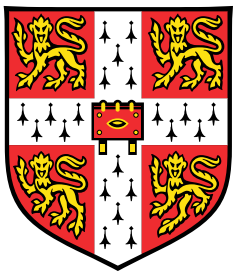
Data assimilation into physics-based thermoacoustic models using Bayesian neural network ensembles



Maximilian Louis Croci

Supervisor: Prof. Matthew P. Juniper

Advisor: Prof. Stewart Cant

Department of Engineering
University of Cambridge

This thesis is submitted for the degree of
Doctor of Philosophy

Pembroke College September 2022

Declaration

I hereby declare that except where specific reference is made to the work of others, the contents of this thesis are original and have not been submitted in whole or in part for consideration for any other degree or qualification in this, or any other university. This thesis is my own work and contains nothing which is the outcome of work done in collaboration with others, except as specified in the text and Acknowledgements. This thesis contains fewer than 65,000 words including appendices, footnotes, tables and equations and has fewer than 150 figures.

Maximilian Louis Croci
September 2022

Abstract

Title: Data assimilation into physics-based thermoacoustic models using Bayesian neural network ensembles
Author: Maximilian Louis Croci

Thermoacoustic instabilities, driven by the interaction between the heat release rate from the combustion process with the combustion chamber pressure waves, have long been a problem in jet and rocket engine design. Unacceptably large oscillations often appear during full-scale engine tests, despite being absent during part-scale tests, leading to costly re-designs. A physics-informed, data-driven model of a flame would allow for important quantities, such as the fluctuating heat release rate of the combustion process, to be estimated for a given burner geometry. This in turn would enable different geometries to be assessed for susceptibility to thermoacoustic instabilities before any physical testing, ensuring a cheaper design process.

In this thesis, data from artificial flame simulations and laboratory experiments are assimilated into increasingly complex physics-based models of the flame front. These physics-based models, based on the G-equation, are qualitatively correct. and their physical parameters must be inferred from the data to render them quantitatively accurate. For the artificial and laboratory Bunsen flames, the ensemble Kalman filter (EnKF) infers the parameters of the physics-based model and their uncertainties from a sequence of images. The method is reliable but computationally expensive: it takes hours for the EnKF to converge to parameter estimates and uncertainties for each test case. An alternative method is proposed, which uses a heteroscedastic Bayesian neural network ensemble (BayNNE) trained on a library of simulated flame fronts with known parameters to infer the parameters and uncertainties of the physics-based model. Generating the library of simulated flame fronts and training the BayNNE on the library are both computationally expensive. However, once trained, the BayNNE infers the parameters and uncertainties from an input sequence of six flame front images in milliseconds, which is fast enough to be used in real-time applications. The BayNNE method is applied to data from a version of the Volvo burner. The Volvo flame is noisy, exhibits transient behaviour and is observed over a limited spatial window, which

means that the EnKF is unable to converge. On the other hand, the BayNNE is able to robustly infer the parameters and uncertainties from a sequence of ten flame front images. Using the physics-based model with the inferred parameters, the flame front is extrapolated downstream of the observation window. This allows for the n and τ fields of a distributed $n-\tau$ model for the heat release of the burner to be calculated. These fields are entered into a Helmholtz solver to predict the growth rates and frequencies of the thermoacoustic system. The BayNNE method is then applied to the same data but with a more physically-informed model of the velocity field using the discrete vortex method. Although this thesis's conclusions for thermoacoustic behaviour are unsurprising, the BayNNE method is a potentially cheap way to combine sparse experimental measurements with complete numerical results and can readily be extended to other experiments and to other models.

As increasingly large quantities of experimental data become available, the researcher's challenge is to extract useful information without becoming overwhelmed by the quantity of data. Assimilation into physics-based models, as performed in this thesis, is attractive because the models are physically-interpretable and extrapolatable. The work presented shows one possible approach, and how it can be applied to a particularly long-standing problem in engineering: the modelling, and ultimately the control of, thermoacoustic instabilities.

"La liberté consiste à choisir soi-même ses chaînes."

— Bernard Willems-Diriken, dit Romain Guilleaumes

This thesis is dedicated to the victims of mental health illnesses
and
in particular
to my friend
Jaspal "Nunni" Thandi
15.07.1997 — 26.07.2020

Acknowledgements

Thank you to my supervisor, Matthew Juniper, for being a mentor and inspiration throughout my time in Cambridge. I hope this thesis goes some way to repaying the belief you placed in me.

A huge thank you to my close friends from the Hopkinson Laboratory in the Engineering Department. From tea-time boardgames to Friday playlists to "applied statistical workshops" in undisclosed locations, it's been quite the show. You've made my time as a PhD student utterly unforgettable. In order of appearance:

Francesco Garita	as	*Ciccio*
Stefano Falco	as	*himself*
Dr Hans Yu	as	*Anselmo*
Dr Ubaid Qadri	as	*Ubello*
Filip Gökstorp	as	*Pippo*
Ushnish Sengupta	as	*himself*
Alexandros Kontogiannis	as	*Papam*
Alberto Racca	as	*Racchino*
Santi Sukma	as	*herself*
Andrea Nóvoa	as	*La Andrea Galiciana*
Matthew Yoko	as	*himself*
Javier Lorente Macias	as	*El Javier Ibérico*
Ekrem Ekici	as	*himself*
Alessandro Giannotta	as	*Giannottino*

Thank you especially to Ushnish Sengupta, Joel Vasanth and Ekrem Ekici for the fruitful collaborations. I am proud and honoured to have co-authored papers with you.

Thank you to those who, through their passion for teaching, sent me down the path I find myself today. From my time in Belgium: Roger Hasler, for letting me try out a computer for the first time. I remember exactly the "game" that was to be played: I had to choose some

numbers at random and the computer would draw beautiful stars or flowers depending on the numbers. I now know these to be spyrographs, and they'll never cease to amaze me. Thank you to Maire Mairtin, for teaching me as much about discipline as about mathematics. To Barbara Josse, for simplifying all of mathematics into terms of *croissants* and *centimes*. A big thank you to Marie-Thérèse Filippi, for pushing me to pursue mathematics outside of the school curriculum and for giving me so much praise and support. To Zoltan Fodor, thank you for coming into school an hour before lessons started to teach me everything from cell biology to constellations. Those years were especially formative, and I still think about some of the philosophical questions you first asked me all those years ago.

In Manchester, I have to first thank Simmone Hewett, for placing your precious time and energy into the clumsy hands of teenagers. I'm sure we made you question your decisions regarding running the after-school engineering and physics clubs. And yet, whenever we needed help regarding the physics course and exams you were always happy to help us. Thank you to Dr Aidan Burrows, for being a mentor and a wicked chess opponent. I like to think you're relishing your retirement, reminiscing about that final game we played. I should have accepted the draw. I'd like to say a special thank you to Claire Metcalfe, for being the warmest, most inspirational and technically brilliant teacher of mathematics I've ever had. We once asked you how high a grade you expected from us in our AS-level exams. The answer filled us with terror which, lesson after lesson, was eventually nursed into belief.

To my best friends from school, Sang-Jin Lee, Hans Kwan, Alberto Meschi and Shaan Bassi. Those two years in Manchester were quite the adventure thanks to you. To my best friend from Fitzwilliam College and partner-in-crime, Jack Maloney: where to next? To Jaspal Thandi, it's so hard to accept that you're gone. Thank you for the Buttery brain teasers, the open-mic sessions, and the late nights playing pool in the JCR. Thank you to Chris Mellor, fellow *connoisseur* of the Fitz Library-Buttery-repeat schedule, for the years of friendship and for our victorious Isaac Newton pub quiz — perhaps our greatest achievement, all things considered.

To my best friends from Pembroke College, Alasdair Neilson, Annika Maus, Boris van Breugel, Campbell McLauchlan, David Morales, Emilie Wigdor, Ian Bretell, Karen Luong and Katharina Ramshorn, I don't know how to thank you enough for these past 4 years. I've had more fun than I ever thought possible, and I have the photos to prove it. People used to say that university was the best time of their lives. I now understand why.

Although they survive only in memory, I would like to thank my grandparents, Elena "Nonna" Burgazzi and Giovanni "Nonno" Croci. Nonna, thank you for teaching me times tables and nursery rhymes. Back then, I was too young to understand the meaning of the word "education". I'm now too old to be able to thank you in person for it. Nonno, thank you

for letting me hammer nails into every piece of wood in your garage workshop. I like to think that you were watching over me as I assembled thermocouples for the laboratory burner rig, dropped expensive custom-made equipment or gave an electric shock to my lab-mate (sorry, Ciccio).

Thank you to my parents, Remo and Dominique, and to my brothers, Victor and Elliott, without whom none of this would have been possible.

Finally, thank you to my *raison d'être*, Antonia Karin Wunnerlich.

—

Cambridge is a stroll down Trinity lane towards King's Parade, watching the cyclists hurry along to their morning lectures. It is a view over Plough Reach under the metronomic sound of oars and breath, heavy in anticipation of the Bumps races. It is a Fitzbillies Chelsea bun, giddily consumed to fight off the chill of a golden, late-autumn afternoon. It is the sound of the organ and evensong from the great sandstone chapels of King's, Trinity and St John's. Cambridge is the cackle of the open fire, the clinking of decanters, the whisper of college gowns in the candle-light.

Table of contents

List of figures

List of tables

List of Algorithms

Nomenclature

Roman Symbols

$\mathscr{D}$	Data
diag	Diagonal matrix
$\mathscr{H}$	Hypothesis; model
$\mathscr{L}$	Loss function
$\mathscr{N}$	Normal distribution
$\mathscr{O}$	Order of magnitude
Re	Reynolds number

Greek Symbols

μ	Mean
Σ	Covariance matrix
σ^2	Covariance
θ	Parameter

Subscripts

$(\cdot)_\theta$	Parameterised by θ

Acronyms / Abbreviations

BayNNE	Bayesian NN ensemble
BMA	Bayesian model averaging

CFD Computational fluid dynamics

CNN Convolutional NN

CPU Central processing unit

DNS Direct numerical simulation

DVM Discrete vortex method

ELBO Evidence lower bound

EnKF Ensemble KF

FC Fully connected

FEM Finite element method

FTF Flame transfer function

GNN Graph NN

GPU Graphics processing unit

GRU Gated recurrent unit

KF Kalman filter

KL Kullback-Leibler divergence

LES Large eddy simulation

LSTM Long short-term memory

MAP Maximum a-posteriori

MCMC Markov chain Monte Carlo

MFC Mass flow controller

MLE Maximum likelihood estimate

MML Maximum marginal likelihood

MVE Mean and variance estimation

NLL Negative log-likelihood

NN Neural network

PCA Principal component analysis

PDE Partial differential equation

PINN Physics-informed NN

PLIF Planar laser-induced fluorescence

RANS Reynolds-averaged Navier-Stokes

ReLU Rectified linear unit

RNN Recurrent NN

SCT Schwartz-Christoffel transformation

URANS Unsteady RANS

VI Variational inference

Chapter 1

Introduction

1.1 Thermoacoustics

1.1.1 Introduction

In a thermoacoustic system, temperature, density and pressure variations of acoustic waves interact with a heat source. The acoustic fields, typically the pressure or the velocity fields, cause the heat release rate at the heat source to fluctuate and this can cause acoustic waves to amplify. The physical mechanism driving thermoacoustic instability was described by Lord Rayleigh in 1878 [1]:

> *If heat be periodically communicated to, and abstracted from, a mass of air vibrating (for example) in a cylinder bounded by a piston, the effect produced will depend upon the phase of the vibration at which the transfer of heat takes place. If heat be given to the air at the moment of greatest condensation, or taken from it at the moment of greatest rarefaction, the vibration is encouraged. On the other hand, if heat be given at the moment of greatest rarefaction, or abstracted at the moment of greatest condensation, the vibration is discouraged.*

This mechanism is similar to that which drives an ideal piston engine. Mechanical work is done on the gas by the piston as it compresses the gas. Next, the gas is ignited and its temperature and pressure increase. Mechanical work is then done by the gas on the piston as the gas expands to its original volume. If the work done by the gas on the piston is greater than the work done by the piston on the gas, then heat released from combustion has been converted to mechanical work. In a thermoacoustic system, the piston is replaced with an acoustic wave. In the case of a ducted, premixed flame, the acoustic waves in the duct (a) perturb the flame and (b) compress and expand the gas around the flame [2]. If the

moments of higher (lower) than average heat release rate coincide with higher (lower) than average local pressure, then there is net conversion of heat into work over the cycle. If this work is not dissipated (for example via acoustic radiation from the open boundaries of the duct) then this work increases the amplitude of the acoustic waves, and the system is known as thermoacoustically unstable. Mathematically, the system is unstable if the following inequality, known as the Rayleigh criterion [3, 4], is true:

$$\int_T \int_V p'(\mathbf{x},t)\dot{q}'(\mathbf{x},t)\mathrm{d}\mathbf{x}\mathrm{d}t > \text{dissipation} , \tag{1.1}$$

where $p'(\mathbf{x},t)$ and $q'(\mathbf{x},t)$ represent the fluctuations of pressure and heat release rate per unit volume respectively, at position $\mathbf{x}$ and time t over a volume V and cycle of time T. This phenomenon has been known since at least the 1800s, when glassblowers produced a sound when blowing a hot bulb at the end of a cold narrow tube [5].

Thermoacoustic instabilities are a persistent problem in jet engine, rocket engine and gas turbine design. These industrial applications have huge power densities, which means that they become unstable even if a small proportion of heat is converted to work over a cycle. At present, there is no realistic alternative to combustion engines that can offer similar performance [2]. Despite decades of research, and the development of sophisticated physics-based models, thermoacoustic instability in these engines remains difficult to predict and eliminate. These instabilities are often absent during the design phase, only occurring during full-scale engine tests [6–8]. These instabilities are difficult to predict, due to their late appearance in the design stage and the sensitivity of the stability of the system to small design changes. Fixes are often *ad hoc*.

The persistence of thermoacoustic instability has motivated research into both prediction and elimination of instabilities. Research methods can be divided into three categories:

- Computational methods: Direct Numerical Simulation (DNS), Large Eddy Simulation (LES), U/RANS (Un/Steady Reynolds-averaged Navier-Stokes). Knowledge of the physics and chemistry of the system is modelled by governing equations (Navier-Stokes), combustion mechanisms (the chemistry of the reaction) and boundary conditions. This category includes low-order models such as the network model [9, 10] and the G-equation model [11].

- Experimental methods: construction of small-scale and full-scale combustor rigs on which experiments can be undertaken to observe thermoacoustic instabilities.

- Hybrid methods: data assimilation involves combining both simulations and experiments to improve predictions of model parameters and uncertainties by recognising

the limitations of experiments (noise, random and systematic) and simulations (lack of knowledge of the physics, model errors) [12]. Data can be assimilated implicitly into a low-order model through, for example, a flame transfer function (FTF) or $n-\tau$ model [13].

This thesis is within the hybrid scope, and uses data from simulations and experiments to improve predictions of the behaviour of thermoacoustic systems (Bunsen flames, gutter-stabilised flames) by assimilating experimental data into a low-order model.

1.1.2 Computational methods

Direct numerical simulation (DNS) is a computational fluid dynamics (CFD) technique that solves the discretised Navier-Stokes equations without any turbulence modelling. The whole range of spatial and temporal scales of the flow must be resolved, which incurs a huge computational cost [14]. Although this prohibits the use of DNS in industry, DNS simulations can supply information that could not be obtained from a laboratory experiment. In thermoacoustics research, the use of DNS is limited to small combustor designs, for examples see Ref. [15].

Large eddy simulation (LES) combines simulation of the large scales with modelling of the Reynolds stress due to turbulence at the small scales. The computational cost of simulations is decreased compared to DNS by modelling turbulence at the smallest length scales in the Navier-Stokes equations. However, the computational cost remains prohibitive in thermoacoustics [16] meaning that Reynolds-averaged Navier-Stokes (RANS), a cheaper alternative, has become the most widely used CFD technique in industry as a compromise between accuracy, performance and ease of use [17, 18]. Comparisons between time-averaged LES and RANS have been performed on simple combustor rigs such as the Volvo validation rig [19]. Unsteady RANS (URANS) is a CFD technique which uses RANS turbulence models in unsteady simulations. This simulates the long timescale motion, while modelling the short timescale motions as increased Reynolds shear stresses. These have been used in thermoacoustics, see for example [18, 20, 21]

All of these CFD methods require models of the coupling between heat release rate fluctuations and acoustic fluctuations in the system. The modelling of this coupling inevitably introduces model error. One of the simplest physics-based models of a flame's heat release rate is the distributed or point-wise $n-\tau$ model [13]: the heat release rate fluctuation is a linear multiple of the velocity perturbation at the base of the flame some time τ earlier. Although this model does not simulate the flame dynamics, the n and τ fields can be used in

a Helmholtz solver or in a network model to determine the growth-rate of a thermoacoustic perturbation in the system..

The flame transfer function (FTF) is another model. This specifies the relationship between acoustic velocity and heat release rate fluctuations as function of the forcing frequency [22–24]. The FTF is an extension of the $n-\tau$ model to include frequency-dependent behaviour, and is valid over a specified range of frequencies. The flame describing function (FDF) extends the FTF model by specifying the flame response as function of both the forcing frequency and the amplitude [22, 25, 26].

Another physics-based model of a flame is the *G*-equation model [11], which is used primarily for the study of laminar premixed flames [9, 27–29]. In this model, the flame's position in space and time is represented by the zero contour (or level-set) of a continuous function that advects with the flow and propagates into the unburnt region. The flow velocity field is an input to this model, and is usually specified using models for the (time-varying) horizontal and vertical components of velocity at any point in the domain [26]. These models restrict the velocity field to obey continuity, and are parameterised by parameters representing physical quantities such as flame dimension, laminar flame speed, perturbation convection speed and amplitude. This model allows the most influential flame dynamics to be simulated cheaply. To render the model quantitatively accurate, the parameters of this model need to be assimilated from experimental data.

1.1.3 Experimental methods

There are many experimental examples of thermoacoustic instability. For a detailed review of these, the reader is referred to [30–32]. The Rijke tube [33] is a cylindrical tube with open ends containing a heat source. Electric wires or Bunsen flames have been used as the heat source. When the heat source is switched on, a sustained tone is sometimes produced, which is driven by the coupling of the heat release rate fluctuations and the acoustic pressure or velocity waves in the tube. The Rijke tube is useful to investigate thermoacoustic oscillations because it contains much of the relevant physics present in a real engine. The Rijke tube can be used to experimentally evaluate the sensitivity of the thermoacoustic stability of the system to changes in tube geometry, boundary conditions and heat source positions in the tube. If the Rijke tube is made of a transparent material such as glass, images of the Bunsen flame can be taken and used to calibrate qualitative models (such as the *G*-equation model) of the flame using data assimilation.

At the larger scale, the Volvo rig is a laboratory-scale combustor which comprises a duct containing a bluff-body stabilised flame burning premixed gases [34, 35]. Intermittent thermoacoustic instability can be observed by varying the gas composition (stoichiometry).

Data from images of the flame taken through a window in the Volvo rig are assimilated into a *G*-equation model of the bluff-body stabilised flame.

1.2 Bayesian inference and prediction

Bayesian inference is a framework in which Bayes' theorem is used to calculate probabilities of unknown quantities, such as model parameters, and to compare candidate models of the data [36, Chapter 28]. The first level of inference involves calculating the likelihood of model parameters, given a model and some data. The results of this inference are often described in terms of the most likely parameter values and their uncertainties. If there are multiple candidate models, this analysis is repeated for each model. The second level of inference involves comparing candidate models that could describe the data. This is difficult because it is not possible simply to choose the model that fits the data best: more complex models can often fit the data better, but may be less plausible and give poor predictions on new data. Different models are compared by calculating each model's marginal likelihood. Predictions about new data can be made either using a best candidate model [36, Chapter 41] or using all candidate models [37].

1.2.1 First level of inference: model fitting

At the first level of inference, we wish to calculate the posterior probability of the parameters θ of a model $\mathscr{H}$, given some data $\mathscr{D}$. In other words, we wish to find the parameters that ensure the model best fits the data. Bayes' theorem states:

$$p(\theta|\mathscr{D},\mathscr{H}) = \frac{p(\mathscr{D}|\theta,\mathscr{H})p(\theta|\mathscr{H})}{p(\mathscr{D}|\mathscr{H})} \quad , \tag{1.2}$$

where $p(\mathscr{D}|\theta,\mathscr{H})$ is the *likelihood*, $p(\theta|\mathscr{H})$ is the *prior*, and $p(\mathscr{D}|\mathscr{H})$ is the *evidence* or *marginal likelihood*. The likelihood function gives the likelihood of the parameters, θ, given the data, $\mathscr{D}$. The prior encapsulates any a priori information about the values of θ, given the model. The marginal likelihood ensures that the posterior probability distribution sums to 1:

$$p(\mathscr{D}|\mathscr{H}) = \int p(\mathscr{D}|\theta,\mathscr{H})p(\theta|\mathscr{H})\mathrm{d}\theta \quad . \tag{1.3}$$

This quantity is ignored at this level of inference because it does not depend on the parameters. The most probable parameters $\hat{\theta}_{MAP}$ (i.e. those which maximise the posterior) are chosen:

$$\hat{\theta}_{MAP} = \mathrm{argmax}_\theta(p(\theta|\mathscr{D},\mathscr{H})) = \mathrm{argmax}_\theta(p(\mathscr{D}|\theta,\mathscr{H})p(\theta|\mathscr{H})) \quad . \tag{1.4}$$

If a uniform prior is used then the parameters, $\hat{\theta}_{MLE}$, that maximise the likelihood are also those that maximise the posterior:

$$\hat{\theta}_{MLE} = \text{argmax}_{\theta}(p(\mathscr{D}|\theta,\mathscr{H})) \quad . \tag{1.5}$$

When $\hat{\theta}_{MAP}$ (or $\hat{\theta}_{MLE}$) are calculated, error bars on these parameters can be obtained from the curvature of the log posterior. To do this, we start by Taylor-expanding the log-posterior about $\hat{\theta}_{MAP}$:

$$\ln p(\theta|\mathscr{D},\mathscr{H}) \approx \ln p(\hat{\theta}_{MAP}|\mathscr{D},\mathscr{H}) - \frac{1}{2}\left(\theta-\hat{\theta}_{MAP}\right)^T \mathbf{A}\left(\theta-\hat{\theta}_{MAP}\right) + \ldots \quad , \tag{1.6}$$

where the matrix $\mathbf{A}$ is the Hessian evaluated at $\hat{\theta}_{MAP}$:

$$\mathbf{A} = -\frac{\partial^2}{\partial\theta^2}\ln p(\theta|\mathscr{D},\mathscr{H})\bigg|_{\theta=\hat{\theta}_{MAP}} \quad . \tag{1.7}$$

By neglecting terms above second order in Eq. 1.6, we assume that the posterior can be locally approximated as a Gaussian with expectation $\hat{\theta}_{MAP}$ and covariance matrix $\mathbf{A}^{-1}$:

$$p(\theta|\mathscr{D},\mathscr{H}) \approx p(\hat{\theta}_{MAP}|\mathscr{D},\mathscr{H})\exp\left(-\frac{1}{2}\left(\theta-\hat{\theta}_{MAP}\right)^T \mathbf{A}\left(\theta-\hat{\theta}_{MAP}\right)\right) \quad . \tag{1.8}$$

Error bars on the parameters $\hat{\theta}_{MAP}$ can be calculated from the diagonal entries of the covariance matrix. This is method is known as *Laplace's method* [36, Chapter 28].

Alternative methods for performing Bayesian inference in general data modelling problems include exact methods [36, Chapter 25], Monte Carlo sampling [36, Chapter 29] and variational methods [36, Chapter 33]. Neural networks can also be used as surrogate models of the posterior $p(\theta|\mathscr{D},\mathscr{H})$ [38, Chapter 2]. However, there are few exact methods for performing Bayesian inference using neural networks. These include Monte Carlo methods [39] and Gaussian approximation methods [38, Chapter 3].

1.2.2 Second level of inference: model comparison

When a set of candidate models, $\{\mathscr{H}_i\}$ is being considered, we wish to infer which model is most plausible given the data. By Bayes' theorem:

$$p(\mathscr{H}|\mathscr{D}) = \frac{p(\mathscr{D}|\mathscr{H})p(\mathscr{H})}{p(\mathscr{D})} \quad . \tag{1.9}$$

The likelihood, $P(\mathscr{D}|\mathscr{H})$, in this case is the marginal likelihood (Eq. 1.3). The prior, $p(\mathscr{H})$, encapsulates our prior preference for each model, before any data are considered. The denominator is a constant which ensures the posterior probability , $p(\mathscr{H}|\mathscr{D})$, sums to 1. If we assume that each candidate model is equally probable, the best candidate model is found as the one with the maximum marginal likelihood (MML):

$$\mathscr{H}_{MML} = \text{argmax}_{\mathscr{H}}(p(\mathscr{D}|\mathscr{H})) \ , \tag{1.10}$$

The task therefore involves calculating the marginal likelihood $p(\mathscr{D}|\mathscr{H})$ of each model:

$$p(\mathscr{D}|\mathscr{H}_i) = \int p(\mathscr{D}|\boldsymbol{\theta}, \mathscr{H}_i) p(\boldsymbol{\theta}|\mathscr{H}_i) \mathrm{d}\boldsymbol{\theta} \ . \tag{1.11}$$

In many cases, the posterior $p(\boldsymbol{\theta}|\mathscr{D}, \mathscr{H}_i) \propto p(\mathscr{D}|\boldsymbol{\theta}, \mathscr{H}_i) p(\boldsymbol{\theta}|\mathscr{H}_i)$ has a strong peak at $\hat{\boldsymbol{\theta}}_{MAP}$ and is well approximated by a Gaussian. The marginal likelihood in these cases can be approximated as follows:

$$p(\mathscr{D}|\mathscr{H}_i) \approx \underbrace{p(\mathscr{D}|\hat{\boldsymbol{\theta}}_{MAP}, \mathscr{H}_i)}_{\text{Best fit likelihood}} \times \underbrace{\frac{p(\hat{\boldsymbol{\theta}}_{MAP}|\mathscr{H}_i)}{\sqrt{\det \mathbf{A}/2\pi}}}_{\text{Occam factor}} \ , \tag{1.12}$$

where $\mathbf{A}$ is the Hessian matrix evaluated at $\hat{\boldsymbol{\theta}}_{MAP}$, with jth row and kth column entry given by:

$$A_{j,k} = -\frac{\partial^2}{\partial \theta_j \partial \theta_k} \ln p(\boldsymbol{\theta}|\mathscr{D}, \mathscr{H}_i) \bigg|_{\boldsymbol{\theta}=\hat{\boldsymbol{\theta}}_{MAP}} \ . \tag{1.13}$$

The Occam factor penalises complex models with many parameters more than simpler models with fewer parameters. Therefore, the best model (the maximum marginal likelihood candidate) will be the one which fits the data well (large best fit likelihood) without being so complex as to be penalised heavily by the Occam factor.

1.2.3 Prediction and Bayesian model averaging

After the first and second level of Bayesian inference have been performed, the following are available:

- a set of candidate models, $\{\mathscr{H}_i\}$, from which the best candidate $\mathscr{H}_{MML}$ is selected;
- posterior probability distributions, $p(\boldsymbol{\theta}|\mathscr{D}, \mathscr{H}_i)$, of the parameters of each model given the data;

- the set $\{\hat{\theta}_{i,MAP}\}$ of maximum a posteriori parameters of each model.

We now wish to predict some unknown variable y from input x. In the simplest case, we can use the best model, $\mathscr{H}_{MML}$, and its MAP parameters $\hat{\theta}_{MAP}$ to calculate the predictive probability distribution:

$$p(y|x,\mathscr{D},\hat{\theta}_{MAP},\mathscr{H}_{MML}) \ . \tag{1.14}$$

However, this fails to take into consideration the uncertainty in the model parameters. Instead, by marginalising over the model parameters, we can compute the following predictive distribution:

$$p(y|x,\mathscr{D},\mathscr{H}_{MML}) = \int p(y|x,\mathscr{D},\theta,\mathscr{H}_{MML})p(\theta|\mathscr{D},\mathscr{H}_{MML})\mathrm{d}\theta \ . \tag{1.15}$$

In this way, the uncertainty in the model parameters is propagated into uncertainty in the predictions. In the Bayesian machine learning literature, this Bayesian marginalisation is sometimes called *Bayesian model averaging* (BMA) [40]. However, this name has also been used to describe the following two different marginalisations:

- Marginalising over models to find the posterior parameters given the data [41, Chapter 35]:

$$p(\theta|\mathscr{D}) = \sum_i p(\theta|\mathscr{D},\mathscr{H}_i)p(\mathscr{H}_i|\mathscr{D}) \ . \tag{1.16}$$

- Marginalising over the parameters of each model *and* over every model for prediction [37]:

$$\begin{aligned} p(y|x,\mathscr{D}) &= \sum_i p(y|x,\mathscr{D},\mathscr{H}_i)p(\mathscr{H}_i|x) \\ &= \sum_i \left[\int p(y|x,\mathscr{D},\theta,\mathscr{H}_i)p(\theta|\mathscr{D},\mathscr{H}_i)\mathrm{d}\theta\right] p(\mathscr{H}_i|x) \ . \end{aligned} \tag{1.17}$$

To avoid confusion, in this thesis the name *Bayesian model averaging* will only be used to refer to the marginalisation in Eq. 1.15.

1.3 Hybrid methods

1.3.1 Hybrid methods in thermoacoustics

Quantifying uncertainties is an important part of scientific modelling. Data assimilation allows for uncertainties in model parameters and predictions to be quantified. It is important

to distinguish between the different sources of uncertainty in the modelling process. In general, uncertainties fall into one of two categories:

- **Epistemic uncertainty**: The model is constructed based on our understanding of the physical processes driving the system. A mismatch (or bias) between the model predictions and experiments arises due to our limited understanding or modelling of the physics of the system. This in turn can be due to limited data (experiments, from which we build our understanding by identifying physical mechanisms expressed as governing equations) and limited prediction capabilities (simulation, through which we test the model predictions and compare with experiments). This uncertainty can be reduced by including progressively more equations describing the physics into the model, although this creates ever more complex models.

- **Aleatoric uncertainty**: Sources of aleatoric uncertainty include random measurement error and stochastic behaviour in the physical system. This uncertainty can be reduced, for example, by repeating experiments and taking an average measurement of a quantity of interest.

By quantifying the epistemic uncertainty in a model, the modeller can decide whether the model needs improving or the systematic error needs to be reduced. By quantifying the aleatoric uncertainty, the modeller can decide whether the random aspects of the experiment need improving or the random error needs to be reduced for example by performing more experiments.

Hybrid methods use data assimilation to calibrate a reduced-order model $\mathscr{H}$ using data $\mathscr{D}$. Once calibrated, the reduced-order model can be used to make predictions for an unknown quantity y from an input quantity x. The causal graphical model describing these assumptions is shown in Fig. 1.1. The first inference task is fitting the parameters of the model to the data: given observations $\mathscr{D}$, the posterior probability distribution, $p(\theta|\mathscr{D},\mathscr{H})$, of the parameters θ of assumed model $\mathscr{H}$ must be found. Predictions can be made using the fitted model. If only one set of parameters, $\hat{\theta}$, is used when making predictions, then the task involves finding the probability distribution, $p(y|x,\mathscr{D},\hat{\theta},\mathscr{H})$, of quantity y given input x, observations $\mathscr{D}$ and fitted model $\mathscr{H}$ with parameters $\hat{\theta}$. If a probability distribution of parameter values is considered when making predictions, then the task involves finding the probability distribution $p(y|x,\mathscr{D},\mathscr{H})$. These tasks can be solved in a Bayesian probability framework, as described in the previous section.

The reduced-order modelling methods mentioned earlier in this chapter involve uncertainty propagation. The uncertainties in model parameters are propagated to the model predictions to calculate the sensitivities of the predictions to the parameters. This can be

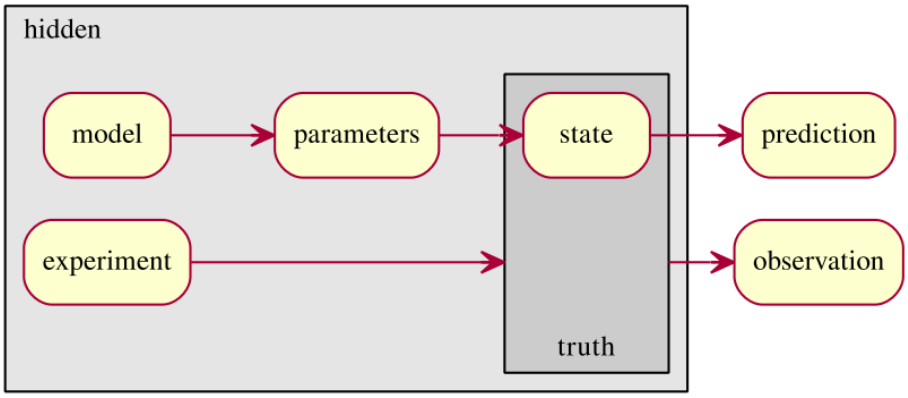


Fig. 1.1 Diagram of the causal graph of reduced-order modelling, reproduced from [42]. A reduced-order model $\mathscr{H}$ with parameters θ is assumed to be a suitable candidate for explaining observations $\mathscr{D}$, and can be used to make predictions y from an input x (for example, x could be the current state of the system). The first inference task is the *fitting the model*: given observations $\mathscr{D}$, what is the posterior probability distribution, $p(\theta|\mathscr{D},\mathscr{H})$, of the parameters θ of assumed model $\mathscr{H}$? The second inference task is the *prediction*: given the observations and the model, what is the probability distribution, $p(y|x,\mathscr{D},\mathscr{H})$, of prediction y from input x, integrating over the posterior distribution of the model parameters?

done using, for example, Monte Carlo methods. First, parameters $\{\theta_i\}$ are sampled from the posterior distribution $p(\theta|\mathscr{D},\mathscr{H})$. For each sample θ_i, we use the model to make a prediction y_i. These y_i are samples from the predictive distribution $p(y|x,\mathscr{D},\theta,\mathscr{H})$. By averaging these sample predictions, we are performing the Monte Carlo estimation:

$$y \sim p(y|x,\mathscr{D},\mathscr{H}) \ , \ \ y \approx \frac{1}{M}\sum_i y_i \ , \ \ y_i \sim p(y|x,\mathscr{D},\theta_i,\mathscr{H}) \ , \ \ \theta_i \sim p(\theta|\mathscr{D},\mathscr{H}) \tag{1.18}$$

This is sometimes written as [40] (for conciseness):

$$p(y|x,\mathscr{D},\mathscr{H}) \approx \frac{1}{M}\sum_i p(y|x,\mathscr{D},\theta_i,\mathscr{H}) \ , \ \ \theta_i \sim p(\theta|\mathscr{D},\mathscr{H}) \ . \tag{1.19}$$

This allows for the uncertainty in the parameters to be propagated into uncertainty in the predictions. If the predictive distribution is assumed to be Gaussian then these samples $\{y_i\}$ can be fitted to a Gaussian distribution, which can be described by a set of expected values and a covariance matrix.

For time series problems, inverse and forward uncertainty calculations are repeated at every time step when observations become available. By doing this, data is assimilated to reduce the uncertainty in model parameters and latent variables by combining model predictions and experiments. The Kalman filter [43] is an iterative method for estimating

the moments of the posterior distribution of the state x and parameters θ of a linear system given a prior $p(x,\theta)$ and noisy observations $\mathbf{y}$. We can append the state and parameters into a single vector $\mathbf{x}$. The method assumes that the forecast $\mathbf{x}^f$ is equal to the true state and parameters $\mathbf{x}^t$ with some model error δ^f:

$$\mathbf{x}^f = \mathbf{x}^t + \delta^f \quad . \tag{1.20}$$

Similarly, the observations are assumed to be noisy measurements of the true state, with measurement error ε:

$$\mathbf{y} = \mathbf{x}^t + \varepsilon \quad . \tag{1.21}$$

The noise terms are both assumed Gaussian:

$$\delta^f \sim \mathcal{N}\left(\mathbf{0}, \Sigma_\delta^f\right) \quad , \tag{1.22}$$

$$\varepsilon \sim \mathcal{N}\left(\mathbf{0}, \Sigma_\varepsilon\right) \quad . \tag{1.23}$$

This means that the prior and likelihood are both Gaussian:

$$\mathbf{x} \sim \mathcal{N}\left(\mathbf{x}^f, \Sigma_\delta^f\right) \quad , \tag{1.24}$$

$$\mathbf{y}|\mathbf{x} \sim \mathcal{N}\left(\mathbf{x}, \Sigma_\varepsilon\right) \quad . \tag{1.25}$$

Due to Bayes' rule, this means that the posterior is also Gaussian:

$$\mathbf{x}|\mathbf{y} \sim \mathcal{N}\left(\mathbf{x}^a, \Sigma_\delta^a\right) \quad , \tag{1.26}$$

$$\mathbf{x}^a = (\mathbf{I} - \mathbf{K})\mathbf{x}^f + \mathbf{K}\mathbf{y} \quad , \tag{1.27}$$

$$\Sigma_\delta^a = (\mathbf{I} - \mathbf{K})\Sigma_\delta^f \quad , \tag{1.28}$$

where $\mathbf{K}$ is the Kalman gain:

$$\mathbf{K} = \Sigma_\delta^f \left(\Sigma_\delta^f + \Sigma_\varepsilon\right)^{-1} \quad . \tag{1.29}$$

Eq. 1.27 shows that the posterior estimate of the state and parameters is a weighted combination of the previous estimate and the noisy measurement. This is the statistically optimal estimate given the modelling assumptions. With the moments of the posterior estimated, forecasts can be generated by integrating in time using the linear operator $\mathbf{F}$. This requires the assumption that the model error in forecasting is Gaussian with mean $\mathbf{0}$ and covariance Σ_w:

$$\mathbf{x}^f(t+1) = \mathbf{F}\mathbf{x}^a(t) \quad , \tag{1.30}$$

$$\Sigma_{\delta}^{f}(t+1) = \mathbf{F}\Sigma_{\delta}^{a}(t)\mathbf{F}^{T} + \Sigma_{w}(t) \quad . \tag{1.31}$$

Starting from initial guesses of the mean and covariance of the state and parameters, Eqs. 1.30 and 1.31 are applied until a measurement becomes available. Eqs. 1.27 and 1.28 are then applied to find the optimal posterior estimate.

Despite the strengths of the Kalman filter, its use is hampered by the assumptions of Gaussian distributions and linear models. Furthermore, the method is computationally expensive for systems with a large number of state variables and parameters: the covariance matrix requires $\mathscr{O}(N^2)$ variables to be stored and $\mathscr{O}(N^3)$ operations to be inverted in the Kalman gain calculation (Eq. 1.29). The ensemble Kalman filter (EnKF [44]) is a variant of the Kalman filter which was developed to avoid these limitations. Instead of storing the mean vector and covariance matrix of the state and parameters, an ensemble of $M << N$ approximately independent samples $\{\mathbf{x}_i\}^M$ are stored. These samples can be used to approximate the mean and covariance matrix using the ensemble averaged quantities:

$$\mathbf{x}^{f} \approx \frac{1}{M}\sum_{i=1}^{M}\mathbf{x}_i \quad , \tag{1.32}$$

$$\Sigma_{\delta}^{f} \approx \frac{1}{M-1}\sum_{i=1}^{M}(\mathbf{x}_i - \mathbf{x}^{f})(\mathbf{x}_i - \mathbf{x}^{f})^{T} \quad . \tag{1.33}$$

In this way, the storage requirements are reduced to $\mathscr{O}(MN)$ and the Kalman gain calculation requires $\mathscr{O}(M^3)$ operations [45]. This means that for modest ensemble sizes, the EnKF is more computationally efficient than the Kalman filter. For linear models, an EnKF with an infinite ensemble size converges to the Kalman filter. The ensemble Kalman filter has been used to assimilate data taken from direct numerical simulations (DNS) and the Rijke tube into a qualitative model of the flame to render the model quantitatively accurate [42, 46, 47].

An alternative method for estimating model parameters and uncertainties uses Bayesian inference and Laplace's method [36, Chapter 27]. In this method, the posterior of the parameters, θ, given the observations $\mathscr{D}$ and the assumption of a model $\mathscr{H}$ is:

$$p(\theta|\mathscr{D},\mathscr{H}) = \frac{p(\mathscr{D}|\theta,\mathscr{H})p(\theta|\mathscr{H})}{p(\mathscr{D}|\mathscr{H})} \quad . \tag{1.34}$$

Given the likelihood function $p(\mathscr{D}|\theta,\mathscr{H})$, the prior distribution $p(\theta|\mathscr{H})$ and observations $\mathscr{D}$, a gradient-based optimisation algorithm is used to calculate θ_{MAP}, the parameters which maximise this posterior distribution $p(\theta|\mathscr{D},\mathscr{H})$. The denominator of Eq. 1.34 is not a function of the parameters, and so is not required when finding θ_{MAP}. Laplace's method is

used to find the uncertainties in θ_{MAP}. This assumes that the posterior is Gaussian around θ_{MAP} and the covariance matrix is:

$$\Sigma = -\left(\frac{\partial^2}{\partial \theta^2} \ln p(\theta|\mathscr{D},\mathscr{H}) \Big|_{\theta=\hat{\theta}_{MAP}} \right)^{-1} . \tag{1.35}$$

The diagonal entries Σ_{ii} of the covariance matrix are the epistemic uncertainties in the parameter θ_i. The off-diagonal entries quantify the uncertainty between pairs of parameters. The most important quantities of the covariance matrix are its eigenvalues, which are used in principal component analysis (PCA, [48]). This method has been used to calculate the uncertainties the parameters of $n-\tau$ models of a Rijke tube [12].

When the probability distributions are not Gaussian, alternative methods must be used. The new task is to sample from the distributions, instead of estimating the moments of an assumed distribution. The Markov Chain Monte Carlo (MCMC) method is a statistical method for sampling from a target posterior distribution $p(\theta|\mathscr{D},\mathscr{H})$. This involves sampling θ from distribution which approximates the target distribution but is easier to sample from. The samples are corrected over time such that the approximate distributions converge to the target distribution. Because the target distribution is not assumed Gaussian, MCMC can be used to sample from nonlinear and multimodal distributions. However, the computational cost of the method is such that it cannot be used for online data assimilation. Where online data assimilation is required, particle filters generalise Kalman filters [49, 50]. MCMC has been used to assimilate pressure measurements from the electrical heater Rijke tube into a thermoacoustic network model for the acoustic oscillations [51].

1.3.2 Deep learning in fluid mechanics and combustion

Recently, deep learning methods have been applied in many fields. These include:

- Learning the chemical kinetics of combustion. For example, neural networks can be used to predict ignition delay times from rate coefficients in kinetics models [52], calculate the sensitivity of the kinetic model to its input parameters [53–55] or predict unknown reaction pathways [56].

- Subgrid modelling of turbulence for LES. Neural networks have been trained to learn probability distributions of mixture fraction and reaction progress variables, with accuracy improvements over traditional methods [57, 58].

- Learning physics-aware surrogate models. Physics-informed neural networks (PINNs, [59]) are neural networks trained with a loss function penalising predictions which do

not satisfy the Navier-Stokes equations. PINNs have been used to learn latent variables [60] and velocity and pressure fields from flow visualisations [61]. Furthermore, the learned PINNs can be used as surrogate models for simulation and uncertainty quantification [62].

