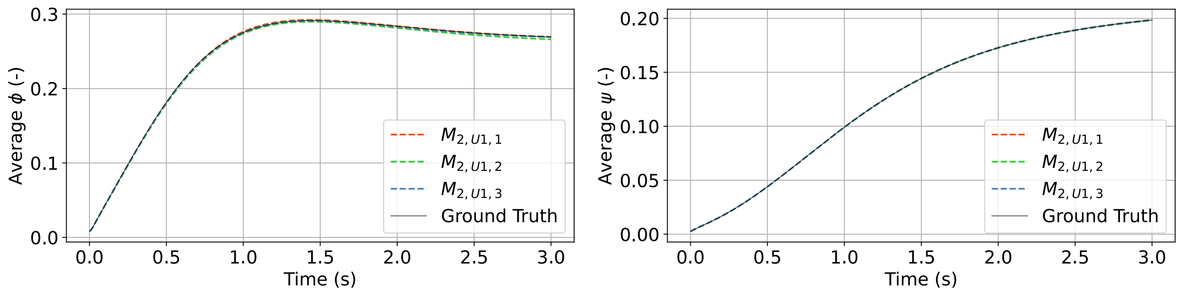


Figure 5.18: The error per time step of the three models for the full PDE exponential case, obtained from the PDE FIND procedure, with ϕ on the left and ψ on the right.

Figure 5.19 compares the coefficients of the recognised models for the functional form of the true model. For the the $\phi^2 f^0$, f^2, f^3 and f^4 terms, the identified models represent the true coefficient well, in some cases in access of less than 0.1 %. The $\phi^2 f^4$ term, however, is barely identified by most of the models, and for the two models where it is identified ($M_{1,U1,1}$ and $M_{2,U1,1}$, being both the lowest a priori error models), $M_{2,U1,1}$ has the lowest a posteriori error. The terms ϕf^1 and ϕf^4 have less models with accurate coefficients, but still four models are returned with a decent coefficient.

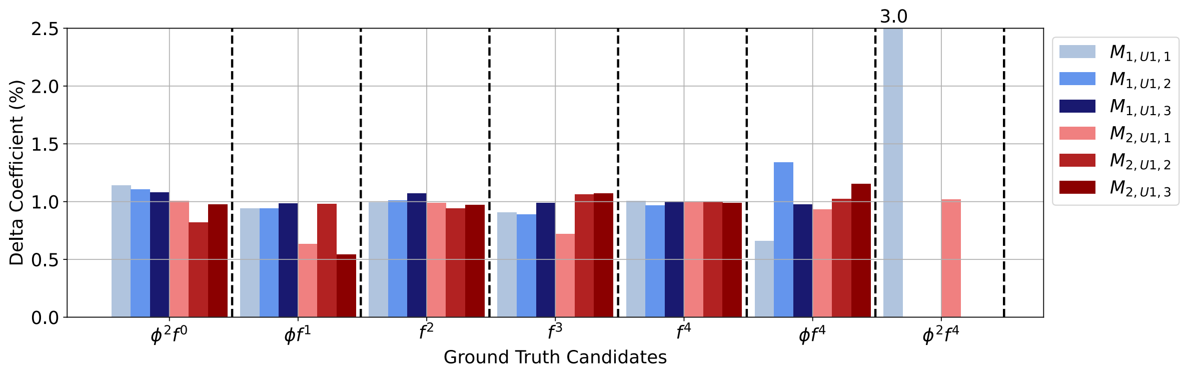


Figure 5.19: Bar plot, showing the predicted coefficient in terms of percentages compared to the true values of the true coefficients of the training data

5.3.2 The Second Case Incomplete Candidate Library

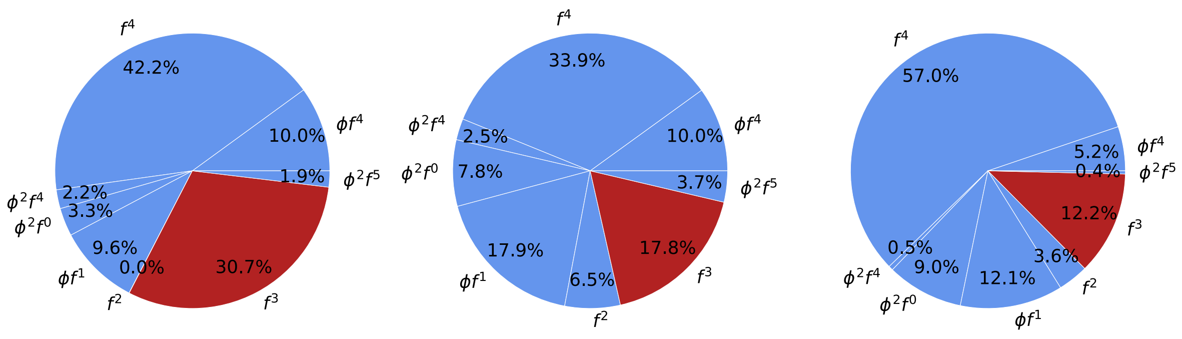


Figure 5.20: Pie chart for the averaged contribution at each step, with from left to right the vortex, exponential and shear case. In red is the function left out from the candidate library set for regression, the cross diffusion term.

For the following case, the cross-diffusion term f^3 is disregarded in the candidate library, resulting in a library given as

$$\frac{D\phi}{Dt} = \sum_{n=0}^{4}\sum_{m=0}^{2}\beta_{nm}\phi^n f^m + \sum_{n=0}^{4}\sum_{m=4}^{5}\beta_{nm}\phi^n f^m \tag{5.3}$$

In Equation (5.3), n varies from 0 to 4 while m varies from 0 to 5 but skips 3, resulting in a total of 25 candidate functions. In Figure 5.20, the average contributions of the f^3 term are rather significant, with as much as 31 % in the vortex case not represented by the candidate functions.

Figures 5.22 and 5.21 show the a priori-a posteriori figures of the vortex and exponential case. Compared to previous training scenarios, PDE FIND has a more difficult time finding suitable models. The graphs have a maximum y axis limit imposed as for the current incomplete library regression case, multiple models with a high a priori error due to significant coefficients of the true form are found. Similar to the previous section, the red crosses in Figure 5.21 are models found in a different search window.

In general, the exponential case (Figure 5.21) has a better training, having a lower a priori error with a lower a posteriori error compared to the vortex case.

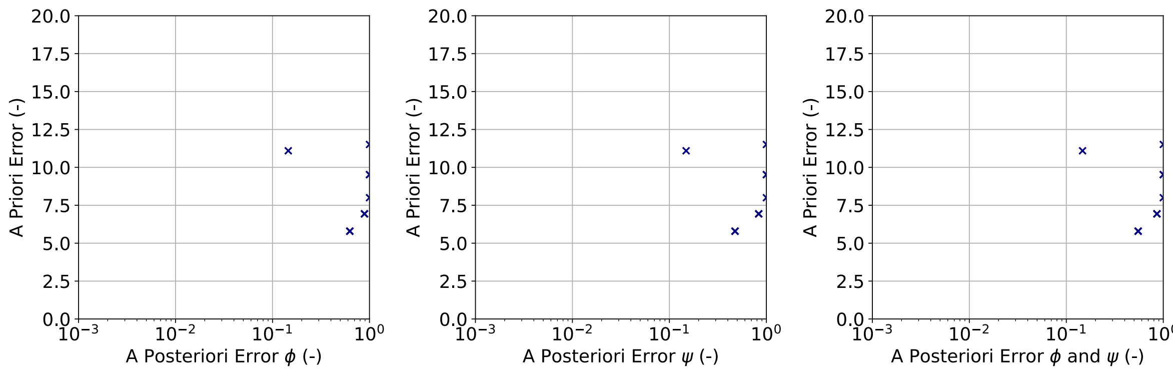


Figure 5.21: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the f^3 term for the full PDE vortex case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ

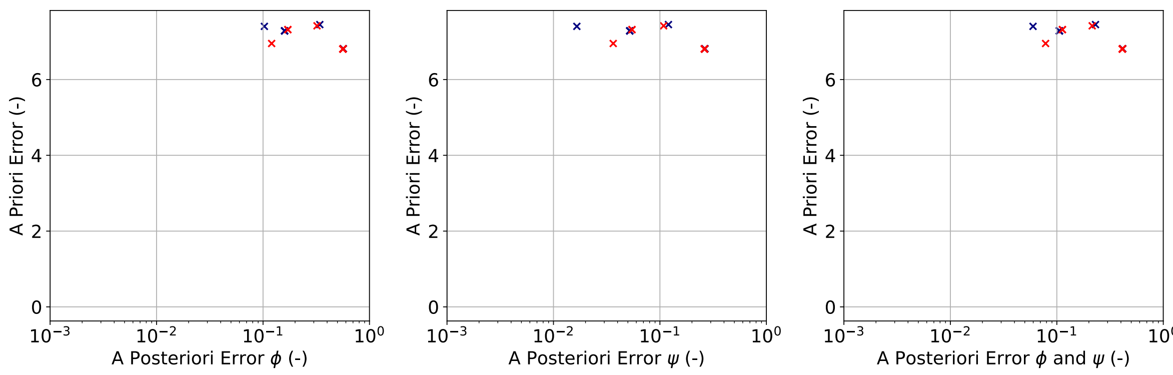


Figure 5.22: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the f^3 term for the full PDE exponential case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ. Note that blue and red represent different search windows.

From Figures 5.22 and 5.21, a selection of models are made and are shown in Table 5.3. As the vortex model is difficult to train, a high a priori model is included in the table as it carries a (relatively) low a posteriori error (model $M_{1,U2,2}$). The errors are significant, with the lowest a posteriori error model being in access of 7 % for ϕ. The value of ψ show better correspondence, with a posteriori errors being in excess of 5 % for exponential trained cases and lower than 2 % for model $M_{2,U2,1}$. The vortex trained models perform poor and the errors are quite significant, with no low a priori error model having an a posteriori error lower than 14.5 % for both ψ and ϕ. The full forms of the chosen models can be found in Appendix H.

Table 5.3: Errors of the candidate models missing the f^3 term for the vortex and exponential cases with the a priori error, a posteriori error and a note what it represents in the a priori-a posteriori plot.

Model	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	**note**
$M_{1,U2,1}$	11.10	14.5	14.9	Combination low a posteriori error and low a priori error
$M_{1,U2,2}$	99.93	9.1	8.3	Lowest a posteriori error
$M_{2,U2,1}$	7.40	10.3	1.7	Lowest a posteriori error
$M_{2,U2,2}$	7.28	15.9	5.2	Low a priori error with a low a posteriori error
$M_{2,U2,3}$	7.31	17.1	5.4	Low a posteriori error and low a priori error in a different search window

Figures 5.23 and 5.24 show the temporal development of the models on their own training data. Model $M_{1,U2,2}$, with the high a priori error, follows the training data well, having similar behaviour with an offset to the ground truth. For the exponential trained models, model $M_{2,U2,1}$ follows the training data well, but has different behaviour. For both the training cases, ψ is better estimated than ϕ, in particular for model $M_{2,U2,1}$. Compared to the previous section, Section 5.3.2, the trained models performed significantly worse.

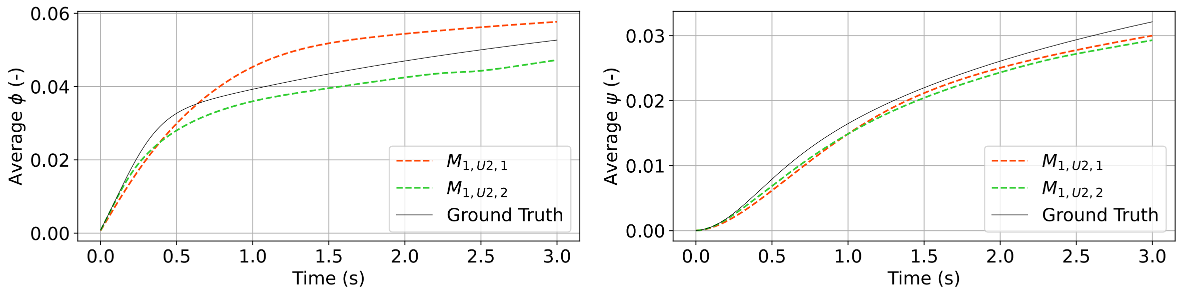


Figure 5.23: The error per time step of the two models for the full PDE vortex case, obtained from the PDE FIND procedure with an incomplete library lacking f^3, with ϕ on the left and ψ on the right.

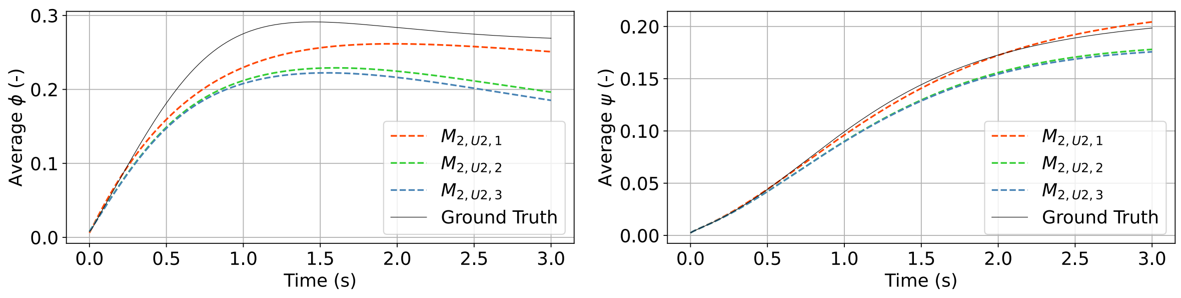


Figure 5.24: The error per time step of the three models for the full PDE exponential case, obtained from the PDE FIND procedure with an incomplete library lacking f^3, with ϕ on the left and ψ on the right.

The difference in coefficients of the model compared to the true model is seen in Figure 5.25. The functional forms related to the ground truth that are discovered the most are the f^2 and f^4 terms, being also the only terms that correspond well to the ground truth. For the exponential case trained models, some of the coefficients of f^4 are close to the original coefficient. Note that model $M_{1,U2,2}$ includes a lot of the true forms with a significantly different coefficient, as seen for functions $\phi^2 f^0$, ϕf^1, ϕf^4, $\phi^2 f^4$ and $\phi^2 f^5$, where the coefficients are multiples higher than the true value. The high a priori error model, model $M_{1,U2,2}$, includes a lot of terms in the system that are highly correlated and have high coefficients to compensate.

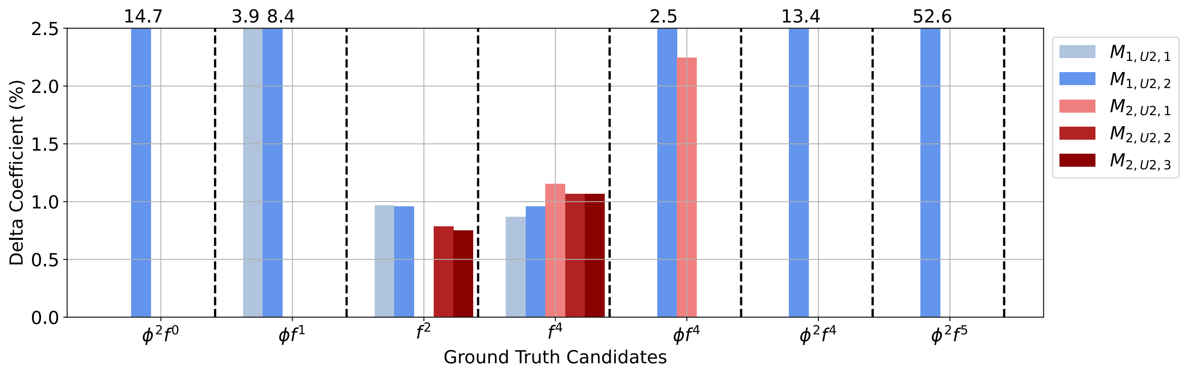


Figure 5.25: Bar plot, showing the predicted coefficient in terms of percentages compared to the true values of the true coefficients of the training data for the cases lacking the f^3 term in the library.

5.3.3 The Third Case Incomplete Candidate Library

The current case is an expansion on the previous case. It lacks the f^3 term in the candidate library, but adds extra candidate function options f^6 and f^7, giving

$$\frac{D\phi}{Dt} = \sum_{n=0}^{4}\sum_{m=0}^{2}\beta_{nm}\phi^n f^m + \sum_{n=0}^{4}\sum_{m=4}^{7}\beta_{nm}\phi^n f^m \tag{5.4}$$

In (5.4), n varies from 0 to 4 while m varies from 0 to 7, missing 3. The result is a candidate library consisting of 35 terms.

Figures 5.26 and 5.27 show the a priori-a posteriori plots for the trained vortex and exponential velocity cases. Compared to Figures 5.21 and 5.22, improvement is found in the a posteriori error with a greater spread in options in the a priori error, in particular for the vortex trained case. For the exponential models, multiple models with an a posteriori error less than 10 % are found. Compared to the full candidate library, and the first incomplete library case, the performance is still lacking.

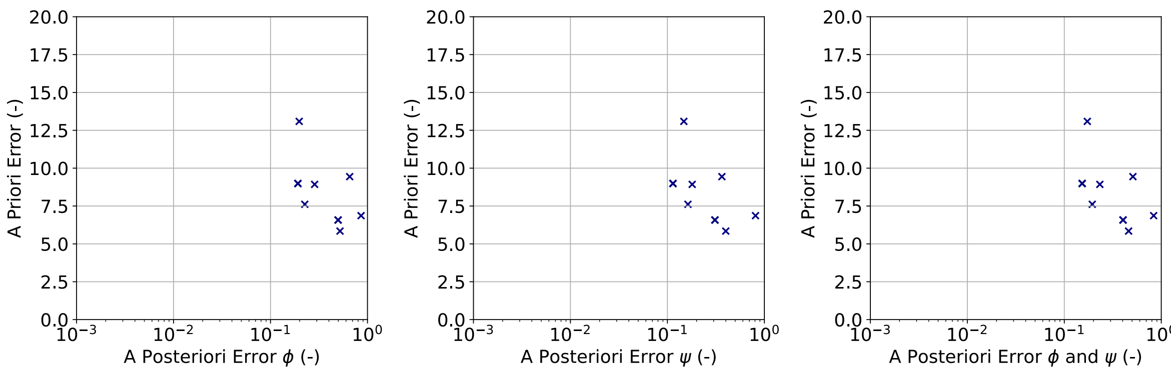


Figure 5.26: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the f^3 term, but including additional terms f^6 and f^7 for the full PDE vortex case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ.

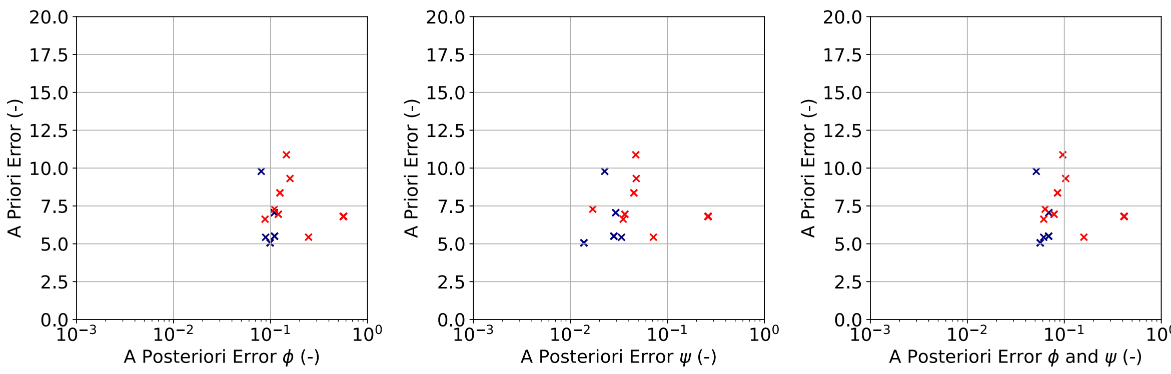


Figure 5.27: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the f^3 term, but including additional terms f^6 and f^7 for the full PDE exponential case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ. Note that blue and red represent different search windows.

Based on the a priori-a posteriori plots of Figures 5.26 and 5.27, three models per case have been elected and summarised in Table 5.4. From all models, $M_{2,U3,1}$ has the lowest a posteriori error for ϕ in excess of 8 %, what is an improvement of 2 % compared to the case without additional candidate functions. Particularly encouraging are a posteriori errors of ψ, with model $M_{2,U3,2}$ lower than 2 %. The vortex models still have difficulty training, with the two decent a priori models performing worse in the current training task with the additional functions than without (Subsection 5.3.2, model $M_{1,U2,1}$ compared to models $M_{1,U3,1}$ and $M_{1,U3,2}$). The high a priori error performs better for the vortex trained model, with the a posteriori error lower than 10 %.

Table 5.4: Errors of the candidate models missing the f^3 term, but with additional f^6 and f^7 terms for the vortex and exponential cases with the a priori error, a posteriori error and a note what it represents in the a priori-a posteriori plot.

Model	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	**note**
$M_{1,U3,1}$	8.99	19.2	11.4	Low a posteriori error with a low a priori error
$M_{1,U3,2}$	7.61	22.6	16.3	Low a priori error with a low a posteriori error
$M_{1,U3,3}$	63.38	9.9	8.2	Lowest a posteriori error
$M_{2,U3,1}$	9.78	8.0	2.2	Lowest a posteriori error
$M_{2,U3,2}$	5.06	9.9	1.4	Lowest a priori error
$M_{2,U3,3}$	6.63	8.7	3.5	Lowest a posteriori error other in a different search window

The temporal development of the models from Table 5.4 can be seen in Figures 5.28 and 5.29. In these plots, the models trained using data of the vortex case ($M_{1,U3,1}$, $M_{1,U3,2}$ and $M_{1,U3,3}$) show a significant difference compared to the training data for both ϕ and ψ. Different behaviour in the development of the solution is prevalent and the result is similar to the trained models without the additional functions in Subsection 5.3.2.

Models $M_{2,U3,1}$, $M_{2,U3,2}$ and $M_{2,U3,3}$ show better behaviour in Figure 5.29 and in particular model $M_{2,U3,3}$ has a satisfactory result, reaching a significantly better steady state value than other models. For that model, however, the development characteristic is different, showing a different peak in ϕ compared to the ground truth. In general, all models show different mechanisms throughout time for the exponential trained models.

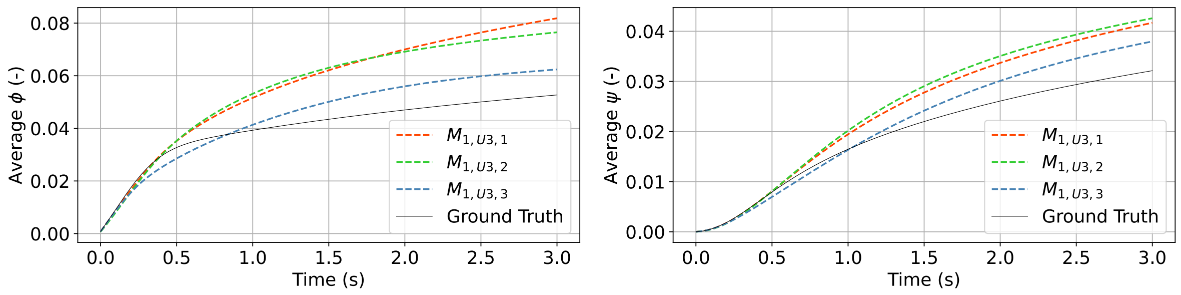


Figure 5.28: The error per time step of the three models for the full PDE vortex case, obtained from the PDE FIND procedure with a candidate library lacking the f^3 terms but with additional f^6 and f^7 terms. ϕ is on the left and ψ on the right.

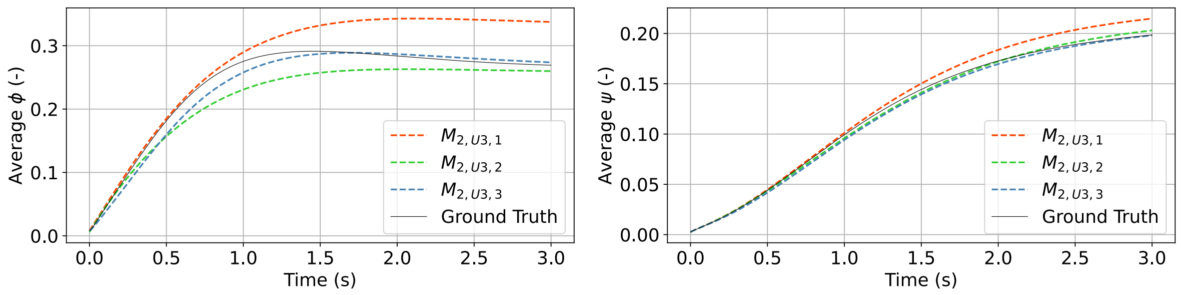


Figure 5.29: The error per time step of the three models for the full PDE exponential case, obtained from the PDE FIND procedure with a candidate library lacking the f^3 terms but with additional f^6 and f^7 terms. ϕ is on the left and ψ on the right.

Figure 5.30 show a bar chart comparing the coefficients of the true form with the trained forms. As in the previous case, f^2 and f^4 are the most recognised terms with the trained coefficients of the exponential case approaching the correct f^4 terms. The ϕf^4 term is also recognised, but the coefficients are not as accurate as for f^4 and f^2. Similar to $M_{1,U2,2}$ from Figure 5.25, the overfit of $M_{1,U3,3}$ is clear with large deviations in coefficients. The terms $\phi^2 f^0$ and $\phi^2 f^4$ are poorly recognised, with coefficients not close to ground truth while for ϕf^1 all the vortex trained cases have a different coefficient. The high coefficients of the vortex trained models contribute to the high a priori errors due to the definition of the a priori error.

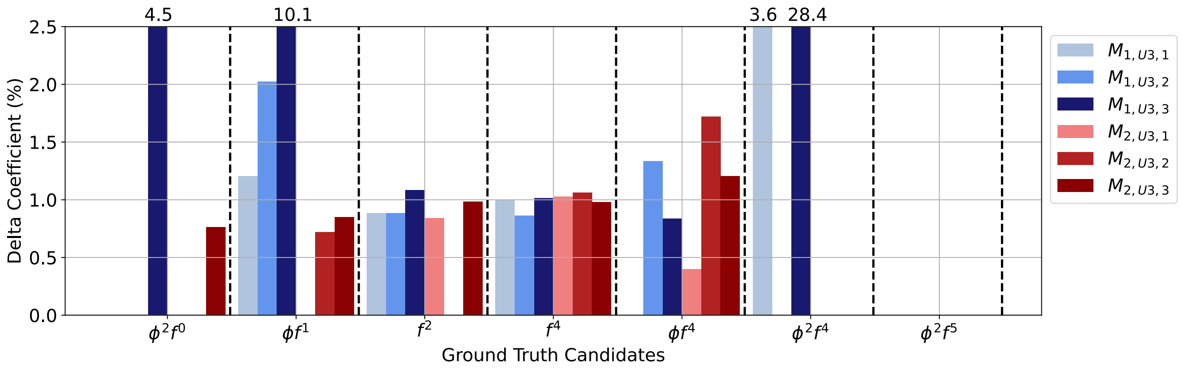


Figure 5.30: Bar plot, showing the predicted coefficient in terms of percentages compared to the true values of the true coefficients of the training data for the candidate library cases lacking the f^3 term, with additional f^6 and f^7 terms.

5.3.4 The Fourth case Incomplete Candidate Library

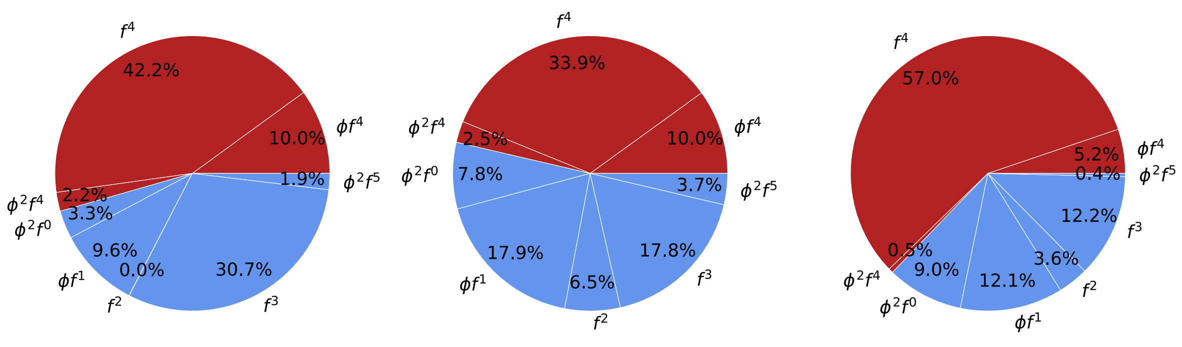


Figure 5.31: Pie chart for the averaged contribution at each step, with from left to right the vortex, exponential and shear case. In red are the functions replaced in the candidate library set for regression, being the strain rate terms.

The last case replaces the strain rate squared f^4 terms by the strain rate quadrupled f^8 terms, resulting in a candidate library as

$$\frac{D\phi}{Dt} = \sum_{n=0}^{4}\sum_{m=0}^{3} \beta_{nm}\phi^n f^m + \sum_{n=0}^{4} \phi^n f^5 + \beta_{nm} \sum_{n=0}^{4} \phi^n f^8 \tag{5.5}$$

In (5.5), n varies from 0 to 4 and m varies from 0 to 8, lacking the 4, 6 and 7 terms resulting in a total of 30 candidate functions. The f^8 terms highly correlates with the f^4

terms (Appendix F) and the share of the f^4 related terms to the averaged field solution can be seen in Figure 5.31. The averaged contribution of the share is significant and more than 50 % for the vortex and shear velocity cases.

Figures 5.32 and 5.33 show the a priori-a posteriori plots of the tested models. The vortex case, Figure 5.32 trains poorly, with no a posteriori error lower than 31.6 % for ϕ. The exponential case, Figure 5.33 provides decent models, with multiple models having a lower than 10 % a posteriori error. It can be noted that the ψ is more accurate than ϕ even if the training is not performed on ψ.

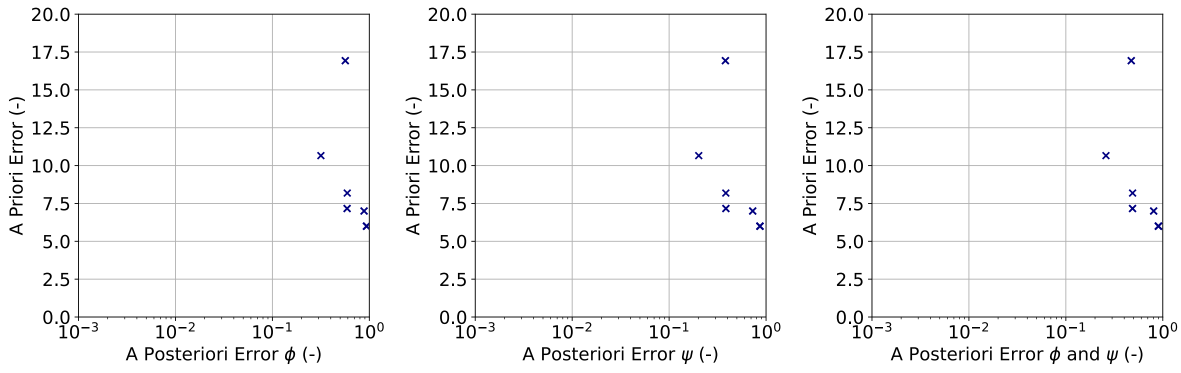


Figure 5.32: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the f^4 term, but including additional term f^8 for the full PDE vortex case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ.

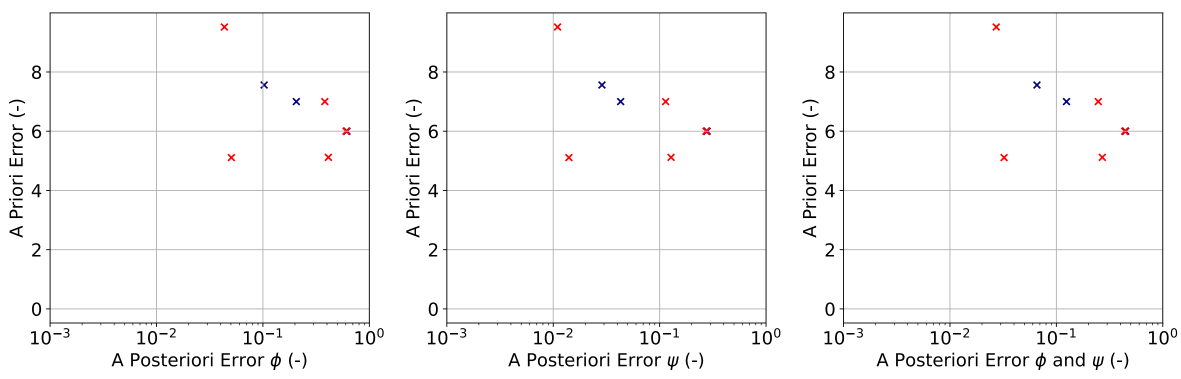


Figure 5.33: A comparison of the errors for different models resulting from the PDE FIND procedure with an incomplete library lacking the f^4 term, but including additional term f^8 for the full PDE exponential case. On the left the errors related to ϕ, in the middle the error relate to ψ and on the right the average error of ϕ and ψ. Note that blue and red represent different search windows.

