Automatic simplification of differential equation models by *a posteriori* analysis



Philip J. Maybank

Linacre College

Department of Computer Science, University of Oxford

A thesis submitted for the degree of

M.Sc. by Research

Trinity 2012

Abstract

Many mathematical models in biology and physiology are represented by systems of nonlinear differential equations. In recent years these models have become increasingly large-scale and multiphysics, as increasing amounts of data are available on the properties and behaviour of biological systems. Often an observed behaviour of interest in a model may be written as a linear functional. A key question therefore is to determine which terms in the model have the greatest effect on functionals of interest.

An approach for answering this question has recently been developed, called model reduction using *a posteriori* analysis. The method was initially developed for systems of nonlinear initial value ordinary differential equations, and automatically identifies regions of the computational domain and components of the model solution where an accurate mathematical representation of the model is required to accurately calculate a linear functional of interest. Initial-value ordinary differential equations can be written as a first-order derivative term plus an algebraic 'reaction' term. In previous work on model reduction using *a posteriori* analysis the algebraic 'reaction' term is removed from the equations in the reduced model.

The first contribution of this thesis is to extend the method so that the first-order derivative term is removed from the differential equation instead of the algebraic 'reaction' term, resulting in a quasi-steady state approximation in automatically identified regions of the domain and components of the solution. The second contribution of this thesis is to extend the method to boundary value problems and partial differential equations. We consider differential equations with algebraic terms, first order terms and second order terms, any combination of which may be nonlinear. The method is used to automatically simplify several differential equation models including models of chemotaxis and tissue-level cardiac electrophysiology.

Acknowledgements

I would like to thank my supervisors Dr Jonathan Whiteley and Professor David Gavaghan for their work, for being both supportive and challenging. Thanks also to the other members of the Computational Biology group for providing such a stimulating and friendly environment to work in, particularly Margarita Satraki for always being in the office and willing to talk about anything.

I would also like to acknowledge the EPSRC for funding this project through the System Biology Doctoral Training Centre. Thanks also to the staff of the DTC for their direct and indirect contributions to my formation as a research student.

Finally I wish to thank all my friends in Oxford. In particular Sofia Piltz, Azim Ansari, and Valentina Carapella for providing good humour and encouragement, and Robert Heimburger for being a patient and joyful flatmate.

Contents

1 Introduction **1**

2 Model analysis in systems biology **4**

2.1 Mathematical modelling in systems biology 4

2.2 Existing methods for model analysis and reduction 6

2.2.1 Sensitivity analysis 6

2.2.2 Perturbation analysis and other methods for model reduction . . 9

2.3 Previous work on the *a posteriori* model reduction method 11

3 Initial-Value Ordinary Differential Equations **13**

3.1 Initial-value approximation 13

3.1.1 Derivation of error bound 13

3.1.2 Application to model problems 17

3.2 Quasi-steady state approximation 20

3.2.1 Derivation of error bound 23

3.2.2 Application to model problems 27

3.3 Robustness of *a posteriori* model reduction to the N_{cont} parameter . . . 28

3.4 Summary 31

4 Boundary-Value Differential Equations **37**

4.1 The finite element method for reaction-diffusion equations 38

4.2 Linear reaction-diffusion equation 40

4.2.1 Derivation of error bound for diffusion approximation 40

4.2.2 Application to a model problem 43

4.2.3 Derivation of error bound for reaction approximation 44

4.2.4 Application to model problem 49

4.3 Nonlinear reaction-diffusion equations 51
4.3.1 Derivation of error bound 51
4.3.2 Application to a single-component model problem 54
4.3.3 Application to a multi-component model of chemotaxis 57
4.4 Robustness of *a posteriori* model reduction to the N_{cont} parameter . . . 59
4.5 Computational Efficiency 62
4.6 Summary 65

5 Partial Differential Equations **69**
5.1 Numerical schemes for time-dependent problems 70
5.2 Fisher's equation 71
5.2.1 Derivation of error bound 71
5.2.2 Application to Fisher's equation 73
5.3 Nonlinear mixed-type PDEs 75
5.3.1 Derivation of error bound 75
5.3.2 Application to monodomain equation 81
5.3.3 Application to a linear advection-diffusion equation 84
5.3.4 Application to a nonlinear advection-diffusion equation 89
5.4 Remarks on applying *a posteriori* model reduction to quasi-linear PDEs 93
5.5 Robustness of *a posteriori* model reduction to algorithm parameters . . 96
5.6 Computational efficiency 101
5.7 Summary 102

6 Discussion **105**

References **109**

Nomenclature

Linear algebra nomenclature

$\boldsymbol{u}$	vector with N components
u_i	ith entry of $\boldsymbol{u}$
A	matrix with $N \times N$ entries
A_{ij}	entry of A in ith row and jth column
$\boldsymbol{a}_i$	ith row of a matrix

Calculus nomenclature

Ω	computational domain, for example, interval $[a, b]$ or $[0, T]$
$L^2(\Omega)$	Hilbert space of square integrable functions, with norm $\|\|f\|\| = \left(\int_\Omega f^2 \, \mathrm{d}x\right)^{1/2}$
$\langle f, g \rangle$	inner product between functions $f(x)$ and $g(x)$, $\int_\Omega f(x)g(x) \, \mathrm{d}x$
$\langle \boldsymbol{f}, \boldsymbol{g} \rangle$	inner product between vector-valued functions $\boldsymbol{f}(x)$ and $\boldsymbol{g}(x)$, $\int_\Omega \boldsymbol{f}(x) \cdot \boldsymbol{g}(x) \, \mathrm{d}x$
$[f(x)]_{x_k^-}^{x_k^+}$	$f(x_k^+) - f(x_k^-)$, where $f(x_k^+)$ is the limit of $f(x)$ as $x \to x_k$ from above, and $f(x_k^-)$ is the limit of $f(x)$ as $x \to x_k$ from below
$J(\cdot)$	functional – a map from $L^2(\Omega)$ to $\mathbb{R}$
L	differential operator – a map from the appropriate subspace of $L^2(\Omega)$ to $L^2(\Omega)$
L_i	differential operator that produces a subset of the terms that the differential operator L produces when applied to a function
u'	derivative of u for u a function of one independent variable
$\delta(x - x_0)$	dirac delta function, $\int_\Omega f(x)\delta(x - x_0) \, \mathrm{d}x = f(x_0)$ for all well-defined functions f.

Nomenclature specific to method

S_i	subregion of Ω that may be made up of several disjoint regions
$\boldsymbol{S}$	vector with entries S_i for i from 1 to N
$\chi_i(x)$	indicator function that is 1 if $x \in S_i$ and 0 otherwise
N_{cont}	number of elements added to $\boldsymbol{S}$ on each iteration of the algorithm
TOL	desired accuracy in $J(\cdot)$, the quantity of interest

1

Introduction

Complex differential equation models are becoming increasingly common in systems biology [10, 31, 36, 54]. Such models may contain tens or hundreds of variables, several different types of differential operator [27, 43, 50, 55], and describe processes over a wide range of spatial and temporal scales [40]. Complex systems biology models are frequently opaque in the sense that it is difficult to identify the components of the model which are responsible for observed behaviours of interest. While the development of detailed biophysical models is gathering pace, there is a lack of tools to automatically analyze these new models. A big challenge in systems biology is understanding the dynamics of these models. This means identifying the components of a model that control healthy and diseased state respectively, and making testable predictions relating to how systems dynamics can be altered with drugs or other kinds of intervention. Greater understanding of the mechanisms responsible for behaviours of interest is needed, and mathematical tools are helpful towards this end.

Model reduction using *a posteriori* analysis is a method for identifying key components that affect a user-defined quantity of interest in a differential equation model and identifying the time-space regions where accurate representation of those components is important [75]. The method borrows ideas from *a posteriori* error analysis in numerical analysis. Whereas error analysis for finite element methods is used to construct a suitably accurate numerical discretisation of the differential operator, model reduction using *a posteriori* analysis is used to construct a sufficiently accurate reduced differential operator. In previous work on *a posteriori* model reduction the method has been applied to coupled nonlinear IVPs, and the reduced model is the original IVP with the

algebraic terms removed in regions of the domain where those terms have little effect on the user-defined quantity of interest in the model [75].

A key benefit of using *a posteriori* model reduction over other methods is that it is automated. This makes the analysis of large-scale models practical. Many mathematical methods such as perturbation methods require significant mathematical expertise, and are difficult to apply to systems with more than a few variables. Furthermore *a posteriori* model reduction automatically gives information about all model components over all regions of the domain, therefore providing more information and insight than many perturbation methods.

In the order that they appear, the main contributions of this thesis are as follows:

- Development of a novel variant of the method that removes the first-order derivative terms in automatically identified regions of the domain for coupled nonlinear IVPs.

- Extension of the methodology to coupled nonlinear Boundary Value Problems (BVPs). We develop the method for a linear steady-state reaction-diffusion equation and two forms of the method are introduced: one that removes the reaction term and another that removes the diffusion term, in automatically identified regions of the domain. We also give a general error bound derivation for nonlinear coupled BVPs, where the nonlinearity can be either in the algebraic terms or in the second-order terms.

- Extension of the methodology to coupled nonlinear reaction-diffusion and advection-diffusion Partial Differential Equations (PDEs).

Each novel development of the method is applied to one or more model problems. For example, the novel variant of the method for IVPs is applied to the Michaelis-Menten equations to identify regions of the domain where a quasi-steady state approximation applies; the method for BVPs is applied to several reaction-diffusion equations including a chemotaxis model with a nonlinear second-order x-derivative term; and the method for PDEs is applied Fisher's equation, the monodomain equation, and two advection-diffusion equations.

The development of the method in this thesis aims to establish two additional benefits of the method. One is that it can be applied to a wide range of different types

of differential equation model. The other is that the method can be used to assess the importance of any of the terms in the model equations, provided that the model is still well-posed after these terms have been removed. Given these aims, we focus on differential equation models that are relatively simple and well understood, in order to demonstrate the effectiveness of the method. It should be remembered throughout that the methodology is general and automated, and can be applied to models with more terms and or components than are present in the models that we analyze.

In Chapter 2 we review the most commonly used methods for model analysis in systems biology, and discuss in more detail questions that may be answered using *a posteriori* model reduction that cannot be answered using other methods.

In Chapter 3 we first re-state the main results from the original paper on *a posteriori* model reduction for IVPs [75], and apply the method to two model problems. We then develop a novel variant of the method for IVPs that automatically identifies regions of the domain where a quasi-steady state approximation is appropriate. This novel variant of the method is then applied to the same two model problems as were analyzed with the original method. Also, we evaluate the effect of varying one of the algorithm parameters, allowing the method to be made more computationally efficient for the problems analyzed.

In Chapter 4 we extend the methodology to BVPs. We first consider a linear BVP and show how the method can be formulated so that either the reaction term or diffusion term is removed in automatically identified regions of the domain. We then consider nonlinear BVPs, where the nonlinearity may be in either the algebraic term, the second order term, or in both terms. The error bound derivation in this section is the most mathematically technical part of the thesis. The error bound is then applied to two model problems that are nonlinear in the algebraic and second-order terms. As in Chapter 3, we evaluate the effect of varying one of the algorithm parameters.

In Chapter 5 we extend the methodology to PDEs with a linear first-order time derivative term and algebraic, first order and second order x-derivative terms, any combination of which may be nonlinear. The method is applied to four different model problems, each of which display a different kind of travelling wave behaviour. The method is shown to give insight into these equations and the results are consistent with what is known mathematically and physiologically. As in Chapters 3 and 4, we evaluate the effect of varying one of the algorithm parameters.

2

Model analysis in systems biology

2.1 Mathematical modelling in systems biology

Biological systems contain many different components and are governed by multiple physical mechanisms. Consequently, mathematical models of the these systems are increasingly large-scale and multiphysics. A key challenge in systems biology is to understand how changes in the dynamics of the model components affect behaviours of interest in the model.

One of the first great successes of mathematical modelling in biology was the Hodgkin-Huxley model of electrophysiology in the squid giant axon [30]. The Hodgkin-Huxley model contained three ionic variables, and a variable for voltage, and accurately reproduced the action potential dynamics of the squid giant axon. Following the publication of the Hodgkin-Huxley model, an early pioneer in mathematical biology, Richard Fitzhugh, worked extensively on analysis of the Hodgkin-Huxley equations [20, 21]. He also introduced a simplification of the Hodgkin-Huxley equations, known as the FitzHugh-Nagumo model [21, 52]. Fitzhugh's work illustrates the close relationship between model analysis and model development. One of Fitzhugh's insights was that in the Hodgkin-Huxley model, two of the four variables act on a fast timescale, and the other two variables act on a slow timescale. The fast variables are responsible for activation and the slow variables are responsible for recovery of the transmembrane potential. This led him to develop what is now known as the Fitzhugh-Nagumo model, which contains a single fast activation variable and a single slow recovery variable. The FitzHugh Nagumo model qualitatively reproduces many features of the Hodgkin-

Huxley model, and has subsequently been used extensively as a model problem for excitable media in cardiac electrophysiology [11] and computational neuroscience [37].

The Hodgkin-Huxley model coupled electrical activity with ion channel kinetics, and was a prototype for many subsequent mathematical models in biology. Generally speaking, mathematical models have become increasingly complex both in the number of components included in the model and in the physical mechanisms included in the model. Ordinary Differential Equation (ODE) models with tens and even hundreds of model components have been developed in cardiac electrophysiology [36, 54], cell cycle dynamics [10], circadian clocks [22, 45], metabolism [33, 68, 72], and cell signalling [32, 53]. Partial Differential Equation (PDE) models have been developed that couple multiple physical effects such as diffusion, advection, viscosity, elasticity, and or other stresses. Some examples include the bidomain equation which couples electrical conduction with ion channel kinetics and biochemical reactions [40]; chemotaxis models that couple transport of cell populations with diffusion of chemical signal, and kinetics of the chemical signal [27, 50]; and crawling cell models that describe the cell as a viscous fluid coupled with contractile and swelling stresses, and chemical kinetics [43, 55].

Many models in systems biology attempt to reproduce the dynamics of as many components in the system as possible, as accurately as possible. This results in differential equation models that contain tens or hundreds of terms. However, for many applications there is some quantity of interest in the model that needs to be accurately computed. It is therefore important to identify which terms need to be included in the model in order to accurately compute the quantity of interest. This leads to a greater understanding of the model, and allows clearer hypotheses to be made about the important components and important physical effects in the biological system.

Another feature of models in systems biology is that different terms may be influential in different regions of the computational domain. A simple example of this is the FitzHugh Nagumo model, where the recovery variable has little effect during the activation phase, but is important during the recovery phase of the action potential. Therefore we are interested not only in identifying which terms in the model are important for accurately calculating the quantity of interest, but also in identifying regions of the computational domain where inclusion of the term in the model has an effect on the quantity of interest.

2.2 Existing methods for model analysis and reduction

2.2.1 Sensitivity analysis

The most commonly used technique for model analysis is sensitivity analysis. Sensitivity analysis studies will often set out to answer the question, "Which parameter values in the model equations have the greatest effect on a model output of interest?" Answering this question is important for deciding which parameters in a model need to be accurately estimated, and also for predicting how system behaviour will change in response to different interventions. A separate question that model analysis will often set out to answer is, "Which terms in the model equations have the greatest effect on a model output of interest?" The method developed in this thesis does not aim to answer the first question but does aim to answer the second one. In reviewing the sensitivity analysis literature we therefore focus on the extent to which the second question can be answered using sensitivity analysis.

We first discuss the relationship between the two questions posed above. Suppose that we are analyzing an ODE system of the form,

$$\frac{\mathrm{d}u}{\mathrm{d}t} + f(t,u) = 0, \qquad u(0) = u_0, \tag{2.1}$$

where $f(t,u) = \sum_{j=1}^{J} k_j f_j(t,u)$. Then $k_1, \ldots, k_J$ are termed linear parameters. Setting a linear parameter, k_j, to zero is equivalent to removing the term $f_j(t,u)$ from the model equations. Therefore if a sensitivity analysis shows that the model output is insensitive to variations in a linear parameter k_j over a range of values that includes zero, it follows that the corresponding term $f_j(t,u)$ has little effect on the model output.

Sensitivity analysis methods can broadly be divided into local methods and global methods. Local methods consider variations of parameter values around a fixed point in parameter space. If a local sensitivity method only considers $\pm 30\%$ variation in parameter values, then it cannot be used to determine whether a term can be removed from the model equations. Global sensitivity methods are capable, at least in theory, of exploring an arbitrarily large region of the parameter space. However because of the high computational cost of global sensitivity methods, these methods are often applied to a relatively small region of parameter space, as for example in [41].

The simplest form of sensitivity analysis is to repeatedly simulate the model with different parameter values and measure how quantities of interest in the model vary as a result of the parameter changes. The studies of Ihekwaba *et al* [34] and Romero *et al* [64] apply sensitivity analysis by repeated simulation to systems biology models, the first for NF-κB signalling, the second for cardiac electrophysiology. Parameters variations of -50%,$+100\%$ in [34] and -30%,$+30\%$ in [64] were applied to one parameter in the model at a time, and the results were used to identify which terms in the model outputs of interest were most sensitive to. The method used in [34] and [64] could be modified so that a -100% variation in parameter values is performed for a full set of linear parameters. This would indicate whether individual terms can be removed from the equations. However since the method considers parameters one at a time it would not be effective in generating a reduced model. An additional drawback of the method is that it is computationally expensive. It is not feasible to apply the method using a desktop computer for typical systems biology models which are nonlinear and contain tens or hundreds of components.

Metabolic Control Analysis (MCA) is a form of sensitivity analysis that calculates local derivatives of a model solution and of biochemical reaction rates [18, 26, 39]. The MCA methodology was developed to analyze properties of the system steady state for ODE models of metabolism. Key quantities in MCA are termed control coefficients and flux control coefficients. These can be interpreted as sensitivities of a parameter-augmented system [61]. Classical MCA is limited to studying the properties of steady state solutions. However a generalization of MCA was introduced in the study of Rand [60]. The method assesses the effect parameter perturbations on transient and or periodic behaviour of models as well as on steady state behaviour. Also, the method is capable of analyzing the effect of parameter perturbations on the whole of the model solution rather than on a single quantity of interest. The goal of the study in [60] was to assess the robustness and fragility of systems biology models such as NF-κB signalling and circadian clock models. Rand's results show that certain features of the model are robust to variations in the majority of their parameters. For example, sensitivities at maxima and minima of the model solution were calculated for the mammalian circadian clock model of Leloup and Goldbeter [45]. It was found that only 12 out of 56 parameters in the model had more than 1% of the maximum sensitivity for these features. This does suggest that it may be possible to remove terms from the model

without greatly changing features of interest in the model solution. However only first-order derivatives of the solution with respect to parameters are calculated in [60]. If higher-order derivatives are non-negligible, then for large parameter perturbations, the approximation using only the first-order derivatives will not accurately predict the effect of the perturbation on the model. If higher-order derivatives are negligible, there is a linear relationship between the parameters and the model solution over the whole parameter space. As we will see below this is not generally the case for systems biology models.

The study of Sobie, [69], analyzed the effect of parameter variations on key properties of cardiac electrophysiology models. Parameters were varied randomly and linear regression was performed for selected outputs of the model. Variations of $\sim \pm 30\%$ in parameter values were applied. Accurate linear regression models were found linking outputs of interest with model parameters. However these linear regression models were only validated within the $\pm 30\%$ range. If there was a linear relationship between parameters and outputs of interest over the whole parameter space, then the method used in [69] would be a global method, and could be used to identify redundant terms in the model. This point is not discussed in [69], but it has been shown for other models of a similar type that the relationship between parameters and outputs of interest is in general nonlinear. For example in [9, 16, 17], Estep, Buzby *et al* combine derivative-based sensitivity analysis with Monte-Carlo methods to estimate regions of parameter space where a linear relationship between model parameters and outputs of interest is valid. For a nonlinear multicomponent model of bubonic plague transmission, it was shown that regions where the linear approximation is accurate are relatively small [9]. Another method that has been used to assess linearity of the relationship between parameters and model outputs is the Morris method [49]. For a given output of interest, the Morris method returns a set of indices, μ_i and σ_i for each model parameter. A high value of μ_i indicates a high linear effect, and a high value of σ_i indicates nonlinear behaviour. In [67], Sher *et al* apply the Morris method to several recent cardiac electrophysiology models, and found that many parameters in the models have high σ_i values. A standard local sensitivity analysis can also be used to evaluate whether there is a nonlinear relationship between parameters and outputs. The sensitivity analysis in [34] showed that for an NF-κB signalling model the sensitivities were strongly dependent on the magnitude of the parameter change applied. This is evidence of a

nonlinear relationship between parameters and outputs of interest. The examples in [9, 16, 17, 34, 67] support the idea that for nonlinear multicomponent models, there is often a nonlinear relationship between model parameters and outputs of interest. This means that neither a linear regression analysis nor a first-order local derivative based sensitivity analysis can in general be used to determine whether terms can removed from systems biology models.

Other sensitivity analysis methods suffer from similar drawbacks to those discussed above. In summary, for the purpose of determining whether terms can be removed from large-scale nonlinear models, existing sensitivity methods are either (i) local in parameter space meaning that it is not possible to evaluate the effect of changing a parameter to zero [9, 16, 17, 18, 26, 39, 60, 69], or (ii) prohibitively computationally expensive [34, 41, 64]. The approach of Estep *et al* which combines local information with Monte Carlo simulations may offer the best approach for model reduction, but no work in this direction has yet been done.

For broader reviews of sensitivity analysis in the context of chemical kinetics, the reader is referred to the work of Rabitz [59], Turanyi [73], and Saltelli [66]. The field of chemical kinetics faces similar challenges to systems biology in the analysis of large-scale nonlinear models. An interesting area of future work, outside the scope of this thesis, would be to review the chemical kinetics literature in more detail in order to evaluate whether there are less well-known sensitivity methods developed in chemical kinetics that would be useful in systems biology. Some work in this direction has been done, for example [47, 71].

2.2.2 Perturbation analysis and other methods for model reduction

In the previous section we assessed whether sensitivity analysis can be used to answer the question, "Which terms in a model's equations have the greatest effect on a model output of interest?" We found that for large-scale nonlinear models it is challenging, if not impossible, to use sensitivity analysis to answer this question.

Here, we review another methodology for answering the same question – perturbation analysis. Broadly speaking the goal of perturbation analysis is to identify small terms that can be removed from differential equation models. An example of this is fast-slow analysis for ODEs, where model components that change on a fast time-scale are set to their quasi-steady state and model components that change on a slow time-scale

are set to their initial conditions. Fast-slow analysis has a long history in chemistry and biology dating back to one of the earliest models of enzyme kinetics, the Michaelis-Menten equations [8, 40]. Canonical model problems where a standard perturbation analysis can be applied include the Van der Pol oscillator [28], the Oregonator model of chemical oscillations [19, 35, 76], and the FitzHugh Nagumo equations [21].

While it is relatively straightforward to apply perturbation methods to equations with a small number of components, applying perturbation methods to larger systems of equations is more difficult. Examples of successful fast-slow analysis for systems biology models include approximations of a cardiac electrophysiology model, Bernus *et al* [6], a Calcium-Induced Calcium Release (CICR) model, Hinch *et al* [29], and an NF-κB signalling model, Krishna *et al* [42]. In each case, considerable knowledge of the system and mathematical skill is required to make the correct approximations. To our knowledge, there does not exist a general theory or a standard methodology for doing fast-slow analysis on large-scale nonlinear systems, and it is therefore a time-consuming process that requires specialist expertise.

In control engineering there are well-developed methods for model reduction of large-scale systems [12, 24, 48, 77]. However control engineers are usually interested in reproducing the input-output properties of a system, rather than understanding the role of each component in the system. As a result many of the methods developed in control engineering, such as balanced truncation, introduce a coordinate transform that turns the model into a black box. It is difficult to identify biologically meaningful variables in the model after such a coordinate transform. Control engineering methods that do not introduce a coordinate transform have been applied in systems biology [1, 58]. For example, in [1], Anderson, Papachristodoulou *et al* develop a method for automatically identifying components in a large-scale EGFR-MAPK signalling pathway model that can be approximated by their quasi-steady state. Although having the benefit of being automatic, the method is limited to studying properties of the linearized system in the neighbourhood of equilibrium points, and therefore cannot give a full characterization of system behaviour.

2.3 Previous work on the *a posteriori* model reduction method

We now consider the methodology that will be the subject of the remainder of the thesis, model reduction using *a posteriori* analysis, [75]. We will often refer to the method as *a posteriori* model reduction. *A posteriori* model reduction sets out to answer the same question as perturbation methods, "Which terms in the model equations have the greatest effect on a model output of interest?" In contrast to perturbation methods, *a posteriori* model reduction is an automatic method that could be applied within a software package by researchers without specialist mathematical skills. Furthermore the method has wide applicability, enabling analysis of models that would be difficult or impossible to analyze with perturbation methods.

A posteriori error analysis is a methodology that originated in the Finite Element Method (FEM) community. The basic idea of *a posteriori* error analysis in FEM is to use a computed numerical solution and the solution to an adjoint problem to identify regions of the computational domain where further refinement of the computational mesh is required in order to accurately calculate some quantity of interest of the model solution. *A posteriori* methods were pioneered in the 1970's by Babuška and Rheinboldt [3, 4], and have since been used extensively as a method for efficiently constructing computational meshes. The resulting meshes indicate regions of the domain where a more accurate discrete approximation is required to accurately calculate some quantity of interest. Comprehensive reviews of *a posteriori* error analysis can be found in [5, 15, 23].

A posteriori error analysis has been used extensively within the Finite Element Method (FEM) for constructing discrete approximations to continuous differential operators. However the application of the *a posteriori* methodology for constructing continuous-domain differential equation approximations is relatively recent. To date there is, to our knowledge, one paper on *a posteriori* model reduction [75], where the method is introduced for initial-value ODEs and applied to a model of cardiac electrophysiology. Briefly, *a posteriori* model reduction uses the computed solution to a reduced differential equation, and the solution to an adjoint problem to identify regions of the domain where the original differential equation needs to be solved in order to accurately calculate some quantity of interest of the model solution. A full exposition

of the method is given in Chapter 3. In [75], the model output of interest was chosen to be the average transmembrane potential (an important variable in electrophysiology models) over a given time interval. Application of the method indicated that the action potential upstroke can be accurately reproduced using the dynamics of only 3 out of 22 components in the model. The result reproduces known physiology, but does so in a way that does not require prior knowledge of the system, or specialist mathematical skills.

The established advantages of using *a posteriori* model reduction over other methods are that it is automatic, in the sense of not requiring any analytical work beyond the development of the method, and it can be applied to large-scale nonlinear models. In this thesis we aim to demonstrate two additional advantages. One is that the method can be applied to general mixed-type BVP and PDE models. This is important in the context of systems biology, where models increasingly include different physical effects that are described mathematically by different types of differential operator. Secondly we will develop the method so that the reduced model can contain any subset of the terms contained in the original model. This means, for example, that a quasi-steady state or an initial value approximation can be made for initial-value ODEs, and an algebraic reaction approximation or a diffusion approximation can be made for reaction-diffusion equations. This allows for a full characterization of the effect of all terms in the model.

3

Initial-Value Ordinary Differential Equations

In the first section of this chapter we review *a posteriori* model reduction as it is presented in [75]. In [75], reduced model variables are approximated by removing algebraic terms from the equations in regions of the domain that are automatically identified by the algorithm. In the second section, we present a novel variant of *a posteriori* model reduction that approximates reduced model variables by their quasi-steady state. We apply both forms of the method to two model problems, one of which, the Michaelis-Menten equations, has a variable that can be well-approximated by its quasi-steady state. This is automatically identified by the new form of the *a posteriori* model reduction method. In the final section we demonstrate that, for the model problems analyzed in this chapter, a 10-20 fold increase in efficiency may be obtained by increasing one of the algorithm parameters, causing a negligible change in the results.