- Learning data-driven surrogate models for computer graphics. When only qualitatively accurate simulations are required, simulating with data-driven surrogate models can be much cheaper than with solvers or physics-aware surrogate models [63, 64]. Random regression forests [65], long short-term memory networks (LSTMs [66]) and graph neural networks [67] have all been used to simulate plausible fluid flows [63, 68–70].

1.4 Bayesian machine learning and probabilistic methods

1.4.1 Regression for homoscedastic data

Consider a dataset comprising N data points $\{x_i, y_i\}^N$. Regression involves fitting a function $f(x;\theta)$ with parameters θ to this dataset. We assume the following relationship:

$$y = f(x;\theta) + \varepsilon \ , \tag{1.36}$$

where ε is assumed Gaussian with zero mean and variance σ^2. The variance σ^2 is the aleatoric uncertainty in the prediction $f(x;\theta)$. When the variance is the same for each point x, this is known as *homoscedasticity*. Alternatively, the variance can be a function of x: $\varepsilon \sim \mathcal{N}(0, \sigma^2(x))$. This is known as *heteroscedasticity*. We wish to find the parameters $\hat{\theta}$ which minimise the sum of the squared error between the prediction $f(x)$ and the correct value y:

$$\hat{\theta} = \text{argmin}_\theta \mathscr{L}(\theta) \ , \tag{1.37}$$

$$\mathscr{L}(\theta) = \sum_i (y_i - f(x_i;\theta))^2 \ . \tag{1.38}$$

In linear regression, the function $f(x;\theta)$ takes the form $f(x;\theta) = \theta_1 x + \theta_2$ and a closed form solution to the value $\hat{\theta}$ which minimises Eq. 1.38 can be found by setting the partial derivatives of $\mathscr{L}(\theta)$ with respect to the parameters to zero and solving for θ_1 and θ_2. Where a closed-form solution cannot be found, gradient-based optimisation methods must be used to minimise $\mathscr{L}(\theta)$. Under the homoscedastic assumption, $\hat{\theta}$ has a probabilistic interpretation: it is the value of θ which maximises the likelihood of the data given the function f. This can

be seen by first considering the likelihood function:

$$y|x,\theta \sim \mathcal{N}(f(x;\theta),\sigma^2) \ , \tag{1.39}$$

$$p(y|x,\theta) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left[-\frac{(y-f(x;\theta))^2}{2\sigma^2}\right] \ . \tag{1.40}$$

Assuming the data are independent and identically distributed:

$$p(\{y_i\}|\{x_i\},\theta) = \prod_i p(y_i|x_i,\theta) \ . \tag{1.41}$$

The negative log-likelihood (NLL) is:

$$\begin{aligned} \text{NLL} &= -\sum_i \log(p(y_i|x_i,\theta)) \\ &= \sum_i \left(\frac{(y_i-f(x_i;\theta))^2}{2\sigma^2} + \frac{1}{2}\log(2\pi\sigma^2)\right) \\ &= \frac{1}{2\sigma^2}\sum_i (y_i-f(x_i;\theta))^2 + \frac{N}{2}\log(2\pi\sigma^2) \end{aligned} \tag{1.42}$$

Maximising the likelihood means minimising the NLL. For homoscedastic σ^2, this means minimising:

$$\sum_i (y_i-f(x_i;\theta))^2 \ , \tag{1.43}$$

which is identical to the objective function, Eq. 1.38.

It is possible to go one step further, and to find θ_{MAP}, the maximum a-posteriori estimates of the parameters. To do this, we need to encapsulate any prior knowledge we may have about the parameters: $p(\theta)$. Let us assume a Gaussian prior with mean μ_θ and variance σ_θ^2: $\theta \sim \mathcal{N}(\mu_\theta,\sigma_\theta^2)$. We now wish to maximise the posterior:

$$p(\theta|\{x_i,y_i\}) \propto p(\{y_i\}|\{x_i\},\theta)p(\theta) \tag{1.44}$$

Taking the negative log of the posterior:

$$\begin{aligned} -\log p(\theta|\{x_i,y_i\}) &= \text{NLL} - \log p(\theta) \\ &= \frac{1}{2\sigma^2}\sum_i (y_i-f(x_i;\theta))^2 + \frac{N}{2}\log(\sigma\sqrt{2\pi}) \\ &+ \frac{(\theta-\mu_\theta)^2}{2\sigma_\theta^2} + \frac{1}{2}\log(2\pi\sigma_\theta^2) \end{aligned} \tag{1.45}$$

Thus, for homoscedastic data, maximising the posterior involves minimising:

$$\mathscr{L}(\theta) = \sum_i (y_i - f(x_i;\theta))^2 + \frac{\sigma^2}{\sigma_\theta^2}(\theta - \mu_\theta)^2 \tag{1.46}$$

This loss function combines the sum of squared errors with a *regulariser* term which penalises large parameter deviations from the prior mean. The ratio $\lambda = \sigma^2/\sigma_\theta^2$ determines the extent to which we deviate from the prior: if we are confident in our prior knowledge, a small σ_θ^2 is chosen which gives a larger λ and $\mathscr{L}(\theta)$ is dominated by the regulariser. If we use an uninformative prior, then σ_θ^2 is large and the likelihood dominates over the prior. If the mean of the prior is zero, the regularisation term becomes $\lambda\theta^2$. This is known as L^2 regularisation or *weight decay*, and is a successful method for avoiding neural network overfitting [71, Chapter 7]. Overfitting occurs when a model of the data learns some or all of the aleatoric noise in the data and poorly performs on new data not seen during training. Deep neural networks have a huge number of model parameters and are therefore especially susceptible to overfitting.

1.4.2 Regression for heteroscedastic data

In general, the aleatoric uncertainty in the data is heteroscedastic. Assuming, again, that the aleatoric uncertainty is due to additive Gaussian noise, the regression task then involves learning the mean $\mu(x;\theta)$ and variance $\sigma^2(x;\theta)$ functions of the Gaussian likelihood:

$$p(y|x,\theta) = \frac{1}{\sigma(x;\theta)\sqrt{2\pi}} \exp\left[-\frac{(y-\mu(x;\theta))^2}{2\sigma^2(x;\theta)}\right] . \tag{1.47}$$

When the functional form of the mean and variance functions is not known, neural networks can be used to learn these from the data. This is because neural networks are universal approximators: they can be used to learn any function to any degree of accuracy, so long as a large enough dataset of input-output pairs $\{x_i, y_i\}^N$ is available for training [72]. Learning the mean and variance functions is known as Mean and Variance Estimation (MVE) [73]. Given the dataset $\{x_i, y_i\}^N$, we wish to maximise the probability of this data under the assumptions of additive heteroscedastic aleatoric noise. Equivalently, this means minimising the Gaussian NLL loss function:

$$\begin{aligned}\mathscr{L}(\theta) &= -\log\left(\prod_i p(y_i|x_i,\theta)\right)\\ &= -\sum_i \log(p(y_i|x_i,\theta))\\ &= \sum_i \left(\frac{(y_i-\mu(x_i;\theta))^2}{2\sigma^2(x;\theta)} + \frac{1}{2}\log(2\pi\sigma^2(x;\theta))\right)\\ &= \sum_i \left(\frac{(y_i-\mu(x_i;\theta))^2}{\sigma^2(x;\theta)} + \log(\sigma^2(x;\theta))\right) + \text{const.}\end{aligned} \tag{1.48}$$

When trained with this loss function, a neural network will output the maximum likelihood mean and variance estimates for any input data point. This means that the aleatoric uncertainty in the data can be estimated. As before, maximum a-posteriori estimation can be performed by adding our prior knowledge as a regularisation term to the Gaussian NLL loss. Assuming a Gaussian prior $\theta \sim \mathscr{N}(\mu_\theta, \sigma_\theta^2)$, the loss function becomes:

$$\begin{aligned}\mathscr{L}(\theta) &= \sum_i \left(\frac{(y_i-\mu(x_i;\theta))^2}{\sigma^2(x;\theta)} + \log(2\pi\sigma^2(x;\theta))\right) + \frac{(\theta-\mu_\theta)^2}{\sigma_\theta^2} + \log(2\pi\sigma_\theta^2)\\ &= \sum_i \left(\frac{(y_i-\mu(x_i;\theta))^2}{\sigma^2(x;\theta)} + \log(\sigma^2(x;\theta))\right) + \frac{(\theta-\mu_\theta)^2}{\sigma_\theta^2} + \text{const.}\end{aligned} \tag{1.49}$$

Training in this way means we can learn any function of the data, whether the aleatoric noise is homoscedastic or heteroscedastic. This method is general but does not provide estimates of the model uncertainty or epistemic uncertainty in the predictions.

1.4.3 Bayesian model averaging: estimating epistemic uncertainty

In classical neural network training, an optimisation algorithm iteratively updates the parameters of the network until convergence to the MLE point estimate θ_{MLE}. For an input x, the network with parameters θ_{MLE} is used to output a single value y, with no estimate of the uncertainty in this value. Neural networks are usually highly underspecified by the available data, and therefore have diffuse likelihoods $p(\mathscr{D}|\theta)$. Different settings of the parameters can provide similarly suitable hypotheses for the data. The posterior $p(\theta|\mathscr{D}) \propto p(\mathscr{D}|\theta)p(\theta)$ is therefore not well approximated by a delta function. By training multiple neural networks on the same training data set but with different initial parameters, an ensemble can be trained. During training, the parameters of each neural network in the ensemble will converge to different values because the loss function has many local minima in general. This means that each neural network will predict different values for y given the same input x. The uncertainty

implied by the range of values predicted by the ensemble has been shown to be qualitatively accurate [74]. Ensembling has been shown to improve both accuracy and generalisation of neural networks [40]. However, this method only gives qualitative uncertainty estimates as it does not have a Bayesian interpretation.

Approximately Bayesian ensembling of neural networks [75] extends neural network ensembling and is a Bayesian model averaging method. In [75], the authors assume a dataset with additive homoscedastic Gaussian noise such that the likelihood takes the form $y|x \sim \mathcal{N}(\mu(x), \sigma^2)$ where the variance is constant and known a-priori. The method involves initialising an ensemble of M neural networks $\{\mu_j(x;\theta_j)\}^M$ each with unique initial weights and biases $\theta_{0,j} = [w_{0,j}, b_{0,j}]^T$ drawn from a prior distribution:

$$\theta_0 \sim \mathcal{N}(\mu_{prior}, \Sigma_{prior}) \ , \tag{1.50}$$

with diagonal covariance matrix $\Sigma_{prior} = \text{diag}(\sigma^2_{prior})$. Each neural network is trained independently. The loss function for the j-th member in the ensemble comprises the sum of squared errors and an anchored regularisation term:

$$\mathcal{L}(\theta_j) = \sum_i (y_i - \mu_j(x_i;\theta_j))^2 + (\theta_j - \theta_{0,j})^T \Lambda (\theta_j - \theta_{0,j}) \tag{1.51}$$

where $\Lambda = \text{diag}(\frac{\sigma^2}{\sigma^2_{prior}})$. When trained in this way, the parameters of each neural network in the ensemble converge to point estimates from the approximate Bayesian posterior $p(\theta|\mathcal{D})$. The M sets of parameters $\{\theta_j\}$ in the ensemble therefore represent M samples from the posterior. These can be used as Monte Carlo estimates in the predictive distribution:

$$p(y|x,\mathcal{D}) = \int p(y|x,\theta)p(\theta|\mathcal{D})\text{d}\theta \approx \frac{1}{M}\sum_j p(y|x,\theta_j) \ , \ \theta_j \sim \ p(\theta|\mathcal{D}). \tag{1.52}$$

The variance in the predictions $p(y|x,\theta_j)$ is the epistemic uncertainty, or model error. In [76], the authors extend this method to heteroscedastic data. The neural networks now estimate both mean and variance functions, $\{\mu_j(x,\theta_j), \sigma^2_j(x,\theta_j)\}^M$. The loss function becomes:

$$\mathcal{L}(\theta) = \sum_i \left(\frac{(y_i - \mu_j(x_i;\theta_j))^2}{\sigma^2_j(x_i;\theta_j)} + \log(2\pi\sigma^2_j(x_i;\theta_j)) \right) + (\theta_j - \theta_{0,j})^T \Lambda (\theta_j - \theta_{0,j}) \ , \tag{1.53}$$

where $\Lambda = \text{diag}(\frac{1}{\sigma^2_{prior}})$. As before, the predictive distribution is approximated using the M posterior parameter estimates $\{\theta_j\}^M$:

$$p(y|x,\mathscr{D}) \approx \frac{1}{M}\sum_j p(y|x,\theta_j)\ ,\ \theta_j \sim\ p(\theta|\mathscr{D}). \tag{1.54}$$

For ease of computing quantiles and predictive probabilities, this predictive distribution is often approximated by a Gaussian distribution:

$$y|x,\mathscr{D} \sim \mathscr{N}(\mu(x),\sigma^2(x)) \tag{1.55}$$

where the mean and variance functions are aggregated from the ensemble predictions:

$$\mu(x) = \frac{1}{M}\sum_j \mu_j(x;\theta_j)\ \ , \tag{1.56}$$

$$\sigma^2(x) = \underbrace{\frac{1}{M}\sum_j \sigma_j^2(x;\theta_j)}_{\text{heteroscedastic aleatoric uncertainty}} + \underbrace{\frac{1}{M-1}\sum_j \big(\mu_j(x;\theta_j)-\mu(x)\big)^2}_{\text{epistemic uncertainty}}\ . \tag{1.57}$$

This method allows us to quantify both aleatoric and epistemic uncertainties in the predictions.

1.4.4 Alternative uncertainty quantification methods in deep learning

One of the simplest methods for estimating the epistemic uncertainty in neural network predictions is Monte Carlo dropout [77]. In standard dropout [78], the outputs of each hidden layer node in a neural network are set to zero with some probability α each time the network is evaluated during training. This helps prevent overfitting by reducing the sensitivity of the output to nodes in the network that would otherwise dominate due to a large weight parameter. Monte Carlo dropout extends this method so that the random switching off of nodes in the network also happens at test time. Multiple evaluations of the network on the same input give a range of outputs, the variance of which is the epistemic uncertainty in the prediction. Although the method performs approximate Bayesian model averaging, the extent to which the method obeys Bayesian theory is debated [79].

Bayesian neural networks [39] are neural networks where each parameter in the network is replaced by a random variable with a probability distribution parameterised by parameters which are learned during training. The likelihood of such a network is intractable

and so no closed form solution to the predictive distribution

$$p(y|x) = \int p(y|x,\theta)p(\theta|\mathscr{D})\mathrm{d}\theta \tag{1.58}$$

exists. In practice, Monte Carlo sampling of each parameter in the network must be performed, which is prohibitively computationally expensive for Bayesian neural networks of practical size (thousands of hidden layer nodes, tens or hundreds of layers). Gaussian processes are more general than Bayesian neural networks (a single layer, infinitely wide Bayesian neural network is a Gaussian process). They are seen as the gold standard method for learning a predictive distribution, though suffer from prohibitive computational costs.

Variational inference (VI) is an alternative, Bayesian, sampling-free method. The method involves learning a tractable, proxy distribution $q(\theta)$ for the posterior distribution of the parameters, $p(\theta|\mathscr{D})$. The optimisation task involves maximising the evidence lower bound (ELBO):

$$\text{ELBO} = \int q(\theta)\log p(\mathscr{D}|\theta)\mathrm{d}\theta + \text{KL}(q(\theta)||p(\theta|\mathscr{D})) \tag{1.59}$$

where $\text{KL}(q(\theta)||p(\theta|\mathscr{D}))$ is the Kullback-Leibler divergence, which measures the similarity between two probability distributions. The main disadvantages of the method are that the the quality of approximation of the posterior is limited by both how restrictive the proxy distribution is, and the extent to which the optimisation routine converged. Both of these are difficult to evaluate in practice [80].

Chapter 2

Parameter inference for a kinematic model of a Bunsen flame

This chapter is based upon the following conference and journal publications. The first two were double-blind peer-reviewed by at least one reviewer. However, the reviews are not made available to the authors, but are used by the conference organisers to decide on acceptance.

- U. Sengupta, M. L. Croci, and M. P. Juniper, "Real-time parameter inference in reduced-order flame models with heteroscedastic Bayesian neural network ensembles," in *Machine Learning and the Physical Sciences workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS)*, (Virtual), 2020 [81];

- M. L. Croci, U. Sengupta, and M. P. Juniper, "Online parameter inference for the simulation of a Bunsen flame using heteroscedastic Bayesian neural network ensembles," in *Deep Learning for Simulation workshop at the 9th International Conference Conference on Learning Representations (ICLR)*, (Virtual), 2021 [82].

The third conference paper was submitted to the Machine Learning and Data Assimilation in Dynamical Systems (MLDADS) at the International Conference of Computer Science (ICCS 2021), and was peer reviewed by 3 reviewers before being further selected for publication in the Lecture Notes in Computer Science (LNCS) journal. The conference and journal publications are the following:

- M. L. Croci, U. Sengupta, and M. P. Juniper, "Data Assimilation Using Heteroscedastic Bayesian Neural Network Ensembles for Reduced-Order Flame Models," in *Machine Learning and Data Assimilation in Dynamical Systems (MLDADS) at the International Conference of Computer Science (ICCS)*, (Virtual), 2021;

- M. L. Croci, U. Sengupta, and M. P. Juniper, "Data Assimilation Using Heteroscedastic Bayesian Neural Network Ensembles for Reduced-Order Flame Models," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12746 LNCS, pp. 408–419, 2021 [83].

In this chapter, Mr Ushnish Sengupta conducted the Bunsen flame experiments and co-designed and co-implemented the neural network ensembles, using a method originally proposed in [76]. I implemented the ensemble Kalman filter [47], LSGEN2D [84] and the image processing.

2.1 Introduction

In this chapter, the parameters of a *G*-equation model of a Bunsen flame are inferred using two different methods: the ensemble Kalman filter [44] and heteroscedastic Bayesian neural network ensembles [76]. The Bunsen flame experiment is chosen as a precursor to larger experiments, such as the Volvo combustor in Chapters 3 and 4. Previously, the ensemble Kalman filter method has been used to assimilate data from *G*-equation simulations [46], direct numerical simulations (DNS) [85] and Bunsen flame experiments [47]. Heteroscedastic Bayesian neural networks ensembles are implemented for faster real-time inference than can be achieved with the ensemble Kalman filter. The method requires the creation of a library of simulated flame fronts with known *G*-equation parameters, which is computationally expensive. The heteroscedastic Bayesian neural networks are trained using supervised machine learning, which is also computationally expensive. However, once these two steps have been performed, the neural networks can infer the the parameters of the *G*-equation model six orders of magnitude faster than the ensemble Kalman filter.

2.2 The Bunsen flame

2.2.1 Experiment

Fig. 2.1 shows the configuration of the Bunsen experiment: a Bunsen burner is placed inside a transparent duct and images of the flame are taken with a high-speed camera at $f_s = 2500$ frames per second and a resolution of 1200×800 pixels. The flame is forced acoustically with a speaker from 250 Hz to 450 Hz. For this range of frequencies, the frame rate is high enough to avoid aliasing effects. The gas composition (methane, ethene and air) and flow rate are varied using mass flow controllers. By varying the forcing frequency and amplitude and gas composition and flow rate, as shown in Table 2.1, flames with different aspect ratios,

propagation speeds and degrees of cusping of the flame front are observed. In some cases, the flame front cusping leads to pinch-off at the flame tip. For each of the 270 different flame operating conditions, 500 images are taken.

Table 2.1 Bunsen flame experiment gas and air flow rates, unstretched flame speeds s_L^0 and excitation frequencies. The unstretched flame speeds are calculated using Cantera [86].

Fig.	Ethene (NL/min)	Methane (NL/min)	Air (NL/min)	s_L^0 (m/s)	Exc. Freq. (Hz)
2.11	0.300	0.300	600	0.726	450
2.12	0.400	0.200	600	0.710	425
2.13	0.267	0.400	600	0.585	300
2.14	0.300	0.150	450	0.810	400
2.15	0.200	0.300	450	0.663	325
2.16	0.225	0.225	450	0.848	275
2.17	0.150	0.150	300	1.122	350
2.18	0.200	0.100	300	1.217	375
2.19	0.133	0.200	300	0.764	250
2.20	0.133	0.200	300	0.764	-

The flame images are processed and the flame front is extracted as a radial location x, which is a single-valued function of the axial co-ordinate y: $x = f(y)$. First, the pixel intensities are thresholded and the flame location x for every vertical co-ordinate y is found by weighted interpolation of the thresholded pixels, where the weights are the pixel intensities. Next, splines with 28 knots are used to smooth $x(y)$. Each flame image is therefore converted into a 90×1 vector of flame front x locations $\mathbf{x}$. The y co-ordinates are the same for all flames, so are discarded. Observation vectors $\mathbf{z}$ are created by stacking ten consecutive $\mathbf{x}$ vectors. These observation vectors are the inputs to the neural networks. All 500 images of each Bunsen flame are processed in this way.

2.2.2 Non-dimensionalisation

The characteristic scales of the problem are the radius of the burner R, the length of the unstretched unforced flame, L, the excitation angular frequency $2\pi f$ and the time taken by a particle to convect from the base to the tip of the flame, L/V, and its inverse V/L which is called here the spatial frequency. The Strouhal number is the ratio of the spatial time scale to the excitation time scale (where the time scales are the reciprocal of the frequencies) $\mathrm{St} = 2\pi fL/V$. By defining the aspect ratio of the unstretched, unforced flame $\beta = L/R$, the Strouhal number may be written equivalently as $\mathrm{St} = 2\pi f\beta R/V$.

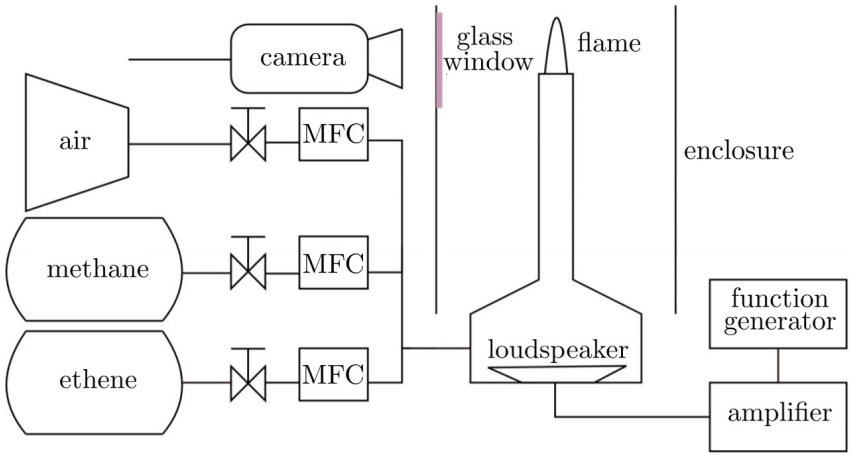


Fig. 2.1 Diagram of the Bunsen flame experiment setup: a Bunsen burner is placed inside a tube and a high-speed camera takes images of the flame through a glass window. The fuel composition (air, methane and ethene) and flow rate are controlled using mass flow controllers (MFCs). The flame is forced acoustically using a loudspeaker. This diagram is due to Mr Ushnish Sengupta.

2.3 The G-equation and LSGEN2D, the G-equation solver

2.3.1 G-equation model

In this thesis the flame front is defined to be the $G = 0$ contour (or level-set) of a scalar field $G(x,y,t)$. Regions of negative and positive G correspond to unburnt and burnt gases respectively (the magnitude of G has no physical significance). The evolution of G is governed by:

$$\frac{\partial G}{\partial t} + \mathbf{v} \cdot \nabla G = s_L |\nabla G|, \tag{2.1}$$

where $\mathbf{v}$ is a prescribed velocity field and s_L is the laminar flame speed: the speed at which the flame front propagates normal to itself into the reactants. The flame speed s_L is a function of the unstretched (adiabatic) flame speed s_L^0, the flame curvature κ and the Markstein length $\mathscr{L}$, and is insensitive to pressure variations:

$$s_L = s_L^0 (1 - \mathscr{L}\kappa). \tag{2.2}$$

The unstretched flame speed s_L^0 depends only on the flame chemistry. The velocity field $\mathbf{v}$ comprises a parabolic base flow profile $V(x)$ and superimposed continuity-obeying velocity

perturbations $u'(x,y,t)$ and $v'(x,y,t)$:

$$\mathbf{v} = \left(V(x) + v'\right)\mathbf{j} + u'\mathbf{i}, \tag{2.3}$$

$$\frac{V(x)}{V} = 1 + \alpha\left(1 - 2\left(\frac{x}{R}\right)^2\right), \tag{2.4}$$

$$\frac{v'(x,y,t)}{V} = \varepsilon \sin\left(\mathrm{St}\left(\frac{Ky}{R} - t\right)\right), \tag{2.5}$$

$$\frac{u'(x,y,t)}{V} = -\frac{\varepsilon K \mathrm{St} x}{\beta R}\cos\left(\mathrm{St}\left(\frac{Ky}{R} - t\right)\right), \tag{2.6}$$

where α determines the shape of the base flow profile ($\alpha = 0$ is uniform flow, $\alpha = 1$ is Poiseuille flow), ε is the amplitude of the vertical velocity perturbation with phase speed V/K, $\mathrm{St} = 2\pi f R\beta/V$ is the Strouhal number with forcing frequency f and flame radius R, and β is the aspect ratio of the unperturbed flame. These velocity perturbations model experimental observations [87, 24] and DNS results [29].

The parameters $K, \varepsilon, \mathscr{L}, \alpha, \mathrm{St}$ and β are tuned to fit an observed flame shape. Fig. 2.2 shows a diagram of the flame front under the G-equation model. This model allows cusps to form at the flame front and pockets of unburnt reactants to detach from the flame tip, as is observed in some experiments. It has proven to be a versatile flame front model in several previous studies, despite having only a few parameters [88].

2.3.2 LSGEN2D: the G-equation solver

Starting from a valid G field state that satisfies Eq. 2.1, LSGEN2D ([84]) evaluates $\partial G/\partial t$ and then updates G in a narrow band around the $G = 0$ contour. The flame wrinkling and cusping are assumed to be symmetric about the centre-line. Axial and radial oscillations of the flame (but not non-axisymmetric oscillations) are therefore modelled. Because the imposed velocity field is periodic, the states converge to a forced cycle, typically within 4 periods. The normalised area variation over one period can be calculated from the forced cycle states, which for uniform s_L[1] is proportional to the normalised heat release rate [9, 27].

LSGEN2D iterates the G-equation for a non-dimensional flame, scaled vertically by βR, horizontally by R and with $V = 1$. The transformation from non-dimensional LSGEN2D coordinates (x^*, y^*) into dimensional coordinates (x,y) of a flame with parameters R and β is $(x,y) = (Rx^*, \beta R y^*)$. The x coordinates take values between 0 and $2R$ (0 corresponds to the

[1] Although s_L is not uniform in regions of high curvature, such as at the flame tip, this effect is small.

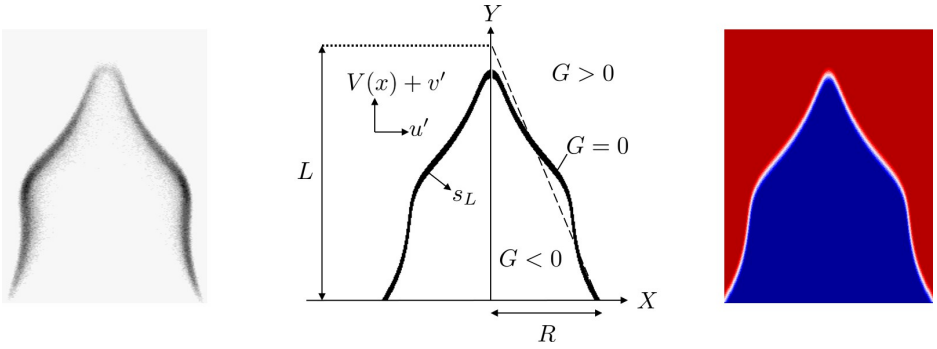


Fig. 2.2 *Left*: An image of a Bunsen flame. *Middle*: In the G-equation model, the flame front is represented by the $G = 0$ contour (or level-set) of a continuous scalar field $G(x,y,t)$. Unburnt and burnt gases are regions where $G < 0$ and $G > 0$ respectively. The flame front travels normal to itself into the unburnt gases with speed s_L. The flame front advects under the prescribed velocity field, which comprises continuity-obeying velocity perturbations $u'(x,y,t)$ and $v'(x,y,t)$ superimposed onto a steady base flow profile $V(x)$. *Right*: A G field solution in LSGEN2D, a level-set solver. Blue and red regions are unburnt and burnt gases respectively, and the thin white band represents the flame front.

centreline, R to the radius width and $2R$ to G field domain boundary), and the y coordinates range from 0 to $1.8\beta R$.

2.3.3 Evidence of transients and implications for data assimilation

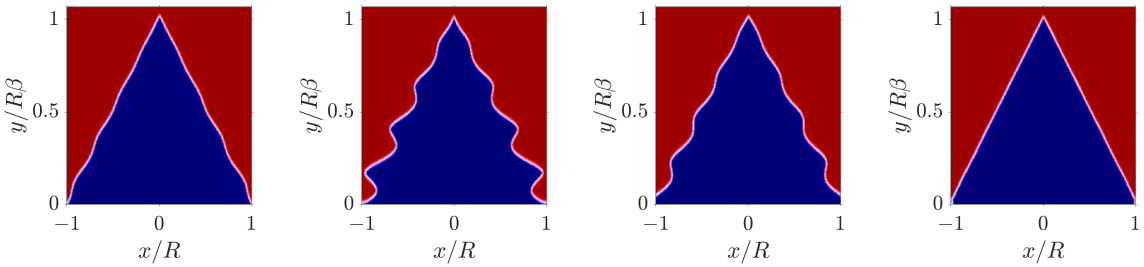
Starting from a steady G field solution and forcing the flame with a prescribed oscillating velocity field, the G field shows transient behaviour before reaching a forced cycle. A linear perturbation analysis (see appendix A) predicts this transient behaviour to last for a duration t_c:

$$t_c = \frac{(1+\beta^2)}{\beta}\frac{R}{V} \quad . \tag{2.7}$$

This agrees with our intuition for flames with large β (tall, slender flames): $t_c \approx \beta R/V$ is the time it takes for a perturbation travelling with speed V to travel from the base of the flame to its tip, a distance βR downstream. For a forced cycle with period $T = 1/f$, the number of periods it takes for the system to reach the forced cycle, p_c, is:

$$\begin{aligned} p_c &= \frac{t_c}{T} \\ &= \frac{(1+\beta^2)}{\beta}\frac{fR}{V} \quad . \end{aligned} \tag{2.8}$$

These transients are indeed found when solving the G-equation with LSGEN2D, as shown in Fig. 2.3. This happens with both pinned and floating (where the flame attachment point is free to move along the burner lip) boundary conditions at the burner lip. Furthermore, if the parameters of the model are varied at any point in the simulation, transients reappear until the system reaches a new forced cycle for this new set of parameters. This same behaviour is confirmed using the MATLAB solver, as seen in Fig. 2.4.



(a) $t = 0.1T$. (b) $t = 0.5T$. (c) $t = 0.8T$. (d) $t = T$.

Fig. 2.3 One cycle of transient nature in the LSGEN2D solution.

The implication of this fact is that when assimilating data into the G-equation model, one must either study cases for which the transients are very short-lived, or the data assimilation process must be slowed down to allow for the transients to die away.

2.4 Library of forced flame simulations

A library of flame front locations is created with the flame front model at known parameter values $K, \varepsilon, \mathscr{L}, \alpha$, St, β and f/f_s in the same format as the observation vectors, $\mathbf{z}$. The parameter values are sampled using quasi-Monte Carlo sampling to ensure good coverage of the parameter space. The parameters are sampled from the ranges in Table 2.2. The values of St are calculated using $\mathrm{St} = 2\pi f R\beta/V$. The parameters are sampled 8500 times, normalised to between 0 and 1 and recorded in target vectors $\{\mathbf{p} = [K, \varepsilon, \mathscr{L}, \alpha, \mathrm{St}, \beta]\}$.

The G field is iterated forward in time for a duration of t_c seconds (Eq. 2.7) (p_c periods , Eq. 2.8, where the period is $1/f$). This allows any transient flame behaviour to die away. For parameters sampled according to Table 2.2, p_c varies between 0.2 and 20. Then, the G field is iterated forward for a further period. The value of the G field (which varies between the range of -0.03 and 0.03) at $N_T = 200$ equally spaced timesteps within this period is recorded: $\{G_1, G_2, \ldots, G_{N_T}\}$. For each G field in the sequence $\{G_i\}$, the flame front $y = f(x)$ is extracted from the $G = 0$ contour for all x in the range of the experiment observation window. The flame front y coordinates are recorded in a column vector $\mathbf{x}_i$. The

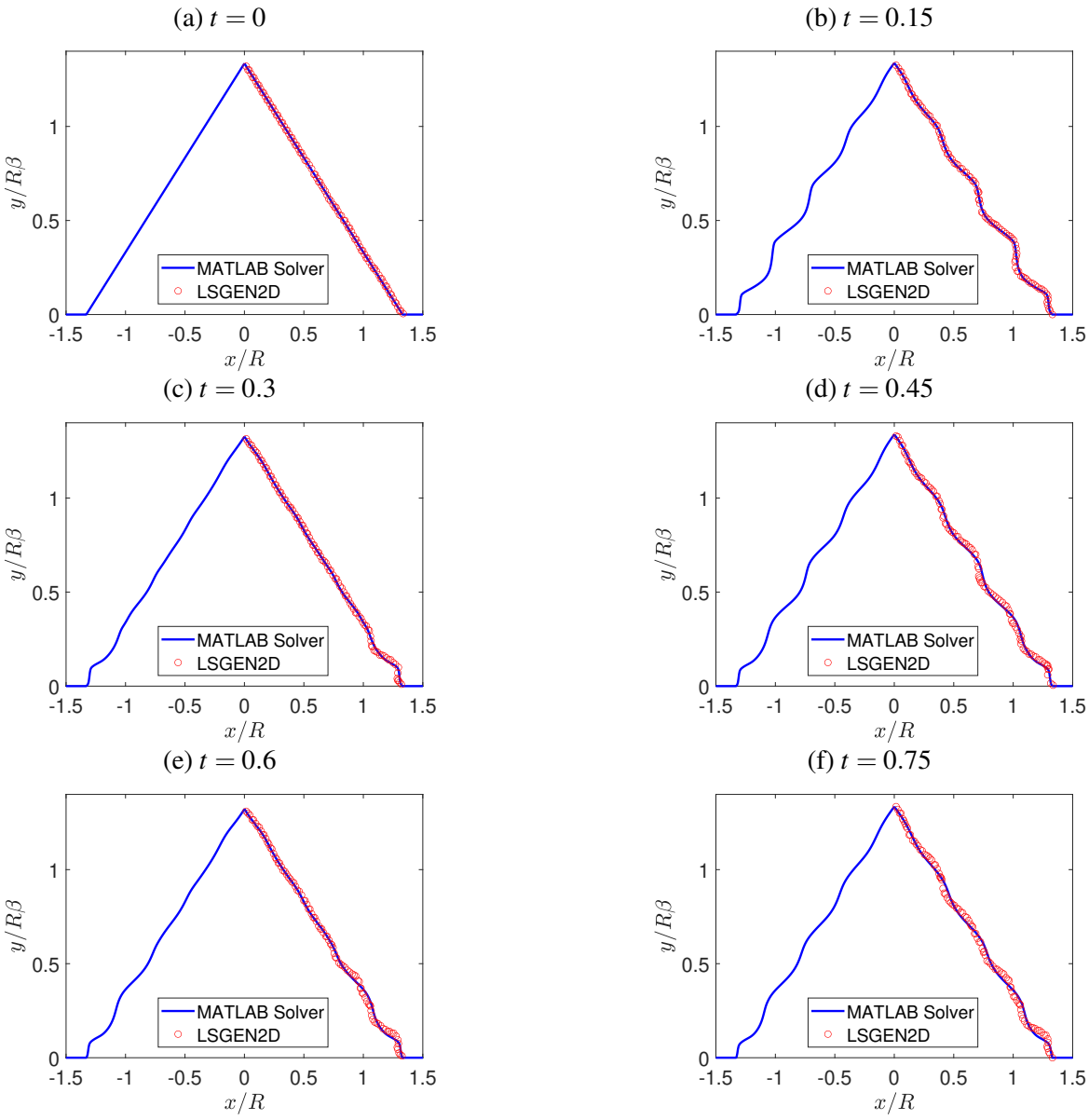


Fig. 2.4 MATLAB solution (blue solid line) to the G-equation flame model with parameters $R = 1.335$, $\beta = 13.2$, St $= 40$, $K = 0.5$ and $\varepsilon = 0.03$. The LSGEN2D solution (red circles) is superposed onto one half of the MATLAB solution for comparison. Note that only the $G = 0$ contour is shown, and the plots are scaled vertically by $1/\beta$. A long-lasting transient can be seen, with period of 0.3 seconds.

resulting sequence $\{\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_{N_T}\}$ represents the flame front position at N_T equally spaced timesteps in the period. Next, the sum of squared errors between the first and last G fields in the sequence $\{G_i\}$ is calculated. If this error is below a tolerance tol, the frame sequence $\{G_i\}$ is deemed to be a valid forced cycle. If the error is above the tolerance, the frame sequence is deemed unconverged, and is discarded along with its target parameters. The

frame rate of the forced cycle sequence is N_T/T which is not, in general, equal to the frame rate of the experimental data ($f_s = 2.5 \times 10^3$ Hz). To create a sequence of ten simulated flame fronts with the same frame rate as the experimental data starting from a timestep t_1, we calculate the ratio $\Delta = N_T f/f_s$ and select the sequence $\{\mathbf{x}_{t_1}, \mathbf{x}_{t_1+\Delta}, \ldots, \mathbf{x}_{t_1+9\Delta}\}$. The vectors in this sequence are appended together to make a column vector $\mathbf{z}$. This is repeated for all N_T of the starting timesteps $t_1, t_2 \ldots t_{N_T}$, and for $P = 8500$ combinations of parameters sampled from the ranges shown in Table 2.2. The result is a library of $PN_T = 1.7 \times 10^6$ observation - parameter pairs. This is summarised in algorithm 1.

The algorithm can be trivially parallelised because each iteration of the `for` loop (sampling the parameters, iterating LSGEN2D and calculating the flame fronts) is independent and can be performed on a single CPU core. The library is generated using the Cambridge Service for Data Driven Discovery's Peta4 supercomputer[2]. The computational budget allows for a maximum of 320 Intel Xeon Skylake CPU cores[3] to be used concurrently. One loop of algorithm 1 takes one to two hours, depending on the transient dissipation time t_c. An upper bound for the library generation's computational cost is $8500 \times 2 = 1.7 \times 10^4$ CPU core-hours. This takes approximately 54 hours when at the limit of maximum concurrent CPU core usage.

Table 2.2 Ranges of parameter values. The parameters are sampled using quasi-Monte Carlo sampling from between these ranges.

Parameter ranges	Description
$0.0 < K \leq 1.5$	Ratio of mean base flow speed to perturbation phase speed
$0.0 < \varepsilon \leq 1.0$	Amplitude of the vertical velocity perturbation
$0.0 \leq \alpha \leq 1.0$	Base flow shape
$0.02 \leq \mathscr{L} \leq 0.08$	Markstein length
$2.0 \leq \beta \leq 10.0$	Aspect ratio of the unperturbed flame
0.002 m $< R \leq 0.004$ m	Flame radius
1.0 m/s $< V \leq 5.0$ m/s	Mean base flow speed
200 Hz $\leq f \leq 500$ Hz	Excitation frequency

[2]https://www.hpc.cam.ac.uk/systems/peta-4

[3]Following an upgrade in November 2021, this number was increased to 448 Intel Xeon Platinum CPU cores.

Algorithm 1: Generate a library of simulated Bunsen flame fronts with known parameters.

Data: Number of parameter samples P, number of timesteps per period N_T, frame rate f_s, tolerance tol.
Result: Library of pairs $(\mathbf{p}_i, \mathbf{z}_i)_{i=1}^{PN_T}$
for $1 \leq p \leq P$ **do**
  $K, \varepsilon, \mathscr{L}, \alpha, f, \beta, R, V \leftarrow sample()$; ▷ Sample parameters from Tab. 2.2.
  $\mathrm{St} \leftarrow 2\pi f \beta R / V$;
  $\mathbf{p} \leftarrow [K, \varepsilon, \mathscr{L}, \alpha, \mathrm{St}, \beta]$; ▷ Record the target parameters.
  $T \leftarrow 1/f$;
  $t_c \leftarrow R(1+\beta^2)/\beta V$; ▷ Calculate the transient duration (see Appendix A).
  $t \leftarrow 0$;
  $dt \leftarrow T/N_T$;
  $LSGEN2D.initialise(\mathbf{p})$; ▷ Initialise LSGEN2D with target parameters.
  while $t \leq t_c$ **do**
    ▷ Iterate the G-equation forward in time for the duration of the transient period.
    $LSGEN2D.iterate()$;
    $t \leftarrow t + dt$;
  end
  while $t_c \leq t \leq t_c + T$ **do**
    ▷ Once the transient period is over, iterate for a further period and record the flame front x coordinates.
    $LSGEN2D.iterate()$;
    $G_t \leftarrow LSGEN2D.G$;
    $\mathbf{x}_t \leftarrow G_t.extractFront()$;
    $t \leftarrow t + dt$;
  end
  if $|G_{t_c+T} - G_{t_c}| < tol.$ **then**
    ▷ Verify that the convergence of the forced cycle by comparing the first and last G fields in the cycle.
    $\Delta \leftarrow N_T f / f_s$;
    for $t_0 = 1 : N_T$ **do**
      $\mathbf{z} \leftarrow concatenate(\mathbf{x}_{t_0}, \mathbf{x}_{t_0+\Delta}, \ldots, \mathbf{x}_{t_0+9\Delta})$; ▷ Create observation vector.
      $library.addPair(\mathbf{p}, \mathbf{z})$; ▷ Save the parameter - observation pair.
    end
  end
end

2.5 Parameter inference for the G-equation model of the Bunsen flame

2.5.1 The heteroscedastic Bayesian neural network ensemble (BayNNE) methodology

We assume that the posterior probability distribution of the parameters, given the observations, can be modelled by a neural network, $p_\theta(\mathbf{p}|\mathbf{z})$, with its own parameters θ. Here, θ comprises all of the weights and biases in the neural network. We assume that this posterior distribution has the form:

$$p_\theta(\mathbf{p}|\mathbf{z}) = \mathscr{N}\left(\boldsymbol{\mu}(\mathbf{z}), \boldsymbol{\Sigma}(\mathbf{z})\right), \tag{2.9}$$

where $\boldsymbol{\Sigma}(\mathbf{z})$, the posterior covariance matrix of the parameters given the data, is diagonal with $\boldsymbol{\sigma}^2(\mathbf{z})$ on its diagonal. This enforces our assumption that the parameters are mutually independent, given the observations $\mathbf{z}$. We use an ensemble of $M = 20$ neural networks. The architecture of each neural network is shown in Fig. 2.5. Each neural network comprises an input layer, four hidden layers with ReLU activations and two output layers: one for the mean vector $\boldsymbol{\mu}(\mathbf{z})$ and one for the variance vector $\boldsymbol{\sigma}^2(\mathbf{z})$. The output layer for the mean uses a sigmoid activation to restrict outputs to the range $(0,1)$. The output layer for the variance uses an exponential activation to ensure positivity. Each neural network in the ensemble is initialised with unique weights $\boldsymbol{\theta}_{j,anc}$ sampled from a Gaussian prior distribution $\mathscr{N}(0, \frac{1}{N_H})$ and biases $\mathbf{b}_{j,anc}$ sampled from a uniform prior distribution in the range $[-\frac{1}{\sqrt{N_H}}, \frac{1}{\sqrt{N_H}}]$.