From Figures 5.32 and 5.33, Table 5.5 shows a selection of the trained models. The difference in errors between the vortex and exponential models is quite significant, with the vortex trained models performing the worst. For the vortex trained models, no a posteriori error is lower than 20 % with ψ performing better than ϕ in the a posteriori sense. Model $M_{2,U4,2}$ has the lowest a priori error, with an error of 5.11. 5.11 is a low a priori error, regarding that the lack of f^4 terms gives a minimum a priori error of at least 3 and with the addition of at least one f^8 term 4. All the models forms are given in Appendix H.

Table 5.5: Errors of the candidate models where the f^4 terms are replaced by f^8 terms in the candidate function for the vortex and exponential cases with the a priori error, a posteriori error and a note what it represents in the a priori-a posteriori plot.

Model	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	**note**
$M_{1,U4,1}$	10.66	31.6	20.2	Lowest a posteriori error
$M_{1,U4,2}$	7.16	59.0	38.5	Low a priori error with a low a posteriori error
$M_{2,U4,1}$	9.52	4.3	1.1	Lowest a posteriori error
$M_{2,U4,2}$	5.11	5.1	1.4	Lowest a priori error
$M_{2,U4,3}$	7.56	10.3	2.8	Lowest a posteriori error in a different search window

Figures 5.35 and Figures 5.34 show the temporal behaviour of the selected models in Table 5.5 compared to the ground truth. The vortex data trained models perform poorly, not being able to reach the steady state value and not being able to map the characteristic behaviour for both models. The exponential trained models are better but lack the same characteristic with regards to the peak in the ground truth. Model $M_{2,U4,3}$ does not have a peak and models $M_{2,U4,1}$ and $M_{2,U4,2}$ have a later peak at a different magnitude compared to the ground truth.

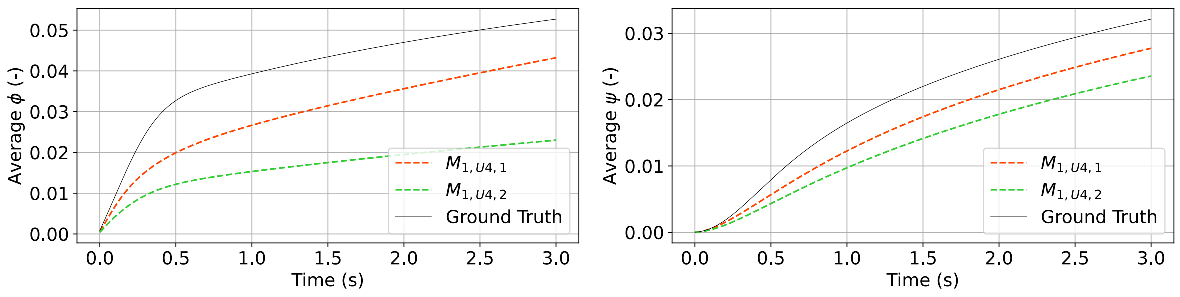


Figure 5.34: The error per time step of the two models for the full PDE vortex case, obtained from the PDE FIND procedure, with ϕ on the left and ψ on the right.

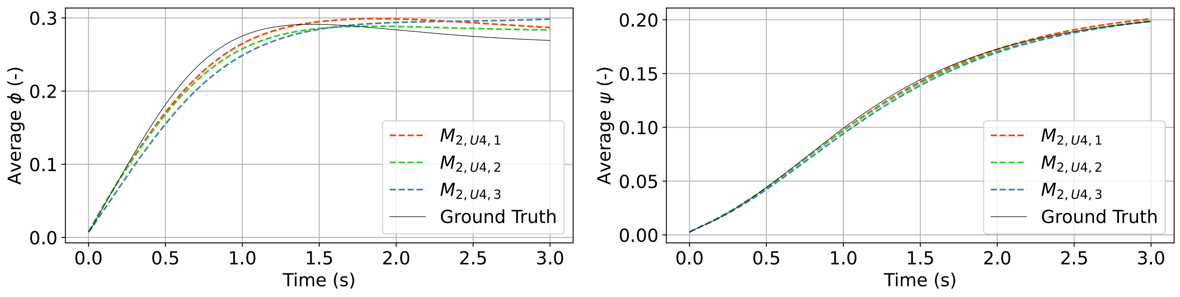


Figure 5.35: The error per time step of the three models for the full PDE exponential case, obtained from the PDE FIND procedure, with ϕ on the left and ψ on the right.

Figure 5.36 shows the bar graph of the coefficients compared to the true coefficient. All coefficients related to the vortex case are poor. For the exponential case, decent behaviour is found for the f^2 term that is most of the time recognised. For models $M_{2,U4,1}$ and $M_{2,U4,2}$, ϕf^1 is found well with a satisfactory coefficient. The $\phi^2 f^5$ is not recognised by any model at all, and the f^3 term is only recognised by two models, one per velocity trained case.

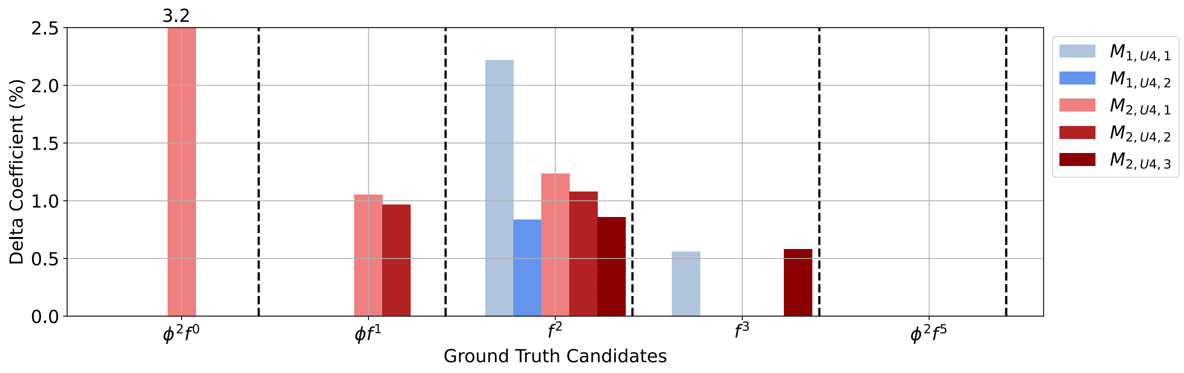


Figure 5.36: Bar plot, showing the predicted coefficient in terms of percentages compared to the true values of the true coefficients of the training data

6. Data Validation

The chapter 'Data Validation' discusses the performance of the trained models of Chapter 5 and tests the models on different test cases. For the complete candidate library trained models, all the velocity cases are cross tested with one another. For the incomplete library trained models, the shear velocity case acts as the test data. For all models, the exponential trained models performed better than the vortex trained models, with the exponential models being able to follow the shear velocity case ground truth better.

For the complete candidate library, the cross tested models performed well. Even if the true functional form was not returned, the models still performed near perfect and no worrying discrepancies were found. The incomplete library case where a small average contribution (the first incomplete library) is lacking, provided near perfect results in cross testing and the replacement terms didn't worsen the solution on the test data.

For the incomplete library lacking a large contribution (the second incomplete library and third incomplete library cases), only one model, model $M_{2,U3,3}$ is found to have satisfactory behaviour cross tested on different cases. The model is build from highly correlated replacement functions to the missing true form and is relatively sparse. For the library cases where the production term is replaced by a different production term (incomplete library case four), the models perform poorly and no well tested model is found.

The z-score (an measure of the likelihood of functions in the solution) are the greatest for f^2 and f^4. These correspond to the greatest amplitude and greatest average contribution cases of Section 4.3 from Chapter 4, similar to the results in Chapter 5 where f^2 and f^4 were often found with a satisfactory coefficient.

The ridge regression step is able to damp highly correlated functions well, and together with the z-score, the importance of correlated functions change depending on the elimination of one of the functions. In the z-score, often a redistribution among highly correlated functions is seen, resulting in eliminating the ground truth form and replacing it by a highly correlated replacement form that has the potential to function well in a model.

Section 6.1 discusses the validation for the full case with the complete library models, models with F in the model tag. Section 6.2 discusses the validation results for an incomplete candidate library, models with the tags U_1 to U_4. Each section will have a further analyses in the best performing model per trained velocity case, with a focus on the selection procedure of PDE FIND. Section 6.3 will close the chapter discussing the results.

6.1 The Complete Candidate Set Validation Results

The current section analyses the models that have F in the model tag. An overview of the models can be found in Appendix H.

6.1.1 Cross Checking on the Shear Case

Table 6.1 shows a table with all the a posteriori errors for the vortex and exponential trained models used on the shear velocity case. All models show satisfactory behaviour, with a low (lower than 1 %) error. Model $M_{2,F,2}$, having the complete true functional form, is the best performing model, in particular approaching steady state.

Table 6.1: Errors of the vortex and exponential trained models tested on the shear case for the full PDE case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{1,F,1}$	0.6	0.1	1.2	0.2
$M_{1,F,2}$	1.0	0.09	1.3	0.3
$M_{1,F,3}$	0.4	0.1	1.4	0.3
$M_{2,F,1}$	0.1	0.01	0.5	0.03
$M_{2,F,2}$	0.2	0.03	0.03	0.02
$M_{2,F,3}$	0.2	0.02	0.7	0.01

The results from Table 6.1 are shown for the temporal progression in Figures 6.1 and 6.2. The figures show a good prediction compared to the ground truth. All models trained with the vortex case does show a difference when approaching steady state, but it is small and is considered to be acceptable. Even if it is small, the exponential trained models follow the ground truth better than the vortex trained models.

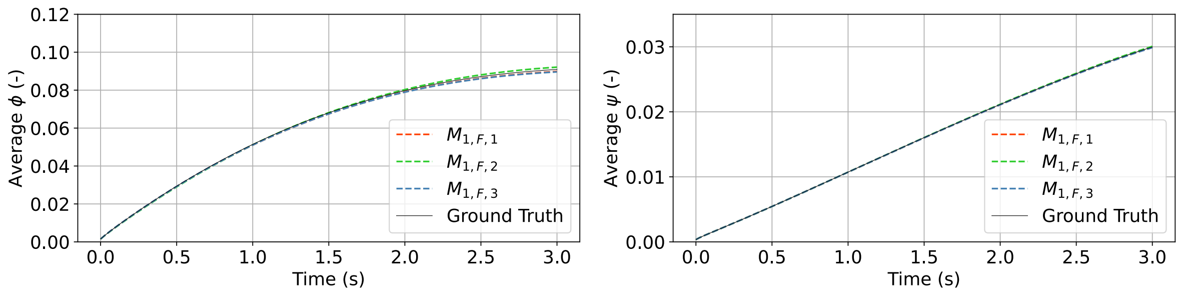


Figure 6.1: The error per time step of the three trained models for the full PDE vortex case tested on the shear case, with ϕ on the left and ψ on the right.

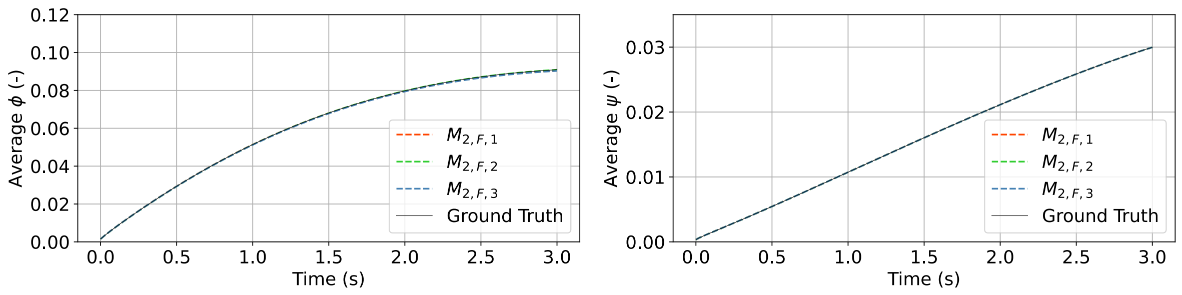


Figure 6.2: The error per time step of the three trained models for the full PDE exponential case tested on the shear case, with ϕ on the left and ψ on the right.

Figures 6.3 and 6.4 compares the cross-section of the models to the ground truth. The trained models from the exponential case show better behaviour than the models trained from the vortex case, with Model $M_{2,F,2}$ approaching the ground truth the best. Model $M_{2,F,2}$ is the true functional form, so the best performance is to be expected.

From the comparison plot, major differences are found at the peak of $y = 0$. Model $M_{1,F,2}$ is the least performing model, over predicting the production at $y = 0$. Models $M_{1,F,1}$ and $M_{1,F,3}$ do the opposite and under predict the solution at $y = 0$. The discrepancies of the vortex trained models are small and would function as satisfactory models.

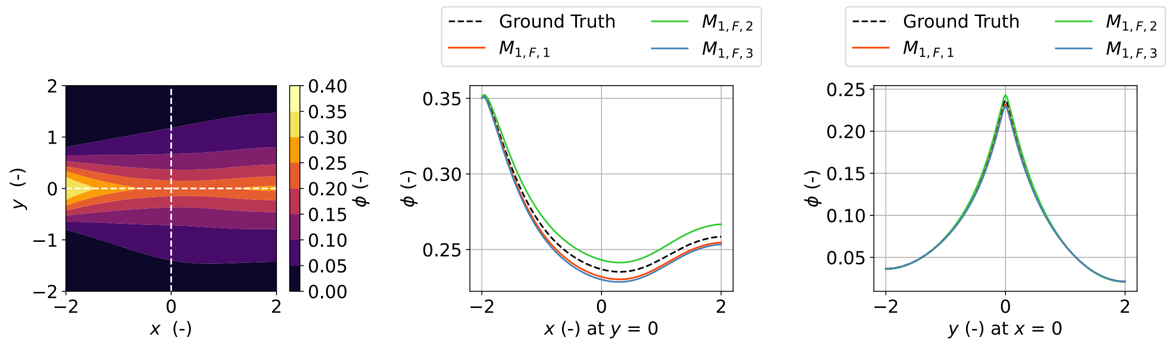


Figure 6.3: Cross checking between models for the models trained with the vortex data tested on the shear case.

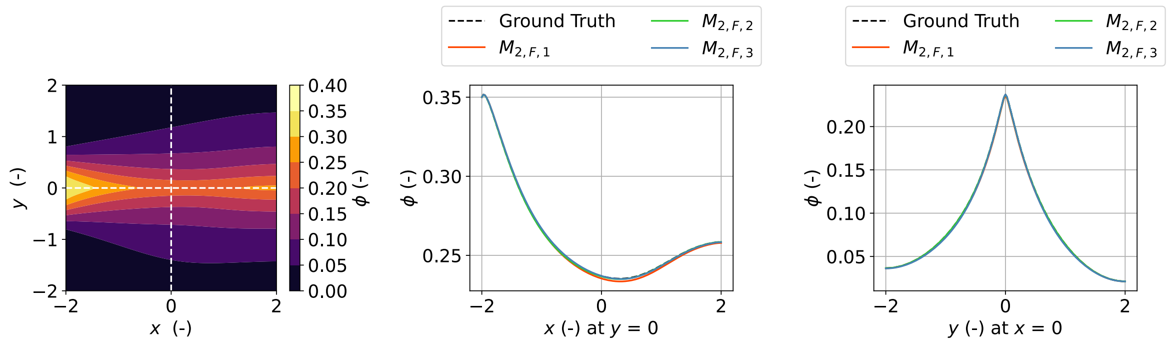


Figure 6.4: Cross checking between models for the models trained with the exponential data tested on the shear case.

6.1.2 Cross Checking on the Vortex Case

Table 6.2 shows a table with all the errors of the exponential and shear trained models tested on the vortex case. The trained models from the exponential case perform considerably better than the trained models of the shear case, with model $M_{3,F,2}$ diverging. From the three shear trained models, model $M_{3,F,1}$ has the lowest reliable error.

Table 6.2: Errors of the exponential and shear trained models tested on the vortex case for the full PDE case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{2,F,1}$	0.6	0.8	0.007	0.2
$M_{2,F,2}$	0.4	0.7	0.02	0.05
$M_{2,F,3}$	0.4	0.7	1.0	0.2
$M_{3,F,1}$	1.1	1.3	0.07	0.7
$M_{3,F,2}$	NAN	NAN	NAN	NAN
$M_{3,F,3}$	20.1	13.0	34.8	18.1

Figures 6.5 and 6.6 shows the temporal progression. The divergence of model $M_{3,F,2}$ is clear and the behaviour of model $M_{3,F,3}$ is different then the ground truth. The exponential trained cases, similar to the shear case of Subsection 6.1.1 perform more than satisfactory, and are close to equal compared to the ground truth.

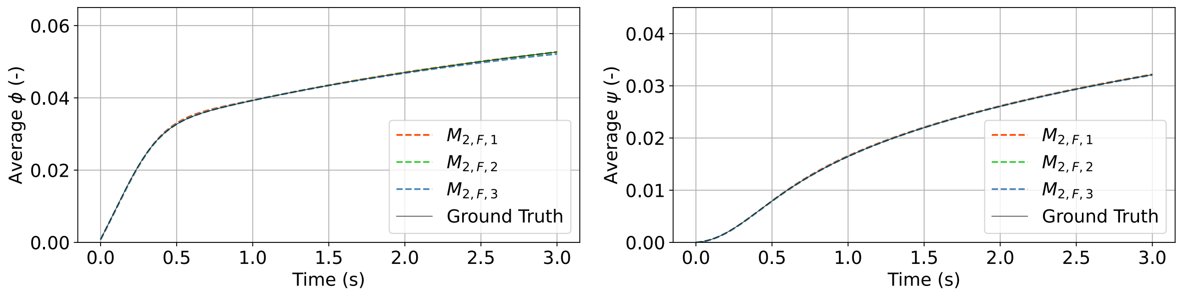


Figure 6.5: The error per time step of the three trained models for the full PDE exponential case tested on the vortex case, with ϕ on the left and ψ on the right.

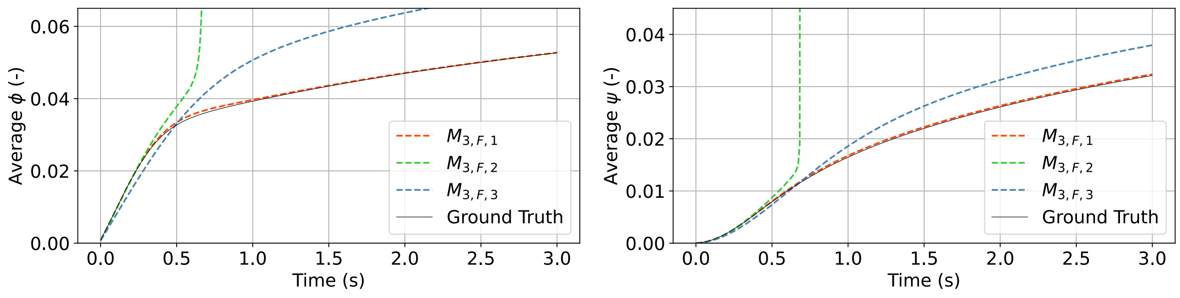


Figure 6.6: The error per time step of the three trained models for the full PDE shear case tested on the vortex case, with ϕ on the left and ψ on the right.

Figures 6.7 and 6.8 compare the models with the ground truth with a cross-cut comparison. The exponential models follow the model and are near identical compared to the ground truth. From the shear trained models, The diverging model $M_{3,F,2}$ is not present and the best performing model from Table 6.2, model $M_{3,F,1}$ is near identical to the ground truth. The other shear model $M_{3,F,3}$ performs poorly.

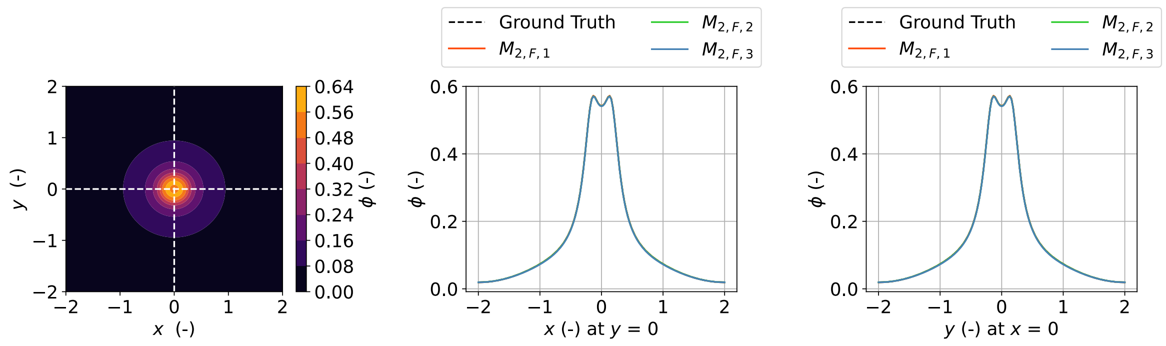


Figure 6.7: Cross checking between models for the models trained with the exponential data tested on the vortex case.

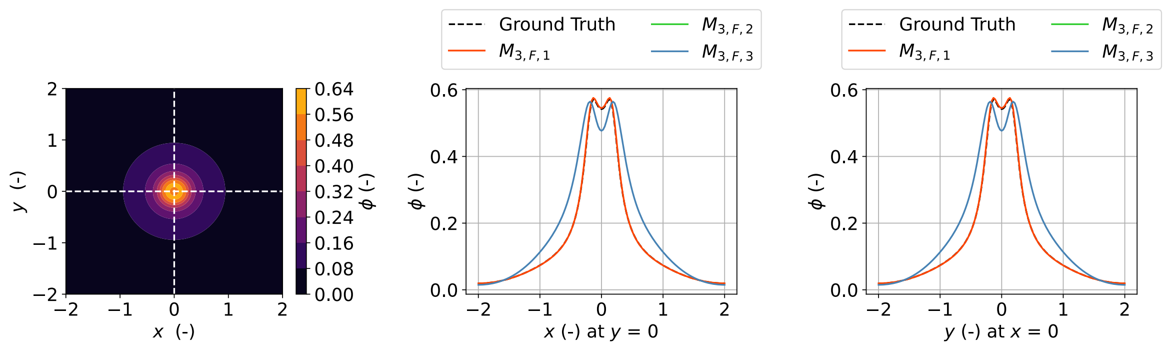


Figure 6.8: Cross checking between models for the models trained with the shear data tested on the vortex case.

Looking at the final time step picture of ϕ for the shear trained models, Figure 6.9, model $M_{3,F,1}$ is similar to the ground truth in Figure 4.4. Model $M_{3,F,3}$ is qualitatively different, with a larger diffusive range and a less prominent core.

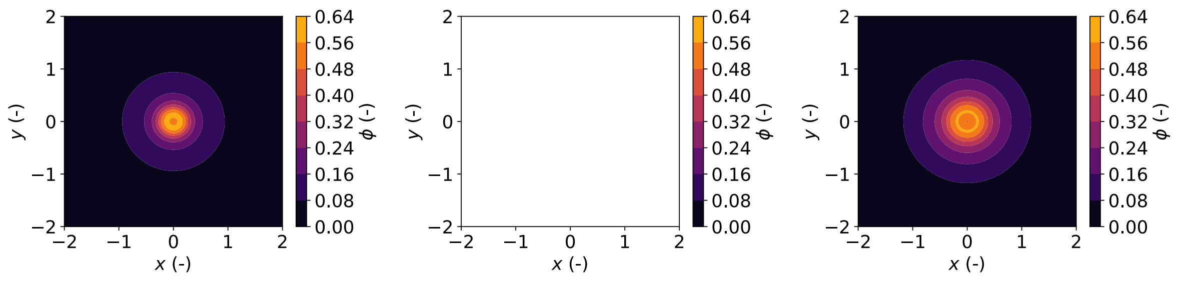


Figure 6.9: Final time step of the shear trained models tested on the vortex case.

6.1.3 Cross Checking on the Exponential Case

Table 6.3 shows a table with all the errors of the vortex and shear trained models tested on the exponential case. The vortex trained models perform better than the shear trained models. The shear case does perform better than the previous subsection on the vortex case.

Table 6.3: Errors of the vortex and shear trained models tested on the exponential case for the full PDE case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{1,F,1}$	0.4	0.2	0.3	0.1
$M_{1,F,2}$	1.4	0.4	1.1	0.6
$M_{1,F,3}$	0.4	0.2	0.7	0.2
$M_{3,F,1}$	0.8	0.2	0.8	0.4
$M_{3,F,2}$	5.1	2.2	6.1	5.3
$M_{3,F,3}$	7.0	3.2	5.9	4.8

Figures 6.10 and 6.11 show the temporal progression of the models compared to the ground truth. All vortex trained cases perform satisfactory while for the shear case only model $M_{3,F,1}$ performs well, similar to the vortex tested case. Compared to the vortex tested case of the previous subsection, model $M_{3,F,2}$ does not diverge, but it does not have the same temporal characteristic and has the poorest performance.

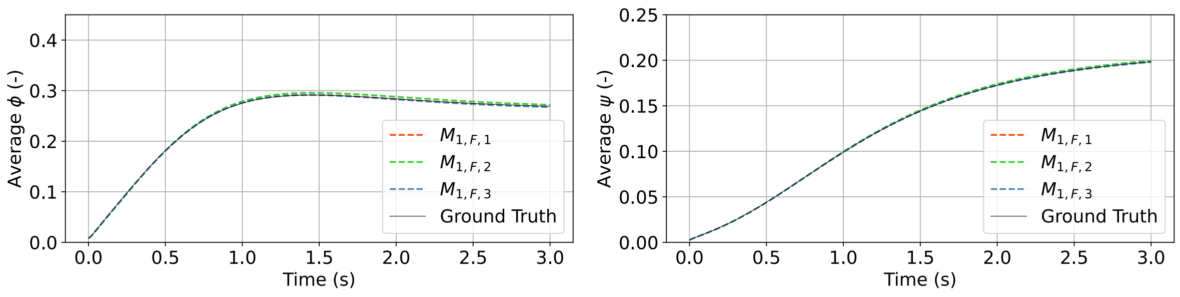


Figure 6.10: The error per time step of the three trained models for the full vortex case tested on the exponential case, with ϕ on the left and ψ on the right.

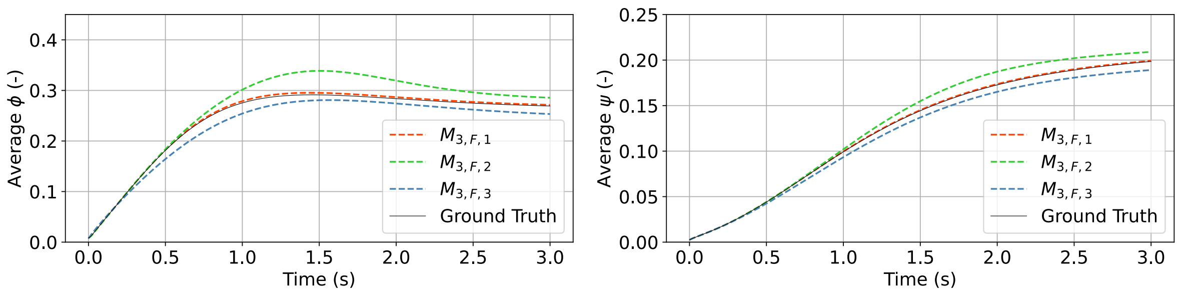


Figure 6.11: The error per time step of the three trained models for the full shear case tested on the exponential case, with ϕ on the left and ψ on the right.

Figures 6.12 and 6.13 show the cross-section comparison between the trained models and the exponential ground truth. Similar to previous statements, the vortex trained models perform well and show near perfect results. A large discrepancy is found in the shear trained cases with model $M_{3,F,2}$ over predicting the the peaks at the largest gradients while $M_{3,F,3}$ under predicts the peaks. Both also show a different mechanism compared to the ground truth.

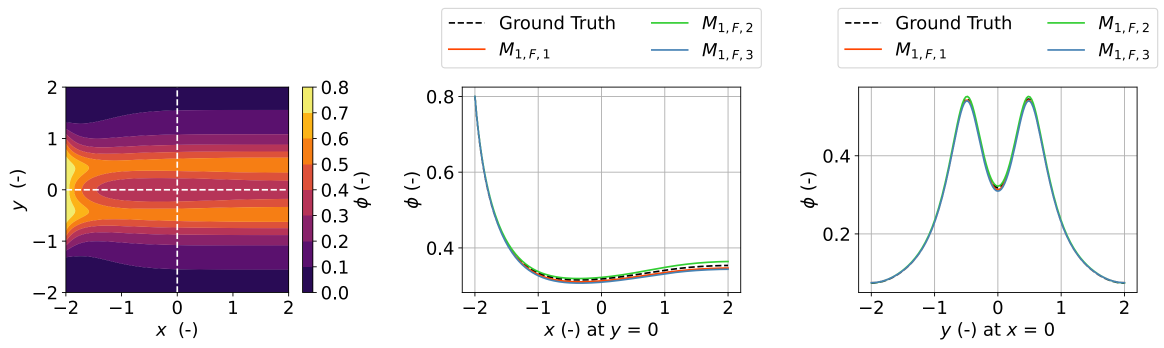


Figure 6.12: Cross checking between models for the models trained with the vortex data tested on the exponential case.

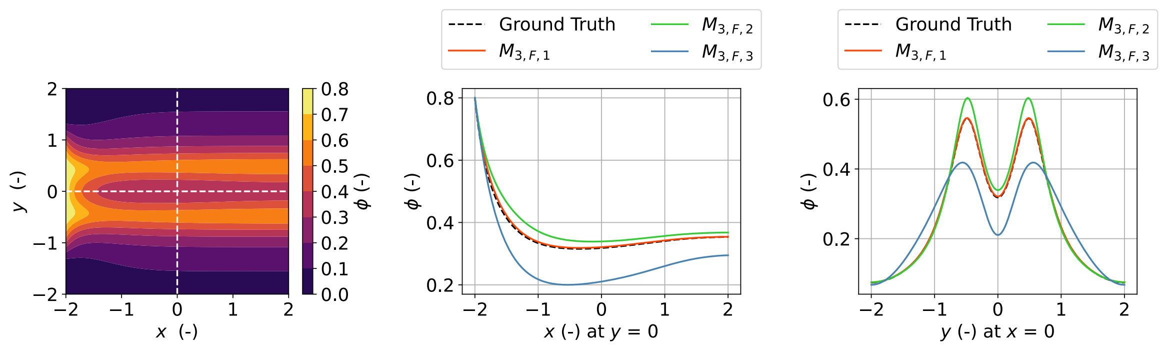


Figure 6.13: Cross checking between models for the models trained with the vortex data tested on the shear case.

For the shear models, Figure 6.14 show the final time step solutions for the shear trained models. A major difference is found between the models. While model $M_{3,F,1}$ shows an almost identical solution to the ground truth (Figure 4.10), models $M_{3,F,2}$ and $M_{3,F,3}$ are different, with a particular visual difference found with the largest velocity gradients of the solution.

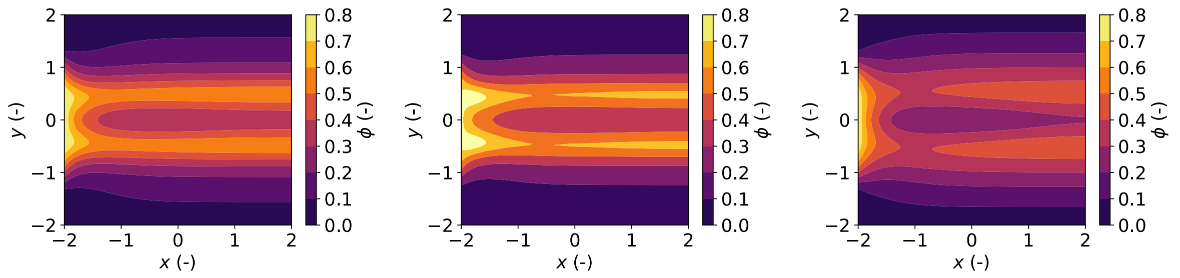


Figure 6.14: Final time step of the shear trained models tested on the vortex case.

6.1.4 Analyses of the Models for the Full Candidate Library

The current section will do a further analyses of the regression behaviour of the most competitive model per velocity training case, being models $M_{1,F,1}$, $M_{2,F,2}$ and $M_{3,F,1}$. The model description as well as the other models used in the section previously can be found in Appendix H.

Model $M_{2,F,2}$, as stated previously, is a model where the full ground truth is returned with an average coefficient error of 1.8 %. Model $M_{1,F,1}$ is a model with more terms, having an extra f^1, $\phi^2 f^1$, ϕf^2, ϕf^3, $\phi^3 f^3$ and $\phi^4 f^5$ term. Some of the coefficients are rather small, being the coefficients of f^1 and ϕf^2. It is not surprising that the vortex case returns a significant amount of terms more, as the vortex case has a relatively greater correlation between functions than the other cases (Appendix F, Figure F.2 for the correlation heat map). The $\phi^2 f^1$ term is highly correlated with other functional forms containing f^1 and f^0 and has a negative correlation with terms containing f^2. Even if it is not the most sparse model, the model provided robust behaviour during testing, showing a good resemblance with the exponential and shear test cases.