3.1 Initial-value approximation

3.1.1 Derivation of error bound

In [75], an *a posteriori* error bound in a user-defined functional of interest is derived for a differential equation approximation to a general system of first-order Ordinary Differential Equations (ODEs). We re-state the derivation here, following closely that given in [75], so that the novel developments of the method in this thesis are easy to identify. The method is given the name *a posteriori* model reduction because a computed solu-

tion to a reduced differential equation is used to estimate localized errors. Progressively more accurate solutions are obtained iteratively by changing the differential operator in the regions of the domain that contribute most to the error in the quantity of interest. *A posteriori* error analysis originates from the Finite Element Method (FEM) literature [14]. In FEM, *a posteriori* error analysis is used to identify regions of the domain where a finer mesh is required to accurately compute a user-defined functional of interest. However in the present context, the focus is on the differential operator rather than the numerical solution (which we assume has been accurately obtained).

Before applying the algorithm the user must decide what the linear functional of interest will be, and what the reduced differential equation is. Here, we construct a reduced differential equation by removing terms from the original differential equation, but the method does not require the reduced differential equation to be constructed in this way.

Initially, the reduced differential equation is solved over the whole domain. Regions of the domain contributing most to the error in the user-defined quantity of interest are identified using *a posteriori* error analysis. Then, the original differential operator is applied in regions identified by the algorithm, and the reduced differential operator is applied elsewhere, and the solution to the reduced model is re-calculated. This process of identifying regions of the domain and re-calculating the solution of the reduced model continues until the user-defined quantity of interest is accurately calculated. This process will always terminate because it can continue until the original differential equation is applied over the whole domain, in which case the user-defined quantity of interest is identical in the reduced and original models. If one or more terms in the equations are not required for accurately calculating the quantity of interest, the algorithm should be able to identify this and generate a reduced model where the original differential operator is applied is some regions of the domain, and the reduced differential operator is applied in others. It is important to note that the goal of the method is to accurately calculate the user-defined functional with the reduced model, and so a reduced model generated using the *a posteriori* method typically will not accurately reproduce the solution to the original problem over the whole domain.

We define the primal problem to be

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \boldsymbol{f}(t, \mathbf{u}) = \mathbf{0}, \tag{3.1}$$

where $\boldsymbol{u}(t)$ is an N-component vector defined on the interval $t = [0, T]$, and specify initial conditions $\boldsymbol{u}(0) = \boldsymbol{u}_0$.

Suppose that $\boldsymbol{U}$ is an approximation of $\boldsymbol{u}$. We require $\boldsymbol{U}$ to satisfy the following conditions, which are required for deriving the error bound below:

1. $\boldsymbol{U}(t)$ is continuous for $0 < t < T$;
2. $\dfrac{\mathrm{d}\boldsymbol{U}(t)}{\mathrm{d}t}$ is continuous for $0 < t < T$, except at a finite set of points in time.
3. $\boldsymbol{U}(t)$ satisfies the initial conditions of the primal problem, i.e., $\boldsymbol{U}(0) = \boldsymbol{u}(0)$.

We are interested in a user-defined linear functional of the solution, which we denote by $J(\boldsymbol{u})$. Examples of linear functionals are the average of a component of the model solution over the interval $[0, T]$, and the value of a component of the model solution at a point in $[0, T]$. By the Riesz representation theorem any linear functional can be written as $J(\boldsymbol{v}) = \int_0^T \boldsymbol{v} \cdot \boldsymbol{g} \, \mathrm{d}t$ where $\boldsymbol{v}, \boldsymbol{g} \in L^2([0, T])$, see for example [65]. We call $\boldsymbol{g}$ a weighting function, and it is usually straightforward to work out what it should be given the definition of $J(\boldsymbol{v})$. If $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$ is small, then $\boldsymbol{U}$ is a good approximation to $\boldsymbol{u}$, at least for calculating the quantity of interest.

We now define an adjoint problem. First we give a formal definition of the adjoint operator. Suppose that $\boldsymbol{u}$ is a member of some function space V, and L is a differential operator, and $\langle \cdot, \cdot \rangle$ is an inner product on V. Then the adjoint, L^*, is the differential operator that satisfies,

$$\langle L\boldsymbol{u}, \boldsymbol{v} \rangle = \langle \boldsymbol{u}, L^*\boldsymbol{v} \rangle, \tag{3.2}$$

for all $\boldsymbol{u}, \boldsymbol{v} \in V$ [44]. However for nonlinear differential operators it is not possible to find an L^* that satisfies (3.2) for all $\boldsymbol{u}, \boldsymbol{v}$. We therefore give a weaker definition of the adjoint that applies to nonlinear operators. Given $\boldsymbol{u}, \boldsymbol{U} \in V$ an adjoint, L^*, is a differential operator that satisfies,

$$\langle L(\boldsymbol{u} - \boldsymbol{U}), \boldsymbol{v} \rangle = \langle \boldsymbol{u} - \boldsymbol{U}, L^*\boldsymbol{v} \rangle, \tag{3.3}$$

for all $v \in V$.

The adjoint equation used below is of the form $L^*\boldsymbol{v} = \boldsymbol{g}$. As will be shown be shown below, the adjoint solution plays a key role in *a posteriori* analysis, because

the adjoint solution quantifies the effect of errors in the reduced model solution on $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$. The adjoint solution is heavily dependent on the choice of $\boldsymbol{g}$. This is important to remember below, in order to understand how the choice of functional affects the reduced model that is generated.

The adjoint problem is defined as follows,

$$-\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t} + A\boldsymbol{\phi} = \mathbf{g}, \tag{3.4}$$

where $\boldsymbol{\phi}(t)$ is an N-component vector defined on the interval $t = [0, T]$. We apply the final condition $\boldsymbol{\phi}(T) = \mathbf{0}$ at $t = T$. The choice of boundary conditions on the primal and adjoint problems is important for the error bound derivation, as correct formulation of the boundary conditions ensures that boundary terms at $t = 0$ and $t = T$ that arise from integration by parts will be zero. The function $\boldsymbol{g}$ is the weighting function in the functional of interest.

The matrix A has entries given by,

$$A_{ij}(t) = \int_0^1 \frac{\partial f_j}{\partial u_i}(t, \boldsymbol{\alpha}(s,t))\ \mathrm{d}s, \quad \text{for } i, j = 1, \ldots, N, \tag{3.5}$$

where

$$\boldsymbol{\alpha}(s,t) = s\boldsymbol{u}(t) + (1-s)\boldsymbol{U}(t). \tag{3.6}$$

The following property holds for the coefficient matrix $A(t)$ and will be used in the error bound derivation below,

$$\begin{aligned}
\sum_{i=1}^{N} A_{ij}(t)(u_i(t) - U_i(t)) &= \sum_{i=1}^{N} \int_0^1 \frac{\partial f_j}{\partial u_i}(\boldsymbol{\alpha}(s,t))(u_i - U_i)\ \mathrm{d}s \\
&= \sum_{i=1}^{N} \int_0^1 \frac{\partial f_j}{\partial \alpha_i}\frac{\partial \alpha_i}{\partial s}\ \mathrm{d}s \\
&= \int_0^1 \frac{\mathrm{d}f_j}{\mathrm{d}s}\ \mathrm{d}s. \\
&= f_j(t, \boldsymbol{u}) - f_j(t, \boldsymbol{U}).
\end{aligned} \tag{3.7}$$

Now we use the solution of the adjoint problem to derive a bound on the error in $J(\cdot)$ introduced by approximating $\boldsymbol{u}$ by $\boldsymbol{U}$.

$$
\begin{aligned}
J(\boldsymbol{u}) - J(\boldsymbol{U}) &= \int_0^T (\boldsymbol{u} - \boldsymbol{U}) \cdot \boldsymbol{g} \, \mathrm{d}t \\
&= \int_0^T (\boldsymbol{u} - \boldsymbol{U}) \cdot \left(-\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t} + A\boldsymbol{\phi} \right) \, \mathrm{d}t \\
&= \int_0^T \sum_{i=1}^{N} (u_i - U_i) \left(-\frac{\mathrm{d}\phi_i}{\mathrm{d}t} + \sum_{j=1}^{N} A_{ij}\phi_j \right) \, \mathrm{d}t \\
&= \int_0^T \sum_{i=1}^{N} \left(\left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + (u_i - U_i) \sum_{j=1}^{N} A_{ij}\phi_j \right) \, \mathrm{d}t \\
&\qquad + \sum_{i=1}^{N} - \Big[(u_i - U_i) \ \phi_i \Big]_0^T,
\end{aligned} \tag{3.8}
$$

using integration by parts. We have assumed that $\boldsymbol{u}, \boldsymbol{U}$, and $\boldsymbol{\phi}$ are continuous. The boundary term is zero because $\boldsymbol{u}(0) = \boldsymbol{U}(0)$, and $\boldsymbol{\phi}(T) = \boldsymbol{0}$. Now we use identity (3.7) to obtain,

$$
\begin{aligned}
&\int_0^T \sum_{i=1}^{N} \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i \ \mathrm{d}t + \int_0^T \sum_{j=1}^{N} \sum_{i=1}^{N} A_{ij}(u_i - U_i)\phi_j \ \mathrm{d}t \\
= &\int_0^T \sum_{i=1}^{N} \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i \ \mathrm{d}t + \int_0^T \sum_{j=1}^{N} \left(f_j(t, \boldsymbol{u}) - f_j(t, \boldsymbol{U}) \right) \phi_j \ \mathrm{d}t.
\end{aligned} \tag{3.9}
$$

All terms involving $\boldsymbol{u}$ disappear by definition of the primal problem, (3.1). We may therefore write our error bound as,

$$
|J(\boldsymbol{u}) - J(\boldsymbol{U})| \leq \sum_{i=1}^{N} \sum_{k=1}^{M} \int_{t_{k-1}}^{t_k} \left| \frac{\mathrm{d}U_i}{\mathrm{d}t} + f_i(t, \boldsymbol{U}) \right| \ |\phi_i| \ \mathrm{d}t,
$$

where $t_0 = 0 < t_1 \ldots < t_M = T$ is a subdivision of the computational domain.

3.1.2 Application to model problems

In [75], the error bound above was used in an algorithm for constructing reduced models. Here, we re-state the algorithm in its original form, and then apply it to two model problems.

We choose the following form for the reduced primal problem,

$$
\frac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}t} + \hat{\boldsymbol{f}}(t, \boldsymbol{U}) = \boldsymbol{0}, \tag{3.10}
$$

where $\boldsymbol{U}(t)$ is an N-component vector defined on the interval $t = [0, T]$, and,

$$\hat{f}_i(t, \boldsymbol{U}) = \begin{cases} f_i(t, \boldsymbol{U}), & \text{for } t \in S_i, \\ 0, & \text{otherwise.} \end{cases}$$

where S_i is a union of one or more intervals in $[0, T]$, for i from 1 to N, to be determined by the algorithm. The initial conditions of the original equations are applied: $\boldsymbol{U}(0) = \boldsymbol{u}_0$. We call this the initial-value approximation, because for any interval, $[t_1, t_2]$ that is not in S_i, U_i is equal to $U_i(t_1)$ over that interval. There are two reasons why the initial-value approximation may be suitable in this context. One reason is that if a component of the model has little or no effect on the quantity of interest, then an accurate solution for that component is not necessary for accurately calculating the quantity of interest. The other reason is that if the initial-value approximation is sufficiently close to the model solution for a given component, then the quantity of interest can be accurately calculated using the initial-value approximation of that component.

The initial-value approximation algorithm is then defined as follows:
Initialization: Set $S_i = \emptyset$ for i from 1 to N, so that, in this case, $\dfrac{\mathrm{d}U_i}{\mathrm{d}t} = 0$ for all t, and for i from 1 to N. Compute $\boldsymbol{\phi}$ using equation (3.4).
Iteration:

1. Compute `err(1:N,1:M)` where

$$\texttt{err}(\texttt{i},\texttt{k}) = \int_{t_{k-1}}^{t_k} \left| \frac{\mathrm{d}U_i}{\mathrm{d}t} + f_i(t, \boldsymbol{U}) \right| |\phi_i| \, \mathrm{d}t. \tag{3.11}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding time intervals to S_i for i from 1 to N.

3. Re-compute $\boldsymbol{U}$, $\boldsymbol{\phi}$.

Termination: Stop when

$$|J(\boldsymbol{u}) - J(\boldsymbol{U})| < TOL.$$

The inputs to the algorithm are as follows: N_{cont}, the number of t-intervals added to $\boldsymbol{S}$ on each iteration of the algorithm; TOL, the desired accuracy in the quantity of interest; and, $\boldsymbol{g}$, the weighting function in the quantity of interest.

Note that if S_i is the entire computational domain for $i = 1, \ldots, N$, then $J(\boldsymbol{u}) = J(\boldsymbol{U})$. This means that the algorithm will always terminate because it is always possible to obtain $|J(\boldsymbol{u}) - J(\boldsymbol{U})| < TOL$, for $TOL > 0$.

To illustrate the effectiveness of the method we apply the algorithm to two model problems. The first is the linear ODE,

$$\begin{aligned} \frac{\mathrm{d}u_1}{\mathrm{d}t} &= -u_1, \\ \frac{\mathrm{d}u_2}{\mathrm{d}t} &= u_1 + u_2, \end{aligned} \tag{3.12}$$

for $0 < t < T$. with the initial conditions, $u_1(0) = u_2(0) = 1$. As shown in Figure 3.1, u_1 decays exponentially to zero, and u_2 grows exponentially. We therefore expect that if the u_1 solution is obtained using a reduced differential operator for large t, the change in the quantity of interest will be negligible. Also, by inspecting equation (3.12), we can see that u_1 is independent of u_2. We therefore expect that if the quantity of interest is independent of u_2, the u_2 solution does not need to be accurately computed. This problem was analyzed in [75], and we reproduce the results. We choose the following quantities of interest,

$$J(\boldsymbol{u}) = \int_0^T u_1 \ \mathrm{d}t,$$

and

$$J(\boldsymbol{u}) = \int_0^T u_2 \ \mathrm{d}t.$$

When $J(\boldsymbol{u}) = \displaystyle\int_0^T u_1 \ \mathrm{d}t = O(1)$, the algorithm correctly identifies that $J(\cdot)$ can be computed to within $TOL = 10^{-3}$ with $\mathrm{d}u_2/\mathrm{d}t = 0$ for all t. When $J(\boldsymbol{u}) = \displaystyle\int_0^T u_2 \ \mathrm{d}t = O(10^4)$, the algorithm identifies that $J(\cdot)$ can be computed to within $TOL = 0.1$ with $\mathrm{d}u_1/\mathrm{d}t = 0$ for $t > 5.75$ – Figure 3.1.

The second problem is the Michaelis-Menten model of enzyme kinetics, defined as follows,

$$\begin{aligned} \frac{\mathrm{d}s}{\mathrm{d}t} &= -s + c(s + K - \lambda), \\ \epsilon\frac{\mathrm{d}c}{\mathrm{d}t} &= s - (s + K)c, \end{aligned} \tag{3.13}$$

where K, λ, ϵ are constants, $0 < \epsilon \ll 1$, $0 < t < T$, with the initial conditions, $s(0) = 1$, $c(0) = 0$. The Michaelis-Menten equations have the property that on an O(1) timescale,

the variable c is at quasi-steady state. Since the existing method is designed to identify slow variables it is not well-suited to this second problem. We present the results here in order to demonstrate that the new variant of the method to be introduced in the next section is better for problems where one or more components are at quasi-steady state over the time-scale of interest. The same quantities of interest as for the previous problem are used, $J(\boldsymbol{u}) = \int_0^T u_1 \, \mathrm{d}t$ and $J(\boldsymbol{u}) = \int_0^T u_2 \, \mathrm{d}t$.

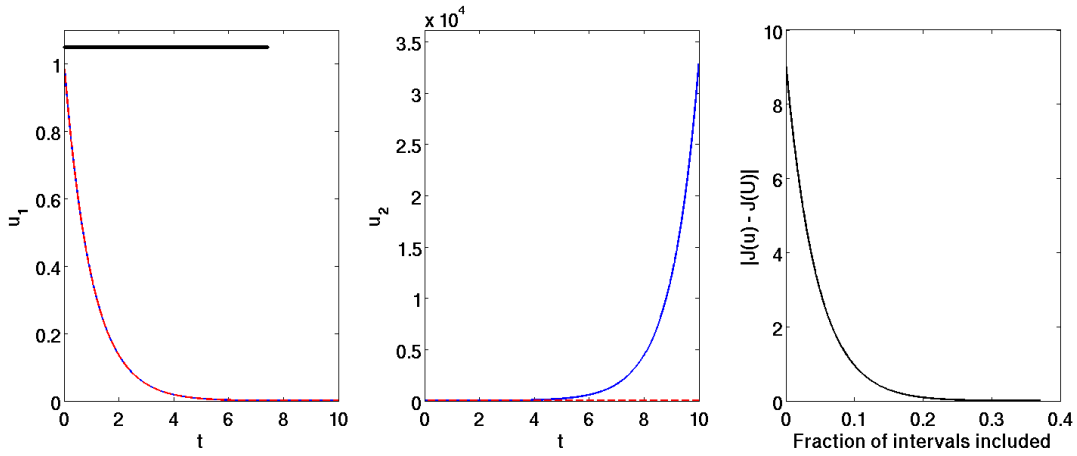
For both quantities of interest, both $\mathrm{d}u_1/\mathrm{d}t$ and $\mathrm{d}u_2/\mathrm{d}t$ cannot be approximated by $\mathrm{d}u_i/\mathrm{d}t = 0$ until $t > 3.8$. The method has failed to identify the desired information in the problem, which is that $\mathrm{d}u_2/\mathrm{d}t$ can be approximated by its quasi-steady state on an $O(1)$ time-scale. The reason the method has failed is because the initial-value approximation has been chosen as the form of the reduced model. We address this issue in the next section, by re-formulating the error bound derivation and the algorithm, so that a quasi-steady state approximation can be used as the reduced model.

3.2 Quasi-steady state approximation

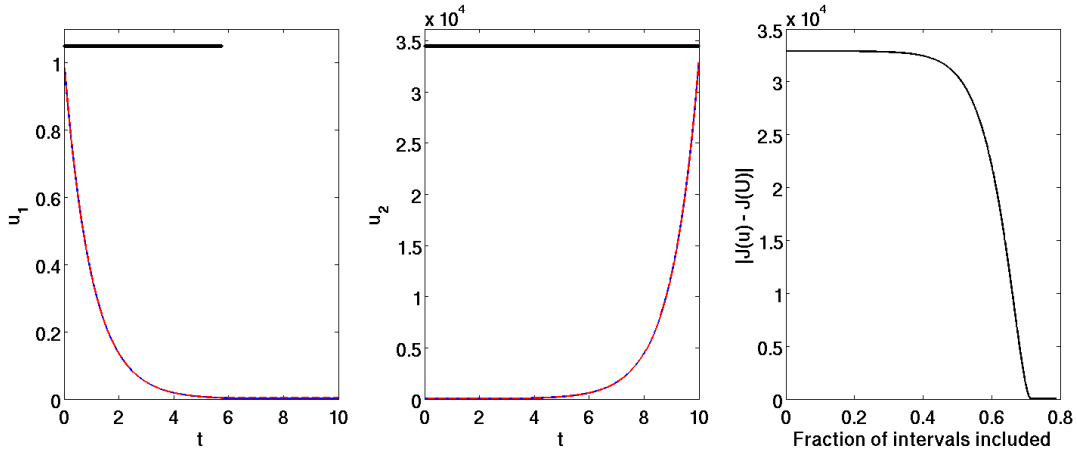
In the previous section we reproduced the form and results of an existing method, from [75], for automatically constructing reduced models. The form of reduced model was called the initial-value approximation because for any interval not included in the reduced model for a given component, that component was set to the initial value of the component at the beginning of the interval.

In this section we introduce a novel variant of the method. We will call it the quasi-steady state approximation because for any interval that is not included in the reduced model for a given component, that component will be set to the quasi-steady state value of the component over the interval. As for the initial-value approximation, there are two reasons why the quasi-steady state approximation may be suitable in this context. One reason is that if a component of the model has little or no effect on the quantity of interest, then an accurate solution for that component is not necessary for accurately calculating the quantity of interest. The other reason is that if the quasi-steady state approximation is sufficiently close to the model solution for a given component, then the quantity of interest can be accurately calculated using the quasi-steady state approximation of that component.

The aim of the new method is to automatically identify the regions of the domain where a quasi-steady state approximation results in a relatively small change in the

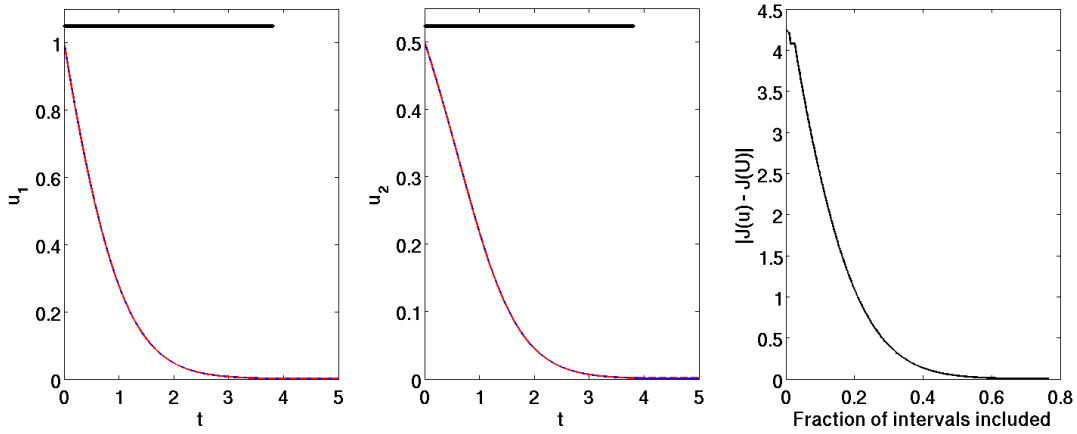


(a) $J(\boldsymbol{u}) = \int_0^{10} u_1 \, \mathrm{d}t$, $TOL = 10^{-3}$

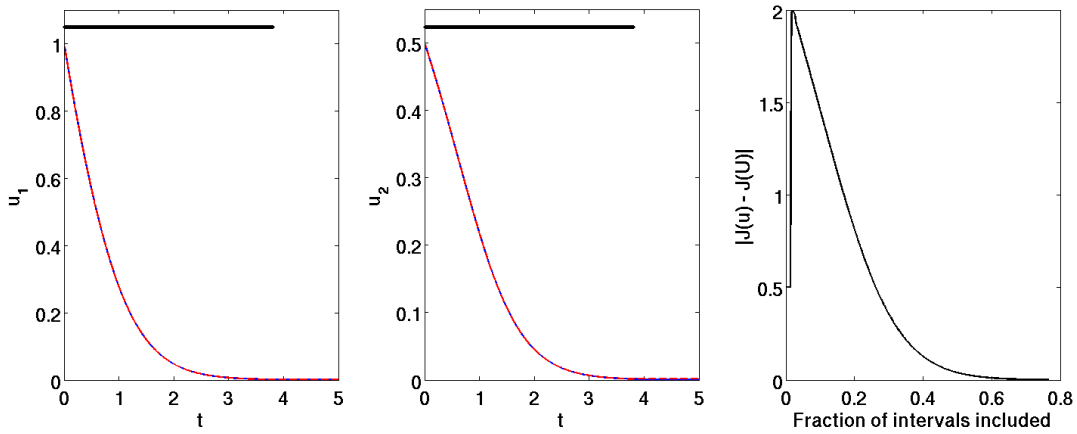


(b) $J(\boldsymbol{u}) = \int_0^{10} u_2 \, \mathrm{d}t$, $TOL = 10^{-1}$

Figure 3.1: Initial-value approximation for linear IVP in automatically identified regions of the domain. *Left and middle figures*: solid blue line is the numerical solution to the linear IVP (3.12), obtained using the forwards Euler method with $\Delta t = 10^{-3}$. The initial conditions were $u_1(0) = u_2(0) = 1$. The dashed red line is U, and the black lines above the curves represent the time intervals where the algebraic terms are included in the reduced model. Here $N_{cont} = 5$. *Right figure* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$, as a function of the fraction of the domain included in the reduced model.



(a) $J(\boldsymbol{u}) = \int_0^5 u_1 \, \mathrm{d}t$



(b) $J(\boldsymbol{u}) = \int_0^5 u_2 \, \mathrm{d}t$

Figure 3.2: Initial-value approximation for Michaelis-Menten equations in automatically identified regions of the domain. *Left and middle figures*: solid blue line is the numerical solution to the Michaelis-Menten equations (3.13), obtained using the forwards Euler method with $\Delta t = 10^{-3}$. The initial conditions were $u_1(0) = s(0) = 1$, $u_2(0) = c(0) = 0$. The dashed red line is U, and the black lines above the curves represent the time intervals where the algebraic terms are included in the reduced model. Here $\epsilon = 0.01$, $K = 1$, $\lambda = 2$, $N_{cont} = 5$, and $TOL = 10^{-3}$. *Right figure* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$, as a function of the fraction of the domain included in the reduced model.

quantity of interest, $J(\cdot)$. One approach to do this would be to start by setting all model components to their steady-state, and then iteratively include the time derivative where it is required. However this approach will fail if the residual function from the previous section is used to identify regions that need to be included. The reason for this is that the steady-state solution satisfies the original problem (without the initial conditions). Therefore the residual of the steady-state approximation is zero everywhere.

This simplest approach fails, and so we need to consider other variations of the method in order to automatically generate the quasi-steady state approximation. When deriving the original error bound, the adjoint problem is formulated so that $\boldsymbol{u}$ is eliminated, and the error is expressed as the product of the adjoint solution with a residual, which is a function of $\boldsymbol{U}$. However the adjoint problem can be re-formulated, so that the error is expressed as the product of the a 'pseudo-adjoint' solution with a residual which is a function of $\boldsymbol{u}$, plus terms that arise from discontinuities in the solution of the reduced primal and adjoint problems. This is what we do below. The re-formulated adjoint problem is referred to as a 'pseudo-adjoint' problem because, as a result of the terms arising from discontinuities, the 'pseudo-adjoint' operator does not satisfy the adjoint definition given in equation (3.3). Alternative constraints can be imposed on the re-formulated adjoint problem to make it to satisfy equation (3.3). However this is not desirable for reasons that are discussed below.

An algorithm for model reduction can be formulated in a similar way as before and has been successful when applied to the Michaelis-Menten ODEs that model enzyme kinetics. In the Michaelis-Menten model, one of the model components can be well-approximated by its quasi-steady state. This is automatically identified by the new variant of *a posteriori* model reduction.

3.2.1 Derivation of error bound

As before, the primal problem is a system of initial-value ODEs of the form,

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \boldsymbol{f}(t, \mathbf{u}) = \mathbf{0}, \tag{3.14}$$

where $\boldsymbol{u}(t)$ is an N-component vector defined on the interval $t = [0, T]$, and the initial conditions are $\boldsymbol{u}(0) = \boldsymbol{u}_0$.

In contrast to the method presented in the previous section, we have to define the form of the reduced primal problem in order to derive the error bound. Therefore we define the reduced primal problem to be,

$$\boldsymbol{\chi}(t)\frac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}t} + \boldsymbol{f}(t,\boldsymbol{U}) = \boldsymbol{0}, \tag{3.15}$$

where

$$\chi_i(t) = \begin{cases} 1 & \text{for } t \in S_i \\ 0 & \text{otherwise,} \end{cases}$$

where S_i is a union of one or more intervals in the t-domain to be determined by the algorithm. Under this definition, component i evolves according to the ith ODE in regions where $\chi_i(t) = 1$, and according the the quasi-steady state in regions where $\chi_i(t) = 0$. The indicator $\chi_i(t)$ may change from 1 to 0, or 0 to 1 an arbitrary number of times. At points where $\chi_i(t)$ changes from 1 to 0, there is a discontinuity in U_i as that component 'jumps' to its quasi-steady state. At points where $\chi_i(t)$ changes from 0 to 1, we choose to enforce continuity of U_i by using the quasi-steady at those time points as the initial condition for the ODE in the adjacent region where $\chi_i(t) = 1$.