For a single observation $\mathbf{z}$, the j-th neural network in the ensemble produces a mean and variance estimate of the G-equation parameters:

$$\boldsymbol{\mu}_j(\mathbf{z}), \boldsymbol{\sigma}_j^2(\mathbf{z}). \tag{2.10}$$

This is achieved by minimising the loss function $\mathfrak{L}_j$:

$$\begin{aligned}\mathfrak{L}_j = \Big(\boldsymbol{\mu}_j(\mathbf{z}) - \mathbf{p}\Big)^T \boldsymbol{\Sigma}_j(\mathbf{z})^{-1} \Big(\boldsymbol{\mu}_j(\mathbf{z}) - \mathbf{p}\Big) + \log\left(|\boldsymbol{\Sigma}_j(\mathbf{z})|\right) \\ + \left(\boldsymbol{\theta}_j - \boldsymbol{\theta}_{anc,j}\right)^T \boldsymbol{\Sigma}_{prior}^{-1} \left(\boldsymbol{\theta}_j - \boldsymbol{\theta}_{anc,j}\right).\end{aligned} \tag{2.11}$$

The first two terms of the loss function are the negative logarithm of the normalised Gaussian likelihood function up to an additive constant. The third term is a regularising term that penalises deviation from prior anchor values $\boldsymbol{\theta}_{anc,j}$. The NNs produce samples from the posterior distribution. This is called randomised maximum a-posteriori (MAP) sampling [75].

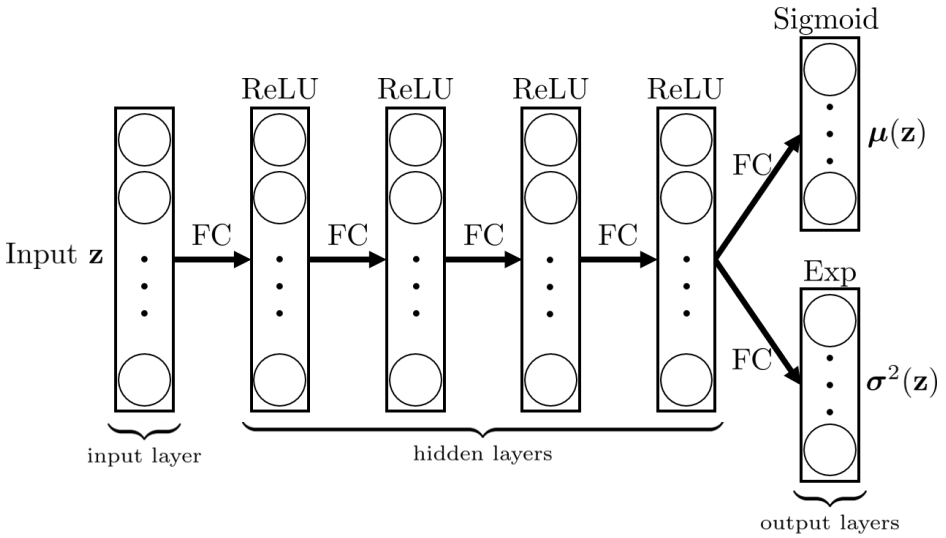


Fig. 2.5 Architecture of each neural network in the ensemble of 20. The input and hidden layers have 900 nodes each, while each output layer has six nodes each. All layers are fully connected (FC). Rectified Linear Unit (ReLU) activation functions are used for the hidden layers and sigmoid and exponential (Exp) activation functions are used for the mean and variance output layers respectively.

For a single observation vector $\mathbf{z}$, the prediction from the ensemble of neural networks is therefore a distribution of M Gaussians, each centred at their respective means $\boldsymbol{\mu}_j(\mathbf{z})$. Following similar treatment in [74], this distribution is then approximated by a single multivariate Gaussian posterior distribution $p(\mathbf{p}|\mathbf{z}) \approx \mathcal{N}(\boldsymbol{\mu}(\mathbf{z}), \boldsymbol{\Sigma}(\mathbf{z}))$ with mean and variance:

$$\boldsymbol{\mu}(\mathbf{z}) = \frac{\Sigma_j \boldsymbol{\mu}_j(\mathbf{z})}{M}, \quad \boldsymbol{\Sigma}(\mathbf{z}) = \mathrm{diag}\left(\boldsymbol{\sigma}^2(\mathbf{z})\right), \tag{2.12}$$

$$\boldsymbol{\sigma}^2(\mathbf{z}) = \frac{\Sigma_j \boldsymbol{\sigma}_j^2(\mathbf{z})}{M} + \frac{\Sigma_j \boldsymbol{\mu}_j^2(\mathbf{z})}{M} - \left(\frac{\Sigma_j \boldsymbol{\mu}_j(\mathbf{z})}{M}\right)^2. \tag{2.13}$$

This is repeated for every observation vector $\mathbf{z}$. The posterior distribution $p(\mathbf{p}|\mathbf{z}_i)$ with the smallest total variance $\sigma^2_{i,\mathrm{tot}} = ||\boldsymbol{\sigma}^2(\mathbf{z}_i)||_1$ is chosen as the best guess to the true posterior. The M parameter samples from the chosen posterior are used for re-simulation, which allows us to check the predicted flame shapes and to calculate the normalised area variation over one cycle.

2.5.2 The ensemble Kalman filter

The Kalman filter iteratively performs Bayesian inference to find the probability distribution of the state of a system given noisy observations of the system and an imperfect model of the

system dynamics. In this study, the state comprises the location of the flame front and the parameters K and ε. These parameters are assumed to be independent given the observations of the flame front and constant for each of the Bunsen flame experiments. The flame front is modelled using the G-equation (Eq. 2.1). The ensemble Kalman filter [44] (EnKF) evolves an ensemble of simulations forward in time. The covariance matrix of these ensembles is assumed to approximate the covariance matrix of the state evolved over the same period of time. The EnKF is more practical when the state contains many variables and the evolution is nonlinear, as discussed in Chapter 1, subsection 1.3.1. In this study, the state contains $O(10^2)$ variables and the governing equation (Eq. 2.1) is nonlinear.

The parameters $\mathscr{L}, \alpha, \beta$ and St are calculated by solving the G-equation (Eq. 2.1) when steady and do not need to be inferred with the EnKF. This reduces the cost of the EnKF but increases the number of steps compared with the BayNNE method. The forcing frequency f is manually set when running the Bunsen flame experiments. The unperturbed laminar flame speed s_L^0 is calculated using Cantera[4] and knowledge of the methane and ethene flow rates. V is calculated from s_L^0 and β: $V = s_L^0\sqrt{1+\beta^2}$. The Strouhal number can then be calculated: $\mathrm{St} = 2\pi f\beta R/V$. Ref. [42] contains details about the implementation of the EnKF.

An ensemble size of 32 is used in this study. A multiplicative inflation factor of 1% is chosen to mitigate the underestimation of the error covariances due to the finite ensemble size [89]. Once the EnKF has converged, the parameters K and ε calculated by each ensemble member are recorded. These are samples from the posterior distribution of the parameters given all the flame x-location vectors: $p(K, \varepsilon|\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_N)$. This differs from the BayNNE, which was given $\mathbf{x}$-location vectors in groups of ten.

2.6 Results and discussion

2.6.1 Neural network training and evaluation

The ensemble of 20 Bayesian neural networks is trained on the forced cycle library for 5000 epochs, with a 80 : 20 train-test split and an Adam optimiser with learning rate 10^{-3}. Training takes approximately twelve hours per neural network on an NVIDIA P100 GPU. The ensemble is trained in parallel on the Cambridge Service for Data Driven Discovery's GPU-accelerated cluster (Wilkes[5]). Once the neural networks are trained, ensemble prediction takes $O(10^{-3})$ seconds on an Intel Core i7 processor on a laptop.

[4]Cantera is a suite of tools for problems involving chemical kinetics, thermodynamics, and transport processes [86].

[5]https://www.hpc.cam.ac.uk/systems/wilkes-2

2.6.2 Ensemble Kalman filter training and evaluation

For each test case, the ensemble Kalman filter technique requires an hour to calculate estimates of $\mathscr{L}, \alpha, \mathrm{St}$ and β on an Intel Core i7 processor on a laptop, followed by 2 hours of data assimilation on a pair of Intel Xeon Skylake 6142 processors to produce estimates of K and ε. The EnKF is fully parallelised: the processors have 32 cores in total, one for each member in the ensemble.

2.6.3 Results on simulated data

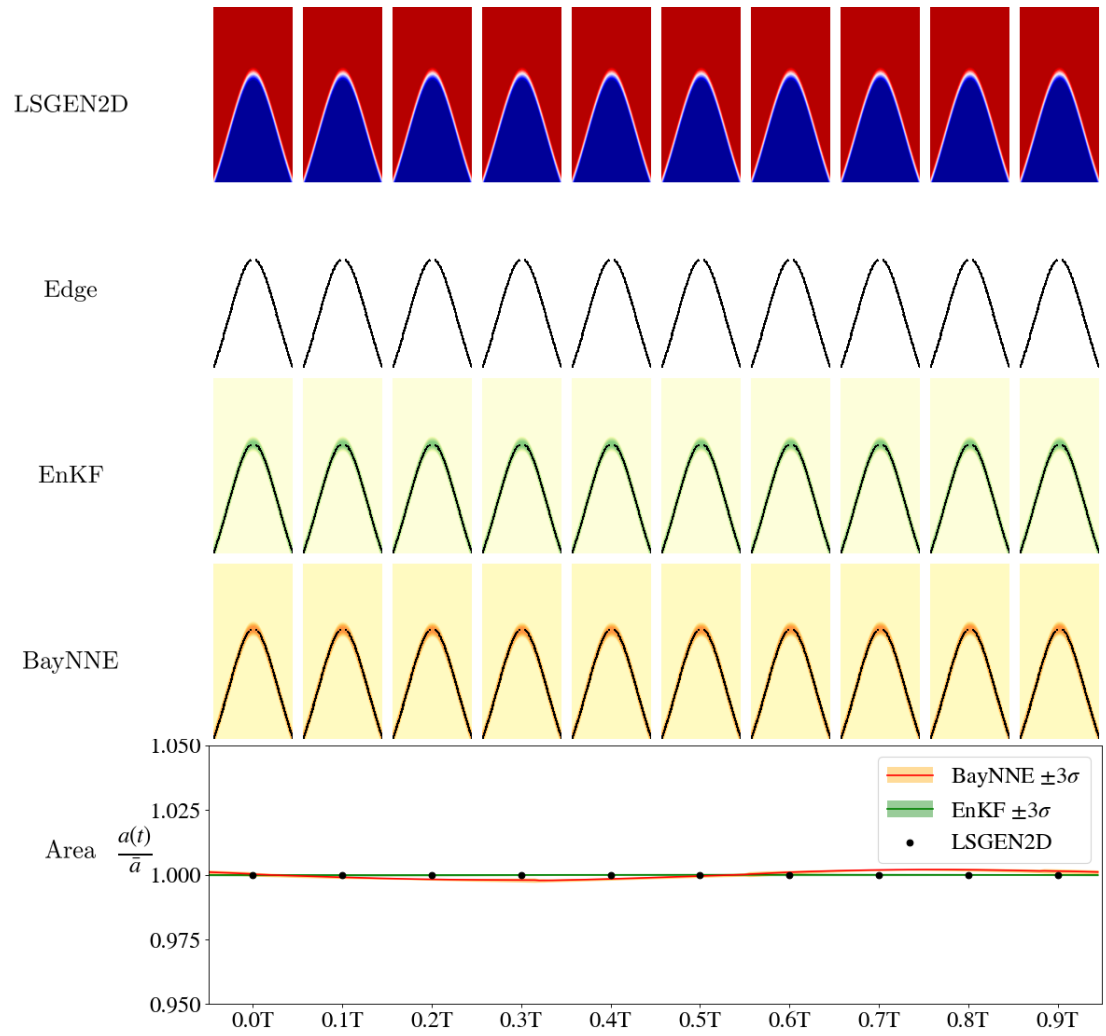
To confirm the validity of the EnKF and BayNNE methods, twin experiments are performed: the EnKF and BayNNE ensembles must recover the parameters of simulated flames generated in LSGEN2D. These flames do not appear in the BayNNE's library but have parameters within the library's parameter space. The parameter estimates are then re-simulated using LSGEN2D to compare the flame shapes and the normalised flame surface area variation over one period, $a(t)/\bar{a}$. Twin experiments are performed for ten different simulated flames, of which five are reported in this thesis, which cover a wide range of flame shapes and behaviour. The ground truth parameters for each of the five reported twin experiments are shown in Table 2.3.

Table 2.3 Ground truth parameter values of the twin experiments.

Twin experiment	β	α	$\mathscr{L}$	St	K	ε	Fig.
Steady	9.217	0.319	0.058	-	-	-	2.6
1	8.128	0.707	0.029	28.83	0.306	0.490	2.7
2	9.833	0.462	0.070	21.05	0.095	0.474	2.8
3	4.389	0.931	0.043	17.03	0.892	0.214	2.9
4	9.359	0.992	0.053	33.92	0.681	0.268	2.10

Both methods perform equally well in recovering the ground truth parameters. The normalised surface area variations agree with the ground truth as well. For cases where pinch-off of the flame tip occurs (figures 2.9, 2.10), the BayNNE predicted parameters are more uncertain. This is expected because pinch-off is a rare occurrence and so there will be fewer examples of pinch-off in the simulated flame-front library than examples without pinch-off. This means that the BayNNE will have had less training data with which to learn to infer the ground truth parameters. This limitation could be addressed by identifying regions of parameter space which give rise to pinch-off behaviour and sampling parameters more frequently in these regions. In contrast, the EnKF predicted parameters are more certain. This is also to be expected because rare behaviour contains more information compared to

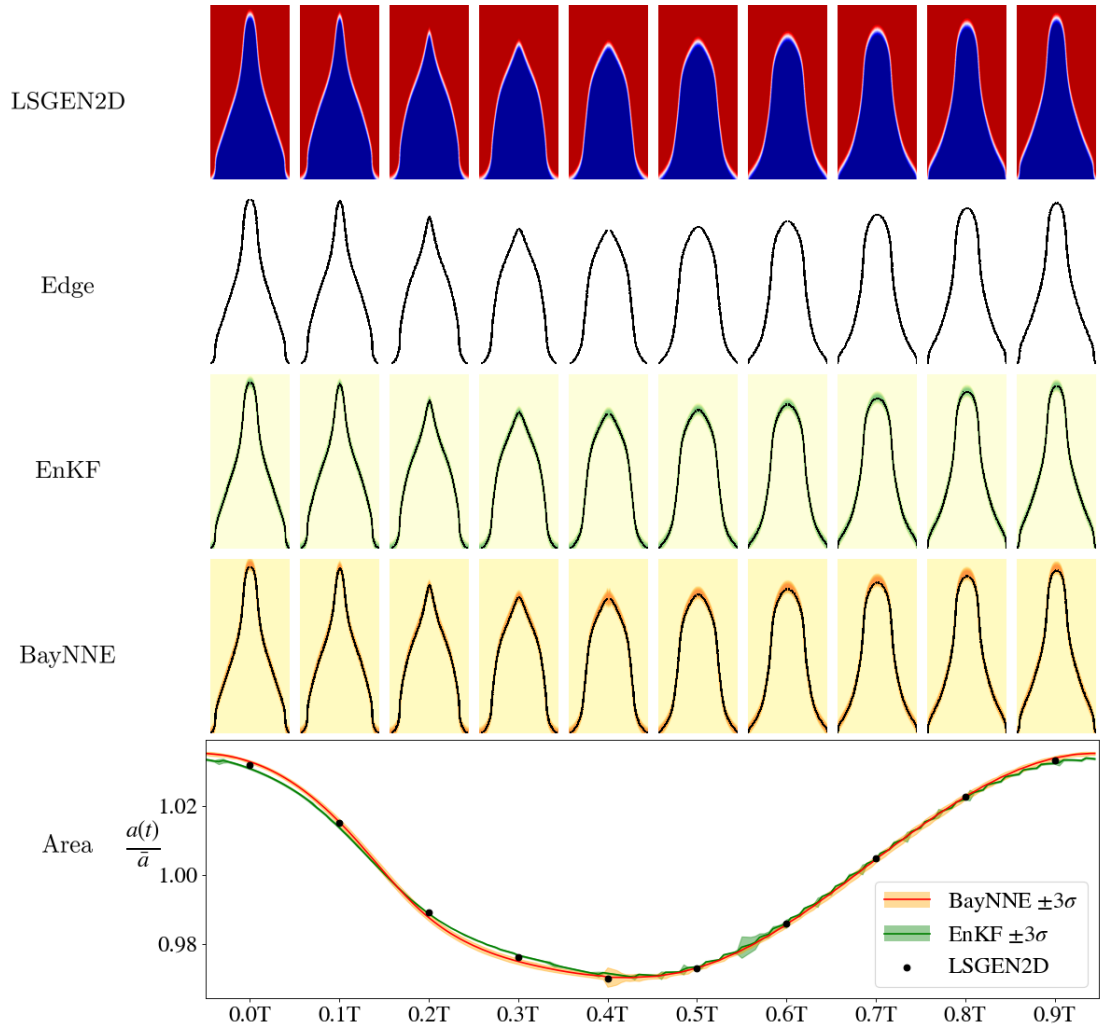
common behaviour. By assimilating multiple periods of data during which pinch-off occurs, the EnKF can converge to the ground-truth parameters much more quickly by learning from this information. These results give us confidence in the validity of both methods, which can now be use to assimilate data from experiments.



Assimilated parameters

Parameter	Visible for	Ground Truth	EnKF	BayNNE
β	steady & oscillating flames	9.217		9.099 ± 0.284
α		0.319		0.317 ± 0.021
$\mathcal{L}$		0.058		0.057 ± 0.003

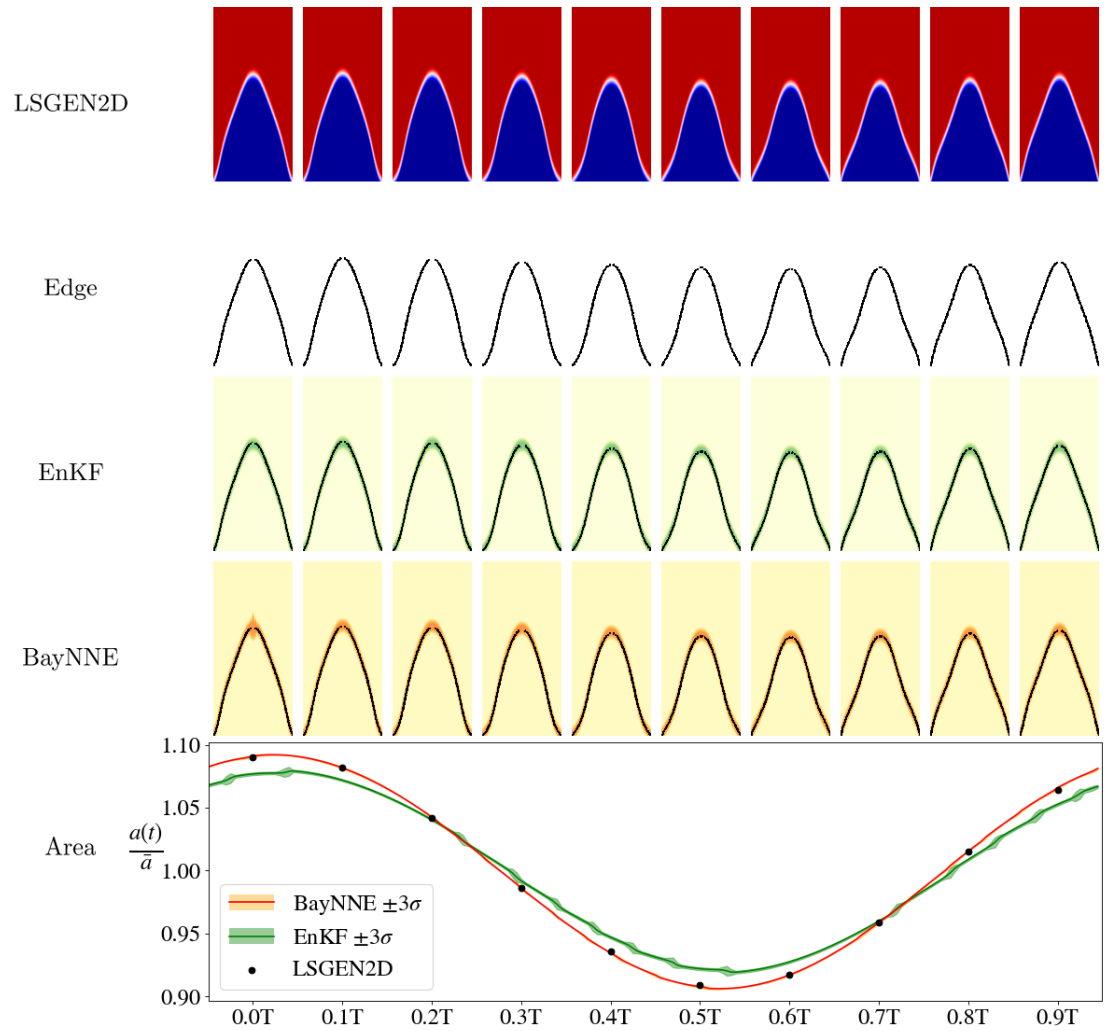
Fig. 2.6 Results of inference on the steady state twin experiment. Top four rows: LSGEN2D flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Ground Truth	EnKF	BayNNE
β	steady & oscillating flames	8.128		8.137 ± 0.638
α		0.707		0.714 ± 0.062
$\mathcal{L}$		0.029		0.030 ± 0.004
St	oscillating flames only	28.83		28.42 ± 3.987
K		0.306	0.303 ± 0.001	0.306 ± 0.022
ϵ		0.490	0.495 ± 0.003	0.492 ± 0.038

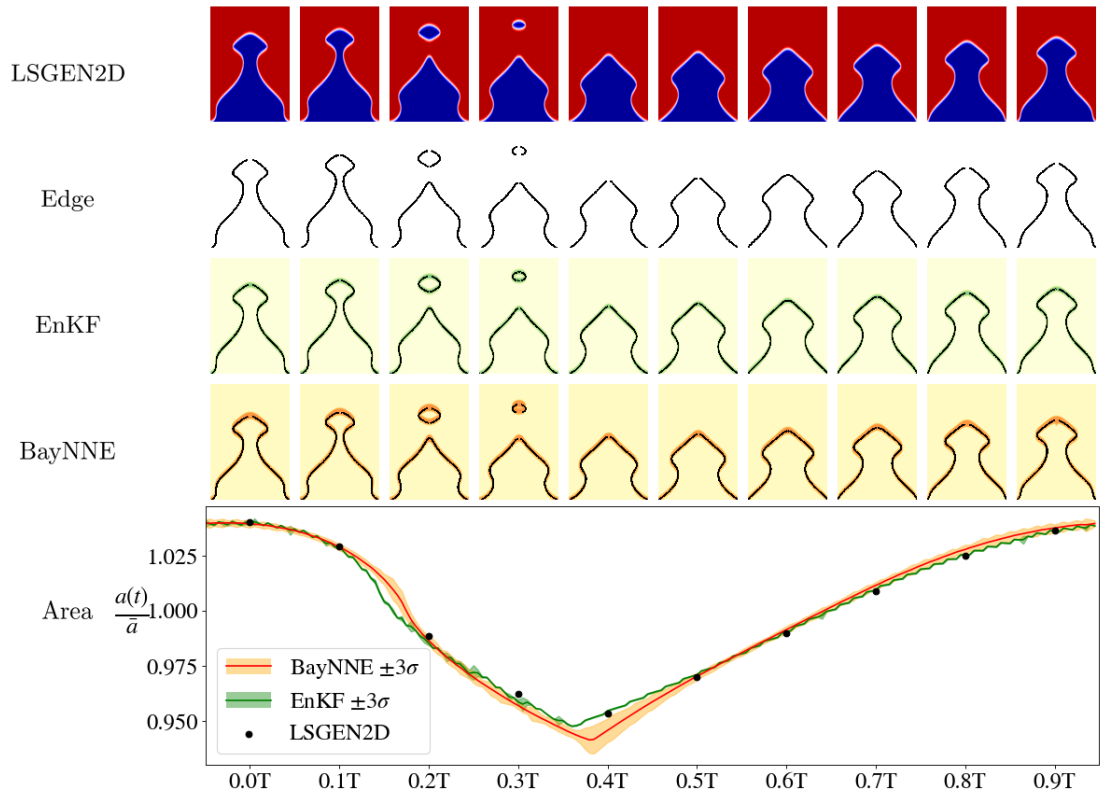
Fig. 2.7 Results of inference on the first forced twin experiment. The non-dimensional forcing frequency $fR/V\,(=\mathrm{St}/2\pi\beta)$ is 0.564. Top four rows: LSGEN2D flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Ground Truth	EnKF	BayNNE
β	steady & oscillating flames	9.833		9.403 ± 0.591
α		0.462		0.441 ± 0.038
$\mathcal{L}$		0.070		0.066 ± 0.005
St	oscillating flames only	21.05		19.83 ± 2.291
K		0.095	0.108 ± 0.004	0.094 ± 0.023
ϵ		0.474	0.401 ± 0.007	0.470 ± 0.038

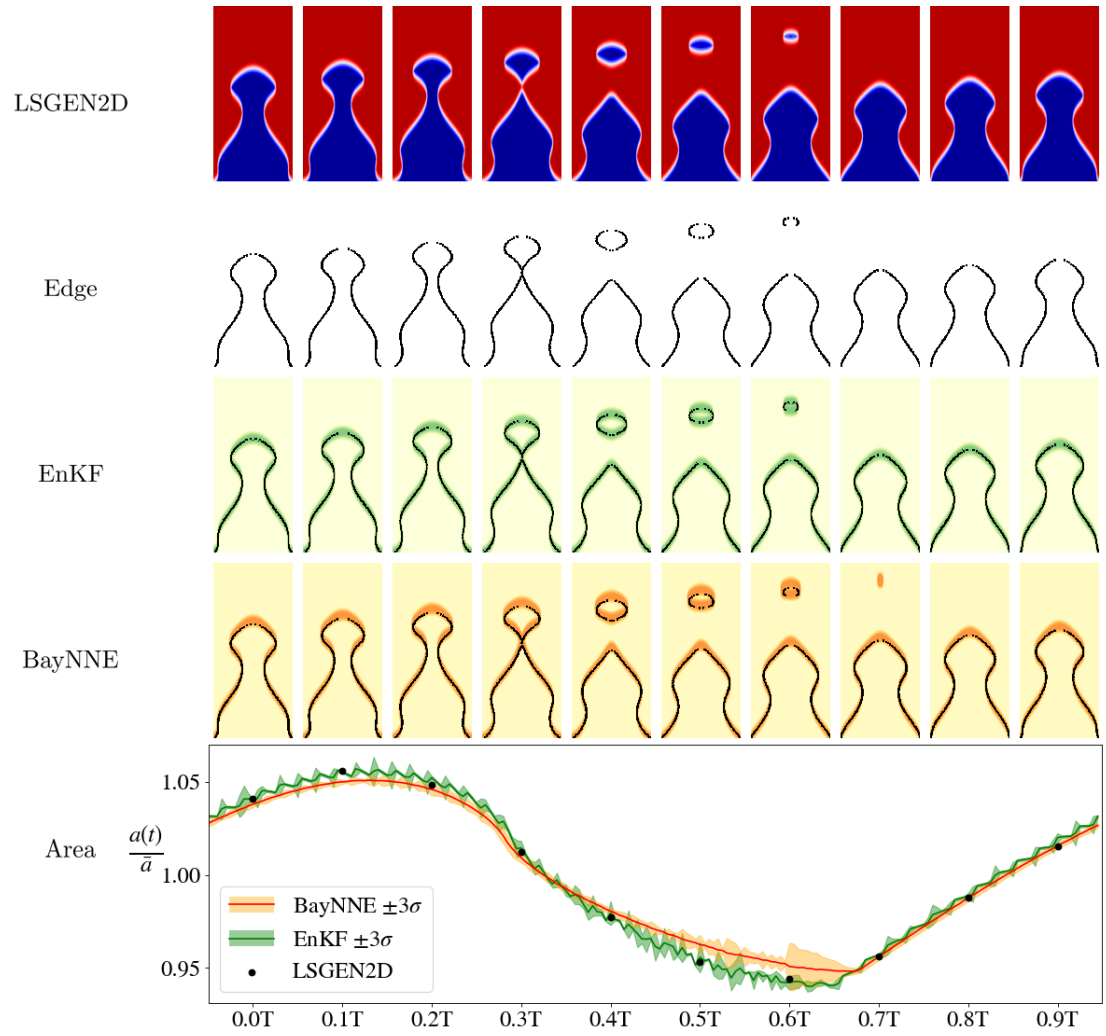
Fig. 2.8 Results of inference on the second forced twin experiment. The non-dimensional forcing frequency fR/V is 0.341. Top four rows: LSGEN2D flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Ground Truth	EnKF	BayNNE
β	steady & oscillating flames	4.389		4.741 ± 0.993
α		0.931		0.917 ± 0.139
$\mathcal{L}$		0.043		0.048 ± 0.015
St	oscillating flames only	17.03		17.39 ± 2.677
K		0.892	0.884 ± 0.001	0.967 ± 0.145
ϵ		0.214	0.216 ± 0.001	0.225 ± 0.049

Fig. 2.9 Results of inference on the third forced twin experiment. The non-dimensional forcing frequency fR/V is 0.614. Top four rows: LSGEN2D flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

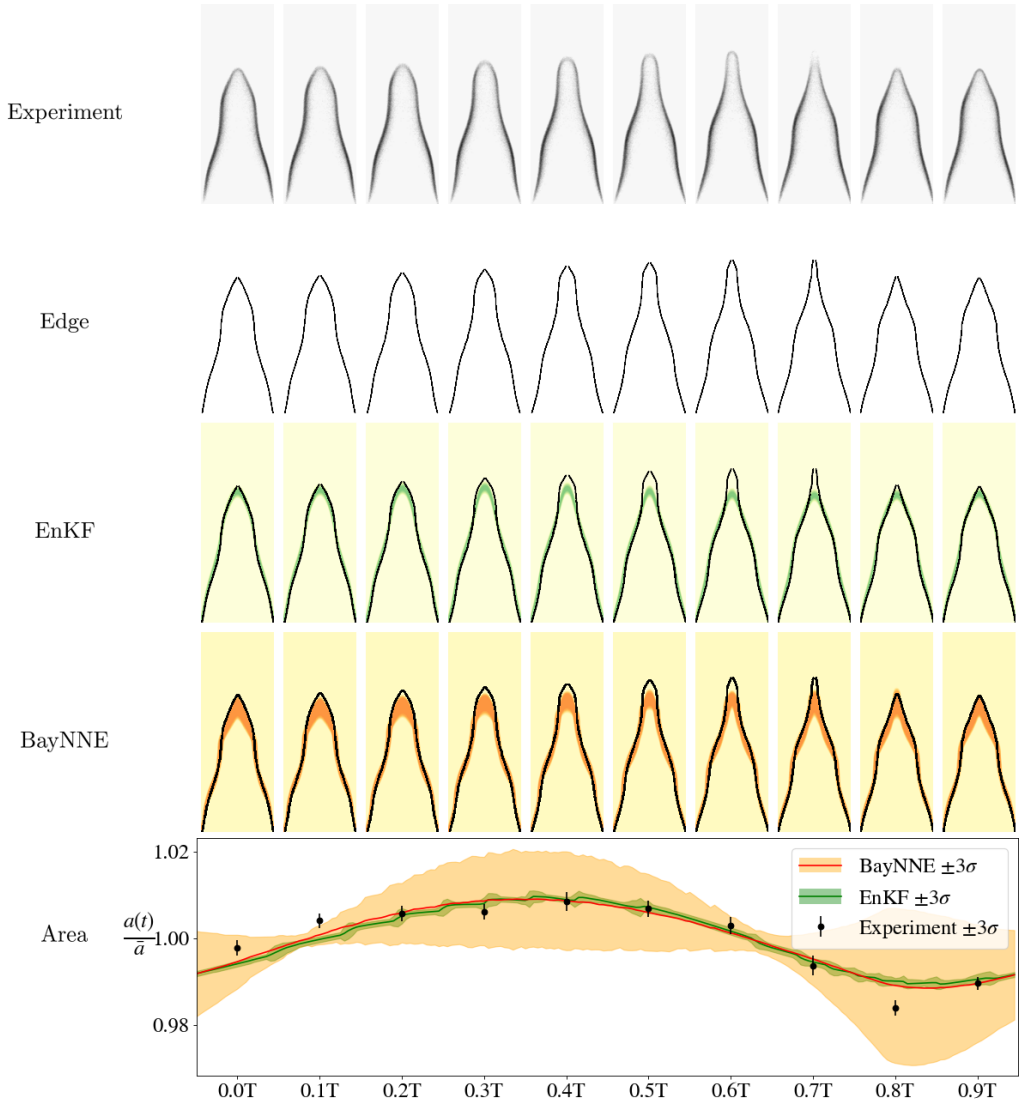
Parameter	Visible for	Ground Truth	EnKF	BayNNE
β	steady & oscillating flames	9.359		8.350 ± 1.053
α		0.992		0.938 ± 0.047
$\mathcal{L}$		0.053		0.044 ± 0.012
St	oscillating flames only	33.92		28.76 ± 6.125
K		0.681	0.677 ± 0.001	0.700 ± 0.070
ϵ		0.268	0.269 ± 0.001	0.248 ± 0.035

Fig. 2.10 Results of inference on the fourth forced twin experiment. The non-dimensional forcing frequency fR/V is 0.577. Top four rows: LSGEN2D flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.

2.6.4 Results on experimental data

Figs. 2.11 to 2.20 show the results of inference on ten different Bunsen flames. Both techniques produce good parameter estimates, in that the predicted flame shapes are in good agreement with the experiments. Furthermore, the BayNNE predicts normalised area variation curves at least as accurate as those predicted by the EnKF. However, the EnKF's predictions of K and ε are more confident than the BayNNE's predictions of all six parameters. The difference between the predicted uncertainties of the EnKF and BayNNE can be explained by the difference between the posterior distributions calculated by both techniques. The EnKF calculates $p(K,\varepsilon|\mathbf{x}_1,\mathbf{x}_2,\ldots,\mathbf{x}_{500})$: the probability distribution of the parameters K and ε given the location vectors $\mathbf{x}$ from all 500 images. For the BayNNE technique, the posterior $p(\mathbf{p}|\mathbf{z}_i) = p(\mathbf{p}|\mathbf{x}_i,\mathbf{x}_{i+1},\ldots,\mathbf{x}_{i+9})$ with the smallest total variance is chosen. This is the probability distribution of the parameters $\mathbf{p}$ given only ten image location vectors, as opposed to the 500 used by the EnKF. Therefore, it is expected that the BayNNE technique produces more uncertain parameter estimates.

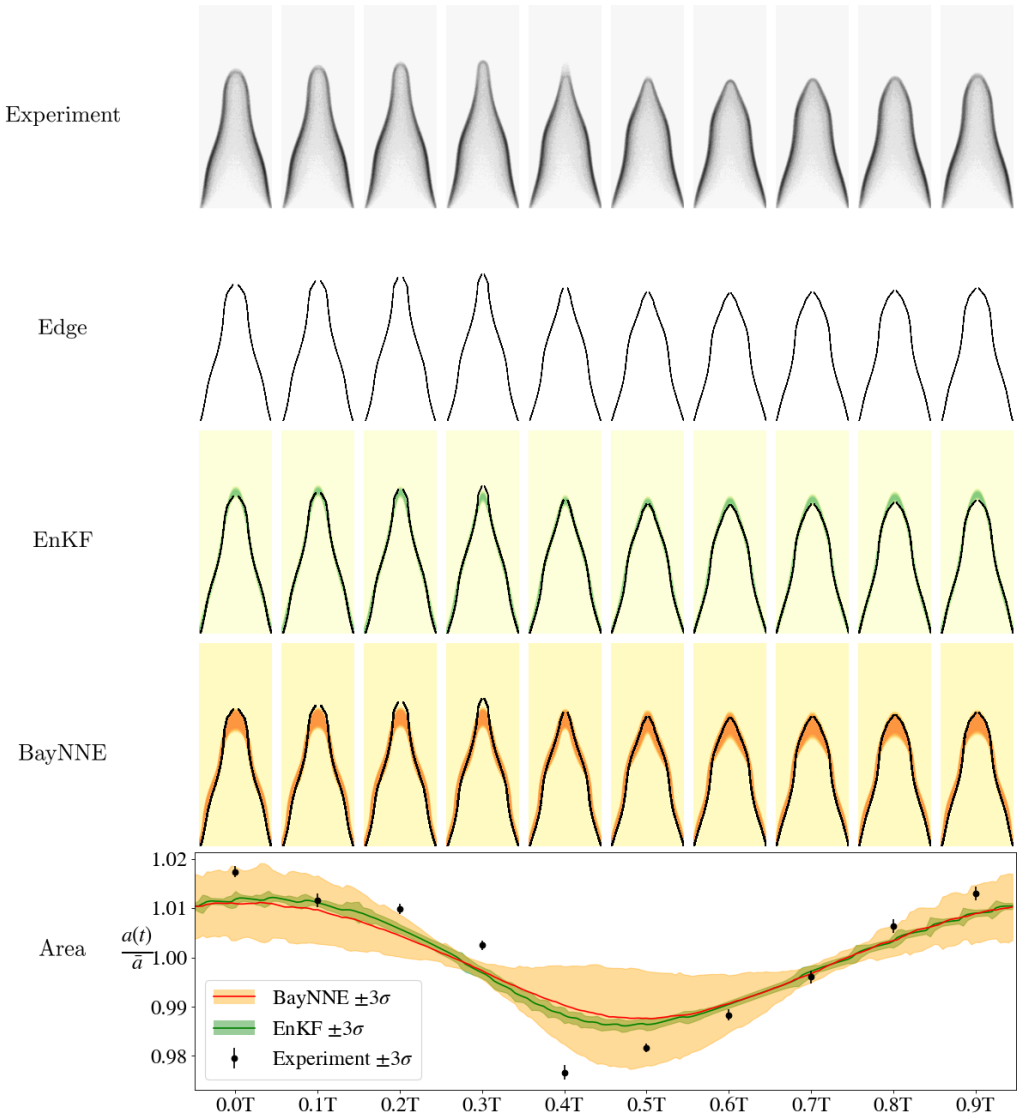
This raises the question as to whether it is possible to increase the certainty of the parameters by providing the BayNNE with more data. It is not possible to infer a posterior $p(\mathbf{p}|\mathbf{z}_1,\mathbf{z}_2,\ldots,\mathbf{z}_N)$ from the individual posteriors $p(\mathbf{p}|\mathbf{z}_i)$ without knowledge of the dependence between any two observations, $p(\mathbf{z}_i|\mathbf{z}_j)$. Two observation vectors are not independent, because the information gained from a first observation restricts the expected subsequent observations to a likely set of forced cycle states. One solution is to increase the number of location vectors $\mathbf{x}$ in each observation vector, which increases the computational cost of training the neural networks. Another solution is a recurrent neural network, which can have variable length sequences of data as inputs. However, a Bayesian ensembling method with recurrent neural networks has not yet been proposed.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	5.455		5.083 ± 0.403
α		0.189		0.162 ± 0.041
$\mathcal{L}$		0.031		0.022 ± 0.002
St	oscillating flames only	11.89		18.00 ± 4.565
K			1.428 ± 0.039	0.663 ± 0.274
ϵ			0.017 ± 0.001	0.052 ± 0.025

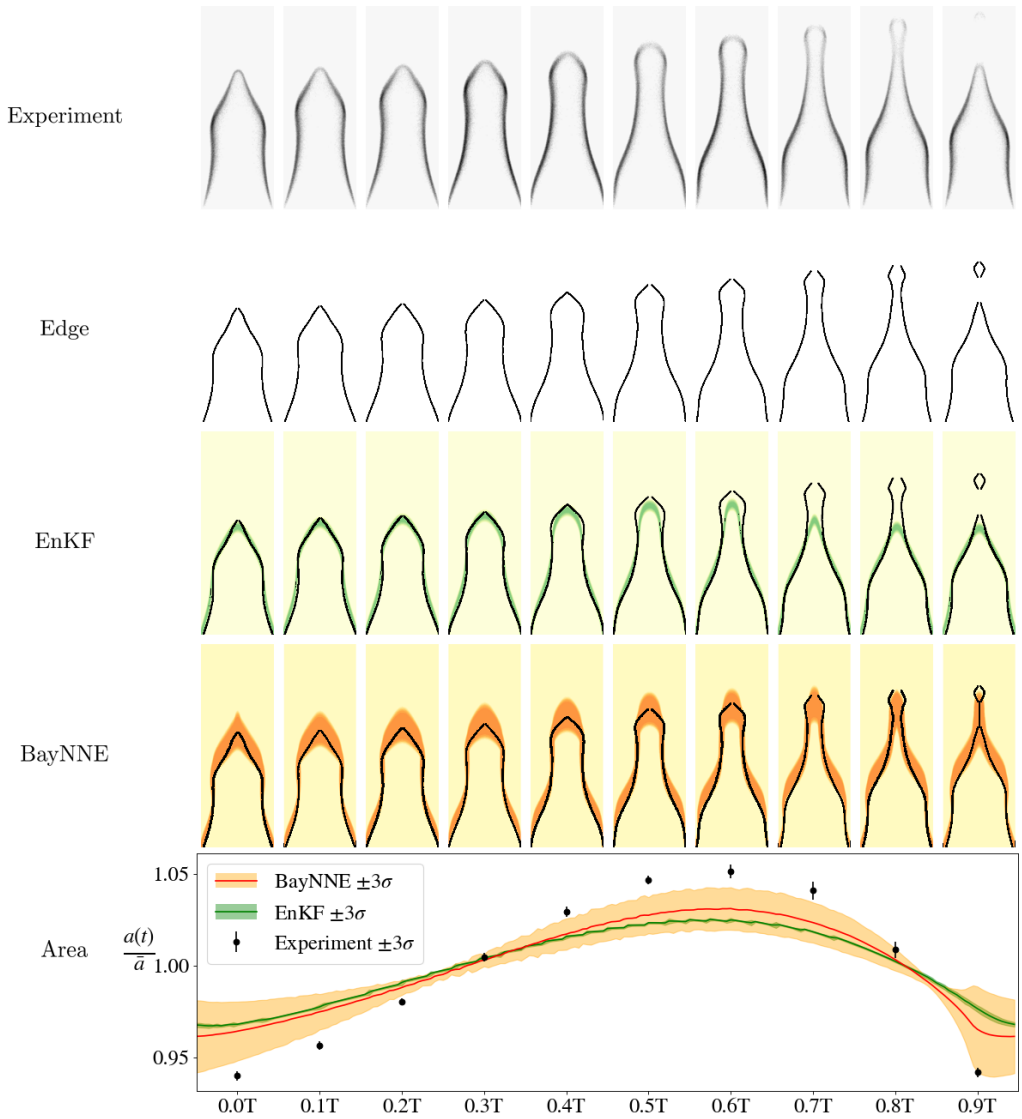
Fig. 2.11 Results of inference on the first Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	5.568		5.765 ± 0.364
α		0.333		0.319 ± 0.039
$\mathcal{L}$		0.036		0.038 ± 0.007
St	oscillating flames only	11.465		20.27 ± 4.785
K			1.230 ± 0.030	0.649 ± 0.308
ϵ			0.022 ± 0.002	0.059 ± 0.026