Model $M_{3,F,1}$ is different to Model $M_{1,F,1}$, as in that it lacks terms compared to the ground truth model and does not substitute it with other terms. The terms missing are the $\phi^2 f^4$ and the $\phi^2 f^5$ terms. These are terms that have a low amplitude and average field contribution in the solution of the shear case (Figures 4.21 and 4.22 from Chapter 4). The greatest difference in coefficient is found in the coefficient of ϕf^4 with a 24 % difference. It can be explained as the function is compensating for the loss of the $\phi^2 f^4$ term that is highly correlated with ϕf^4 (Appendix F, Figure F.4). $\phi^2 f^5$ is not necessarily compensated by other functions that correlate well with $\phi^2 f^5$ and from Figures 4.21 and 4.22 it can be seen that $\phi^2 f^5$ contribution is small and the function might have been lost in the hard cut-off of the STRidge procedure. All models that are trained and selected from the shear data have no contribution to the f^5 function.

Figures 6.15, 6.16 and 6.17 show difference between the least squares prediction and the ridge prediction at the first iteration. Ridge regression has an unique solution, given as

$$\beta_i = (X_{km}X_{ki} - \lambda\delta_{im})^{-1}X_{jm}y_j \tag{6.1}$$

In (6.1), $X_{km}X_{ki}$ is a scaled measure for the covariance matrix (4.13), λ the ridge regression hyper parameter and δ_{im} the dirac operator being 1 for i=m and 0 else and y is the target data. With (6.1), it can be seen that λ affects the largest variances the most, reducing the importance of regression on the variance (the diagonals of the matrix) and making the covariance more important.

From Figures 6.15, 6.16 and 6.17, the effect of Ridge Regression is clear. Based on the variances from Figure F.1 in Appendix F, it can be seen that the terms with a greater variance are more muted after the ridge regression procedure. From the figures, it can be seen that in ordinary least squares, often large absolute coefficients are found. The large coefficients reflect correlation with other functions, as a function with a positive large coefficient might have a highly correlated function with a negative large coefficient that in itself has a highly correlated function with a large positive coefficient, creating a oscillating pattern between functions. For instance, in Figure 6.15, all the f^3 terms shows a large positive-negative coefficient behaviour and the least squares solution is said to be over-fit. Ridge regression mutes the behaviour well and reduces the correlation between the functions. For all cases, f^4 has a large coefficient and as the variance is rather small keeps a high coefficient after the ridge procedure. In particular, it is the case for Figure 6.16, where not only f^4 is significant, but also f^2 and f^3. It can be interpreted as those functions correlating well with the ground truth data.

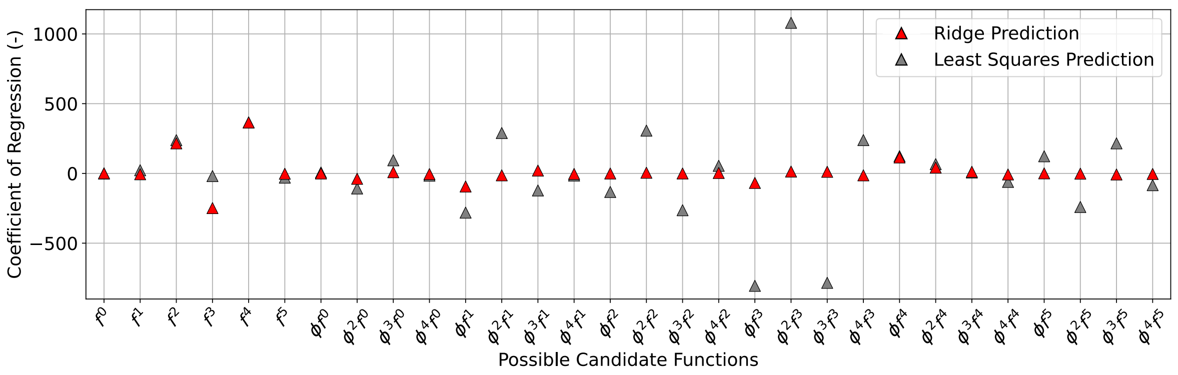


Figure 6.15: The normalised coefficient prediction of least squares and ridge regression for the vortex case for a full library of terms.

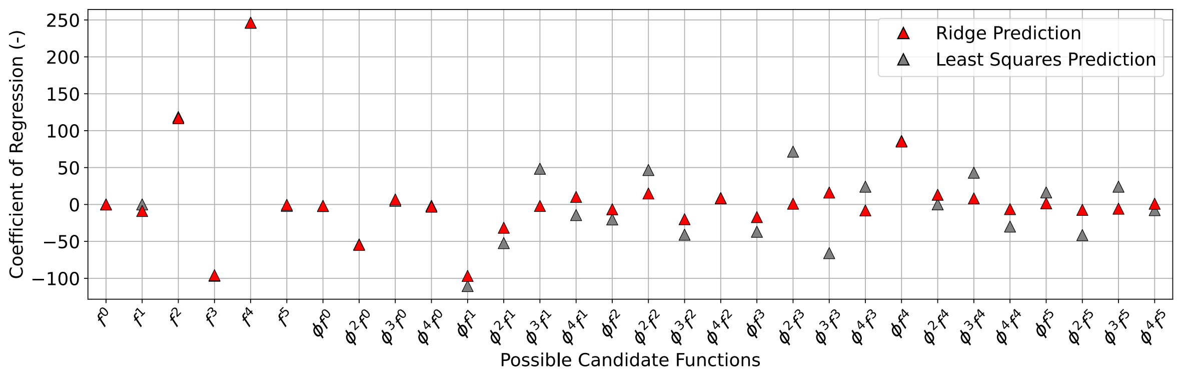


Figure 6.16: The normalised coefficient prediction of least squares and ridge regression for the exponential case for a full library of terms.

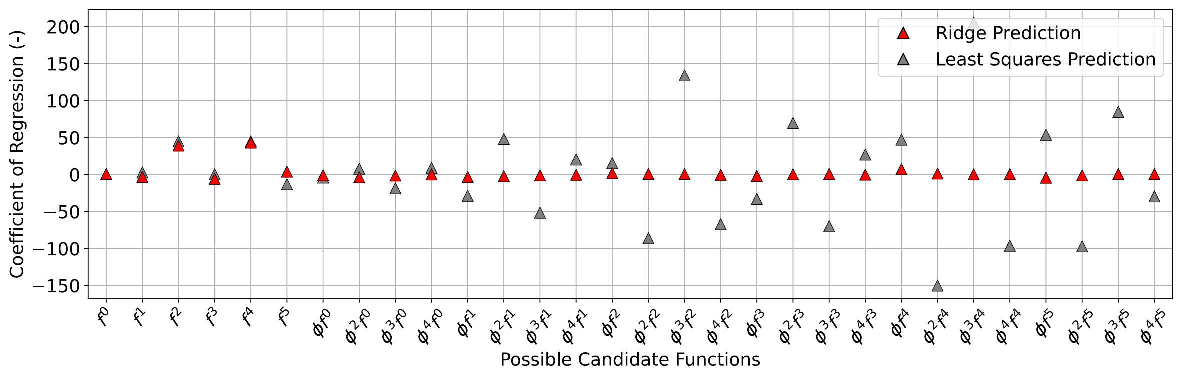


Figure 6.17: The normalised coefficient prediction of least squares and ridge regression for the shear case for a full library of terms.

Figures 6.18, 6.19 and 6.20 show the z-score and the amount of iterations for the regression of the trained models $M_{1,F,1}$, $M_{2,F,2}$ and $M_{3,F,1}$. The z-score is a measure for how likely a function is a part of the solution, obtained from the book of Hastie [84]. The greater the z-score, the greater the likelihood, and it is defined as

$$z_i = \frac{\beta_k}{\sigma_y \sqrt{(X_{ji}X_{jk})^{-1}}} \tag{6.2}$$

In (6.2), β is the coefficient, σ_y is the standard deviation of the target test data and z the z-score. As explained in Section 4.4 of Chapter 4, the standard deviation is taken as the standard deviation of the complete set, assuming all data is uncorrelated what is inherently false. Similar to 4.4, the standard deviation works as a scaling factor and it should not make a difference in the current section, for the current analyses.

From Figures 6.18, 6.19 and 6.20, f^2 and f^4 have the highest z-score, with the highest likelihood of being in the solution. ϕf^4 also has a significant contribution, in particular for the exponential case. Model $M_{1,F,1}$ has the most significant z-score per function, what is to be expected as the correlation between functions is more significant than for the shear and exponential case (for heatmaps of the correlation coefficient please look at Appendix F). The result corresponds to the functional form of model $M_{1,F,1}$, that has the most amount of contributions from the three models. For the model $M_{1,F,1}$, the amount of iterations is significant, implying that the error is relatively high in the beginning of the regression process and the l_0 error penalty of PDE FIND has a significant contribution to the error prediction.

For model $M_{2,F,2}$, the z-score related to the ground truth is visible and a lot of functions (in particular related to f^2) get cut at the last iteration (iteration 4), where the l_0 penalty of PDE FIND penalises the extra functions more and searches for a more sparse solution.

In model $M_{3,F,1}$, after one iteration, the z-score for a lot of functions decreases rather significantly as a few functions get eliminated after the initial elimination. The cut means that the need for highly correlated compensation functions decreases and therefore the z-score of those functions decreases. For instance. $\phi^4 f^2$ is eliminated after one iteration, leading to all other functions related to f^2 to decrease in z-score.

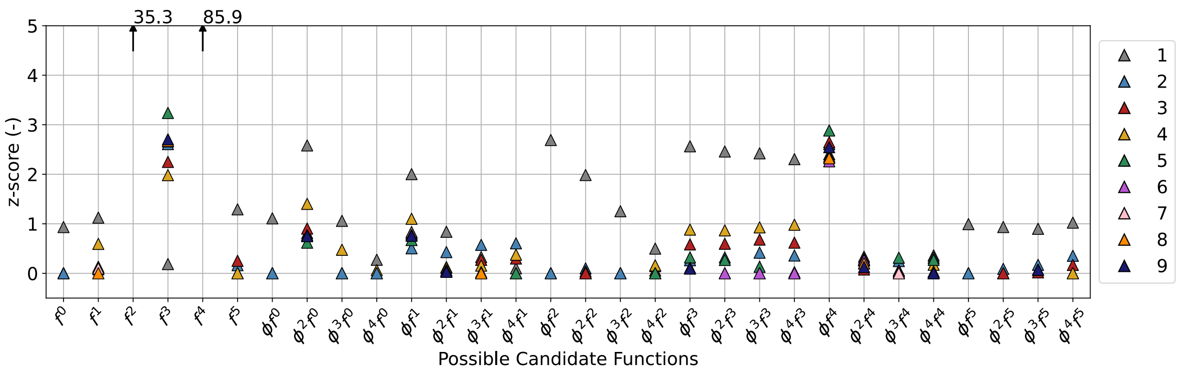


Figure 6.18: The z-score of model $M_{1,F,1}$, with a z-score of 0 meaning that a function is eliminated in the process.

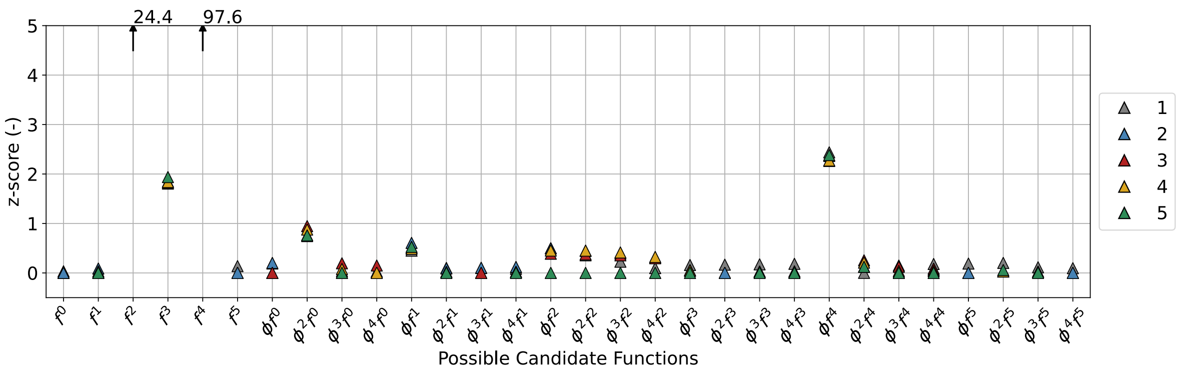


Figure 6.19: The z-score of model $M_{2,F,2}$, with a z-score of 0 meaning that a function is eliminated in the process.

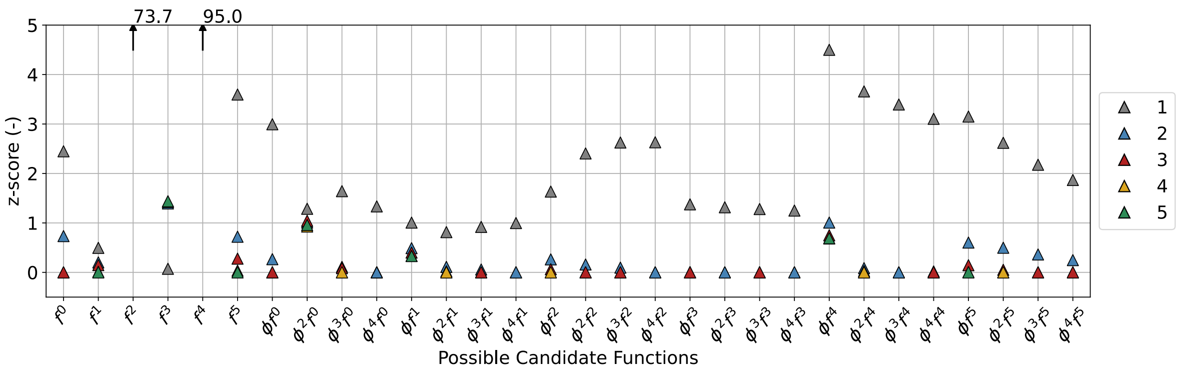


Figure 6.20: The z-score of model $M_{3,F,1}$, with a z-score of 0 meaning that a function is eliminated in the process.

6.2 The Incomplete Candidate Set Validation Results

The current section will validate the models from Section 5.3 from Chapter 5. Subsection 6.2.1 discusses the validation of the models trained on the missing $\phi^2 f^5$ value, Subsection 6.2.2 discusses the validation of the models missing the f^3 term while Subsection 6.2.3 discusses the validation of the models missing the f^3 terms with the additional library functions consisting of f^6 and f^7. Subsection 6.2.4 discusses the validation of the models where the f^4 terms are replaced with the highly correlated f^8 terms. All subsections, similar to the previous section (Section 6.1), have an in depth analyses of the regression performance of the most well functioning models per case. An overview of the models can be found in Appendix H.

6.2.1 Cross Checking for the First Incomplete Candidate Library

The current section analyses the models that have $U1$ in the model tag. An overview of the models can be found in Appendix H.

Cross Checking for the First Incomplete Candidate Library

Table 6.4 show the trained vortex and exponential models on the shear case where the $\phi^2 f^5$ term is lacking from the candidate library. The $\phi^2 f^5$ term has a small contribution to the average ϕ solution as well as the amplitude of ϕ (Figures 4.21 and 4.22). The result is that all models function well on the test case, with model $M_{2,U1,1}$ functioning with a lower than 0.1 % error.

Table 6.4: Errors of the vortex and exponential trained models tested on the shear case for the first incomplete library function case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{1,U1,1}$	0.5	0.05	1.7	0.3
$M_{1,U1,2}$	0.7	0.1	0.3	0.02
$M_{1,U1,3}$	0.3	0.09	0.6	0.1
$M_{2,U1,1}$	0.08	0.02	0.1	0.09
$M_{2,U1,2}$	0.8	0.05	0.4	0.3
$M_{2,U1,3}$	0.8	0.06	2.0	0.4

The average temporal development for the models of Table 6.4 can be found in Figures 6.21 and 6.22. The trained models performed well, and only models $M_{1,U1,1}$ and $M_{2,U1,3}$ show a notable deviation, but the performance is still acceptable.

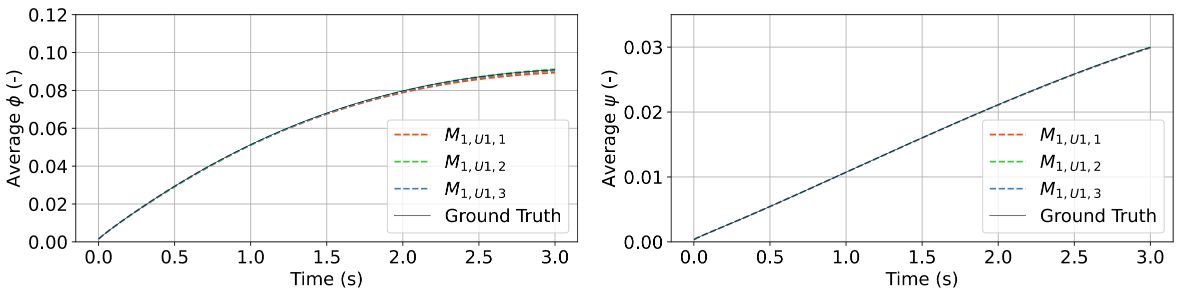


Figure 6.21: The error per time step of the three trained models for the vortex case missing the $\phi^2 f^5$ term tested on the shear case, with ϕ on the left and ψ on the right.

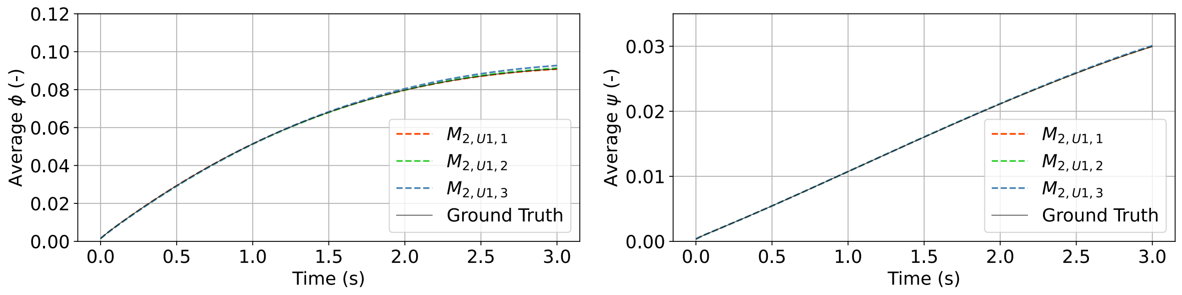


Figure 6.22: The error per time step of the three trained models for the exponential case missing the $\phi^2 f^5$ term tested on the shear case, with ϕ on the left and ψ on the right.

Figures 6.23 and 6.24 compares the trained models to the ground truth. The vortex trained functions tend to perform better than the exponential trained function at around the y = 0 station, with model $M_{1,U1,1}$ following the ground truth the best at the given location. The models trained with the exponential case tend to over predicted the production value (the gradient driven functions), what is unexpected given the errors provided in Table 6.4, but note that the figures provide one moment and does not reflect the whole field.

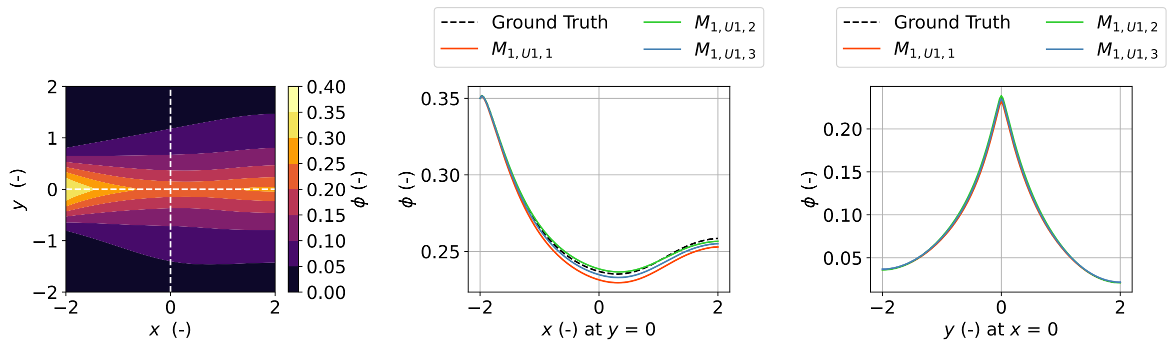


Figure 6.23: Cross checking between models for the models trained with the exponential data tested on the shear case for the trained cases where the $\phi^2 f^5$ term is lacking from the candidate library.

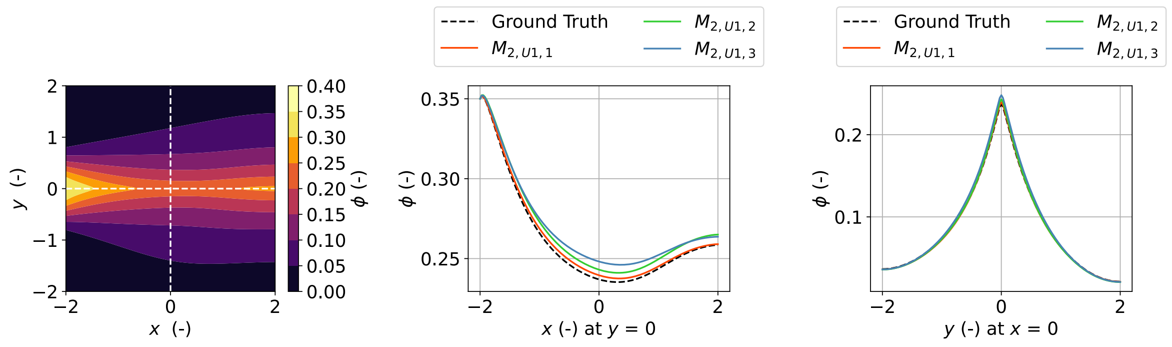


Figure 6.24: Cross checking between models for the models trained with the vortex data tested on the shear case for the trained cases where the $\phi^2 f^5$ term is lacking from the candidate library.

Deep Dive for Models with the First Incomplete Candidate Library

The best performing models are $M_{1,U1,3}$ and $M_{2,U1,1}$ per trained velocity case. Model $M_{1,U1,3}$ consists of more terms that the true functional form, having the terms ϕf^2, $\phi^2 f^2$, $\phi^3 f^2$, $\phi^4 f^2$, $\phi^4 f^3$ and $\phi^3 f^4$ extra. All the f^2 related functional forms are correlated with other f^2 and explains the addition of the f^2 terms (Appendix F, Figure F.2 for the correlation heat map). The $\phi^2 f^4$ term is not present, rather the $\phi^3 f^4$ term is present, a term that is highly correlated with the original term. The missing $\phi^2 f^5$ term is quite correlated with the $\phi^3 f^4$, what could also explain the presence of that term.

For $M_{2,U1,1}$, also a less sparse then the true functional form is found, with the functions f^1, $\phi^2 f^1$, $\phi^3 f^1$, $\phi^4 f^1$, ϕf^3, $\phi^2 f^3$, $\phi^3 f^3$, $\phi^4 f^3$ and $\phi^3 f^4$ extra. Similar to $M_{1,U1,3}$ the presence of many f^1 and f^3 terms can be explained by the correlation among one another (Appendix F, Figure F.3 for the heat map). The $\phi^2 f^5$ correlate well with the f^3 term and the contribution of the term could have been compensated by including multiple f^3 terms.

Figures 6.25 and 6.26 show the least squares and ridge regression steps. In both figures, a high correlation is found among the f^3 terms, while for the vortex model, a high correlation is found among the f^2 terms and for the exponential model among the f^1 terms. In Figure 6.25, it is prevalent that the ridge regression step does not mute all the terms regarding f^2, showing the aforementioned positive-negative fluctuation of highly correlated terms. The result is a non-sparse trained model $M_{1,U1,3}$. The high correlation of the forms could be due to the lack of $\phi^2 f^5$ as those functional forms do correlate well with $\phi^2 f^5$ (Appendix F, Figures F.3 and F.2 for the correlation heat map of both cases).

A similar explanation for the exponential case can be made, where the terms regarding f^1 are not damped as much by ridge, explaining the structure of model $M_{2,U1,1}$. As the f^1 and f^0 terms are all destruction terms, it can also be related as a redistribution of terms, giving a different formulation for the destruction terms. In general, similar to Subsection 6.1.4, the terms with the greatest variance from Appendix F Figure F.1 are damped out the most by ridge regression.

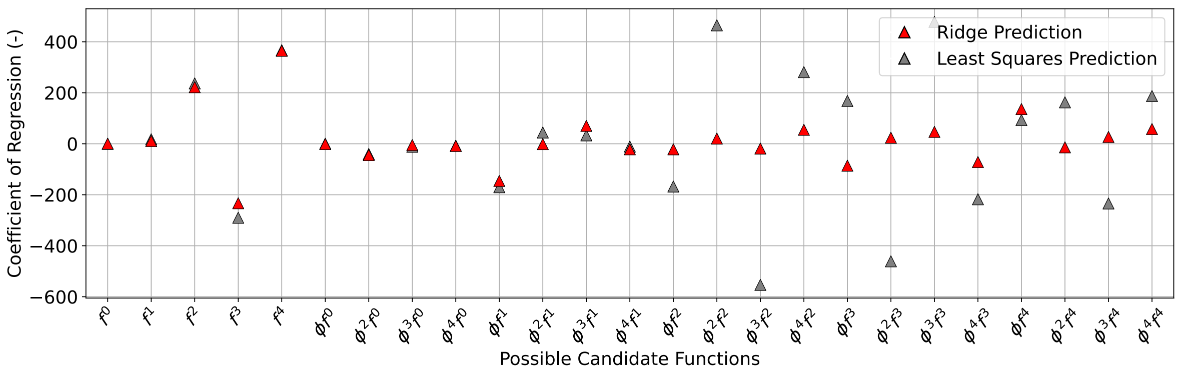


Figure 6.25: The normalised coefficient prediction of least squares and ridge regression for the vortex case for an incomplete library missing the f^5 related terms.

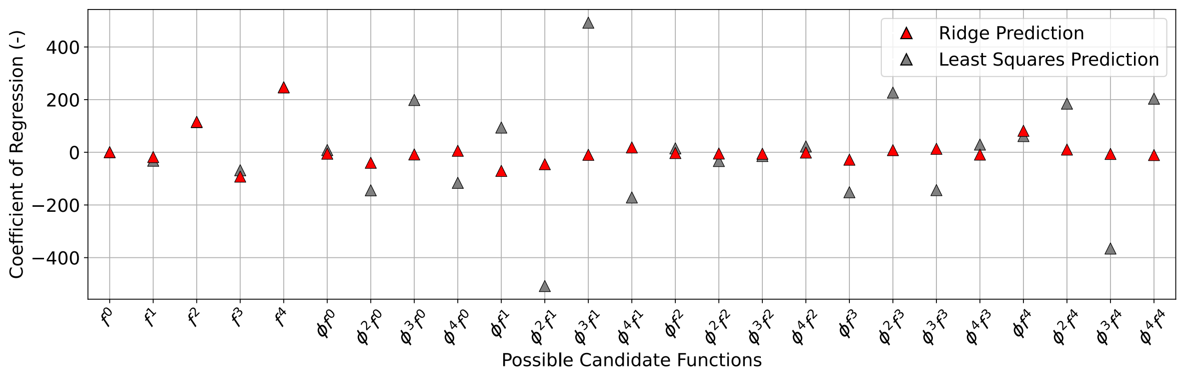


Figure 6.26: The normalised coefficient prediction of least squares and ridge regression for the exponential case for an incomplete library missing the f^5 related terms.

The z-score and the amount of iterations until convergence of models $M_{1,U1,3}$ and $M_{2,U1,1}$ are given in Figure 6.27 and 6.28. Similar to Section 6.1, f^2 and f^4 have a high z-score. For $M_{1,U1,3}$, the z-score is high for the f^2 related functions and although per iteration the z-score reduces, it is still significant. An increase in the amount of iterations in the STRidge algorithm might push the those coefficients to 0, but it is not guaranteed. A similar conclusion can be drawn for the f^1 functions in $M_{2,U1,1}$, where there is a downwards trend in z-score per iteration. For the f^3 related functions in model $M_{2,U1,1}$, an increase in z-score is happening per iteration. The f^3 terms are highly correlated with the missing term what might be a reason. Both models took five iterations to convergence.

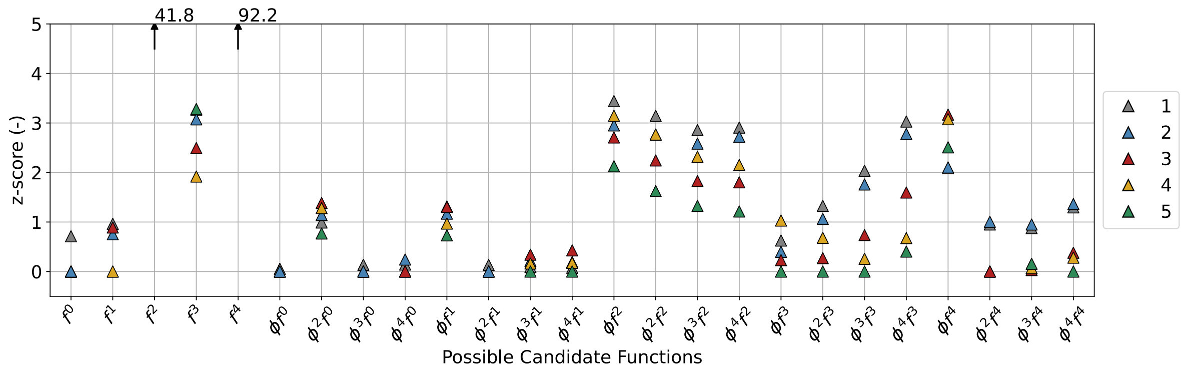


Figure 6.27: The z-score and amount of iterations for the vortex trained model $M_{1,U1,3}$, lacking the f^5 related terms in the library with a z-score of 0 meaning that a function is eliminated in the process.

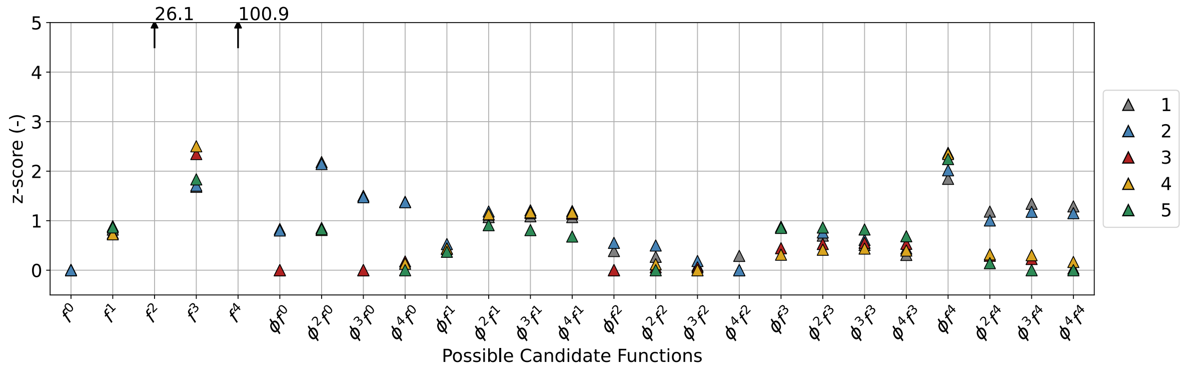


Figure 6.28: The z-score and amount of iterations for the exponential trained model $M_{2,U1,1}$, lacking the f^5 related terms in the library with a z-score of 0 meaning that a function is eliminated in the proces.

6.2.2 Cross Checking for the Second Incomplete Candidate Library

The current section analyses the models that have $U2$ in the model tag. An overview of the models can be found in Appendix H.

Cross Checking for the Second Incomplete Candidate Library

Table 6.5 show the trained models with the f^3 term lacking in the candidate library term, tested on the shear velocity case. Compared to the first case, and the complete candidate library case, the validation of the models is less well. The models trained with the exponential case show a good overall error, lower than 5 %, but the error in the final time step is off with significant errors exceeding 20 % for models $M_{2,U2,2}$ and $M_{2,U2,3}$. The vortex trained models perform worse, and the lowest ϕ error is 9.7 %.

Table 6.5: Errors of the vortex and exponential trained models tested on the shear case for the second incomplete library function case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{1,U2,1}$	12.6	1.4	28.2	4.9
$M_{1,U2,2}$	9.7	1.6	26.2	6.3
$M_{2,U2,1}$	4.7	7.7	9.8	11.5
$M_{2,U2,2}$	3.8	0.5	20.9	1.9
$M_{2,U2,3}$	3.9	0.5	23.7	2.2

Figures 6.29 and 6.30 show the temporal development of ϕ for the models. Eventhough the errors of Table 6.5 are decent, the resulting temporal behaviour is off as the result converges to different steady state values. Models $M_{2,U2,2}$ and $M_{2,U2,3}$ perform well for ψ, what means that the production term for ψ still develops on average accurate results even if ϕ is quite different.