Setting the initial conditions in the reduced model is more complicated than before, because if we are solving the quasi-steady state at $t = 0$ then an initial condition is not required, and cannot be applied. If the system has a single steady-state, we set the initial conditions of the reduced model as follows:

$$U_i(0) = \begin{cases} u_i(0) & \text{if } 0 \in S_i \\ \text{solution to } f_i(0,\boldsymbol{U}) = 0 & \text{otherwise.} \end{cases}$$

If the system has multiple steady states then $f_i(0,\boldsymbol{U}) = 0$ has multiple solutions. In this case the initial conditions of the reduced model can be set as follows:

$$U_i(0) = \begin{cases} u_i(0) & \text{if } 0 \in S_i \ , \\ \lim\limits_{s\to\infty} v_i(s), \text{ where } \dfrac{\mathrm{d}v_i}{\mathrm{d}t} + f_i(s,\boldsymbol{v}) = 0 & \text{otherwise,} \end{cases}$$

where $\boldsymbol{v}(0) = \boldsymbol{u}(0)$. If one or more components of the system have no steady state solution, the method is not applicable to these components, and the first-order derivative term has to included over the whole domain.

We now re-define the adjoint problem. We can think of the re-formulated adjoint problem as being approximately adjoint to the reduced model, whereas the adjoint

problem used before is adjoint to the original model. We will refer to the re-formulated adjoint problem as an adjoint, even though it does not satisfy equation (3.3).

We are still interested in the same functional of the solution, $J(\boldsymbol{v}) = \langle \boldsymbol{v}, \boldsymbol{g} \rangle$. The new adjoint problem is to find $\boldsymbol{\phi}$ satisfying,

$$-\boldsymbol{\chi}(t)\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t} + A(t)\boldsymbol{\phi} = \boldsymbol{g}, \tag{3.16}$$

where A is defined in the same way as before:

$$A_{ij}(t) = \int_0^1 \frac{\partial f_j}{\partial u_i}(t, \boldsymbol{\alpha}(s,t))\mathrm{d}s, \tag{3.17}$$

where

$$\boldsymbol{\alpha}(s,t) = s\boldsymbol{u}(t) + (1-s)\boldsymbol{U}(t). \tag{3.18}$$

As in the previous section, the adjoint problem is solved backwards in time. In contrast to the previous section, the adjoint problem now contains an indicator function $\boldsymbol{\chi}$. Therefore the final condition at $t = T$ is set in a similar way to the initial condition of the reduced model:

$$\phi_i(T) = \begin{cases} 0 & \text{if } T \in S_i \\ \text{solution to } \displaystyle\sum_{j=1}^{N} A_{ij}(T)\ \phi_j = g_i & \text{otherwise.} \end{cases}$$

Note, a solution to these equations is not guaranteed. However once A has been calculated, it is easy to check that whether these equations have a solution by checking that the linear system is non-singular. If there are multiple solutions to the equations, a unique condition can be defined as follows:

$$\phi_i(T) = \begin{cases} 0 & \text{if } T \in S_i\ , \\ \displaystyle\lim_{s\to-\infty} v_i(s), \text{ where } \frac{\mathrm{d}v_i}{\mathrm{d}t} + \sum_{j=1}^{N} A_{ij}(T)\ \phi_j = g_i & \text{otherwise,} \end{cases}$$

where $\boldsymbol{v}(T) = 0$.

Now we use the solution of the adjoint problem to derive a bound on the error in $J(\cdot)$ introduced by approximating $\boldsymbol{u}$ by $\boldsymbol{U}$.

$$
\begin{aligned}
J(\boldsymbol{u}) - J(\boldsymbol{U}) &= \int_0^T (\boldsymbol{u} - \boldsymbol{U}) \cdot \boldsymbol{g} \, \mathrm{d}t \\
&= \int_0^T (\boldsymbol{u} - \boldsymbol{U}) \cdot \left(-\boldsymbol{\chi} \frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t} + A\boldsymbol{\phi} \right) \, \mathrm{d}t \\
&= \int_0^T \sum_{i=1}^{N} (u_i - U_i) \left(-\chi_i(t) \frac{\mathrm{d}\phi_i}{\mathrm{d}t} + \sum_{j=1}^{N} A_{ij}\phi_j \right) \mathrm{d}t \\
&= \int_0^T \sum_{i=1}^{N} \left(\left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \chi_i \phi_i + (u_i - U_i) \sum_{j=1}^{N} A_{ij}\phi_j \right) \, \mathrm{d}t + \\
&\qquad \sum_{i=1}^{N} - \Big[(u_i - U_i) \, \chi_i \phi_i \Big]_0^T + \sum_{i=1}^{N} \sum_{k=0}^{M_i} \Big[(u_i - U_i) \chi_i \phi_i \Big]_{t_k^-}^{t_k^+}, \qquad (3.19)
\end{aligned}
$$

using integration by parts. The term $[(u_i - U_i)\chi_i\phi_i]_{t_k^-}^{t_k^+}$ is the jump in $(u_i(t_k) - U_i(t_k))\chi_i(t_k)$ $\phi_i(t_k)$ at the points, $\{t_k\}$ where χ_i is discontinuous. The notation t_k^+ denotes the limit as t tends to t_k from above, and t_k^- denotes the limit as t tends to t_k from below. These discontinuities arise on the boundary of $\boldsymbol{S}$, where there is a transition from solving a 1st order ODE to solving a quasi-steady state equation. The first boundary term disappears because either $u_i(0) - U_i(0) = 0$, or $\chi_i(0) = 0$ at $t = 0$, and either $\phi_i(T) = 0$ or $\chi_i(T) = 0$ at $t = T$.

Now we use identity (3.7),

$$
\begin{aligned}
&\int_0^T \sum_{i=1}^{N} \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \chi_i \phi_i \, \mathrm{d}t + \int_0^T \sum_{j=1}^{N} \sum_{i=1}^{N} A_{ij}(u_i - U_i)\phi_j \, \mathrm{d}t \\
&= \int_0^T \sum_{i=1}^{N} \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \chi_i \phi_i \, \mathrm{d}t + \int_0^T \sum_{j=1}^{N} \left(f_j(t, \boldsymbol{u}) - f_j(t, \boldsymbol{U}) \right) \phi_j \, \mathrm{d}t. \qquad (3.20)
\end{aligned}
$$

The integral terms involving $\boldsymbol{U}$ disappear by definition of the reduced primal problem, (3.15). We may therefore write our error bound as,

$$
|J(\boldsymbol{u}) - J(\boldsymbol{U})| \leq \sum_{i=1}^{N} \sum_{k=1}^{M} \int_{t_{k-1}}^{t_k} \left| \chi_i(t) \frac{\mathrm{d}u_i}{\mathrm{d}t} + f_i(t, \boldsymbol{u}) \right| |\phi_i| \, \mathrm{d}t + \Big[(u_i - U_i)\chi_i\phi_i \Big]_{t_k^-}^{t_k^+}.
$$

From the error bound it is clear that for $\boldsymbol{v} = \boldsymbol{g}$, $\langle L(\boldsymbol{u} - \boldsymbol{U}), \boldsymbol{v} \rangle \neq \langle \boldsymbol{u} - \boldsymbol{U}, L^* \boldsymbol{v} \rangle$ because of the interior boundary terms. Hence L^* does not satisfy the definition of the

adjoint operator, equation (3.3). We could make the boundary conditions disappear by setting $U_i(t_k^+) = u_i(t_k^+)$ when $\chi_i(t_k^+) = 1$, and $\phi_i(t_k^-) = 0$ when $\chi_i(t_k^-) = 1$. However, applying these conditions would be against the philosophy of the method because the solution of the reduced model should depend only on the terms in the reduced model, not on terms that have been removed. It is therefore undesirable to make the boundary terms disappear.

3.2.2 Application to model problems

We now define the algorithm for generating reduced models with the quasi-steady state approximation. The algorithm has a similar structure to the initial-value approximation algorithm. The differences are that the adjoint solution is now obtained using equation (3.16), the expression for the residual is different, and the reduced model is now the quasi-steady state.

The quasi-steady state approximation algorithm is then defined as follows:
Initialization: Set $S_i = \emptyset$ for i from 1 to N, so that, in this case, $f_i(t, \boldsymbol{U}) = 0$ for all t, and for i from 1 to N. Compute $\boldsymbol{\phi}$ using equation (3.16).
Iteration:

1. Compute `err(1:N,1:M)` where

$$\texttt{err(i,k)} = \int_{t_{k-1}}^{t_k} \left| \chi_i(t) \frac{\mathrm{d}u_i}{\mathrm{d}t} + f_i(t, \boldsymbol{u}) \right| |\phi_i| \mathrm{d}t + \left[(u_i - U_i)\chi_i \phi_i \right]_{t_k^-}^{t_k^+}. \tag{3.21}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding time intervals to S_i for i from 1 to N.

3. Re-compute $\boldsymbol{U}$, $\boldsymbol{\phi}$.

Termination: Stop when

$$|J(\boldsymbol{u}) - J(\boldsymbol{U})| < TOL.$$

As before, the inputs to the algorithm are as follows: N_{cont}, the number of t-intervals added to $\boldsymbol{S}$ on each iteration of the algorithm; TOL, the desired accuracy in the quantity of interest; and, $\boldsymbol{g}$, the weighting function in the quantity of interest.

We apply the quasi-steady state approximation algorithm to the same two model problems that were analyzed in the previous section: the linear IVP (3.12) and the

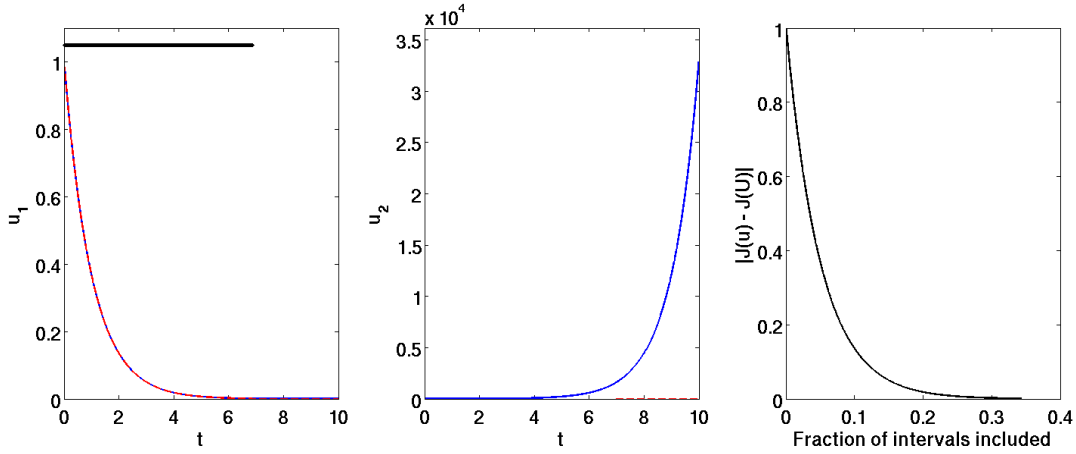
Michaelis-Menten equations (3.13). We consider the same quantities of interest as before, $J(\boldsymbol{u}) = \int_0^T u_1 \, \mathrm{d}t$, and $J(\boldsymbol{u}) = \int_0^T u_2 \, \mathrm{d}t$.

A comparison of Figures 3.1 and 3.3 demonstrates that, for the linear IVP, the regions of the domain where the reduced model is applied are very similar for the quasi-steady state and the initial-value approximations. When the average of u_1 is the quantity of interest, the choice of approximation for u_2 has no effect on the results because the variable u_2 has no effect on u_1. However u_1 does have an effect on u_2. For the initial-value approximation, u_1 is close to 0 in the region where the reduced model is applied. The quasi-steady state of u_1 is $u_1 = 0$. Therefore when the average of u_2 is the quantity of interest, the results are similar because the two different approximations give very similar solutions in the region of the domain where the reduced model is applied.

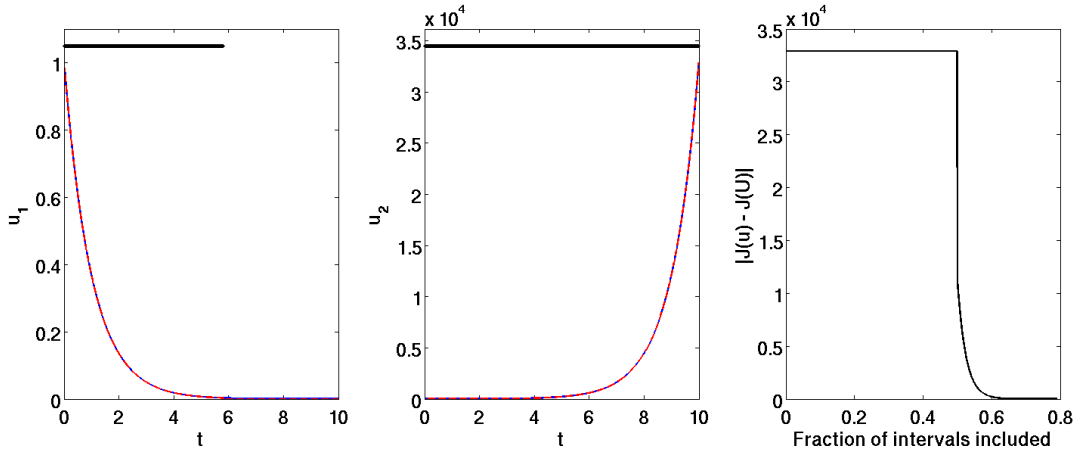
For the Michaelis-Menten equations, the results are quite different when the quasi-steady state approximation is used – Figure 3.4. The quasi-steady state approximation is a good approximation for u_2 for $t > 0.01$. For this reason the *a posteriori* method identifies that when $TOL = 10^{-3}$, u_2 can be approximated by its quasi-steady state for $t > 0.01$ for both quantities of interest – Figure 3.4.

3.3 Robustness of *a posteriori* model reduction to the N_{cont} parameter

The robustness of *a posteriori* model reduction to the algorithm parameter N_{cont} was tested by comparing the included regions of the domain for a range of N_{cont} values. N_{cont} is the number of t-intervals added to $\boldsymbol{S}$ on each iteration of the algorithm. (A region is included if the method has identified it as a region where the full model needs to be solved in order to accurately compute the quantity of interest.) Figures 3.5-3.8 show that both the initial-value and quasi-steady state forms of *a posteriori* model reduction are robust to the algorithm parameter N_{cont}. For the two model problems considered in this chapter, a 10-20 fold increase in efficiency may be obtained by increasing N_{cont} so that fewer iterations are required to obtain a reduced model that accurately reproduces the quantity of interest. The results here support the idea that in general the method is robust to the value of N_{cont}. This is important for more computationally expensive problems where running the algorithm with a small value of N_{cont} takes a prohibitively long time.

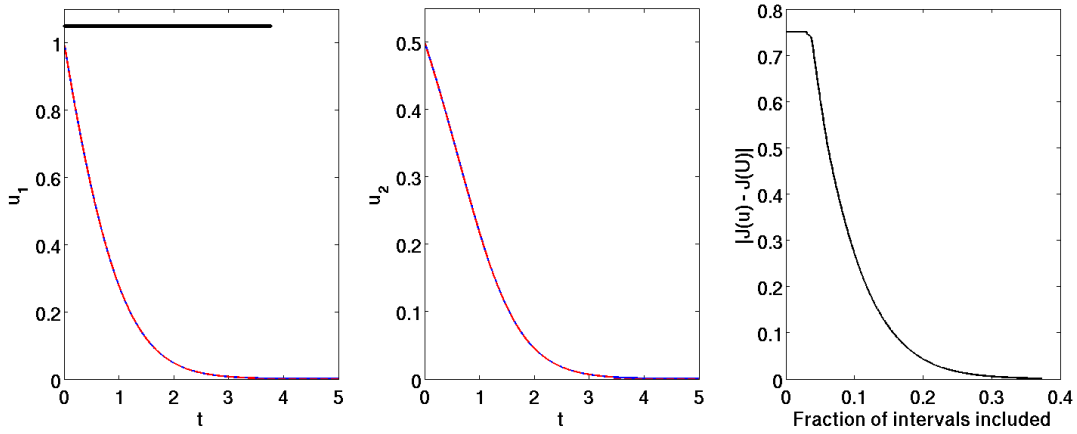


(a) $J(\boldsymbol{u}) = \int_0^{10} u_1 \, \mathrm{d}t$, $TOL = 10^{-3}$

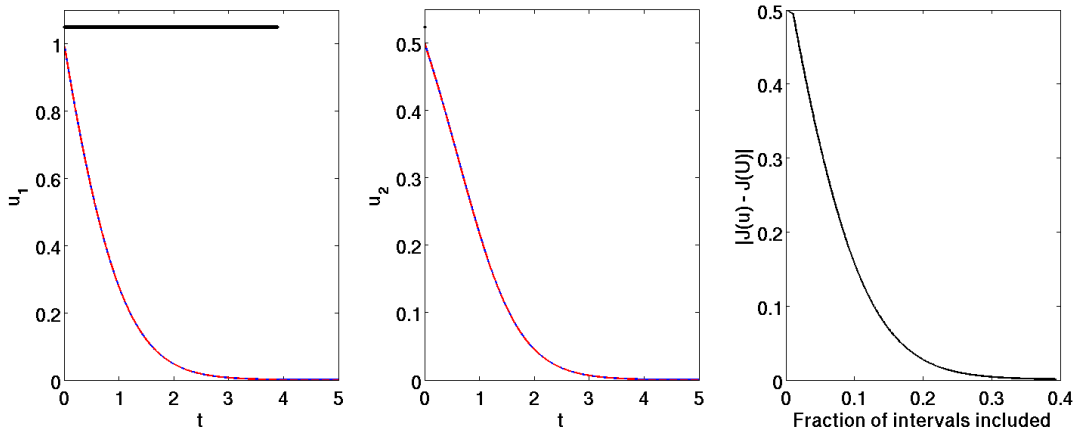


(b) $J(\boldsymbol{u}) = \int_0^{10} u_2 \, \mathrm{d}t$, $TOL = 10^{-1}$

Figure 3.3: Quasi-steady state approximation for linear IVP in automatically identified regions of the domain. *Left and middle figures*: solid blue line is the numerical solution to the linear IVP (3.12), obtained using the forwards Euler method with $\Delta t = 10^{-3}$. The initial conditions were $u_1(0) = u_2(0) = 1$. The dashed red line is $\boldsymbol{U}$, and the black lines above the curves represent the time intervals where the time derivative term is included in the reduced model. Here $N_{cont} = 5$. *Right figure* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$, as a function of the fraction of the domain included in the reduced model.



(a) $J(\boldsymbol{u}) = \int_0^5 u_1 \, \mathrm{d}t$



(b) $J(\boldsymbol{u}) = \int_0^5 u_2 \, \mathrm{d}t$

Figure 3.4: Quasi-steady state approximation for Michaelis-Menten equations in automatically identified regions of the domain. *Left and middle figures*: solid blue line is the numerical solution to the Michaelis-Menten equations (3.13), obtained using the forwards Euler method with $\Delta t = 10^{-3}$. The initial conditions were $u_1(0) = s(0) = 1$, $u_2(0) = c(0) = 0$. The dashed red line is $\boldsymbol{U}$, and the black lines above the curves represent the time intervals where the time derivative term is included in the reduced model. Here $\epsilon = 0.01$, $K = 1$, $\lambda = 2$, $N_{cont} = 5$, and $TOL = 10^{-3}$. *Right figure* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$, as a function of the fraction of the domain included in the reduced model.

There is a limit to how much N_{cont} can be increased without substantially changing the results. This limit will depend on how finely the error has been subdivided over the computational domain. For all the results in this thesis, the mesh used to obtain the numerical solution was used to subdivide the error, so that each element of the error array corresponds to an element of the mesh for a particular component of the solution. An important number therefore is the ratio between the total number of elements in the error array, NM, and N_{cont}, the number of t-intervals added to $\boldsymbol{S}$ on each iteration of the algorithm. This ratio is the maximum number of iterations required by the algorithm: $MaxIter = NM/N_{cont}$. On the basis of our numerical experiments, we recommend that $MaxIter$ should be at least 100.

3.4 Summary

In this chapter we have derived a novel variant of *a posteriori* model reduction that approximates reduced model variables by their quasi-steady state in automatically identified regions of the domain. The problem in applying the quasi-steady approximation in the original form of the method is that the residual is a function of $\boldsymbol{U}$ and when $\boldsymbol{U}$ is the quasi-steady state solution this residual is zero. We therefore re-formulated the adjoint problem so that the residual in the error bound is a function of $\boldsymbol{u}$, and is non-zero in regions of the domain where the reduced model is applied. For comparison, both forms of *a posteriori* model reduction were applied to two model problems, a linear IVP (3.12), and the Michaelis-Menten equations (3.13).

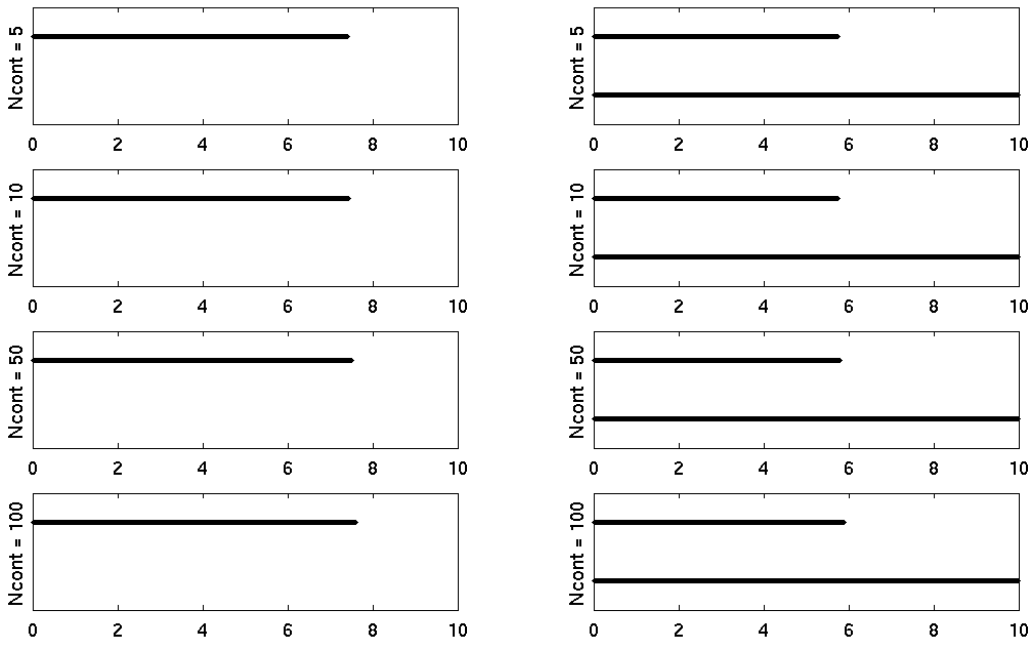
For the linear IVP, it was found that the two methods gave similar results. However, for the Michaelis-Menten equations the initial-value method and the quasi-steady state method gave different results – the quasi-steady state method automatically identified where a quasi-steady state approximation is suitable for accurately computing the quantity of interest.

Above we said that there are two reasons why a given approximation may be suitable. Either, the component has little or no effect on the quantity of interest, or the approximation is accurate in the region of the domain where the approximation is applied. By applying both methods to both problems we can identify more precisely which of the two reasons is true. For the linear BVP the results were similar for both approximations. For one of the quantities of interest, this was because the accuracy

of the two different approximations was very similar. For the other quantity of interest considered, it was because the component being approximated had no effect on the quantity of interest. For the Michaelis-Menten equations, the results were different with the two different methods. A region of the domain was identified where the quasi-steady state approximation can be applied, but the initial value approximation cannot be applied. This implies that in this region of the domain the quasi-steady state approximation is accurate.

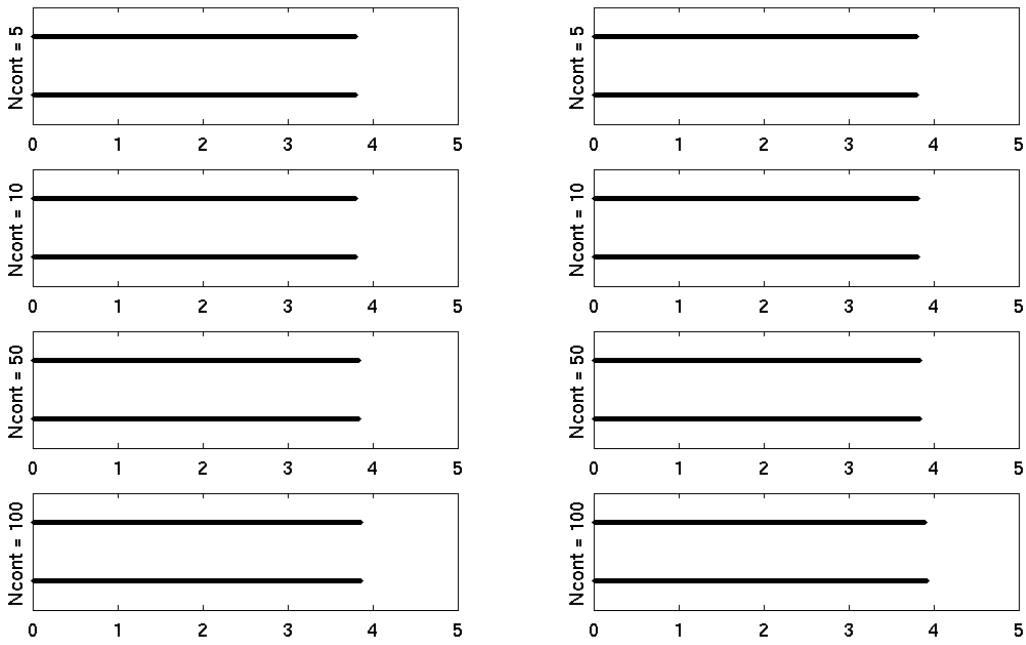
As the method is automatic and can be applied to ODEs with an arbitrary number of components, it could be used to identify variables that are near quasi-steady state in large systems of ODEs that are otherwise difficult to analyze with fast–slow analysis.

We also assessed the effect of varying the algorithm parameter N_{cont}, the number of t-intervals added to $\boldsymbol{S}$ on each iteration of the algorithm. It was shown that varying this parameter has a negligible effect on the results of the method. The method can be made computationally more efficient by increasing the value of N_{cont}, since fewer iterations are required for larger values of N_{cont}. For the model problems in this chapter no information is lost by increasing N_{cont} from 5 to 100. This discussion is further developed in the next chapter, where different types of problems are analyzed, and the effect of varying N_{cont} is assessed.