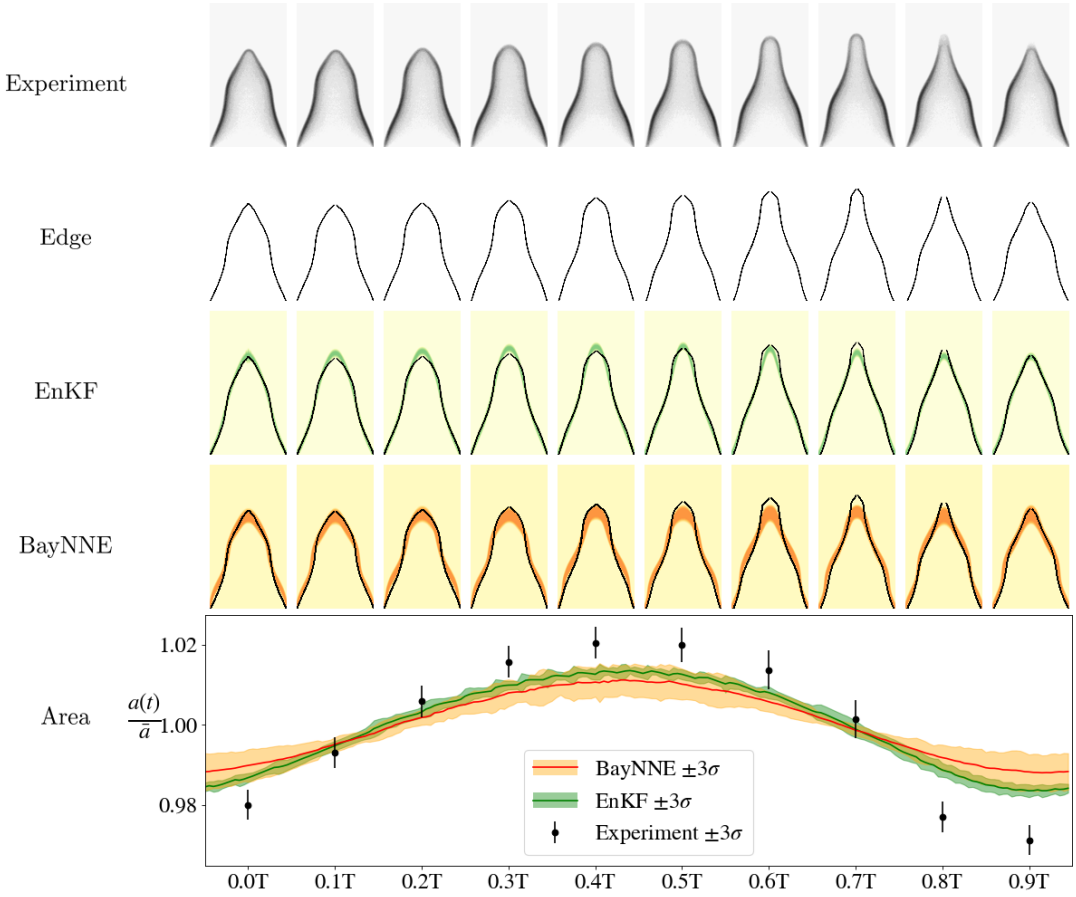
Fig. 2.12 Results of inference on the second Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	6.788		6.774 ± 0.833
α		0.356		0.207 ± 0.091
$\mathcal{L}$		0.037		0.031 ± 0.012
St	oscillating flames only	9.768		17.12 ± 3.746
K			1.415 ± 0.013	0.533 ± 0.124
ϵ			0.055 ± 0.001	0.193 ± 0.067

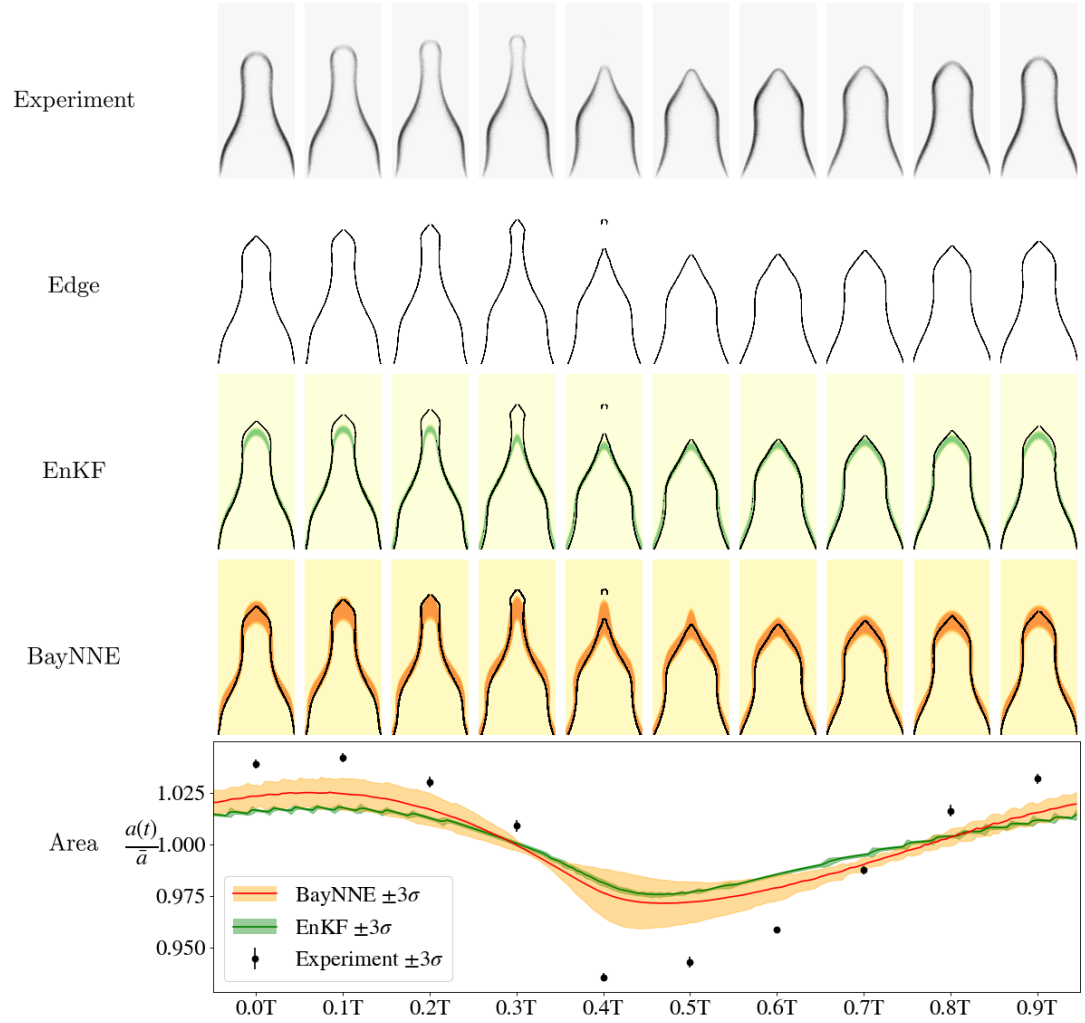
Fig. 2.13 Results of inference on the third Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	3.739		4.081 ± 0.206
α		0.387		0.327 ± 0.043
$\mathcal{L}$		0.045		0.051 ± 0.008
St	oscillating flames only	9.660		20.40 ± 4.393
K			1.035 ± 0.032	0.630 ± 0.173
ϵ			0.032 ± 0.002	0.086 ± 0.033

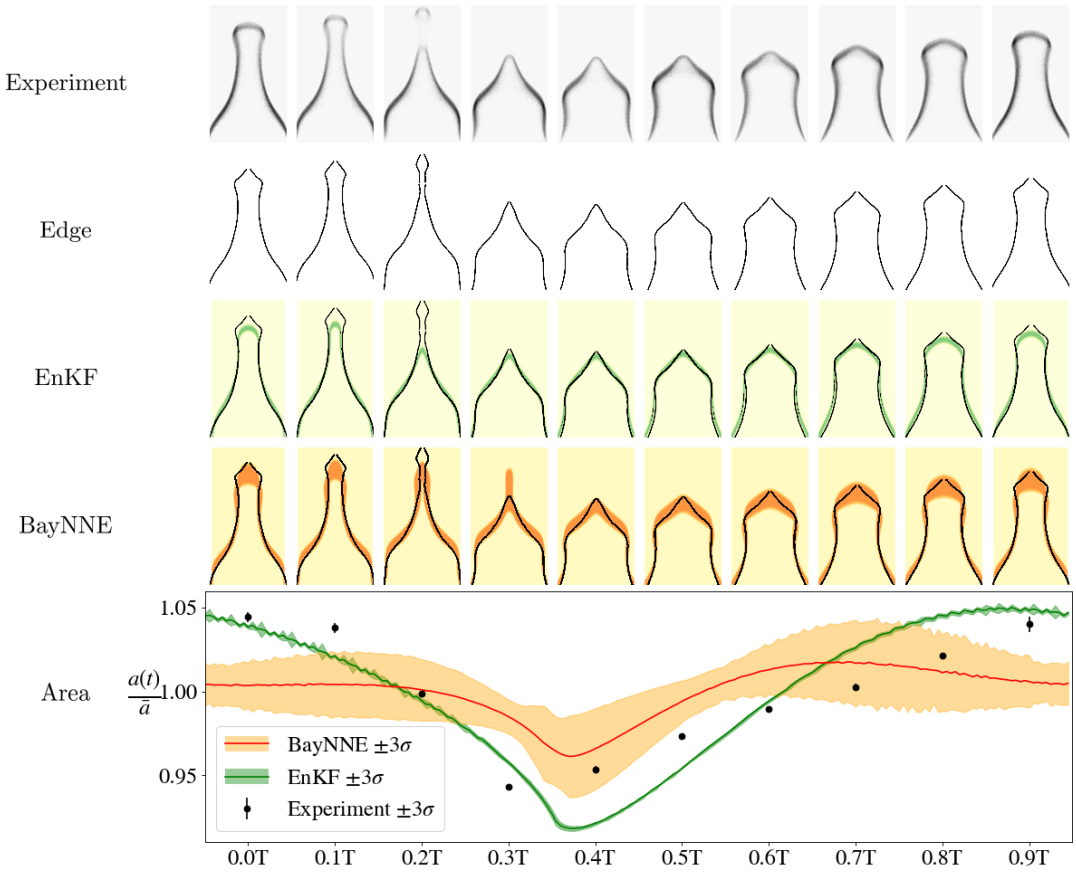
Fig. 2.14 Results of inference on the fourth Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	4.558		4.838 ± 0.340
α		0.315		0.198 ± 0.052
$\mathcal{L}$		0.039		0.038 ± 0.010
St	oscillating flames only	9.475		16.99 ± 3.199
K			1.558 ± 0.016	0.571 ± 0.110
ϵ			0.055 ± 0.001	0.167 ± 0.044

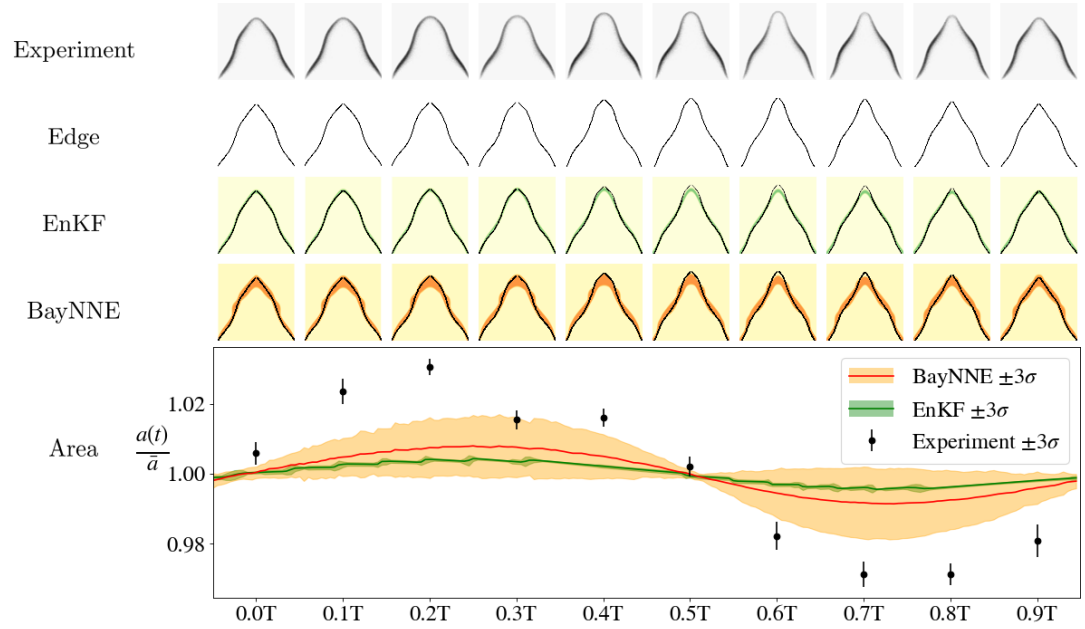
Fig. 2.15 Results of inference on the fifth Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	3.584		3.978 ± 0.416
α		0.194		0.139 ± 0.081
$\mathcal{L}$		0.031		0.040 ± 0.014
St	oscillating flames only	6.367		13.12 ± 2.116
K			1.521 ± 0.009	0.523 ± 0.091
ϵ			0.109 ± 0.002	0.322 ± 0.062

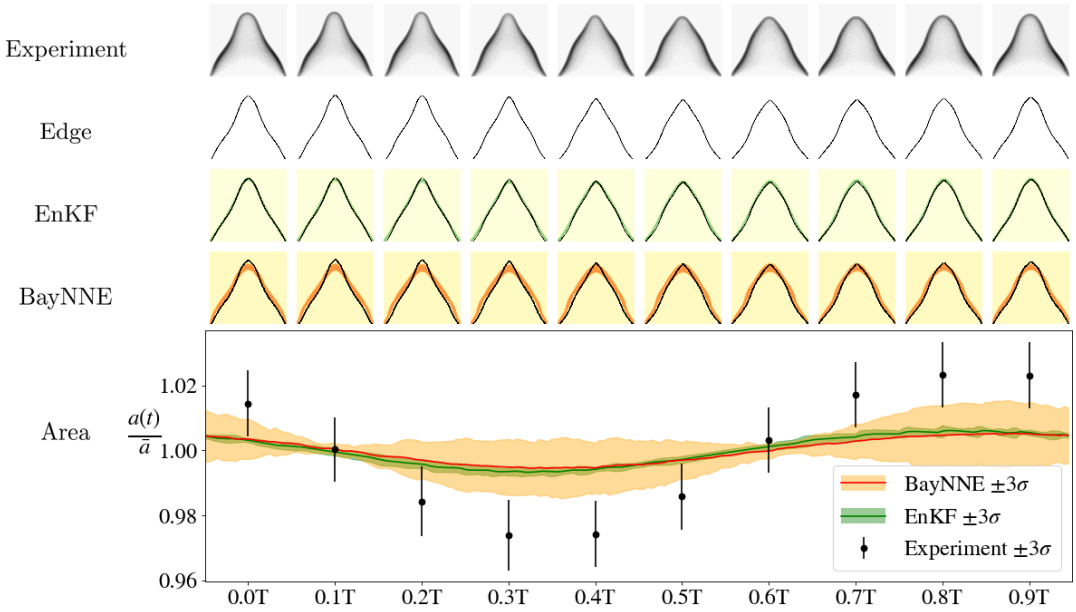
Fig. 2.16 Results of inference on the sixth Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	2.001		2.041 ± 0.063
α		0.112		0.182 ± 0.082
$\mathcal{L}$		0.031		0.036 ± 0.017
St	oscillating flames only	6.788		12.01 ± 4.793
K			2.107 ± 0.062	0.631 ± 0.434
ϵ			0.024 ± 0.002	0.097 ± 0.067

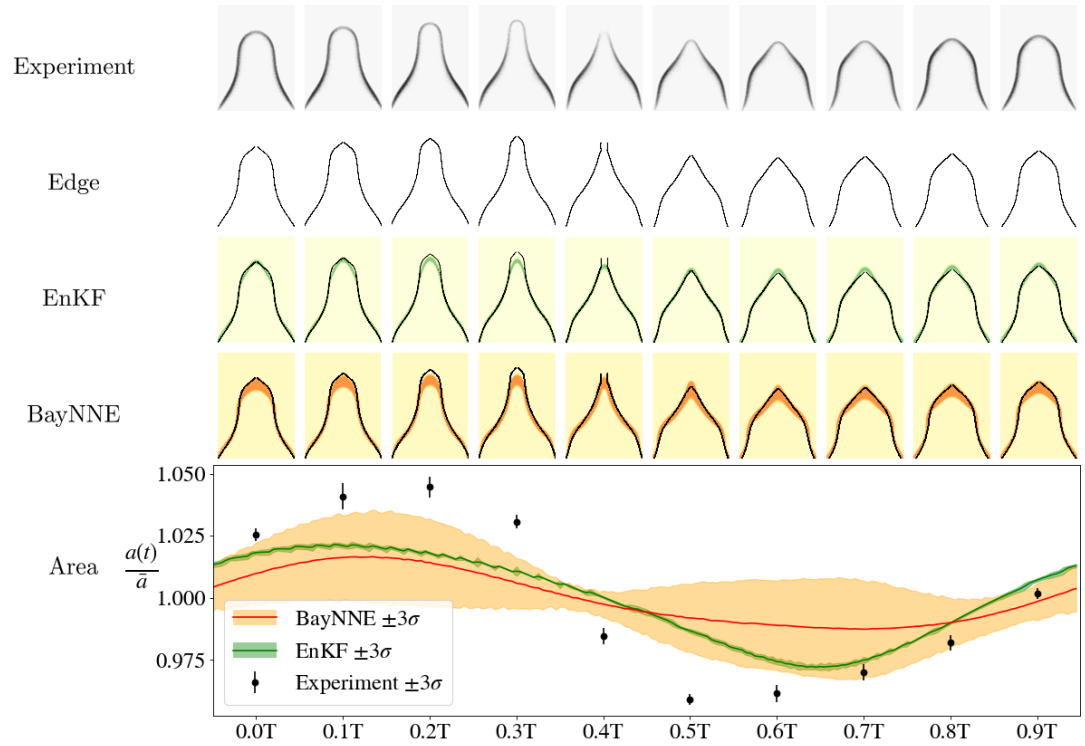
Fig. 2.17 Results of inference on the seventh Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	1.885		2.048 ± 0.071
α		0.285		0.332 ± 0.094
$\mathcal{L}$		0.041		0.060 ± 0.020
St	oscillating flames only	6.850		12.85 ± 5.200
K			1.257 ± 0.073	0.650 ± 0.330
ϵ			0.017 ± 0.003	0.098 ± 0.049

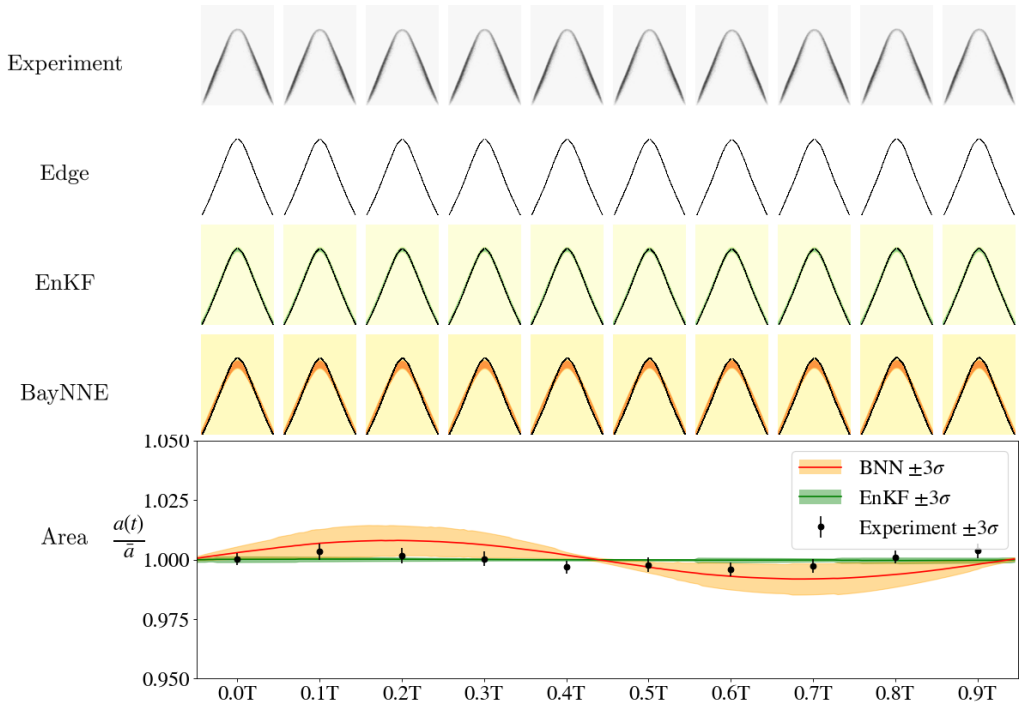
Fig. 2.18 Results of inference on the eighth Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	2.758		2.759 ± 0.171
α		0.241		0.138 ± 0.072
$\mathcal{L}$		0.044		0.035 ± 0.009
St	oscillating flames only	6.615		11.05 ± 1.653
K			2.042 ± 0.025	0.484 ± 0.152
ϵ			0.073 ± 0.002	0.231 ± 0.074

Fig. 2.19 Results of inference on the ninth Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.



Assimilated parameters

Parameter	Visible for	Brute force & Cantera	EnKF	BayNNE
β	steady & oscillating flames	2.758		2.672 ± 0.089
α		0.241		0.243 ± 0.032
$\mathcal{L}$		0.044		0.040 ± 0.007

Fig. 2.20 Results of inference on the steady Bunsen flame experiment. Top four rows: Bunsen flame whose parameters are to be estimated, the detected flame front, predicted flames using EnKF estimated parameters and predicted flames using BayNNE estimated parameters. The green and orange bands are regions of high likelihood of the flame in the EnKF and BayNNE plots. Fifth row: area variations over one period. Bottom: table of assimilated parameters.

2.7 Conclusions

To summarise, this chapter proposed a Bayesian machine learning method for assimilating data from a Bunsen flame experiment into a physics-based model of the flame. The method uses a heteroscedastic Bayesian neural network ensemble (BayNNE) trained on a library of simulated forced cycle flame fronts. Both generating the library of simulated flame fronts and training the neural networks on the library are computationally expensive steps. However, once the BayNNE has been trained on the forced cycle library it can be used to reliably infer all six parameters of the flame front model based on ten consecutive snapshots of the Bunsen flame experiments in $O(10^{-3})$ seconds. If more snapshots are provided, this method finds the sequence of ten snapshots that minimises the uncertainty in the parameters. The method is compared with the ensemble Kalman filter (EnKF) method, another data assimilation method which has been demonstrated to work well on this data (both in this chapter and previous publications, such as [47]). The EnKF requires hours to infer the parameters of each Bunsen flame. The trained BayNNE provides similarly accurate parameter estimates at a fraction of the computational cost. Because the BayNNE infers parameters and their uncertainties in milliseconds, it can be used in real-time tasks, such as the control of thermoacoustic instabilities.

With the BayNNE method demonstrated on data from a Bunsen flame, the next chapter will demonstrate the method on data from a version of the Volvo combustor, which is more complex than the Bunsen burner. Possible improvements to the method, for example incorporating recurrent neural network architectures to address the limitations discussed in the previous section, will also be investigated.

Chapter 3

Parameter inference for a kinematic model of a bluff-body-stabilised flame

This chapter is based upon the following conference publications, which were peer-reviewed by at least 1 reviewer:

- M. L. Croci, U. Sengupta and M. P. Juniper, "Real-time parameter inference of non-linear bluff-body-stabilized flame models using Bayesian neural network ensembles", in *Symposium on Thermoacoustics in Combustion: Industry meets Academia (SoTiC 2021)*, (Munich, Germany), 2021 [90];
- M. L. Croci, U. Sengupta and M. P. Juniper, "Bayesian inference in non-linear flame models", in *Deep Learning and Inverse Problems workshop at the 35th Conference on Neural Information Processing Systems (NeurIPS)*, (Virtual), 2021 [91].

The experimental data is supplied by Mr Brett Rankin (United States Air Force Office for Scientific Research). As in the previous chapter, the neural network method is that which was first proposed in [76] and LSGEN2D is implemented as in [84]. The heat release model was primarily implemented by Professor Matthew Juniper, and the code for calculating the normalised flame surface area was written by Mr Alexandros Kontogiannis. The implementation of, and results from, the Helmholtz solver are due to Mr Ekrem Ekici.

3.1 Introduction

In the previous chapter, a Bayesian machine learning method was used to infer the parameters of a physics-based model of a Bunsen flame, as a precursor to applying the method to more complex burner rigs. In this chapter, a G-equation model of a version of the Volvo

burner [92, 93, 34, 35] is proposed. Whereas the Bunsen flame which was studied in the previous chapter was fully observed, this version of the Volvo burner has an observation window through which the flame is only partially observed. This means quantifying the uncertainty in any inferred parameters is especially important, as we do not have information on the downstream behaviour of the flame.

A library of partially-observed, simulated flame fronts is generated using the model, from which a heteroscedastic Bayesian neural network ensemble is trained to infer the model parameters from 10 flame front snapshots. The trained ensemble is then used to infer the *G*-equation model parameters that best fit experimental data from the Volvo burner. Due to the abrupt changes in behaviour observed in the turbulent flame data, the ensemble Kalman filter method cannot be used to assimilate the data into a *G*-equation model. By re-simulating the flames using the inferred parameters, distributed $n-\tau$ models are calculated which can be used in a Helmholtz solver to calculate eigenmodes and eigenfrequencies of the thermoacoustic instability of the burner.

3.2 The Volvo burner

3.2.1 Experiment

This data is taken from experiments performed on a version of the Volvo burner shown schematically in Fig. 3.1. Premixed air and propane flow into the burner and are burned by a flame stabilised on a triangular bluff body with side length $D = 3.8$ cm. The depth of the duct is $4D$. Images of the flame such as those in Fig. 3.2(a) are recorded at 10 kHz using OH planar laser-induced fluorescence (OH PLIF) through a window $3D$ tall and $3.4D$ wide. A sequence of 7998 images each with resolution 69×58 pixels is recorded. Additionally, the horizontal and vertical velocities within the observation window at a resolution of 69×58 pixels are recorded at each timestep. As the air-fuel mixture flows through the burner, vortices are shed periodically from the bluff body. These vortices cause wrinkling and cusping of the flame front. For full details of the Volvo experiment, the reader is referred to: [34, 35].

The Volvo flame images are processed and the flame front is extracted as a radial location y, which is a singularly-valued function of the axial co-ordinate x: $y = f(x)$. Although the flame shows greater wrinkling than the Bunsen flame, it does not wrinkle back onto itself and so a $y = f(x)$ discretisation of the flame front is still possible. Starting from an OH PLIF intensity image, the bottom half is discarded because the *G*-equation model assumes a symmetric flame front. Furthermore, the rightmost 9 columns of data are discarded from each image due to the fact that the flame front is occasionally too faint to be detected here.

The flame front is found by thresholding the magnitude of the gradient vector, as shown in 3.2*(b)*. Next, splines with 10 knots are used to smooth $y(x)$. Each flame image is therefore converted into a 60×1 vector of flame front y locations $\mathbf{y}$. The x coordinates are the same for all flames, so are discarded. Observation vectors $\mathbf{z}$ are created by appending 10 consecutive $\mathbf{y}$ vectors together. All 7998 images taken from the Volvo burner experiment are processed in this way.

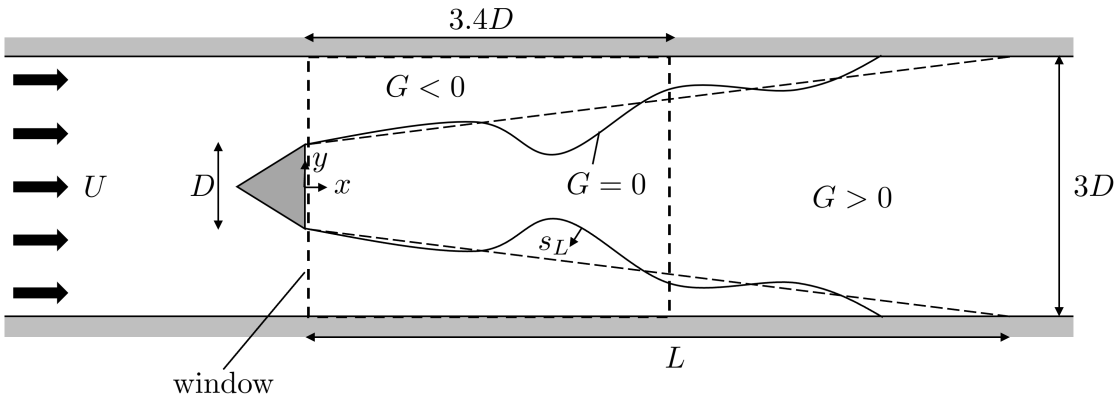


Fig. 3.1 Diagram of the Volvo burner rig and G-equation model of the flame. A flame burns premixed air and propane stabilised on a triangular bluff body. In the G-equation model, the flame front is represented by the $G = 0$ contour (or level-set) of a continuous scalar field $G(x,y,t)$. Unburnt and burnt gases are regions where $G < 0$ and $G > 0$ respectively. The flame front travels normal to itself into the unburnt gases with speed s_L. The flame front advects under the prescribed velocity field, which comprises continuity-obeying velocity perturbations $u'(x,t)$ and $v'(x,y,t)$ superimposed onto a steady base flow U.

3.2.2 Non-dimensionalisation

The characteristic scales of the problem are the side length D of the bluff body, the length of the unstretched unforced flame, L, the spatial frequency U/L where U is mean base flow speed and the excitation angular frequency $2\pi f$. The Strouhal number is the ratio of the spatial time scale to the excitation time scale (where the time scales are the reciprocal of the frequencies) $\mathrm{St} = 2\pi fL/U$. By defining the aspect ratio of the unstretched, unforced flame $\beta = L/D$, the Strouhal number may be written equivalently as $\mathrm{St} = 2\pi f\beta D/U$.

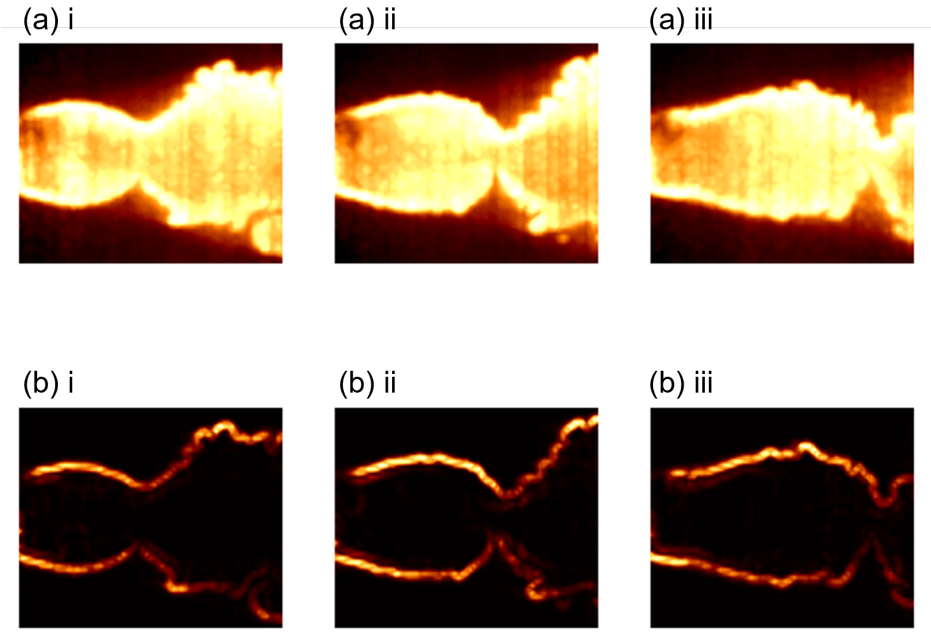


Fig. 3.2 *(a)* Plots of OH planar laser-induced fluorescence (OH PLIF) intensity images of the flame taken through the observation window of the rig. *(b)* Plots of the magnitude of the gradient vector of OH intensity, which is taken to be the flame front [34].

3.3 The G-equation model of the Volvo burner

3.3.1 The G-equation model

The flame front is assumed to be a thin boundary between unburnt and burnt gases (see Fig. 3.1). The flame travels normal to itself into the unburnt gases with laminar flame speed s_L, which depends on the gas composition. The velocity in the burnt gases does not affect the flame kinematics. The unburnt and burnt gases are assumed to travel with velocity $\mathbf{u}(x,y,t)$. Under these assumptions, the flame front is modelled by the $G(x,y,t) = 0$ contour (or level-set) of a continuous scalar field G whose motion is governed by the G-equation:

$$\frac{\partial G}{\partial t} + \mathbf{u} \cdot \nabla G = s_L |\nabla G|. \tag{3.1}$$

The flame speed s_L is equal to the unstretched (adiabatic) flame speed s_L^0 but is no longer a function of the local flame curvature. We make this assumption because the flame is turbulent, as opposed to the laminar Bunsen flame studied in the previous chapter. The flow velocity field $\mathbf{u}$ is assumed to comprise a constant and uniform base flow U and super-imposed

continuity-obeying velocity perturbations $u'(x,t)$ and $v'(x,y,t)$:

$$\frac{\mathbf{u}(x,y,t)}{U} = \big(1+u'(x,t)\big)\,\mathbf{i} + v'(x,y,t)\mathbf{j}, \tag{3.2}$$

$$u'(x,t) = \varepsilon\left(\frac{x}{D}\right)^{\gamma}\sin\left(\mathrm{St}\left(K\left(\frac{x}{D}\right)-t\right)\right), \tag{3.3}$$

$$\begin{aligned} v'(x,y,t) =& \varepsilon\gamma\left(\frac{x}{D}\right)^{\gamma-1}\left(\frac{y}{D}\right)\sin\left(\mathrm{St}\left(K\left(\frac{x}{D}\right)-t\right)\right) \\ &- \mathrm{St}K\varepsilon\left(\frac{x}{D}\right)^{\gamma}\left(\frac{y}{D}\right)\cos\left(\mathrm{St}\left(K\left(\frac{x}{D}\right)-t\right)\right), \end{aligned} \tag{3.4}$$

where U is a characteristic speed, ε is a non-dimensional amplitude, St is the Strouhal number of the flame with characteristic length D, excitation frequency f, and nominal aspect ratio β: $\mathrm{St} = 2\pi f\beta D/U$, and K is the ratio of the characteristic speed U to the perturbation phase speed. The parameter γ is introduced to the flame perturbation model to allow for the horizontal velocity perturbations to increase in size with distance from the flame holder, which is the qualitative behaviour observed in the experiment. To make the *G*-equation model quantitatively accurate, the parameters $K, \varepsilon, \gamma, \mathrm{St}$ and β must be tuned to fit an observed flame shape over the observation window for which data is supplied.

3.3.2 Transients in the model and the data

It can be shown that transients occur in the *G*-equation model of the Volvo burner, see A.3. The transient dissipation time t_c is:

$$t_c = \frac{(1+\beta^2)}{\beta}\frac{D}{U} \quad . \tag{3.5}$$

Therefore, according to a linear perturbation analysis, the number of periods, p_c, it takes for the *G*-equation model of the system to reach the forced cycle with forcing frequency f is:

$$\begin{aligned} p_c &= \frac{t_c}{T} \\ &= \frac{(1+\beta^2)}{\beta}\frac{fD}{U} \quad . \end{aligned} \tag{3.6}$$

We observe transient behaviour in the experimental data: although the operating conditions (fuel flow rate and equivalence ratio) are constant, the flame exhibits large amplitude oscillations at a frequency of $120-125$ Hz between equally long periods of

aperiodic behaviour. This makes it difficult for traditional data assimilation methods such as the ensemble Kalman filter to converge. Instead, by assimilating the data 10 images at a time, the Bayesian neural network ensemble can infer different values for the parameters depending on what the flame behaviour is within the 10 images.

3.4 Library of forced flame simulations

A library of flame front locations is created with the flame front model at known parameter values $K, \varepsilon, \gamma, \mathrm{St}, \beta$ and f/f_s in the same format as the observation vectors, $\mathbf{z}$. The parameter values are sampled using quasi-Monte Carlo sampling to ensure good coverage of the parameter space. The parameters are sampled from the ranges in Table 3.1. The values of St are calculated using $\mathrm{St} = 2\pi f D\beta/U$. The parameters are sampled 2300 times, normalised to between -1 and 1 and recorded in target vectors $\{\mathbf{p} = [K, \varepsilon, \gamma, \mathrm{St}, \beta]\}$. Fewer parameter samples are required compared to the Bunsen flame experiment due to the flame being only partially observed.

The G field is iterated forward in time for a duration of t_c seconds (Eq. 3.5) (p_c periods, Eq. 3.6, where the period is $1/f$). This allows any transient flame behaviour to die away. For parameters sampled according to Table 3.1, p_c varies between 0.85 and 4.88. Then the G field is iterated forward for a further period. The value of the G field at $N_T = 200$ equally spaced steps, $dt = 1/fN_T$, within this period is recorded: $\{G_1, G_2, \ldots, G_{N_T}\}$. For each G field in the sequence $\{G_i\}$, the flame front $y = f(x)$ is extracted from the $G = 0$ contour for all x in the range of the experiment observation window. The flame front y coordinates are recorded in a column vector $\mathbf{y}_i$. The resulting sequence $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{N_T}\}$ represents the flame front position at N_T equally-spaced timesteps in the period. Next, the sum of squared errors between the first and last G fields in the sequence $\{G_i\}$ is calculated. As before, the frame sequence $\{G_i\}$ is deemed to be a valid forced cycle only if this sum of squared errors falls below a tolerance tol, otherwise it is discarded along with its target parameters. The frame rate of the forced cycle sequence is N_T/T which is not, in general, equal to the frame rate of the experimental data ($f_s = 10^4$ Hz). To create a sequence of 10 simulated flame fronts with the same frame rate as the experimental data starting from a timestep t_1, we calculate the ratio $\Delta = N_T f/f_s$ and select the sequence $\{\mathbf{y}_{t_1}, \mathbf{y}_{t_1+\Delta}, \ldots, \mathbf{y}_{t_1+9\Delta}\}$. Any flame coordinates which are outside of the experimental observation window are discarded. The vectors in this sequence are appended together to make a column vector $\mathbf{z}$. This is repeated for all N_T of the starting timesteps $t_1, t_2 \ldots t_{N_T}$, and for $P = 2300$ combinations of parameters sampled from the ranges shown in Table 3.1. The result is a library of $PN_T = 4.6 \times 10^5$ observation - parameter pairs. This is summarised in algorithm 2.

As with the algorithm to generate a library of simulated Bunsen flame fronts, the algorithm can be trivially parallelised. One loop of algorithm 2 takes up to one hour, depending on the transient dissipation time t_c. An upper bound for the library generation's computational cost is $2300 \times 1 = 2.3 \times 10^3$ CPU core-hours. This takes approximately seven hours when at the limit of maximum concurrent CPU core usage.

Table 3.1 Parameters of the *G*-equation model that are varied in this chapter and the range over which they are varied in order to generate the simulated flame front library.

Parameter ranges	Description
$0.5 \leq K \leq 2.0$	Ratio of mean base flow speed to perturbation phase speed
$0.0 < \varepsilon \leq 0.5$	Amplitude of the vertical velocity perturbation
$0.0 \leq \gamma \leq 3.0$	Perturbation growth exponent
$4.0 \leq \beta \leq 8.0$	Aspect ratio of the unperturbed flame
$0.03 \text{ m} \leq D \leq 0.04 \text{ m}$	Bluff-body side length
$10 \text{ m/s} \leq U \leq 15 \text{ m/s}$	Mean base flow speed
$100 \text{ Hz} \leq f \leq 150 \text{ Hz}$	Excitation frequency (self-excited)

3.5 Parameter inference for the *G*-equation model of the Volvo burner using BayNNEs

The heteroscedastic Bayesian neural network ensemble method is unchanged compared to that described in the previous chapter. The architecture of each neural network in the ensemble is changed in light of the differences in the size of the input data vectors compared to that in the previous chapter. First, the each neural network in the ensemble's input and hidden layers are 600 units wide, and each output layer is 5 units wide. For full hyperparameter details, see Table 3.2.

At this stage, the alternative neural network architectures that were explored were: the recurrent neural network (RNN, [94]), the long short-term memory network (LSTM, [66]), the gated recurrent unit (GRU, [95]) and the transformer [96]. These architectures were developed for sequence to sequence tasks, such as machine translation, where for some input sequence an output sequence is produced. For our purposes the input sequence would be the flame images and the output sequence would be parameter predictions for each input image. Although each of these architectures can be used for regression tasks, no Bayesian uncertainty quantification variant has been developed. I attempted to extend the Bayesian neural network ensemble method theory to the RNN architecture but without success: it is

Algorithm 2: Generate a library of simulated, partially observed Volvo flame fronts with known parameters.

Data: Number of parameter samples P, number of timesteps per period N_T, frame rate f_s, tolerance tol.
Result: Library of pairs $(\mathbf{p}_i, \mathbf{z}_i)_{i=1}^{PN_T}$
for $1 \leq p \leq P$ **do**
  $K, \varepsilon, \gamma, f, \beta, D, U \leftarrow sample()$; ▷ Sample parameters from Tab. 3.1.
  $\mathrm{St} \leftarrow 2\pi f \beta D / U$;
  $\mathbf{p} \leftarrow [K, \varepsilon, \gamma, \mathrm{St}, \beta]$; ▷ Record the target parameters.
  $T \leftarrow 1/f$;
  $t_c \leftarrow D(1+\beta^2)/\beta U$; ▷ Calculate the transient duration (see Appendix A.3).
  $t \leftarrow 0$;
  $dt \leftarrow T/N_T$;
  $LSGEN2D.initialise(\mathbf{p})$; ▷ Initialise LSGEN2D with target parameters.
  while $t \leq t_c$ **do**
    ▷ Iterate the G-equation forward in time for the duration of the transient period.
    $LSGEN2D.iterate()$;
    $t \leftarrow t + dt$;
  end
  while $t_c \leq t \leq t_c + T$ **do**
    ▷ Once the transient period is over, iterate for a further period and record the flame front y coordinates.
    $LSGEN2D.iterate()$;
    $G_t \leftarrow LSGEN2D.G$;
    $\mathbf{y}_t \leftarrow G_t.extractFrontWithinWindow()$;
    $t \leftarrow t + dt$;
  end
  if $|G_{t_c+T} - G_{t_c}| < tol.$ **then**
    ▷ Verify that the convergence of the forced cycle by comparing the first and last G fields in the cycle.
    $\Delta \leftarrow N_T f / f_s$;
    for $t_0 = 1 : N_T$ **do**
      $\mathbf{z} \leftarrow concatenate(\mathbf{y}_{t_0}, \mathbf{y}_{t_0+\Delta}, \ldots, \mathbf{y}_{t_0+9\Delta})$; ▷ Create observation vector.
      $library.addPair(\mathbf{p}, \mathbf{z})$; ▷ Save the parameter - observation pair.
    end
  end
end

not clear that the strong assumptions made in [97] about correlations between the weights of the hidden layers must necessarily hold between recurrent layers. Ensembling of each

of these architectures can still be performed to provide qualitative uncertainty estimates, however.