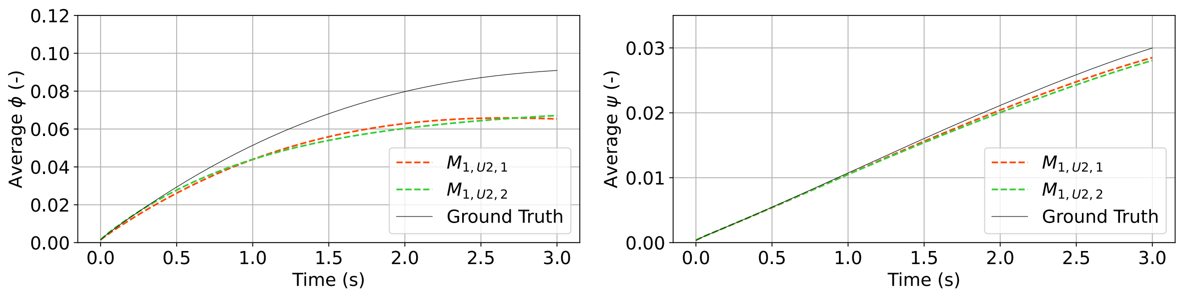


Figure 6.29: The error per time step of the two trained models for the vortex case missing the f^3 term tested on the shear case, with ϕ on the left and ψ on the right.

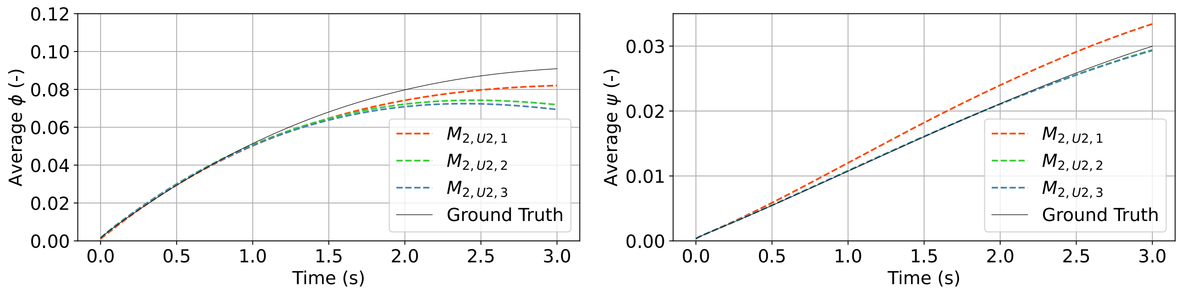


Figure 6.30: The error per time step of the three trained models for the exponential case missing the f^3 term tested on the shear case, with ϕ on the left and ψ on the right.

Figures 6.31 and 6.32 show the cross section of the models compared to the trained data. No model performs well, showing quite a difference from the boundary to the core, with different mechanisms in areas with a high gradient and different diffusive properties.

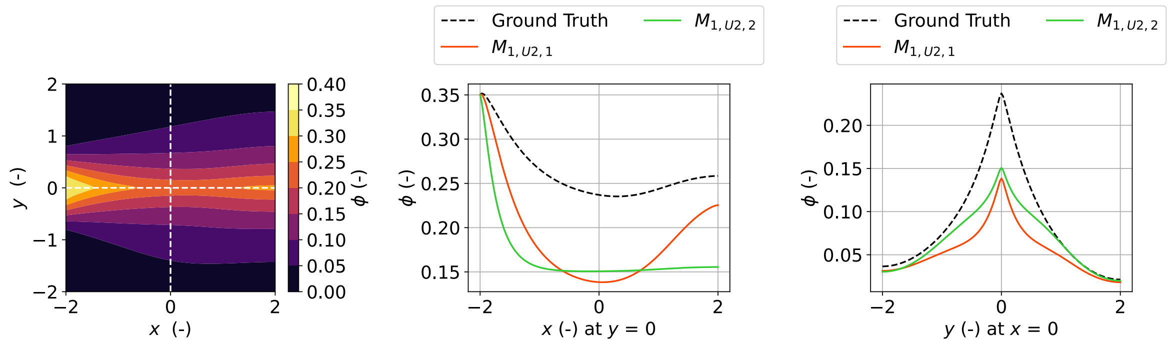


Figure 6.31: Cross checking between models for the models trained with the vortex data tested on the shear case for the trained cases where the f^3 term is lacking from the candidate library.

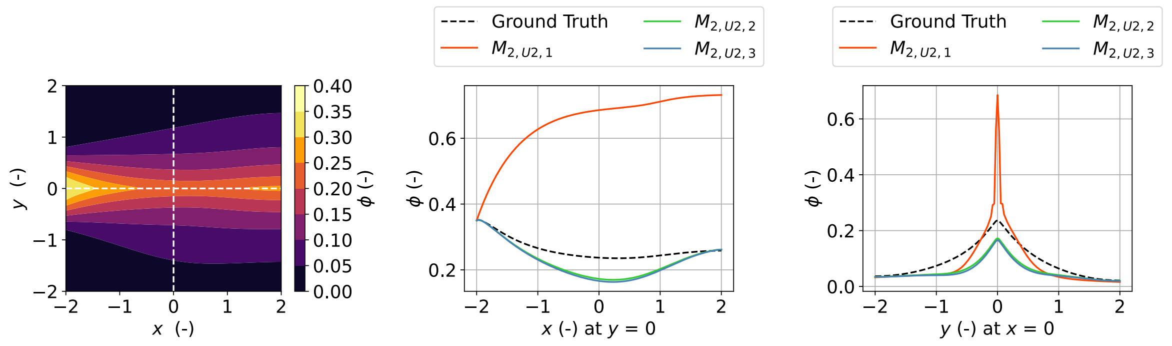


Figure 6.32: Cross checking between models for the models trained with the exponential data tested on the shear case for the trained cases where the f^3 term is lacking from the candidate library.

The result at the final time step, Figures 6.33 and 6.34 show that the results are off and qualitatively not recognisable to the original solution, Figure 4.16, meaning that no model provides satisfactory and usable behaviour.

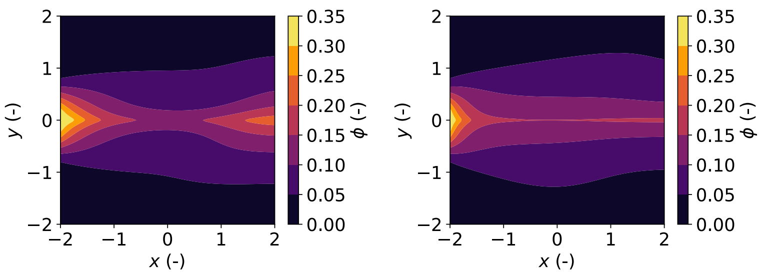


Figure 6.33: The final time steps of the models trained with the vortex data for a candidate library lacking the f^3 term with on the left model $M_{1,U2,1}$ and on the right model $M_{1,U2,2}$.

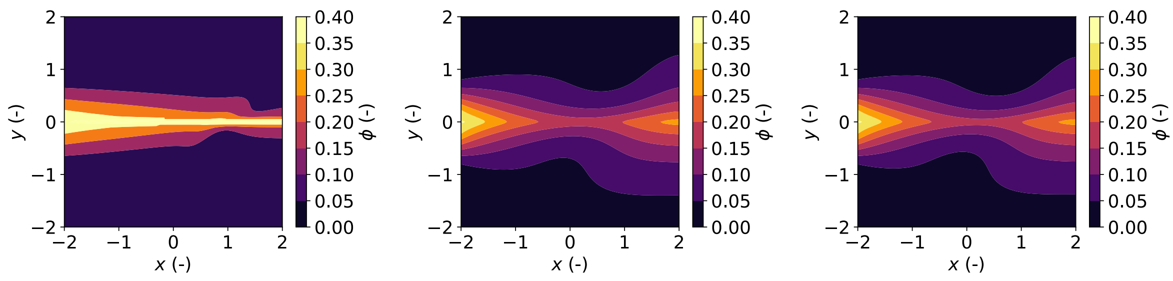


Figure 6.34: The final time steps of the models trained with the exponential data for a candidate library lacking the f^3 term with on the left model $M_{2,U2,1}$, in the middle $M_{2,U2,2}$ and on the right model $M_{2,U2,3}$.

Deep Dive for Models with the Second Incomplete Candidate Library

The best performing models per case are models $M_{1,U2,2}$ and $M_{2,U2,2}$. Model $M_{1,U2,2}$ has a large a priori error (Table 5.1 from Chapter 5), resulting due to significant values of the true model coefficients. The model also recognises a lot of functions that are correlated among one another in the same form (Appendix F Figure F.2 for the correlation heat map), as there are a lot of f^2 forms for example. The same accounts for f^0, f^1 and f^5. The f^3 term is the most correlated with the f^5 functional terms, what might explain the contribution of the f^5 terms in model $M_{1,U2,2}$.

Model $M_{2,U2,2}$ is different, as the functional form is rather short, lacking the production terms ϕf^4 and $\phi^2 f^4$, the quatric term $\phi^2 f^5$ and the destruction terms $\phi^2 f^0$ and ϕf^1. Rather, $M_{2,U2,2}$ consists of extra terms not related to the true model such as f^1 and ϕf^5. The terms ϕf^4 and $\phi^2 f^4$ could have collapsed into f^4, as the coefficient of the form is different compared to true form and the function correlates well (Appendix F Figure F.3 for the correlation heat map). From the correlation, it can be seen that $\phi^2 f^5$ could have collapsed in the form of ϕf^5 while $\phi^2 f^0$ and ϕf^1 could have collapsed in f^1. Note that both $M_{2,U2,2}$ and $M_{1,U2,2}$ are models that perform poorly to the ground truth.

Figures 6.35 and 6.36 show the least squares prediction and the ridge regression prediction. The coefficients of the least squares prediction are significant in both cases, signalling a high correlation between terms triggered by missing the the f^3 function from the regression library. For the vortex model, ridge regression is not able to shrink these coefficients enough, leaving coefficients with a large value, such as the f^2 related terms, creating a non-sparse final form of the function. Note that this is subject to the size of the hyper parameter λ. For the exponential data, also significant coefficients are found, but these are just the forms of f^1 and ridge regression is able to shrink them well. The large coefficients exhibit a positive-negative fluctuation, related to the correlation among the terms (Appendix F Figure F.2 and Figure F.3 for the correlation heat maps)

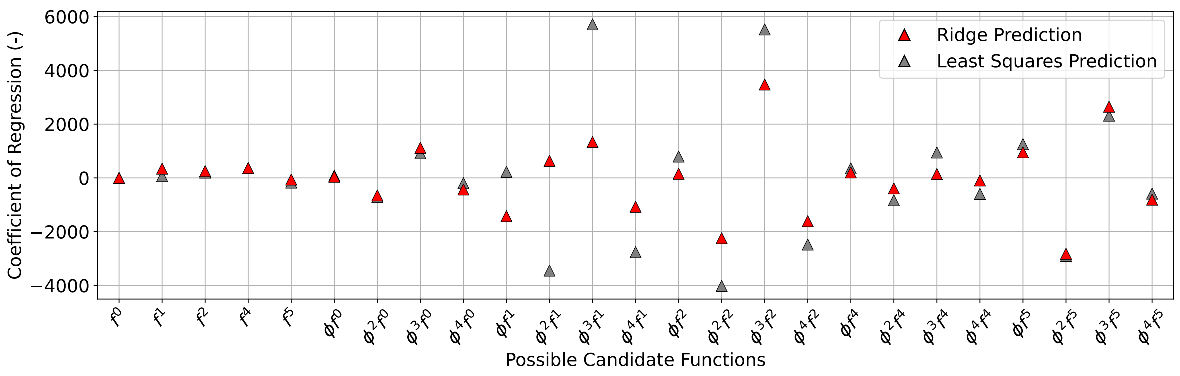


Figure 6.35: The normalised coefficient prediction of least squares and ridge regression for the vortex case for an incomplete library missing the f^3 related terms.

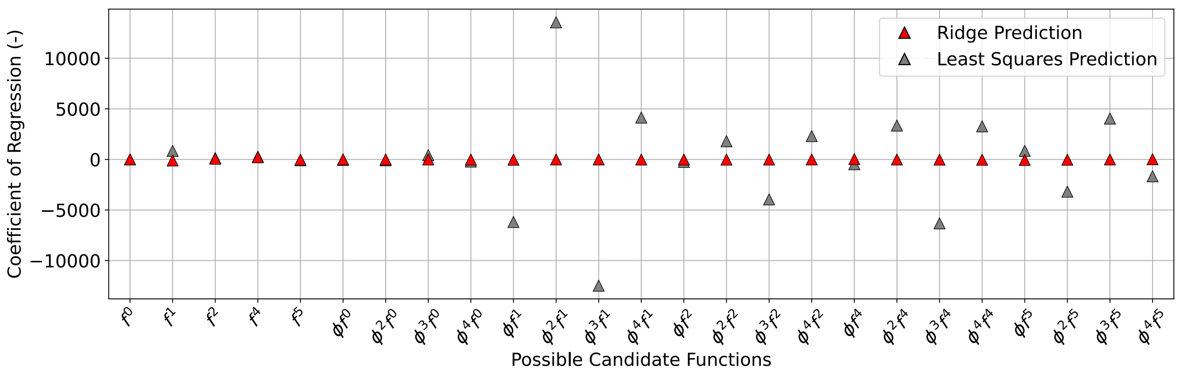


Figure 6.36: The normalised coefficient prediction of least squares and ridge regression for the exponential case for an incomplete library missing the f^3 related terms.

The z-score in Figures 6.37 and 6.38 reflect the findings of Figures 6.35 and 6.36. Model $M_{1,U2,2}$ shows a high z-score for a lot of values, pushing all other quantities towards 0 where the z-score is less than 5. As the z-score is a measure of a function that most likely to be in the solution, the z-score reflects the functional form as an highly non-sparse solution. Four iterations are needed for convergence, showing that the l^0 error penalty has a difficult time penalising the non-sparse and over fit nature of the trained model. Increasing the l^0 penalty could provide more sparse results, but is not tested in the current work.

Model $M_{2,U2,2}$ is different. More iterations are needed for convergence, showing the strength of the l^0 penalty as the addition of a function introduces a less significant reduction in error than the the l^0 penalty. A significant amount of the functional forms show a significant z-score, with the z-score most of the time lowered per step. The f^2 and f^4 terms have the greatest z-score, similar to previous cases. Some of the true functional forms increase in importance per iteration step, such as $\phi^2 f^5$ and ϕf^1 but are lowered in the end. Terms like f^1 and ϕf^5 start with a greater z-score and have a better z-score throughout the process compared to the true highly correlated counterparts, acting as potential highly correlated replacement functions for the true counterparts.

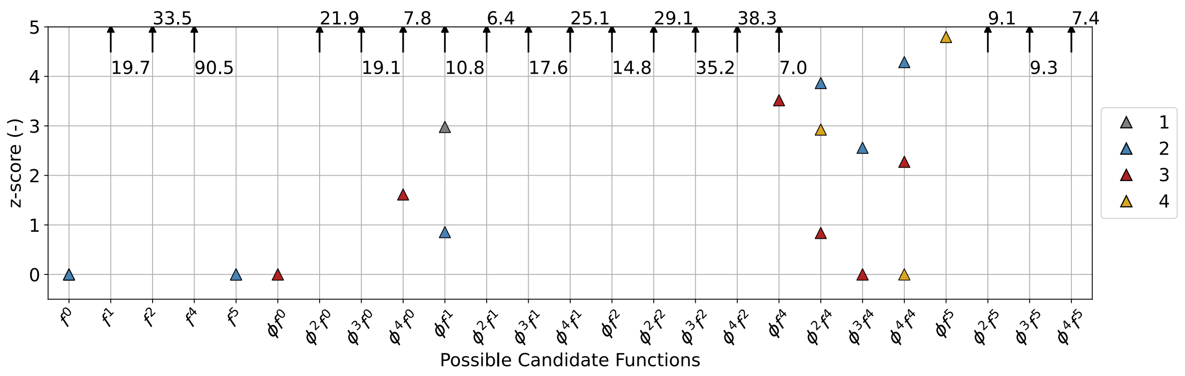


Figure 6.37: The z-score and amount of iterations for the vortex trained model $M_{1,U2,2}$, lacking the f^3 related terms in the library with a z-score of 0 meaning that a function is eliminated in the process.

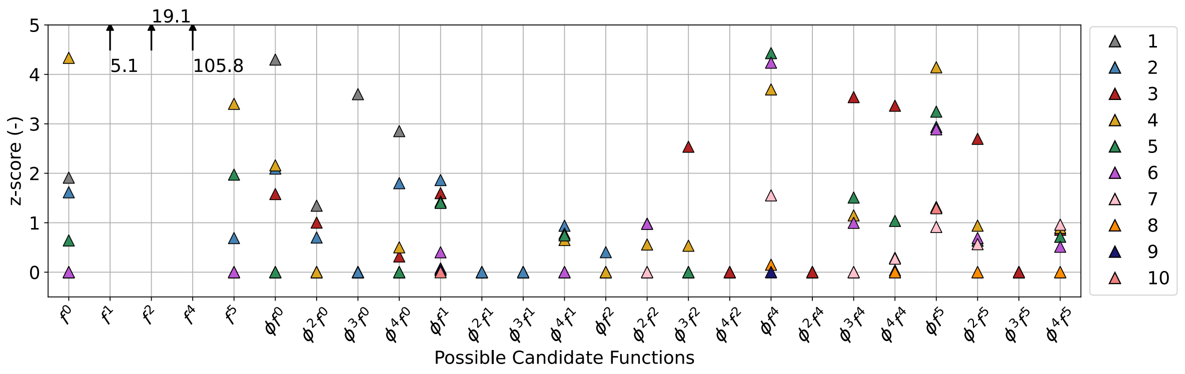


Figure 6.38: The z-score and amount of iterations for the exponential trained model $M_{2,U2,2}$, lacking the f^3 related terms in the library with a z-score of 0 meaning that a function is eliminated in the process.

6.2.3 Cross Checking for the Third Incomplete Candidate Library

The current section analyses the models that have $U3$ in the model tag. An overview of the models can be found in Appendix H.

Cross Checking for the Third Incomplete Candidate Library

Table 6.6 shows the models for the third case where additional highly correlated functions f^6 and f^7 are added to the a candidate library lacking the f^3 term. The trained models perform significantly better than shown in Subsection 6.2.2, with $M_{1,U3,2}$ having an error lower than 3 % for ϕ and lower than 0.5% for ψ and model $M_{2,U3,3}$ having an error for ϕ lower than 1% and for ψ an error lower than 0.2 %. The results are peculiar, as the trained models perform less well on their own training data in Subsection 5.3.3 of Chapter 5 Table 5.4 then they do on the testing data.

Table 6.6: Errors of the vortex and exponential trained models tested on the shear case for the third incomplete library function case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{1,U3,1}$	4.8	0.6	14.8	2.4
$M_{1,U3,2}$	2.7	0.2	0.7	0.2
$M_{1,U3,3}$	9.4	1.5	29.1	6.6
$M_{2,U3,1}$	4.5	0.8	6.6	1.9
$M_{2,U3,2}$	4.7	7.7	7.5	13.5
$M_{2,U3,3}$	1.0	0.1	0.7	0.2

The temporal behaviour of the trained models for Table 6.6 are shown in Figures 6.39 and 6.40. In particular model $M_{1,U3,2}$ and $M_{2,U3,3}$ show near ground truth accuracy. For models $M_{1,U3,1}$ and $M_{2,U3,2}$, the temporal behaviour of ψ is quite satisfactory. Models $M_{1,U3,3}$ and $M_{2,U3,2}$ are the worst performing per case, but the latter model still functions well.

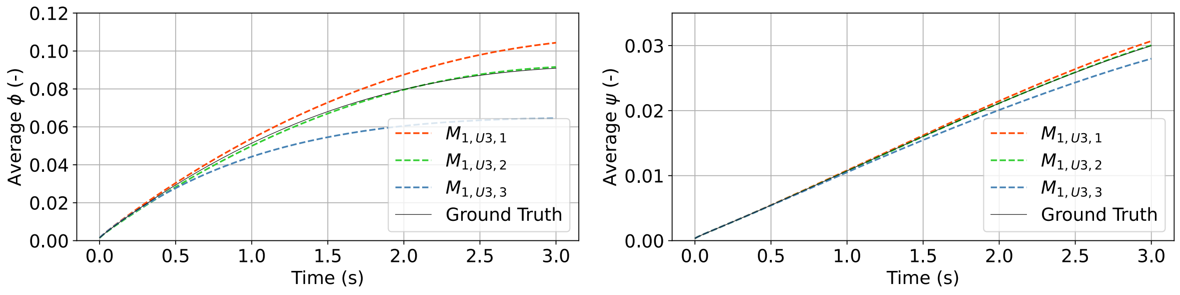


Figure 6.39: The average of $\psi - \phi$ of the three trained models for the vortex case missing the f^3 term with a additional terms including f^6 and f^7 tested on the shear case, with ϕ on the left and ψ on the right.

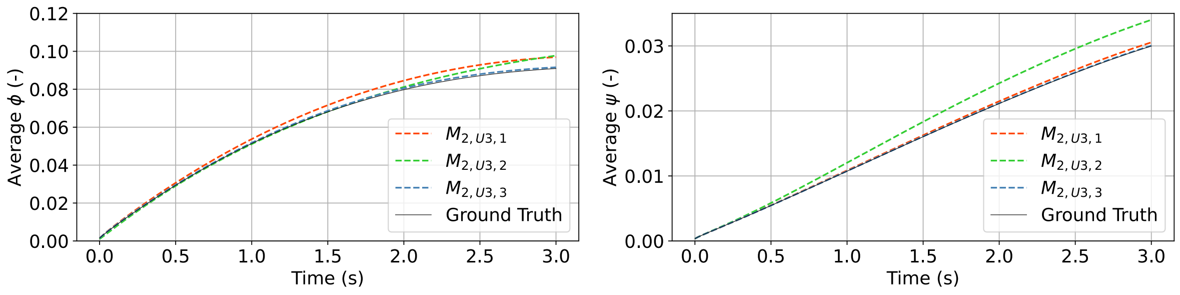


Figure 6.40: The average of $\psi - \phi$ of the three trained models for the exponential case missing the f^3 term with a additional terms including f^6 and f^7 tested on the shear case, with ϕ on the left and ψ on the right.

Figures 6.41 and 6.42 show the cross cut comparisons. Model $M_{1,U3,2}$, the best performing vortex trained model, shows some inaccuracy compared to the ground truth, having a different behaviour with the production at $y = 0$, close to the right border. Model $M_{2,U3,3}$, the exponential trained model, follows the ground truth rather well with a small deviation near the right border.

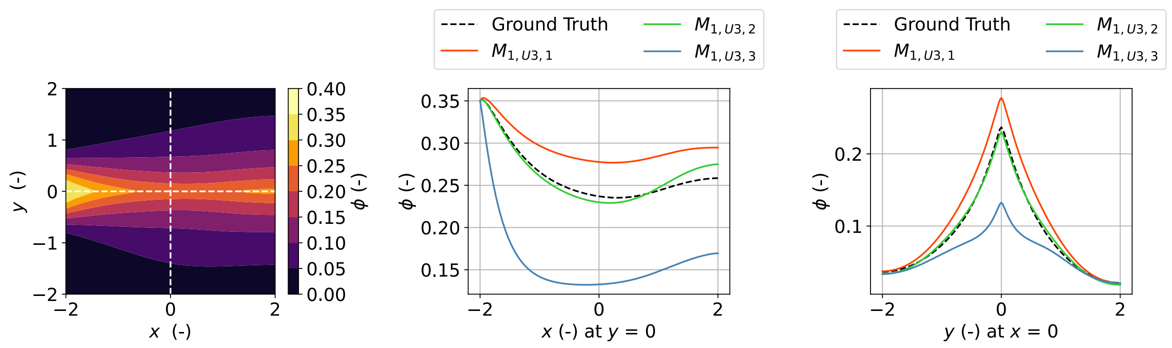


Figure 6.41: Cross checking between models for the models trained with the vortex data tested on the shear case for the trained cases where the f^3 term is lacking from the candidate library but with additional f^6 and f^7 terms.

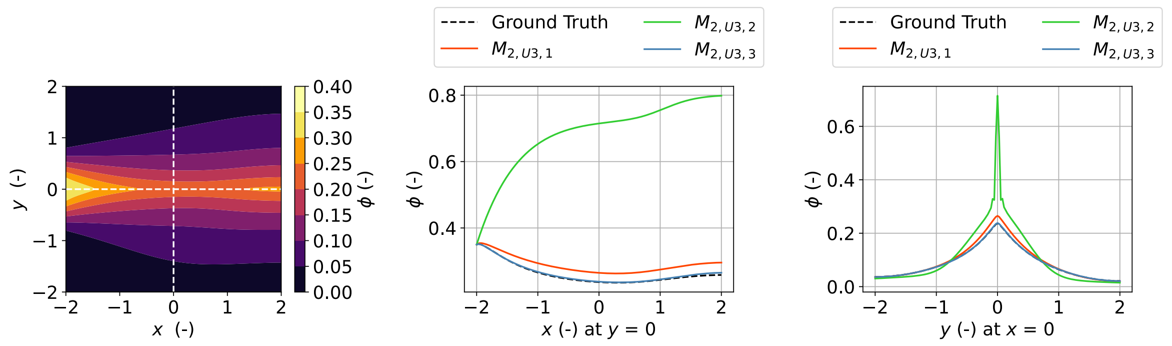


Figure 6.42: Cross checking between models for the models trained with the exponential data tested on the shear case for the trained cases where the f^3 term is lacking from the candidate library but with additional f^6 and f^7 terms.

The results from Figures 6.39 and 6.40 are visualised in the final step plot of ϕ in Figures 6.43 and 6.44, where model $M_{1,U3,2}$ and $M_{2,U3,3}$ shows similar qualitative behaviour compared to the ground truth in Figure 4.16.

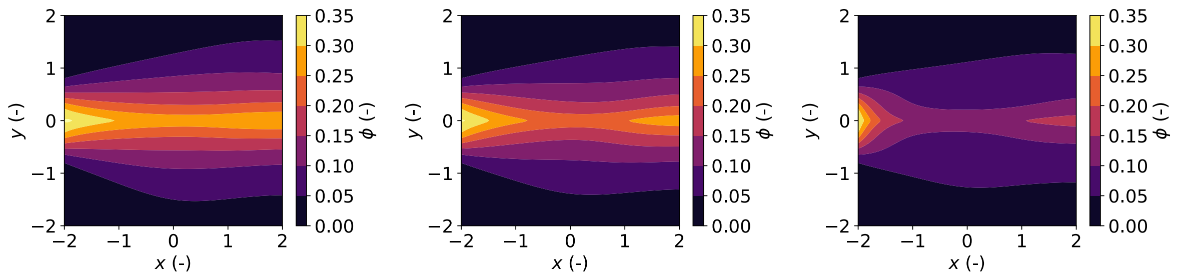


Figure 6.43: The final time steps of the models trained with the vortex data for a candidate library lacking the f^3 term with additional f^6 and f^7 terms with on the left model $M_{1,U3,1}$, in the middle $M_{1,U3,2}$ and on the right model $M_{1,U3,3}$.

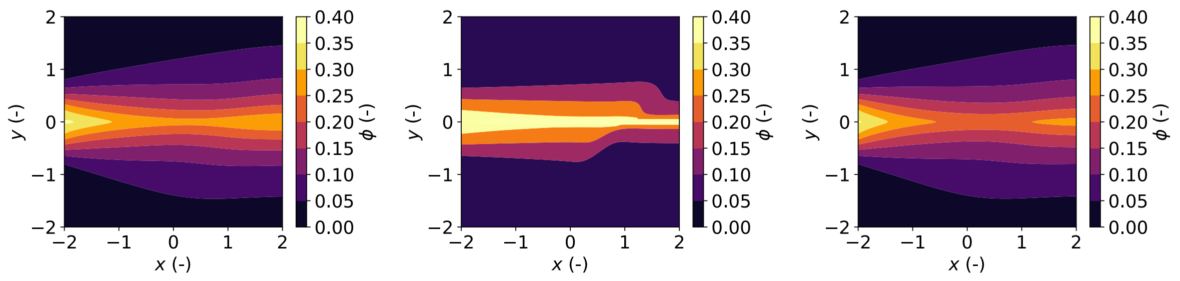


Figure 6.44: The final time steps of the models trained with the exponential data for a candidate library lacking the f^3 term with additional f^6 and f^7 terms with on the left model $M_{2,U3,1}$, in the middle $M_{2,U3,2}$ and on the right model $M_{2,U3,3}$.

It is peculiar that the trained models in the current section perform better on the test data than on their own training data. To verify the statement, the models have been tested among one-another, so the vortex trained model with exponential test settings and visa versa. The results are noted in Table 6.7. The models perform worse than tested on the shear case, and are in line with the error in the model training phase, with model $M_{1,U3,2}$ diverging. Only model $M_{2,U3,3}$ has shown decent behaviour, where the general error is low. For model $M_{2,U3,3}$, note that the final time step error is off and a better final time step error for ϕ is found with Model $M_{2,U3,2}$.

Table 6.7: Errors of the vortex and exponential trained models tested on the vortex case for the exponential models and tested on the exponential models for the vortex models, for trained models with the third incomplete library function case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{1,U3,1}$	8.9	2.4	27.4	9.5
$M_{1,U3,2}$	10.2	3.1	39.2	16.0
$M_{1,U3,3}$	NAN	NAN	NAN	NAN
$M_{2,U3,1}$	15.8	8.5	46.0	26.1
$M_{2,U3,2}$	8.1	11.0	5.1	25.3
$M_{2,U3,3}$	2.8	1.7	12.9	4.2

The result from Table 6.7 are seen in Figures 6.45 and 6.46, showing the average development of ψ and ϕ. It can be seen that models $M_{2,U3,2}$ and $M_{2,U3,3}$ performed decently, with model $M_{2,U3,2}$ being closer to the steady state value compared to model $M_{2,U3,3}$. Model $M_{2,U3,3}$ has a better initial development than model $M_{2,U3,2}$. The vortex trained models, $M_{1,U3,1}$ and $M_{1,U3,2}$ are both off and perform poorly with different behaviour overall.

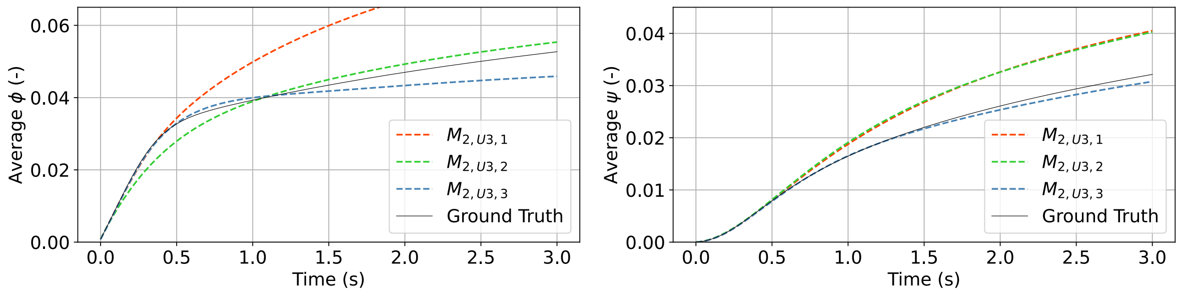


Figure 6.45: The average of $\psi - \phi$ of the three trained models for the exponential case missing the f^3 term with a additional terms including f^6 and f^7 tested on the vortex case, with ϕ on the left and ψ on the right.

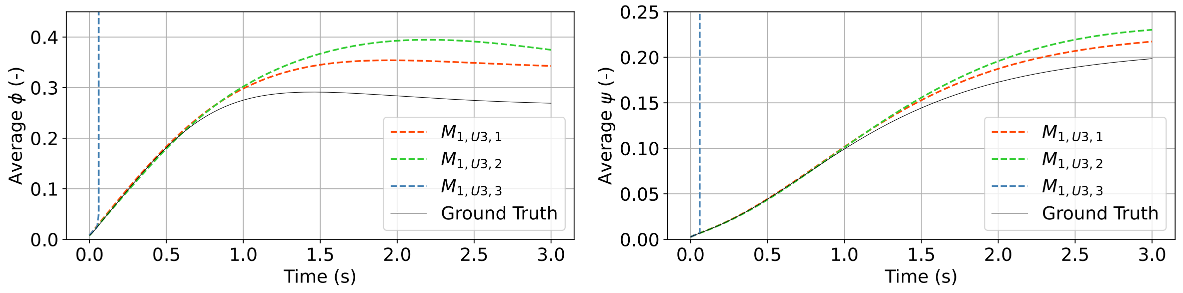


Figure 6.46: The average of $\psi - \phi$ of the three trained models for the vortex case missing the f^3 term with a additional terms including f^6 and f^7 tested on the exponential case, with ϕ on the left and ψ on the right.