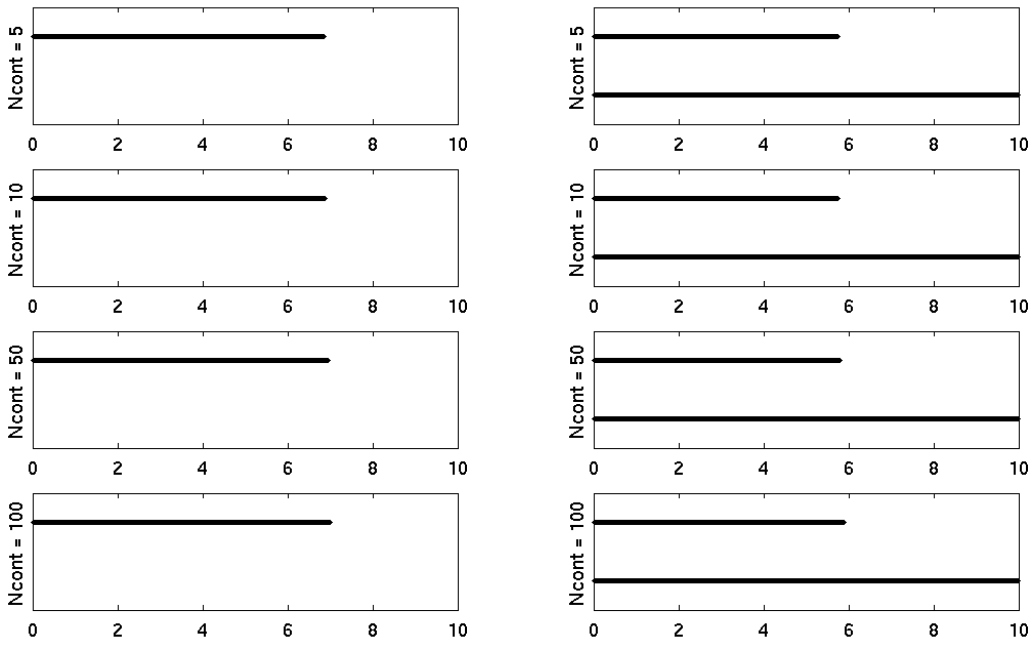
(a) $J(\boldsymbol{u}) = \int_0^{10} u_1 \, \mathrm{d}t$ **(b)** $J(\boldsymbol{u}) = \int_0^{10} u_2 \, \mathrm{d}t$

Figure 3.5: Linear IVP. *A posteriori* model reduction with initial-value approximation is robust to changes in N_{cont} parameter. As in Figures 3.1 and 3.3, the black lines represent the time intervals that are included in the reduced model. In each panel, the top line represents u_1, and the bottom line represents u_2. Apart from N_{cont}, all other parameter values are the same as in Figures 3.1 and 3.3.



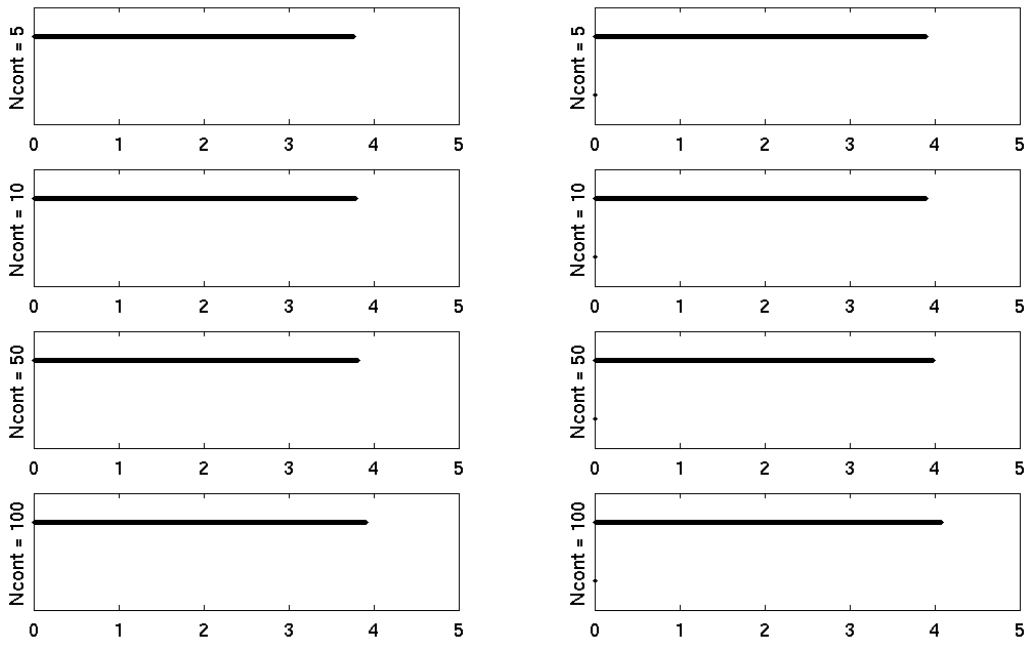
(a) $J(\boldsymbol{u}) = \int_0^5 u_1 \, \mathrm{d}t$ **(b)** $J(\boldsymbol{u}) = \int_0^5 u_2 \, \mathrm{d}t$

Figure 3.6: Michaelis-Menten equations. *A posteriori* model reduction with initial-value approximation is robust to changes in N_{cont} parameter. As in Figures 3.2 and 3.4, the black lines represent the time intervals that are included in the reduced model. In each panel, the top line represents u_1, and the bottom line represents u_2. Apart from N_{cont}, all other parameter values are the same as in Figures 3.2 and 3.4.



(a) $J(\boldsymbol{u}) = \int_0^{10} u_1 \, \mathrm{d}t$ **(b)** $J(\boldsymbol{u}) = \int_0^{10} u_2 \, \mathrm{d}t$

Figure 3.7: Linear IVP. *A posteriori* model reduction with quasi-steady state approximation is robust to changes in N_{cont} parameter. As in Figures 3.1 and 3.3, the black lines represent the time intervals that are included in the reduced model. In each panel, the top line represents u_1, and the bottom line represents u_2. Apart from N_{cont}, all other parameter values are the same as in Figures 3.1 and 3.3.



(a) $J(\boldsymbol{u}) = \int_0^5 u_1 \, \mathrm{d}t$ **(b)** $J(\boldsymbol{u}) = \int_0^5 u_2 \, \mathrm{d}t$

Figure 3.8: Michaelis-Menten equations. *A posteriori* model reduction with quasi-steady state approximation is robust to changes in N_{cont} parameter. As in Figures 3.2 and 3.4, the black lines represent the time intervals that are included in the reduced model. In each panel, the top line represents u_1, and the bottom line represents u_2. Apart from N_{cont}, all other parameter values are the same as in Figures 3.2 and 3.4.

4

Boundary-Value Differential Equations

In the literature review (Chapter 2) we discussed the most widely used methods for model analysis in systems biology, and evaluated whether these methods are suitable for answering the question "Which terms in the model equations have the greatest effect on a model output of interest?" We concluded that existing methods for answering this question, such as perturbation methods, require specialist mathematical skill to apply. Furthermore application of these methods becomes more difficult as the number of components and different types of term in the model increases. A further limitation of many existing methods is that they only give information about whether terms are negligible over the whole computational domain, and are unable to say, for example, whether a term is negligible over a given time interval in the domain. In contrast, the *a posteriori* model reduction methodology requires no more mathematical skill than is required to solve the model equations numerically. It can be applied to models with arbitrary number of components, and, as we will see below, equations with a wide range of different types of terms. And, it is more powerful than perturbation methods in that it automatically gives insight into the structure and behaviour of all model components in all regions of the computational domain.

In the previous chapter we applied *a posteriori* model reduction to systems of initial value ODEs. We reproduced the error bound derivation from [75], showing how the error in a user-defined quantity of interest can be expressed as the inner product of a residual with an adjoint solution. We also developed a novel variant of *a posteriori*

model reduction for automatically determining regions of the problem domain where a quasi-steady state approximation is suitable, given a user-defined quantity of interest. Both forms of the method were applied to two initial-value problems, and the results showed that the reduced model generated by the algorithm depends on the quantity of interest and on whether the initial-value or quasi-steady state approximation is applied in the reduced model.

In this chapter we consider Boundary-Value Problems (BVPs), with one dependent variable. The dependent variable can be thought of as space for the problems considered here. A similar methodology to that used in the previous chapter is used to express the error in a user-defined quantity of interest as the product of a residual with an adjoint solution. We start by considering a linear reaction-diffusion equation. For this problem we consider two types of approximation, which we call the diffusion approximation and the reaction approximation, based on whether the diffusion or the reaction term remains in the reduced model in regions of the domain where the original differential operator is not applied. As in the previous chapter the error bound derivation is different for different types of approximation. We then move on to consider general nonlinear reaction-diffusion equations, with the nonlinearity being either in the diffusion term or the reaction term, or in both terms. Equations of this type can be used to model gradient-dependent advection, for example in chemotaxis [27], and also viscosity, an effect considered in many fluid mechanics models. The error bound is then applied to reduce a single-component nonlinear reaction-diffusion equation and a coupled nonlinear reaction-diffusion equation that has been used to model chemotaxis.

Finally we assess the robustness of the results to the algorithm parameter N_{cont} (the number of elements included in the reduced model on each step of the iteration). We show that there are minor differences in the output of the algorithm when N_{cont} is varied. This means that increasing the efficiency of the algorithm by increasing N_{cont} causes a minor reduction in the information gained by applying the method.

4.1 The finite element method for reaction-diffusion equations

The method developed in this thesis requires the equations for the primal, reduced primal and adjoint problems to be solved. In special cases, one or more of these equations may have an analytic solution. However, many of the problems we consider

are nonlinear, and do not have a known analytic solution. We therefore require accurate numerical approximations for these problems. *A posteriori* model reduction does not require a particular numerical method to be used. We choose to use the finite element method, [62], because it is a general methodology that applies to all problems considered in this thesis, and in its simplest form it is relatively easy to implement. Computational efficiency is not a major consideration for the problems analyzed in this thesis. If it was a major consideration, other numerical methods may be more appropriate, depending on the problem.

The Finite Element Method gives an approximate solution of a differential equation as the solution of a system of algebraic equations. To illustrate how the system of algebraic equations is derived we work through the method for a simple example. Suppose we would like to find u such that,

$$-\nabla^2 u + c(x)u = f(x),$$

for $x \in \Omega$, where $c(x), f(x) \in L^2(\Omega)$. We apply the boundary conditions $u(x) = 0$ for $x \in \partial\Omega$. The weak formulation of the problem is to find $u \in H_0^1(\Omega)$ such that,

$$\int_\Omega \nabla u \, \nabla v \, \mathrm{d}x + \int_\Omega c(x) \, u \, v \, \mathrm{d}x = \int_\Omega f(x) \, v \, \mathrm{d}x,$$

for all $v \in H_0^1(\Omega)$, where $H_0^1(\Omega)$ denotes the space of functions that are zero on the boundary of the domain, $\partial\Omega$, and whose first derivatives are in $L^2(\Omega)$.

The weak formulation is useful for proving that a unique solution exists, in this case by the Lax-Milgram theorem [70]. However the weak formulation is not directly useful for computing a solution because $H_0^1(\Omega)$ is an infinite-dimensional space. The finite element method approximates the weak formulation by restricting u and v to a finite-dimensional subspace of $H_0^1(\Omega)$, termed V_h. The finite element method is to find $u_h \in V_h$ such that,

$$\int_\Omega \nabla u_h \, \nabla v \, \mathrm{d}x + \int_\Omega c(x) \, u_h \, v \, \mathrm{d}x = \int_\Omega f(x) \, v \, \mathrm{d}x,$$

for all $v \in V_h$. In this chapter we consider problems on a 1-D domain, that is partitioned by $a = x_0 < x_1 < \ldots < x_n = b$, where $x_i = x_{i-1} + h$, $h = (b-a)/n$, and n is a freely chosen integer. We choose the basis functions of our finite element space to be,

$$\phi_k(x) = \begin{cases} 1 - \frac{|x-x_k|}{h} & \text{if } x \in [x_{k-1}, x_{k+1}], \\ 0 & \text{otherwise,} \end{cases}$$

for $k = 1, \ldots, n-1$. These basis functions are generally termed piecewise linear, or hat functions, and through the finite element method are used to construct a piecewise linear approximation to the solution of the original differential equation. If the boundary conditions were not zero Dirichlet, then we would also need an extra basis function at each boundary of the domain.

The finite element method then becomes, find $U_1, \ldots, U_N$ such that

$$\sum_{j=1}^{N} \int_\Omega \nabla U_j \phi_j \, \nabla \phi_i \, \mathrm{d}x + \sum_{j=1}^{N} \int_\Omega c(x) \, U_j \phi_j \, \phi_i \, \mathrm{d}x = \int_\Omega f(x) \, \phi_i \, \mathrm{d}x,$$

for $i = 1, \ldots, N$. This is a system of N equations for N unknowns. In this case the system is linear and the error of interpolation into V_h from $H_0^1(\Omega)$ is $O(h^2)$, [38]. For nonlinear differential equations the system is nonlinear, and we have evaluated accuracy in a heuristic manner by evaluating and inspecting the finite element solution for several different values of h.

More details on the finite element method can be found in a number of standard textbooks, for example [13, 62]. A reference with good practical advice on implementing the finite element method for nonlinear problems is [63].

When implementing the Finite Element Method (FEM) within the *a posteriori* model reduction algorithm we recommend looping over elements in the domain when calculating the finite element integrals. In any case this is advisable for complex problems, but in the context of *a posteriori* model reduction an additional advantage is gained because looping over elements in the domain makes it straightforward to apply one differential operator to a user-defined set of elements and another differential operator to the other elements.

4.2 Linear reaction-diffusion equation

4.2.1 Derivation of error bound for diffusion approximation

We begin by deriving an *a posteriori* error bound for a linear elliptic ODE BVP. The error bound forms the basis for a method to iteratively generate a reduced model. As in the previous chapter we are interested in accurately computing some quantity of

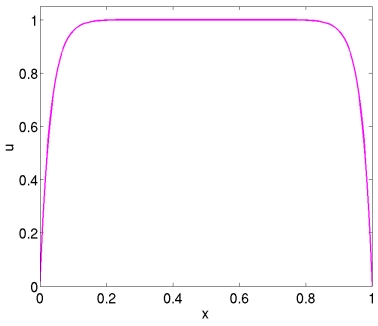


Figure 4.1: Solution to $-\epsilon u'' + u = 1, u(0) = u(1) = 0$, with $\epsilon = 10^{-3}$.

interest of the model solution. It will be seen below that the results of the *a posteriori* method depend heavily on the quantity of interest chosen.

The derivation is a modification of the error-bound derivation for initial-value ODEs, found in [75] and presented in Chapter 3. Deriving an error bound for the BVP requires modification of the adjoint problem and modification of the analysis underpinning the calculation of $|J(u) - J(U)|$.

Here we define an example primal problem to be

$$-\epsilon u'' + u = 1, \tag{4.1}$$

with u defined on the interval $[0, 1]$ and $0 < \epsilon \ll 1$. We apply the boundary conditions, $u(0) = u(1) = 0$. This problem has the feature that the solution changes rapidly near $x = 0$ and $x = 1$, but is relatively flat in other regions of the domain – see Figure 4.1. The analysis below can be extended to general BVPs, but for clarity and ease we delay consideration of more general cases to later sections.

We are interested in some functional of the solution, which we denote by $J(u)$. By the Riesz representation theorem the functional can be written as $J(v) = \langle v, g \rangle$ where $v, g \in L^2([0, 1])$, see for example [65]. The primal problem is self-adjoint. An adjoint problem is therefore,

$$-\epsilon \phi'' + \phi = g, \tag{4.2}$$

with ϕ defined on the interval $[0, 1]$. We apply the boundary conditions, $\phi(0) = 0$, $\phi(1) = 0$.

Suppose that U is an approximation of u. If $|J(u) - J(U)|$ is small, then U is a good approximation to u, at least for the purpose of calculating the quantity of interest. We

require U to satisfy the following conditions, which are required for deriving the error bound below:

1. $U(x)$ and $U'(x)$ are continuous for $0 < x < 1$;
2. $U(x)$ satisfies the boundary conditions of the primal problem, i.e., $U(0) = U(1) = 0$.

Now we use the solution of the adjoint problem to derive a bound on the error in $J(\cdot)$ introduced by approximating u by U.

$$\begin{aligned} J(u) - J(U) &= \int_0^1 (u - U)\ g\ \mathrm{d}x \\ &= \int_0^1 (u - U)(-\epsilon\phi'' + \phi)\ \mathrm{d}x \\ &= -\int_0^1 (u' - U')(-\epsilon\phi') + (u - U)\phi\ \mathrm{d}x + \left[(u - U)(-\epsilon\phi')\right]_0^1, \end{aligned}$$

using integration by parts. This uses the assumptions that u, U, ϕ' are continuous on $0 < x < 1$. These assumptions follow directly from the definitions above. The boundary term is zero as $u - U = 0$ at $x = 0$ and $x = 1$. We now apply integration by parts again to obtain,

$$\begin{aligned} J(u) - J(U) &= \int_0^1 (u'' - U'')(-\epsilon\phi) + (u - U)\phi\ \mathrm{d}x - \left[(u' - U')(-\epsilon\phi)\right]_0^1 \\ &= \int_0^1 (1 + \epsilon U'' - U)\phi\ \mathrm{d}x. \end{aligned} \tag{4.3}$$

This uses the assumptions that u', U', ϕ are continuous on $0 < x < 1$. Again these assumptions follow directly from the definitions above. All terms involving u disappear by definition of the primal problem, (4.1). The boundary term is zero as $\phi = 0$ at $x = 0$ and $x = 1$.

Therefore we may write our error bound as,

$$|J(u) - J(U)| \leq \sum_{j=1}^{N} \int_{x_{j-1}}^{x_j} |1 + \epsilon U'' - U||\phi|\ \mathrm{d}x, \tag{4.4}$$

where $a = x_0 < x_1 \ldots < x_N = b$ is a subdivision of the computational domain.

The reason for splitting up the integral into a sum of smaller integrals is so that we can identify the relative contribution of each (x_{j-1}, x_j) interval to $|J(u) - J(U)|$.

Equation (4.4) is convenient because it expresses $|J(u) - J(U)|$ as a sum of easily computable local errors. This is useful for determining the region of the domain where an accurate representation of the original model is required to accurately calculate the quantity of interest.

4.2.2 Application to a model problem

The algorithm we now present generates a reduced model of the form

$$\begin{aligned} -\epsilon U'' + U &= 1 \quad \text{for} \quad x \in S \\ -\epsilon U'' &= 0 \quad \text{otherwise,} \end{aligned} \tag{4.5}$$

where S is a union of one or more intervals in the x-domain that is determined by the algorithm, and U is defined on the interval $[0, 1]$. We apply the boundary conditions $U(0) = U(1) = 0$. To fully determine the problem we require that $U(x)$ is continuously differentiable on the interval $[0, 1]$. Requiring $U(x)$ to be continuously differentiable means that the error bound in equation (4.4) can be applied. One way of describing the reduced model is to say that the reaction and diffusion terms are present in the region S, whereas only the diffusion term is present otherwise.

For the results below we choose the quantity of interest firstly to be,

$$J(u) = \int_0^1 u \, \mathrm{d}x,$$

which may be interpreted as the average of u over $[0, 1]$. Then we consider,

$$J(u) = u(0.5) = \int_0^1 u \; \delta(x - 0.5) \, \mathrm{d}x,$$

where $\delta(\cdot)$ is the Dirac delta function. The Dirac delta function can be thought of as the limit of a zero-centred normal distribution as the standard deviation tends to zero.

The algorithm is then defined as follows:

Initialization: Set $S = \emptyset$, so that, in this case, $-\epsilon U'' = 0$ for all x. Compute ϕ using equation (4.2).

Iteration:

1. Compute `err(1:N)` where

$$\texttt{err(j)} = \int_{x_{j-1}}^{x_j} |1 + \epsilon U'' - U||\phi| \, \mathrm{d}x. \tag{4.6}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding space intervals to S.

3. Re-compute U.

Termination: Stop when

$$|J(u) - J(U)| < TOL.$$

As in the previous chapter, the inputs to the algorithm are as follows: N_{cont}, the number of x-intervals added to S on each iteration of the algorithm; TOL, the desired accuracy in the quantity of interest; and, g, the weighting function in the quantity of interest.

The finite element method with linear basis functions was used to solve both the primal and adjoint problems. The reduced models generated by the algorithm are shown in Figure 4.2. As expected, when $J(u) = u(0.5)$, the reaction term is present in a smaller region of the reduced model than when $J(u) = \int_0^1 u \, \mathrm{d}x$.

4.2.3 Derivation of error bound for reaction approximation

In the previous section the problem of interest was a linear reaction-diffusion equation. The form of reduced model was the diffusion approximation: for $x \notin S$ a linear diffusion equation was solved, and for $x \in S$ the original reaction-diffusion equation was solved. More precisely, the problem of interest was of the form $Lu = 0$, where $L = L_{\text{diffusion}} + L_{\text{reaction}}$, and the form of the reduced model was as follows: solve $L_{\text{diffusion}}u = 0$ for $x \notin S$, and solve $(L_{\text{diffusion}} + L_{\text{reaction}})u = 0$ for $x \in S$.

In this section we adapt the method developed in the previous section so that a reaction approximation can be applied. The new algorithm will construct a reduced model such that for $x \notin S$ an algebraic reaction equation is solved, and for $x \in S$ the original reaction-diffusion equation is solved. More precisely, this can be written as follows: solve $L_{\text{reaction}}u = 0$ for $x \notin S$, and solve $(L_{\text{diffusion}} + L_{\text{reaction}})u = 0$ for $x \in S$.

As in the previous section we are interested in some user-defined functional of the solution, and the region S is determined by the algorithm so that the user-defined functional can be accurately computed from U.

The simplest approach for applying the reaction approximation would be to use the method as presented in the previous section, replacing the diffusion approximation with the reaction approximation. The algorithm would then start by solving the algebraic

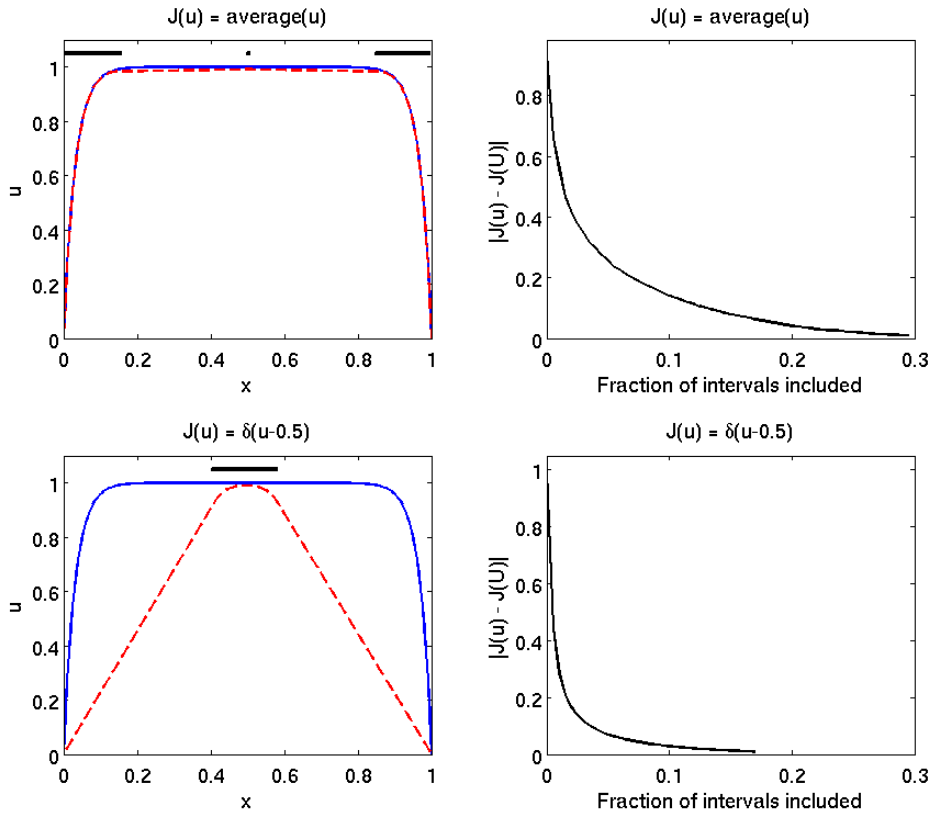


Figure 4.2: Diffusion approximation for linear reaction-diffusion model problem. Different quantities of interest result in different reduced models. *Left panels*: solid blue line is the finite element solution to $-\epsilon u'' + u = 1$ with 200 elements, dashed red line is solution to the reduced model, equation (4.5), and the black lines above the curves represent the space intervals where the reaction term is included in the reduced model. Here $\epsilon = 10^{-3}$, $N_{cont} = 1$ and $TOL = 10^{-3}$. *Right panels*: $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

reaction equation over the whole domain, and iteratively include the diffusion term where it is needed. However this approach will fail if the residual function from the previous section is used, for analogous reasons to those discussed in section 3.2. The reaction approximation solution satisfies the original problem (without the boundary conditions), hence the residual of the reaction approximation is zero everywhere.

We therefore take a similar approach to section 3.2. The adjoint problem is re-formulated, so that the error is expressed as the product of a 'pseudo-adjoint' solution with a residual which is a function of u, plus some extra terms that arise from discontinuities in the first derivatives of the reduced primal and adjoint problems. The error is used in an algorithm for model reduction which is applied to the same linear reaction-diffusion equation that was analyzed in the previous section. Regions of the domain where the reaction approximation results in a relatively small change in $J(\cdot)$ are automatically identified.

As in the previous chapter, when integration by parts is applied to the re-formulated adjoint problem, not all of the boundary terms disappear. Therefore the re-formulated adjoint problem does not satisfy the definition of the adjoint given in Chapter 3. It is more accurate to say that the re-formulated adjoint problem is approximately adjoint. However we still refer to it as the adjoint problem.

As before, the primal problem is the following linear reaction-diffusion equation,

$$-\epsilon u'' + u = 1, \tag{4.7}$$

with u defined on the interval $[0, 1]$, $0 < \epsilon \ll 1$, and boundary conditions $u(0) = u(1) = 0$.

In contrast to the method presented in the previous section, we have to define the form of the reduced primal problem in order to derive the error bound. Therefore we define the reduced primal problem to be,

$$-\epsilon\chi(x)U'' + U = 1, \tag{4.8}$$

where

$$\chi(x) = \begin{cases} 1 & \text{for } x \in S \\ 0 & \text{otherwise,} \end{cases}$$

where S is a union of one or more intervals in the x-domain, to be determined by the algorithm. This form of reduced model corresponds to solving $U = 1$ for $x \notin S$, and the original reaction diffusion equation for $x \in S$. As in section 3.2, we must take account of χ when setting the boundary conditions of the reduced primal problem:

$$U(0) = \begin{cases} 0 & \text{if } 0 \in S \\ 1 & \text{otherwise,} \end{cases} \qquad U(1) = \begin{cases} 0 & \text{if } 1 \in S \\ 1 & \text{otherwise.} \end{cases}$$

We now re-define the adjoint problem. As in section 3.2, this adjoint problem is adjoint to the reduced model. The new adjoint problem is to find ϕ satisfying,

$$-\epsilon\chi(x)\phi'' + \phi = g, \tag{4.9}$$

with ϕ defined on the interval $[0, 1]$. The boundary conditions are set in a similar way to the boundary conditions of the reduced model:

$$\phi(0) = \begin{cases} 0 & \text{if } 0 \in S \\ g(0) & \text{otherwise,} \end{cases} \qquad \phi(1) = \begin{cases} 0 & \text{if } 1 \in S \\ g(1) & \text{otherwise.} \end{cases}$$

Now we use the solution of the adjoint problem to derive a bound on the error in $J(\cdot)$ introduced by approximating u by U.

$$\begin{aligned} J(u) - J(U) &= \int_0^1 (u - U)g \, \mathrm{d}x \\ &= \int_0^1 (u - U)(-\epsilon\chi\phi'' + \phi) \, \mathrm{d}x \\ &= -\int_0^1 (u' - U')(-\epsilon\chi\phi') + (u - U)\phi \, \mathrm{d}x + \left[(u - U)(-\epsilon\chi\phi')\right]_0^1 \\ &\qquad + \sum_{j=1}^{M} \left[(u - U)(-\epsilon\chi\phi')\right]_{x_j^-}^{x_j^+}, \end{aligned} \tag{4.10}$$

using integration by parts. The boundary term $[(u - U)(-\epsilon\chi\phi')]_{x_j^-}^{x_j^+}$ is the jump in $(u(x_j) - U(x_j))(-\epsilon\chi(x_j)\phi'(x_j))$ at the points, $\{x_j\}$ where $\chi(x_j)$ and $\phi'(x_j)$ are discontinuous. The notation x_j^+ denotes the limit as x tends to x_j from above, and x_j^- denotes the limit as x tends to x_j from below. These discontinuities arise on the boundary of S, where there is a transition from solving the 2nd order BVP to solving the algebraic reaction term. The boundary terms at $x = 0$ and $x = 1$ disappear because either

$u(0) - U(0) = 0$, or $\chi(0) = 0$ at $x = 0$, and either $u(1) - U(1) = 0$, or $\chi(1) = 0$ at $x = 1$. Now we apply integration by parts again to obtain,

$$\begin{aligned} J(u) - J(U) &= \int_0^1 (u'' - U'')(-\epsilon\chi\phi) + (u - U)\phi \, \mathrm{d}x + \left[(u' - U')(-\epsilon\chi\phi)\right]_0^1 \\ &\quad + \sum_{j=1}^{M} \left[(u - U)(-\epsilon\chi\phi')\right]_{x_j^-}^{x_j^+} + \sum_{j=1}^{M} \left[(u' - U')(-\epsilon\chi\phi)\right]_{x_j^-}^{x_j^+} \\ &= \int_0^1 (-\epsilon\chi u'' + u - 1)\phi \, \mathrm{d}x + \sum_{j=1}^{M} \left[(u - U)(-\epsilon\chi\phi')\right]_{x_j^-}^{x_j^+} \\ &\quad + \sum_{j=1}^{M} \left[(u' - U')(-\epsilon\chi\phi)\right]_{x_j^-}^{x_j^+}. \end{aligned} \tag{4.11}$$

Again, a jump term arises on the boundary of S. The boundary terms at $x = 0$ and $x = 1$ disappear because either $\phi(0) = 0$, or $\chi(0) = 0$ at $x = 0$, and either $\phi(1) = 0$, or $\chi(1) = 0$ at $x = 1$. The integral terms involving U disappear by definition of the reduced primal problem, (4.8).