Another architectural choice is the choice of input layer(s) to the neural networks. Convolutional neural network (CNN) layers [98, 99] were developed to allow images, stored as matrices of pixel intensities, to be used as inputs into neural networks directly. For our purposes, this would avoid the pre-processing of the flame images into flame front coordinates. However, the *G*-equation model of the flame front models the flame as a thin boundary between unburnt and burnt gases. An $N \times N$ image of the flame generated with this model would only have $O(N)$ pixels of useful information (the thin line representing the flame), whereas the image contains N^2 pixels. Training the CNNs on a library of simulated flame images would be more computationally expensive than training the fully-connected neural networks used earlier in the chapter.

Table 3.2 Hyperparameter settings used for neural network training.

Hyperparameter	Value
Training	
Train-test split	80:20
Batch size	256
Epochs	100
Optimiser	Adam
Learning rate	10^{-4}
Architecture	
Input units	600
Hidden layers	4
Units per hidden layer	600
Output layers	2
Units per output layer	5
Ensemble size	20

3.6 The heat release model

The heat release model is based on a distributed $n-\tau$ model proposed in [100, 101]. The local heat release rate perturbation per unit volume $\tilde{q}_h$ is assumed to be proportional to the horizontal velocity perturbation $\tilde{u}(\mathbf{x},t)$, where $\mathbf{x} = (x,y)$, integrated over a measurement region $w(\mathbf{x})$ some time $\tau(\mathbf{x})$ earlier:

$$\tilde{\dot{q}}_h(\mathbf{x},t) = \eta h(\mathbf{x}) \int_\Omega w(\hat{\mathbf{x}}) \tilde{u}(\hat{\mathbf{x}}, t-\tau(\mathbf{x})) d\hat{\mathbf{x}} \,, \tag{3.7}$$

where $h(\mathbf{x})$ is a distribution of heat release in space which integrates to one over the whole domain ($\int_\Omega h(\mathbf{x})d\mathbf{x} = 1$), $w(\mathbf{x})$ is a distribution of measurement in space which integrates to one over the whole domain ($\int_\Omega w(\mathbf{x})d\mathbf{x} = 1$) and η is a constant that contains the amplitude of the relationship between the the horizontal velocity perturbation and the heat release rate perturbation. Here the measurement region is defined to be a pair of Gaussians, one each side of the centre-line:

$$w(x,y) = \exp\left(-a_r(x-x_0)^2 - a_r(|y|-y_0)^2)\right) \,, \tag{3.8}$$

where $a_r = 6200$, and $(x_0, y_0) = (D/2, D)$ where D is the bluff-body side length. This is somewhat arbitrary so long as the measurement region is the same in the thermoacoustic model as it is in the heat release model.

An estimate for the heat release $h(\mathbf{x})$ and time delay $\tau(\mathbf{x})$ fields is calculated from one period of re-simulated flame shapes, the re-simulated horizontal velocity field and the experiment horizontal velocity field. First, the normalised heat release rate $\dot{q}_h(\mathbf{x},t)/\bar{\dot{Q}}_h$ is calculated from the flame front positions over one period (see appendix B.1 for a derivation) using a level-set integration method implemented by Mr Alexandros Kontogiannis. The mean heat release rate $\bar{\dot{Q}}_h$ of the Volvo burner is 1.125 MW (see appendix B.2 for a detailed calculation). By taking the Fourier transform of the normalised heat release rate at the fundamental frequency and multiplying by $\bar{\dot{Q}}_h$, the (un-normalised) heat release perturbation $\tilde{\dot{q}}_h(\mathbf{x},\omega)$ is calculated. Next, the Fourier transform of the re-simulated horizontal velocity field at the fundamental frequency $\tilde{u}(\mathbf{x},\omega)$ is calculated to find the phase between the horizontal velocity field and the heat release rate perturbation field. In the simulations, the amplitude of the oscillating velocity field at the bluff-body is too small to be used. Instead, the measurement region is set one grid cell downstream of the bluff body in order to obtain the correct phase relationship between the velocity perturbation field and heat release rate perturbation field, and then is re-scaled to the experimental velocity amplitude. This ensures that the thermoacoustic model has the correct phase and amplitude relationships between velocity and heat release rate. Using this information, and calculating:

$$I = \int_\Omega w(\hat{\mathbf{x}}) |\tilde{u}(\hat{\mathbf{x}},\omega)| d\hat{\mathbf{x}} \,, \tag{3.9}$$

where $|\tilde{u}(\hat{\mathbf{x}},t)|$ is the amplitude of the experiment horizontal velocity field, n and τ fields are calculated:

$$n(\mathbf{x}) = \left| \frac{\tilde{\dot{q}}_h(\mathbf{x},\omega)}{I} \right| , \tag{3.10}$$

$$\tau(\mathbf{x}) = \angle \left(\frac{\tilde{\dot{q}}_h(\mathbf{x},\omega)}{I} \right) . \tag{3.11}$$

The phase of $\tilde{\dot{q}}_h(\mathbf{x},\omega)/I$ is wrapped between $\pm\pi$ and is unwrapped with the algorithm in [102]. The phase is then divided by ω, the angular frequency of the velocity field, to give the spatially-distributed time delay $\tau(\mathbf{x})$. Finally, $n(\mathbf{x})$ is normalised to find $h(\mathbf{x})$ and η:

$$h(\mathbf{x}) = \frac{n(\mathbf{x})}{\int_\Omega n(\hat{\mathbf{x}}) d\hat{\mathbf{x}}} = \frac{n(\mathbf{x})}{\eta}. \tag{3.12}$$

The growth rates and frequencies of thermoacoustic instabilities of the system can be estimated by using $n(\mathbf{x})$,$\tau(\mathbf{x})$ and $w(\mathbf{x})$ in a Helmholtz solver, as described in the next section.

3.7 The Helmholtz solver

The thermoacoustic Helmholtz equation, which is the frequency-domain equivalent of the thermoacoustic wave equation, is used to model the Volvo burner system. For a given heat release model, such as the distributed $n-\tau$ model described in the previous section, the Helmholtz equation is solved with a Helmholtz solver. The solution is a set of eigenmodes, each of which comprises an eigenvalue and an eigenfunction. The real and imaginary parts of the eigenvalue are the frequency and growth rate respectively of the thermoacoustic mode whose shape is given by the eigenfunction. The growth rate allows us to determine the thermoacoustic stability of the mode: a positive (negative) growth rate means an unstable (stable) mode. For further details on the Helmholtz solver, the reader is referred to: [103, Chapter 2].

3.8 Results and discussion

3.8.1 Neural network training and evaluation

The ensemble of 20 Bayesian neural networks is trained on the forced cycle library for 100 epochs, with a 80 : 20 train-test split and an Adam optimiser [104] with learning rate 10^{-4}. Training takes approximately one hour per neural network on an NVIDIA P100 GPU. The

ensemble is trained in parallel on GPU clusters available through Google Colaboratory[1]. Once the neural networks are trained, ensemble prediction takes $O(10^{-3})$ seconds on an Intel Core i7 processor on a laptop.

3.8.2 Results on experimental data

The top-left quadrant of each image in figure 3.3 shows the experimental flame position for four of the 430 timesteps in an experimental run. The BayNNE recognises the parameters (K, ε, γ, St, β) of the simulations that best match this experimental data and also outputs its uncertainty in those parameters, in the form of a standard deviation. These parameter values tend to remain approximately constant for many timesteps and then change abruptly. This concurs with a visual inspection of the experimental data which also exhibits periods of similar behaviour interspersed with moments of abrupt change. The BayNNE is able to assimilate the data reliably through these abrupt changes.

From figure 3.3, we observe that ε and γ appear strongly correlated. This is as expected, as these both relate to the amplitude of the perturbation. The Strouhal number is inferred with higher uncertainty than the other parameters in most of the 430 timesteps. This suggests that this parameter is less easily recovered from the input sequences of 10 flame front snapshots. Increasing the number of flame front snapshots in one observation vector input into the neural network ensembles could be one way of reducing this uncertainty. However, this would not be appropriate for such cases where the flame behaviour changes rapidly because a constant value of the parameters may not be a valid assumption over this longer sequence of snapshots. Additionally, this would increase the computational cost of training the neural networks: doubling the size of the input and hidden layers of a neural network increases the number of weights and biases by a factor of four.

Using the re-simulated flame shapes, the shapes of the flame envelope, $h(\mathbf{x})$, and the time delay fields, $\tau(\mathbf{x})$, of the distributed $n-\tau$ heat release model described earlier in this chapter can be calculated (recall that $n(\mathbf{x}) = \eta h(\mathbf{x})$, where η is a constant which ensures $h(\mathbf{x})$ integrates to 1). Figs. 3.4 to 3.7 show the re-simulated flame shapes and $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields at the four timesteps of interest. Additionally, the flame shapes are re-simulated at $1/4, 1/2$ and $3/4$ the amplitude inferred by the BayNNE (all other inferred parameters are kept constant). The $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields calculated from these flame shapes are shown in figures 3.8 and 3.9. At a given timestep, $h(\mathbf{x})$ becomes narrower as the amplitude ε reduces, but retains high amplitude regions on its inner and outer edges. On the other hand, η increases only slightly with amplitude. This shows that, once $h(\mathbf{x})$ has been fixed, the amplitude of

[1] https://colab.research.google.com

the global fluctuating heat release rate depends only slightly on the amplitude of oscillation. The time delay field, $\tau(\mathbf{x})$, becomes narrower as the amplitude reduces and has the same envelope as the $h(\mathbf{x})$ field. The influential region of $\tau(\mathbf{x})$ is simply the region in which $h(\mathbf{x})$ is large, which is primarily the outer edge of the envelope of $h(\mathbf{x})$. In this region, the value of $\tau(\mathbf{x})$ at a given x-position depends very little on amplitude and only slightly on the timestep. This is because $\tau(\mathbf{x})$ is determined by the convection speed, K. The small dependence on amplitude is expected because the K is not affected at all by the amplitude of oscillation, ε. The dependence on timestep is also expected to be small because the convection speed is similar at each timestep

The $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields calculated at the four timesteps and amplitudes can be used to calculated the eigenfrequencies and growth rates of the thermoacoustic system using a Helmholtz solver. This is discussed in the next section.

Table 3.3 Parameter values and uncertainties inferred by the BayNNEs at four different timesteps. The variation in the inferred parameters with time suggests that the flame behaviour changes during the time series between moments of higher and lower amplitude oscillations.

Parameter	$t = 48$	$t = 108$	$t = 216$	$t = 312$
K	0.953 ± 0.083	0.716 ± 0.084	0.690 ± 0.016	0.932 ± 0.013
ε	0.133 ± 0.005	0.102 ± 0.003	0.314 ± 0.003	0.184 ± 0.001
γ	1.117 ± 0.140	1.009 ± 0.105	1.577 ± 0.016	1.145 ± 0.010
St	21.243 ± 8.417	16.409 ± 5.315	25.656 ± 4.865	14.267 ± 1.859
β	7.932 ± 0.024	7.694 ± 0.115	7.739 ± 0.046	6.714 ± 0.116

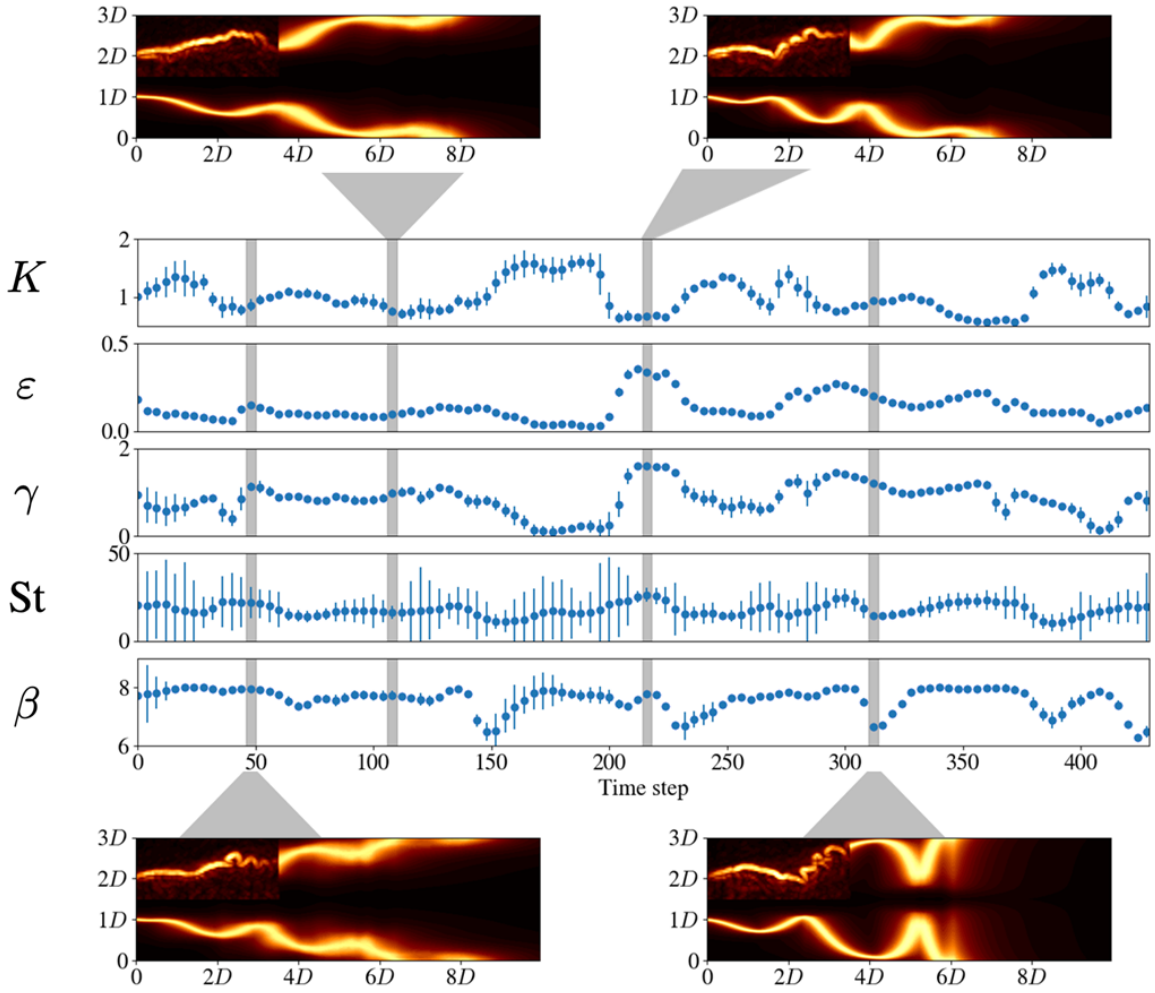


Fig. 3.3 Experimental flame images (top-left quadrant of each image) and level-set model predictions (remainder of image) at four timesteps within a sequence of 430 timesteps. The graphs show the means and $\pm$ 2 standard deviations of the five parameters of the *G*-equation simulation, which are assimilated from the experimental data. The level-set model predictions use these assimilated values to simulate the flame downstream, beyond the observation window. The inferred parameter values are listed in Table 3.3.

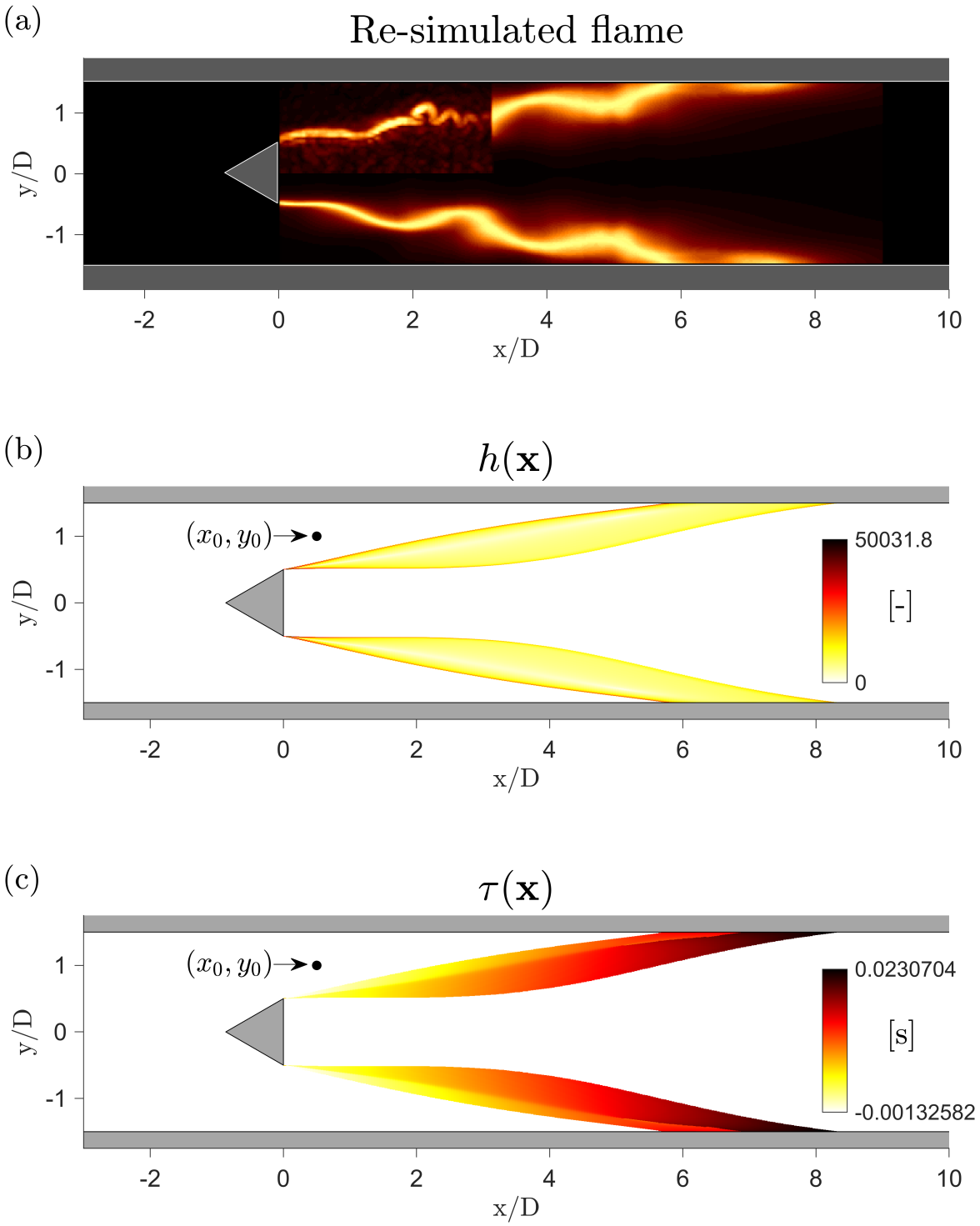


Fig. 3.4 *(a)* Re-simulated flame shape at $t = 48$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 48$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

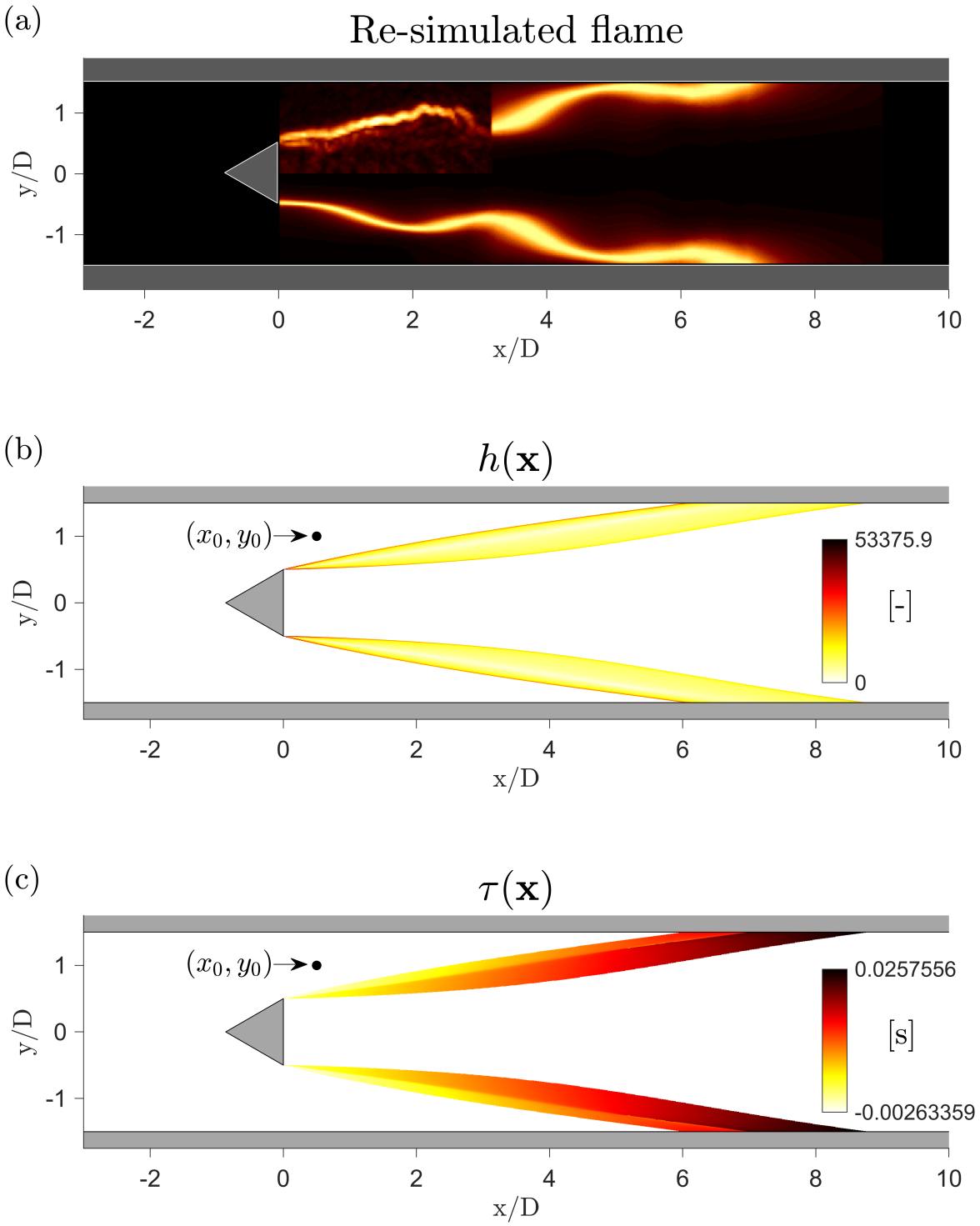


Fig. 3.5 *(a)* Re-simulated flame shape at $t = 108$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 108$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

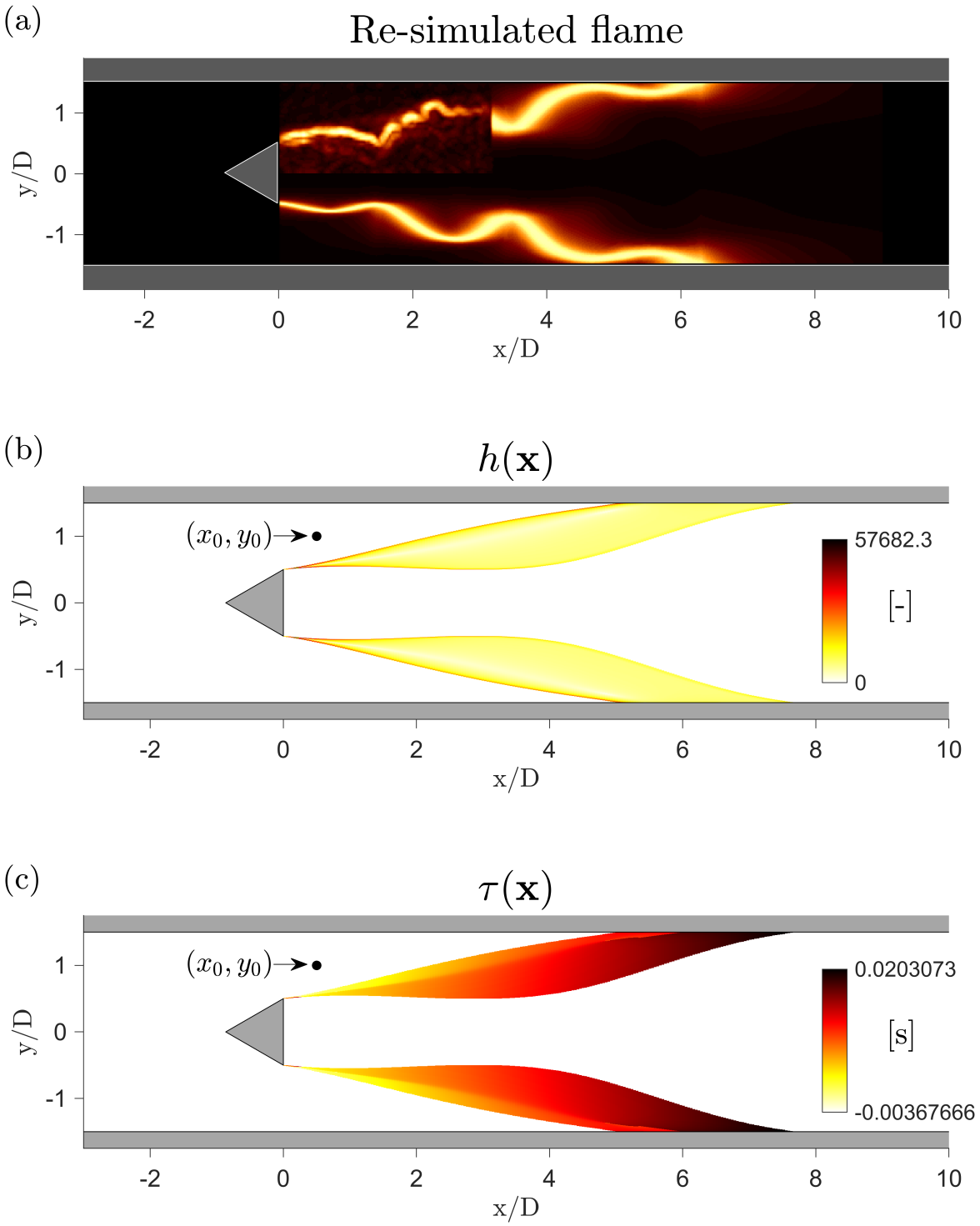


Fig. 3.6 *(a)* Re-simulated flame shape at $t = 216$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 216$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

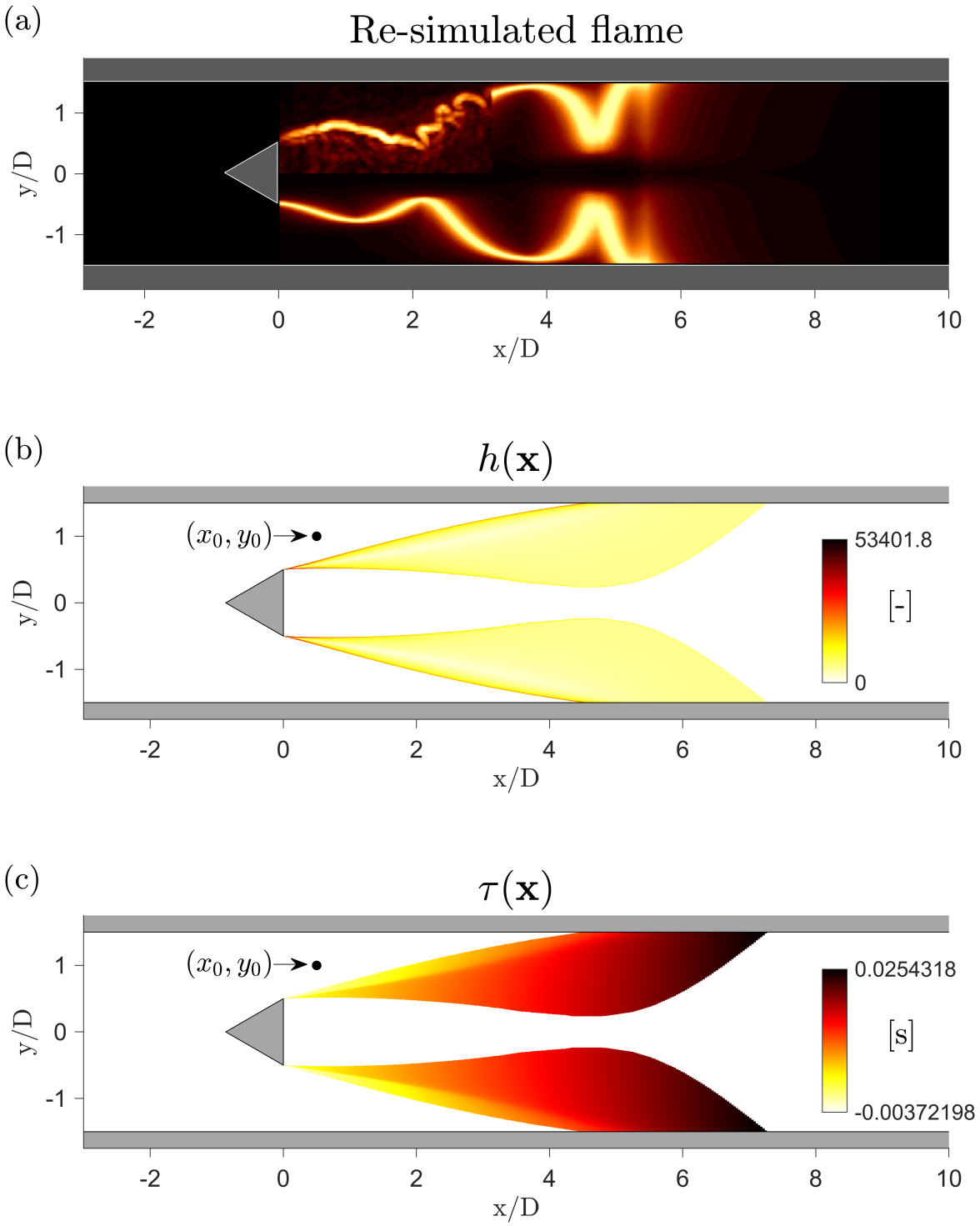


Fig. 3.7 *(a)* Re-simulated flame shape at $t = 312$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 312$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

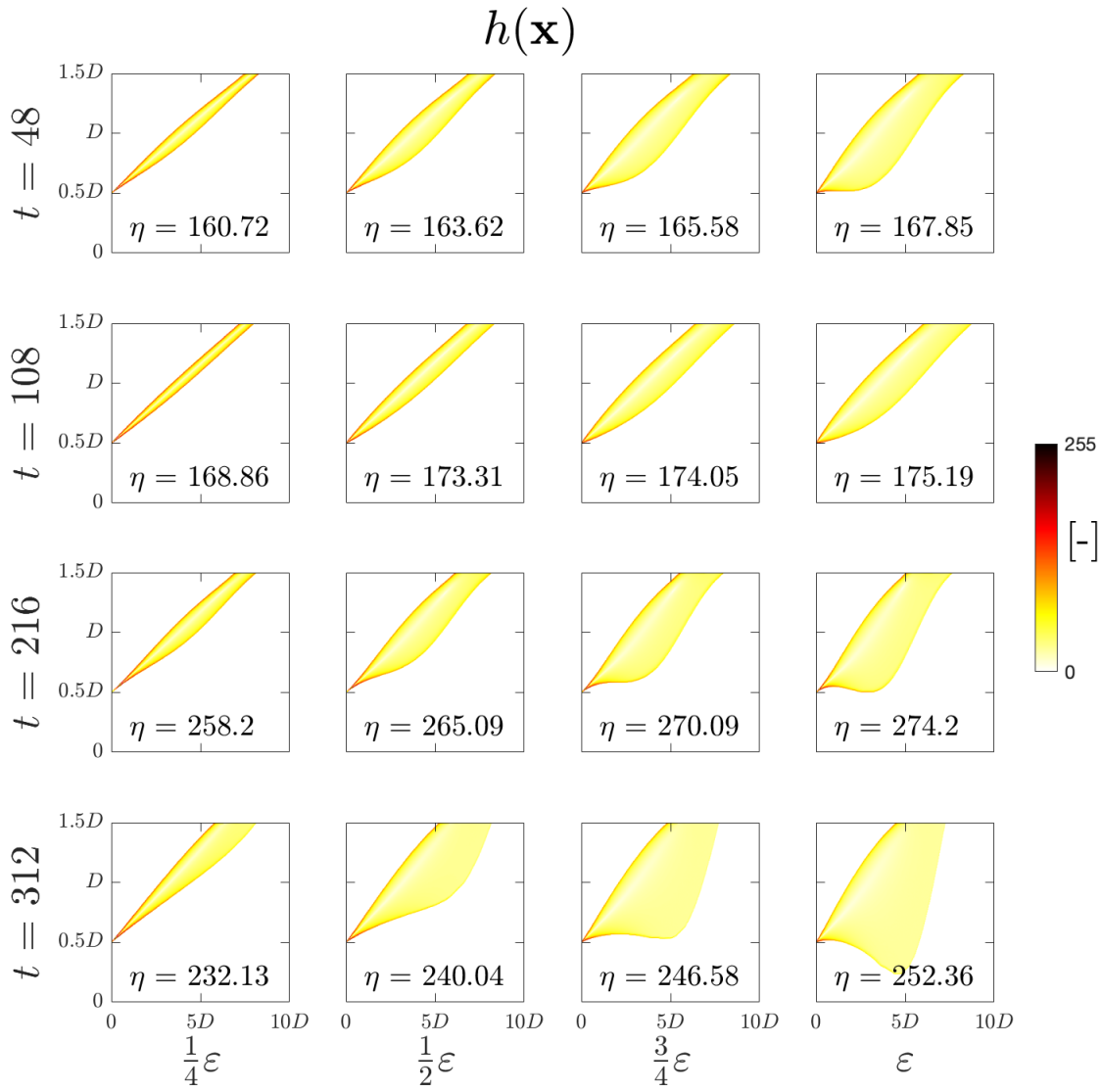


Fig. 3.8 Plots of $h(\mathbf{x})$ at four different timesteps t and at four amplitudes: $\varepsilon/4, \varepsilon/2, 3\varepsilon/4$ and ε, the amplitude inferred by the BayNNE. The values of η are also displayed for each timestep and amplitude.

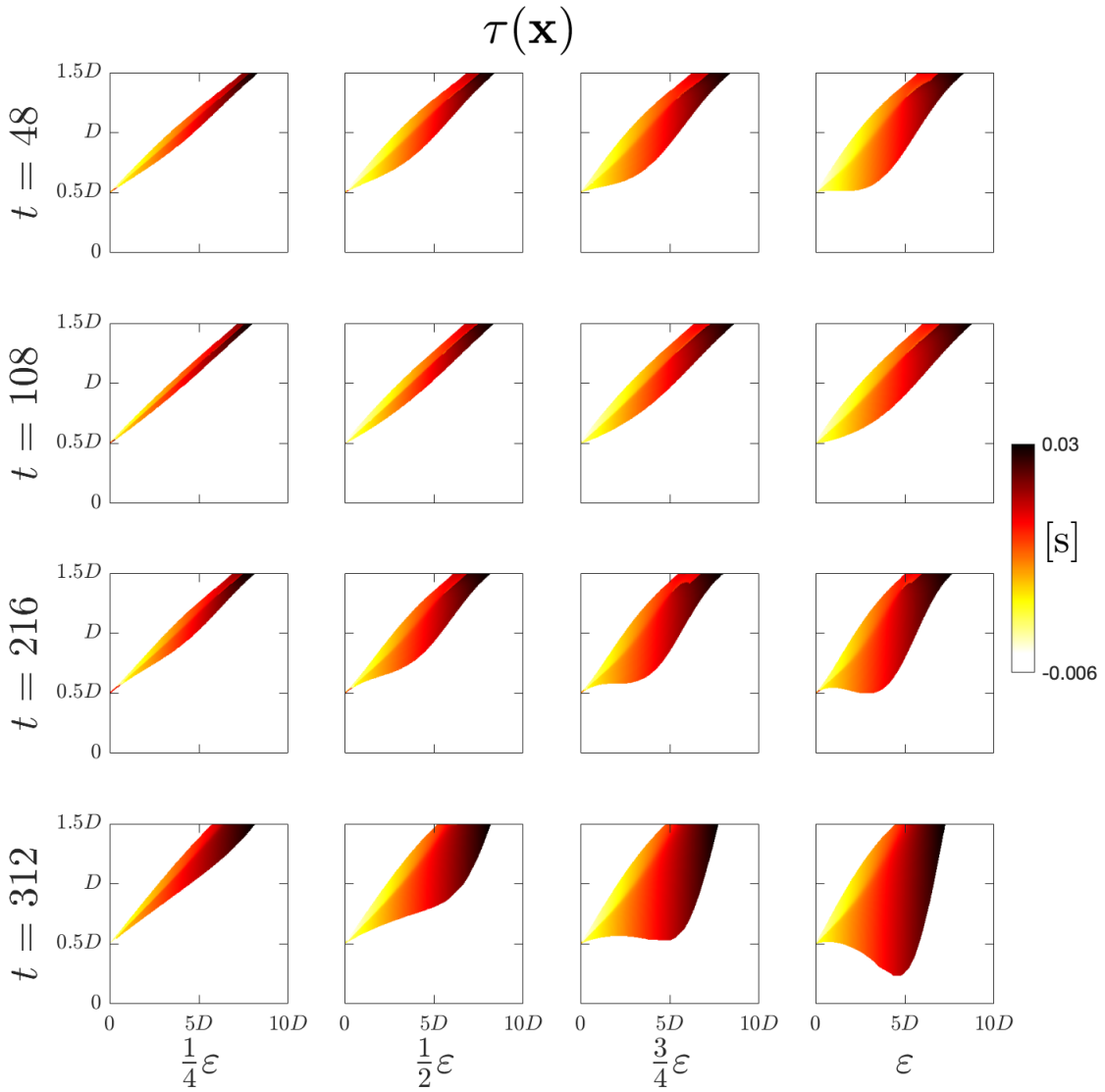


Fig. 3.9 Plots of $\tau(\mathbf{x})$ at four different timesteps t and at four amplitudes: $\varepsilon/4, \varepsilon/2, 3\varepsilon/4$ and ε, the amplitude inferred by the BayNNE.

3.8.3 Helmholtz solver results

Fig. 3.10 shows a typical eigenfunction for the first thermoacoustic mode. This has the same structure as the natural modes observed experimentally in a similar configuration by [35]. The eigenfunctions at the other timesteps and amplitudes are nearly identical. Fig. 3.11 shows the first pure acoustic eigenmode (red square) and the thermoacoustic eigenmodes (grey) at the four amplitudes and four timesteps. The grey symbols deviate from the red square, showing that the thermoacoustic effect is active. At each timestep, the thermoacoustic effect stabilises the mode and shifts the frequency slightly.

We expect the shape of $h(\mathbf{x})$ to have little influence on the thermoacoustic behaviour because this behaviour is determined by the integral of $h(\mathbf{x})$ multiplied by the pressure $p(\mathbf{x},t)$ over an acoustic wave, and $p(\mathbf{x},t)$ has a long wavelength, as seen in figure 3.10. On the other hand, $\tau(\mathbf{x})/T$ typically ranges from 0 to 4.5 in the region in which $h(\mathbf{x})$ has high amplitude. Small changes in $\tau(\mathbf{x})$ would therefore cause big changes in the phase between the heat release rate $\tilde{q}_h(\mathbf{x},t)$ and $p(\mathbf{x},t)$, and therefore big changes in the thermoacoustic growth rate, as described in [2]. However, $\tau(\mathbf{x})$ does not change with amplitude or timestep in the inner and outer regions of its envelope because it is determined by the convection speed, which is the same through the experimental run. The thermoacoustic behaviour is therefore similar at all four timesteps and all four amplitudes.

Fig. 3.10 Real (top) and imaginary (bottom) components of the first mode of the acoustic pressure eigenfunctions for the flame at t = 312 and amplitude ε. The eigenfunction is normalised such that $\int p^2 dx = 1$. This figure is due to Mr Ekrem Ekici.

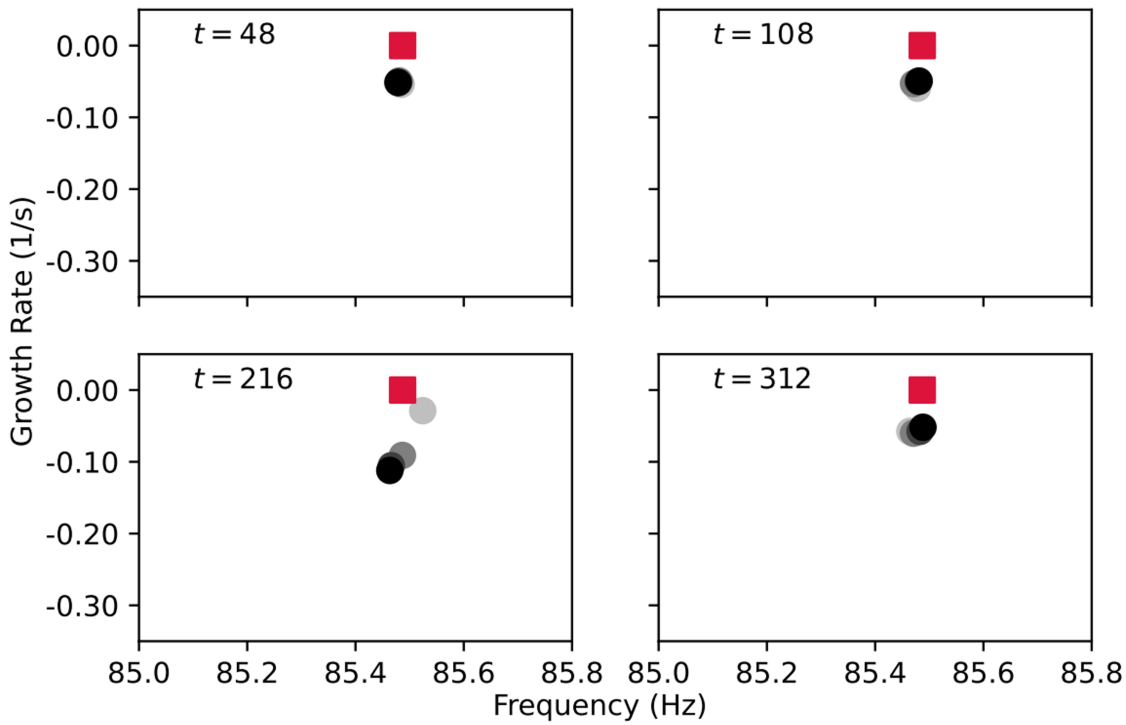


Fig. 3.11 The eigenvalues for the first mode at four timesteps and amplitudes. The horizontal and vertical axes denote the frequency and the growth rate of the system. The red squares in each subplot show the eigenfrequecies of the system when no flame is present. Each frame corresponds to a different timestep, while the greyscale within each frame corresponds to a different velocity forcing amplitude, ε. The lightest grey corresponds to $\varepsilon/4$ and the darkest to ε. This figure is due to Mr Ekrem Ekici.