Figures 6.47 and 6.48 show the cross sections of the trained models and the ground truth. No particular model has a perfect fit, but $M_{2,U3,3}$ shows decent prediction of the slope and the core of the vortex and has similar characteristics compared to the ground truth. Model $M_{2,U3,3}$ is the best performing model from Subsection 5.3.3 of Chapter 5 and the results are in line with the findings in that section. Model $M_{2,U3,3}$ is the only decent functioning model lacking the functional form f^3 term, using additional f^6 and f^7 terms, even if there is a small deviation with the value in the final step.

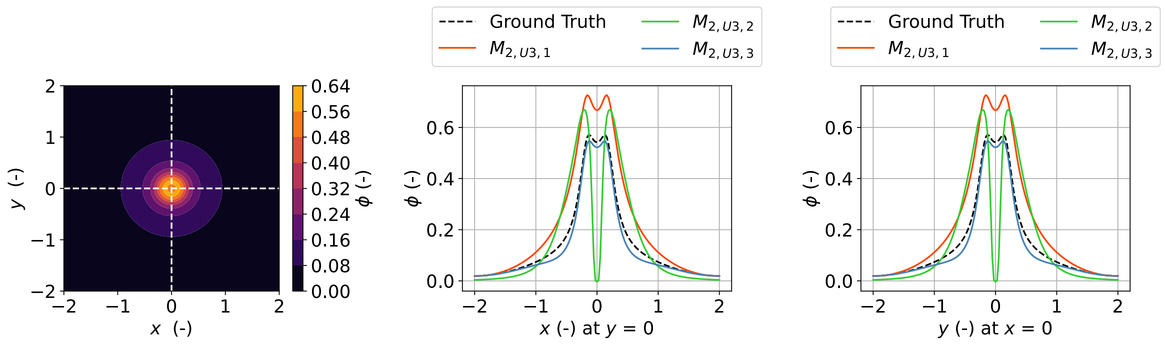


Figure 6.47: Cross checking between models for the models trained with the exponential data tested on the vortex case for the trained cases where the f^3 term is lacking from the candidate library but with additional f^6 and f^7 terms.

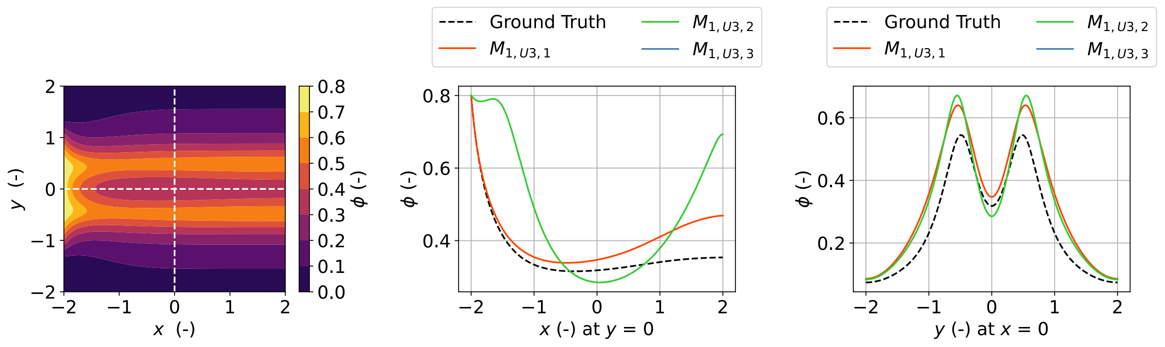


Figure 6.48: Cross checking between models for the models trained with the vortex data tested on the exponential case for the trained cases where the f^3 term is lacking from the candidate library but with additional f^6 and f^7 terms.

Deep Dive for Models with the Third Incomplete Candidate Library

The best performing models per trained velocity case are the $M_{1,U3,2}$ and $M_{2,U3,3}$ models. The $M_{1,U3,2}$ model has a single additional ϕf^7 term from the additional candidate options while also having an additional $\phi^3 f^1$ and ϕf^5 term in the overall model. The model lacks the $\phi^2 f^0$, $\phi^2 f^4$ and $\phi^2 f^5$ terms of the ground truth. The lack of $\phi^2 f^0$ can be explained by the high correlation with f^1 and f^0 terms (Appendix F, Figure F.2), resulting in a redistribution of the destruction term. The $\phi^2 f^4$ term correlates well with the functions of f^4, and can be explained in the different coefficient of the f^4 terms compared to the true functional form. In a similar extent, the loss of $\phi^2 f^5$ be explained by the gain of ϕf^5. ϕf^7 is the only form to be introduced of the candidate library, and is highly correlated with f^3.

For $M_{2,U3,3}$, the f^7 term is the only new term from the new candidate library functions. The model lacks the $\phi^2 f^4$ and $\phi^2 f^5$ terms with an addition in the f^1, $\phi^2 f^1$ and ϕf^5 term. Similar to model $M_{1,U3,2}$, the lack of $\phi^2 f^5$ and $\phi^2 f^4$ can be explained by the highly correlated functional forms containing f^5 and f^4. Similar to model $M_{1,U3,2}$, the f^7 term is selected to replace the f^3 term and the addition of f^1 and $\phi^2 f^1$ can be seen as a redistribution of the destruction term among highly correlated terms (Appendix F, Figure F.3 for the correlation heat map). Models $M_{1,U3,2}$ and $M_{2,U3,3}$ have a similar build up of the equation with different testing results.

Figures 6.49 and 6.50 show the workings of the ridge prediction versus the least squares prediction. For both cases, the terms that are highly correlated with one another are damped by the ridge penalty, with high correlation between terms for the f^2 related functions in Figure 6.50. For Figure 6.49, the f^1, f^2 and f^4 cases are damped significantly by the ridge penalty eliminating the positive-negative coefficient behaviour discussed in Section 6.1. The cases with a high variance (Figure F.1 in Appendix F for the variance figure) are damped more significantly than the lower variance values as expected with Ridge regression.

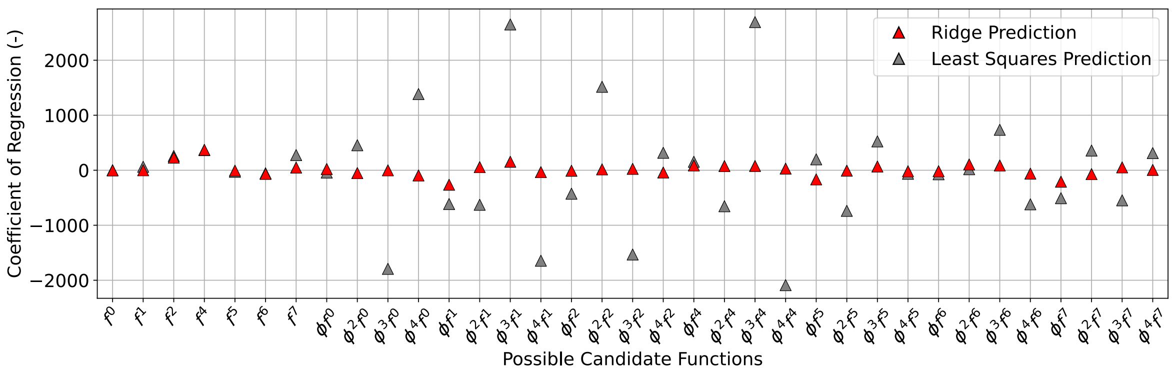


Figure 6.49: The normalised coefficient prediction of least squares and ridge regression for the vortex case for an incomplete library missing the f^3 related terms with additional f^6 and f^7 terms.

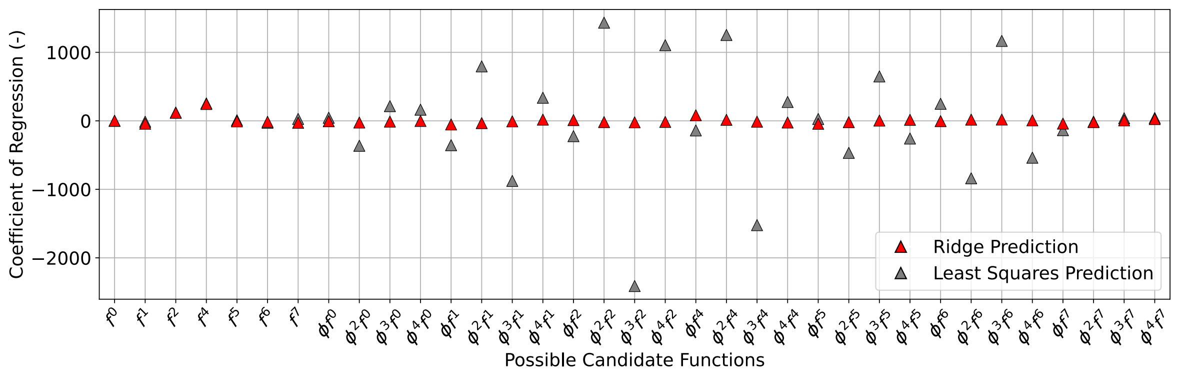


Figure 6.50: The normalised coefficient prediction of least squares and ridge regression for the exponential case for an incomplete library missing the f^3 related terms with additional f^6 and f^7 terms.

The z-score and the amount of iterations can be seen in Figures 6.51 and 6.52. Similar to previous cases, the z-scores for f^2 and f^4 are high while the z-scores of f^6 and f^7 are prevalent.

For model $M_{1,U3,2}$, all the z-scores for the added candidate functions f^6 and f^7 are high in value, with some increasing in importance during the iteration procedure before eventually being eliminated. The term that ends up in the final form does not have a high z-score from the onset and at the end of iterating only has a low value.

For model $M_{2,U3,3}$, more functions carry a higher z-score than model $M_{1,U3,2}$, in particular for functions related to f^2, f^4 and f^5. For the current case, after the first iteration, the importance on functions related to f^7 increases, but while eventually all multiples with ϕ go to zero, only f^7 survives on a low z-score, similar to model $M_{2,U3,3}$. The explanation might be that the selected functions are not compensating for the presence of other highly correlated functions (the positive-negative oscillation discussed in Section 6.1). If a function is eliminated, it might trigger highly correlated functions to be eliminated at a further stage while the low z-score function is not affected, even if highly correlated.

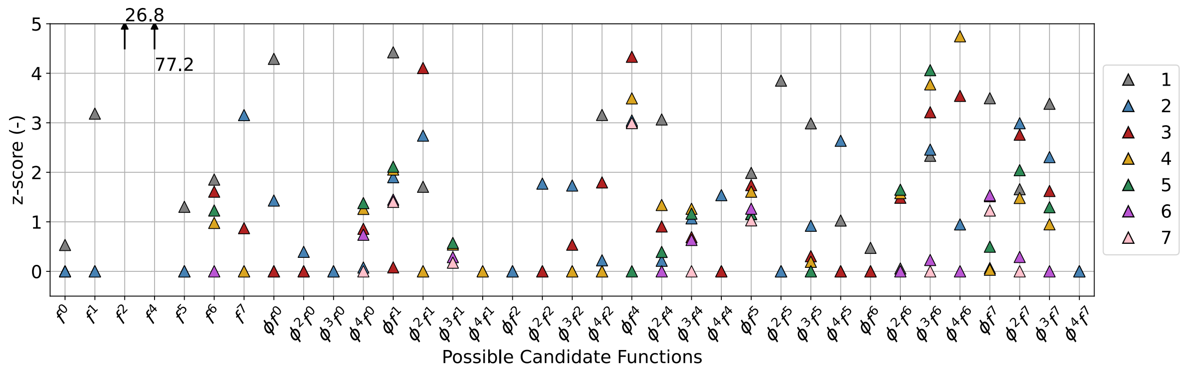


Figure 6.51: The z-score and amount of iterations for the vortex trained model $M_{1,U3,2}$, lacking the f^3 related terms in the library with additional f^6 and f^7 terms with a z-score of 0 meaning that a function is eliminated in the process..

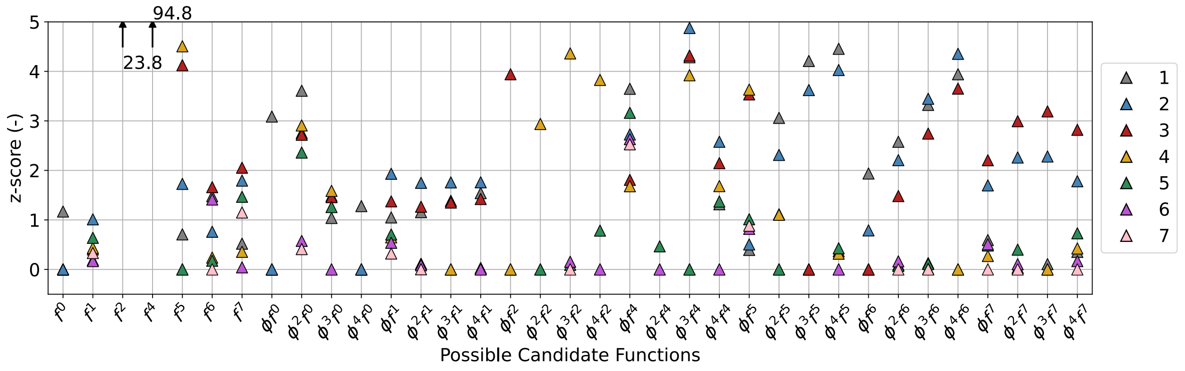


Figure 6.52: The z-score and amount of iterations for the exponential trained model $M_{2,U3,3}$, lacking the f^3 related terms in the library with additional f^6 and f^7 terms with a z-score of 0 meaning that a function is eliminated in the process..

6.2.4 Cross Checking for the Fourth Incomplete Candidate Library

The current section analyses the models that have $U4$ in the model tag. An overview of the models can be found in Appendix H.

Cross Checking for the Fourth Incomplete Candidate Library

Table 6.8, show the trained models with a candidate library not including the f^4 terms, but the highly correlated f^8 terms tested on the shear case. The numbers show quite a large error for all cases in ϕ and a moderate to acceptable error for ψ. Only the final step value of model $M_{2,U4,3}$ shows a low error, but note that the overall error for ϕ is significant.

Table 6.8: Errors of the vortex and exponential trained models tested on the shear case for the fourth incomplete library function case.

Model	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	$\xi_{post,\phi}$ at t_{fin}(%)	$\xi_{post,\psi}$ at t_{fin} (%)
$M_{1,U4,1}$	41.0	5.5	27.6	7.0
$M_{1,U4,2}$	53.3	6.3	38.6	7.8
$M_{2,U4,1}$	32.8	3.2	35.6	6.0
$M_{2,U4,2}$	31.2	2.8	23.6	3.8
$M_{2,U4,3}$	31.7	2.6	3.6	1.3

The poor errors from Table 6.8 can be seen in the temporal time progression in Figures 6.53 and 6.54. Model $M_{2,U4,3}$ has unique behaviour, with a large error at an early time stage and a moderate error when approaching steady state. Model $M_{2,U4,3}$ shows different overall behaviour compared to the other models that function more as a scaled solution of the true value. All models follow ψ well, but an error build-up is present with increasing time.

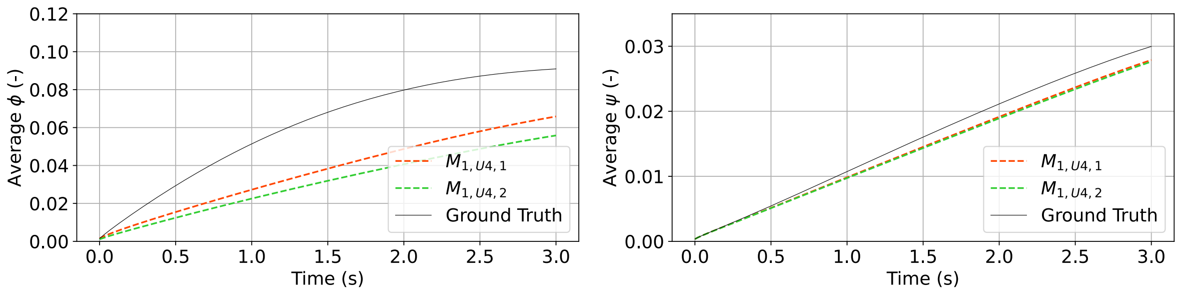


Figure 6.53: The error per time step of the two trained models for the vortex case missing the f^4 term with a additional terms including f^8 terms tested on the shear case, with ϕ on the left and ψ on the right.

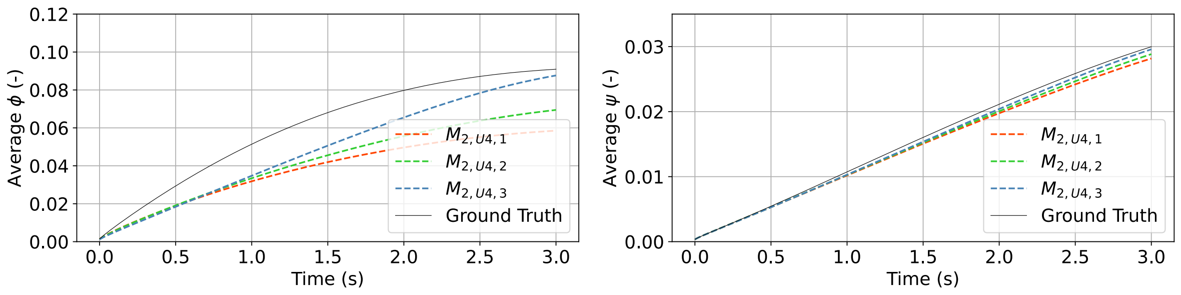


Figure 6.54: The error per time step of the three trained models for the vortex case missing the f^4 term with a additional terms including f^8 terms tested on the shear case, with ϕ on the left and ψ on the right.

The results from Table 6.8 and Figures 6.53 and 6.54 coincide with the findings in the cross-cut pictures of Figures 6.55 and 6.56, where all models show a different behaviour compared to the ground truth. All models have difficulty with prediction of the local production, underestimating the production in all cases. Model $M_{2,U4,3}$ is an exception, as it follows the ground-truth the best, but compared to earlier trained models with incomplete libraries it performs poorly and the behaviour compared to the ground truth is not similar.

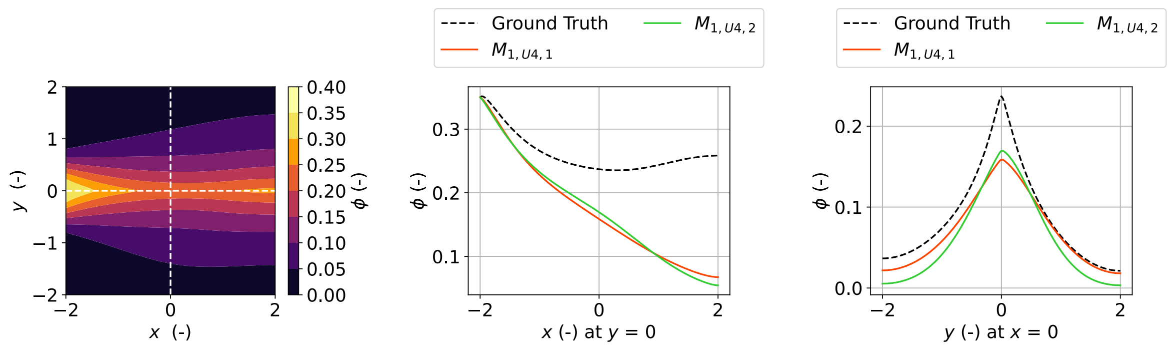


Figure 6.55: Cross checking between models for the models trained with the vortex data tested on the shear case for the trained cases where the f^4 term is lacking from the candidate library but with additional f^8 terms.

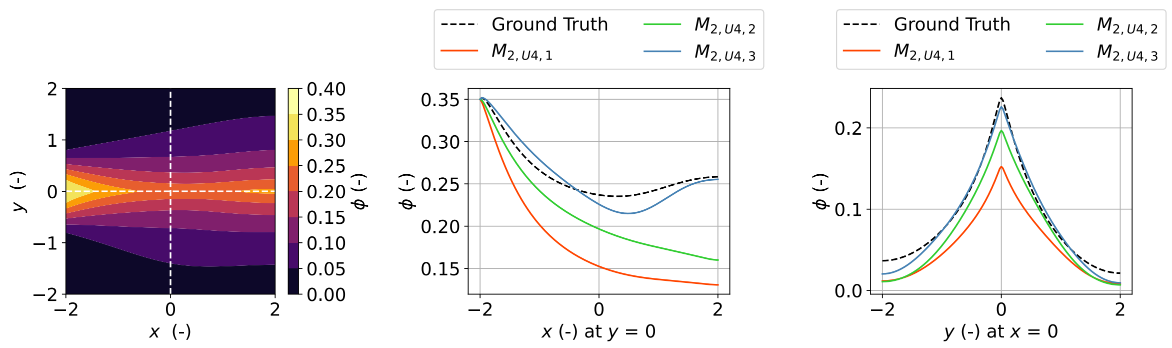


Figure 6.56: Cross checking between models for the models trained with the exponential data tested on the shear case for the trained cases where the f^4 term is lacking from the candidate library but with additional f^8 terms.

Figures 6.57 and 6.58 show the final time step of the solution and are also not recognisable compared to the ground truth, Figure 4.16. No model is able to correctly show qualitatively similar behaviour and model $M_{2,U4,3}$, that arguably has shown the best performance, shows quite different behaviour.

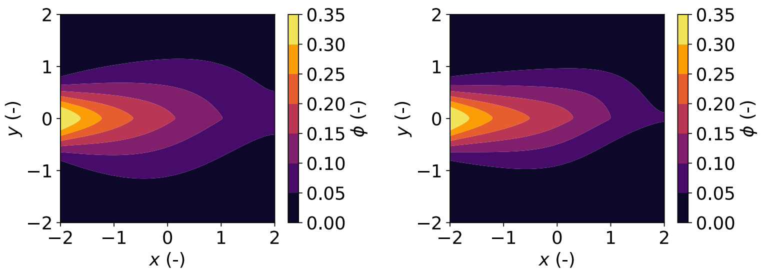


Figure 6.57: The final time steps of the models trained with the vortex data for a candidate library lacking the f^4 term with additional f^8 terms with on the left model $M_{1,U4,1}$ and on the right model $M_{1,U4,2}$.

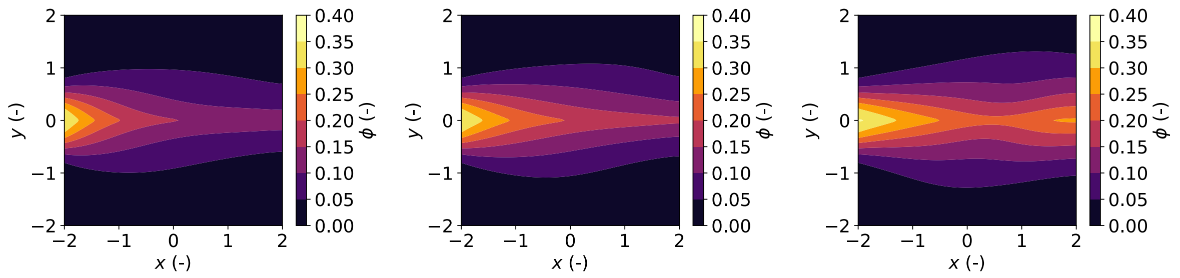


Figure 6.58: The final time steps of the models trained with the exponential data for a candidate library lacking the f^4 term with additional f^8 terms with on the left model $M_{2,U4,1}$, in the middle $M_{2,U4,2}$ and on the right model $M_{2,U4,3}$.

Deep Dive for Models with the Third Incomplete Candidate Library

The models used for a further analyses per velocity case are models $M_{1,U4,1}$ and $M_{2,U4,2}$. Model $M_{1,U4,1}$ has only the f^2 and f^3 terms of the original model, adding different function forms such as ϕf^2, $\phi^3 f^2$, f^8, ϕf^8, $\phi^2 f^8$ and $\phi^4 f^8$. All the f^8 terms are highly correlated among one another and with the replacement f^4 function (Appendix F, Figure F.2 for the correlation heat map). The result is a positive-negative oscillation, similar to Section 6.1, where significantly varying coefficients of the f^8 functions are found. The additional f^2 functions can be an explanation of the negative correlation it has with the lacking f^4

function. As the PDE FIND procedure struggles to find a decent result, additional f^2 functions might be chosen over the desired f^8. The addition of f^2 terms can also be a replacement for destruction terms as no destruction terms (f^0 and f^1) are present in the current work. The lacking destruction terms correlate negatively with f^2, but to a lesser extent than the functions containing f^8.

For model $M_{2,U4,2}$, only the f^8 term is included as an additional term, with the f^8 term needed to obtain an initial production for ϕ. Compared to the original, the solution lacks the $\phi^2 f^5$, $\phi^2 f^0$ and ϕf^1 term. It has an additional ϕf^0 and $\phi^3 f^1$ term. The destruction functions related to f^1 and f^0 are all highly correlated among one another (Appendix F, Figure F.3 for the correlation heat map) and different terms for f^1 and f^0 can be explained as the procedure favouring different highly correlated destruction terms, giving a redistribution of the terms.

Figures 6.59 and 6.60 show the difference between the least squares prediction and the ridge prediction. Ridge regression damps out multiple functions that are correlated with one another, such as the f^1 functions having high fluctuating coefficients in Figure 6.59. The f^8 related functions are damped less and explains the result that there are multiple functions with f^8 in the model $M_{1,U4,1}$. All the highly correlated destruction terms are damped well by the ridge regressor.

Model $M_{2,U4.2}$ has a more pronounced damping with the ridge regression. It reduces all the correlated functions, as in the functions related to f^3 and shrinks the coefficients more than $M_{1,U4,1}$.

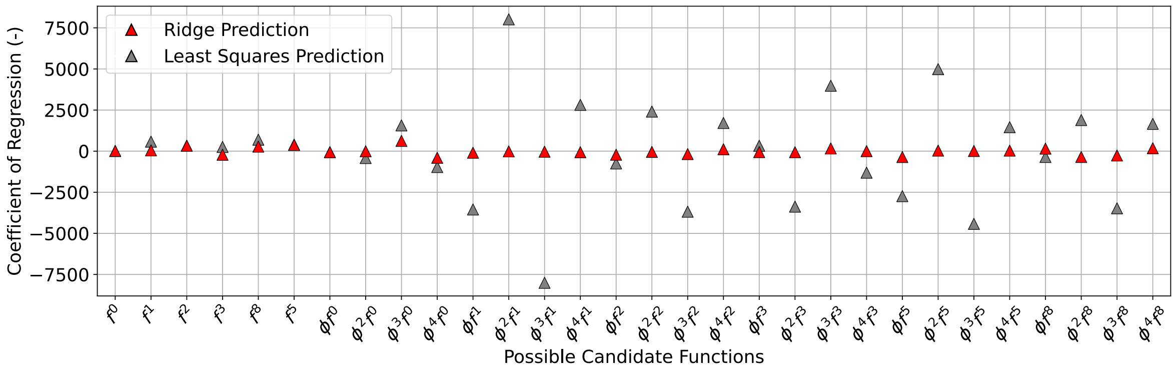


Figure 6.59: The normalised coefficient prediction of least squares and ridge regression for the vortex case for an incomplete library missing the f^4 related terms with additional f^8 terms.

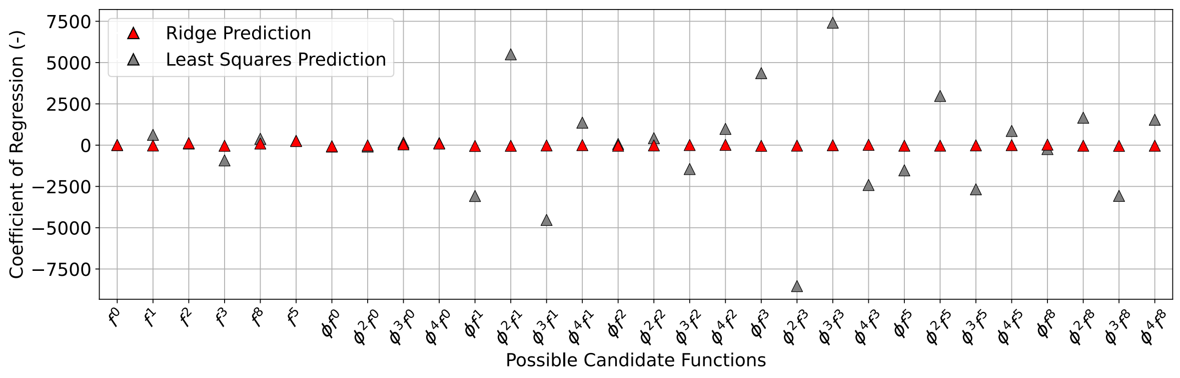


Figure 6.60: The normalised coefficient prediction of least squares and ridge regression for the exponential case for an incomplete library missing the f^4 related terms with additional f^8 terms.

Figures 6.61 and 6.62 show the z-score and the amount of iterations to convergence. The f^2 term carries a significant z-score, increasing the likelihood that it is involved. Rather than f^4 having a high z-score, f^8 has a high z-score. The f^8 term is the only function that is dependent on the velocity of the flow in the current candidate library and is able to generate ϕ from an initial condition of zero among the domain.

In Figure 6.61, multiple functions carry a z-score greater than 5, including the function ϕf^2, what is not in the true system. All the f^0 and f^1 related functions have a low z-score, meaning that ϕf^2 functions are a substitute for those functions. All the f^8 related functions have a large z-score, a consequence of the not so strong shrinkage of the ridge regression.

In Figure 6.62, a more similar picture to the true full system result is given, Figure 6.19, where a greater z-score is given for the destruction driven functions, f^0 and f^1. It is good to note that after an initial elimination of a few functions, f^2 and f^3 related functions have a rise in z-score, however the l_0 error penalty of PDE FIND works well here and penalises a less sparse solution, forcing elimination of those terms.

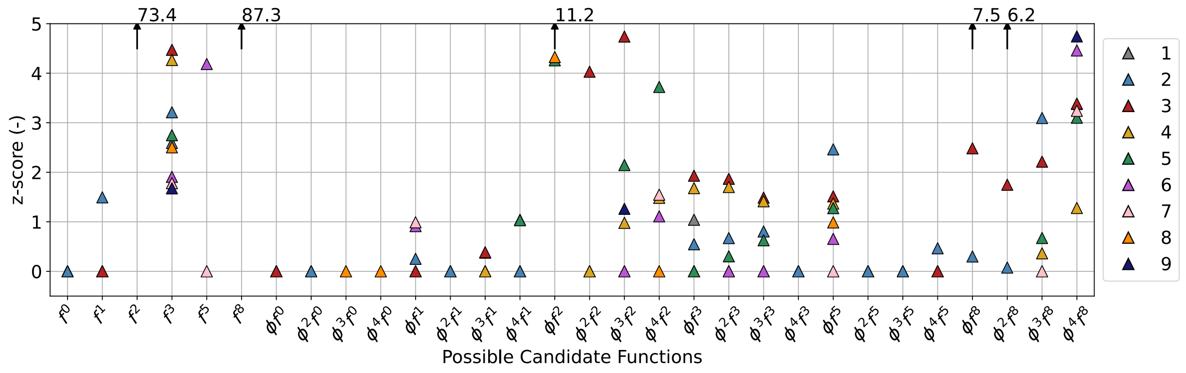


Figure 6.61: The z-score and amount of iterations for the exponential trained model $M_{1,U4,1}$, lacking the f^4 related terms in the library with additional f^8 terms with a z-score of 0 meaning that a function is eliminated in the process..

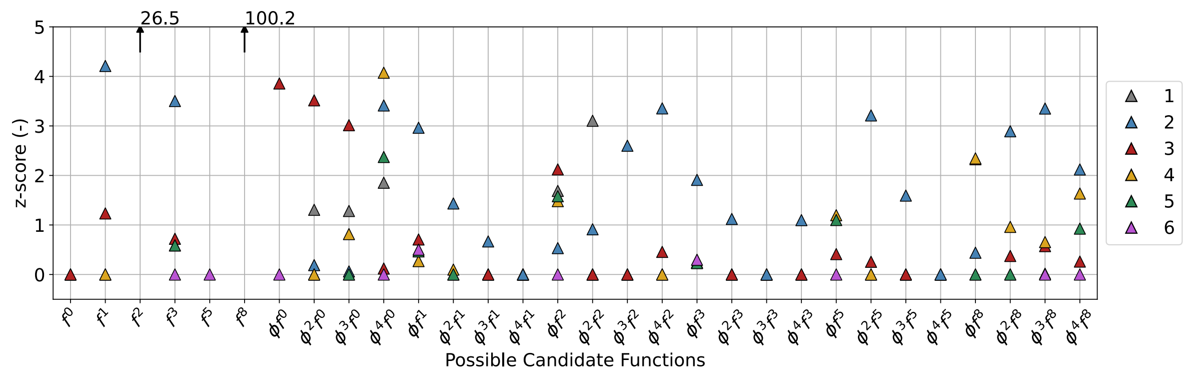


Figure 6.62: The z-score and amount of iterations for the exponential trained model $M_{2,U4,2}$, lacking the f^4 related terms in the library with additional f^8 terms with a z-score of 0 meaning that a function is eliminated in the process..