We may therefore write our error bound as,

$$\begin{aligned} |J(u) - J(U)| \leq \sum_{j=1}^{M} \int_{x_{j-1}}^{x_j} \left|-\epsilon\chi u'' + u - 1\right| |\phi| \, \mathrm{d}x + \left[(u - U)(-\epsilon\chi\phi')\right]_{x_j^-}^{x_j^+} \\ + \left[(u' - U')(-\epsilon\chi\phi)\right]_{x_j^-}^{x_j^+}, \end{aligned} \tag{4.12}$$

where $a = x_0 < x_1 \ldots < x_N = b$ is a subdivision of the computational domain.

From the error bound it is clear that for $v = g$, $\langle L(u-U), v\rangle \neq \langle u-U, L^*v\rangle$ because of the interior boundary terms. Hence L^* does not satisfy the definition of the adjoint operator, equation (3.3). We could make the boundary conditions disappear by setting,

$$\begin{aligned} &u(x_j^-) - U(x_j^-) = 0, \ \phi(x_j^-) = 0, \quad \text{if } \chi(x_j^-) = 1, \\ &u(x_j^+) - U(x_j^+) = 0, \ \phi(x_j^+) = 0, \quad \text{if } \chi(x_j^+) = 1, \end{aligned} \tag{4.13}$$

for j from 1 to M. This would ensure that both $(u - U)(-\epsilon\chi\phi')$ and $(u' - U')(-\epsilon\chi\phi)$ would be zero at x_j^- and x_j^+ for each boundary term. However, as in the previous chapter, applying these conditions would be against the philosophy of the method because the solution of the reduced model should depend only on the terms in the reduced model, not on terms that have been removed. It is therefore undesirable to make the boundary terms disappear.

4.2.4 Application to model problem

We now define the algorithm for generating reduced models with the reaction approximation. The algorithm has a similar structure to the diffusion approximation. The differences are that the adjoint solution is now obtained using equation (4.9), the expression for the residual is different, and the reduced model is now the solution to the algebraic reaction equation.

The reaction approximation algorithm is then defined as follows:

Initialization: Set $S = \emptyset$, so that, in this case, $U = 1$ for all x. Compute ϕ using equation (4.9).

Iteration:

1. Compute `err(1:N)` where

$$\texttt{err(j)} = \int_{x_{j-1}}^{x_j} \left|-\epsilon\chi u'' + u - 1\right| |\phi| \ \mathrm{d}x + \left[(u-U)(-\epsilon\chi\phi')\right]_{x_j^-}^{x_j^+} + \left[(u'-U')(-\epsilon\chi\phi)\right]_{x_j^-}^{x_j^+}. \tag{4.14}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding space intervals to S.

3. Re-compute U, ϕ.

Termination: Stop when

$$|J(u) - J(U)| < TOL.$$

The algorithm was implemented for the same quantities of interest as in the previous section, $J(u) = \int_0^1 u \ \mathrm{d}x$, and $J(u) = u(0.5) = \int_0^1 u \cdot \delta(x - 0.5) \ \mathrm{d}x$. The reduced models generated by the algorithm are shown in Figure 4.3. From looking at the solution to the original equation it is clear that $U = 1$ is a good approximation away from the boundary. The output of the algorithm confirms this: when $J(u) = \int_0^1 u \ \mathrm{d}x$ and $TOL = 10^{-3}$ the resulting reduced model is the original reaction-diffusion equation near the boundary, and the reaction approximation away the from the boundary. When $J(u) = u(0.5) = \int_0^1 u \cdot \delta(x - 0.5) \ \mathrm{d}x$ and $TOL = 10^{-3}$ the resulting reduced model is $U = 1$ everywhere since $u(0.5) - 1 < 10^{-3}$.

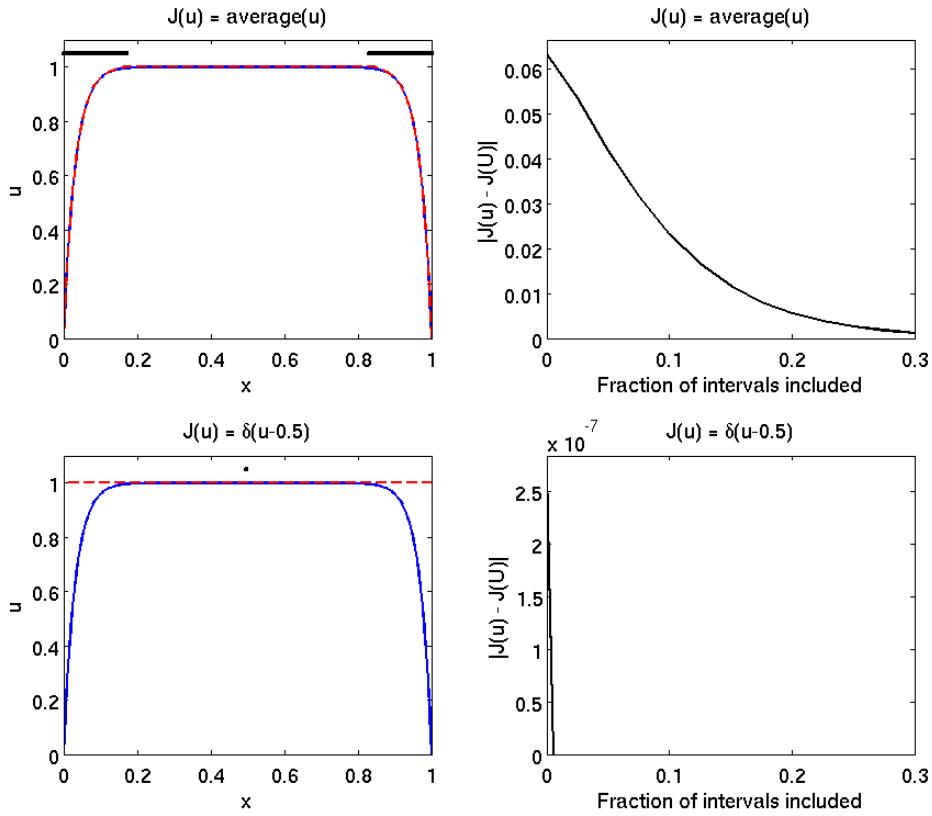


Figure 4.3: Reaction approximation for linear reaction-diffusion model problem. Different quantities of interest result in different reduced models. *Left panels*: solid blue line is the finite element solution to $-\epsilon u'' + u = 1$ with 200 elements, dashed red line is solution to the reduced model, equation (4.8), and the black lines above the curves represent the space intervals where the diffusion term is included in the reduced model. Here $\epsilon = 10^{-3}$, $N_{cont} = 1$ and $TOL = 10^{-3}$. *Right panels*: $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

4.3 Nonlinear reaction-diffusion equations

4.3.1 Derivation of error bound

In the previous sections we analysed a second-order linear differential equation. We saw that integration by parts could be used to derive an error bound for a user-defined quantity of interest. The error bound was expressed as the product of a residual with the solution to an adjoint problem. In this section, we derive an error bound in a more general setting – for coupled nonlinear systems of BVP ODEs. We will see that a combination of linearisation and integration by parts can be used to derive a similar error bound to the one derived in section 4.2.1.

Let the general form of the primal problem be defined as,

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(h(\boldsymbol{u},x)\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}x}\right)+\boldsymbol{f}(\boldsymbol{u},x)=\boldsymbol{0},\quad \text{for } a<x<b, \tag{4.15}$$

where $h(\boldsymbol{u},x)$ is a general $N\times N$ matrix. We apply the boundary conditions $\boldsymbol{u}(a)=\boldsymbol{u}_a$ and $h(\boldsymbol{u},x)\dfrac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}x}(b)=\boldsymbol{n}_b$, where $\boldsymbol{n}_b$ is a vector. Choosing one Neumann and one Dirichlet boundary condition illustrates the minor differences in the analysis between Dirichlet and Neumann boundary conditions. Using the methodology presented here it is straight-forward to modify the error bound for a problem with two Dirichlet conditions or two Neumann conditions.

Define a quantity of interest,

$$J(\boldsymbol{u})=\int_a^b \boldsymbol{u}\cdot\boldsymbol{g}\ \mathrm{d}x.$$

Suppose that we have a solution to a reduced primal problem, $\boldsymbol{U}(x)$. We require that:

1. $\boldsymbol{U}(x)$ and $h(\boldsymbol{u},x)\dfrac{\mathrm{d}\boldsymbol{U}(x)}{\mathrm{d}x}$ are continuous for $0<x<1$;
2. $\boldsymbol{U}(x)$ satisfies the boundary conditions of the primal problem, i.e., $\boldsymbol{U}(a)=\boldsymbol{u}(a)$ and $h(\boldsymbol{U},x)\dfrac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}x}(b)=h(\boldsymbol{u},x)\dfrac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}x}(b)$.

Then the adjoint problem is,

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(A\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}x}\right)-A'\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}x}+F\boldsymbol{\phi}=\boldsymbol{g}, \tag{4.16}$$

where the matrices A, A', and F have entries given by,

$$\begin{aligned} A_{ij}(x) &= \int_0^1 h_{ji}(\boldsymbol{\alpha}(s,x))\ \mathrm{d}s \\ A'_{ij}(x) &= \int_0^1 \sum_{k=1}^N \frac{\partial h_{jk}}{\partial u_i}(\boldsymbol{\alpha}(s,x))\frac{\partial \alpha_k}{\partial x}\ \mathrm{d}s \\ F_{ij}(x) &= \int_0^1 \frac{\partial f_j}{\partial u_i}(\boldsymbol{\alpha}(s,x))\ \mathrm{d}s, \end{aligned}$$

where $\boldsymbol{\alpha}(s,x) = s\boldsymbol{u}(x) + (1-s)\boldsymbol{U}(x)$. We apply the boundary conditions $\boldsymbol{\phi}(a) = \mathbf{0}, \dfrac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}x}(b) = \mathbf{0}$.

We now derive the following property for the coefficients $A(x)$ and $A'(x)$ which will be needed when deriving the *a posteriori* error bound below,

$$\begin{aligned} &\sum_{i=1}^N A_{ij}(x)\left(\frac{\mathrm{d}u_i}{\mathrm{d}x} - \frac{\mathrm{d}U_i}{\mathrm{d}x}\right) + \sum_{i=1}^N A'_{ij}(x)(u_i - U_i) \\ &= \sum_{i=1}^N \left(\int_0^1 h_{ji}(\boldsymbol{\alpha}(s,x))\frac{\partial^2 \alpha_i}{\partial x \partial s}\ \mathrm{d}s + \int_0^1 \sum_{k=1}^N \frac{\partial h_{jk}}{\partial u_i}(\boldsymbol{\alpha}(s,x))\frac{\partial \alpha_k}{\partial x}\frac{\partial \alpha_i}{\partial s}\ \mathrm{d}s \right) \\ &= \int_0^1 \sum_{k=1}^N h_{jk}\frac{\partial^2 \alpha_k}{\partial x \partial s} + \sum_{k=1}^N \sum_{i=1}^N \frac{\partial h_{jk}}{\partial \alpha_i}\frac{\partial \alpha_i}{\partial s}\frac{\partial \alpha_k}{\partial x}\ \mathrm{d}s \\ &= \int_0^1 \frac{\partial}{\partial s}\left(\sum_{k=1}^N h_{jk}\frac{\partial \alpha_k}{\partial x} \right) \mathrm{d}s \\ &= \sum_{k=1}^N h_{jk}(\boldsymbol{u})\frac{\mathrm{d}u_k}{\mathrm{d}x} - \sum_{k=1}^N h_{jk}(\boldsymbol{U})\frac{\mathrm{d}U_k}{\mathrm{d}x}. \end{aligned} \tag{4.17}$$

The following property holds for the coefficient $F(x)$,

$$\begin{aligned} \sum_{i=1}^N F_{ij}(x)(u_i(x) - U_i(x)) &= \sum_{i=1}^N \int_0^1 \frac{\partial f_j}{\partial u_i}(\boldsymbol{\alpha}(s,x))(u_i - U_i)\ \mathrm{d}s \\ &= \sum_{i=1}^N \int_0^1 \frac{\partial f_j}{\partial \alpha_i}\frac{\partial \alpha_i}{\partial s}\ \mathrm{d}s \\ &= \int_0^1 \frac{\mathrm{d}f_j}{\mathrm{d}s}\ \mathrm{d}s. \\ &= f_j(\boldsymbol{u},x) - f_j(\boldsymbol{U},x). \end{aligned} \tag{4.18}$$

Now we can express the error in the quantity of interest introduced by the approximation $\boldsymbol{U}$ to $\boldsymbol{u}$ as,

$$\begin{aligned} J(\boldsymbol{u}) - J(\boldsymbol{U}) &= \int_a^b (\boldsymbol{u} - \boldsymbol{U}) \cdot \boldsymbol{g} \, \mathrm{d}x \\ &= \int_a^b (\boldsymbol{u} - \boldsymbol{U}) \cdot \left(\frac{\mathrm{d}}{\mathrm{d}x} \left(A \frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}x} \right) - A' \frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}x} \right) + \int_a^b (\boldsymbol{u} - \boldsymbol{U}) \cdot F\boldsymbol{\phi} \, \mathrm{d}x \\ &= I + II. \end{aligned}$$

We first analyze term I,

$$\begin{aligned} I &= \int_a^b \sum_{i=1}^N (u_i - U_i) \cdot \left(\frac{\mathrm{d}}{\mathrm{d}x} \left(\sum_{j=1}^N A_{ij}(x) \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \right) - \sum_{j=1}^N A'_{ij}(x) \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \right) \mathrm{d}x \\ &= - \int_a^b \sum_{i=1}^N \left(\frac{\mathrm{d}u_i}{\mathrm{d}x} - \frac{\mathrm{d}U_i}{\mathrm{d}x} \right) \cdot \left(\sum_{j=1}^N A_{ij}(x) \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \right) + \sum_{i=1}^N (u_i - U_i) \cdot \left(\sum_{j=1}^N A'_{ij}(x) \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \right) \mathrm{d}x \\ &\qquad + \sum_{i=1}^N \left[(u_i - U_i) \cdot \left(\sum_{j=1}^N A_{ij}(x) \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \right) \right]_a^b, \end{aligned}$$

using integration by parts. This uses the assumptions that $\boldsymbol{u}$, $\boldsymbol{U}$, and $\mathrm{d}\boldsymbol{\phi}/\mathrm{d}x$ are continuous. These assumptions follow from the primal and adjoint problem definitions above. Since $\boldsymbol{U}(a) = \boldsymbol{u}(a)$ and $\dfrac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}x}(b) = \mathbf{0}$, the boundary term disappears. Therefore,

$$\begin{aligned} I &= - \int_a^b \sum_{i=1}^N \left(\frac{\mathrm{d}u_i}{\mathrm{d}x} - \frac{\mathrm{d}U_i}{\mathrm{d}x} \right) \cdot \left(\sum_{j=1}^N A_{ij}(x) \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \right) + \sum_{i=1}^N (u_i - U_i) \cdot \left(\sum_{j=1}^N A'_{ij}(x) \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \right) \mathrm{d}x \\ &= - \int_a^b \sum_{j=1}^N \left(\sum_{i=1}^N A_{ij}(x) \left(\frac{\mathrm{d}u_i}{\mathrm{d}x} - \frac{\mathrm{d}U_i}{\mathrm{d}x} \right) + \sum_{i=1}^N A'_{ij}(x)(u_i - U_i) \right) \cdot \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \, \mathrm{d}x \\ &= - \int_a^b \sum_{j=1}^N \left(\sum_{k=1}^N h_{jk}(\boldsymbol{u}) \frac{\partial u_k}{\partial x} - h_{jk}(\boldsymbol{U}) \frac{\partial U_k}{\partial x} \right) \cdot \frac{\mathrm{d}\phi_j}{\mathrm{d}x} \, \mathrm{d}x \qquad \text{[by (4.17)]} \\ &= \int_a^b \sum_{j=1}^N \left(\frac{\mathrm{d}}{\mathrm{d}x} \left(\sum_{k=1}^N h_{jk}(\boldsymbol{u}) \frac{\mathrm{d}u_k}{\mathrm{d}x} \right) - \frac{\mathrm{d}}{\mathrm{d}x} \left(\sum_{k=1}^N h_{jk}(\boldsymbol{U}) \frac{\mathrm{d}U_k}{\mathrm{d}x} \right) \right) \cdot \phi_j \, \mathrm{d}x \\ &\qquad - \sum_{j=1}^N \left[\left(\sum_{k=1}^N h_{jk}(\boldsymbol{u}) \frac{\mathrm{d}u_k}{\mathrm{d}x} - \sum_{k=1}^N h_{jk}(\boldsymbol{U}) \frac{\mathrm{d}U_k}{\mathrm{d}x} \right) \cdot \phi_j \right]_a^b, \end{aligned} \tag{4.19}$$

using integration by parts again. This uses the assumptions that $\mathrm{d}\boldsymbol{u}/\mathrm{d}x$, $\mathrm{d}\boldsymbol{U}/\mathrm{d}x$, and $\boldsymbol{\phi}$ are continuous. These assumptions follow from the primal and adjoint problem

definitions above. In this case the boundary term disappears because $\boldsymbol{\phi}(a) = \mathbf{0}$ and $h(\boldsymbol{U}, x)\frac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}x}(b) = h(\boldsymbol{u}, x)\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}x}(b)$. For term II we have,

$$\begin{aligned} II &= \int_a^b \sum_{i=1}^{N} (u_i - U_i) \cdot \left(\sum_{j=1}^{N} F_{ij}(x)\phi_j \right) \mathrm{d}x \\ &= \int_a^b \sum_{j=1}^{N} \left(\sum_{i=1}^{N} F_{ij}(x)(u_i - U_i) \right) \phi_j \ \mathrm{d}x \\ &= \int_a^b \sum_{j=1}^{N} \left(f_j(\boldsymbol{u}) - f_j(\boldsymbol{U}) \right) \phi_j \ \mathrm{d}x \qquad \text{[by (4.18)]} \end{aligned}$$

When we calculate $(I + II)$, all terms involving $\boldsymbol{u}$ disappear by definition of the primal problem (4.15). We may therefore write our error bound as,

$$|J(\boldsymbol{u}) - J(\boldsymbol{U})| \leq \sum_{i=1}^{N} \sum_{j=1}^{M} \int_{x_{j-1}}^{x_j} \left| \frac{\mathrm{d}}{\mathrm{d}x} \left(\sum_{k=1}^{N} h_{ik}(\boldsymbol{U}) \frac{\mathrm{d}U_k}{\mathrm{d}x} \right) + f_i(\boldsymbol{U}) \right| \cdot \left| \phi_i \right| \ \mathrm{d}x, \tag{4.20}$$

where $x_0 = a < x_1 \ldots < x_M = b$ is a subdivision of the computational domain.

4.3.2 Application to a single-component model problem

The derivation above applies to a wide class of second-order differential equations. The novel feature of the derivation is that it allows for the nonlinear terms to be a function of the solution and the solution derivatives, whereas in the derivation for nonlinear initial-value ODEs (see Chapter 3, and [75]), the nonlinear terms were a function of the solution only. We therefore apply the derivation to obtain a reduced model for a single-component model problem that contains a term that is a nonlinear function of the solution and the solution derivatives. The model problem is defined by,

$$-\epsilon \frac{\mathrm{d}}{\mathrm{d}x} \left(u \frac{\mathrm{d}u}{\mathrm{d}x} \right) - u^2 + 1 = 0, \quad \text{for } 0 < x < 1, \tag{4.21}$$

with $0 < \epsilon \ll 1$, and boundary conditions $u(0) = u(1) = 0$. Similarly to the linear model problem, this problem has the feature that the solution changes rapidly near $x = 0$ and $x = 1$, but is relatively flat in other regions of the domain – Figure 4.4. For the results below the same quantities of interest as for the previous model problem are used (i.e., the average and a point-value of the solution).

We choose the following form for the reduced primal problem,

$$\begin{aligned} -\epsilon\frac{\mathrm{d}}{\mathrm{d}x}\left(U\frac{\mathrm{d}U}{\mathrm{d}x}\right) - U^2 + 1 = 0 \qquad & \text{for} \quad x \in S \\ -\epsilon\frac{\mathrm{d}^2U}{\mathrm{d}x^2} = 0 \qquad & \text{otherwise,} \end{aligned} \tag{4.22}$$

where S is a union of one or more intervals in the x-domain to be determined by the algorithm; U is defined on the interval $[0,1]$, and the boundary conditions are $U(0) = U(1) = 0$. To fully determine the problem we require that $U(x)$ is continuously differentiable on the interval $[0,1]$. Requiring $U(x)$ to be continuously differentiable means that the error bound from equation (4.20) can be applied. In the reduced model, the reaction and diffusion terms are present in the region S, whereas only the diffusion term is present otherwise, as in section 4.2.1.

The algorithm is then defined as follows:

Initialization: Set $S = \emptyset$, so that, in this case, $-\epsilon\dfrac{\mathrm{d}^2U}{\mathrm{d}x^2} = 0$ for all x. Compute ϕ using equation (4.16).

Iteration:

1. Compute `err(1:N)` where

$$\texttt{err(j)} = \int_{x_{j-1}}^{x_j} \left| -\epsilon\frac{\mathrm{d}}{\mathrm{d}x}\left(U\frac{\mathrm{d}U}{\mathrm{d}x}\right) - U^2 + 1\right| |\phi| \, \mathrm{d}x. \tag{4.23}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding space intervals to S.

3. Re-compute U and ϕ.

Termination: Stop when

$$|J(u) - J(U)| < TOL.$$

The algorithm was implemented in Matlab using the Finite Element Method with linear basis functions. Again, the algorithm successfully generates reduced models – Figure 4.4. As in section 4.2.1, it is shown that when the quantity of interest is $u(0.5)$, the reaction term in the equation needs to be present in a smaller region of the domain than when the quantity of interest is $\displaystyle\int_0^1 u \, \mathrm{d}x$.

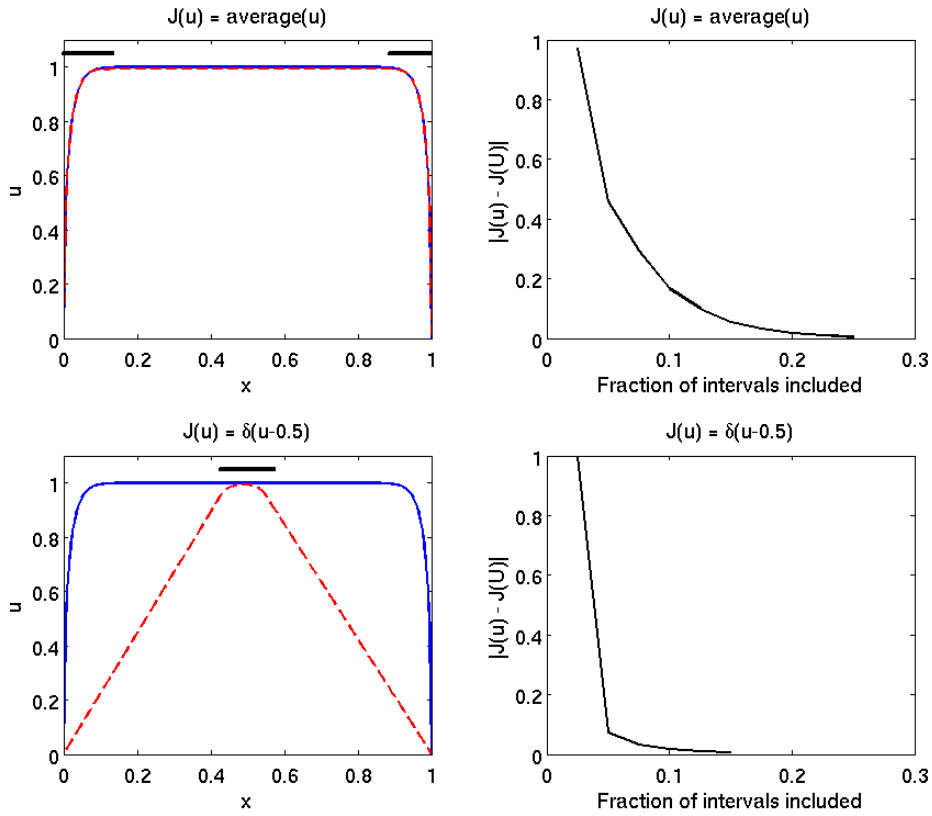


Figure 4.4: Diffusion approximation for nonlinear reaction-diffusion model problem. Different quantities of interest result in different reduced models. *Left panels*: solid blue line is the finite element solution to (4.21) with 200 elements, dashed red line is solution to the reduced model, equation (4.22), and the black lines above the curves represent the space intervals where the reaction term is included in the reduced model. Here $\epsilon = 0.001$, $N_{cont} = 5$ and $TOL = 0.01$. *Right panels*: $|J(u) - J(U)|$, as a function of the fraction of intervals included in the reduced model divided by total number of intervals in domain.

4.3.3 Application to a multi-component model of chemotaxis

The derivation presented in section 4.3.1 was for coupled systems of nonlinear differential equations. We have thus far applied the derivation to analyze a single-component nonlinear reaction-diffusion equation. In this section, we therefore apply the derivation to a two-component system of nonlinear reaction-diffusion equations. The equations come from the chemotaxis literature, [27], and are defined as follows,

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\begin{pmatrix} D & -\chi u \\ 0 & 1 \end{pmatrix}\frac{\mathrm{d}\boldsymbol{u}}{\mathrm{d}x}\right) + \begin{pmatrix} 0 \\ \dfrac{u}{1+\psi u} - v \end{pmatrix} = 0, \tag{4.24}$$

with $\boldsymbol{u} = (u, v)$ defined on the interval $[0, 1]$, $\chi \gg D > 0$, and $\psi > 0$. For the results below we specify the boundary conditions $u(0) = 10$, $u(1) = 0$, $v(0) = 1$, $v(1) = 0$. In this model, u represents cell density, and v represents the concentration of a chemical signal. In the first equation, the term $\dfrac{\mathrm{d}}{\mathrm{d}x}\left(D\dfrac{\mathrm{d}u}{\mathrm{d}x}\right)$ represents undirected diffusive cell migration, and $\dfrac{\mathrm{d}}{\mathrm{d}x}\left(-\chi u\dfrac{\mathrm{d}v}{\mathrm{d}x}\right)$ represents advective cell migration dependent on the gradient of the chemical signal. In the second equation, the term $\dfrac{\mathrm{d}}{\mathrm{d}x}\left(\dfrac{\mathrm{d}v}{\mathrm{d}x}\right)$ represents diffusion of the chemical signal, and $\dfrac{u}{1+\psi u} - v$ represents production and degradation of the chemical signal. Numerous different representations of the chemical signal reaction term have been suggested [27]. The representation chosen here, *saturating signal production*, was chosen because it is nonlinear but still relatively simple, and has been used in a number of studies, for example [46, 51].