3.9 Conclusions

In this chapter, the Bayesian machine learning method proposed in the previous chapter was applied to data from a version of the Volvo burner. In this burner, the flame is only partially observed and exhibits some turbulent, transient behaviour. The velocity model used in the G-equation model of the flame front is adapted to better describe the data: a parameter γ is introduced which determines the growth rate of the velocity perturbations with downstream coordinate. The ensemble Kalman filter is unable to assimilate this data because the flame behaviour varies too quickly for the Kalman filter to converge to parameter estimates. Instead, the G-equation model is used to generate a library of forced cycle frame fronts from which the heteroscedastic Bayesian neural network ensembles (BayNNEs) are trained. The BayNNEs learn to infer the five parameters of the velocity model and their uncertainties from 10 consecutive flame front snapshots. By re-simulating the flame fronts

using the inferred parameters, the behaviour of the flame downstream of the observation window is predicted. Using the predicted flame shapes over one period, the flame envelope, n, and time delay, τ, of a distributed $n-\tau$ heat release model are calculated. Furthermore, the $n-\tau$ fields are calculated for flames with the same parameters as those inferred by the BayNNE but with $1/4$, $1/2$ and $3/4$ of the inferred amplitude. By entering these data into a Helmholtz solver, the growth rates and frequencies of the thermoacoustic instability of the system are calculated. Crucially, we observe that the growth rate and frequency of the thermoacoustic instability do not show a strong dependency on the amplitude of forcing. This suggests that a flame transfer function derived at high amplitude, when it is observable, is also valid at low amplitude, when it is not observable.

Assimilation into physics-based models, as performed here, is attractive because the models are physically-interpretable and extrapolatable. In addition, this provides a cheap way to store CFD data: the parameters of the most relevant CFD solution for a given experiment can be extracted cheaply, and the CFD solution then re-calculated. Secondly, it shows how sparse experimental results can be combined with complete numerical results to extrapolate, with defined confidence levels, beyond experimental observations. This chapter described a robust way and potentially cheap method to achieve this, which can readily be extended to other experiments and to other models.

Chapter 4

Parameter inference for a kinematic model of a bluff-body stabilised flame augmented with a discrete vortex velocity model

This chapter is based upon the following conference paper, which was double-blind peer-reviewed by at least one reviewer:

- M. L. Croci, J. V. Vasanth, U. Sengupta, E. Ekici and M. P. Juniper, "Bayesian parameter inference of a vortically perturbed flame model for the prediction of thermoacoustic instability", in *AI for Science workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS)*, (New Orleans, USA), 2022 [105].

As in the previous chapter, the experimental data is supplied by Mr Brett Rankin, the neural network method is that which was first proposed in [76] and LSGEN2D is implemented as in [84]. The heat release model was primarily implemented by Professor Matthew Juniper, and the code for calculating the normalised flame surface area was written by Mr Alexandros Kontogiannis. The implementation of, and results from, the Helmholtz solver are due to Mr Ekrem Ekici. The discrete vortex method was developed and implemented by Mr Joel Vasanth. I incorporated the discrete vortex method code within LSGEN2D and compiled the the code into a single executable program.

4.1 Introduction

In the previous chapter, the parameters of a physics-based model of a version of the Volvo burner were inferred from experiments using a heteroscedastic Bayesian neural network ensemble. This has the advantage of simplicity and periodicity, but lacks flame wrinkling and other dynamics that may be important in this flow. In this chapter, an alternative model of the velocity field is used which wrinkles the flame while retaining incompressibility, which is physically realistic in the fresh gases before dilatation at the flame. In the experiments, large-scale flame front oscillations due to vortices shedding from the bluff body are observed. A discrete vortex method with five tunable parameters is therefore used to calculate the velocity field in an attempt to model this behaviour more faithfully. Aside from this difference, the Bayesian machine learning method is unchanged: a library of simulated flame fronts using LSGEN2D and the discrete vortex method is produced, from which the neural networks are trained to recover the parameters of the discrete vortex method. By re-simulating the flame fronts using these inferred parameters, the behaviour of the flame downstream of the observation window is evaluated. As was performed in the previous chapter, the $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields of the heat release model are calculated and entered into a Helmholtz solver to calculate the frequency and growth rate of the thermoacoustic instability of the system.

4.2 The G-equation model of the Volvo burner augmented with a discrete vortex velocity model

4.2.1 The G-equation model

The G-equation model of flame front is unchanged except for the velocity field $\mathbf{u}$. The G-equation is:

$$\frac{\partial G}{\partial t} + \mathbf{u} \cdot \nabla G = s_L |\nabla G| \,, \tag{4.1}$$

and $\mathbf{u}$ is now calculated using a discrete vortex method instead of the continuity-obeying sinusoidal horizontal and vertical velocity perturbations used in the previous chapter. As before, the flame speed s_L is equal to the unstretched flame speed s_L^0 but is no longer a function of the local flame curvature.

4.2.2 The discrete vortex method

The velocity field $\mathbf{u}(\mathbf{x},t)$ is calculated using a discrete vortex method (DVM, [106, 107]). This method aims to model the large-scale vortical structures that drive flame front oscillations

using the vorticity formulation of the incompressible Euler equations in two dimensions:

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \mathbf{u} \cdot \nabla \boldsymbol{\omega} = \boldsymbol{\omega} \cdot \nabla \mathbf{u} \ , \tag{4.2}$$

$$\nabla \cdot \mathbf{u} = 0 \ , \tag{4.3}$$

where $\boldsymbol{\omega}(\mathbf{x},t)$ is the vorticity field. This vorticity field is discretised into N_V individual fluid elements, each with position $\mathbf{x}_i(t)$, core radius $r_{D,0}$ and circulation Γ_i, where $i = 1,2 \ldots N_V$:

$$\boldsymbol{\omega} = \sum_{i}^{N_V} \Gamma_i f_\delta(r) \ , \tag{4.4}$$

where the core function $f_\delta(r)$ is the distribution of vorticity within the vortex element. The vortices are advected by the flow, whose velocity field is the solution to the Poisson equation $\nabla^2 \mathbf{u} = -\nabla \times \boldsymbol{\omega}$:

$$\mathbf{u}(\mathbf{x},t) = \mathbf{u}(\mathbf{x},t)_{\text{irr}} + \mathbf{u}(\mathbf{x},t)_{\text{rot}} \ . \tag{4.5}$$

The irrotational velocity field $\mathbf{u}(\mathbf{x},t)_{\text{irr}}$ is the solution to the potential flow problem, calculated via a Schwartz-Christoffel conformal mapping [108]. The assumption that this base flow is irrotational ignores the gradually increasing turbulent behaviour which is observed downstream of the bluff body. The acoustic forcing on the flow field is modelled with a harmonically oscillating inlet flow velocity $u(t)/U = 1 + \alpha \sin(\text{St}\, t)$ where α is the amplitude of forcing. The rotational velocity field is:

$$\mathbf{u}(\mathbf{x},t)_{\text{rot}} = \nabla \mathscr{G}(\mathbf{x},\mathbf{x}') * \boldsymbol{\omega}^h \ , \tag{4.6}$$

where $*$ is the convolution operator and $\mathscr{G}(\mathbf{x},\mathbf{x}')$ is the Green's function for the Laplacian operator in two dimensions:

$$\nabla^2 \mathscr{G}(\mathbf{x},\mathbf{x}') = \delta(\mathbf{x} - \mathbf{x}') \ . \tag{4.7}$$

The vortices advect with the flow according to:

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}(\mathbf{x}_i,t) \quad i = 1,\ldots,N_V \tag{4.8}$$

Vortices enter the domain at each timestep Δt with circulation $\Gamma_0 = -u_0(t)^2 \Delta t/2$. The velocity u_0 is computed at a location $\mathbf{x}_0 = (0, z_{0,\text{i}})$, which is directly above the bluff body lip. For further details on the method, see appendix C.

4.2.3 Transients in the flame fronts simulated with the discrete vortex method

Despite the periodic forcing of the base flow, the effect of the vortices on the velocity field is not periodic. This is illustrated in figure 4.1: for a given set of parameter values, the flame shows aperiodic behaviour over four periods. The effect is particularly strong starting from a distance of $4D$ downstream of the bluff body. However, over the window for which experimental data are available (a distance $3.4D$ downstream), the flame front is almost perfectly periodic. For this reason, it is possible to generate a library of simulated flame fronts in the same way as was done in previous chapters. This goes some way to modelling the uncertainty in the flame front position downstream of the observation window. An alternative library generation method would calculate phase-averaged flame front coordinates, taken over many periods of simulation. The computational cost and storage requirements of the library generation are linear in the number of periods simulated, however. Furthermore, the turbulent, nearly periodic behaviour in the experiments cannot be as faithfully modelled if it is used. Therefore, we choose to generate the simulated flame front library by assuming perfectly periodic flame front behaviour, as was assumed in the previous chapters.

4.3 Library of forced flame simulations

A library of flame front locations is created with the flame front model at known parameter values $\alpha, r_{D,0}, z_{0,\mathrm{i}}, \mathrm{St}, \beta$ and f/f_s in the same format as the observation vectors, $\mathbf{z}$. The parameter values are sampled using quasi-Monte Carlo sampling to ensure good coverage of the parameter space. The parameters are sampled from the ranges in Table 4.1. The values of St are calculated using $\mathrm{St} = 2\pi f D\beta/U$. The parameters are sampled 10^4 times, normalised to between -1 and 1 and recorded in target vectors $\{\mathbf{p} = [\alpha, r_{D,0}, z_{0,\mathrm{i}}, \mathrm{St}, \beta]\}$. A larger number of parameter samples are required compared to the previous method due to the range of flame shapes produced with the discrete vortex method.

The G field is iterated forward in time for a duration of seven periods, where the period is $1/f$. This allows any transient flame behaviour to die away. Then the G field is iterated forward for a further two periods. The value of the G field at $N_T = 200$ equally-spaced timesteps within the first period is recorded: $\{G_1, G_2, \ldots, G_{N_T}\}$. For each G field in the sequence $\{G_i\}$, the flame front $y = f(x)$ is extracted from the $G = 0$ contour for all x in the range of the experiment observation window. The flame front y coordinates are recorded in a column vector $\mathbf{y}_i$. The resulting sequence $\{\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_{N_T}\}$ represents the flame front position at N_T equally spaced timesteps within the two periods recorded. Next, the sum of squared

errors between the first G field and the G field one period after is calculated. As before, the frame sequence $\{G_i\}$ is deemed to be a valid forced cycle only if this sum of squared errors falls below a tolerance tol, otherwise it is discarded along with its target parameters. The frame rate of the forced cycle sequence is N_T/T which is not, in general, equal to the frame rate of the experimental data ($f_s = 10^4$ Hz). To create a sequence of ten simulated flame fronts with the same frame rate as the experimental data starting from a timestep t_1, we calculate the ratio $\Delta = N_T f/f_s$ and select the sequence $\{\mathbf{y}_{t_1}, \mathbf{y}_{t_1+\Delta}, \ldots, \mathbf{y}_{t_1+9\Delta}\}$.

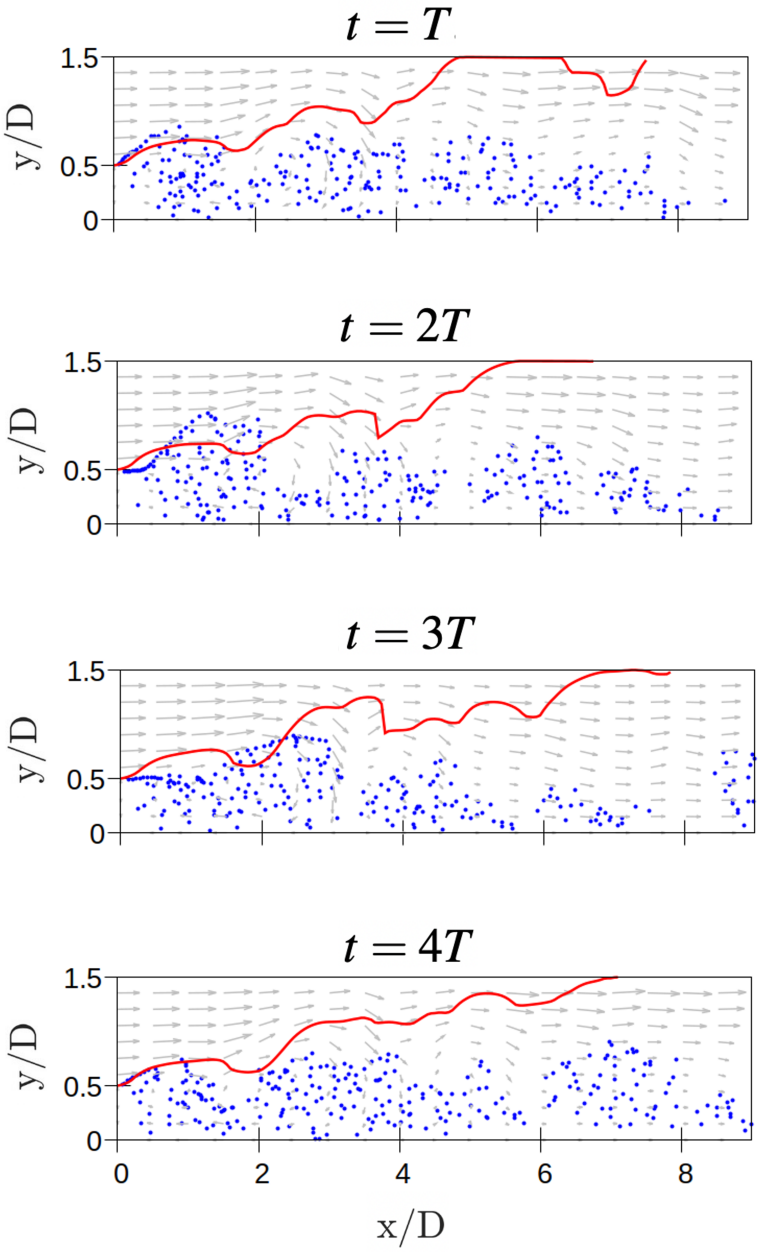


Fig. 4.1 Flame fronts (red line) one period apart simulated using the *G*-equation and the discrete vortex method. The blue dots are the centres of the vortices, advected by the velocity field shown by the grey arrows. Despite constant parameter values, the figures show increasingly aperiodic behaviour with downstream distance from the bluff body. This is due to the non-linear response of the flame dynamics to the velocity field induced by the discrete vortices.

Any flame coordinates which are outside of the experimental observation window are discarded. The vectors in this sequence are appended together to make a column vector $\mathbf{z}$. This is repeated for all N_T of the starting timesteps $t_1, t_2 \ldots t_{N_T}$, and for $P = 10^4$ combinations of parameters sampled from the ranges shown in Table 4.1. The result is a library of $PN_T = 2 \times 10^6$ observation - parameter pairs. This is summarised in algorithm 3.

As before, the algorithm can be trivially parallelised. One loop of algorithm 3 takes up to 90 minutes. An upper bound for the library generation's computational cost is 1.5×10^4 CPU core-hours. This takes approximately 23 hours when at the limit of maximum concurrent CPU core usage.

Table 4.1 Parameters of the discrete vortex method used in the G-equation model that are varied in this chapter and the range over which they are varied in order to generate the simulated flame front library.

Parameter ranges	Description
$0.05 \leq \alpha \leq 0.5$	Amplitude of base flow forcing
$0.01 \leq r_{D,0} \leq 0.2$	Vortex radius
$0.6 \leq z_{0,\mathrm{i}} \leq 0.7$	Initial velocity coordinate
$4 \leq \beta \leq 8$	Aspect ratio of the unperturbed flame
$0.03\ \mathrm{m} \leq D \leq 0.04\ \mathrm{m}$	Bluff-body side length
$10\ \mathrm{m/s} \leq U \leq 15\ \mathrm{m/s}$	Mean base flow speed
$100\ \mathrm{Hz} \leq f \leq 150\ \mathrm{Hz}$	Frequency of base flow forcing

4.4 Parameter inference for the G-equation model of the Volvo burner using BayNNEs

The heteroscedastic Bayesian neural network ensemble method is unchanged compared to that described in the previous chapter. The architecture of each neural network in the ensemble is unchanged: the output layers are five units wide, one for each parameter to be inferred, although three of the five parameters are different to those from the previous chapter. For full hyperparameter details, see Table 4.2.

Algorithm 3: Generate a library of simulated, partially observed Volvo flame fronts with known discrete vortex method parameters.

Data: Number of parameter samples P, number of timesteps per period N_T, frame rate f_s, tolerance tol.
Result: Library of pairs $(\mathbf{p}_i, \mathbf{z}_i)_{i=1}^{PN_T}$
for $1 \leq p \leq P$ **do**
  $\alpha, r_{D,0}, z_{0,\mathrm{i}}, f, \beta, D, U \leftarrow sample()$; ▷ Sample parameters from Tab. 4.1.
  $\mathrm{St} \leftarrow 2\pi f \beta D / U$;
  $\mathbf{p} \leftarrow [\alpha, r_{D,0}, z_{0,\mathrm{i}}, \mathrm{St}, \beta]$; ▷ Record the target parameters.
  $T \leftarrow 1/f$;
  $t \leftarrow 0$;
  $dt \leftarrow T/N_T$;
  $LSGEN2D.initialise(\mathbf{p})$; ▷ Initialise LSGEN2D with target parameters.
  while $t \leq 7T$ **do**
    ▷ Iterate the G-equation forward in time for the duration of the transient period.
    $LSGEN2D.iterate()$;
    $t \leftarrow t + dt$;
  end
  while $t_c \leq t \leq t_c + 2T$ **do**
    ▷ Once the transient period is over, iterate for a further two periods and record the flame front y coordinates.
    $LSGEN2D.iterate()$;
    $G_t \leftarrow LSGEN2D.G$;
    $\mathbf{y}_t \leftarrow G_t.extractFrontWithinWindow()$;
    $t \leftarrow t + dt$;
  end
  if $|G_{7T} - G_{8T}| < tol.$ **then**
    ▷ Verify that the convergence of the forced cycle by comparing G fields one period apart.
    $\Delta \leftarrow N_T f / f_s$;
    for $t_0 = 1 : N_T$ **do**
      $\mathbf{z} \leftarrow concatenate(\mathbf{y}_{t_0}, \mathbf{y}_{t_0+\Delta}, \ldots, \mathbf{y}_{t_0+9\Delta})$; ▷ Create observation vector.
      $library.addPair(\mathbf{p}, \mathbf{z})$; ▷ Save the parameter - observation pair.
    end
  end
end

Table 4.2 Hyperparameter settings used for neural network training.

Hyperparameter	Value
Training	
Train-test split	80:20
Batch size	256
Epochs	40
Optimiser	Adam
Learning rate	10^{-4}
Architecture	
Input units	600
Hidden layers	4
Units per hidden layer	600
Output layers	2
Units per output layer	5
Ensemble size	20

4.5 Results and discussion

4.5.1 Neural network training and evaluation

The ensemble of 20 Bayesian neural networks is trained on the forced cycle library for 40 epochs, with a 80 : 20 train-test split and an Adam optimiser with learning rate 10^{-3}. The ranges from which the *G*-equation model parameters are sampled from are listed in Table 4.1 Training takes approximately 20 minutes per neural network on an NVIDIA P100 GPU. The ensemble is trained in parallel on GPU clusters available through Google Colaboratory. Once the neural networks are trained, ensemble prediction takes $O(10^{-3})$ seconds on an Intel Core i7 processor on a laptop.

4.5.2 Results on experimental data

For five of the 430 timesteps in an experimental run, the top-left quadrant of each image in figure 4.2 shows the experimental flame position. The BayNNE recognises the parameters (α, $r_{D,0}$, $z_{0,\mathrm{i}}$, St, β) of the simulations that best match this experimental data and also outputs its uncertainty in those parameters, in the form of a standard deviation. As observed in the previous chapter, these parameter values tend to remain approximately constant for many timesteps and then change abruptly. The BayNNE is able to assimilate the data reliably through these abrupt changes.

From figure 4.2, we see that the uncertainties in $r_{D,0}$, the vortex core radius, are high: the two standard deviation range covers nearly the whole range from which the parameter was sampled (see Table 4.1). This suggests that this parameter is difficult to recover because it only has a weak effect on the flame front over ten consecutive images.

Between $t \approx 100$ and $t \approx 340$ we observe approximately periodic behaviour in the plots of the inferred values of α and $z_{0,\mathrm{i}}$. The period of this behaviour is 80 timesteps, which is equal to the period of the oscillations observed in the experiment. This implies that there is some model error present: the neural network ensemble predicts different flame shapes for different moments in the period because no single shape describes the whole period well. However, unlike the core radius parameter $r_{D,0}$, these two parameters are more easily recovered, implying that their effect on the flame front over ten consecutive images is strong.

In general, all five parameters are inferred with higher uncertainties than the parameters of the previous model. This is explained by the nearly periodic behaviour in the flame fronts simulated using the discrete vortex method and the G-equation. For a given set of parameters in the training library, the corresponding flame fronts are recorded over two periods. Simulating the flame fronts over a further two periods would give rise to slightly different flame behaviour. Table 4.3 lists the parameter values and uncertainties inferred by the BayNNEs at five different moments in the time series, with the Strouhal number being particularly sensitive to the time at which the parameters are inferred. This is due to the rapidly changing behaviour of the flame front from one period to the next. The BayNNE cannot learn the parameters with very high certainty from these flame fronts because it does not have enough examples of all possible flame behaviour. One remedy would be to simulate many periods of the flame front for a given set of target parameters and use a phase-averaged period for training. However, the computational cost of generating the training library is linear in the number of periods over which the flame front is simulated. Given that the current library takes 23 hours to generate, this is prohibitively expensive.

From the re-simulated flame images we notice that the flame front position is predicted with low uncertainty up to a distance approximately $2D$ downstream of the bluff body. Further downstream, the uncertainty increases. This is expected because the flame experiments are more turbulent in this region and so do not show perfectly periodic behaviour here. The flame shapes produced with the discrete vortex model are only nearly periodic, which matches this observed behaviour. Downstream of the observation window, the uncertainty in the flame position continues to increase, as expected.

Using the re-simulated flame shapes, $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields of the distributed $n-\tau$ heat release model (which is unchanged from the previous chapter) are calculated. Figs. 4.3 to 4.7 show the re-simulated flame shapes and $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields at the five timesteps of

interest. Additionally, the flame shapes are re-simulated at $1/4, 1/2$ and $3/4$ the amplitude inferred by the BayNNE for a duration of ten periods and a phase-average of the flame shapes is taken. The $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields calculated from these flame shapes are shown in figures 4.8 and 4.9. Due to computer storage limits, phase averaging more than ten periods of flame shape data is impractical. This leads to noisy artefacts seen in the inner regions of the $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields. However, the regions of highest $h(\mathbf{x})$ are on the inner and outer edges of the envelope of $h(\mathbf{x})$, as was the case in the previous chapter. The influential region of $\tau(\mathbf{x})$ is, as before, primarily the outer edge of the envelope of $h(\mathbf{x})$. In this region, the value of $\tau(\mathbf{x})$ at a given x-position depends more on the amplitude and on the timestep than was seen in the previous chapter.

The envelopes of the $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields are smaller than those of the previous chapter: the perturbations decay faster with downstream distance than under the previous model. The effect of the amplitude of velocity forcing, ε, on the envelopes of $h(\mathbf{x})$ and $\tau(\mathbf{x})$ is less pronounced. This suggests that the shapes of the envelopes are more sensitive to the rotational part of the velocity field than to the irrotational part. Under the discrete vortex model, the flame dynamics are driven more by the flow induced by the vortices than by the oscillating base flow.

As in the previous chapter, the $h(\mathbf{x})$ and $\tau(\mathbf{x})$ fields at each timestep and amplitude can be used to determine the eigenfrequencies and growth rates of the thermoacoustic system calculated with a Helmholtz solver. This is discussed in the next section.

Table 4.3 Parameter values and uncertainties inferred by the BayNNEs at five different timesteps.

Param.	$t = 32$	$t = 60$	$t = 304$	$t = 376$	$t = 416$
α	0.433 ± 0.105	0.303 ± 0.142	0.435 ± 0.092	0.452 ± 0.089	0.305 ± 0.137
$r_{D,0}$	0.099 ± 0.049	0.096 ± 0.061	0.100 ± 0.069	0.092 ± 0.052	0.078 ± 0.035
$z_{0,\mathrm{i}}$	0.615 ± 0.022	0.614 ± 0.017	0.612 ± 0.021	0.613 ± 0.021	0.635 ± 0.028
St	14.30 ± 3.458	16.19 ± 4.413	15.78 ± 5.411	15.29 ± 5.860	13.17 ± 4.156
β	7.974 ± 0.050	7.931 ± 0.156	7.918 ± 0.221	7.913 ± 0.210	7.957 ± 0.109

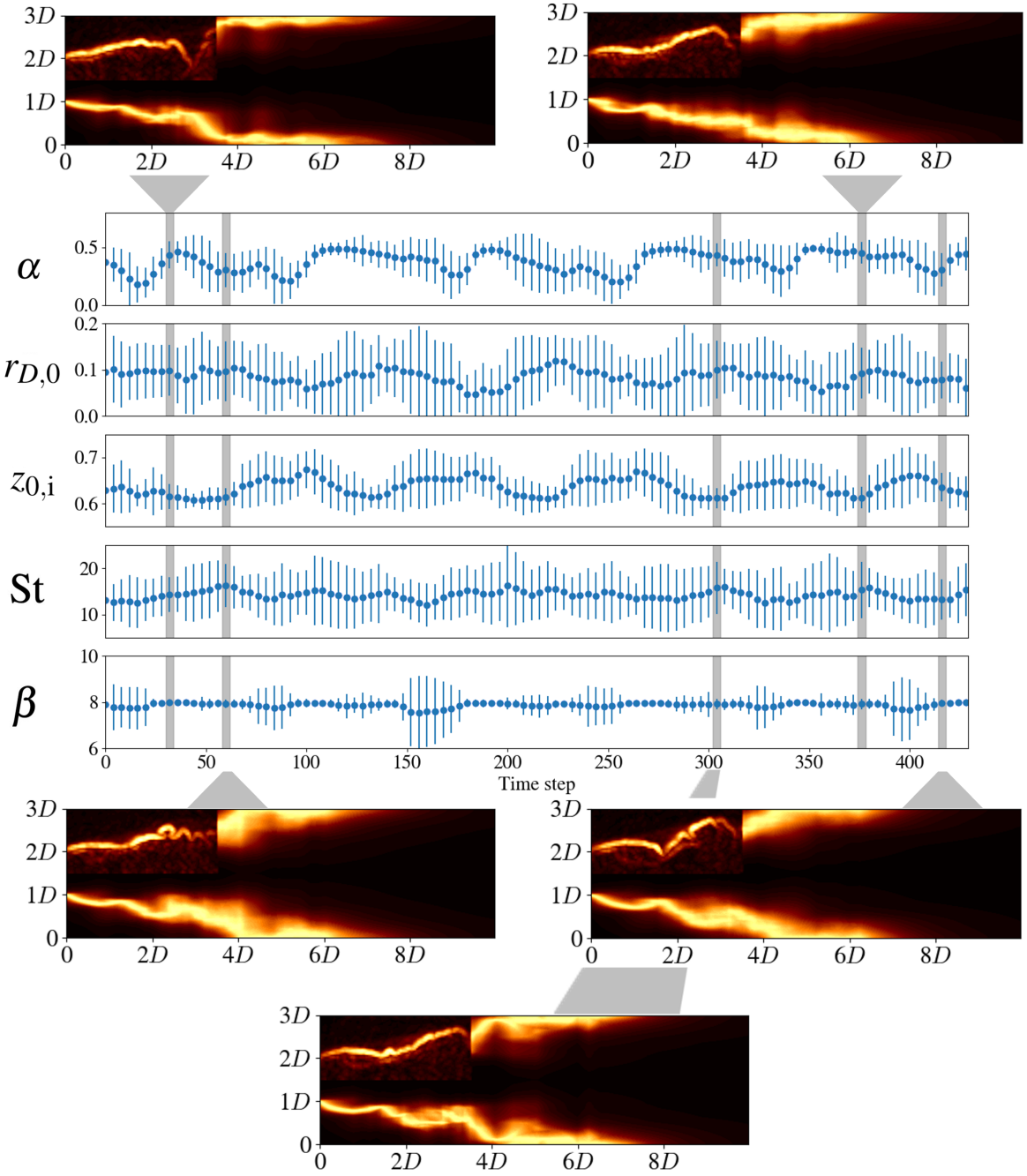


Fig. 4.2 Experimental flame images (top-left quadrant of each image) and level-set model predictions (remainder of image) at five timesteps within a sequence of 430 timesteps. The graphs show the means and $\pm$ 2 standard deviations of the five parameters of the discrete vortex method velocity model, which are assimilated from the experimental data. The level-set model predictions use these assimilated values to simulate the flame downstream, beyond the observation window. The inferred parameter values are listed in Table 4.3.

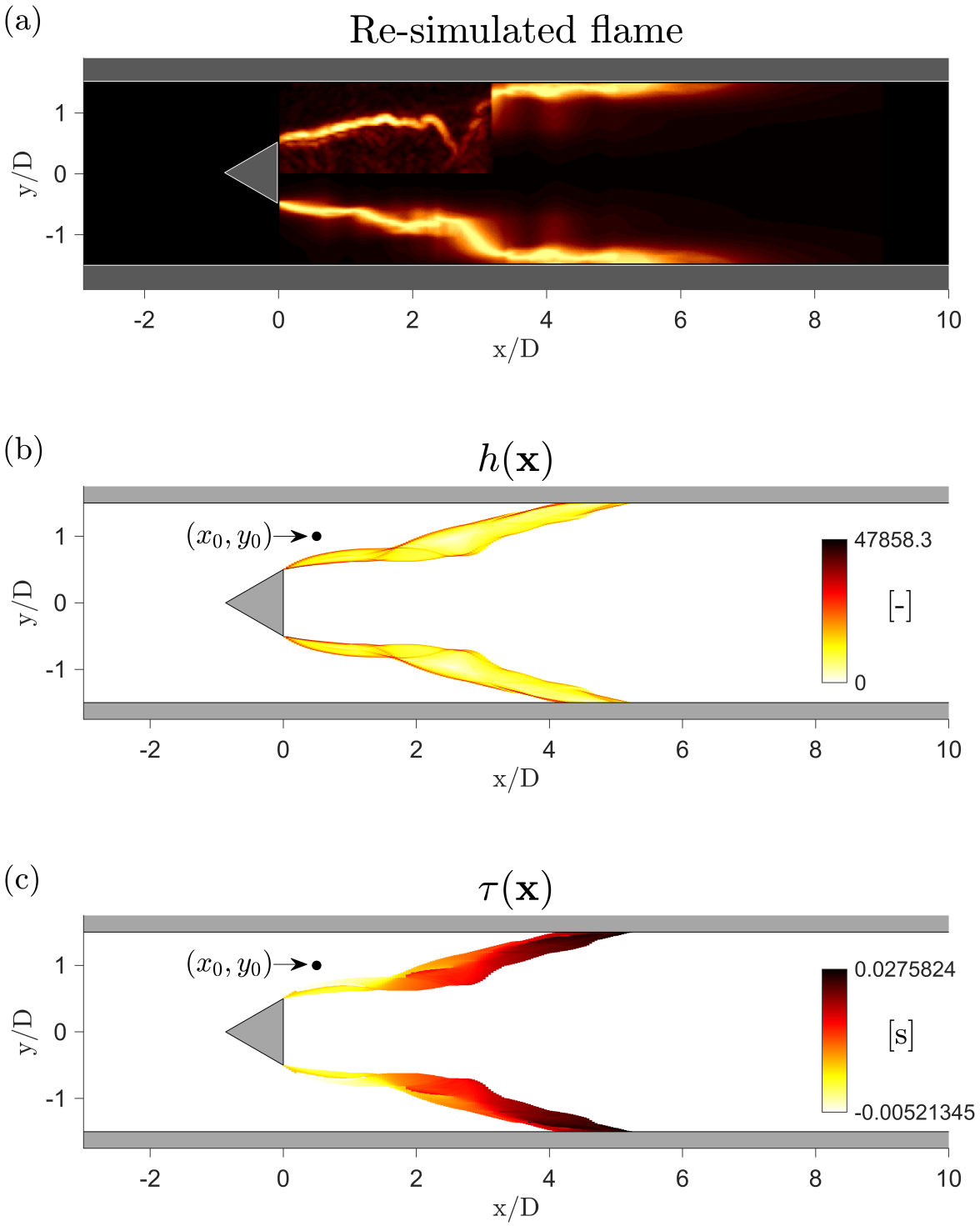


Fig. 4.3 *(a)*: Re-simulated flame shape at $t = 32$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 32$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

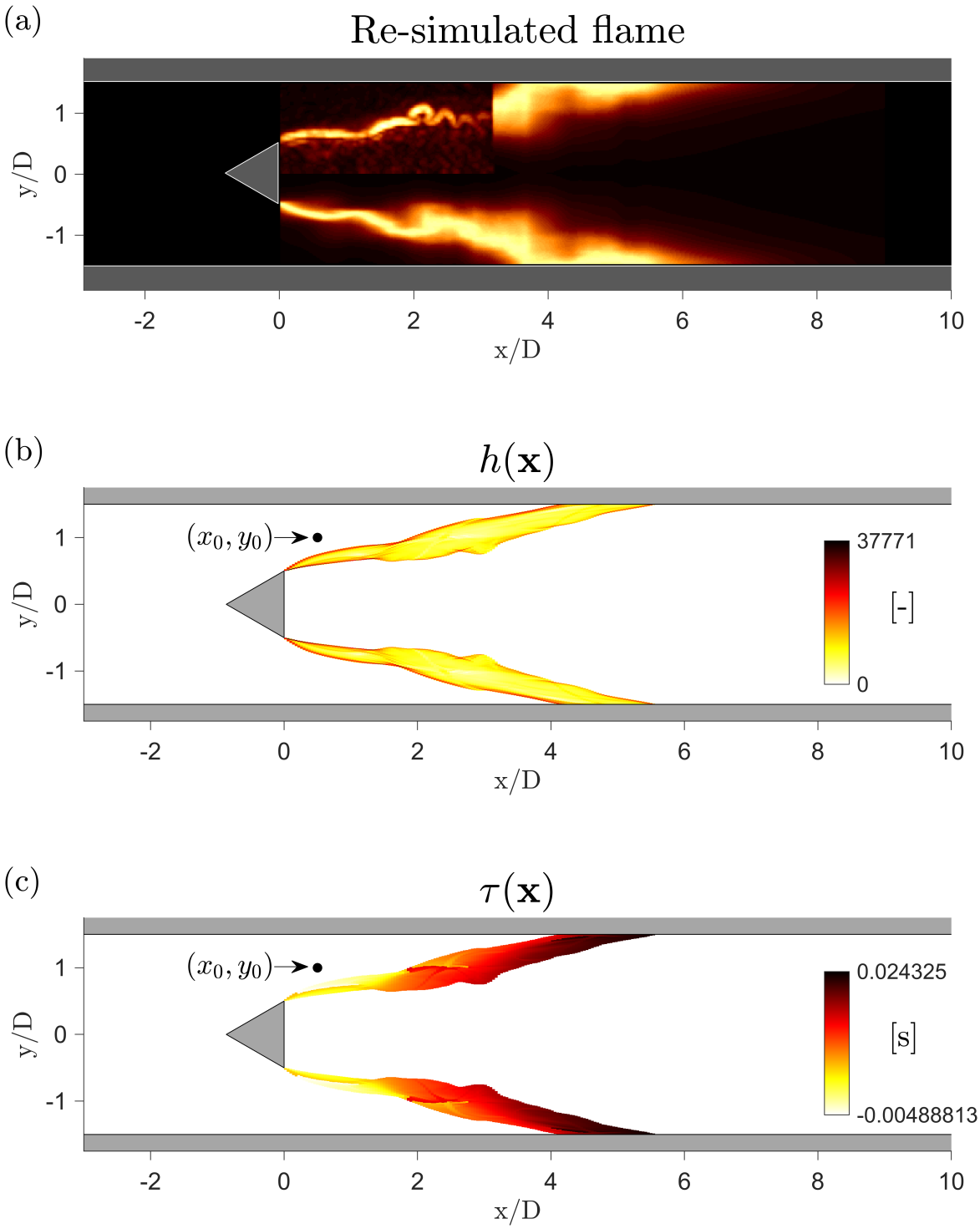


Fig. 4.4 *(a)*: Re-simulated flame shape at $t = 60$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 60$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

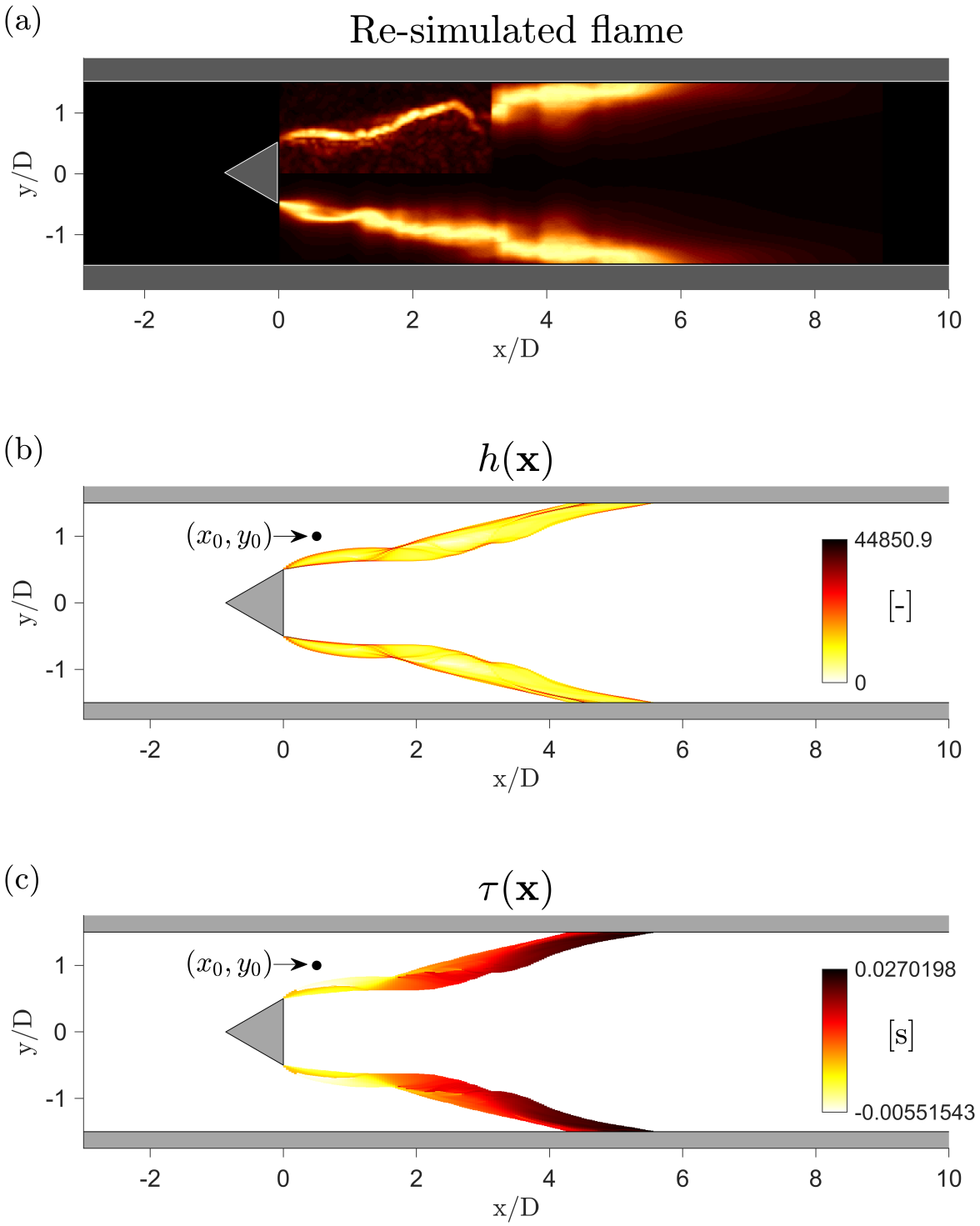


Fig. 4.5 *(a)*: Re-simulated flame shape at $t = 304$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 304$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

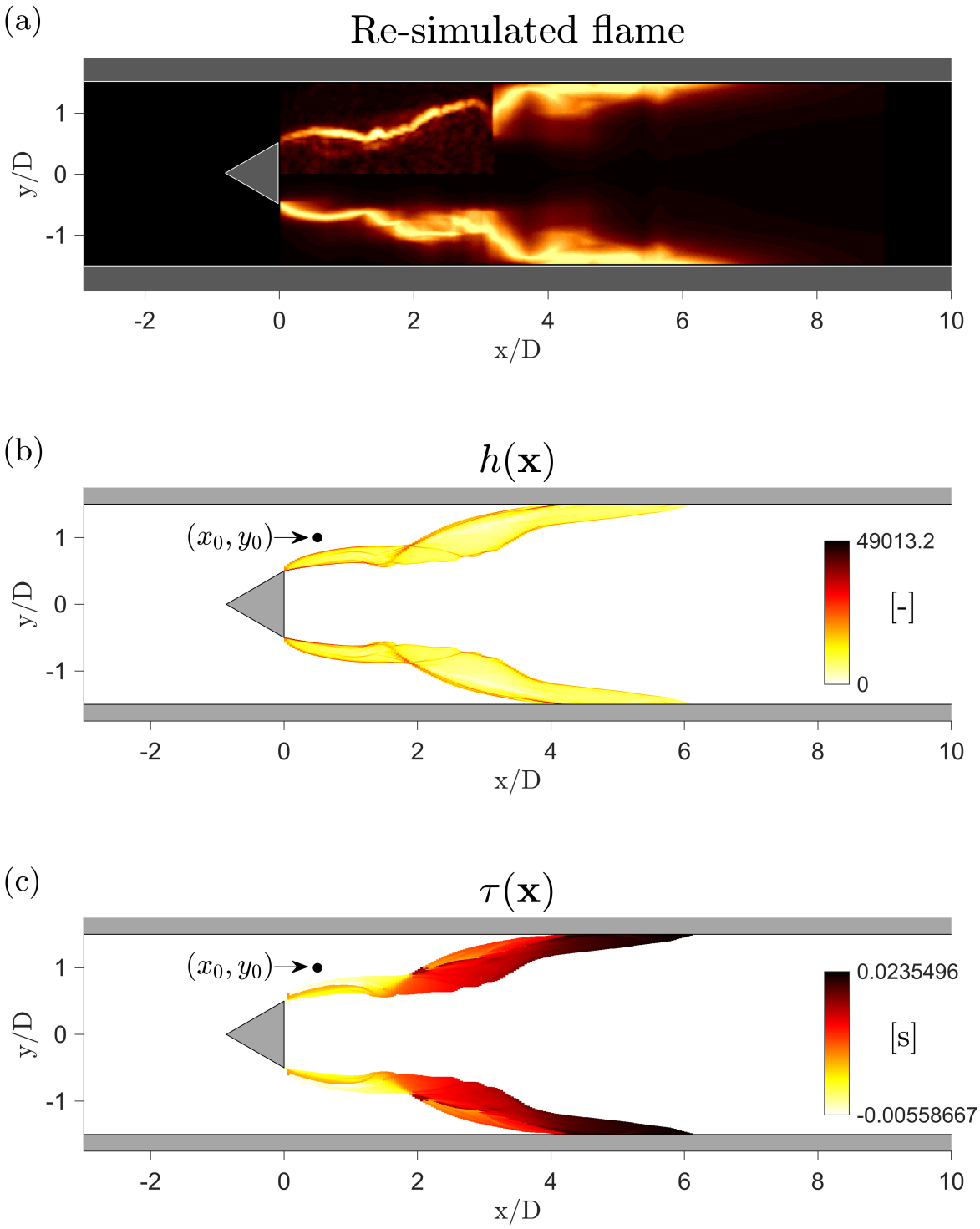


Fig. 4.6 *(a)*: Re-simulated flame shape at $t = 376$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 376$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

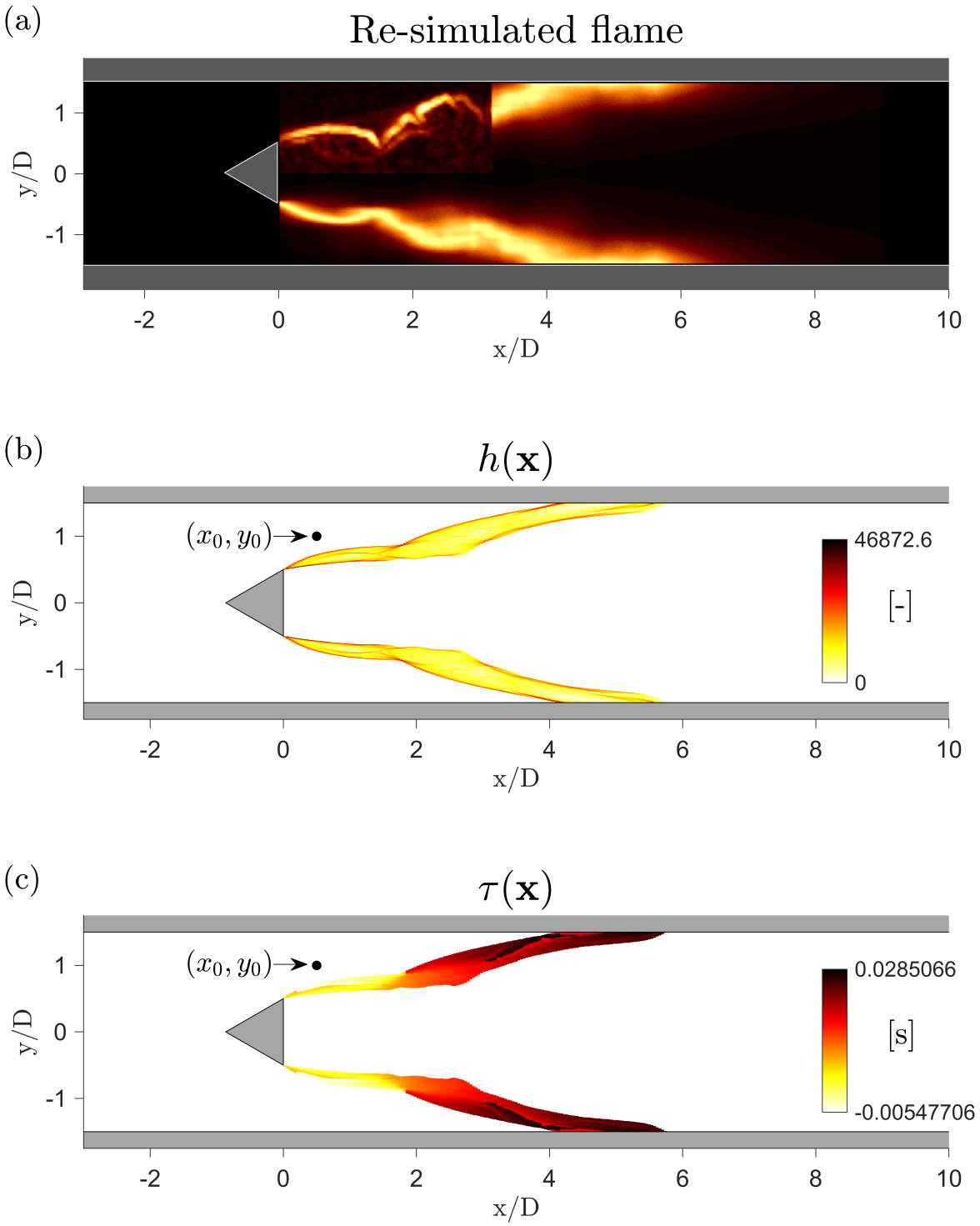


Fig. 4.7 *(a)*: Re-simulated flame shape at $t = 416$. *(b)* and *(c)*: n and τ fields calculated from the re-simulated flame shape at $t = 416$. The centre (x_0, y_0) of the measurement region $w(x, y)$ is also shown. Part of the flame is upstream of the measurement point, which leads to a negative time delay.