6.3 The Current Work and Turbulence Modelling

There are six different library candidate cases for three specific velocity test cases in the current report. From the six library cases, there are five main cases that are validated, the full PDE system, a PDE system where a small contribution of the solution is unknown in the candidate library (the first incomplete candidate library), a PDE system where a large contribution of the system is unknown in the candidate library (the second incomplete candidate library), a PDE system where a large contribution of the system is unknown in the candidate library with a larger candidate library with highly correlated functions (the third incomplete candidate library) and a case where the invariant/production term is incorrect but highly correlated with the true functional form (the fourth incomplete candidate library). For a full rundown of all the trained models, please look at Appendix H.

For the full candidate library case, the complete model was able to be returned. That happened only for the exponential velocity case at just one specific λ and tolerance combination in one specific search window (a λ value of 10^{-5} with a tolerance of 1). The other models found for the exponential case, all show good behaviour with the training data (See Figures 5.7, 5.8 and 5.9 from Chapter 5) and created a near perfect result when cross checked (Figures 6.12, 6.3, 6.7, 6.13 and 6.4, 6.8). For a complete candidate library, PDE FIND will find suitable results even if the true form is not detected by the algorithm.

For the case where a small contribution was lacking (The first incomplete library case from Section 3.3 from Chapter 3), being the quartic/$\phi^2 f^5$ term, satisfying models were found for both the trained velocity cases. The trained models performed well on their own data (Figures 5.17 and 5.18 from Chapter 5) and performed well for validating with the shear case (Figures 6.21 and 6.22). For all cases, but most importantly the shear case, the quartic/$\phi^2 f^5$ term has a small contribution to the overall result (Figures 4.21 and 4.22), and the replacement functions did not create a worse results. PDE FIND will find suitable models when a small term is lacking and will not create a worse performing model where the contribution has a small influence.

The case where a large contribution was lacking, being the cross diffusion/f^3 term, the two velocity trained models performed poorly (The second incomplete library case from Section 3.3 from Chapter 3) . The models didn't provide a good prediction compared to their own training cases (Figure 5.23 and 5.18 from Chapter 5) and the validation on the shear case (Figures 6.31 and for 6.24) was not satisfactory, not providing a reliable result. The values of ψ were better predicted than ϕ, what could have been due to a single term connecting ϕ to ψ in the ψ equation that is driven by the strain rate (Please see (3.1)), damping the affect of an incorrect ϕ prediction. PDE FIND is not comfortable with finding replacement models for an incorrect candidate library with a large and unique contribution lacking.

The addition of extra highly correlated functional forms to the missing f^3 term, being f^6 and f^7, in the incomplete library (The third incomplete library case from Section 3.3 from Chapter 3) provided an improvement. One model was found, being $M_{2,U3,3}$ that resembles the trained and two test cases decently. For these cases, it is interesting that the tested shear cases performed better than the trained a posteriori error. The results from model $M_{2,U3,3}$ are satisfactory and the model is sparse (except for the destruction related f^1 terms). There is a small offset in the solution for the validation with the vortex case, Figure 6.47, but the characteristic behaviour is similar and the solution is acceptable. The addition of extra highly correlated functions does help PDE FIND with finding acceptable models for candidate libraries that lack a significant contribution.

For the incomplete library case where all the strain rate driven (f^4) processes are replaced by a different highly correlated production terms (The fourth incomplete library case from Section 3.3 from Chapter 3) poor and unreliable results were found. In the training step, the a posteriori error and the time development (Figures 5.34 and 5.35 from Chapter 5) for the vortex trained models are poor while the exponential trained models are satisfactory. When the models are cross validated with the test shear case (Figures 6.55 and 6.56), both the vortex and exponential cases showed an unreliable and poor result, with no particular model able to partially represent the mechanics of the solution. Note that ψ, similar to previous cases, is predicted rather well, even if there is a poor prediction of ϕ ($M_{2,U4,1}$ and $M_{2,U4,2}$).

Through-out the report, different search windows were used by the author on a trial-by-error basis. Different search windows yield different results for the regression routine. An overview of the search windows used is given in Appendix C. The true identified model, $M_{2,F,2}$ was identified in a specific search window for only one specific case. Similar for model $M_{2,U3,3}$ where only one specific search window for one specific condition provided one good result. Without the different search window, the results described here would be different.

Through the whole experiment, the functions that were most of the time discovered by the regression algorithm, were f^2 and f^4. The f^4 function has a large contribution in the overall solution (Figure 4.21) while f^2 has a large amplitude in the solution (Figure 4.22). The regression algorithm was able to find these functions with coefficients similar to the true form. The z-scores (Figures 6.18, 6.19, 6.20, 6.27, 6.28, 6.37, 6.38, 6.51, 6.52, 6.61 and 6.62) of f^2 and f^4 were distinctive and PDE FIND was able to reliable find the functions with often a decent coefficient (often within a few percent with 20 % as the worst). The f^3 term was similar, but less consistent through-out the whole experiment and for the incomplete library case with a different production term, the f^3 term was poorly recognised.

A lot of functions that were highly correlated among one another, showed a redistribution of terms. Most often it was the case with functions consisting of the f^0 and f^1 terms (destruction terms), where the ground truth form was not recognised well, but a different highly correlated function was recognised. The result is a different, but good performing a solution. The distributions mostly occurred with the incomplete candidate libraries, where from the first iterations onwards a higher z-score, and therefore a higher likelihood, was present for the incorrect correlated function.

The use of sparse regression algorithms for transport equation modelling is two-fold. If a large part of the functions is known, or a large enough candidate library is used, a potential solution can be found. Solutions that are slightly non-sparse, such as model $M_{2,U3,3}$ can be valid approximations of the solution. However, using incorrect invariant/production terms or a too sparse of a candidate library can generate poor and unreliable results. With the sensitivity to the production term, special care should be made to have an extensive candidate library with viable production terms. The most important functional forms to the solution will always be found by sparse regression and for recognising the most important contributions to the solution, PDE FIND will provide insight.

The use of the material derivative rather than the time derivative in PDE FIND was satisfactory and no problems were found, similar with the use of rotational invariant functions for the library functions. On the contrary, the use of non-rotational invariant forms provided difficulty in finding result. For the diffusion term, not grouping the x and y components together provided cases where often one of the directional components was not detected. Grouping them together provided robust detection of the form.

7. Conclusions and Recommendations

Following the results and discussion from the Chapters 4, 5 and 6, an answer to the research question will be provided. The main conclusion, answering the research question is written in Section 7.1 while recommendations are given in Section 7.2.

7.1 Conclusion

The current work analysed PDE FIND by Rudy et al [40], for the (future) purpose of creating a non-local transport equation that is able to model the residual from k-frozen RANS to create a universal RANS model that is more accurate than the standard models. The research question reads: *How can sparse regression provide a trained model for non-local temporal behaviour in a model problem to represent the residual in k-frozen RANS?*

A model problem was created, dubbed the $\psi - \phi$ system, where the ψ function modelled the k-equation and ϕ modelled the Reynolds stress with the main assumption that most of the residual was contained within the Reynolds stress. On the model problem, six different model training experiments were performed, being training on a smaller partial $\psi-\phi$ system, training on the full $\psi-\phi$ system and four training cases with an incomplete candidate library with varying difficulty, aimed to mimic the uncertainties between RANS and high fidelity data. These models were tested among test cases that were different than the training case to show the universality of the models.

From the experiments, it can be found that the linear sparse regression routine can provide satisfactory models for a transport equation. If a large portion of the true functional form is not represented by the candidate functions, a large candidate library with highly correlated candidates to the missing form is able to train a well performing model. The exception is for the invariant/production terms, where a full understanding of those terms is beneficial for creating a correct transport equation.

Sparse regression is able to correctly identify the most important functions with an accurate enough coefficient, even if a trained model performs poorly. Functions having a significant contribution to the average solution or having a large amplitude in the solution are reliably found by PDE FIND with the z-score being a sufficient tool in analysing the importance of these terms.

For turbulence modelling, a highly correlated set of candidate functions, with a complete as possible invariant candidate collection could potentially result in a viable transport equation for the residual in k-frozen RANS, where PDE FIND should be able to find the most important terms from high fidelity turbulence data.

7.2 Recommendations

From the current work, multiple recommendations are provided. The recommendations are split in two specific categories, being an improvement in the testing conditions, Subsection 7.2.1 and a future recommendation regarding application of the work, Subsection 7.2.2.

7.2.1 Improvements for the Experiment

The current $\psi - \phi$ model has limitations. It uses a steady state velocity over the domain for the purpose of ease and understanding. An unsteady velocity connected to the ϕ or ψ equations could provide a better representation of the chaotic behaviour found in turbulence. For the equation, having ϕ as a tensor would be more analogous to the Reynolds stress it represents, or the analogy should be changed to better argue the use of a scalar variable.

The test cases are currently two dimensional with no noise. Turbulence is chaotic and three dimensional in nature and an expansion to a three dimensional test case is advised. Noise can be added to the problem, to reflect incoherencies in the averaging process used in data driven turbulence models.

The spatial discretisation is rather poor, with for ψ being about $O(10^{-1.2})$ and for ϕ about $O(10^{-1})$. A better discretisation method could provide a greater compliance to reality and could improve regression potentially. If improved, the spatial discretisation could also go down, improving computational efficiency and generating faster (a posteriori) results.

For the a priori-a posteriori diagrams, a specific a priori error was defined, where additional terms, not included in the true functional form, got a penalty of one. The penalty did not take into account the size of the coefficients, meaning that a relatively low a priori error could be found while the coefficients where excessive for the additional, not true functional form terms. An addition to the a priori error that penalises the size of the coefficients could work and one could take the norm of the coefficients for instance, although that might change the way a non-sparsity error is enforced.

The current a priori error also becomes large for the wrong coefficients of the true functional form and masks the penalty of including the wrong terms. A hard limit for such a value could generate more interesting a priori-a posteriori diagrams and change the result.

The sparse linear regression algorithm used in the current work, PDE FIND, is tuned only for λ and *tol* in the experiments. PDE FIND has more hyper parameters, such as μ and the amount of iterations for STRidge. Performing the same study with different values for μ could be interesting, in particular an increase in μ. It can allow for different results as it penalises non-sparse solutions more and provide different models, in particular for models that have a high correlation among functions.

In the current work, only one specific sparse regression method, being PDE FIND by Rudy et al [40], was tested. Different regression methods, such as SBL (Zhang et al [47], Nayek et al [90], Yuan et al [48], Zhang and Ling [49] and [50]) or weak formulation strategies (group of Grigoriev [51, 52, 53], Messenger and Bortz [54, 55]) and more, exist for creating sparse models from data. A comparison of methods for incomplete candidate libraries can be interesting, in particular with a focus on how a method converges to a result.

7.2.2 Future Developments and Strategies

The z-score has proven to be a reliable tool for finding the most important functional forms, even in trained models that perform poorly a posteriori. A good future work will be to develop a strategy in finding a new transport equations, with sparse linear regression as a tool to help find functional forms for the solution, rather than creating the solution. Such a strategy might allow for more control in the model form electing process, and create a better understanding in how the terms work with one another.

If a good strategy is defined for finding functional forms, real turbulence data such as the periodic hill can be used to train transport equation models. A more analogous system can also be considered, one where a more representative, multi-equation system is used. For using real turbulence data, an extensive study should be performed for all the terms, having an complete as possible invariant/strain-rate driven term library with a large set of candidate functions, that do not correlate with other candidates. The most sparse solution might also not always be the most accurate solution, even if it easier to understand, and with turbulence data a researcher should be open to vary in the level of sparsity of a transport function for the residual.

One of the main conclusions found by accident is the sensitivity of the search windows. Often enough, a specific solution was found for a specific search windows with a specific set of hyper parameters. The only satisfying model where a large part of the solution was lacking (incomplete candidate libraries 2 and 3), was model $M_{2,U3,3}$, found at one specific condition for one specific set of hyper parameters. The same is for model $M_{2,F,2}$ where the true functional form was obtained for one specific search window and one specific set of hyper parameters. A search window specific study might be interesting and for the purpose of using sparse regression for finding models, multiple search windows can definitely provide more satisfactory models.

A. Tensor bases

The current appendix provides the full mathematical bases related to the non-linear anisotropic generalisation of the eddy viscosity assumption, also known as the generalisation by Pope [17]. The generalisation states that the Reynolds stress can be build-up from a finite set tensor bases with invariants that can act as coefficients. The following ten tensor bases are found

$$T^1_{ij} = S_{ij} \tag{A.1}$$

$$T^2_{ij} = S_{ik}\Omega_{kj} - \Omega_{ik}S_{kj} \tag{A.2}$$

$$T^3_{ij} = S_{ik}S_{kj} - \frac{1}{3}\delta_{ij}S_{mn}S_{nm} \tag{A.3}$$

$$T^4_{ij} = \Omega_{ik}\Omega_{kj} - \frac{1}{3}\delta_{ij}\Omega_{mn}\Omega_{nm} \tag{A.4}$$

$$T^5_{ij} = \Omega_{ik}S_{kn}S_{nj} - S_{ik}S_{kn}\Omega_{nj} \tag{A.5}$$

$$T^6_{ij} = \Omega_{ik}\Omega_{kn}S_{nj} + S_{ik}\Omega_{kn}\Omega_{nj} - \frac{2}{3}\delta_{ij}S_{lk}\Omega_{kn}\Omega_{nl} \tag{A.6}$$

$$T^7_{ij} = \Omega_{ik}S_{kn}\Omega_{nm}\Omega_{mj} - \Omega_{ik}\Omega_{kn}S_{nm}\Omega_{mj} \tag{A.7}$$

$$T^8_{ij} = S_{ik}\Omega_{kn}S_{nm}S_{mj} - S_{ik}S_{kn}\Omega_{nm}S_{mj} \tag{A.8}$$

$$T^9_{ij} = \Omega_{ik}\Omega_{kn}S_{nm}S_{mj} + S_{ik}S_{kn}\Omega_{nm}\Omega_{mj} - \frac{2}{3}\delta_{ij}S_{lk}S_{kn}\Omega_{nm}\Omega_{ml} \tag{A.9}$$

$$T^{10}_{ij} = \Omega_{ik}S_{kn}S_{nm}\Omega_{ml}\Omega_{lj} - \Omega_{ik}\Omega_{kn}S_{nm}S_{ml}\Omega_{lj} \tag{A.10}$$

The invariants, that can form a part of the coefficients are

$$I_1 = S_{ik}S_{kj} \tag{A.11}$$

$$I_2 = \Omega_{ik}\Omega_{kj} \tag{A.12}$$

$$I_3 = S_{ik}S_{kn}S_{nj} \tag{A.13}$$

$$I_4 = \Omega_{ik}\Omega_{kn}S_{nj} \tag{A.14}$$

$$I_5 = \Omega_{ik}\Omega_{kn}S_{nm}S_{mj} \tag{A.15}$$

B. The Weak Form for the Simulations

The derivation of the weak form for FEnICsx of the $\psi - \phi$ of Equation (3.1), (3.2) and (3.4) are found in the current Appendix.

B.1 The Weak Form

FEniCSx is a finite element solver, and solves the PDE equation in terms of the weak (or variational) form. The weak form is obtained by multiplying the PDE with a function v and subsequently integrating it over a certain domain Ω. The result is a PDE of the form in Equation (B.1). The weak form states that a relation holds over a domain rather than at a point (as is the case for the hard form, the form used throughout the report) and an equation is sought that follows

$$L(f, v) = a(u, v) \tag{B.1}$$

where a is the bilinear form, L the linear form, u is the parameter of interest, f an arbitrary source term and v is the test function. The function v is able to vary independently.

In the case of the heat equation,

$$-\frac{\partial^2 u}{\partial x^2} = f \tag{B.2}$$

the resulting weak formulation starts with multiplication with the test function v and integrate it over Ω, resulting in

$$-\int_\Omega \frac{\partial^2 u}{\partial x^2} v \,\mathrm{d}x = \int_\Omega f v \,\mathrm{d}x \tag{B.3}$$

Using integration by parts, (B.3) becomes

$$\int_\Omega \frac{\partial u}{\partial x}\frac{\partial v}{\partial x} \,\mathrm{d}x - \int_{\partial\Omega} \frac{\partial u}{\partial n} v \,\mathrm{d}s = \int_\Omega f v \,\mathrm{d}x \tag{B.4}$$

At the boundaries, the flux of u is 0 (Neumann boundary condition) resulting in

$$\int_\Omega \frac{\partial u}{\partial x}\frac{\partial v}{\partial x} \,\mathrm{d}x = \int_\Omega f v \,\mathrm{d}x \tag{B.5}$$

where the left hand side of the equation is equal to $a(u, v)$ and the right hand side equal to $L(f, v)$.

In the given weak formulation, the Neumann boundary condition is stated to be natural. For a Neumann boundary condition in (B.7), the boundary integral of (B.4) becomes

$$\int_{\partial\Omega} \frac{\partial u}{\partial n} v \, \mathrm{d}s = gv \, \mathrm{d}s \tag{B.6}$$

A Dirichlet boundary condition (As in (B.8)) could be essential and has to be inferred on the solution. For a Dirichlet boundary condition, the boundary integral of (B.4) drops, as the trail function v will vanish to zero as the solution for u is known is on the boundary.

$$\frac{\partial u}{\partial x} = g \tag{B.7}$$

$$u(x) = u_d \tag{B.8}$$

For a more in-depth look into finite element methods and how it is used in FEniCSx, like for instance the Solobov space, one is recommend to look at the FEniCSx tutorial by Langtangen and Logg [104] or the finite element book of Brenner and Scott [105]

B.2 The Weak Formulation of ψ

For solving ψ in FEniCSx, the form in (B.1) is sought. The ψ equation is found in (3.1) and applying an forward-backward Euler scheme, the equation becomes

$$\frac{\psi^{n+1} - \psi^n}{\Delta t} + u_i \frac{\partial \psi^{n+1}}{\partial x_i} = c_1 \phi^n S_{ij} S_{ij} + c_2 \frac{\partial^2 \psi^{n+1}}{\partial x_i \partial x_i} - c_3 \left(\psi^n\right)^2 \tag{B.9}$$

Here, all non-linear terms take the value from the previous iteration (forward Euler) and all linear terms take the value from the to be predicted iteration (backward Euler) to conform to the bi-linear form. The value of ϕ is taken from the previous time step.

Equation (B.9) can be split in the future time step $n + 1$ as equation a and the previous known time step equation n as L. The future time step a grouped together for ψ gives

$$a(\psi^{n+1}) = \psi^{n+1} + \Delta t \left(u_i \frac{\partial \psi^{n+1}}{\partial x_i} - d_2 \frac{\partial^2 \psi^{n+1}}{\partial x_i \partial x_i} \right) \tag{B.10}$$

From (B.10), the weak form can be found by integrating the whole system and multiplying it by the test function q, resulting in

$$a(\psi^{n+1}, q) = \int_\Omega \left(\psi^{n+1} + \Delta t \left(u_i \frac{\partial \psi^{n+1}}{\partial x_i} - d_2 \frac{\partial^2 \psi^{n+1}}{\partial x_i \partial x_i} \right) \right) q \, \mathrm{d}x \tag{B.11}$$

From (B.11), the diffusion term can be converted with integration by parts. Using a zero Neumann boundary condition along the border, the final weak form used in FEniCSx for a becomes

$$a(\psi^{n+1}, q) = \int_\Omega \left(\psi^{n+1} + \Delta t \left(u_i \frac{\partial \psi^{n+1}}{\partial x_i} q - d_2 \frac{\partial \psi^{n+1}}{\partial x_i} \frac{\partial q}{\partial x_i} \right) \right) \mathrm{d}x \tag{B.12}$$

For the current time step equation L, the collection of terms are

$$L = \psi^n - \Delta t \left(d_1 \phi^n S_{ij} S_{ij} - d_3 \left(\psi^n\right)^2 \right) \tag{B.13}$$

Equation (B.13) is similar to the a equation and can be integrated and multiplied by the test function q to obtain the final L equation used for FEnICsx. The L equation in the weak form becomes

$$L(q) = \int_\Omega \left(\psi^n - \Delta t \left(d_1 \phi^n S_{ij} S_{ij} - d_3 \left(\psi^n\right)^2 \right) \right) q \, \mathrm{d}x \tag{B.14}$$

B.3 The Weak Formulation of ϕ

For the ϕ equation, a similar approach for obtaining a form in like (B.1) can be performed. The ϕ equation is found in (3.2) and applying a forward-backward Euler scheme, the equation becomes

$$\frac{\phi^{n+1} - \phi^n}{\Delta t} + u_i \frac{\partial \phi^{n+1}}{\partial x_i} = d_1 \phi^{n+1} S_{ij} S_{ij} + d_2 \frac{\partial^2 \phi^{n+1}}{\partial x_i \partial x_i} - d_3 \left(\phi^n\right)^2 + add \tag{B.15}$$

Here, all the non-linear terms take the value from the previous iteration (forward Euler) and all linear terms the to be predicted value (backward Euler) to conform to the bi-linear form. The value of ψ is taken from the previous time step. The additional terms associated with Equation (B.15) are

$$add = d_4 S_{ij} S_{ij} + d_5 \frac{\partial \phi^n \partial \psi^n}{\partial x_i \partial x_i} + d_6 \left(\phi^n\right)^2 S_{ij} S_{ij} - d_7 \psi^n \phi^n - d_8 \left(\phi^n\right)^2 \frac{\partial \phi^n}{\partial x_i} \frac{\partial \phi^n}{\partial x_i} \tag{B.16}$$

Equations (B.15) and (B.16) can be split in the future time step $n+1$ as equation a and the previous known time step equation n as L. The future time step a grouped together for ϕ gives

$$a(\phi^{‘n+1}) = \phi^{n+1} + \Delta t \left(u_i \frac{\partial \phi^{n+1}}{\partial x_i} - d_1 \phi^{n+1} S_{ij} S_{ij} - d_2 \frac{\partial^2 \phi^{n+1}}{\partial x_i \partial x_i} \right) \tag{B.17}$$

From (B.17), the weak form can be found by integrating the whole system and multiplying it by the test function q, resulting in

$$a(\phi^{‘n+1}, v) = \int_\Omega \left(\phi^{n+1} + \Delta t \left(u_i \frac{\partial \phi^{n+1}}{\partial x_i} - d_1 \phi^{n+1} S_{ij} S_{ij} - d_2 \frac{\partial^2 \phi^{n+1}}{\partial x_i \partial x_i} \right) \right) v \, \mathrm{d}x \tag{B.18}$$

With (B.18), the diffusion term can be converted using integration by parts. Using a zero Neumann boundary condition along the border, the final weak form used in FEniCSx for a becomes

$$a(\phi^{n+1}, v) = \int_\Omega \left(\phi^{n+1} + \Delta t \left(u_i \frac{\partial \phi^{n+1}}{\partial x_i} - d_1 \phi^{n+1} S_{ij} S_{ij} \right) v - \Delta t d_2 \frac{\partial \phi^{n+1}}{\partial x_i} \frac{\partial v}{\partial x_i} \right) \mathrm{d}x \quad \text{(B.19)}$$

For the current time step equation L, the collection of terms are

$$L = \phi^n - \Delta t \left(d_3 \left(\phi^n\right)^2 + d_4 S_{ij} S_{ij} + d_5 \frac{\partial \phi^n \partial \psi^n}{\partial x_i \partial x_i} + d_6 \left(\phi^n\right)^2 S_{ij} S_{ij} \right. \\ \left. - d_7 \psi^n \phi^n - d_8 \left(\phi^n\right)^2 \frac{\partial \phi^n}{\partial x_i} \frac{\partial \phi^n}{\partial x_i} \right) \quad \text{(B.20)}$$

Equation (B.11) is similar to the a equation and can be integrated and multiplied by the test function q to obtain the final L equation used for FEnICsx. The L equation in the weak form becomes

$$L(q) = \int_\Omega \left(\phi^n - \Delta t \left(d_3 \left(\phi^n\right)^2 + d_4 S_{ij} S_{ij} + d_5 \frac{\partial \phi^n \partial \psi^n}{\partial x_i \partial x_i} + d_6 \left(\phi^n\right)^2 S_{ij} S_{ij} \right. \right. \\ \left. \left. - d_7 \psi^n \phi^n - d_8 \left(\phi^n\right)^2 \frac{\partial \phi^n}{\partial x_i} \frac{\partial \phi^n}{\partial x_i} \right) \right) v \, \mathrm{d}x \quad \text{(B.21)}$$

C. The Sampling Windows

The current appendix provides the reader with the sampling space for each tested velocity case.

C.1 Sampling Windows for the Partial PDE System

Vortex Velocity Case

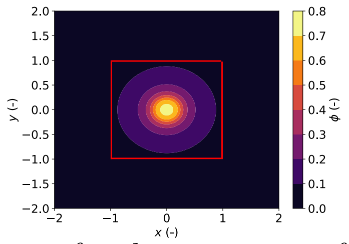


Figure C.1: The sampling area for the vortex case for the partial PDE system.

Exponential Velocity Case

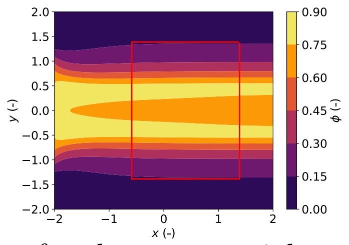


Figure C.2: The sampling area for the exponential case for the partial PDE system.

Shear Velocity Case

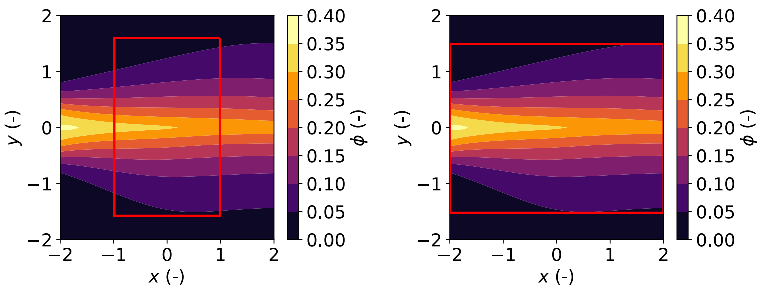


Figure C.3: The sampling area for the shear case for the partial PDE system, with on the left the sampling windows denoted with blue marks and on the right the sampling window denoted with red marks in the a priori-a posteriori diagrams.

C.2 Sampling Windows for the Full PDE System

Vortex Velocity Case

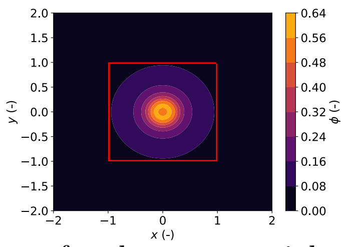


Figure C.4: The sampling area for the exponential case for the full PDE system.

Exponential Velocity Case

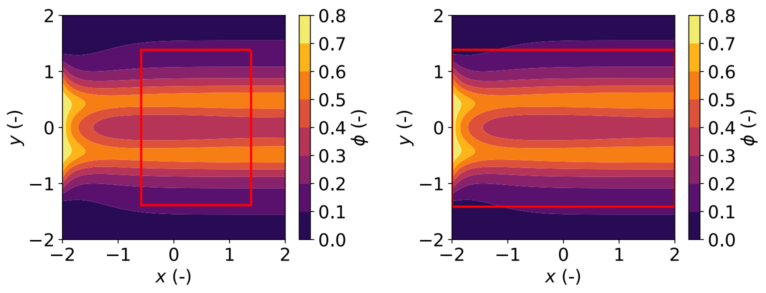


Figure C.5: The sampling area for the exponential case for the full PDE system, with on the left the sampling windows denoted with blue marks and on the right the sampling window denoted with red marks in the a priori-a posteriori diagrams.

Shear Velocity Case

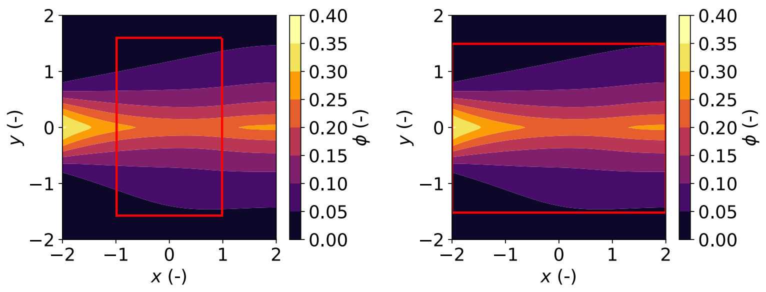


Figure C.6: The sampling area for the shear case for the full PDE system, with on the left the sampling windows denoted with blue marks and on the right the sampling window denoted with red marks in the a priori-a posteriori diagrams.

D. The Candidate Dictionary

Table D.1 show the full candidate libraries associated with the report for all experiments

Table D.1: All candidates for each experiment used in the current report.

Number	**Partial**	**Full**	$\boldsymbol{U_1}$	$\boldsymbol{U_2}$	$\boldsymbol{U_3}$	$\boldsymbol{U_4}$
1	f^0	f^0	f^0	f^0	f^0	f^0
2	f^1	f^1	f^1	f^1	f^1	f^1
3	f^2	f^2	f^2	f^2	f^2	f^2
4	f^3	f^3	f^3	f^4	f^4	f^3
5	f^4	f^4	f^4	f^5	f^5	f^8
6	f^5	f^5	ϕf^0	ϕf^0	f^6	f^5
7	ϕf^0	ϕf^0	$\phi^2 f^0$	$\phi^2 f^0$	f^7	ϕf^0
8	ϕf^1	ϕf^1	$\phi^3 f^0$	$\phi^3 f^0$	ϕf^0	$\phi^2 f^0$
9	ϕf^2	ϕf^2	$\phi^4 f^0$	$\phi^4 f^0$	$\phi^2 f^0$	$\phi^3 f^0$
10	ϕf^3	ϕf^3	ϕf^1	ϕf^1	$\phi^3 f^0$	$\phi^4 f^0$
11	ϕf^4	ϕf^4	$\phi^2 f^1$	$\phi^2 f^1$	$\phi^4 f^0$	ϕf^1
12	ϕf^5	ϕf^5	$\phi^3 f^1$	$\phi^3 f^1$	ϕf^1	$\phi^2 f^1$
13	$\phi^2 f^0$	$\phi^2 f^0$	$\phi^4 f^1$	$\phi^4 f^1$	$\phi^2 f^1$	$\phi^3 f^1$
14	$\phi^2 f^1$	$\phi^2 f^1$	ϕf^2	ϕf^2	$\phi^3 f^1$	$\phi^4 f^1$
15	$\phi^2 f^2$	$\phi^2 f^2$	$\phi^2 f^2$	$\phi^2 f^2$	$\phi^4 f^1$	ϕf^2
16	$\phi^2 f^3$	$\phi^2 f^3$	$\phi^3 f^2$	$\phi^3 f^2$	ϕf^2	$\phi^2 f^2$
17	$\phi^2 f^4$	$\phi^2 f^4$	$\phi^4 f^2$	$\phi^4 f^2$	$\phi^2 f^2$	$\phi^3 f^2$
18	$\phi^2 f^5$	$\phi^2 f^5$	ϕf^3	ϕf^4	$\phi^3 f^2$	$\phi^4 f^2$
19	$\phi^3 f^0$	$\phi^3 f^0$	$\phi^2 f^3$	$\phi^2 f^4$	$\phi^4 f^2$	ϕf^3
20	$\phi^3 f^1$	$\phi^3 f^1$	$\phi^3 f^3$	$\phi^3 f^4$	ϕf^4	$\phi^2 f^3$
21	$\phi^3 f^2$	$\phi^3 f^2$	$\phi^4 f^3$	$\phi^4 f^4$	$\phi^2 f^4$	$\phi^3 f^3$
22	$\phi^3 f^3$	$\phi^3 f^3$	ϕf^4	ϕf^5	$\phi^3 f^4$	$\phi^4 f^3$
23	$\phi^3 f^4$	$\phi^3 f^4$	$\phi^2 f^4$	$\phi^2 f^5$	$\phi^4 f^4$	ϕf^8
24	$\phi^3 f^5$	$\phi^3 f^5$	$\phi^3 f^4$	$\phi^3 f^5$	ϕf^5	$\phi^2 f^8$
25	$\phi^4 f^0$	$\phi^4 f^0$	$\phi^4 f^4$	$\phi^4 f^5$	$\phi^2 f^5$	$\phi^3 f^0$
26	$\phi^4 f^1$	$\phi^4 f^1$			$\phi^3 f^5$	$\phi^4 f^8$
27	$\phi^4 f^2$	$\phi^4 f^2$			$\phi^4 f^5$	ϕf^5
28	$\phi^4 f^3$	$\phi^4 f^3$			ϕf^6	$\phi^2 f^5$
29	$\phi^4 f^4$	$\phi^4 f^4$			$\phi^2 f^6$	$\phi^3 f^5$
30	$\phi^4 f^5$	$\phi^4 f^5$			$\phi^3 f^6$	$\phi^4 f^5$
31					$\phi^4 f^6$	
32					ϕf^7	
33					$\phi^2 f^7$	
34					$\phi^3 f^7$	
35					$\phi^4 f^7$	

E. Test Cases and Hyper parameters

The PDE FIND algorithm by Rudy et al [40] uses hyper parameters for the regression to obtain results. Table (E.1) show the hyper parameters for all test cases in the current report

Table E.1: Table showing all PDE FIND simulations that are ran through the report

Candidate case		**Tolerance** (-)	λ (-)
Partial PDE	Vortex	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Shear	1, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Shear different window	0.25, 0.5, 1.5, 3, 5	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
Full PDE	Vortex	1, 2, 3, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential different window	1, 3, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Shear	0.5, 1.5, 3, 5, 7.5	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Shear different window	0.5, 1, 3.5, 5	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
Incomplete library one	Vortex	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential different window	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
Incomplete library two	Vortex	2.5, 3.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential different window	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
Incomplete library three	Vortex	1, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential	1.5, 2.5, 5, 7.5, 10	10^{-2}, 10^{-3}, 10^{-4}
	Exponential different window	1, 1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
Incomplete library four	Vortex	2.5, 5, 7.5, 10, 12.5, 15	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}
	Exponential different window	1.5, 2.5, 5, 7.5, 10	10^{-2} ,10^{-3}, 10^{-4}, 10^{-5}, 10^{-6}

F. Statistics of the Candidates

The current Appendix shows the statistics between all the functions giving multiple figures, with statistics.