For the parameters and boundary conditions chosen above, the advective cell migration term produces high cell density near the $x = 0$ boundary, and low cell density over the rest of the domain. The chemical signal concentration equation is dominated by the diffusion term, and so the concentration solution is approximately a straight line between $v(0) = 1$ and $v(1) = 0$, see Figure 4.5.

As for previous problems we chose the quantities of interest to be averages and point-values of the solution. For this problem there are two components, so we consider four different quantities of interest - the average of each component in the model, and the point-value at $x = 0.5$ of each component in the model.

For the initial reduced model we choose to neglect the advective cell migration term and the reaction term for chemical signal production and degradation. This

leaves a term for diffusive cell migration in the first equation and a term for diffusion of the chemical signal in the second equation. In order to define the reduced model mathematically we introduce the linear diffusion tensor,

$$d = \begin{pmatrix} \boldsymbol{d}_1 \\ \boldsymbol{d}_2 \end{pmatrix} = \begin{pmatrix} D & 0 \\ 0 & 1 \end{pmatrix},$$

where $\boldsymbol{d}_1$ and $\boldsymbol{d}_2$ are vectors representing the first and second rows of the matrix d. Similarly, $\boldsymbol{h}_1$ and $\boldsymbol{h}_2$ are vectors representing the first and second rows of the matrix h.

The reduced model is then defined as follows,

$$\frac{\mathrm{d}}{\mathrm{d}x}\left(\hat{h}(\boldsymbol{U}, x)\frac{\mathrm{d}\boldsymbol{U}}{\mathrm{d}x}\right) + \hat{\boldsymbol{f}}(\boldsymbol{U}, x) = \boldsymbol{0}, \quad \text{for } a < x < b, \tag{4.25}$$

where,

$$\hat{f}_i(\boldsymbol{U}, x) = \begin{cases} f_i(\boldsymbol{U}, x) & \text{for } x \in S_i, \\ 0 & \text{otherwise,} \end{cases}$$

and,

$$\hat{\boldsymbol{h}}_i(\boldsymbol{U}, x) = \begin{cases} \boldsymbol{h}_i(\boldsymbol{U}, x) & \text{for } x \in S_i, \\ \boldsymbol{d}_i & \text{otherwise,} \end{cases}$$

where S_i is a union of one or more intervals in the x-domain to be determined by the algorithm, $\boldsymbol{U} = (U_1, U_2)$ is defined on the interval $[0, 1]$, and we specify boundary conditions $\boldsymbol{U}(0) = \boldsymbol{u}(0)$, $\boldsymbol{U}(1) = \boldsymbol{u}(1)$. To fully determine the problem we require that $\boldsymbol{U}(x)$ is continuously differentiable on the interval $[0, 1]$. Requiring $\boldsymbol{U}(x)$ to be continuously differentiable means that the error bound from equation (4.20) can be applied.

The algorithm is then defined as follows:

Initialization: Set $\boldsymbol{S} = \emptyset$, so that, in this case, $D\dfrac{\mathrm{d}^2 U_1}{\mathrm{d}x^2} = 0$, $\dfrac{\mathrm{d}^2 U_2}{\mathrm{d}x^2} = 0$ for all x. Compute $\boldsymbol{\phi}$ using equation (4.16).

Iteration:

1. Compute `err(1:2,1:M)` where

$$\texttt{err}(\texttt{i}, \texttt{j}) = \int_{x_{j-1}}^{x_j} \left| \frac{\mathrm{d}}{\mathrm{d}x}\left(\sum_{k=1}^{N} a_{ik}(\boldsymbol{U})\frac{\mathrm{d}U_k}{\mathrm{d}x}\right) + f_i(\boldsymbol{U}) \right| \cdot \left| \phi_j \right| \mathrm{d}x. \tag{4.26}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding space intervals to $\boldsymbol{S}$.

3. Re-compute $\boldsymbol{U}$ and $\boldsymbol{\phi}$.

Termination: Stop when

$$|J(\boldsymbol{u}) - J(\boldsymbol{U})| < TOL.$$

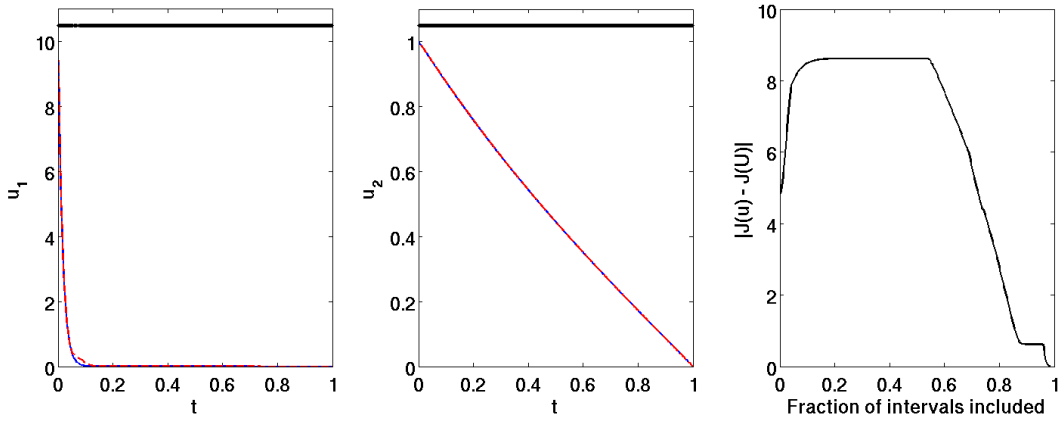
The reduced models generated by the algorithm are shown in Figures 4.5 and 4.6. The results show that for $J(\boldsymbol{u}) = \int_0^1 u_1 \, \mathrm{d}x$, $J(\boldsymbol{u}) = \int_0^1 u_2 \, \mathrm{d}x$, and $J(\boldsymbol{u}) = \int_0^1 u_2 \, \delta(x - 0.5) \, \mathrm{d}x$ the reduced model is almost identical to the original model, i.e., the original differential equation is solved on almost the entire domain. If there does exist a larger fraction of the domain where the reduced model is a good approximation, then the algorithm is unable to identify it. For $J(\boldsymbol{u}) = \int_0^1 u_1 \, \delta(x - 0.5) \, \mathrm{d}x$, the advective cell migration term can be neglected for $0 < x < 0.25$, because an accurate solution for u_1 is not required in this region to accurately compute the quantity of interest.

4.4 Robustness of *a posteriori* model reduction to the N_{cont} parameter

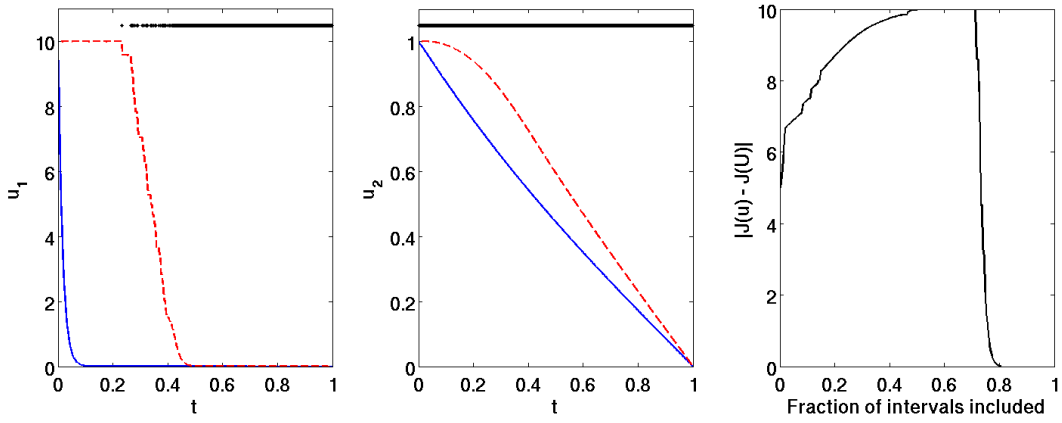
As in the previous chapter, the robustness of *a posteriori* model reduction to the algorithm parameter N_{cont} was tested by comparing the regions of the domain identified by the algorithm for a range of N_{cont} values. For the linear BVP, the results are relatively robust to N_{cont} – Figure 4.7. In the case of the diffusion approximation there are differences because on the first iteration a region in the middle of the domain is identified, and on subsequent iterations regions near the boundary of the domain are identified, for all values of N_{cont}. Therefore for larger values of N_{cont} a greater region in the middle of the domain is identified because more intervals are identified on the first iteration. When the reaction approximation is used the results are robust to the value of N_{cont} (not shown).

For the chemotaxis equations the results are less easy to interpret. There do appear to be differences in the region identified by the algorithm as N_{cont} is varied, although these differences are relatively minor – Figures 4.8-4.9.

Note that regardless of the value of N_{cont}, the algorithm will always construct a reduced model such that $|J(\boldsymbol{u}) - J(\boldsymbol{U})| < TOL$. If the original differential operator

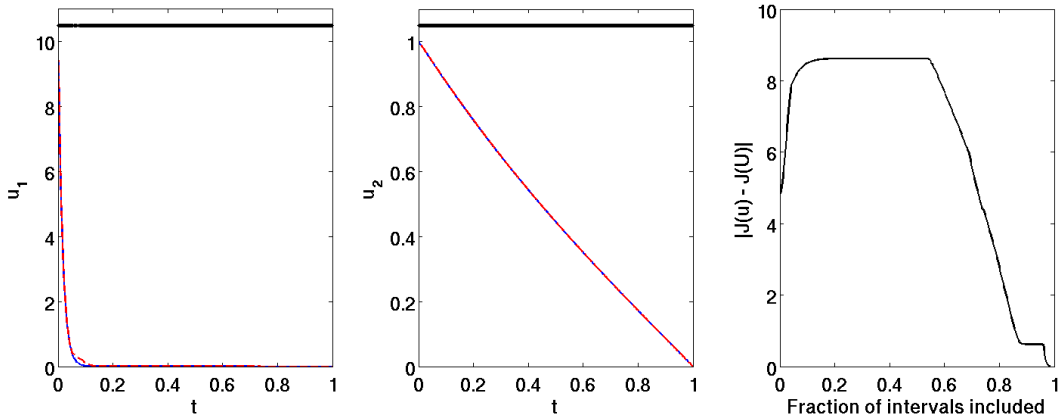


(a) $J(\boldsymbol{u}) = \int_0^1 u_1 \, \mathrm{d}x$

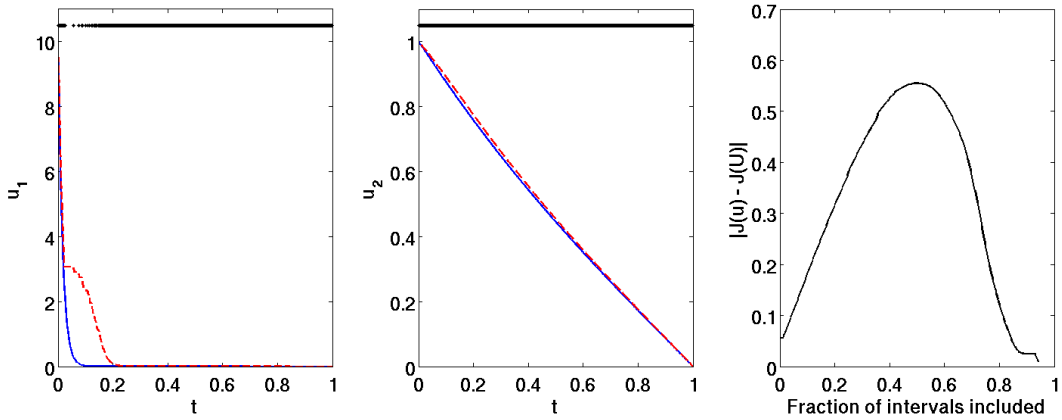


(b) $J(\boldsymbol{u}) = \int_0^1 u_1 \, \delta(x - 0.5) \, \mathrm{d}x$

Figure 4.5: Diffusion approximation for nonlinear reaction-diffusion model of chemotaxis. Different quantities of interest result in different reduced models. *Left and middle figures*: solid blue line is the numerical solution to the chemotaxis model (4.24), obtained using the finite element method with linear basis functions. The boundary conditions were $\boldsymbol{u}(0) = [10,\ 1]$, $\boldsymbol{u}(1) = [0,\ 0]$. The dashed red line is $\boldsymbol{U}$, and the black lines above the curves represent the space intervals where the reaction term is included in the reduced model. Here $N_{cont} = 5$, $TOL = 1e-2$. *Right figure* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$, as a function of the fraction of the domain included in the reduced model.

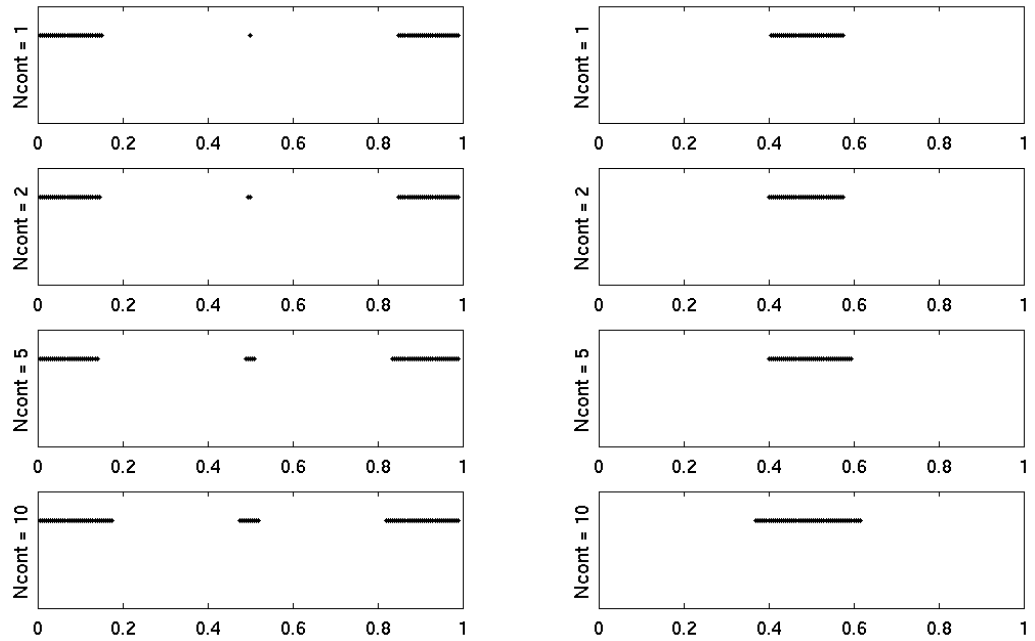


(a) $J(\boldsymbol{u}) = \int_0^1 u_2 \, \mathrm{d}x$



(b) $J(\boldsymbol{u}) = \int_0^1 u_2 \, \delta(x - 0.5) \, \mathrm{d}x$

Figure 4.6: Diffusion approximation for nonlinear reaction-diffusion model of chemotaxis. Different quantities of interest result in different reduced models. *Left and middle figures*: solid blue line is the numerical solution to the chemotaxis model (4.24), obtained using the finite element method with linear basis functions. The boundary conditions were $\boldsymbol{u}(0) = [10,\ 1]$, $\boldsymbol{u}(1) = [0,\ 0]$. The dashed red line is $\boldsymbol{U}$, and the black lines above the curves represent the space intervals where the reaction term is included in the reduced model. Here $N_{cont} = 5$, $TOL = 1e-2$. *Right figure* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$, as a function of the fraction of the domain included in the reduced model.



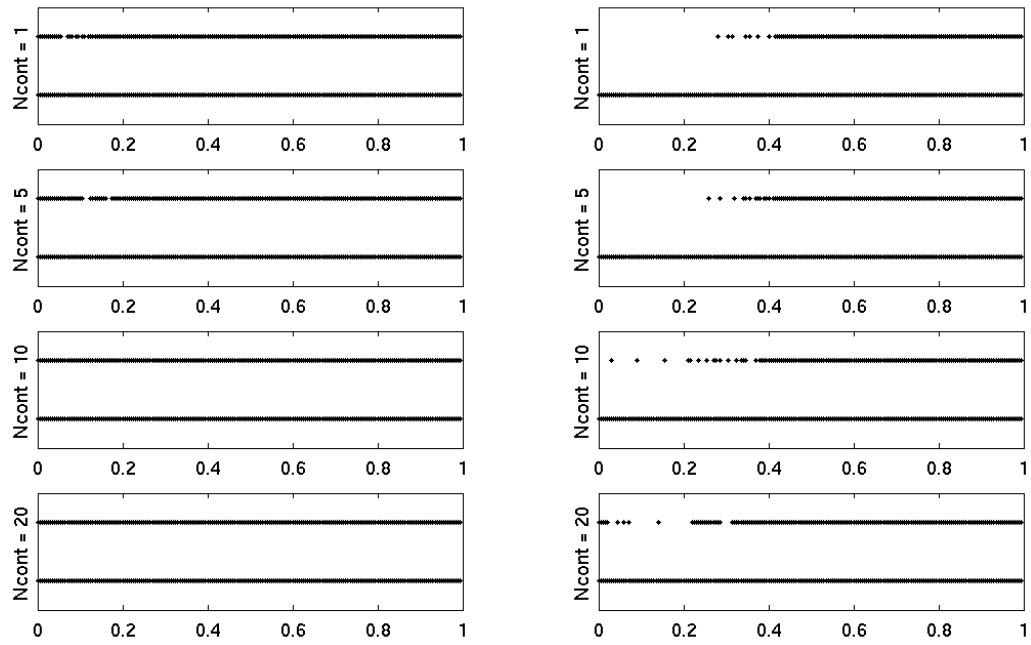
(a) $J(u) = \int_0^1 u \, \mathrm{d}x$ **(b)** $J(u) = u(0.5) = \int_0^1 u \, \delta(x - 0.5) \, \mathrm{d}x$

Figure 4.7: Robustness of *a posteriori* model reduction to the N_{cont} parameter for linear reaction-diffusion model problem, diffusion approximation. As in Figure 4.2 and, the black lines represent the time intervals that are included in reduced model. In each panel, the top line represents u_1, and the bottom line represents u_2. Apart from N_{cont}, all other parameter values are the same as in Figure 4.2.

is applied over a greater proportion of the domain because a larger value of N_{cont} has been used, this means that there is more redundancy in the reduced model than there would be if a smaller N_{cont} had been used. Another way of saying this is that less information about the model may be obtained if a larger value of N_{cont} is used. For some problems it may be justified to choose a larger value of N_{cont} in order to reduce computation time.

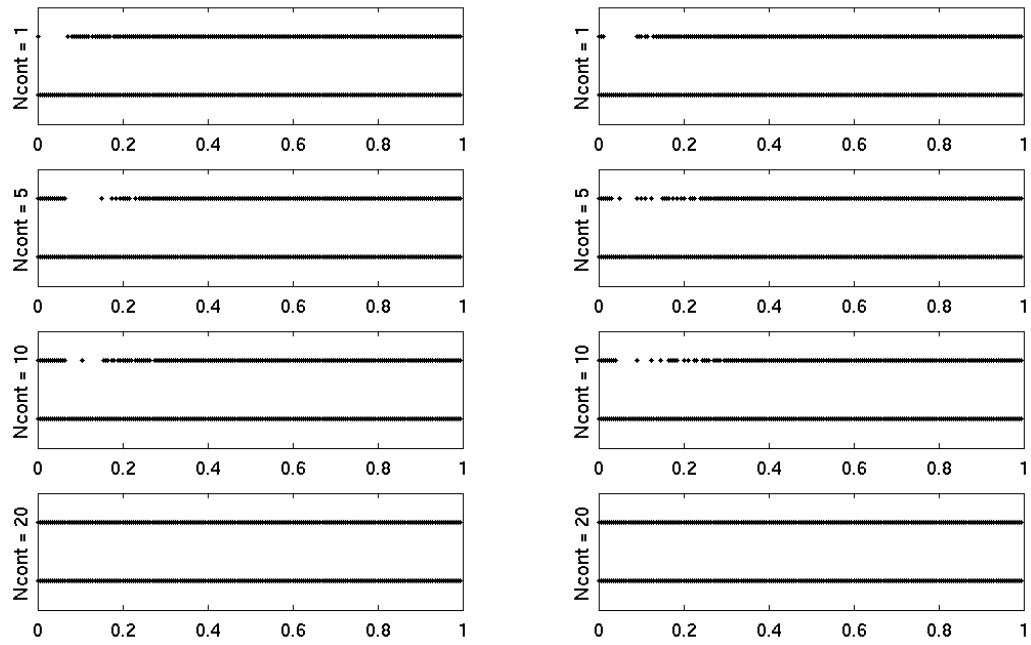
4.5 Computational Efficiency

Using Matlab on a standard desktop computer, the results obtained in this chapter require from 1-2 minutes to 2-3 hours of computer time depending on the complexity of



(a) $J(\boldsymbol{u}) = \int_0^1 u_1 \, \mathrm{d}x$ **(b)** $J(\boldsymbol{u}) = \int_0^1 u_1 \, \delta(x - 0.5) \, \mathrm{d}x$

Figure 4.8: Robustness of *a posteriori* model reduction to the N_{cont} parameter nonlinear reaction-diffusion model of chemotaxis. As in Figure 4.5, the black lines represent the time intervals that are included in reduced model. In each panel, the top line represents u_1, and the bottom line represents u_2. Apart from N_{cont}, all other parameter values are the same as in Figure 4.5.



(a) $J(\boldsymbol{u}) = \int_0^1 u_2 \, \mathrm{d}x$ **(b)** $J(\boldsymbol{u}) = \int_0^1 u_2 \, \delta(x - 0.5) \, \mathrm{d}x$

Figure 4.9: Robustness of *a posteriori* model reduction to the N_{cont} parameter nonlinear reaction-diffusion model of chemotaxis. As in Figure 4.6, the black lines represent the time intervals that are included in reduced model. In each panel, the top line represents u_1, and the bottom line represents u_2. Apart from N_{cont}, all other parameter values are the same as in Figure 4.6.

the problem, and the value of N_{cont} (the number of elements included on each iteration). Computational savings can be made by using a compiled language such C++, as shown in Chapter 5 for the monodomain equation.

In this chapter we have only considered 1-D problems. We believe that the method would also be computationally feasible for a large number of 2-D steady-state problems. A 2-D problem will typically be solved on a computational mesh that contains $O(n^2)$ elements, compared to a corresponding 1-D problem that is solved on a mesh containing $O(n)$ elements. As the *a posteriori* model reduction method is iterative and includes a fixed number of elements on each iteration, the total computation time for a 2-D problem is at least the square of that for a 1-D problem. The other factor that needs to be taken into account is the extra computation time required on each iteration for solving a 2-D problem as opposed to a 1-D problem. We estimate that 2-D models that require 30-60 seconds to solve on a computational mesh of 10,000 elements, require up to 1-2 days of computer time for the *a posteriori* model reduction method to be applied if $N_{cont} = 10$, and up to 2-4 weeks of computer time if $N_{cont} = 1$. This information is summarized in Table 4.1.

Table 4.1: Example computation times for steady-state problems. The first two rows are actual computation times for the chemotaxis model, equation (4.24), in Matlab. The third row is a hypothetical computation time for a 2-D steady-state problem. A single-component model on a 100×100 quadrilateral mesh would have 10,000 elements, as would a four-component model on a 50×50 quadrilateral mesh.

Number of elements	N_{cont}	Time required to solve original model	Maximum number of iterations	Maximum amount of time required
200	10	4.77s	400	12 minutes
1,000	1	30s	$2,000$	55.1 hours
10,000	1	60s	$10,000$	4 weeks

4.6 Summary

In this chapter we have extended the *a posteriori* model reduction methodology to a large class of Boundary-Value Problems. The first problem we considered was a linear reaction-diffusion equation. For this equation we developed the method so that the algebraic reaction term is removed in automatically identified regions of the domain. We then developed the method so that the diffusion term is removed in automatically

identified regions of the domain. We called these methods the diffusion approximation and the reaction approximation respectively. These developments parallel the developments for initial-value ODEs that we presented in the previous chapter. In both cases different adjoint problems were required for different types of approximation. Also, when the highest order derivative in the reduced model is lower than the highest order derivative in the original problem, then jump terms arise in the error bound because of discontinuities in the reduced model solution or its derivatives and the adjoint solution or its derivatives. This is the case for both the quasi-steady state approximation presented in the previous chapter, and for the reaction approximation presented in this chapter. In applying the error bounds, these jump terms have been incorporated into the element-wise errors. An area for further work would be to investigate what effect, if any, ignoring these jump terms would make on the resulting reduced model. We have done some preliminary experiments into this and found that for the problems considered here, the resulting reduced model is similar when the jump terms are ignored. For the 1 dimensional problems considered in this chapter it is not challenging to calculate jump terms. However, for 2 or 3 dimensional problems calculating the jump terms is complicated as it is necessary to identify neighbouring elements in the domain, and calculate integrals involving derivatives of the solution across element boundaries.

An interesting feature of the results for the linear reaction-diffusion equation is how similar the included region in the reduced model is for the diffusion approximation and the reaction approximation. From inspecting the equation, $-\epsilon u'' + u = 1, u(0) = u(1) = 0$, we can see that if ϵ is small, the dominant terms away from the boundary are u and 1. Doing an elementary perturbation analysis would therefore tell us that $u = 1$ is a good approximation away from the boundary, and this is confirmed by applying *a posteriori* model reduction. However applying the same perturbation analysis would not tell you that $-\epsilon u'' = 0$ is also a good approximation away from the boundary, provided that $-\epsilon u'' + u = 1$ is solved near the boundary. For this particular equation, it is clear from inspecting the solution that $-\epsilon u'' = 0$ can be a good approximation away from the boundary because the solution is relatively flat in this region, and the solution to $-\epsilon u'' = 0$ is a straight line. For other problems this kind of information may not be so readily available, and applying *a posteriori* model reduction could yield non-intuitive insights that cannot be obtained using perturbation methods. A more

in depth comparison of perturbation methods and *a posteriori* model reduction is a potentially fruitful area for further work.

In the discussion above we were implicitly assuming that we are interested in accurately computing the solution over the whole domain. For the linear BVP, this corresponds to taking the quantity of interest to be the average of the solution over the whole domain. We also obtained results for when the quantity of interest is a point-value of the solution. In this case, an accurate solution to the linear reaction-diffusion equation is only required in the immediate neighbourhood of the point of interest. This was demonstrated by applying *a posteriori* model reduction.

In the second part of the chapter we derived an error bound for nonlinear reaction-diffusion equations. The novel feature of the derivation is that the second-order term in the equation can be nonlinear. This allows for the method to be applied to models that include effects such as gradient-dependent advection and viscosity. As for the preceding problems, the error in the quantity of interest was expressed as the product of a residual with an adjoint solution.

We applied the error bound to a single-component nonlinear reaction diffusion equation, and to a coupled nonlinear model of chemotaxis. The solution of the nonlinear reaction-diffusion equation was qualitatively the same as the solution to the linear reaction-diffusion equation considered earlier. As a consequence of this the results obtained for the nonlinear reaction-diffusion equation are very similar to the results for the linear reaction-diffusion equation. We have not presented the reaction approximation for nonlinear equations, although it would be relatively straightforward to apply the reaction approximation by using the methods from this chapter.