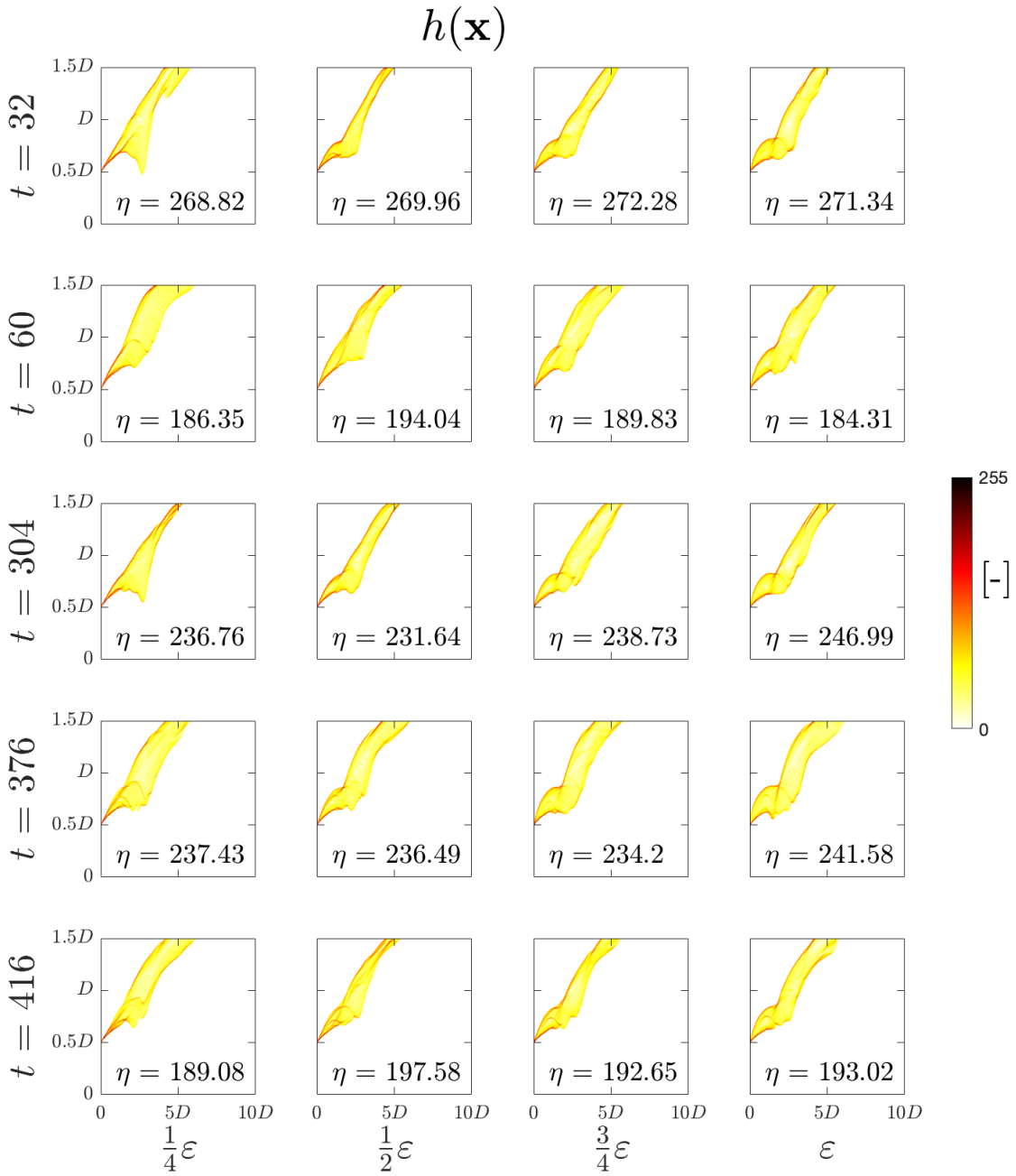


Fig. 4.8 Plots of $h(\mathbf{x})$ at five different timesteps t and at four amplitudes: $\varepsilon/4, \varepsilon/2, 3\varepsilon/4$ and ε, the amplitude inferred by the BayNNE. The values of η are also displayed for each timestep and amplitude.

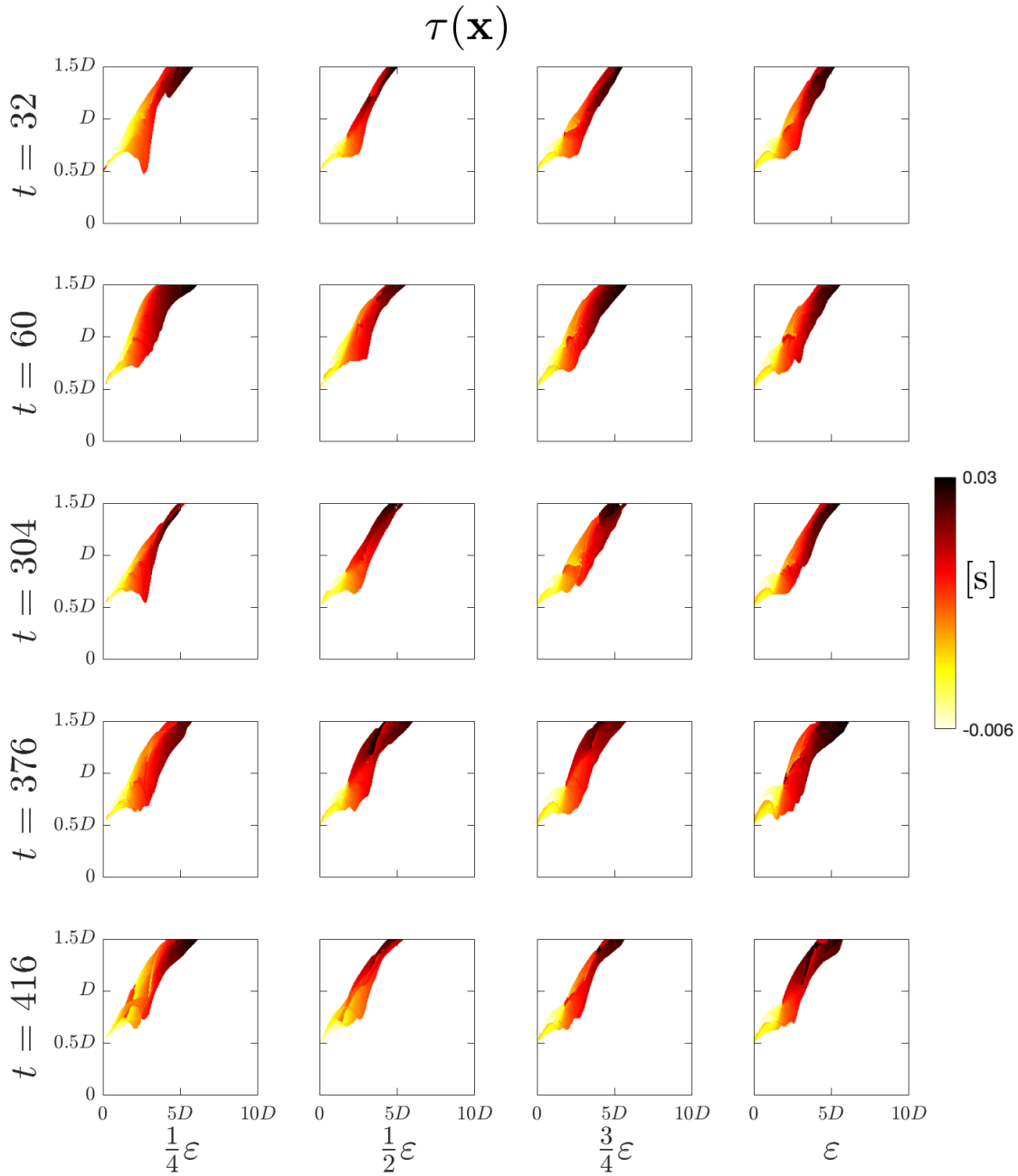


Fig. 4.9 Plots of $\tau(\mathbf{x})$ at five different timesteps t and at four amplitudes: $\varepsilon/4, \varepsilon/2, 3\varepsilon/4$ and ε, the amplitude inferred by the BayNNE.

4.5.3 Helmholtz solver results

Fig. 4.10 shows the first pure acoustic eigenmode (red square) and the thermoacoustic eigenmodes (grey) at the four amplitudes and five timesteps. The grey symbols deviate from the red square, showing that the thermoacoustic effect is active. At each timestep, the thermoacoustic effect stabilises the mode and shifts the frequency more so than was observed in the previous chapter. In all but the $t = 416$ case, there is no clear relationship between the velocity forcing amplitude and the growth rates and frequencies of the thermoacoustic system. The seemingly erroneous values of the growth rate and frequency in the $t = 416$, amplitude $\varepsilon/4$ and $\varepsilon/2$ cases could be due to the larger jumps seen in all regions of the corresponding $\tau(\mathbf{x})$ fields. To improve the quality of these $\tau(\mathbf{x})$ fields, phase averaging over more periods than was performed here would have to be done. This is currently impractical due to storage limitations.

4.6 Conclusions

This chapter built upon the previous chapter by using a more complicated but more physically-informed model of the velocity field in the Volvo burner: the discrete vortex method. The methodology for assimilating the data into the improved model is otherwise unchanged. The heteroscedastic Bayesian neural network ensembles, trained on a library of flame fronts simulated using this new model, learn to infer the five parameters of the velocity model and their uncertainties from ten consecutive flame front snapshots. Because the model is physics-based, it can extrapolate in physical space (i.e. beyond the observation window) and in parameter space (in this case to lower amplitudes) to obtain distributed heat release rate models that can be used successfully in a thermoacoustic model. Once again, we observe that the growth rate and frequency of the thermoacoustic instability do not show a strong dependency on the amplitude of forcing. This suggests that, with this model, a flame transfer function derived at high amplitude, when it is observable, is also valid at low amplitude, when it is not observable.

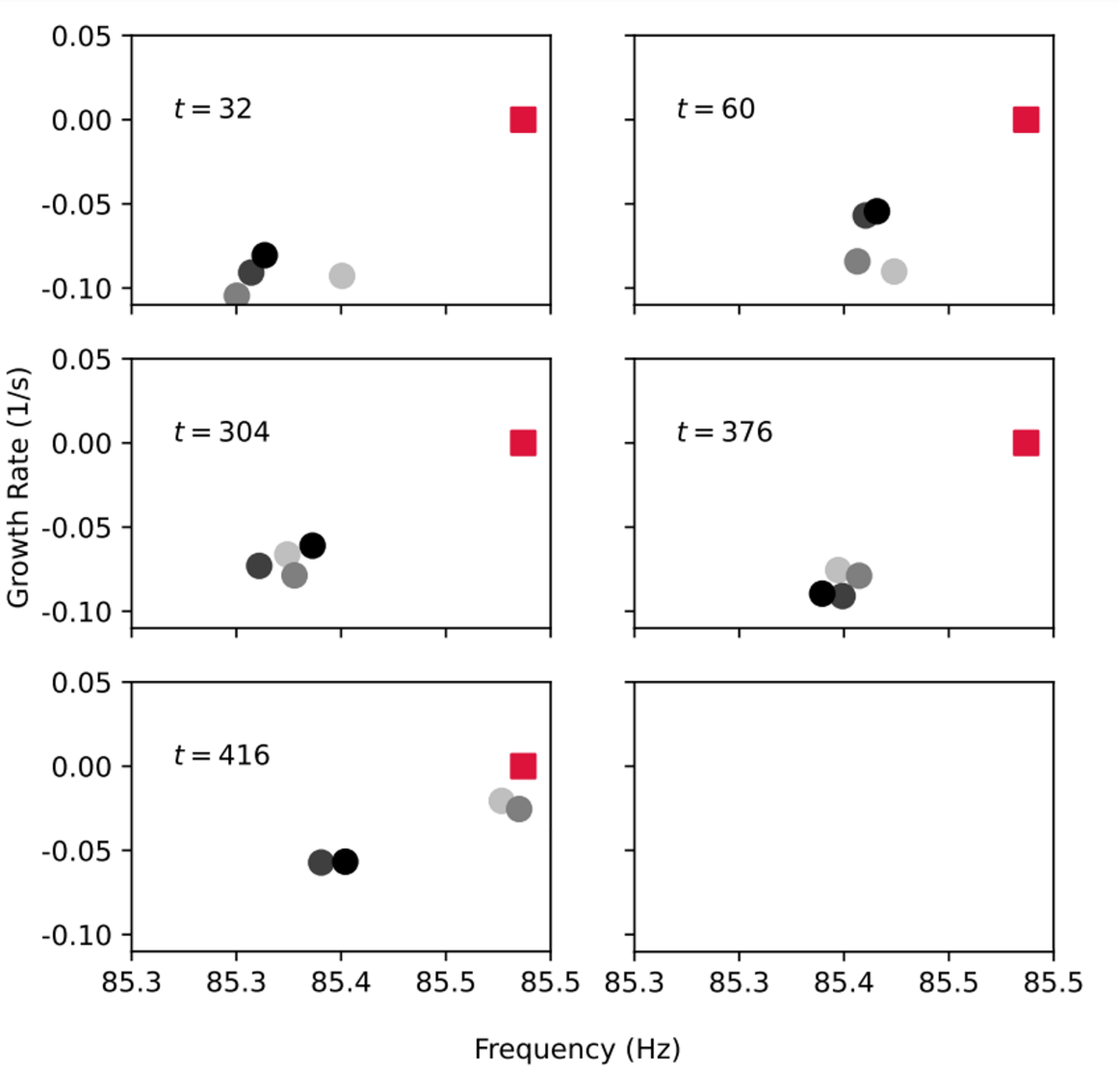


Fig. 4.10 The eigenvalues for the first mode at five timesteps and four amplitudes. The horizontal and vertical axes denote the frequency and the growth rate of the system. The red squares in each subplot show the eigenfrequecies of the system when no flame is present. Each frame corresponds to a different timestep, while the greyscale within each frame corresponds to a different velocity forcing amplitude, ε. The lightest grey corresponds to $\varepsilon/4$ and the darkest to ε. This figure is due to Mr Ekrem Ekici.

Chapter 5

Conclusions

5.1 Summary

This thesis demonstrates a Bayesian machine learning method for assimilating data from progressively more complex burners into physics-based flame models. The method uses heteroscedastic Bayesian neural network ensembles (BayNNEs) trained on a library of simulated flame fronts to infer tunable parameters and their uncertainties from pre-processed images of the flame. Generating the library of simulated flame fronts and then training the neural networks on the library are both computationally expensive steps. However, once the BayNNE has been trained on the forced cycle library it can be used to reliably infer the tunable parameters of the flame front model based on consecutive snapshots of the Bunsen flame experiments in $O(10^{-3})$ seconds. If more snapshots are provided, this method finds the sequence of snapshots that minimises the uncertainty in the parameters.

In chapter 2, the method is first applied to artificial images of flames simulated with a kinematic, physics-based model and compared to the ensemble Kalman filter (EnKF). The EnKF is another data assimilation method which has been demonstrated to work well on this data (both in this chapter and previous publications, such as [47]). Both methods recover the ground-truth parameters, and the flames re-simulated with these inferred parameters match the input flame shapes. Next, both methods are applied to data from a Bunsen burner experiment with varying operating conditions. The re-simulated flame shapes using the inferred parameters from both methods match the experiment images well but model error still exists, as seen in the plots of normalised surface area over time. The EnKF requires hours to infer the parameters of each Bunsen flame. The trained BayNNE provides similarly accurate parameter estimates at a fraction of the computational cost and can be used in real-time tasks, such as the control of thermoacoustic instabilities.

In chapter 3, data from a version of the Volvo burner experiment, which is more complex than the Bunsen burner experiment, is assimilated. The flame is observed over a limited window and exhibits some turbulent, transient behaviour. The velocity model used in the G-equation model of the flame front is adapted to better describe the data: a parameter η is introduced which determines the growth rate of the velocity perturbations with downstream coordinate. The ensemble Kalman filter is unable to assimilate this data because the flame behaviour varies too quickly for the Kalman filter to converge to parameter estimates. The BayNNE, trained on a library of simulated flame fronts, is able to infer the parameters of the physics-based model and their uncertainties robustly. Using the physics-based model, the flame front is extrapolated downstream of the observation window. This allows for the n and τ fields of a distributed $n-\tau$ model for the heat release of the burner to be calculated. These fields are entered into a Helmholtz solver to predict the growth rates and frequencies of the thermoacoustic system. The growth rates and frequencies do not show a strong dependency on the amplitude of velocity forcing. This suggests that a flame transfer function derived at high amplitude, when it is observable, is also valid at low amplitude, when it is not observable.

In chapter 4, the physics-based model of the velocity field used in the previous chapter is replaced with one due to the discrete vortex method. This aims to model better the vortex shedding behind the bluff body in the Volvo burner, which drives the flame response. The flames simulated with this model are more complicated than with the previous model: despite an oscillating base flow, the flame shapes exhibit aperiodic behaviour. The BayNNE trained on a library of simulated flame fronts using this model infers the parameters of the model and their uncertainties robustly, and the n and τ fields of the heat release model are calculated using flame shapes re-simulated with these parameters. By entering these into a Helmholtz solver, the growth rates and frequencies of the thermoacoustic system are calculated and compared with the previous chapter's results: a greater shift in frequency is predicted with the discrete vortex method but, as with the previous model, the growth rates and frequencies do not show a strong dependency on the amplitude of velocity forcing.

To summarise, this thesis has demonstrated the validity and usefulness of a Bayesian machine learning method for parameter inference and uncertainty quantification across different burner experiments and physics-based models of the flames inside these burners. This contribution aims to bridge the gap between cheap, qualitatively-accurate, physics-based models and expensive, state-of-the-art CFD models through data assimilation using principled, Bayesian machine learning.

5.2 Future work

As increasingly large quantities of experimental data become available, the researcher's challenge is to extract useful information without becoming overwhelmed by the quantity of data. Assimilation into physics-based models, as performed in this thesis, is attractive because the models are physically-interpretable and extrapolatable. Firstly, this provides a cheap way to store CFD data: for example, the parameters of the most relevant CFD solution for a given experiment can be extracted cheaply, and the CFD solution then re-calculated. The method may be applied to infer any number of parameters: the number of nodes in the output layers of the neural networks scales linearly with the number of parameters to be inferred. Secondly, it shows how sparse experimental results can be combined with complete numerical results to extrapolate, with defined confidence levels, beyond experimental observations. A promising next step is to apply the Bayesian machine learning method used in this thesis to incrementally more complex data from larger burners used in industry. Ultimately, such a method could be used in real-time within the design process of gas turbines in order to eliminate thermoacoustic instability.

Graph neural networks (GNNs) [67] offer an alternative, promising avenue for simulating physics. GNNs are a class of neural network for processing data that can be represented as graphs. Here, a graph is defined to be a set of vertices and a set of edges connecting these to each other. GNNs encode assumptions about the local properties of the graph. For example, in a social network where users and their relationships are represented by vertices and edges respectively, the size of a user's network may be defined as the number of users immediately connected to the user (the user's neighbourhood). This property holds true for every user in the network, and so is called *invariant* in the graph. Furthermore, this property is not a function of the order in which the users in the neighbourhood are counted: the property is said to be *equivariant*. In physics, invariance and equivariance can be found in the familiar laws of mass conservation (continuity) and Newton's second law. In fluid mechanics, both of these are encapsulated by the Navier-Stokes equations. In the former, mass is conserved everywhere within a fluid, irrespective of the size and position of the local volume over which the law is considered. In the latter, the change in momentum of the local volume due to the forces acting on it does not depend on the order in which the individual forces are considered. By encoding these assumptions, GNNs are particularly well designed for physics simulation. In CFD, the mesh over which computations are performed is a graph. By training on URANS simulations, GNNs learn a surrogate model which can be used to simulate fluid flows much more quickly than the solver which produced the URANS simulations [63, 70]. Furthermore, these surrogate models can be used for gradient-based

optimisation [109]. Augmenting such methods with Bayesian uncertainty quantification is an open problem which, if solved, could provide a useful tool in many a design process.

The volume of data, from simulations and experiments combined with the ever-increasing and ever-cheaper supply of computer processing power means that the intersection of machine learning and scientific modelling contains many opportunities for useful progress. The work presented in this thesis shows one possible approach, and how it can be applied to a particularly long-standing problem in engineering: the modelling, and ultimately the control of, thermoacoustic instabilities.

References

[1] J. W. S. B. Rayleigh, "The explanation of certain acoustical phenomena," *Nature*, vol. 18, no. 455, pp. 319–321, 1878.

[2] M. P. Juniper and R. I. Sujith, "Sensitivity and nonlinearity of thermoacoustic oscillations," *Annual Review of Fluid Mechanics*, vol. 50, pp. 661–689, 2018.

[3] A. A. Putnam and W. R. Dennis, "Burner oscillations of the gauze-tone type," *Journal of the Acoustical Society of America*, vol. 26, no. 5, 1954.

[4] B. T. Chu, "On the energy transfer to small disturbances in fluid flow (Part I)," *Acta Mechanica 1965 1:3*, vol. 1, pp. 215–234, sep 1965.

[5] C. Sondhauss, "Ueber die Schallschwingungen der Luft in erhitzten Glasröhren und in gedeckten Pfeifen von ungleicher Weite," *Annalen der Physik und Chemie*, vol. 155, no. 1, pp. 1–34, 1850.

[6] H. C. Mongia, T. J. Held, G. C. Hsiao, and R. P. Pandalai, "Challenges and progress in controlling dynamics in gas turbine combustors," *Journal of Propulsion and Power*, vol. 19, no. 5, pp. 822–829, 2003.

[7] T. Lieuwen and K. McManus, "Introduction: Combustion dynamics in lean-premixed prevaporized (LPP) gas turbines," *Journal of Propulsion and Power*, vol. 19, no. 5, p. 721, 2003.

[8] Y. Huang and V. Yang, "Dynamics and stability of lean-premixed swirl-stabilized combustion," *Progress in Energy and Combustion Science*, vol. 35, no. 4, pp. 293–364, 2009.

[9] A. P. Dowling, "The calculation of thermoacoustic oscillations," *Journal of Sound and Vibration*, vol. 180, pp. 557–581, mar 1995.

[10] J. G. Aguilar, L. Magri, and M. P. Juniper, "Adjoint-based sensitivity analysis of low-order thermoacoustic networks using a wave-based approach," *Journal of Computational Physics*, vol. 341, pp. 163–181, jul 2017.

[11] F. A. Williams, "Turbulent combustion," in *Frontiers in Applied Mathematics*, ch. 3, pp. 97–131, Society for Industrial and Applied Mathematics, jan 1985.

[12] M. P. Juniper and M. Yoko, "Data assimilation with Laplace's method in thermoacoustics," *Symposium on Thermoacoustics in Combustion: Industry meets Academia*, pp. 1–8, 2021.

[13] L. Crocco, "Research on combustion instability in liquid propellant rockets," *Symposium (International) on Combustion*, vol. 12, pp. 85–99, jan 1969.

[14] L. Y. M. Gicquel, G. Staffelbach, and T. Poinsot, "Large eddy simulations of gaseous flames in gas turbine combustion chambers," *Progress in Energy and Combustion Science*, vol. 38, pp. 782–817, dec 2012.

[15] E. Courtine, L. Selle, and T. Poinsot, "DNS of intrinsic thermoacoustic modes in laminar premixed flames," *Combustion and Flame*, vol. 162, no. 11, pp. 4331–4341, 2015.

[16] A. Giusti and E. Mastorakos, "Turbulent combustion modelling and experiments: recent trends and developments," *Flow, Turbulence and Combustion*, vol. 103, pp. 847–869, nov 2019.

[17] X. Han and A. S. Morgans, "Simulation of the flame describing function of a turbulent premixed flame using an open-source LES solver," *Combustion and Flame*, vol. 162, pp. 1778–1792, may 2015.

[18] C. A. Armitage, R. Balachandran, E. Mastorakos, and R. S. Cant, "Investigation of the nonlinear response of turbulent premixed flames to imposed inlet velocity oscillations," *Combustion and Flame*, vol. 146, pp. 419–436, aug 2006.

[19] C. T. Wey, "Model validation for propulsion - On the TFNS subgrid models for a bluff body stabilized flame." 2017.

[20] A. Sadiki, A. Maltsev, B. Wegner, F. Flemming, A. Kempf, and J. Janicka, "Unsteady methods (URANS and LES) for simulation of combustion systems," *International Journal of Thermal Sciences*, vol. 45, pp. 760–773, 2006.

[21] D. M. Arthur, A. P. Dowling, R. S. Cant, and M. Zhu, "Forced flame behaviour in a spray combustor," in *Proceedings of ASME Turbo Expo 2008: Power for Land, Sea and Air*, (Berlin, Germany), jun 2008.

[22] T. Poinsot and D. P. Veynante, *Theoretical and numerical combustion*. Edwards, 3rd ed., 2012.

[23] S. Ducruix, D. Durox, and S. M. Candel, "Theoretical and experimental determinations of the transfer function of a laminar premixed flame," *Proceedings of the Combustion Institute*, vol. 28, pp. 765–773, jan 2000.

[24] T. Schuller, S. Ducruix, D. Durox, and S. M. Candel, "Modeling tools for the prediction of premixed flame transfer functions," *Proceedings of the Combustion Institute*, vol. 29, no. 1, pp. 107–113, 2002.

[25] A. P. Dowling, "Nonlinear self-excited oscillations of a ducted flame," *Combustion and Flame*, vol. 346, pp. 271–290, 1997.

[26] A. P. Dowling, "A kinematic model of a ducted flame," *Journal of Fluid Mechanics*, vol. 394, pp. 51–72, 1999.

[27] M. Fleifil, A. M. Annaswamy, Z. A. Ghoneim, and A. F. Ghoniem, "Response of a laminar premixed flame to flow oscillations: A kinematic model and thermoacoustic instability results," *Combustion and Flame*, vol. 106, no. 4, pp. 487–510, 1996.

[28] Preetham, H. Santosh, and T. Lieuwen, "Dynamics of laminar premixed flames forced by harmonic velocity disturbances," *Journal of Propulsion and Power*, vol. 24, no. 6, pp. 1390–1402, 2008.

[29] K. Kashinath, S. Hemchandra, and M. P. Juniper, "Nonlinear thermoacoustics of ducted premixed flames: The influence of perturbation convection speed," *Combustion and Flame*, vol. 160, no. 12, pp. 2856–2865, 2013.

[30] K. T. Feldman, "Review of the literature on Rijke thermoacoustic phenomena," *Journal of Sound and Vibration*, vol. 7, pp. 83–89, jan 1968.

[31] R. L. Raun, M. W. Beckstead, J. C. Finlinson, and K. P. Brooks, "A review of Rijke tubes, Rijke burners and related devices," *Progress in Energy and Combustion Science*, vol. 19, pp. 313–364, jan 1993.

[32] G. Bisio and G. Rubatto, "Sondhauss and Rijke oscillations - Thermodynamic analysis, possible applications and analogies," *Energy*, vol. 24, pp. 117–131, feb 1999.

[33] P. L. Rijke, "Notice of a new method of causing a vibration of the air contained in a tube open at both ends ," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 17, pp. 419–422, jun 1859.

[34] A. W. Caswell, B. A. Rankin, B. C. Huelskamp, A. C. Lynch, V. Belovich, and J. R. Gord, "Spatiotemporal characterization of bluff-body stabilized turbulent premixed flames using simultaneous high-repetition-rate OH PLIF and PIV measurements," *53rd AIAA Aerospace Sciences Meeting*, pp. 1–12, 2015.

[35] C. A. Fugger, T. Yi, J. P. Sykes, A. W. Caswell, B. A. Rankin, J. D. Miller, and J. R. Gord, "The structure and dynamics of a bluff-body stabilized premixed reacting flow," *AIAA Aerospace Sciences Meeting*, no. 210059, 2018.

[36] D. J. C. MacKay, *Information Theory, Inference, and Learning Algorithms*. Cambridge University Press, 2003.

[37] C. Sammut and G. I. Webb, "Bayesian model averaging," in *Encyclopedia of Machine Learning*, pp. 81–81, Springer US, 2011.

[38] D. J. C. MacKay, *Bayesian methods for adaptive methods*. PhD thesis, California Institute of Technology, 1991.

[39] R. M. Neal, *Bayesian Learning for Neural Networks*. PhD thesis, University of Toronto, 1995.

[40] A. G. Wilson and P. Izmailov, "Bayesian deep learning and a probabilistic perspective of generalization," in *Advances in Neural Information Processing Systems (NeurIPS)*, (Virtual), 2020.

[41] W. Penny, K. Friston, J. Ashburner, S. Kiebel, and T. Nichols, *Statistical Parametric Mapping: The Analysis of Functional Brain Images*. Elsevier Ltd, 2007.

[42] H. Yu, *Inverse Problems in Thermoacoustics*. PhD thesis, University of Cambridge, 2020.

[43] R. E. Kalman, "A new approach to linear filtering and prediction problems," *Journal of Fluids Engineering, Transactions of the ASME*, vol. 82, no. 1, pp. 35–45, 1960.

[44] G. Evensen, "Sequential data assimilation with a nonlinear quasi-geostrophic model using Monte Carlo methods to forecast error statistics," *Journal of Geophysical Research*, vol. 99, pp. 10143–10162, may 1994.

[45] A. C. Miller, N. J. Foti, and R. P. Adams, "Variational boosting: Iteratively refining posterior approximations," in *34th International Conference on Machine Learning (ICML)*, vol. 5, (Sydney, Australia), pp. 3732–3747, 2017.

[46] H. Yu, M. P. Juniper, and L. Magri, "Combined state and parameter estimation in level-set methods," *Journal of Computational Physics*, vol. 399, pp. 1–51, 2019.

[47] H. Yu, M. P. Juniper, and L. Magri, "A data-driven kinematic model of a ducted premixed flame," *Proceedings of the Combustion Institute*, vol. 38, no. 4, pp. 6231–6239, 2021.

[48] K. Pearson, "On lines and planes of closest fit to systems of points in space," *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, vol. 2, pp. 559–572, nov 1901.

[49] P. Del Moral, "Nonlinear filtering: Interacting particle resolution," *Comptes Rendus de l'Académie des Sciences - Series I - Mathematics*, vol. 325, pp. 653–658, sep 1997.

[50] C. Vergé, C. Dubarry, P. Del Moral, and E. Moulines, "On parallel implementation of sequential Monte Carlo methods: the island particle model," *Stat Comput*, vol. 25, pp. 243–260, 2015.

[51] F. Garita, *Physics-Based Statistical Learning in Thermoacoustics*. PhD thesis, University of Cambridge, 2021.

[52] S. Li, B. Yang, and F. Qi, "Accelerate global sensitivity analysis using artificial neural network algorithm: Case studies for combustion kinetic model," *Combustion and Flame*, vol. 168, pp. 53–64, jun 2016.

[53] J. An, G. He, F. Qin, R. Li, and Z. Huang, "A new framework of global sensitivity analysis for the chemical kinetic model using PSO-BPNN," *Computers & Chemical Engineering*, vol. 112, pp. 154–164, apr 2018.

[54] J. Wang, Z. Zhou, K. Lin, C. K. Law, and B. Yang, "Facilitating Bayesian analysis of combustion kinetic models with artificial neural networks," *Combustion and Flame*, vol. 213, pp. 87–97, mar 2020.

[55] R. S. Freitas, F. A. Rochinha, D. Mira, and X. Jiang, "Parametric and model uncertainties induced by reduced order chemical mechanisms for biogas combustion," *Chemical Engineering Science*, vol. 227, p. 115949, dec 2020.

[56] W. Ji and S. Deng, "Autonomous discovery of unknown reaction pathways from data by chemical reaction neural networks," *Journal of Physical Chemistry A*, vol. 125, pp. 1082–1092, feb 2021.

[57] M. T. Henry de Frahan, S. Yellapantula, R. King, M. S. Day, and R. W. Grout, "Deep learning for presumed probability density function models," *Combustion and Flame*, vol. 208, pp. 436–450, oct 2019.

[58] Z. X. Chen, S. Iavarone, G. Ghiasi, V. Kannan, G. D'Alessio, A. Parente, and N. Swaminathan, "Application of machine learning for filtered density function closure in MILD combustion," *Combustion and Flame*, vol. 225, pp. 160–179, mar 2021.

[59] M. Raissi and G. E. Karniadakis, "Hidden physics models: Machine learning of nonlinear partial differential equations," *Journal of Computational Physics*, vol. 357, pp. 125–141, mar 2018.

[60] M. Raissi, P. Perdikaris, and G. E. 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, feb 2019.

[61] M. Raissi, A. Yazdani, and G. E. Karniadakis, "Hidden fluid mechanics: Learning velocity and pressure fields from flow visualizations," *Science*, vol. 367, no. 6481, pp. 1026–1030, 2020.

[62] Y. Zhu, N. Zabaras, P. S. Koutsourelakis, and P. Perdikaris, "Physics-constrained deep learning for high-dimensional surrogate modeling and uncertainty quantification without labeled data," *Journal of Computational Physics*, vol. 394, pp. 56–81, oct 2019.

[63] A. Sanchez-Gonzalez, J. Godwin, T. Pfaff, R. Ying, J. Leskovec, and P. W. Battaglia, "Learning to simulate complex physics with graph networks," in *37th International Conference on Machine Learning (ICML)*, (Virtual), 2020.

[64] S. He, Y. Li, Y. Feng, S. Ho, S. Ravanbakhsh, W. Chen, and B. Póczos, "Learning to predict the cosmological structure formation," *Proceedings of the National Academy of Sciences of the United States of America*, vol. 116, no. 28, pp. 13825–13832, 2019.

[65] T. K. Ho, "Random decision forests," *Proceedings of the International Conference on Document Analysis and Recognition (ICDAR)*, vol. 1, pp. 278–282, 1995.

[66] S. Hochreiter and J. Schmidhuber, "Long short-term memory," *Neural Computation*, vol. 9, pp. 1735–1780, nov 1997.

[67] F. Scarselli, M. Gori, A. C. Tsoi, M. Hagenbuchner, and G. Monfardini, "The graph neural network model," *IEEE Transactions on Neural Networks*, vol. 20, pp. 61–80, jan 2009.

[68] L. Ladický, S. Jeong, B. Solenthaler, M. Pollefeys, and M. Gross, "Data-driven fluid simulations using regression forests," *ACM Transactions on Graphics (TOG)*, vol. 34, p. 1, nov 2015.

[69] S. Wiewel, M. Becher, and N. Thuerey, "Latent space physics: Towards learning the temporal evolution of fluid flow," *Computer Graphics Forum*, vol. 38, pp. 71–82, may 2019.

[70] T. Pfaff, M. Fortunato, A. Sanchez-Gonzalez, and P. W. Battaglia, "Learning mesh-based simulation with graph networks," in *38th International Conference on Machine Learning (ICML)*, (Virtual), 2021.

[71] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016.

[72] K. Hornik, M. Stinchcombe, and H. White, "Multilayer feedforward networks are universal approximators," *Neural Networks*, vol. 2, pp. 359–366, jan 1989.

[73] D. A. Nix and A. S. Weigend, "Estimating the mean and variance of the target probability distribution," *IEEE International Conference on Neural Networks - Conference Proceedings*, vol. 1, pp. 55–60, 1994.

[74] B. Lakshminarayanan, A. Pritzel, and C. Blundell, "Simple and scalable predictive uncertainty estimation using deep ensembles," in *31st Conference on Neural Information Processing Systems (NeurIPS)*, (Long Beach, CA, USA), 2017.

[75] T. Pearce, F. Leibfried, A. Brintrup, M. Zaki, and A. Neely, "Uncertainty in neural networks: approximately Bayesian ensembling," in *Proceedings of the 23rd International Conference on Artificial Intelligence and Statistics (AISTATS)*, vol. 108, (Virtual), 2020.

[76] U. Sengupta, M. Amos, J. S. Hosking, C. E. Rasmussen, M. P. Juniper, and P. J. Young, "Ensembling geophysical models with Bayesian neural networks," in *34th Conference on Neural Information Processing Systems (NeurIPS)*, (Vancouver, Canada), 2020.

[77] Y. Gal and Z. Ghahramani, "Dropout as a Bayesian approximation: representing model uncertainty in deep learning," in *Proceedings of the 33rd International Conference on Machine Learning*, (New York, NY, USA), W&CP, 2016.

[78] N. Srivastava, G. E. Hinton, A. Krizhevsky, and R. Salakhutdinov, "Dropout: a simple way to prevent neural networks from overfitting," *Journal of Machine Learning Research*, vol. 15, pp. 1929–1958, 2014.

[79] J. Hron, A. G. d. G. Matthews, and Z. Ghahramani, "Variational Bayesian dropout: pitfalls and fixes," in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, 2018.

[80] Y. Yao, A. Vehtari, D. Simpson, and A. Gelman, "Yes, but did it work?: Evaluating variational inference," in *Proceedings of the 35th International Conference on Machine Learning*, PMLR, 2018.

[81] U. Sengupta, M. L. Croci, and M. P. Juniper, "Real-time parameter inference in reduced-order flame models with heteroscedastic Bayesian neural network ensembles," in *Machine Learning and the Physical Sciences workshop at the 34th Conference on Neural Information Processing Systems (NeurIPS)*, (Virtual), 2020.

[82] M. L. Croci, U. Sengupta, and M. P. Juniper, "Online parameter inference for the simulation of a Bunsen flame using heteroscedastic Bayesian neural network ensembles," in *Deep Learning for Simulation workshop at the 9th International Conference Conference on Learning Representations (ICLR)*, (Virtual), 2021.

[83] M. L. Croci, U. Sengupta, and M. P. Juniper, "Data assimilation using heteroscedastic Bayesian neural network ensembles for reduced-order flame models," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12746 LNCS, pp. 408–419, 2021.

[84] S. Hemchandra, *Dynamics of Turbulent Premixed Flames in Acoustic Fields*. PhD thesis, University of Cambridge, 2009.

[85] H. Yu, T. Jaravel, M. Ihme, M. P. Juniper, and L. Magri, "Data assimilation and optimal calibration in nonlinear models of flame dynamics," *Journal of Engineering for Gas Turbines and Power*, vol. 141, no. 12, pp. 1–10, 2019.

[86] D. G. Goodwin, H. K. Moffat, I. Schoegl, R. L. Speth, and B. W. Weber, "Cantera: An object-oriented software toolkit for chemical kinetics, thermodynamics, and transport processes," 2022.