The variance of all the data for each case is given in Figure F.1 and is given as the trace of the covariance matrix,

$$\mathrm{COV}(x_i, x_k) = \sigma_y^2 (X_{ij} X_{kj})^{-1} \tag{F.1}$$

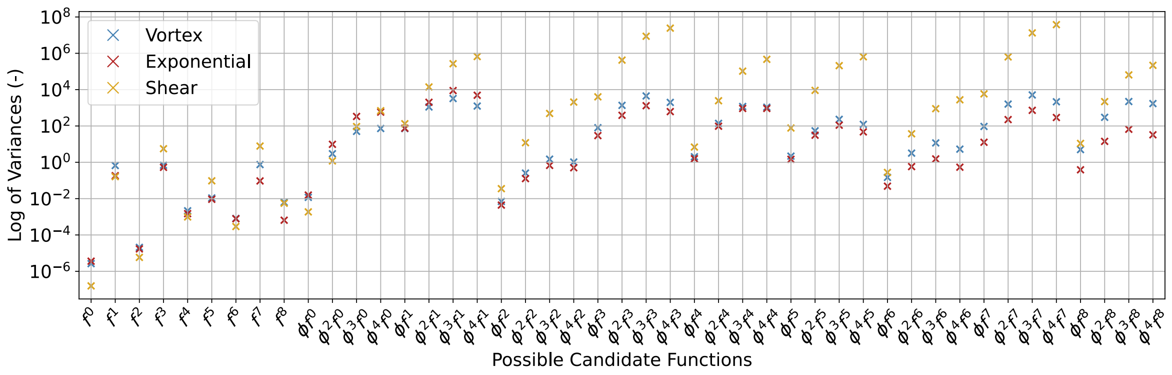


Figure F.1: Variance of all the possible candidate functions for the full PDE case for the STRridge regression algorithm

The Pearson correlation coefficient shows correlation between data, where one is an excellent positive correlation and negative one is an excellent negative correlation. The equation for the Pearson correlation coefficient is

$$\rho_{cor} = \frac{\mathrm{E}[f^x f^y] - \mathrm{E}[f^x] E[f^y]}{\sqrt{\mathrm{VAR}[f^x] \mathrm{VAR}[f^y]}} \tag{F.2}$$

Heatmaps are shown in Figures F.2, F.3 and F.4 for the vortex, exponential case and shear cases.

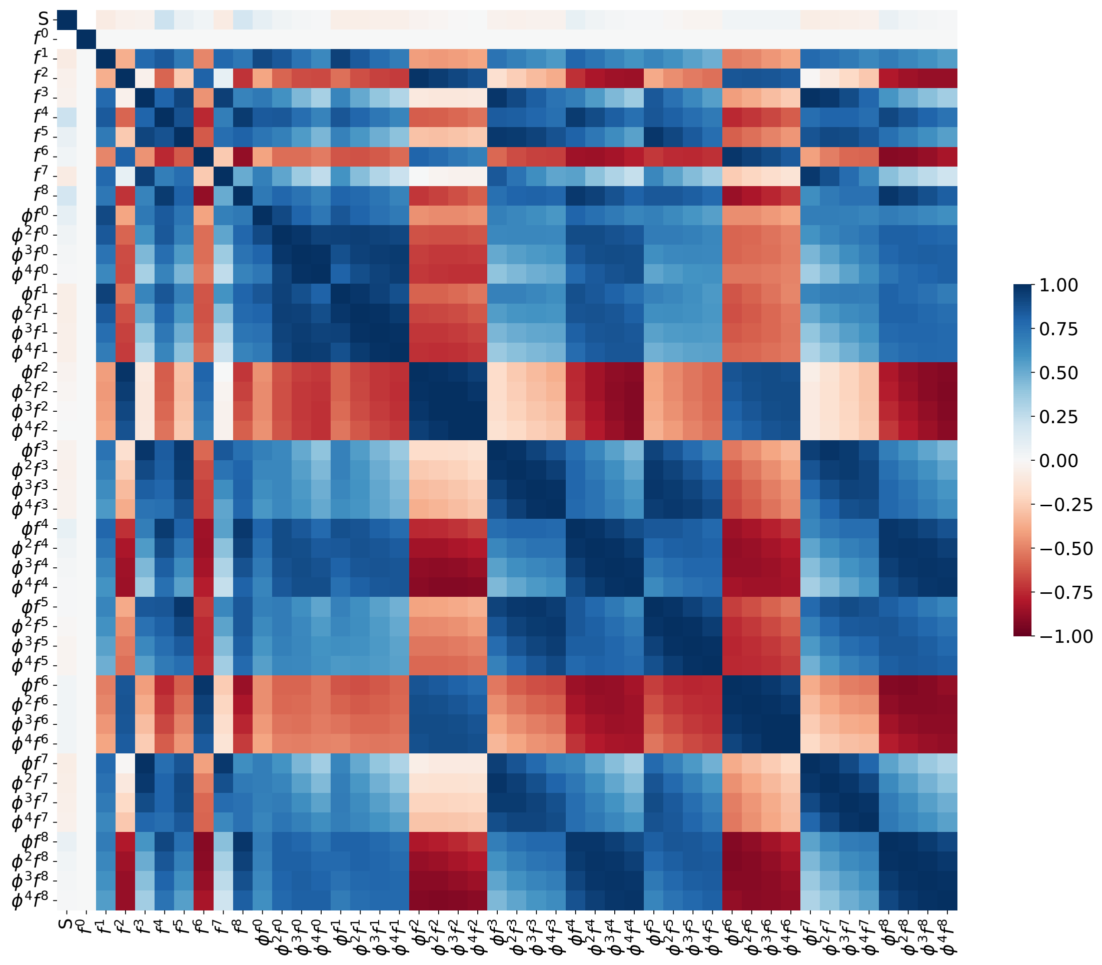


Figure F.2: Heatmap of the pearson correlation coefficient between functions related to the vortex velocity case for the full case where S is the solution.

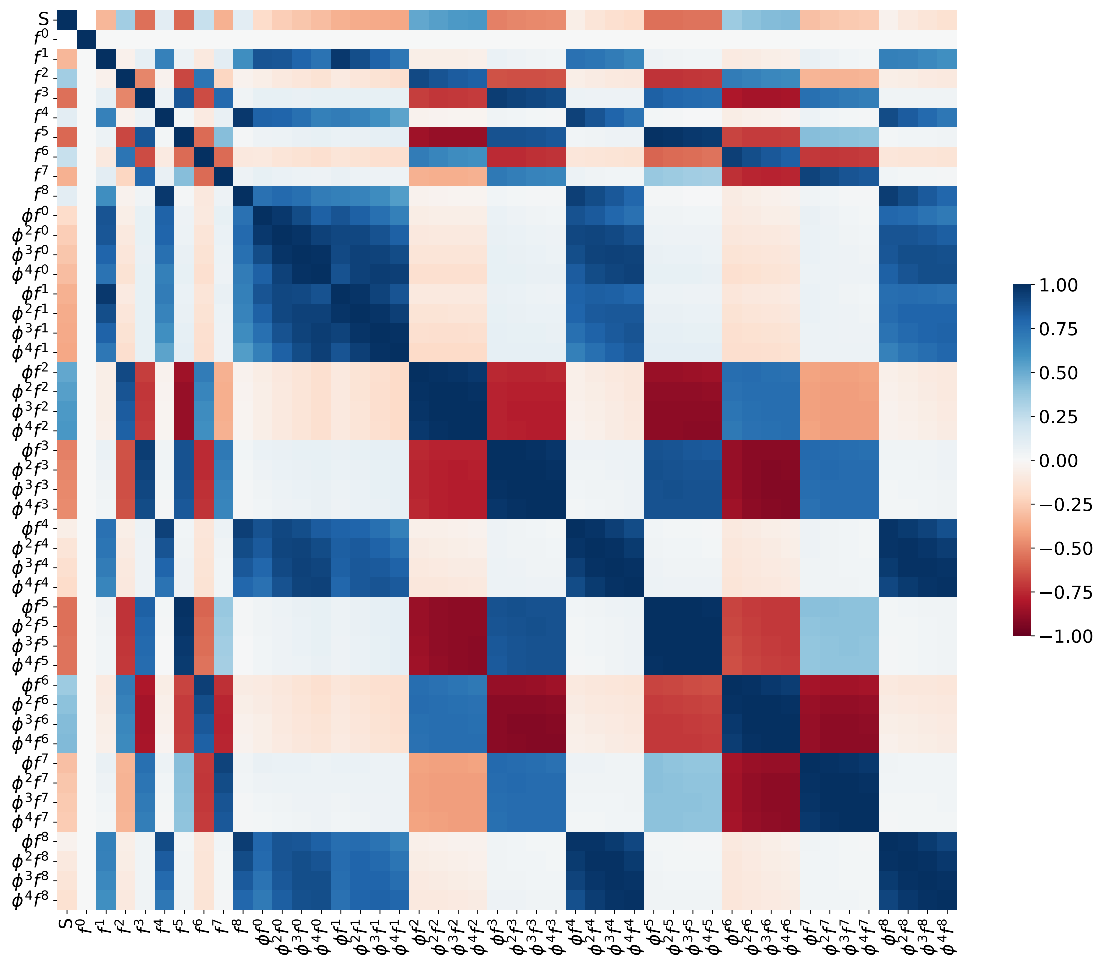


Figure F.3: Heatmap of the pearson correlation coefficient between functions related to the exponential velocity case for the full case where S is the solution.

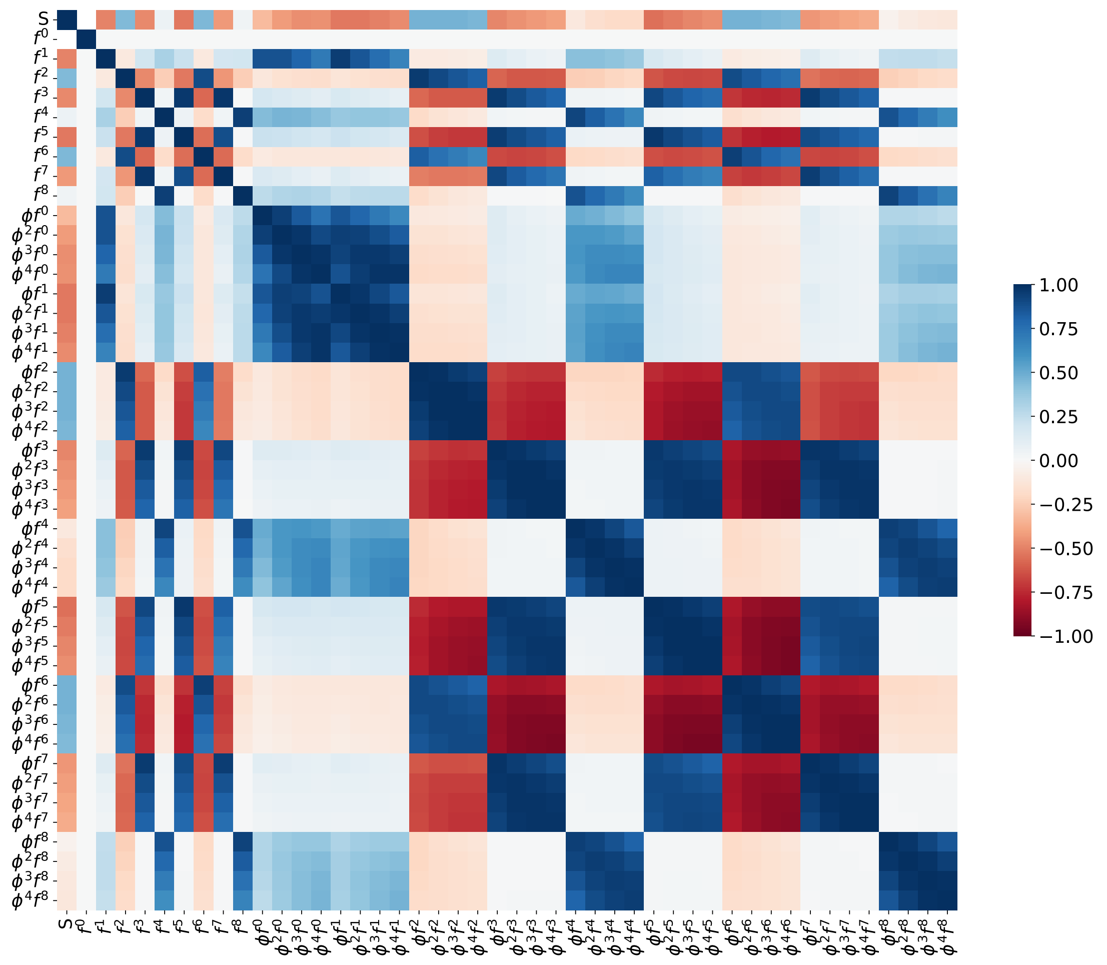


Figure F.4: Heatmap of the pearson correlation coefficient between functions related to the shear velocity case for the full case where S is the solution.

G. Results of the Training Simulations

The current appendix shows the training result for each specific candidate library and velocity case. Places denoted with - show setting that have not been elected to run, N/D is for not determined, and is related to PDE FIND not being able to run a case. It can be due to threshold ridge regression not being able to eliminate terms or threshold ridge regression eliminating all of the terms. NAN is for not a number and is stated when a run has diverged. NAN is only related to the a posteriori error.

Vortex Velocity with Partial PDE

Table G.1: PDE FIND results for the vortex velocity case with the partial PDE with a full candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	6.94	0.01	0.01	11.52	0.01	0.01	3.52	0.009	0.01	3.61	0.007	0.009	3.25	0.02	0.02
2.5	6.94	0.01	0.01	3.25	0.02	0.02	3.52	0.009	0.01	0.033	0.003	0.004	3.25	0.02	0.02
5	5.58	0.73	0.60	3.25	0.02	0.02	3.25	0.02	0.02	0.033	0.003	0.004	3.25	0.02	0.02
7.5	5.92	0.87	0.81	3.25	0.02	0.02	3.25	0.02	0.02	0.033	0.003	0.004	3.25	0.02	0.02
10	5.92	0.87	0.81	5.58	0.87	0.81	5.58	0.73	0.60	5.58	0.73	0.60	3.25	0.02	0.02

Exponential Velocity with Partial PDE

Table G.2: PDE FIND results for the exponential velocity case with the partial PDE with a full candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	6.31	0.01	0.002	7.54	0.001	0.001	0.058	0.003	0.001	0.058	0.003	0.001	0.058	0.003	0.001
2.5	7.00	0.01	0.002	2.71	0.006	0.001	0.058	0.003	0.001	0.058	0.003	0.001	2.69	0.001	0.009
5	4.40	0.03	0.007	3.30	0.07	0.03	0.058	0.003	0.001	0.058	0.003	0.001	2.69	0.001	0.009
7.5	5.74	0.53	0.24	5.74	0.53	0.24	0.058	0.003	0.001	0.058	0.003	0.001	2.69	0.001	0.009
10	5.74	0.53	0.24	5.74	0.63	0.24	N/D	N/D	N/D	N/D	N/D	N/D	2.69	0.001	0.009

Shear Velocity with Partial PDE

Table G.3: PDE FIND results for the shear velocity case with the partial PDE with a full candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1	7.98	0.07	0.009	4.71	0.13	0.01	3.59	0.04	0.004	1.14	0.001	0.0002	N/D	N/D	N/D
2.5	5.88	0.52	0.05	3.59	0.04	0.003	3.59	0.04	0.004	3.87	0.006	0.0005	4.71	0.13	0.01
5	5.88	0.52	0.05	5.88	0.52	0.05	5.88	0.52	0.05	N/D	N/D	N/D	4.71	0.13	0.01
7.5	5.88	0.52	0.05	5.88	0.52	0.05	5.98	NAN	NAN	N/D	N/D	N/D	4.71	0.13	0.01
10	5.88	0.52	0.05	5.88	0.52	0.05	5.98	NAN	NAN	N/D	N/D	N/D	4.71	0.13	0.01
0.25 DSW	3.69	0.04	0.003	3.69	0.04	0.003	3.62	0.004	0.009	1.16	0.0006	0.0001	80.84	0.11	0.01
0.5 DSW	3.69	0.04	0.003	4.68	0.12	0.01	3.69	0.04	0.003	1.16	0.0006	0.0001	11.60	0.01	0.001
1.5 DSW	3.69	0.04	0.003	5.81	0.48	0.03	3.69	0.04	0.003	3.69	0.04	0.003	11.10	0.007	0.001
3 DSW	3.69	0.04	0.003	5.81	0.48	0.03	3.69	0.04	0.003	3.69	0.04	0.003	5.12	0.02	0.002
5 DSW	5.81	0.48	0.03	5.81	0.48	0.03	5.81	0.47	0.03	5.81	0.48	0.03	5.12	0.02	0.002

Vortex Velocity with Full PDE

Table G.4: PDE FIND results for the vortex velocity case with the full PDE with a full candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1	8.78	0.02	0.02	7.41	0.009	0.009	4.59	0.009	0.009	17.3	0.03	0.02	N/D	N/D	N/D
2	8.78	0.02	0.02	4.85	0.01	0.01	7.80	0.009	0.009	8.14	0.05	0.03	N/D	N/D	N/D
3	5.40	0.11	0.09	4.85	0.01	0.01	5.45	0.009	0.01	8.14	0.05	0.03	11.0	0.03	0.01
5	7.93	0.89	0.83	4.85	0.01	0.01	5.40	0.11	0.09	1.73	0.01	0.01	4.51	0.03	0.02
7.5	7.93	0.89	0.83	4.51	0.03	0.02	5.40	0.11	0.09	2.47	0.01	0.01	11.0	0.03	0.01
10	7.93	0.89	0.83	4.51	0.03	0.02	5.40	0.11	0.09	1.73	0.01	0.01	4.51	0.03	0.02

Exponential Velocity with Full PDE

Table G.5: PDE FIND results for the exponential velocity case with the full PDE with a full candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	5.43	0.02	0.005	11.72	0.01	0.005	9.27	0.003	0.001	10.2	0.003	0.001	8.65	0.01	0.004
2.5	4.37	0.02	0.005	4.37	0.02	0.005	4.71	0.007	0.002	4.37	0.02	0.005	16.1	0.003	0.002
5	4.37	0.02	0.005	4.37	0.02	0.005	4.71	0.007	0.002	8.05	0.11	0.03	5.20	0.007	0.002
7.5	8.05	0.11	0.03	4.37	0.02	0.005	4.71	0.007	0.002	8.05	0.11	0.03	5.20	0.007	0.002
10	4.37	0.02	0.005	4.37	0.02	0.005	8.05	0.11	0.03	8.05	0.11	0.03	5.20	0.007	0.002
1 DSW	6.08	0.01	0.002	5.59	0.006	0.002	6.75	0.003	0.001	0.15	0.003	0.001	16.6	0.006	0.002
3 DSW	7.95	0.12	0.04	7.95	0.12	0.04	4.74	0.005	0.001	4.36	0.02	0.005	4.36	0.02	0.005
5 DSW	7.95	0.12	0.04	7.95	0.12	0.04	5.79	0.005	0.001	6.44	0.25	0.07	6.44	0.25	0.07
7.5 DSW	7.95	0.12	0.04	7.95	0.12	0.04	7.95	0.12	0.04	6.44	0.25	0.07	7.81	0.56	0.26
10 DSW	7.95	0.12	0.04	7.95	0.12	0.04	7.95	0.12	0.04	6.44	0.25	0.07	7.81	0.56	0.26

Shear Velocity with Full PDE

Table G.6: PDE FIND results for the shear velocity case with the full PDE with a full candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
0.5	7.34	0.85	0.08	6.91	0.09	0.01	6.51	0.08	0.008	10.25	0.006	0.0004	291.54	0.13	0.02
1.5	7.90	0.52	0.05	7.34	0.85	0.08	6.51	0.08	0.008	10.25	0.006	0.0004	78.97	0.03	0.003
3	7.90	0.52	0.05	7.90	0.52	0.05	6.51	0.08	0.008	7.90	0.52	0.05	83.40	0.03	0.003
5	7.90	0.52	0.05	7.90	0.52	0.05	7.90	0.52	0.05	7.90	0.52	0.05	11.47	0.009	0.002
7.5	7.90	0.52	0.05	7.90	0.52	0.05	N/D	N/D	N/D	7.90	0.52	0.05	6.05	0.03	0.001
0.5 DSW	7.16	0.02	0.004	2.32	0.002	0.0002	6.73	0.09	0.008	2.32	0.002	0.0002	138.40	0.02	0.003
1.5 DSW	7.16	0.02	0.004	6.73	0.09	0.008	7.83	0.48	0.04	2.32	0.002	0.0002	49.21	NAN	NAN
3.5 DSW	7.83	0.48	0.04	7.83	0.48	0.04	7.83	0.48	0.04	7.83	0.48	0.04	15.33	0.02	0.002
5 DSW	7.83	0.46	0.03	7.83	0.48	0.04	7.83	0.48	0.04	7.83	0.48	0.04	65.92	0.03	0.002

Vortex Velocity with First Incomplete Library

Table G.7: PDE FIND results for the vortex velocity case with the full PDE with the first incomplete candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	7.79	0.02	0.02	8.74	0.007	0.009	2.66	0.01	0.01	7.21	0.007	0.01	18.16	0.02	0.02
2.5	4.40	0.11	0.09	3.51	0.03	0.02	2.66	0.01	0.01	7.21	0.007	0.01	3.51	0.03	0.02
5	6.93	0.89	0.83	3.51	0.03	0.02	2.66	0.01	0.01	4.40	0.11	0.09	8.64	0.02	0.02
7.5	6.93	0.89	0.83	3.51	0.03	0.02	6.59	0.75	0.63	6.59	0.75	0.63	8.64	0.02	0.02
10	6.93	0.89	0.83	3.51	0.03	0.02	2.66	0.01	0.01	6.93	0.89	0.83	11.18	0.007	0.009

Exponential Velocity with First Incomplete Library

Table G.8: PDE FIND results for the exponential velocity case with the full PDE with the first incomplete candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	4.43	0.02	0.005	10.95	0.009	0.005	3.37	0.02	0.005	N/D	N/D	N/D	N/D	N/D	N/D
2.5	3.37	0.02	0.005	7.05	0.11	0.03	3.37	0.02	0.005	3.37	0.02	0.005	12.01	0.01	0.005
5	3.37	0.02	0.005	7.05	0.11	0.03	7.05	0.11	0.03	5.06	0.10	0.01	4.74	0.007	0.002
7.5	7.05	0.11	0.03	7.05	0.11	0.03	7.05	0.11	0.03	7.05	0.11	0.03	4.74	0.007	0.002
10	3.37	0.02	0.03	7.05	0.11	0.03	7.05	0.11	0.03	7.05	0.11	0.03	4.74	0.007	0.002
1.5 DSW	5.21	0.03	0.005	3.36	0.02	0.005	8.75	0.001	0.001	1.34	0.008	0.002	1.34	0.008	0.001
2.5 DSW	6.95	0.12	0.04	6.95	0.12	0.04	3.74	0.001	0.001	3.36	0.02	0.005	3.36	0.02	0.005
5 DSW	6.95	0.12	0.04	6.95	0.12	0.04	3.74	0.001	0.001	5.44	0.25	0.07	5.44	0.25	0.07
7.5 DSW	6.95	0.12	0.04	6.95	0.12	0.04	6.95	0.04	0.04	5.44	0.25	0.07	5.44	0.25	0.07
10 DSW	6.95	0.12	0.04	6.95	0.12	0.04	6.95	0.04	0.04	5.44	0.25	0.07	5.44	0.25	0.07

Vortex Velocity with Second Incomplete Library

Table G.9: PDE FIND results for the vortex velocity case with the full PDE with the second incomplete candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
2.5	5.80	0.62	0.47	13.95	NAN	NAN	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D
3.5	5.80	0.62	0.47	11.10	0.15	0.15	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D
5	6.93	0.89	0.83	13.95	NAN	NAN	38.42	0.34	0.31	46.20	0.28	0.27	99.93	0.09	0.08
7.5	5.80	0.63	0.47	9.78	NAN	NAN	11.51	1.00	1.00	9.52	1.00	1.00	99.93	0.09	0.08
10	6.93	0.89	0.83	6.93	0.89	0.83	8.0	1.00	1.00	9.51	1.00	1.00	99.93	0.09	0.08

Exponential Velocity with Second Incomplete Library

Table G.10: PDE FIND results for the exponential velocity case with the full PDE with the second incomplete candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	7.28	0.16	0.05	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	-	-	-
2.5	7.28	0.16	0.05	7.28	0.16	0.05	18.29	NAN	NAN	N/D	N/D	N/D	-	-	-
5	7.28	0.16	0.05	7.45	0.34	0.12	18.29	NAN	NAN	66.17	NAN	NAN	-	-	-
7.5	7.28	0.16	0.05	6.81	0.57	0.26	9.24	NAN	NAN	66.17	NAN	NAN	-	-	-
10	7.40	0.10	0.02	6.81	0.57	0.26	9.24	NAN	NAN	66.17	NAN	NAN	-	-	-
1.5 DSW	7.31	0.17	0.05	13.84	NAN	NAN	16.59	NAN	NAN	N/D	N/D	N/D	N/D	N/D	N/D
2.5 DSW	7.41	0.32	0.11	7.31	0.17	0.05	12.17	NAN	NAN	17.43	NAN	NAN	N/D	N/D	N/D
5 DSW	6.81	0.56	0.26	6.81	0.56	0.26	6.95	0.12	0.04	12.22	NAN	NAN	115.18	NAN	NAN
7.5 DSW	6.81	0.56	0.26	6.81	0.56	0.26	6.81	0.56	0.26	6.81	0.56	0.26	115.18	NAN	NAN
10 DSW	6.81	0.56	0.26	6.81	0.56	0.26	6.81	0.56	0.26	6.81	0.56	0.26	115.18	NAN	NAN

Vortex Velocity with Third Incomplete Library

Table G.11: PDE FIND results for the vortex velocity case with the full PDE with the third incomplete candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1	8.93	0.28	0.18	9.44	0.66	0.36	26.45	NAN	NAN	63.38	0.10	0.08
2.5	6.58	0.50	0.31	8.98	0.19	0.11	26.45	NAN	NAN	16.14	NAN	NAN
5	6.58	0.50	0.31	8.98	0.19	0.11	26.45	NAN	NAN	16.14	NAN	NAN
7.5	6.58	0.50	0.31	6.86	0.86	0.81	13.09	0.20	0.15	16.14	NAN	NAN
10	5.85	0.52	0.40	8.98	0.19	0.11	7.61	0.23	0.16	16.14	NAN	NAN

Exponential Velocity with Third Incomplete Library

Table G.12: PDE FIND results for the exponential velocity case with the full PDE with the third incomplete candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	5.43	0.09	0.03	N/D	N/D	N/D	N/D	N/D	N/D	-	-	-	-	-	-
2.5	5.06	0.10	0.01	N/D	N/D	N/D	N/D	N/D	N/D	-	-	-	-	-	-
5	5.06	0.10	0.01	9.78	0.08	0.02	5.50	0.11	0.03	-	-	-	-	-	-
7.5	5.06	0.10	0.01	7.05	0.11	0.03	5.50	0.11	0.03	-	-	-	-	-	-
10	6.81	0.57	0.26	7.05	0.11	0.03	5.50	0.11	0.03	-	-	-	-	-	-
1 DSW	8.35	0.13	0.05	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D
1.5 DSW	7.29	0.11	0.02	83.12	NAN	NAN	29.43	0.09	0.03	N/D	N/D	N/D	N/D	N/D	N/D
2.5 DSW	6.81	0.56	0.26	6.62	0.09	0.03	29.43	0.09	0.03	37.00	0.07	0.03	56.72	NAN	NAN
5 DSW	6.81	0.56	0.26	8.36	0.13	0.05	29.43	0.09	0.03	37.00	0.07	0.03	56.72	NAN	NAN
7.5 DSW	6.81	0.56	0.26	6.95	0.12	0.04	9.31	0.16	0.05	15.79	NAN	NAN	62.46	NAN	NAN
10 DSW	6.81	0.56	0.26	6.95	0.12	0.04	5.44	0.25	0.07	10.88	0.15	0.05	56.72	NAN	NAN

Vortex Velocity with Fourth Incomplete Library

Table G.13: PDE FIND results for the vortex velocity case with the full PDE with the fourth incomplete candidate library

λ (-)	10^{-1}			10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
2.5	6.0	0.94	0.86	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D
5	7.0	0.88	0.73	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D
7.5	7.0	0.88	0.73	6.0	0.94	0.86	11.71	NAN	NAN	16.93	0.56	0.38	21.29	0.53	0.35	194.56	1.0	1.0
10	6.0	0.94	0.86	8.19	0.59	0.38	7.16	0.59	0.39	23.29	0.48	0.30	21.29	0.53	0.35	194.56	1.0	1.0
12.5	6.0	0.94	0.86	6.0	0.94	0.86	7.16	0.59	0.39	16.93	0.56	0.38	21.29	0.53	0.35	194.56	1.0	1.0
15	6.0	0.94	0.86	6.0	0.94	0.86	6.0	0.94	0.86	6.0	0.94	0.86	10.66	0.32	0.20	194.56	1.0	1.0

Exponential Velocity with Fourth Incomplete Library

Table G.14: PDE FIND results for the exponential velocity case with the full PDE with the fourth incomplete candidate library

λ (-)	10^{-2}			10^{-3}			10^{-4}			10^{-5}			10^{-6}		
error / *tol* (-)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)	ξ_{prio} (-)	$\xi_{post,\phi}$ (%)	$\xi_{post,\psi}$ (%)
1.5	7.57	0.09	0.03	N/D	N/D	N/D	N/D	N/D	N/D	-	-	-	-	-	-
2.5	7.57	0.10	0.01	N/D	N/D	N/D	N/D	N/D	N/D	-	-	-	-	-	-
5	6.0	0.10	0.01	N/D	N/D	N/D	N/D	N/D	0N/D	-	-	-	-	-	-
7.5	6.0	0.10	0.01	6.0	0.11	0.03	7.0	0.11	0.03	-	-	-	-	-	-
10	6.0	0.57	0.26	7.56	0.11	0.03	6.0	0.11	0.03	-	-	-	-	-	-
1.5 DSW	9.03	NAN	NAN	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D	N/D
2.5 DSW	4.16	NAN	NAN	16.46	NAN	NAN	30.46	NAN	NAN	9.51	0.04	0.01	N/D	N/D	N/D
5 DSW	6.0	0.61	0.27	6.09	NAN	NAN	24.04	NAN	NAN	8.23	NAN	NAN	N/D	N/D	N/D
7.5 DSW	6.0	0.61	0.27	5.10	0.05	0.01	14.09	NAN	NAN	8.23	NAN	NAN	56.72	NAN	NAN
10 DSW	6.0	0.61	0.27	7.0	0.38	0.11	24.04	NAN	NAN	5.12	0.41	0.13	56.72	NAN	NAN

H. Model Dictionary

The current appendix is a summation of all models used through out the report. the models are subjected as $M_{x,yy,z}$ what means the following:

- **x** The model type, with 1 being the vortex case, 2 the exponential case and 3 the shear case.
- **yy** The candidate function conditions, with P being the partial set PDE, F the full PDE and $U1$, $U2$, $U3$ and $U4$ being the four test with an incomplete candidate function test.
- **z** The model number, usually two or three different models are chosen per test case and candidate function setting.