For the chemotaxis equations, the *a posteriori* model reduction method returns a reduced model that is very similar to the original model for three out of four of the quantities of interest chosen. By this we mean that in the model produced by the algorithm, the original differential operator is applied over almost the entire domain. These results are somewhat surprising given that one of the model component solutions is flat over a large portion of the domain. One would expect that a diffusion approximation would be appropriate in this region. However this is not identified by the method. For the other quantity of interest, $u(0.5)$, the method demonstrated that, on a domain $x = [0, 1]$, the cell density at $x = 0.5$ can be accurately computed with a reduced model where the concentration-dependent advection term is removed over the

interval $0 < x < 0.25$. In this case, the quantity of interest is accurately calculated even though the solution over the whole domain is inaccurate for both model components.

We also assessed the robustness of the method to varying the algorithm parameter N_{cont}. This was done in order to find out whether the method can be made computationally faster by increasing N_{cont}. The results of varying N_{cont} for the chemotaxis model show that the resulting reduced models are not widely different. However the resulting reduced models are different enough to suggest that caution should be used in setting the value of N_{cont}, particularly for nonlinear multi-component models.

5

Partial Differential Equations

In Chapter 3 we developed the *a posteriori* model reduction methodology for IVPs, and in Chapter 4 we developed the methodology for BVPs. In this chapter we consider reaction-diffusion and advection-diffusion PDEs in one space dimension. Reaction-diffusion and advection-diffusion PDEs are common models in biology, for examples see [40, 50]. The problems will contain the same type of terms that the problems encountered in Chapters 3 and 4 contained. Therefore we combine the results for IVPs with the results for BVPs to derive an error bound first for a specific PDE – Fisher's equation – and then for a general class of PDEs. The second of these error bounds can be applied to PDEs with a linear first-order time derivative and algebraic, first-order and second-order x-derivative terms, any combination of which may be nonlinear. We apply the general error bound to study three more PDEs, each of which have different travelling wave properties. Applying the method to these problems demonstrates the wide applicability of the method, and also gives insight into the properties of different types of travelling waves.

A feature of all the problems analyzed in this chapter is that they contain a diffusion term. For PDEs that do not contain a diffusion term, there are restrictions in applying *a posteriori* model reduction. We briefly discuss these restrictions for linear and quasi-linear first-order PDEs.

We also discuss computational efficiency concluding that the current implementation of the algorithm makes it challenging to apply the method to problems in more than two independent variables.

The structure of the chapter is as follows. In the first section we briefly discuss the numerical schemes used to solve the equations in the chapter. In the second section we derive an error bound for Fisher's equation - a single-component reaction-diffusion equation. The error bound is then applied to generate a reduced model for Fisher's equation. In the third section we give an error bound derivation for a more general PDE system containing a linear first-order time derivative term and algebraic, first order and second order x-derivative terms, any combination of which may be nonlinear. The error bound is then applied to the monodomain equation (a coupled reaction-diffusion equation), a linear advection diffusion equation, and a nonlinear advection-diffusion equation. Then we briefly discuss how the method could be applied to linear and quasi-linear first-order PDEs, section 5.4. This is followed by an analysis of the efficiency of the algorithm, sections 5.5, 5.6.

5.1 Numerical schemes for time-dependent problems

For similar reasons to those given in the previous chapter we use the Finite Element Method (FEM) to generate a numerical solution to all problems in this chapter. When applying the finite element method to problems with time and space as independent variables, it is common to apply the finite element method in space, and use some other method to discretise time. As in the previous chapter we do not require state-of-the-art numerical methods to solve the equations because computational efficiency is not a major consideration here. All equations in this chapter are solved with a uniform time-step. For Fisher's equation, and the advection-diffusion equations we apply an implicit scheme, equivalent to the backward Euler method. For the monodomain equation a semi-implicit scheme is used, details of which can be found in [56, 74]. This semi-implicit scheme for the monodomain equation is part of the Chaste environment, [57], and has shown to be a stable and computationally efficient scheme for the monodomain equation [56, 74].

As in the previous chapter, we have evaluated accuracy in a heuristic manner by evaluating and inspecting the finite element solution for several different values of h and Δt, where Δt is the size of the time-step used in the numerical scheme.

5.2 Fisher's equation

5.2.1 Derivation of error bound

We begin by deriving an error bound for Fisher's equation, a single-component time-dependent reaction-diffusion equation that has travelling wave solutions. The derivation follows a similar pattern to the derivation for initial-value ODEs and BVPs in previous chapters, the difference being that there are now two independent variables rather than one.

Fisher's equation is defined as follows,

$$\frac{\partial u}{\partial t} - \frac{\partial^2 u}{\partial x^2} - f(u) = 0, \tag{5.1}$$

where $f(u) = k\,u(1-u)$, u is defined for $0 < x < 10$, $0 < t < T$. For the results below, $k = 10$, and $T = 3$. Elementary analysis shows that $u = 0$ and $u = 1$ are unstable and stable steady states of this equation respectively. We apply the initial conditions $u(x, 0) = \exp(-x^2)$, and the boundary conditions $u(0, t) = 1, u(10, t) = 0$. The solution to equation (5.1) is shown in Figure 5.1. Since $u = 1$ and $u = 0$ are steady states, removing $f(u)$ from the equation in regions where the solution is near 1 or 0 should cause a relatively small change in the solution.

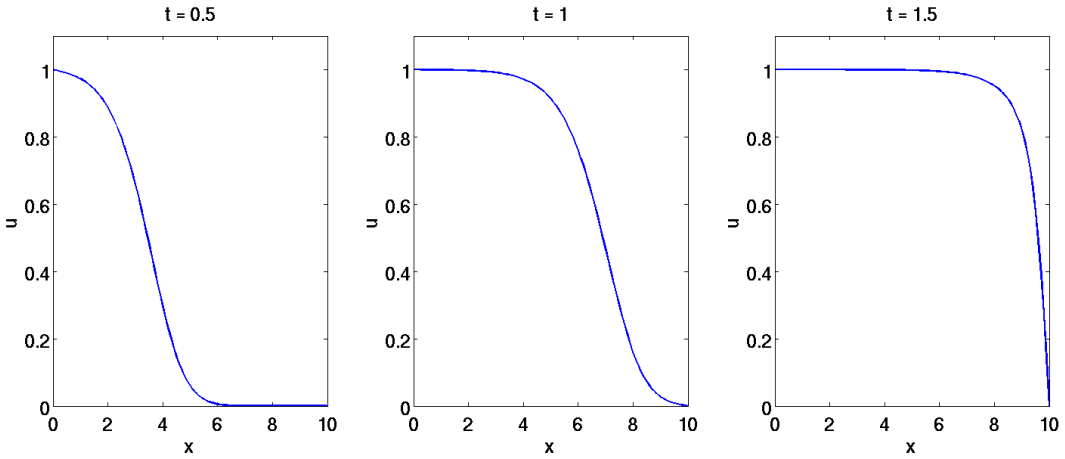


Figure 5.1: Solution to Fisher's equation, (5.1), at selected time-points. Here $u(x, 0) = \exp(-x^2)$, $u(0, t) = 1, u(10, t) = 0$. The finite element method with linear basis functions was used for the x domain (100 elements) and the backwards Euler formula was used for the time domain ($\Delta t = 0.02$). For solution at $t = 0$ and $t = 3$ see Figure 5.2.

As before we are interested in some functional of the solution, which we denote by $J(u) = \langle u, g \rangle = \int_0^T \int_0^{10} u \; g \; \mathrm{d}x \, \mathrm{d}t$. Suppose that we have a solution, U, to a reduced problem. We require $U(x,t)$ and $\partial U(x,t)/\partial x$ to be continuous, and to satisfy the boundary and initial conditions of the original primal problem. The adjoint problem is then,

$$-\frac{\partial \phi}{\partial t} - \frac{\partial^2 \phi}{\partial x^2} - A\phi = g, \tag{5.2}$$

where ϕ is defined for $0 < x < 10$, $0 < t < T$, and the coefficient A is given by,

$$A(x,t) = \int_0^{10} \frac{\partial f}{\partial u}(\alpha(s,x,t)) \; \mathrm{d}s, \tag{5.3}$$

where $\alpha(s,x,t) = su(x,t) + (1-s)U(x,t)$. We apply the condition $\phi(x,T) = 0$ at $t = T$, and the boundary conditions $\phi(0,t) = \phi(10,t) = 0$.

Since A is of the same form as in Chapter 3, we can apply equation (3.7) in the error bound below, i.e., $A(u-U) = f(u) - f(U)$.

Now we can express the error in the quantity of interest introduced by the approximation U to u as,

$$\begin{aligned}
J(u) - J(U) &= \int_0^T \int_0^{10} (u-U) \; g \; \mathrm{d}x \, \mathrm{d}t \\
&= \int_0^T \int_0^{10} (u-U) \left(-\frac{\partial \phi}{\partial t} - \frac{\partial^2 \phi}{\partial x^2} - A\phi \right) \mathrm{d}x \, \mathrm{d}t \\
&= \int_0^T \int_0^{10} \left(\frac{\partial u}{\partial t} - \frac{\partial U}{\partial t} \right) \phi \; \mathrm{d}x \, \mathrm{d}t - \int_0^{10} \Big[(u-U)\phi \Big]_0^T \mathrm{d}x \\
&\quad + \int_0^T \int_0^{10} \left(\frac{\partial u}{\partial x} - \frac{\partial U}{\partial x} \right) \frac{\partial \phi}{\partial x} \; \mathrm{d}x \, \mathrm{d}t - \int_0^T \left[(u-U) \frac{\partial \phi}{\partial x} \right]_0^{10} \mathrm{d}t \\
&\quad - \int_0^T \int_0^{10} A \; (u-U) \; \phi \; \mathrm{d}x \, \mathrm{d}t,
\end{aligned} \tag{5.4}$$

using integration by parts. This uses the assumptions that $u, \phi, \dfrac{\partial \phi}{\partial x}$ are continuous on $0 < x < 10$, $0 < t < T$. These assumptions follow directly from the definitions above. The first boundary term is zero as $u(x,0) - U(x,0) = 0$ and $\phi(x,T) = 0$. The second boundary term is zero as $u(0,t) - U(0,t) = 0$ and $u(10,t) - U(10,t) = 0$.

We now apply integration by parts again and use equation (3.7) to obtain,

$$\begin{aligned} J(u) - J(U) &= \int_0^T \int_0^{10} \left(\frac{\partial u}{\partial t} - \frac{\partial U}{\partial t} \right) \phi \, \mathrm{d}x \, \mathrm{d}t \\ &\quad - \int_0^T \int_0^{10} \left(\frac{\partial^2 u}{\partial x^2} - \frac{\partial^2 U}{\partial x^2} \right) \phi \, \mathrm{d}x \, \mathrm{d}t + \int_0^T \left[\left(\frac{\partial u}{\partial x} - \frac{\partial U}{\partial x} \right) \phi \right]_0^{10} \mathrm{d}t \\ &\quad - \int_0^T \int_0^{10} \left(f(u) - f(U) \right) \phi \, \mathrm{d}x \, \mathrm{d}t \\ &= \int_0^T \int_0^{10} \left(-\frac{\partial U}{\partial t} + \frac{\partial^2 U}{\partial x^2} + f(U) \right) \phi \, \mathrm{d}x \, \mathrm{d}t. \end{aligned} \tag{5.5}$$

This uses the assumption that $\dfrac{\partial u}{\partial x}$, $\dfrac{\partial U}{\partial x}$, ϕ are continuous on $0 < x < 10$, $0 < t < T$. The boundary term is zero as $\phi(0,t) = \phi(10,t) = 0$. All terms involving u disappear by definition of the primal problem, (5.1).

Therefore we may write our error bound as,

$$|J(u) - J(U)| \leq \sum_{j=1}^{M} \sum_{k=1}^{K} \int_{t_{k-1}}^{t_k} \int_{x_{j-1}}^{x_j} \left| \frac{\partial U}{\partial t} - \frac{\partial^2 U}{\partial x^2} - f(U) \right| |\phi| \, \mathrm{d}x \, \mathrm{d}t, \tag{5.6}$$

where $0 = x_0 < x_1 \ldots < x_M = 10$ is a subdivision of the interval $[0, 10]$ and $0 = t_0 < t_1 \ldots < t_K = T$ is a subdivision of the interval $[0, T]$. Note that the expression $\left| \dfrac{\partial U}{\partial t} - \dfrac{\partial^2 U}{\partial x^2} - f(U) \right|$ is the left-hand side of the original PDE, equation (5.1), with U instead of u. This expression is called the residual and it measures the extent to which U satisfies the original PDE.

5.2.2 Application to Fisher's equation

The algorithm we now present generates a reduced model of the form,

$$\begin{aligned} \frac{\partial U}{\partial t} - \frac{\partial^2 U}{\partial x^2} - f(U) = 0, \quad &\text{for } (x,t) \in S \\ \frac{\partial U}{\partial t} - \frac{\partial^2 U}{\partial x^2} = 0, \quad &\text{otherwise,} \end{aligned} \tag{5.7}$$

where S is a union of one or more intervals in the (x,t)-domain that is determined by the algorithm, and U is defined for $0 < x < 10$, $0 < t < T$. We apply the initial conditions and boundary conditions of the original problem: $U(x,0) = u(x,0) = \exp(-x^2)$, and $U(0,t) = u(0,t) = 1, U(10,t) = u(10,t) = 0$. This form of reduced model corresponds to a time-dependent version of the diffusion approximation from the previous chapter.

The reaction and diffusion terms are present in the region S, whereas only the diffusion term is present otherwise.

For the results below we choose the quantity of interest firstly to be,

$$J(u) = \frac{1}{10\,T}\int_0^T \int_0^{10} u \; \mathrm{d}x \, \mathrm{d}t, \tag{5.8}$$

which may be interpreted as the average of u over $t = [0, T]$ and $x = [0, 10]$. Then we consider,

$$J(u) = \frac{1}{T}\int_0^T u(x^*, t) \; \mathrm{d}t = \frac{1}{T}\int_0^T \int_0^{10} u \; \delta(x - x^*) \; \mathrm{d}x \, \mathrm{d}t, \tag{5.9}$$

where $0 < x^* < 10$. This can be interpreted as the average value of u at $x = x^*$ over $t = [0, T]$. In the results below $x^* = 5$. As before $\delta(\cdot)$ is the Dirac delta function, and can be thought of as the limit of a zero-centred normal distribution as the standard deviation tends to zero.

The algorithm is then defined as follows:

Initialization: Set $S = \emptyset$, so that, in this case, $\dfrac{\partial U}{\partial t} - \dfrac{\partial^2 U}{\partial x^2} = 0$ for all x. Compute ϕ using equation (5.2).

Iteration:

1. Compute `err(1:N,1:K)` where

$$\texttt{err}(\texttt{j},\texttt{k}) = \int_{t_{k-1}}^{t_k} \int_{x_{j-1}}^{x_j} \left| \frac{\partial U}{\partial t} - \frac{\partial^2 U}{\partial x^2} - f(U) \right| |\phi| \; \mathrm{d}x \, \mathrm{d}t. \tag{5.10}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding regions of the domain to S.

3. Re-compute U, ϕ.

Termination: Stop when

$$|J(u) - J(U)| < TOL.$$

As in the previous chapters, the inputs to the algorithm are as follows: N_{cont}, the number of (x, t)-intervals added to S on each iteration of the algorithm; TOL, the desired accuracy in the quantity of interest; and, g, the weighting function in the quantity of interest.

The reduced models generated by the algorithm are shown in Figures 5.2 and 5.3. The resulting reduced models are different for the different quantities of interest. In Figure 5.2, which shows the application of the algorithm when the quantity of interest is the average over time and space, the reaction term is included in a band that moves across the domain. This band is located on the wavefront of the solution as it travels from near $x = 0$ to near $x = 10$. The results show that the reaction term only needs to be included on the wave-front, i.e., in regions of the domain where u is not close to either 1 or 0. Since we have set the Dirichlet condition $u(10, t) = 0$, the reaction term is included near $x = 10$ for $1 < t < 3$. If the reaction term was not included at all, the solution would tend to a straight line between 1 and 0. When the quantity of interest is the average of $u(5, t)$ over time, the included region is again a band that moves across the domain, except in this case the band stops moving across the domain when it reaches $x = 5$ – Figure 5.3.

5.3 Nonlinear mixed-type PDEs

5.3.1 Derivation of error bound

In the previous section we analysed Fisher's equation – a single-component time-dependent reaction-diffusion equation. We combined the analysis for IVPs (Chapter 3, [75]) with integration by parts to derive an error bound for a user-defined quantity of interest. The error bound for Fisher's equation has the same form as error bounds in previous chapters. In this section, we derive an error bound, again with the same form, in a more general setting – for PDEs with a linear first-order time derivative term and algebraic, first order and second order x-derivative terms, any combination of which may be nonlinear. The derivation is a straightforward application of the mathematical results obtained in previous sections. Results from previous sections will be cited in order to shorten the derivation.

Let the general form of the primal problem be defined as,

$$\frac{\partial \boldsymbol{u}}{\partial t} + \frac{\partial}{\partial x}\left(h(\boldsymbol{u}, x, t)\frac{\partial \boldsymbol{u}}{\partial x}\right) + \frac{\partial}{\partial x}\left(\boldsymbol{c}(\boldsymbol{u}, x, t)\right) + \boldsymbol{f}(\boldsymbol{u}, x, t) = \boldsymbol{0}, \tag{5.11}$$

for $a < x < b$, $0 < t < T$, where $h(\boldsymbol{u}, x, t)$ is a general $N \times N$ matrix, $\boldsymbol{u}(x, t)$, $\boldsymbol{c}(\boldsymbol{u}, x, t)$, $\boldsymbol{f}(\boldsymbol{u}, x, t)$ are vectors of length N. We specify initial conditions $\boldsymbol{u}(x, 0) = \boldsymbol{u}_0$, and Dirichlet boundary conditions, $\boldsymbol{u}(a, t) = \boldsymbol{u}_a$, $\boldsymbol{u}(b, t) = \boldsymbol{u}_b$. The derivation would be

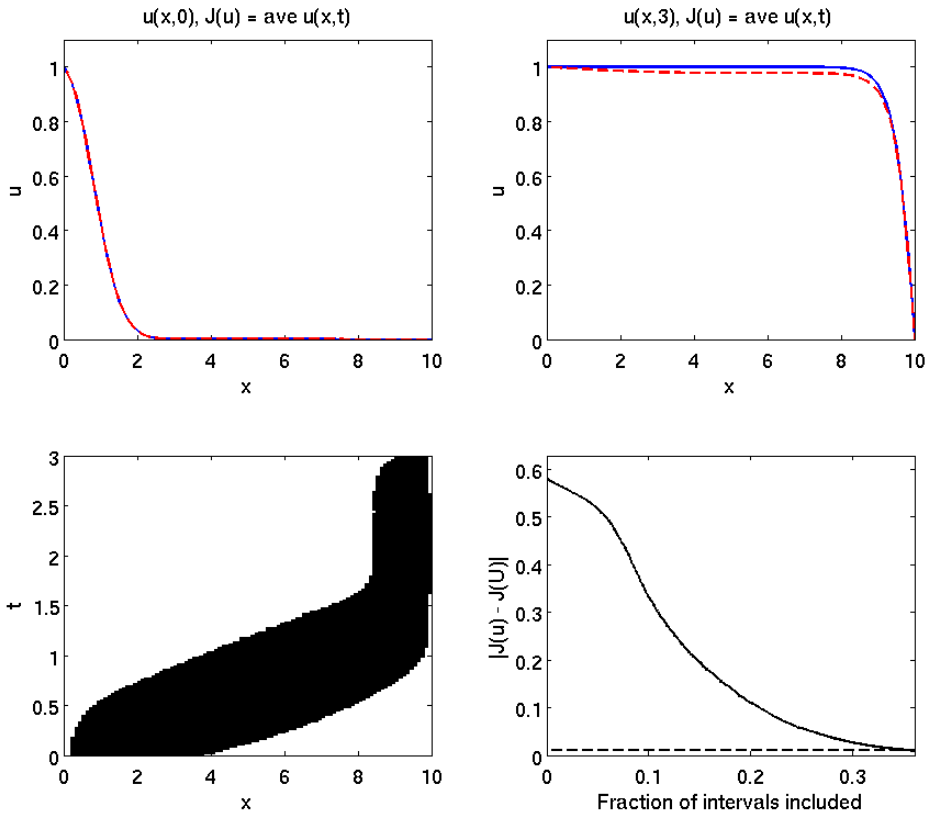


Figure 5.2: Time-dependent diffusion approximation for Fisher's equation, $J(u) = \frac{1}{10\,T}\int_0^T \int_0^{10} u \,\mathrm{d}x\,\mathrm{d}t$. *Top panels*: solid blue line is the numerical solution to Fisher's equation, (5.1). The finite element method with linear basis functions was used for the x domain (100 elements) and the backwards Euler formula was used for the time domain ($\Delta t = 0.02$). The dashed red line is U. Here $T = 3$, $N_{cont} = 10$ and $TOL = 10^{-2}$. *Bottom left panel*: region of the domain where the reaction term is included in the reduced model. *Bottom right panel*: $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

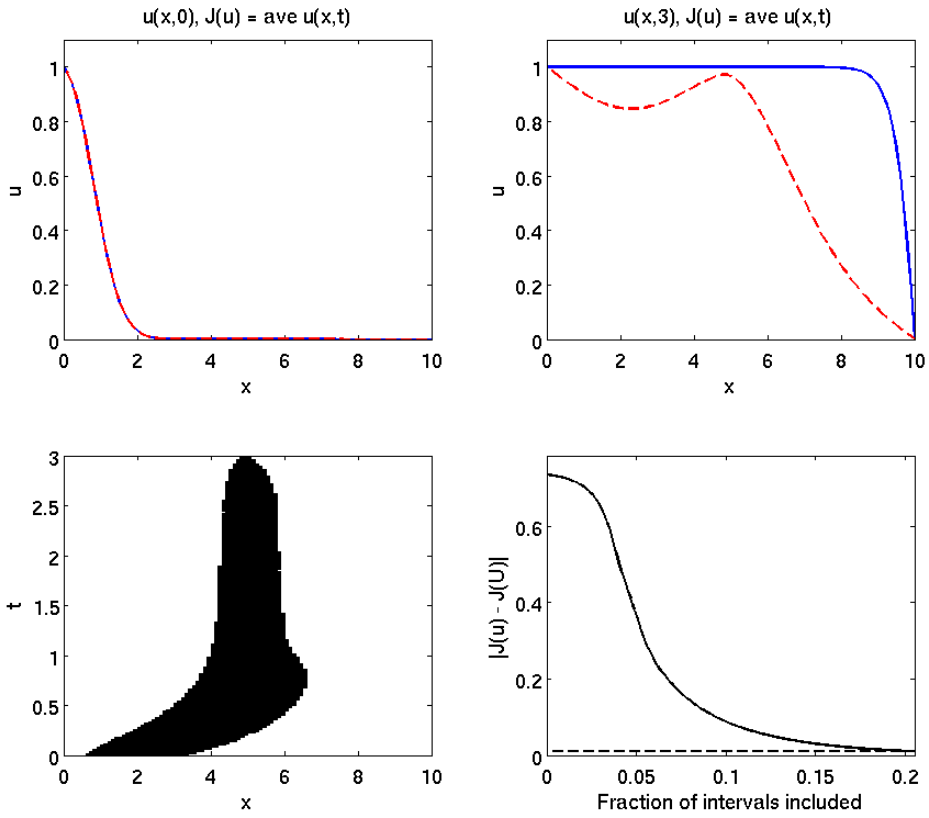


Figure 5.3: Time-dependent diffusion approximation for Fisher's equation, $J(u) = \frac{1}{T}\int_0^T u(5,t)\,\mathrm{d}t$. *Top panels*: solid blue line is the numerical solution to Fisher's equation, (5.1). The finite element method with linear basis functions was used for the x domain (100 elements) and the backwards Euler formula was used for the time domain ($\Delta t = 0.02$). The dashed red line is U. Here $T = 3$, $N_{cont} = 10$ and $TOL = 10^{-2}$. *Bottom left panel*: region of the domain where the reaction term is included in the reduced model. *Bottom right panel*: $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

very similar if Neumann boundary conditions were applied at one or both boundaries instead of Dirichlet boundary conditions. See section 4.3.1 for a general error bound derivation with a Neumann boundary condition.

Define a quantity of interest,

$$J(\boldsymbol{u}) = \int_0^T \int_a^b \boldsymbol{u} \cdot \boldsymbol{g} \, \mathrm{d}x \, \mathrm{d}t.$$

Suppose that we have a solution to a reduced primal problem, $\boldsymbol{U}(x,t)$. We require that:

1. $\boldsymbol{U}(x,t)$ and $h(\boldsymbol{U},x,t)\dfrac{\partial \boldsymbol{U}(x,t)}{\partial x}$ are continuous for $0 < x < 1$;
2. $\boldsymbol{U}(x,t)$ has identical initial conditions to $\boldsymbol{u}(x,t)$, i.e., $\boldsymbol{U}(x,0) = \boldsymbol{u}(x,0)$.
3. $\boldsymbol{U}(x,t)$ has identical boundary conditions to $\boldsymbol{u}(x,t)$, i.e., $\boldsymbol{U}(a,t) = \boldsymbol{u}(a,t)$, $\boldsymbol{U}(b,t) = \boldsymbol{u}(b,t)$.

Then the adjoint problem is,

$$-\frac{\partial \boldsymbol{\phi}}{\partial t} + \frac{\partial}{\partial x}\left(A\frac{\partial \boldsymbol{\phi}}{\partial x}\right) - A'\frac{\partial \boldsymbol{\phi}}{\partial x} - C\frac{\partial \boldsymbol{\phi}}{\partial x} + F\boldsymbol{\phi} = \boldsymbol{g}, \tag{5.12}$$

where the matrices A, A', F, and C have entries given by,

$$\begin{aligned}
A_{ij}(x,t) &= \int_0^1 h_{ji}(\boldsymbol{\alpha}(s,x,t)) \, \mathrm{d}s \\
A'_{ij}(x,t) &= \int_0^1 \sum_{k=1}^{N} \frac{\partial h_{jk}}{\partial u_i}(\boldsymbol{\alpha}(s,x,t)) \frac{\partial \alpha_k}{\partial x} \, \mathrm{d}s \\
F_{ij}(x,t) &= \int_0^1 \frac{\partial f_j}{\partial u_i}(\boldsymbol{\alpha}(s,x,t)) \, \mathrm{d}s \\
C_{ij}(x,t) &= \int_0^1 \frac{\partial c_j}{\partial u_i}(\boldsymbol{\alpha}(s,x,t)) \, \mathrm{d}s,
\end{aligned}$$

where $\boldsymbol{\alpha}(s,x,t) = s\boldsymbol{u}(x,t) + (1-s)\boldsymbol{U}(x,t)$. We specify the final conditions $\boldsymbol{\phi}(x,T) = \mathbf{0}$, and the boundary conditions $\boldsymbol{\phi}(a,t) = \mathbf{0}$, $\boldsymbol{\phi}(b,t) = \mathbf{0}$.