[87] F. Baillot, D. Durox, and R. Prud'homme, "Experimental and theoretical study of a premixed vibrating flame," *Combustion and Flame*, vol. 88, pp. 149–168, feb 1992.

[88] K. Kashinath, L. K. Li, and M. P. Juniper, "Forced synchronization of periodic and aperiodic thermoacoustic oscillations: Lock-in, bifurcations and open-loop control," *Journal of Fluid Mechanics*, vol. 838, pp. 690–714, mar 2018.

[89] X. Luo and I. Hoteit, "Robust ensemble filtering and its relation to covariance inflation in the ensemble Kalman filter," *Monthly Weather Review*, vol. 139, pp. 3938–3953, dec 2011.

[90] M. L. Croci, U. Sengupta, and M. P. Juniper, "Real-time parameter inference of nonlinear bluff-body-stabilized flame models using Bayesian neural network ensembles," in *Symposium on Thermoacoustics in Combustion: Industry meets Academia (SoTiC)*, (Munich, Germany), 2021.

[91] M. L. Croci, U. Sengupta, and M. P. Juniper, "Bayesian inference in physics-based nonlinear flame models," in *Deep Learning and Inverse Problems workshop at the 35th Conference on Neural Information Processing Systems (NeurIPS)*, (Virtual), 2021.

[92] B. Kiel, K. Garwick, A. C. Lynch, J. R. Gord, and T. Meyer, "Non-reacting and combusting flow investigation of bluff bodies in cross flow," *Collection of Technical Papers - AIAA/ASME/SAE/ASEE 42nd Joint Propulsion Conference*, vol. 11, pp. 8743–8753, 2006.

[93] B. Kiel, K. Garwick, J. R. Gord, J. D. Miller, A. C. Lynch, R. Hill, and S. Phillips, "A detailed investigation of bluff body stabilized flames," *Collection of Technical Papers - 45th AIAA Aerospace Sciences Meeting*, vol. 3, pp. 2019–2028, 2007.

[94] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, "Learning representations by back-propagating errors," *Nature*, vol. 323, no. 6088, pp. 533–536, 1986.

[95] K. Cho, B. van Merriënboer, D. Bahdanau, and Y. Bengio, "On the properties of neural machine translation: encoder–decoder approaches," *Proceedings of SSST 2014 - 8th Workshop on Syntax, Semantics and Structure in Statistical Translation*, pp. 103–111, 2014.

[96] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin, "Attention is all you need," *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[97] T. Pearce, M. Zaki, and A. Neely, "Bayesian neural network ensembles," in *Third workshop on Bayesian Deep Learning at the 32nd Conference on Neural Information Processing Systems (NeurIPS)*, (Montréal, Canada), 2018.

[98] K. Fukushima, "Biological cybernetics neocognitron: A self-organizing neural network model for a mechanism of pattern recognition unaffected by shift in position," *Biol. Cybernetics*, vol. 36, p. 202, 1980.

[99] Y. LeCun and Y. Bengio, "Convolutional networks for images, speech, and time-series," in *The handbook of brain theory and neural networks 3361*, ch. 10, 1995.

[100] M. P. Juniper, "Sensitivity analysis of thermoacoustic instability with adjoint Helmholtz solvers," *Physical Review Fluids*, vol. 3, 2018.

[101] F. Nicoud, L. Benoit, C. Sensiau, and T. Poinsot, "Acoustic modes in combustors with complex impedances and multidimensional active flames," *AIAA Journal*, vol. 45, no. 2, 2007.

[102] M. A. Herráez, D. R. Burton, M. J. Lalor, and M. A. Gdeisat, "Fast two-dimensional phase-unwrapping algorithm based on sorting by reliability following a noncontinuous path," *Applied optics*, vol. 41, p. 7437, dec 2002.

[103] S. Falco, *Shape optimization for thermoacoustic instability with an adjoint Helmholtz solver*. PhD thesis, University of Cambridge, 2021.

[104] D. P. Kingma and J. Lei Ba, "Adam: A method for stochastic optimization," in *3rd International Conference on Learning Representations (ICLR)*, (San Diego, CA, USA), 2015.

[105] M. L. Croci, J. V. Vasanth, U. Sengupta, E. Ekici, and M. P. Juniper, "Bayesian parameter inference of a vortically perturbed flame model for the prediction of thermoacoustic instability," in *AI for Science workshop at the 36th Conference on Neural Information Processing Systems (NeurIPS)*, (New Orleans, LA, USA), 2022.

[106] A. Leonard, "Vortex methods for flow simulation," *Journal of Computational Physics*, vol. 37, pp. 289–335, oct 1980.

[107] G.-H. Cottet and P. D. Koumoutsakos, *Vortex Methods*. Cambridge University Press, mar 2000.

[108] T. A. Driscoll and L. N. Trefethen, *Schwartz-Christoffel mapping*. Cambridge University Press, 2002.

[109] K. R. Allen, T. Lopez-Guevara, K. Stachenfeld, A. Sanchez-Gonzalez, P. W. Battaglia, J. Hamrick, and T. Pfaff, "Physical design using differentiable learned simulators." feb 2022.

[110] L. M. Milne-Thomson, *Theoretical hydrodynamics*. New York: Dover Publications, 5 ed., 1968.

[111] M. Pastoor, L. Henning, B. R. Noack, R. King, and G. Tadmor, "Feedback shear layer control for bluff body drag reduction," *Journal of Fluid Mechanics*, vol. 608, pp. 161–196, 2008.

[112] M. Perlman, "On the accuracy of vortex methods," *Journal of Computational Physics*, vol. 59, pp. 200–223, jun 1985.

[113] R. R. Clements, "An inviscid model of two-dimensional vortex shedding," *Journal of Fluid Mechanics*, vol. 57, no. 2, pp. 321–336, 1973.

[114] A. F. Ghoniem and Y. Cagnon, "Vortex simulation of laminar recirculating flow," *Journal of Computational Physics*, vol. 68, pp. 346–377, feb 1987.

Appendix A

Transients in the G-equation model

A.1 Linear perturbation analysis of the G-equation model of the Bunsen flame

The following analysis is based upon work in [27], in which the results of a linear perturbation analysis of the G-equation model using a purely vertical velocity perturbation are stated. This analysis considers instead a velocity perturbation with both horizontal and vertical components, and uses the same Laplace transform method suggested by [27] to solve the governing PDE.

The G-equation is:

$$\frac{\partial G}{\partial t} + \mathbf{v} \cdot \nabla G = s_L \mid \nabla G \mid \ . \tag{A.1}$$

If the flame position can be represented as a single-valued function of x then $G = y - \psi(x,t) = 0$ and $\mathbf{v} = [u\ v]^T$. Substituting into the above gives:

$$-\frac{\partial \psi}{\partial t} + v - u\frac{\partial \psi}{\partial x} = s_L \left(1 + \left(\frac{\partial \psi}{\partial x}\right)^2\right)^{1/2} \ . \tag{A.2}$$

Steady state conditions have $\psi = \bar{\psi} \neq f(t)$, $u = 0$ and $v = V$, which leads to:

$$V = s_L \left(1 + \left(\frac{\partial \bar{\psi}}{\partial x}\right)^2\right)^{1/2} \ . \tag{A.3}$$

In the perturbed state $u = u'$, $v = V + v'$ and $\psi = \bar{\psi} + \psi'$. Substituting this into (2) gives:

$$-\frac{\partial(\bar{\psi}+\psi')}{\partial t} + V + v' - u'\frac{\partial(\bar{\psi}+\psi')}{\partial x} = s_L\left(1+\left(\frac{\partial(\bar{\psi}+\psi')}{\partial x}\right)^2\right)^{1/2}, \tag{A.4}$$

or

$$-\frac{\partial\psi'}{\partial t} + V + v' - u'\frac{\partial\bar{\psi}}{\partial x} = s_L\left(1+\left(\frac{\partial(\bar{\psi}+\psi')}{\partial x}\right)^2\right)^{1/2}. \tag{A.5}$$

To linearise the RHS requires the approximation, for small z':

$$f(z+z') \approx f(z) + \frac{\mathrm{d}f(z)}{\mathrm{d}z}z', \tag{A.6}$$

where

$$f(z) = \left(1+z^2\right)^{1/2}. \tag{A.7}$$

Therefore

$$f(z+z') \approx \left(1+z^2\right)^{1/2} + \frac{z}{\left(1+z^2\right)^{1/2}}z'. \tag{A.8}$$

Thus

$$\left(1+\left(\frac{\partial(\bar{\psi}+\psi')}{\partial x}\right)^2\right)^{1/2} \approx \left(1+\left(\frac{\partial\bar{\psi}}{\partial x}\right)^2\right)^{1/2} + \frac{\frac{\partial\bar{\psi}}{\partial x}}{\left(1+\left(\frac{\partial\bar{\psi}}{\partial x}\right)^2\right)^{1/2}}\frac{\partial\psi'}{\partial x}. \tag{A.9}$$

Equation (A.5) becomes

$$-\frac{\partial\psi'}{\partial t} + V + v' - u'\frac{\partial\bar{\psi}}{\partial x} = s_L\left(\left(1+\left(\frac{\partial\bar{\psi}}{\partial x}\right)^2\right)^{1/2} + \frac{\frac{\partial\bar{\psi}}{\partial x}}{\left(1+\left(\frac{\partial\bar{\psi}}{\partial x}\right)^2\right)^{1/2}}\frac{\partial\psi'}{\partial x}\right), \tag{A.10}$$

or, using the steady state result:

$$-\frac{\partial\psi'}{\partial t} + v' - u'\frac{\partial\bar{\psi}}{\partial x} = \frac{s_L\frac{\partial\bar{\psi}}{\partial x}}{\left(1+\left(\frac{\partial\bar{\psi}}{\partial x}\right)^2\right)^{1/2}}\frac{\partial\psi'}{\partial x}, \tag{A.11}$$

and hence:

$$\boxed{\frac{\partial\psi'}{\partial t} - \bar{s}_L\frac{\partial\psi'}{\partial x} = v' + \beta u'}. \tag{A.12}$$

Where we note that $-\frac{\partial \bar{\psi}}{\partial x} \equiv \beta \geq 0$, the ratio of flame length to flame width, and define:

$$\bar{s}_L = \frac{\beta}{(1+\beta^2)^{1/2}} s_L \, . \tag{A.13}$$

We prescribe the following continuity-obeying velocity perturbations:

$$u' = -\frac{K\mathrm{St}\varepsilon}{\beta} x \cos(\mathrm{St}(Ky - t)) \, ; \tag{A.14}$$

$$v' = \varepsilon \sin(\mathrm{St}(Ky - t)) \, . \tag{A.15}$$

Substituting these into (A.12):

$$\frac{\partial \psi'}{\partial t} - \bar{s}_L \frac{\partial \psi'}{\partial x} = \varepsilon \sin(\mathrm{St}(Ky - t)) - KSt\varepsilon x \cos(\mathrm{St}(Ky - t)) \, . \tag{A.16}$$

This PDE is of the form:

$$\frac{\partial \psi'}{\partial t} - \bar{s}_L \frac{\partial \psi'}{\partial x} = a_1 \sin(\omega t + b) + a_2 x \cos(\omega t + b) \, , \tag{A.17}$$

where $a_1 = -\varepsilon$, $a_2 = -K\mathrm{St}\varepsilon$, $\omega = \mathrm{St}$ and $b = -\mathrm{St}Ky$. As this PDE is linear, we can separate the solution $\psi' = \psi'_1 + \psi'_2$:

$$\frac{\partial \psi'_1}{\partial t} - \bar{s}_L \frac{\partial \psi'_1}{\partial x} = a_1 \sin(\omega t + b) \, ; \tag{A.18}$$

$$\frac{\partial \psi'_2}{\partial t} - \bar{s}_L \frac{\partial \psi'_2}{\partial x} = a_2 x \cos(\omega t + b) \, , \tag{A.19}$$

and consider these problems individually. We now tackle PDE (A.18) by applying the Laplace transform to both sides, noting that x and y remain parameters when the Laplace transform is taken over variable t. PDE (A.18) becomes:

$$s\Psi'_1 - \psi'_1(x, t = 0) - \bar{s}_L \frac{\mathrm{d}\Psi'_1}{\mathrm{d}x} = a_1 \frac{s \sin b + \omega \cos b}{s^2 + \omega^2} \tag{A.20}$$

using standard results. We note that the initial condition for ψ'_1 is $\psi'_1(x, t = 0) = 0$. This leads to:

$$-\frac{s}{\bar{s}_L}\Psi'_1 + \frac{\mathrm{d}\Psi'_1}{\mathrm{d}x} = -\frac{a_1}{\bar{s}_L} \frac{s \sin b + \omega \cos b}{s^2 + \omega^2} \, . \tag{A.21}$$

This is solved using the integration factor $\exp(-sx/\bar{s}_L)$:

$$\frac{\mathrm{d}}{\mathrm{d}x}(e^{-sx/\bar{s}_L}\Psi_1') = -\frac{a_1}{\bar{s}_L}\frac{s\sin b+\omega\cos b}{s^2+\omega^2}e^{-sx/\bar{s}_L} \tag{A.22}$$

$$e^{-sx/\bar{s}_L}\Psi_1' = -\frac{a_1}{\bar{s}_L}\frac{s\sin b+\omega\cos b}{s^2+\omega^2}\int e^{-sx/\bar{s}_L}\,\mathrm{d}x \tag{A.23}$$

$$e^{-sx/\bar{s}_L}\Psi_1' = -\frac{a_1}{\bar{s}_L}\frac{s\sin b+\omega\cos b}{s^2+\omega^2}\left[-\frac{\bar{s}_L}{s}e^{-sx/\bar{s}_L}+C\right] \tag{A.24}$$

or:

$$\Psi_1' = \frac{a_1}{\bar{s}_L}\frac{s\sin b+\omega\cos b}{s^2+\omega^2}\left[\frac{\bar{s}_L}{s}-e^{sx/\bar{s}_L}C\right]\ . \tag{A.25}$$

Using the initial condition $\Psi_1'(x=R,y,s)=0$ results in:

$$\Psi_1' = \frac{a_1}{\bar{s}_L}\frac{s\sin b+\omega\cos b}{s^2+\omega^2}\left[\frac{\bar{s}_L}{s}-\frac{\bar{s}_L}{s}e^{-s(R-x)/\bar{s}_L}\right] \tag{A.26}$$

$$\Psi_1' = a_1\frac{s\sin b+\omega\cos b}{s(s^2+\omega^2)}\left[1-e^{-s(R-x)/\bar{s}_L}\right]\ , \tag{A.27}$$

which is of the form:

$$\Psi_1' = F_1(s)-e^{-cs}F_1(s)\ . \tag{A.28}$$

This implies a solution of the form:

$$\psi_1'(x,y,t) = f_1(x,y,t)-f_1(x,y,t-c)H(t-c)\ , \tag{A.29}$$

where $c=\frac{R-x}{\bar{s}_L}$ and $H(t-c)$ is the Heaviside step function:

$$H(t-c) = \begin{cases} 1 & t\geq c \\ 0 & t<c \end{cases}$$

and f_1 is found to be (see appendix A.2.1):

$$f_1(t) = \frac{a_1}{\omega}(\cos b-\cos(\omega t+b))\ . \tag{A.30}$$

We next tackle PDE (A.19):

$$\frac{\partial\psi_2'}{\partial t}-\bar{s}_L\frac{\partial\psi_2'}{\partial x} = a_2 x\cos(\omega t+b)\ .$$

Applying the Laplace transform to both sides:

$$s\Psi_2' - \psi_2'(x,t=0) - \bar{s}_L \frac{\mathrm{d}\Psi_2'}{\mathrm{d}x} = a_2 \frac{s\cos b - \omega \sin b}{s^2+\omega^2} x \,. \tag{A.31}$$

This leads to:

$$-\frac{s}{\bar{s}_L}\Psi_2' + \frac{\mathrm{d}\Psi_2'}{\mathrm{d}x} = -\frac{a_2}{\bar{s}_L}\frac{s\cos b - \omega \sin b}{s^2+\omega^2} x \,, \tag{A.32}$$

which is again solved using the integration factor $\exp(-sx/\bar{s}_L)$:

$$\frac{\mathrm{d}}{\mathrm{d}x}(e^{-sx/\bar{s}_L}\Psi_2') = -\frac{a_2}{\bar{s}_L}\frac{s\cos b - \omega \sin b}{s^2+\omega^2} x e^{-sx/\bar{s}_L} \tag{A.33}$$

$$e^{-sx/\bar{s}_L}\Psi_2' = -\frac{a_2}{\bar{s}_L}\frac{s\cos b - \omega \sin b}{s^2+\omega^2} \int x e^{-sx/\bar{s}_L}\,\mathrm{d}x \tag{A.34}$$

$$e^{-sx/\bar{s}_L}\Psi_2' = -\frac{a_2}{\bar{s}_L}\frac{s\cos b - \omega \sin b}{s^2+\omega^2} \left[-\frac{\bar{s}_L}{s} x e^{-sx/\bar{s}_L} - \left(\frac{\bar{s}_L}{s}\right)^2 e^{-sx/\bar{s}_L} + C \right] \tag{A.35}$$

$$\Psi_2' = -\frac{a_2}{\bar{s}_L}\frac{s\cos b - \omega \sin b}{s^2+\omega^2} \left[-\frac{\bar{s}_L}{s} x - \left(\frac{\bar{s}_L}{s}\right)^2 + C e^{sx/\bar{s}_L} \right] . \tag{A.36}$$

The initial condition $\Psi_2'(x=R,y,s)=0$ results in:

$$\Psi_2' = -\frac{a_2}{\bar{s}_L}\frac{s\cos b - \omega \sin b}{s^2+\omega^2} \left[-\frac{\bar{s}_L}{s} x - \left(\frac{\bar{s}_L}{s}\right)^2 + \left(\frac{\bar{s}_L}{s} R + \left(\frac{\bar{s}_L}{s}\right)^2 \right) e^{-s(R-x)/\bar{s}_L} \right] \tag{A.37}$$

$$\begin{aligned} \Psi_2' = a_2 \frac{s\cos b - \omega \sin b}{s^2+\omega^2} \Bigg[& \frac{R}{s}\left(\frac{x}{R} - e^{-s(R-x)/\bar{s}_L} \right) \\ & + \frac{\bar{s}_L}{s^2}\left(1 - e^{-s(R-x)/\bar{s}_L} \right) \Bigg] , \end{aligned} \tag{A.38}$$

which is of the form:

$$\Psi_2' = F_2(s)(x/R - e^{-cs}) + F_3(s)(1-e^{-cs}) \,. \tag{A.39}$$

This implies a solution of the form:

$$\begin{aligned} \psi_2'(x,y,t) = & \frac{x}{R} f_2(x,y,t) - f_2(x,y,t-c)H(t-c) \\ & + f_3(x,y,t) - f_3(x,y,t-c)H(t-c) \,, \end{aligned} \tag{A.40}$$

and f_2 and f_3 are found to be (see appendix A.2.2 and A.2.3):

$$f_2(t) = \frac{a_2 R}{\omega}(\sin(\omega t + b) - \sin b) \; ; \tag{A.41}$$

$$f_3(t) = \frac{a_2 \bar{s}_L}{\omega^2}(\cos b - \omega t \sin b - \cos(\omega t + b)) \; . \tag{A.42}$$

The overall solution is then:

$$\begin{aligned}
\psi' =& \frac{a_1}{\omega}(\cos b - \cos(\omega t + b)) \\
&+ \frac{a_2 x}{\omega}(\sin(\omega t + b) - \sin b) \\
&+ \frac{a_2 \bar{s}_L}{\omega^2}(\cos b - \omega t \sin b - \cos(\omega t + b)) \\
&- \frac{a_1}{\omega}(\cos b - \cos(\omega(t-c) + b))H(t-c) \\
&- \frac{a_2 R}{\omega}(\sin(\omega(t-c) + b) - \sin b)H(t-c) \\
&- \frac{a_2 \bar{s}_L}{\omega^2}(\cos b - \omega(t-c)\sin b - \cos(\omega(t-c) + b))H(t-c) \; .
\end{aligned} \tag{A.43}$$

To see why this solution is transient, we note that at any point on the flame surface where $t < \frac{R-x}{\bar{s}_L}$:

$$\begin{aligned}
\psi' =& \frac{a_1}{\omega}(\cos b - \cos(\omega t + b)) \\
&+ \frac{a_2 x}{\omega}(\sin(\omega t + b) - \sin b) \\
&+ \frac{a_2 \bar{s}_L}{\omega^2}(\cos b - \omega t \sin b - \cos(\omega t + b)) \; ,
\end{aligned} \tag{A.44}$$

whereas at any point on the surface where $t > \frac{R-x}{\bar{s}_L}$, the solution is:

$$\begin{aligned}
\psi' =& \frac{a_1}{\omega}(\cancel{\cos b} - \cos(\omega t + b)) \\
&+ \frac{a_2 x}{\omega}(\sin(\omega t + b) - \sin b) \\
&+ \frac{a_2 \bar{s}_L}{\omega^2}(\cancel{\cos b} - \cancel{\omega t \sin b} - \cos(\omega t + b)) \\
&- \frac{a_1}{\omega}(\cancel{\cos b} - \cos(\omega(t-c) + b)) \\
&- \frac{a_2 R}{\omega}(\sin(\omega(t-c) + b) - \sin b) \\
&- \frac{a_2 \bar{s}_L}{\omega^2}(\cancel{\cos b} - \omega(t-c)\sin b - \cos(\omega(t-c) + b)) \; ;
\end{aligned} \tag{A.45}$$

$$\begin{aligned}\psi' =& \frac{a_1}{\omega}(\cos(\omega(t-c)+b)) - \cos(\omega t+b)) \\ &+ \frac{a_2}{\omega}(x\sin(\omega t+b) - R\sin(\omega(t-c)+b) + (R-x)\sin b) \\ &+ \frac{a_2\bar{s}_L}{\omega^2}(\cos(\omega(t-c)+b) - \omega c\sin b - \cos(\omega t+b)) \ ,\end{aligned} \tag{A.46}$$

which is different to the initial oscillation in (A.44). Equation (A.46) applies everywhere on the flame surface when the system reaches its forced cycle after a time t_c:

$$t_c = \frac{R}{\bar{s}_L} \tag{A.47}$$

Using the definition of $\bar{s}_L$ (A.13) and the steady result (A.3), t_c is:

$$\begin{aligned} t_c &= \frac{R}{\bar{s}_L} \\ &= \frac{R\sqrt{1+\beta^2}}{s_L\beta} \\ &= \frac{R(1+\beta^2)}{\beta V} \ . \end{aligned} \tag{A.48}$$

This agrees with our intuition for flames with large β (tall, slender flames): $t_c \approx \beta R/V$ is the time it takes for a perturbation travelling with speed V to travel from the base of the flame to its tip, a distance βR downstream. For a forced cycle with period $T = 1/f$, the number of periods it takes for the system to reach the forced cycle, p_c, is:

$$\begin{aligned} p_c &= \frac{t_c}{T} \\ &= \frac{fR(1+\beta^2)}{\beta V} \ . \end{aligned} \tag{A.49}$$

A.2 Inverse Laplace transforms

A.2.1 $f_1(t)$, the inverse Laplace transform of $F_1(s)$

$$\begin{aligned} F_1(s) &= a_1 \frac{s\sin b + \omega\cos b}{s(s^2+\omega^2)} \\ &= a_1 \frac{\sin b}{s^2+\omega^2} + a_1 \frac{\omega\cos b}{s(s^2+\omega^2)} \\ &= \frac{a_1\sin b}{\omega}\frac{\omega}{s^2+\omega^2} + \frac{a_1\cos b}{\omega}\left(\frac{1}{s} - \frac{s}{s^2+\omega^2}\right) \end{aligned} \tag{A.50}$$

Thus:

$$\begin{aligned} f_1(t) &= \frac{a_1\sin b}{\omega}\sin\omega t + \frac{a_1\cos b}{\omega}(1-\cos\omega t) \\ &= \frac{a_1}{\omega}(\cos b - \cos(\omega t + b)) \end{aligned} \tag{A.51}$$

A.2.2 $f_2(t)$, the inverse Laplace transform of $F_2(s)$

$$\begin{aligned} F_2(s) &= a_2 R \frac{s\cos b - \omega\sin b}{s(s^2+\omega^2)} \\ &= a_2 R \frac{\cos b}{s^2+\omega^2} - a_2 R \frac{\omega\sin b}{s(s^2+\omega^2)} \\ &= \frac{a_2 R\cos b}{\omega}\frac{\omega}{s^2+\omega^2} - \frac{a_2 R\sin b}{\omega}\left(\frac{1}{s} - \frac{s}{s^2+\omega^2}\right) \end{aligned} \tag{A.52}$$

Thus:

$$\begin{aligned} f_2(t) &= \frac{a_2 R\cos b}{\omega}\sin\omega t - \frac{a_2 R\sin b}{\omega}(1-\cos\omega t) \\ &= \frac{a_2 R}{\omega}(\sin(\omega t + b) - \sin b) \end{aligned} \tag{A.53}$$

A.2.3 $f_3(t)$, the inverse Laplace transform of $F_3(s)$

$$\begin{aligned} F_3(s) &= a_2\bar{s}_L \frac{s\cos b - \omega\sin b}{s^2(s^2+\omega^2)} \\ &= a_2\bar{s}_L \frac{\cos b}{s(s^2+\omega^2)} - a_2\bar{s}_L \frac{\omega\sin b}{s^2(s^2+\omega^2)} \\ &= \frac{a_2\bar{s}_L\cos b}{\omega^2}\left(\frac{1}{s} - \frac{s}{s^2+\omega^2}\right) - \frac{a_2\bar{s}_L\sin b}{\omega}\left(\frac{1}{s^2} - \frac{1}{s^2+\omega^2}\right) \end{aligned} \tag{A.54}$$

Thus:

$$\begin{aligned} f_3(t) &= \frac{a_2\bar{s}_L\cos b}{\omega^2}(1-\cos\omega t) - \frac{a_2\bar{s}_L\sin b}{\omega}(t-\frac{1}{\omega}\sin\omega t) \\ &= \frac{a_2\bar{s}_L}{\omega^2}(\cos b - \omega t\sin b - \cos(\omega t+b)) \end{aligned} \tag{A.55}$$

A.3 Linear perturbation analysis of the *G*-equation model of the Volvo combustor

By considering a vertically-aligned Volvo combustor, the linear perturbation analysis follows that of the Bunsen flame up until the prescribing of the velocity perturbations. Due to the addition of the parameter η, the linearised PDE now takes the following form (c.f. equation A.17):

$$\frac{\partial \psi'}{\partial t} - \bar{s}_L \frac{\partial \psi'}{\partial x} = a_1 \sin(\omega t + b) + a_2 x \cos(\omega t + b) + a_3 x \sin(\omega t + b) \ , \tag{A.56}$$

for appropriate terms a_1, a_2 and a_3. As this PDE is linear, the solution can be separated, as was done in the Bunsen flame perturbation analysis. We note from the Bunsen flame perturbation analysis that the transient dissipation time t_c is only a function of the laminar flame speed $\bar{s}_L$ and the Bunsen flame radius, R. The transient dissipation time is not a function of the sinusoids on the right and side of equation A.56. For the Volvo combustor, the equivalent length is the bluff-body side length, D. Therefore, the transient dissipation time t_c for transients in the Volvo combustor model is:

$$\begin{aligned} t_c &= \frac{D}{\bar{s}_L} \\ &= \frac{D\sqrt{1+\beta^2}}{s_L \beta} \\ &= \frac{D(1+\beta^2)}{\beta U} \ . \end{aligned} \tag{A.57}$$

For a forced cycle with period $T = 1/f$, the number of periods it takes for the system to reach the forced cycle, p_c, is:

$$\begin{aligned} p_c &= \frac{t_c}{T} \\ &= \frac{fD(1+\beta^2)}{\beta U} \ . \end{aligned} \tag{A.58}$$

Appendix B

Calculating $\bar{\dot{Q}}_h$, the mean heat release rate of the Volvo burner

B.1 Derivation of $\dot{q}_h(\mathbf{x},t)/\bar{\dot{Q}}_h$, the normalised distribution of heat release rate field

B.1.1 Nomenclature

$\mathbf{x}$: coordinate (x,y) lying on the flame front in the domain Ω.
t : time [s]
$\dot{q}_h(\mathbf{x},t)$: the heat release rate at coordinate $\mathbf{x}$ and time t [J/s]
$\bar{\dot{Q}}_h$: the mean heat release rate of the Volvo burner (or the total power output) [J/s]
T : period [s]
s_L : laminar flame speed [m/s]
$a(\mathbf{x},t)$: surface area of the flame at coordinate $\mathbf{x}$ and time t [m^2]
dt : infinitesimal change in time [s]
$V(\mathbf{x})$: volume of premixed gas per unit depth consumed in a period (time T) [m^3]
$E(\mathbf{x})$: energy output per unit volume of premixed gas at coordinate $\mathbf{x}$ [J/m^3]

B.1.2 Derivation

Energy output over a period $= \bar{\dot{Q}}_h T$.
Volume of premixed gases consumed in time dt at coordinate $\mathbf{x}$: $dV = s_L a(\mathbf{x},t)dt$
Volume of premixed gases consumed over a period at coordinate $\mathbf{x}$: $V(\mathbf{x}) = s_L \int_0^T a(\mathbf{x},t)dt$
Energy per unit volume of unburnt gas at coordinate $\mathbf{x}$: $E(\mathbf{x}) = \bar{\dot{Q}}_h T / V(\mathbf{x})$
Energy released in time dt at coordinate $\mathbf{x}$: $\dot{q}_h(\mathbf{x},t) = s_L a(\mathbf{x},t) dt E(\mathbf{x})/T$

$$\frac{\text{Energy released in time } dt}{\bar{\dot{Q}}_h} = \frac{\dot{q}_h(\mathbf{x},t)}{\bar{\dot{Q}}_h} = \frac{s_L a(\mathbf{x},t) dt E(\mathbf{x})}{\bar{\dot{Q}}_h T} \tag{B.1}$$

$$\frac{\dot{q}_h(\mathbf{x},t)}{\bar{\dot{Q}}_h} = \frac{s_L a(\mathbf{x},t) dt}{V(\mathbf{x})} \tag{B.2}$$

$$\frac{\dot{q}_h(\mathbf{x},t)}{\bar{\dot{Q}}_h} = \frac{s_L a(\mathbf{x},t) dt}{\int_0^T s_L a(\mathbf{x},t) dt} \tag{B.3}$$

$$\frac{\dot{q}_h(\mathbf{x},t)}{\bar{\dot{Q}}_h} = \frac{a(\mathbf{x},t) dt}{\int_0^T a(\mathbf{x},t) dt} \tag{B.4}$$

Therefore, the normalised heat release rate at any location on the flame surface $\mathbf{x}$ and for any time t is found by calculating the normalised flame surface area at the same location and time.

B.2 Calculation of $\bar{\dot{Q}}_h$

The following data about the Volvo burner are from [35]:

- Air flow rate $\dot{m}_{air}$ of 0.35 kg/s;
- Fuel-air equivalence ratio ϕ of 1.0;
- Fuel: propane (C_3H_8), with heat of combustion 50430 kJ/kg[1]).

Air by mass is 23.1% oxygen so $m_{O2} = 0.231 m_{air}$. For our combustion data, the fuel-air equivalence ratio ϕ is 1.0, which is defined as:

$$\phi = \frac{\text{fuel-to-oxidiser ratio}}{(\text{fuel-to-oxidiser ratio})_{stoich}} = \frac{m_{fuel}/m_{ox}}{(m_{fuel}/m_{ox})_{stoich}} = \frac{n_{fuel}/n_{ox}}{(n_{fuel}/n_{ox})_{stoich}} \tag{B.5}$$

where $m_{\cdot}$ and $n_{\cdot}$ are the molecular mass and the moles in the reaction equation:

$$C_3H_8 + 5O_2 = 3CO_2 + 4H_20. \tag{B.6}$$

The stoichiometric molar fuel to oxidiser ratio, $FOR_{stoich} = 0.2$ (as combustion of propane requires 5 O_2 molecules for every C_3H_8 molecule). The (non-stoichiometric) molar fuel to

[1] https://www.engineeringtoolbox.com/propane-d_1423.html

oxidiser ratio FOR is therefore $FOR = \phi FOR_{stoich} = 1.0 \times 0.2 = 0.2$. Now, as

$$\frac{m_{fuel}}{m_{ox}} = \frac{n_{fuel}}{n_{ox}} \times \frac{\text{mass of } C3H8}{\text{mass of } O2} = FOR \times \frac{\text{mass of } C3H8}{\text{mass of } O2} \tag{B.7}$$

then $m_{fuel}/m_{ox} = 0.2 \times 44.1/32 = 0.276$. Finally:

$$m_{fuel} = 0.276 m_{ox} = 0.276 \times 0.231 \times m_{air}. \tag{B.8}$$

This means that, for our combustion data which has an air mass flow rate of 0.35 kg/s the mass flow rate of fuel is $0.276 \times 0.231 \times 0.35 = 0.0223$ kg/s. For a energy density of 50430 kJ/kg, this suggests a power output $\bar{\dot{Q}}_h = 50430 \times 0.0223 = 1125$ kW $= 1.125$ MW.

Appendix C

The discrete vortex method

This appendix is due to Mr Joel Vasanth.

C.1 Modelling

We begin with the incompressible 2D Euler equations in vorticity form:

$$\frac{\partial \boldsymbol{\omega}}{\partial t} + \mathbf{u} \cdot \nabla \boldsymbol{\omega} = \boldsymbol{\omega} \cdot \nabla \mathbf{u}, \quad \nabla \cdot \mathbf{u} = 0, \tag{C.1}$$

where the term on the right hand side is the stretching of vorticity due to velocity gradients and the terms on the left form the material derivative $\mathrm{D}\boldsymbol{\omega}/\mathrm{D}t$, or the Lagrangian transport term. The vorticity in the domain is discretised into a number of vortex elements at locations $\mathbf{x}_i$, each carrying an elementary amount of vorticity within a core of radius $r_{D,0}$. The distribution of vorticity within the vortex element is given by a core function f_δ, which ensures a finite velocity at the core centre. The discrete vorticity field $\boldsymbol{\omega}^h$ is then

$$\boldsymbol{\omega}^h = \sum_{i=1}^{N_V} \Gamma_i f_\delta(r), \tag{C.2}$$

where Γ_i are the vortex circulations and the summation is over the total number of vortices in the domain. The advection of the vortices are given by the solution of the N_V ODEs

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}(\mathbf{x}_i, t). \tag{C.3}$$

The local flow velocities $\mathbf{u}(\mathbf{x}_i, t)$ are obtained from the solution of the Poisson equation $\nabla^2 \mathbf{u} = -\nabla \times \boldsymbol{\omega}$:

$$\mathbf{u} = \nabla G(\mathbf{x}, \mathbf{x}') \star \boldsymbol{\omega}^h + \mathbf{u}_p, \tag{C.4}$$

where $G(\mathbf{x},\mathbf{x}')$ is the Green's function for the Laplacian operator in 2D, i.e., $\nabla^2 G(\mathbf{x},\mathbf{x}') = \delta(\mathbf{x}-\mathbf{x}')$; and $\star$ is the convolution operator. In the absence of any boundaries, $G(\mathbf{x}-\mathbf{x}') = \ln(|\mathbf{x}-\mathbf{x}'|)/2\pi$. The first term is the rotational component of the flow and the velocity $\mathbf{u}_p$ is the irrotational part of the flow field, which is uniquely chosen such that the wall-normal boundary condition $\mathbf{u}\cdot\hat{\mathbf{n}} = 0$ is satisfied.

The solution of the potential flow problem can be found via a Schwartz-Christoffel transformation (SCT). Here, the physical domain is modelled in a complex plane $z = x + yj$ (Fig. C.1(a)). Symmetry about the x-axis is assumed, since the experimental oscillating flame shapes are similarly symmetric. The z-plane is mapped onto the upper-half of a transformed s-plane (Fig. C.1(c)). If $s_{V,\,i} \quad i = 1,\ldots,N_V$ represents the vortex locations, an image system of vortices at their conjugates $s^*_{V,\,i}$ is then defined in the lower-half of the s-plane, such that the resulting wall-normal velocity cancels to 0. The mapping is performed in two steps via an intermediate mapping to an infinite-strip in the ξ-plane (Fig. C.1(b)), which simplifies the mathematical expression for the SCT ([108]). The mapping from z to ξ is obtained by integrating the expression

$$\frac{dz}{d\xi} = 1.5\left[\sinh\frac{\pi}{2}\xi\right]^{-1/4}\left[\sinh\frac{\pi}{2}(\xi-\xi_E)\right]^{3/4}\left[\sinh\frac{\pi}{4}(\xi-\xi_F)\right]^{-1/2}, \tag{C.5}$$

and the mapping from ξ to s is simply $s = e^{\pi\xi}$. The values ξ_E, ξ_F and ξ_B in Fig. C.1(b) are obtained numerically using the SCT toolbox developed by [108]. These are 0.95, 0.63 and $0.58 + j$ respectively.

The potential flow velocity field is then given by a spatially-uniform inlet flow $U(t)$ in the ξ-plane and induced velocities from vortex elements and their images in s-plane. Both are transformed to the physical z-plane using the chain rule.The chain rule is applied to transform these velocities into the physical domain. The inlet flow is given as

$$\sigma_{inlet} = U(t)\frac{d\xi}{dz}, \tag{C.6}$$

where σ represents complex velocities. The flow due to the vortices and their images as ([110])

$$\sigma_{vort} = \sum_{i=1}^{N_V} j\frac{\Gamma_i}{2\pi}\left[-\frac{1}{s-s_{V,\,i}} + \frac{1}{s-s^*_{V,\,i}}\right]\frac{ds}{d\xi}\frac{d\xi}{dz}. \tag{C.7}$$

In Eq. C.7, the sum is over all the vortex elements each with circulation Γ_i and location $s_{V,\,i}$. The x and y components of the physical velocity field $\mathbf{u}$ are then $\mathrm{Re}(\sigma_{inlet} + \sigma_{vort})$ and

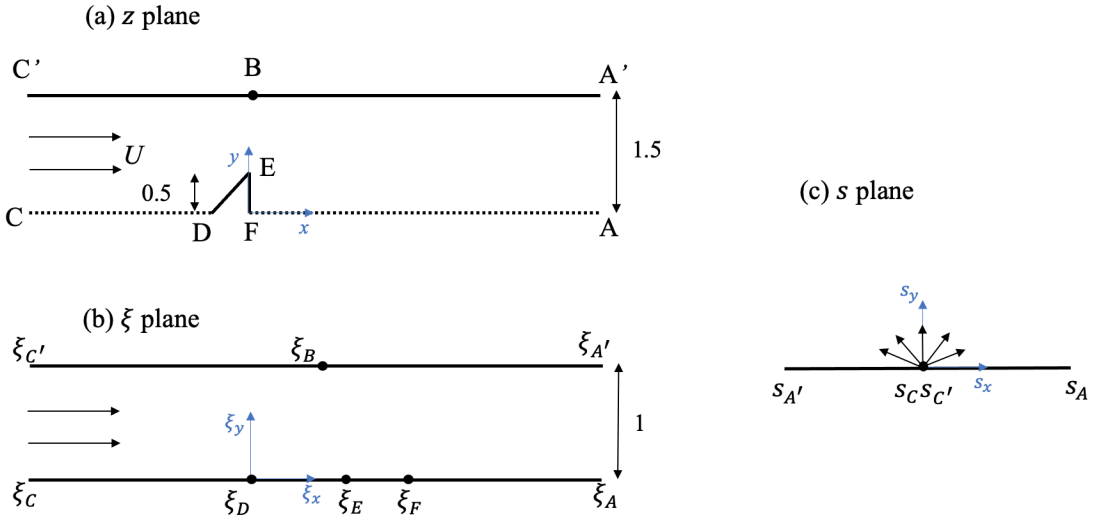


Fig. C.1 Illustration of the Schwartz-Christoffel mappings from (a) the physical z plane to (b) the transformed ξ plane, which is an infinite strip, and then to (c) the upper half of the s plane. The dotted line in (a) is a line of symmetry. The subscripts in the labels in the ξ and s planes correspond to the respective vertices in the z-plane. Marked dimensions in the z plane in (a) are scaled by the size D of the bluff body.

$-\mathrm{Im}(\sigma_{inlet}+\sigma_{vort})$ respectively. The vortices advect with the flow as

$$\frac{d\mathbf{x}_i}{dt} = \mathbf{u}(\mathbf{x}_i,t) \quad i = 1,\ldots,N_V \tag{C.8}$$

A generalised Kutta condition is used to model the vortex generation process. According to this, the vorticity is discretised by introducing a single vortex element into the domain at each time step Δt with a circulation $\Gamma_0 = -u_0(t)^2\Delta t/2$. The velocity u_0 is computed at a location $z_0 = z_{0,\mathrm{r}} + z_{0,\mathrm{i}}j$ directly above the bluff body edge E as in Fig. C.1(a) ($z_{0,\mathrm{r}} = 0$). The vortex elements are then advected as per Eq. C.8.

The above gives a point vortex description of the potential flow field. The singularity at the center of the vortex elements is removed using a refined Lamb-Oseen model. According to this model, a vortex 'core' of size r_D is defined, along with an azimuthal velocity u_θ that varies smoothly with distance r from the center as

$$u_\theta = \frac{\Gamma_0}{2\pi r}(1-e^{-r^2/r_D^2}) \tag{C.9}$$

In the model in Eq. C.9 the velocity field tends to that of a rotational vortex as $r \to 0$, and it tends to that of the potential flow solution as $r \to \infty$. We implement a linear temporal growth

of the core radius as $r_D(t) = r_{D,0} + 2\sqrt{\nu t}$, where ν is the kinematic viscosity and $r_{D,0}$ is the initial core radius of the vortex elements at the instant of being introduced into the domain. The linear temporal growth model thus accounts for the viscous diffusion of vorticity ([111]).

To incorporate the effect of the boundary layer on the vorticity, we model the decay of vorticity at the walls as $d\Gamma/dt = -\Gamma$, when the distance between the vortex center of any vortex element and the wall is less than r_D. All lengths in the DVM are non-dimensionalised with the bluff body side length D, velocities with the mean inlet U and time with U/D.

The acoustic forcing on the flow field is modelled with a harmonically oscillating inlet flow velocity $u(t)/U = 1 + \alpha \sin(\mathrm{St}\, t)$ in Eq. C.6, where $\mathrm{St} = 2\pi f \beta D/U$ is the forcing Strouhal number expressed in terms of the forcing frequency f and the nominal aspect ratio β. This adds a time-dependent potential to the existing potential flow, while still fulfilling the boundary conditions. The oscillatory flow $u_0(t)$ at z_0 produces vortices of varying Γ_0 and so the strengths of the vortex elements are also affected by the forcing. The effect of the forcing is thereby incorporated into the vortex shedding process. In order to ensure that N_V does not increase without bound, vortices are deleted when they cross a streamwise location $x_{\max}/D = 25$.

The time integration in Eq. C.8 is solved using a 4th-order Runge-Kutta method, and the numerical integration of the SCT in Eq. C.5 is performed using Simpson's rule. The accuracy of the DVM in predicting the velocity field depends especially on its discretisation parameters - the core size, the core function and the number of vortex elements shed per time step ([112]). In many studies ([111, 113, 114]), these parameters are hand-tuned and pre-set before the model is used for predictions. In order to render the flame-vortex model more quantitatively accurate in its predictions, the parameters $r_{D,0}$, $z_{0,\mathrm{i}}$, St, a and β are inferred from a machine learning model in this work.