H.1 The Original Models

H.1.1 The Partial PDE Model

$$\frac{D\phi}{Dt} = -0.75\phi^2 f^0 + 0.1f^2 + (1.5 + \phi)f^4 - 0.35\phi^2 f^5 \tag{H.1}$$

H.1.2 The Full PDE Model

$$\frac{D\phi}{Dt} = -0.75\phi^2 f^0 - 3\phi f^1 + 0.1f^2 - f^3 + (1.5 + \phi + 0.5\phi^2)f^4 - 0.35\phi^2 f^5 \tag{H.2}$$

H.2 The Partial PDE Linear Regression Trained Models

$\boldsymbol{M_{1,P,1}}$

$$\frac{D\phi}{Dt} = -0.75\phi^2 f^0 + 0.1f^2 - 1.0f^3 + (1.5 + 1.01\phi)f^4 - 0.35\phi^2 f^5 \tag{H.3}$$

$\boldsymbol{M_{2,P,1}}$

$$\frac{D\phi}{Dt} = -0.76\phi^2 f^0 + 0.1f^2 - 1.0f^3 + (1.5 + 1.03\phi)f^4 - 0.35\phi^2 f^5 \tag{H.4}$$

$\boldsymbol{M_{3,P,1}}$

$$\frac{D\phi}{Dt} = -0.79\phi^2 f^0 + 0.1f^2 - 1.01f^3 + (1.50 + 1.08\phi)f^4 \tag{H.5}$$

H.3 The Full Case Linear Regression Trained Models

$\boldsymbol{M_{1,F,1}}$

$$\begin{aligned}\frac{D\phi}{Dt} &= -0.85\phi^2 f^0 + (-0.003 - 3.26\phi + 0.69\phi^2)f^1 + (0.10 - 0.0001\phi)f^2 \\ &+ (-0.93 - 0.28\phi - 0.11\phi^3)f^3 + (1.47 + 1.09\phi + 0.47\phi^2)f^4 - 0.57\phi^4 f^5\end{aligned} \tag{H.6}$$

$\boldsymbol{M_{1,F,2}}$

$$\frac{D\phi}{Dt} = -0.55\phi^2 f^0 - 3.12\phi f^1 + (0.10)f^2 - 1.01f^3 + (1.46 + 1.30\phi)f^4 - 0.32\phi^2 f^5 \tag{H.7}$$

$\boldsymbol{M_{1,F,3}}$

$$\frac{D\phi}{Dt} = -0.85\phi^2 f^0 + (-0.23 - 2.37\phi)f^1 + 0.10f^2 - 0.83f^3 + (1.46 + 1.30\phi)f^4 - 0.61\phi f^5 \tag{H.8}$$

$\boldsymbol{M_{2,F,1}}$

$$\begin{aligned}\frac{D\phi}{Dt} &= -0.74\phi^2 f^0 + (0.03 + 3.02\phi + 10.22\phi^2 - 22.01\phi^3 + 15.7\phi^4)f^1 + 0.1f^2 \\ &- 0.88f^3 + (1.50 + 0.95\phi + 0.48\phi^2)f^4 - 0.43\phi f^5\end{aligned} \tag{H.9}$$

$\boldsymbol{M_{2,F,2}}$

$$\frac{D\phi}{Dt} = -0.74\phi^2 f^0 - 3.02\phi f^1 + 0.1f^2 - f^3 + (1.49 + 1.05\phi + 0.47\phi^2)f^4 - 0.39\phi^2 f^5 \tag{H.10}$$

$\boldsymbol{M_{2,F,3}}$

$$\begin{aligned}\frac{D\phi}{Dt} &= -0.74\phi^2 f^0 + (-0.29 + 0.27\phi - 12.40\phi^2 + 19.33\phi^3 - 10.80\phi^4)f^1 + 0.1f^2 \\ &+ (-0.94 - 0.38\phi + 0.90\phi^3)f^3 + (1.50 + 0.95\phi + 0.62\phi^2)f^4 - 1.17\phi^4 f^5\end{aligned} \tag{H.11}$$

$\boldsymbol{M_{3,F,1}}$

$$\frac{D\phi}{Dt} = -0.79\phi^2 f^0 - 2.99\phi f^1 + 0.1f^2 - 1f^3 + (1.48 + 1.24\phi)f^4 \tag{H.12}$$

$\boldsymbol{M_{3,F,2}}$

$$\frac{D\phi}{Dt} = -0.80\phi^2 f^0 - 2.96\phi f^1 + 0.1f^2 - 1.02f^3 + (1.46 + 1.88\phi - 4.11\phi^2 + 7.74\phi^3)f^4 \tag{H.13}$$

$\boldsymbol{M_{3,F,3}}$

$$\frac{D\phi}{Dt} = -5.01\phi f^1 + 0.07f^2 - 1f^3 + 1.32f^4 \tag{H.14}$$

H.4 The First Incomplete Library case Linear Regression Trained Models

$\boldsymbol{M_{1,U1,1}}$

$$\frac{D\phi}{Dt} = -0.86\phi^2 f^0 + (-0.06 - 2.83\phi)f^1 + (0.10 + 0.03\phi^3)f^2 + (-0.91 - 0.33\phi + 0.65\phi^3 - 2.75\phi^4)f^3 + (1.51 + 0.66\phi + 1.55\phi^2 - 0.40\phi^3)f^4 \tag{H.15}$$

$\boldsymbol{M_{1,U1,2}}$

$$\frac{D\phi}{Dt} = (-0.83\phi^2)f^0 + (-2.82\phi)f^1 + 0.10f^2 + (-0.89 - 0.49\phi)f^3 + (1.45 + 1.34\phi)f^4 \tag{H.16}$$

$\boldsymbol{M_{1,U1,3}}$

$$\frac{D\phi}{Dt} = -0.81\phi^2 f^0 + -2.95\phi f^1 + (0.11 - 0.09\phi + 0.34\phi^2 - 0.59\phi^3 + 0.42\phi^4)f^2 - (1.0 + 2.54\phi^4)f^3 + (1.50 + 0.98\phi + 1.26\phi^3)f^4 \tag{H.17}$$

$\boldsymbol{M_{2,U1,1}}$

$$\frac{D\phi}{Dt} = -0.75\phi^2 f^0 + (-0.38 - 1.91\phi - 19.34\phi^2 + 30.24\phi^3 - 16.01\phi^4 f^1 + 0.10f^2 + (-0.72 - 3.37\phi + 13.23\phi^2 - 22.02\phi^3 + 12.01\phi^4)f^3 + (0.93\phi + 0.51\phi^2)f^4 \tag{H.18}$$

$\boldsymbol{M_{2,U1,2}}$

$$\frac{D\phi}{Dt} = -0.62\phi^2 f^0 - 2.94\phi f^1 + 0.09f^2 - 1.06f^3 + (1.50 + 1.02\phi)f^4 \tag{H.19}$$

$\boldsymbol{M_{2,U1,3}}$

$$\frac{D\phi}{Dt} = -0.73\phi^2 f^0(-1.63\phi - 4.09\phi^2 + 5.74\phi^4)f^1 + 0.10f^2 - 1.07f^3 + (1.49 + 1.15\phi)f^4 \tag{H.20}$$

H.5 The Second Incomplete Library case Linear Regression Trained Models

$\boldsymbol{M_{1,U2,1}}$

$$\frac{D\phi}{Dt} = (-11.79\phi + 25.88\phi^3)f^1 + (0.10 - 0.22\phi + 0.38\phi^3)f^2 + 1.30f^4 - 0.68\phi)f^5 \tag{H.21}$$

$\boldsymbol{M_{1,U2,2}}$

$$\begin{aligned}\frac{D\phi}{Dt} &= (-11.07\phi^2 + 31.82\phi^3 - 13.96\phi^4)f^0 \\ &+ (3.45 - 25.44\phi - 71.76\phi^2 + 385.65\phi^3 - 372.73\phi^4)f^1 \\ &+ (0.10 - 0.57\phi - 5.02\phi^2 + 10.94\phi^3 - 7.47\phi^4)f^2 \\ &+ (1.44 + 2.57\phi - 6.70\phi^2)f^4 - (2.42\phi - 18.42\phi^2 + 32.34\phi^3 - 15.88\phi^4)f^5\end{aligned} \tag{H.22}$$

$\boldsymbol{M_{2,U2,1}}$

$$\frac{D\phi}{Dt} = -0.86f^0 + (1.73 - 2.25\phi)f^4 \tag{H.23}$$

$\boldsymbol{M_{2,U2,2}}$

$$\frac{D\phi}{Dt} = -1.51f^1 + 0.08f^2 + 1.60f^4 - 1.13\phi f^5 \tag{H.24}$$

$\boldsymbol{M_{2,U2,3}}$

$$\frac{D\phi}{Dt} = -1.56f^0 + (0.08 - 1.12\phi)f^2 + 1.60f^4 \tag{H.25}$$

H.6 The Third Incomplete Library case Linear Regression Trained Models

$\boldsymbol{M_{1,U3,1}}$

$$\frac{D\phi}{Dt} = -3.61\phi f^1 + 0.09f^2 + (1.5 + 1.83\phi^2)f^4 - 0.75\phi f^5(-2.86\phi + 2.89\phi^2)f^7 \tag{H.26}$$

$\boldsymbol{M_{1,U3,2}}$

$$\frac{D\phi}{Dt} = (-6.08\phi + 8.14\phi^3)f^1 + 0.09f^2 + (1.29 + 1.34\phi)f^4 - 0.62\phi f^5 - 1.77\phi f^7 \tag{H.27}$$

$\boldsymbol{M_{1,U3,3}}$

$$\begin{gathered}\frac{D\phi}{Dt} = -12.53\phi^4 f^0 + (-0.35\phi - 91.43\phi^2 + 273.13\phi^3 - 203.67\phi^4)f^1 \\ + (0.11 + 0.30\phi^3 - 0.65\phi^4)f^2 + (1.45 + 1.93\phi - 3.24\phi^2 + 8.16\phi^3)f^4 \\ + (-1.04\phi + 1.29\phi^3)f^5 + (-0.03 - 0.96\phi^2 + 3.41\phi^3 - 2.25\phi^4)f^6 \\ + (-0.13 + 2.18\phi - 15.39\phi^2 + 15.96\phi^3)f^7\end{gathered} \tag{H.28}$$

$\boldsymbol{M_{2,U3,1}}$

$$\begin{aligned}\frac{D\phi}{Dt} = -0.80f^0 - 2.84\phi^2 f^1 + 0.08f^2 + (1.54 + 0.40\phi)f^4 - 0.76\phi f^5 \\ +(-2.28\phi + 6.68\phi^4)f^7\end{aligned} \tag{H.29}$$

$\boldsymbol{M_{2,U3,2}}$

$$\frac{D\phi}{Dt} = -2.16\phi f^1 + (1.60 - 1.72\phi)f^4 \tag{H.30}$$

$\boldsymbol{M_{2,U3,3}}$

$$\begin{aligned}\frac{D\phi}{Dt} = -0.57\phi^2 f^0 + (-0.25 - 2.55\phi + 0.12\phi^2)f^1 + 0.1f^2 + \\ (1.47 + 1.20\phi)f^4 - 0.85\phi f^5 - 0.72f^7\end{aligned} \tag{H.31}$$

H.7 The Fourth Incomplete Library case Linear Regression Trained Models

$\boldsymbol{M_{1,U4,1}}$

$$\frac{D\phi}{Dt} = (0.22 - 0.50\phi + 0.54\phi^3)f^2 - 0.56f^3 + (0.94 + 1.66\phi - 8.18\phi^2 + 10.21\phi^4)f^8 \quad \text{(H.32)}$$

$\boldsymbol{M_{1,U4,2}}$

$$\frac{D\phi}{Dt} = 0.08f^2 - 2.81\phi f^3 0.66f^8 + 0.35f^5 \quad \text{(H.33)}$$

$\boldsymbol{M_{2,U4,1}}$

$$\frac{D\phi}{Dt} = (-2.42\phi^2 + 5.73\phi^3)f^0 - 3.16\phi f^1 + 0.12f^2 - 3.80\phi^2 f^3(2.52 + 1.79\phi - 5.81\phi^3)f^8 \quad \text{(H.34)}$$

$\boldsymbol{M_{2,U4,2}}$

$$\frac{D\phi}{Dt} = 0.11f^2 - 2.90\phi f^1 - 1.54\phi f^3 + 2.65f^8 \quad \text{(H.35)}$$

$\boldsymbol{M_{2,U4,3}}$

$$\frac{D\phi}{Dt} = (0.64\phi - 2.16\phi^3)f^0 - 1.36f^1 + 0.09f^2 - 0.58f^3 + 2.23f^8 \quad \text{(H.36)}$$

Bibliography

[1] J. Boussinesq, "Essai sur la théorie des eaux courantes," in *Mémoires présentés par divers savants à l'Académie des sciences de l'Institut national de France*, Imprimerie Nationale, 1877.

[2] P. R. Spalart, "Philosophies and fallacies in turbulence modeling," *Progress in Aerospace Sciences*, vol. 74, pp. 1–15, 2015.

[3] D. C. Wilcox, *Turbulence Modeling for CFD.* No. 3rd Edition in Turbulence Modeling for CFD, DCW Industries, Inc., La Canada, 2006.

[4] B. Tracey, K. Duraisamy, and J. Alonso, "Application of supervised learning to quantify uncertainties in turbulence and combustion modeling," in *51st AIAA Aerospace Sciences Meeting including the New Horizons Forum and Aerospace Exposition 2013*, 2013.

[5] K. Duraisamy, Z. J. Zhang, and A. P. Singh, "New approaches in turbulence and transition modeling using data-driven techniques," in *53rd AIAA Aerospace Sciences Meeting*, 2015.

[6] E. J. Parish and K. Duraisamy, "A paradigm for data-driven predictive modeling using field inversion and machine learning," *Journal of Computational Physics*, vol. 305, pp. 758–774, 2016.

[7] K. Duraisamy, A. P. Singh, and S. Pan, "Augmentation of turbulence models using field inversion and machine learning," in *55th AIAA Aerospace Sciences Meeting*, 2017.

[8] P. Spalart and S. Allmaras, "A one-equation turbulence model for aerodynamic flows," *AIAA*, vol. 439, 1992.

[9] A. Ferrero, A. Iollo, and F. Larocca, "Field inversion for data-augmented rans modelling in turbomachinery flows," *Computers & Fluids*, vol. 201, 2020.

[10] L. Franceschini, D. Sipp, and O. Marquet, "Mean-flow data assimilation based on minimal correction of turbulence models: Application to turbulent high Reynolds number backward-facing step," *Physical Review Fluids*, vol. 5, no. 9, 2020.

[11] A. S. Cato, P. S. Volpiani, V. Mons, O. Marquet, and D. Sipp, "Comparison of different data-assimilation approaches to augment rans turbulence models," *Computers & Fluids*, vol. 266, 2023.

[12] P. S. Volpiani, M. Meyer, L. Franceschini, J. Dandois, F. Renac, E. Martin, O. Marquet, and D. Sipp, "Machine learning-augmented turbulence modeling for RANS

simulations of massively separated flows," *Physical Review Fluids*, vol. 6, no. 6, pp. 1–24, 2021.

[13] P. S. Volpiani, R. F. Bernardini, and L. Franceschini, "Neural network-based eddy-viscosity correction for rans simulations of flows over bi-dimensional bumps," *International Journal of Heat and Fluid Flow*, vol. 97, 2022.

[14] P. S. Volpiani, "Are random forests better suited than neural networks to augment rans turbulence models?," *International Journal of Heat and Fluid Flow*, vol. 107, 2024.

[15] A. Crivellini, V. D'Alessandro, and F. Bassi, "A spalart–allmaras turbulence model implementation in a discontinuous galerkin solver for incompressible flows," *Journal of Computational Physics*, vol. 241, pp. 388–415, 2013.

[16] J. Ling, A. Kurzawski, and J. Templeton, "Reynolds averaged turbulence modelling using deep neural networks with embedded invariance," *Journal of Fluid Mechanics*, vol. 807, pp. 155–166, 2016.

[17] S. B. Pope, "A more general effective-viscosity hypothesis," *Journal of Fluid Mechanics*, vol. 72, no. 2, pp. 331–340, 1975.

[18] J. Cai, P.-E. Angeli, J.-M. Martinez, G. Damblin, and D. Lucor, "Revisiting tensor basis neural network for reynolds stress modeling: Application to plane channel and square duct flows," *Computers & Fluids*, vol. 275, 2024.

[19] R. Fang, D. Sondak, P. Protopapas, and S. Succi, "Deep learning for turbulent channel flow," 2018.

[20] M. L. Kaandorp and R. P. Dwight, "Data-driven modelling of the reynolds stress tensor using random forests with invariance," *Computers & Fluids*, vol. 202, 2020.

[21] J.-L. Wu, J.-X. Wang, and H. Xiao, "A bayesian calibration–prediction method for reducing model-form uncertainties with application in rans simulations," *Flow, Turbulence and Combustion*, vol. 97, no. 3, p. 761–786, 2016.

[22] J.-X. Wang, J.-L. Wu, and H. Xiao, "Physics-informed machine learning approach for reconstructing reynolds stress modeling discrepancies based on dns data," *Physical Review Fluids*, vol. 2, no. 3, 2017.

[23] J.-L. Wu, H. Xiao, and E. Paterson, "Physics-informed machine learning approach for augmenting turbulence models: A comprehensive framework," *Physical Review Fluids*, vol. 3, no. 7, 2018.

[24] F. D. S. Banerjee, R. Krahl and C. Zenger, "Presentation of anisotropy properties of turbulence, invariants versus eigenvalue approaches," *Journal of Turbulence*, vol. 8, 2007.

[25] C. Jiang, J. Mi, S. Laima, and H. Li, "A novel algebraic stress model with machine-learning-assisted parameterization," *Energies*, vol. 13, no. 1, 2020.

[26] C. Jiang, R. Vinuesa, R. Chen, J. Mi, S. Laima, and H. Li, "An interpretable framework of data-driven turbulence modeling using deep neural networks," *Physics of Fluids*, vol. 33, no. 5, 2021.

[27] D. S. Haitz Sáez de Ocáriz Borde and P. Protopapas, “Convolutional neural network models and interpretability for the anisotropic reynolds stress tensor in turbulent one-dimensional flows,” *Journal of Turbulence*, vol. 23, no. 1-2, pp. 1–28, 2022.

[28] C. Ferreira, “Gene expression programming: A new adaptive algorithm for solving problems,” *Complex Systems*, vol. 13, 2001.

[29] J. Weatheritt and R. Sandberg, “A novel evolutionary algorithm applied to algebraic modifications of the rans stress–strain relationship,” *Journal of Computational Physics*, vol. 325, pp. 22–37, 2016.

[30] J. Weatheritt and R. Sandberg, “The development of algebraic stress models using a novel evolutionary algorithm,” *International Journal of Heat and Fluid Flow*, vol. 68, pp. 298–318, 2017.

[31] Y. Zhao, H. D. Akolekar, J. Weatheritt, V. Michelassi, and R. D. Sandberg, “Rans turbulence model development using cfd-driven machine learning,” *Journal of Computational Physics*, vol. 411, 2020.

[32] A. Bleh and G. Geiser, “Finding transition models using dimensional analysis gene expression programming,” *AIAA SCITECH 2024 Forum*, 2024.

[33] W. Ma, J. Zhang, K. Feng, H. Xing, and D. Wen, “Dimensional homogeneity constrained gene expression programming for discovering governing equations,” *Journal of Fluid Mechanics*, vol. 985, 2024.

[34] M. Schmelzer, R. P. Dwight, and P. Cinnella, “Discovery of Algebraic Reynolds-Stress Models Using Sparse Symbolic Regression,” *Flow, Turbulence and Combustion*, vol. 104, no. 2-3, pp. 579–603, 2020.

[35] J. Steiner, R. Dwight, and A. Viré, “Data-driven turbulence modeling for wind turbine wakes under neutral conditions,” *Journal of Physics: Conference Series*, vol. 1618, no. 6, 2020.

[36] J. Steiner, R. P. Dwight, and A. Viré, “Data-driven rans closures for wind turbine wakes under neutral conditions,” *Computers & Fluids*, vol. 233, 2022.

[37] J. Steiner, A. Vire, and R. Dwight, “Classifying regions of high model error within a data-driven rans closure: Application to wind turbine wakes,” *Flow, Turbulence and Combustion*, vol. 109, pp. 1–26, 2022.

[38] Y. Stöcker, C. Golla, R. Jain, J. Fröhlich, and P. Cinnella, “Dns-based turbulent closures for sediment transport using symbolic regression,” *Flow, Turbulence and Combustion*, vol. 112, 2023.

[39] S. Cherroud, X. Merle, P. Cinnella, and X. Gloerfelt, “Sparse bayesian learning of explicit algebraic reynolds-stress models for turbulent separated flows,” *International Journal of Heat and Fluid Flow*, vol. 98, 2022.

[40] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, “Data-driven discovery of partial differential equations,” *Science Advances*, vol. 3, no. 4, pp. 1–7, 2017.

[41] S. H. Rudy, S. L. Brunton, J. L. Proctor, and J. N. Kutz, “Supplement: Data-driven discovery of partial differential equations,” *Science Advances*, vol. 3, no. 4, 2017.

[42] S. L. Brunton, J. L. Proctor, and J. N. Kutz, "Discovering governing equations from data by sparse identification of nonlinear dynamical systems," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 113, no. 15, pp. 3932–3937, 2016.

[43] I. Abramovic, E. P. Alves, and M. Greenwald, "Data-driven model discovery for plasma turbulence modelling," *Journal of Plasma Physics*, vol. 88, no. 6, pp. 1–18, 2022.

[44] B. Zhao, M. He, and J. Wang, "Data-driven discovery of the governing equation of granular flow in the homogeneous cooling state using sparse regression," *Physics of Fluids*, vol. 35, no. 1, 2023.

[45] S. Rudy, A. Alla, S. L. Brunton, and J. N. Kutz, "Data-driven identification of parametric partial differential equations," *SIAM Journal on Applied Dynamical Systems*, vol. 18, no. 2, pp. 643–660, 2019.

[46] H. Schaeffer, "Learning partial differential equations via data discovery and sparse optimization," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 473, no. 2197, 2017.

[47] S. Zhang and G. Lin, "Robust data-driven discovery of governing physical laws with error bars," *Proceedings of the Royal Society A: Mathematical, Physical and Engineering Sciences*, vol. 474, no. 2217, 2018.

[48] Y. Yuan, X. Li, L. Li, F. J. Jiang, X. Tang, F. Zhang, J. Goncalves, H. U. Voss, H. Ding, and J. Kurths, "Machine discovery of partial differential equations from spatiotemporal data: A sparse Bayesian learning framework," *Chaos: An Interdisciplinary Journal of Nonlinear Science*, vol. 33, no. 11, 2023.

[49] S. Zhang and G. Lin, "Subtsbr to tackle high noise and outliers for data-driven discovery of differential equations," *Journal of Computational Physics*, vol. 428, 2021.

[50] A. Chen, Y. Du, L. M. Gao, and G. Lin, "Bayesian data-driven discovery of partial differential equations with variable coefficients," 2024.

[51] D. R. Gurevich, P. A. Reinbold, and R. O. Grigoriev, "Robust and optimal sparse regression for nonlinear PDE models," *Chaos*, vol. 29, no. 10, 2019.

[52] P. A. K. Reinbold, D. R. Gurevich, and R. O. Grigoriev, "Using noisy or incomplete data to discover models of spatiotemporal dynamics," *Phys. Rev. E*, vol. 101, 2020.

[53] P. A. K. Reinbold, L. M. Kageorge, M. F. Schatz, and R. O. Grigoriev, "Robust learning from noisy, incomplete, high-dimensional experimental data via physically constrained symbolic regression," *Nature Communications*, vol. 12, no. 1, 2021.

[54] D. A. Messenger and D. M. Bortz, "Weak sindy: Galerkin-based data-driven model selection," *Multiscale Modeling & Simulation*, vol. 19, no. 3, p. 1474–1497, 2021.

[55] D. A. Messenger and D. M. Bortz, "Weak sindy for partial differential equations," *Journal of Computational Physics*, vol. 443, 2021.

[56] A. C. Bekar and E. Madenci, "Peridynamics enabled learning partial differential equations," *Journal of Computational Physics*, vol. 434, 2021.

[57] E. Madenci, A. Barut, and M. Dorduncu, *Peridynamic Differential Operator for Numerical Analysis*. Springer Cham, 2019.

[58] H. Xu and D. Zhang, "Robust discovery of partial differential equations in complex situations," *Physical Review Research*, vol. 3, 2021.

[59] Z. Zhang and Y. Liu, "Parsimony-Enhanced Sparse Bayesian Learning for Robust Discovery of Partial Differential Equations," *Mechanical Systems and Signal Processing*, vol. 171, no. December 2021, 2022.

[60] N. Joemon, M. Pradeep, L. K. Rajulapati, and R. Rengaswamy, "Discovering governing partial differential equations from noisy data," *Computers and Chemical Engineering*, vol. 180, no. May 2023, 2024.

[61] Z. Long, Y. Lu, X. Ma, and B. Dong, "PDE-net: Learning PDEs from data," in *Proceedings of the 35th International Conference on Machine Learning*, vol. 80, pp. 3208–3216, Proceedings of Machine Learning Research, 2018.

[62] M. Raissi, P. Perdikaris, and G. Karniadakis, "Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations," *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[63] H. Vaddireddy, A. Rasheed, A. E. Staples, and O. San, "Feature engineering and symbolic regression methods for detecting hidden physics from sparse sensor observation data," *Physics of Fluids*, vol. 32, no. 1, 2020.

[64] H. Xing, J. Zhang, W. Ma, and D. Wen, "Using gene expression programming to discover macroscopic governing equations hidden in the data of molecular simulations," *Physics of Fluids*, vol. 34, no. 5, 2022.

[65] S. Pope, *Turbulent Flows*. Cambridge University Press, 2000.

[66] A. Kolmogorov, "The Local Structure of Turbulence in Incompressible Viscous Fluid for Very Large Reynolds' Numbers," *Akademiia Nauk SSSR Doklady*, vol. 30, pp. 301–305, 1941.

[67] H. Xiao, J.-L. Wu, S. Laizet, and L. Duan, "Flows over periodic hills of parameterized geometries: A dataset for data-driven turbulence modeling from direct simulations," *Computers & Fluids*, vol. 200, 2020.

[68] M. Marquille, U. Ehrenstein, and J.-P. Laval, "Instability of streaks in wall turbulence with adverse pressure gradient," *Journal of Fluid Mechanics*, vol. 681, p. 205–240, 2011.

[69] P. Balakumar and G. I. Park, "Dns/les simulations of separated flows at high reynolds numbers," in *45th AIAA Fluid Dynamics Conference*, American Institute for Aeronautics and Astronautics (AIAA), 2015.

[70] M. Waldrop, "The chips are down for moore's law," *Nature News*, vol. 530, p. 144, 2016.

[71] T. N. Theis and H.-S. P. Wong, "The end of moore's law: A new beginning for information technology," *Computing in Science & Engineering*, vol. 19, no. 2, pp. 41–50, 2017.

[72] X. Gloerfelt and P. Cinnella, “Large eddy simulation requirements for the flow over periodic hills,” *Flow Turbulence and Combustion*, vol. 103, p. 55–91, 2019.

[73] L. Prandtl, *Über ein neues Formelsystem für die ausgebildete Turbulenz.* Nachrichten der Akademie der Wissenschaften zu Göttingen, Mathematisch-Physikalische Klasse, Vandenhoeck & Ruprecht, 1945.

[74] W. Jones and B. Launder, “The prediction of laminarization with a two-equation model of turbulence,” *International Journal of Heat and Mass Transfer*, vol. 15, no. 2, pp. 301–314, 1972.

[75] D. C. Wilcox, “Reassessment of the scale-determining equation for advanced turbulence models,” *AIAA Journal*, vol. 26, no. 11, pp. 1299–1310, 1988.

[76] F. R. Menter, “Two-equation eddy-viscosity turbulence models for engineering applications,” *AIAA Journal*, vol. 32, no. 8, pp. 1598–1605, 1994.

[77] C. G. Speziale, R. Abid, and E. C. Anderson, “Critical evaluation of two-equation models for near-wall turbulence,” *AIAA Journal*, vol. 30, no. 2, pp. 324–331, 1992.

[78] B. Launder and G. Reece , “Progress in the development of a reynolds stress turbulence closure,” *Journal of Fluid Mechanics*, vol. 68, pp. 537 – 566, 1975.

[79] W. Rodi, “A new algebraic relation for calculating the reynolds stresses,” *ZAMM - Journal of Applied Mathematics and Mechanics / Zeitschrift für Angewandte Mathematik und Mechanik*, vol. 56, 1976.

[80] F. R. Menter, “Two-equation eddy-viscosity turbulence models for engineering applications,” *AIAA Journal*, vol. 32, no. 8, pp. 1598–1605, 1994.

[81] P. R. Spalart, W.-H. Jou, M. Strelets, and S. R. Allmaras, “Comments on the feasibility of LES for wings, and on a hybrid rans/les approach,” in *Advances in DNS/LES: Direct numerical simulation and large eddy simulation*, pp. 137–148, Greyden Press;, 1997.

[82] K. Duraisamy, G. Iaccarino, and H. Xiao, “Turbulence modeling in the age of data,” *Annual Review of Fluid Mechanics*, vol. 51, pp. 357–377, 2019.

[83] H. Xiao and P. Cinnella, “Quantification of model uncertainty in rans simulations: A review,” *Progress in Aerospace Sciences*, vol. 108, pp. 1–31, 2019.

[84] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning.* Springer Series in Statistics, New York, NY, USA: Springer New York Inc., 2001.

[85] A. E. Hoerl and R. W. Kennard, “Ridge regression: Biased estimation for nonorthogonal problems,” *Technometrics*, vol. 12, no. 1, pp. 55–67, 1970.

[86] R. Tibshirani, “Regression shrinkage and selection via the lasso,” *Journal of the Royal Statistical Society. Series B (Methodological)*, vol. 58, no. 1, pp. 267–288, 1996.

[87] G. Jin, H. Xing, R. Zhang, Z. Guo, and L. Junbiao, “Data-driven discovery of governing equations for transient heat transfer analysis,” *Computational Geosciences*, vol. 26, pp. 1–19, 2022.

[88] J. Zhang and W. Ma, “Data-driven discovery of governing equations for fluid dynamics based on molecular simulation,” *Journal of Fluid Mechanics*, vol. 892, 2020.

[89] H. Hazimeh, R. Mazumder, and A. Saab, “Sparse regression at scale: branch-and-bound rooted in first-order optimization,” *Mathematical Programming*, vol. 196, no. 1-2, pp. 347–388, 2022.

[90] R. Nayek, R. Fuentes, K. Worden, and E. Cross, “On spike-and-slab priors for bayesian equation discovery of nonlinear dynamical systems via sparse linear regression,” *Mechanical Systems and Signal Processing*, vol. 161, 2021.

[91] R. Fuentes, R. Nayek, P. Gardner, N. Dervilis, T. Rogers, K. Worden, and E. Cross, “Equation discovery for nonlinear dynamical systems: A bayesian viewpoint,” *Mechanical Systems and Signal Processing*, vol. 154, 2020.

[92] X. Xu and M. Ghosh, “Bayesian variable selection and estimation for group lasso,” *Bayesian Analysis*, vol. 10, no. 4, 2015.

[93] M. Raissi, P. Perdikaris, and G. E. Karniadakis, “Numerical gaussian processes for time-dependent and nonlinear partial differential equations,” *SIAM Journal on Scientific Computing*, vol. 40, no. 1, 2018.

[94] M. Raissi and G. E. Karniadakis, “Hidden physics models: Machine learning of nonlinear partial differential equations,” *Journal of Computational Physics*, vol. 357, p. 125–141, 2018.

[95] J. Berg and K. Nyström, “Data-driven discovery of PDEs in complex datasets,” *Journal of Computational Physics*, vol. 384, pp. 239–252, 2019.

[96] K. Duraisamy, “Perspectives on machine learning-augmented Reynolds-averaged and large eddy simulation models of turbulence,” *Physical Review Fluids*, vol. 6, no. 5, pp. 1–25, 2021.

[97] A. P. Singh and K. Duraisamy, “Using field inversion to quantify functional errors in turbulence closures,” *Physics of Fluids*, vol. 28, no. 4, 2016.

[98] J. Ling, R. Jones, and J. Templeton, “Machine learning strategies for systems with invariance properties,” *Journal of Computational Physics*, vol. 318, pp. 22–35, 2016.

[99] F. Millstein, *Convolutional Neural Networks In Python: Beginner’s Guide To Convolutional Neural Networks In Python.* CreateSpace Independent Publishing Platform, 2018.

[100] C. Rapp and M. Manhart, “Flow over periodic hills: An experimental study,” *Experiments in Fluids*, vol. 51, pp. 247–269, 2011.

[101] S. Beetham and J. Capecelatro, “Formulating turbulence closures using sparse regression with embedded form invariance,” *Physical Review Fluids*, vol. 5, no. 8, pp. 1–25, 2020.

[102] M. E. Tipping, “Sparse bayesian learning and the relevance vector machine,” *J. Mach. Learn. Res.*, vol. 1, p. 211–244, 2001.

[103] C. Oseen, *Über Wirbelbewegung in einer reibenden Flüssigkeit.* Arkiv för matematik, astronomi och fysik, Almqvist & Wiksell, 1911.

[104] H. P. Langtangen and A. Logg, *Solving PDEs in Python: The FEniCS Tutorial I.* Springer Publishing Company, Incorporated, 1st edition ed., 2017.

[105] S. Brenner and L. Scott, *The Mathematical Theory of Finite Element Methods*. Texts in Applied Mathematics, Springer New York, 2002.