The matrices A and A' are defined in the same way as in section 4.3.1 so equation (4.17) holds. The matrices F and C are defined in the same way as the matrix A in section 3.1 so equation (3.7) holds. Summarizing this information we have.

$$\sum_{i=1}^{N} A_{ij}\left(\frac{\partial u_i}{\partial x}-\frac{\partial U_i}{\partial x}\right)+\sum_{i=1}^{N} A'_{ij}(u_i-U_i)=\sum_{m=1}^{N} h_{jm}(\boldsymbol{u})\frac{\partial u_m}{\partial x}-\sum_{m=1}^{N} h_{jm}(\boldsymbol{U})\frac{\partial U_m}{\partial x} \tag{5.13}$$

$$\sum_{i=1}^{N} F_{ij}(x,t)(u_i(x,t)-U_i(x,t))=f_j(\boldsymbol{u},x,t)-f_j(\boldsymbol{U},x,t) \tag{5.14}$$

$$\sum_{i=1}^{N} C_{ij}(x,t)(u_i(x,t)-U_i(x,t))=c_j(\boldsymbol{u},x,t)-c_j(\boldsymbol{U},x,t). \tag{5.15}$$

Now we can express the error in the quantity of interest introduced by the approximation $\boldsymbol{U}$ to $\boldsymbol{u}$ as,

$$\begin{aligned}
J(\boldsymbol{u})-J(\boldsymbol{U}) &= \int_0^T\int_a^b (\boldsymbol{u}-\boldsymbol{U})\cdot\boldsymbol{g}\,\mathrm{d}x\,\mathrm{d}t \\
&= \int_0^T\int_a^b (\boldsymbol{u}-\boldsymbol{U})\cdot\left(-\frac{\partial\boldsymbol{\phi}}{\partial t}+\frac{\partial}{\partial x}\left(A\frac{\partial\boldsymbol{\phi}}{\partial x}\right)-A'\frac{\partial\boldsymbol{\phi}}{\partial x}-C\frac{\partial\boldsymbol{\phi}}{\partial x}+F\boldsymbol{\phi}\right)\,\mathrm{d}x\,\mathrm{d}t \\
&= \int_0^T\int_a^b (\boldsymbol{u}-\boldsymbol{U})\cdot\left(-\frac{\partial\boldsymbol{\phi}}{\partial t}-C\frac{\partial\boldsymbol{\phi}}{\partial x}\right)\,\mathrm{d}x\,\mathrm{d}t \\
&\quad+\int_0^T\int_a^b (\boldsymbol{u}-\boldsymbol{U})\cdot\left(\frac{\partial}{\partial x}\left(A\frac{\partial\boldsymbol{\phi}}{\partial x}\right)-A'\frac{\partial\boldsymbol{\phi}}{\partial x}+F\boldsymbol{\phi}\right)\,\mathrm{d}x\,\mathrm{d}t \\
&= I+II,
\end{aligned} \tag{5.16}$$

where we have simply re-arranged the terms so that II is the error of the elliptic terms of PDE, integrated over time, and I is the error of the other terms that we have introduced.

We now analyze term I,

$$\begin{aligned}
I &= \int_0^T\int_a^b \sum_{i=1}^{N}(u_i-U_i)\left(-\frac{\partial\phi_i}{\partial t}-\sum_{j=1}^{N} C_{ij}\frac{\partial\phi_j}{\partial x}\right)\,\mathrm{d}x\,\mathrm{d}t \\
&= \int_0^T\int_a^b \sum_{i=1}^{N}\left(\frac{\partial u_i}{\partial t}-\frac{\partial U_i}{\partial t}\right)\phi_i\,\mathrm{d}x\,\mathrm{d}t-\int_a^b\sum_{i=1}^{N}\Big[(u_i-U_i)\phi_i\Big]_0^T\,\mathrm{d}x \\
&\quad-\int_0^T\int_a^b\sum_{j=1}^{N}\sum_{i=1}^{N} C_{ij}(u_i-U_i)\frac{\partial\phi_j}{\partial x}\,\mathrm{d}x\,\mathrm{d}t,
\end{aligned} \tag{5.17}$$

using integration by parts. We have assumed that $\boldsymbol{u}$, $\boldsymbol{U}$, and $\boldsymbol{\phi}$ are continuous in t. The boundary term is zero because $\boldsymbol{u}(x,0) = \boldsymbol{U}(x,0)$, and $\boldsymbol{\phi}(x,T) = \mathbf{0}$. Now we use identity (5.15) to obtain,

$$\begin{aligned} I &= \int_0^T \int_a^b \sum_{i=1}^N \left(\frac{\partial u_i}{\partial t} - \frac{\partial U_i}{\partial t} \right) \phi_i \, \mathrm{d}x \, \mathrm{d}t - \int_0^T \int_a^b \sum_{j=1}^N (c_j(\boldsymbol{u}) - c_j(\boldsymbol{U})) \frac{\partial \phi_j}{\partial x} \, \mathrm{d}x \, \mathrm{d}t, \\ &= \int_0^T \int_a^b \sum_{i=1}^N \left(\frac{\partial u_i}{\partial t} - \frac{\partial U_i}{\partial t} \right) \phi_i \, \mathrm{d}x \, \mathrm{d}t + \int_0^T \int_a^b \sum_{i=1}^N \frac{\partial}{\partial x} (c_i(\boldsymbol{u}) - c_i(\boldsymbol{U})) \phi_i \, \mathrm{d}x \, \mathrm{d}t, \\ &\quad - \int_0^T \sum_{i=1}^N \Big[(c_i(\boldsymbol{u}) - c_i(\boldsymbol{U})) \phi_i \Big]_a^b \, \mathrm{d}t, \end{aligned} \tag{5.18}$$

using integration by parts again. We have assumed that $\boldsymbol{u}$, $\boldsymbol{U}$, and $\boldsymbol{\phi}$ are continuous in x. The boundary term is zero because $\boldsymbol{\phi}(a,t) = \boldsymbol{\phi}(b,t) = \mathbf{0}$.

For term II we state the result rather than proving it because the analysis is the same as the analysis for nonlinear BVPs in section 4.3.1.

$$\begin{aligned} II = &\int_0^T \int_a^b \sum_{j=1}^N \left(\frac{\partial}{\partial x} \left(\sum_{m=1}^N h_{jm}(\boldsymbol{u}) \frac{\partial u_m}{\partial x} \right) - \frac{\partial}{\partial x} \left(\sum_{m=1}^N h_{jm}(\boldsymbol{U}) \frac{\partial U_m}{\partial x} \right) \right) \cdot \phi_j \, \mathrm{d}x \, \mathrm{d}t \\ &+ \int_0^T \int_a^b \sum_{j=1}^N \left(f_j(\boldsymbol{u}) - f_j(\boldsymbol{U}) \right) \phi_j \, \mathrm{d}x \, \mathrm{d}t. \end{aligned} \tag{5.19}$$

When we calculate $(I + II)$, all terms involving $\boldsymbol{u}$ disappear by definition of the primal problem (5.11). We may therefore write our error bound as,

$$\begin{aligned} &|J(\boldsymbol{u}) - J(\boldsymbol{U})| \leq \\ &\sum_{i=1}^N \sum_{j=1}^M \sum_{k=1}^K \int_{t_{k-1}}^{t_k} \int_{x_{j-1}}^{x_j} \left| \frac{\partial U_i}{\partial t} + \frac{\partial}{\partial x} \left(\sum_{m=1}^N h_{im}(\boldsymbol{U}) \frac{\partial U_m}{\partial x} \right) + \frac{\partial}{\partial x} \Big(c_i(\boldsymbol{U}) \Big) + f_i(\boldsymbol{U}) \right| \Big| \phi_i \Big| \, \mathrm{d}x \, \mathrm{d}t, \end{aligned} \tag{5.20}$$

where $a = x_0 < x_1 \ldots < x_M = b$ is a subdivision of the interval $[a,b]$ and $0 = t_0 < t_1 \ldots < t_K = T$ is a subdivision of the interval $[0,T]$. As for Fisher's equation, the error bound is the product of the left-hand side of the original PDE evaluated at $\boldsymbol{U}$ (i.e., the residual), with the solution to the adjoint equation.

5.3.2 Application to monodomain equation

The derivation above applies to a wide class of PDEs. We have already analyzed Fisher's equation, which is an example of an equation to which the derivation applies. We now apply the derivation to analyze the monodomain equation – a coupled reaction-diffusion PDE that models tissue-level cardiac electrophysiology [40]. The diffusion term in the monodomain equation models electrical conduction, and the reaction terms model the ion channel kinetics and biochemical reactions of cardiac cells. Here we choose to use the FitzHugh-Nagumo equations as the cell model. However other cell models could also be used.

The monodomain equation with the FitzHugh-Nagumo cell model is defined as,

$$\begin{aligned} \chi C_m \frac{\partial V}{\partial t} &= \nabla \cdot (\sigma \nabla V) - \chi f_1(t, \boldsymbol{u}) - I_{stim}(t, x), \\ \frac{\mathrm{d}w}{\mathrm{d}t} &= -f_2(t, \boldsymbol{u}), \end{aligned} \tag{5.21}$$

where $\boldsymbol{u} = (V, w)$ is defined for $0 < x < 1$, $0 < t < T$, C_m represents membrane capacitance, σ is a conductivity tensor, χ is the surface-area-to-volume ratio of a cardiac cell, and $I_{stim}(t, x)$ is an external stimulus. In the simulations below we set $I_{stim}(t, 0) = -10$ for $0 < t < 0.5$, and $I_{stim}(t, x) = 0$ otherwise. We apply the initial condition $\boldsymbol{u}(0, t) = 0$, and the Neumann boundary conditions $\partial V(0, t)/\partial x = 0$, $\partial V(1, t)/\partial x = 0$.

The variable V represents transmembrane potential, and the variable w is termed the recovery variable because it applies negative feedback to V, bringing it back towards its resting potential. The function $\boldsymbol{f}(t, \boldsymbol{u})$ is defined as follows,

$$\begin{aligned} f_1(t, \boldsymbol{u}) &= V(V - \alpha)(1 - V) - w \\ f_2(t, \boldsymbol{u}) &= \epsilon(V - \gamma w). \end{aligned} \tag{5.22}$$

In what follows we set $\alpha = -0.08$, $\gamma = 3.00$, $\epsilon = 0.005$, so that $V = w = 0$ is an unstable steady-state [40], and the system has a stable limit cycle. The solution to equation (5.21) at $t = 5$, $t = 25$, and $t = 50$ is shown in Figure 5.4.

The monodomain equation is of the form (5.11), with

$$\boldsymbol{a}(\boldsymbol{u}, x) = \begin{pmatrix} -\sigma & 0 \\ 0 & 0 \end{pmatrix},$$

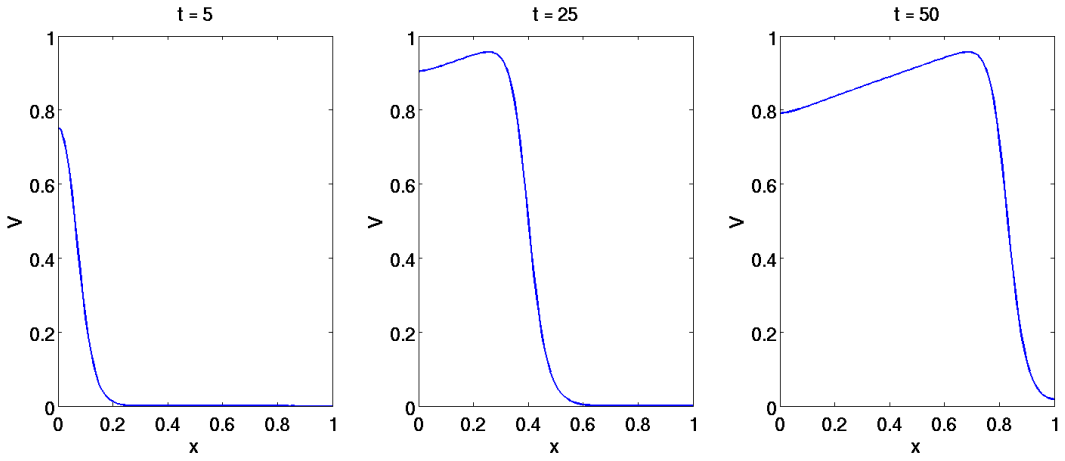


Figure 5.4: Solution to the monodomain equation, (5.21), at selected time-points. Here $\boldsymbol{u}(0,t) = 0$, $\partial V(0,t)/\partial x = 0$, $\partial V(1,t)/\partial x = 0$. The finite element method with linear basis functions was used for the x domain (100 elements) and a time-step of 0.5 was used.

$\boldsymbol{c}(\boldsymbol{u}, x, t) = 0$, and $\boldsymbol{f}(t, \boldsymbol{u})$ as in equation (5.22). Therefore the error bound in equation (5.20) applies. The error bound derivation for the monodomain equation would not be exactly the same as in the previous section, because we applied Neumann boundary conditions to the monodomain equation, and Dirichlet boundary conditions were applied in the error bound. However the end result is the same. For an error bound derivation that includes treatment of a Neumann boundary condition see the derivation in section 4.3.1. For implementing the algorithm, the important difference is that Neumann boundary conditions are applied to the adjoint problem when Neumann boundary conditions are specified in the primal problem.

As for Fisher's equation we choose the quantities of interest to be the average over time and space,

$$J(\boldsymbol{u}) = \frac{1}{T}\int_0^T \int_0^1 V \, \mathrm{d}x \, \mathrm{d}t,$$

and the average over time at a point in space,

$$J(\boldsymbol{u}) = \frac{1}{T}\int_0^T V(x^*, t) \, \mathrm{d}t = \frac{1}{T}\int_0^T \int_0^1 V \, \delta(x - x^*) \, \mathrm{d}x \, \mathrm{d}t.$$

We define the reduced primal problem as follows,

$$\begin{aligned}\chi C_m \frac{\partial U_1}{\partial t} &= \nabla \cdot (\sigma \nabla U_1) - \chi \hat{f}_1(t, \boldsymbol{U}) - I_{stim}(t, x), \\ \frac{\mathrm{d}U_2}{\mathrm{d}t} &= -\hat{f}_2(t, \boldsymbol{U}),\end{aligned} \tag{5.23}$$

where

$$\hat{f}_i(t, \boldsymbol{U}) = \begin{cases} f_i(t, \boldsymbol{U}) & \text{for } (x,t) \in S_i \\ 0 & \text{otherwise,} \end{cases}$$

where S_i is a region of the (x,t) domain to be determined by the algorithm, and $\boldsymbol{U}$ is defined for $0 < x < 1$, $0 < t < T$. We apply the initial conditions and boundary conditions of the original problem: $\boldsymbol{U}(0,t) = \boldsymbol{u}(0,t) = 0$, and $\partial U_1(0,t)/\partial x = \partial V(0,t)/\partial x = 0$, $\partial U_1(1,t)/\partial x = \partial V(1,t)/\partial x = 0$. We require $\boldsymbol{U}$ to be continuously differentiable in x. In the reduced model the term for electrical conduction is present everywhere in the domain, whereas the term for the transmembrane potential cell dynamics is only present in the region S_1, and the term for the recovery variable cell dynamics is only present in the region S_2.

The algorithm is then defined as follows:

Initialization: Set $S_1, S_2 = \emptyset$, so that, in this case,

$$\begin{aligned}\chi C_m \frac{\partial U_1}{\partial t} &= \nabla \cdot (\sigma \nabla U_1) - I_{stim}(t, x), \\ \frac{\mathrm{d}U_2}{\mathrm{d}t} &= 0,\end{aligned} \tag{5.24}$$

for all (x,t). Compute $\boldsymbol{\phi}$ using equation (5.12) with Neumann boundary conditions.

Iteration:

1. Compute `err(1:N, 1:M, 1:K)` where

$$\begin{aligned}\texttt{err}(1,\texttt{j},\texttt{k}) &= \int_{t_{k-1}}^{t_k} \int_{x_{j-1}}^{x_j} \left| C_m \frac{\partial U_1}{\partial t} - \nabla \cdot (\sigma \nabla U_1) + f_1(t, \boldsymbol{U}) + I_{stim}(t,x) \right| |\phi_1| \, \mathrm{d}x \, \mathrm{d}t \\ \texttt{err}(2,\texttt{j},\texttt{k}) &= \int_{t_{k-1}}^{t_k} \int_{x_{j-1}}^{x_j} \left| \frac{\mathrm{d}U_2}{\mathrm{d}t} + f_2(t, \boldsymbol{U}) \right| |\phi_2| \, \mathrm{d}x \, \mathrm{d}t.\end{aligned}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding (x,t) regions to the appropriate S_i.

3. Re-compute $\boldsymbol{U}$ and $\boldsymbol{\phi}$.

Termination: Stop when

$$|J(\boldsymbol{u}) - J(\boldsymbol{U})| < TOL.$$

The algorithm was implemented in the Chaste environment [57]. Although the FitzHugh-Nagumo equations were used as the cell model, the algorithm would also work for other cell models. As for previous examples, different reduced models are generated for different choices of $J(\cdot)$. In Figure 5.5, which shows the application of the algorithm when $J(\boldsymbol{u}) = ave_{|\Omega}(V)$, the average value of V over the whole domain and over time, we see that the only regions where the reaction terms are not included are those that the propagating wave has not yet reached – e.g. regions far away from the location of the initial stimulus at early times. Another feature of the solution that is identified by the algorithm is that the reaction term for the recovery variable, w, is included later than the reaction term for the activation variable, V. This confirms that the reaction term for the recovery variable is not required during the action potential upstroke, i.e., recovery can only happen after activation. So, applying the method to the monodomain equation gives results that are physiologically correct.

In Figure 5.6, which shows the application of the algorithm when $J(\boldsymbol{v}) = ave_{|x=0.3}(V)$, the average value of V over time at the point $x = 0.3$, we see that the reaction terms are required over a smaller region of the domain than in the previous case. This is what we expect because the functional of interest depends on a smaller region of the domain. Since the action potential propagates from $x = 0$ to $x = 1$, the regions nearer $x = 0$ are more important for calculating $J(\boldsymbol{u}) = ave_{|x=0.3}(V)$. This fact is confirmed by the reduced model. It is also interesting to observe in Figure 5.6 that, towards the end of the simulation, the reaction terms are not included in the reduced model in the region near $x = 0$. This is because the action potential in this region does not affect the action potential at $x = 0.3$ before the end of the simulation.

5.3.3 Application to a linear advection-diffusion equation

In the previous section we applied the general error bound derivation from section 5.3.1 to the monodomain equation – a coupled reaction-diffusion equation. The error bound derivation allowed for the PDE to contain a first-order x-derivative term. Therefore in this section we apply the error bound derivation to a linear advection-diffusion equation.

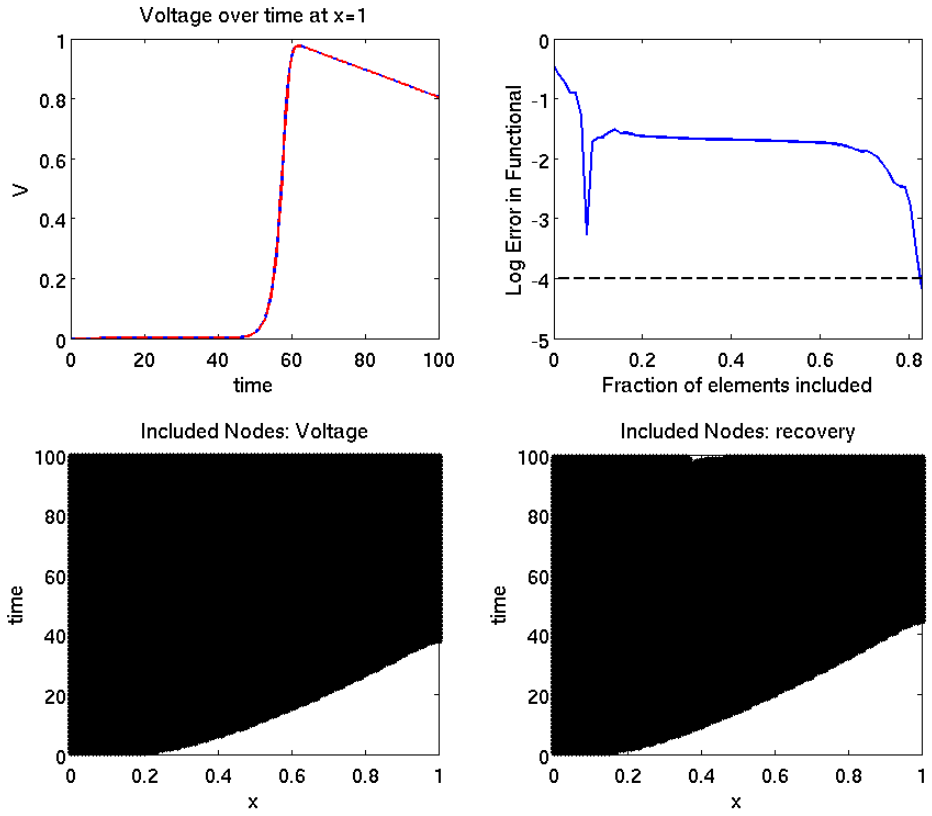


Figure 5.5: Time-dependent diffusion approximation for the monodomain equation, $J(\boldsymbol{u}) = ave_{|\Omega}(V)$. *Top-left panel:* Solid blue line is solution to monodomain equation (5.21) at $x = 1$, with FitzHugh-Nagumo as the cell model, and dashed red line is solution to reduced monodomain equation (5.23). *Top-right panel:* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$ as a function of fraction of intervals included in the reduced model. *Bottom-left panel:* Region where $f_1(\boldsymbol{u}, t)$ is included in the reduced model. *Bottom-right panel:* Region where $f_2(\boldsymbol{u}, t)$ is included in the reduced model. The following parameters were used: $\chi = 1$, $C_m = 1$, $N_{cont} = 50$, $TOL = 10^{-4}$. All equations were solved on a 100 element mesh with the finite element method and a time-step of 0.5 was used.

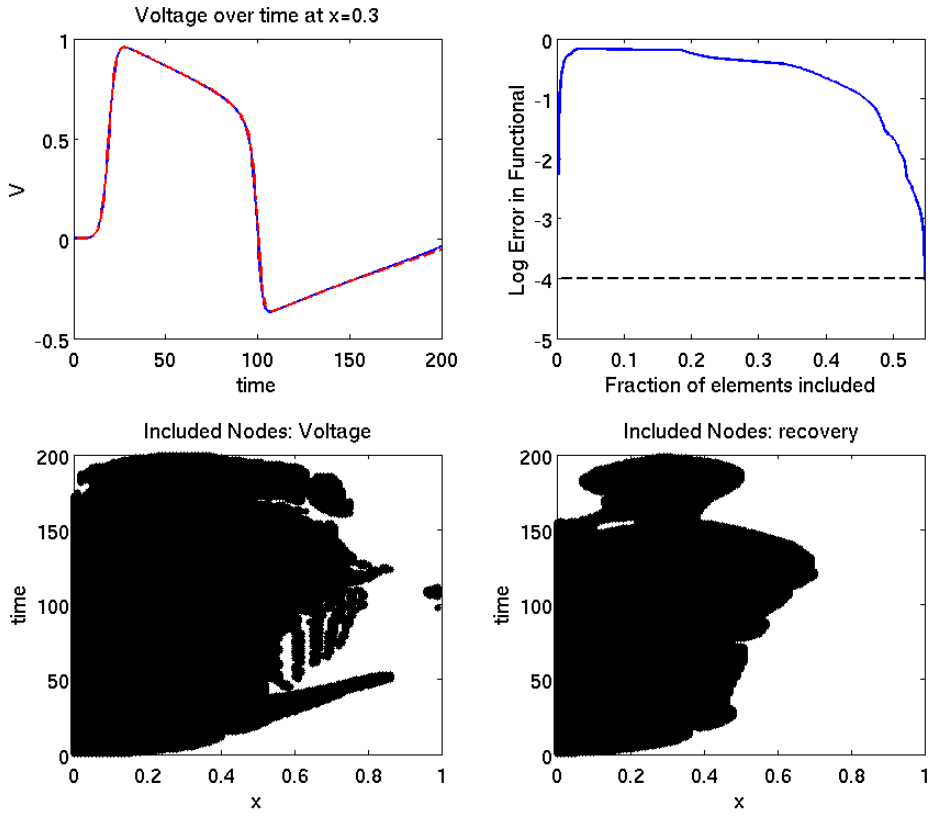


Figure 5.6: Time-dependent diffusion approximation for the monodomain equation, $J(\boldsymbol{u}) = ave_{|x=0.3}(V)$. *Top-left panel:* Solid blue line is solution to monodomain equation (5.21) at $x = 0.3$, with FitzHugh-Nagumo as the cell model, and dashed red line is solution to reduced monodomain equation (5.23). *Top-right panel:* $|J(\boldsymbol{u}) - J(\boldsymbol{U})|$ as a function of fraction of intervals included in the reduced model. *Bottom-left panel:* Region where $f_1(\boldsymbol{u}, t)$ is included in the reduced model. *Bottom-right panel:* Region where $f_2(\boldsymbol{u}, t)$ is included in the reduced model. The following parameters were used: $\chi = 1$, $C_m = 1$, $N_{cont} = 50$, $TOL = 10^{-4}$. All equations were solved on a 100 element mesh with the finite element method and a time-step of 0.5 was used.

We define the primal problem to be,

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = \epsilon \frac{\partial^2 u}{\partial x^2}, \tag{5.25}$$

where $0 < \epsilon \ll 1$ and u is defined for $0 < x < 2\pi$, $0 < t < T$. We apply periodic boundary conditions to u, and the initial condition $u(x, 0) = \sin(x)$. The solution to equation (5.25) at $t = 0.5$, $t = 1$, and $t = 1.5$ is shown in Figure 5.7. The boundary conditions are different from the type of boundary conditions considered in the error bound above. However with periodic boundary conditions, the boundary terms disappear because contributions from each boundary cancel each other out. Periodic boundary conditions are also applied to the adjoint problem.

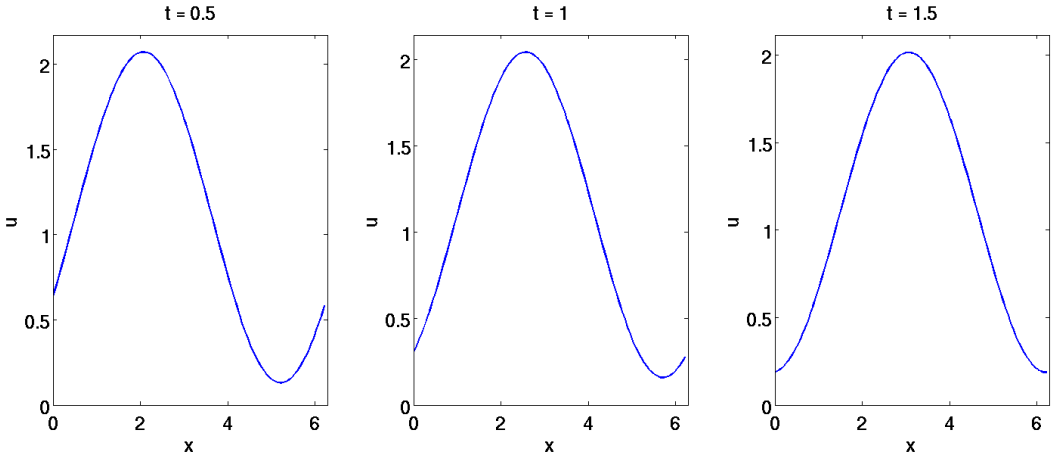


Figure 5.7: Solution to linear advection-diffusion equation, (5.25), at selected time-points. Here $u(x, 0) = \sin(x)$, and u has periodic boundary conditions. The finite element method with linear basis functions was used for the x domain (100 elements) and the backwards Euler formula was used for the time domain ($\Delta t = 0.02$).

We choose the following form for the reduced primal problem,

$$\begin{aligned} \frac{\partial U}{\partial t} + \frac{\partial U}{\partial x} &= \epsilon \frac{\partial^2 U}{\partial x^2} \quad \text{for} \quad (x, t) \in S \\ \frac{\partial U}{\partial t} &= \epsilon \frac{\partial^2 U}{\partial x^2} \quad \text{otherwise,} \end{aligned} \tag{5.26}$$

where S is a region of the (x, t) domain to be determined by the algorithm, and U is defined for $0 < x < 2\pi$, $0 < t < T$. We apply the initial conditions and boundary conditions of the original problem: $U(0, t) = u(0, t) = \sin(x)$, and U has periodic

boundary conditions. We require U to be continuously differentiable in x. In the reduced model the term for diffusion is included everywhere in the domain, whereas the advection term is only included in the region S.

For the results below we choose the quantity of interest to be,

$$J(u) = \frac{1}{T}\int_0^T u(x^*, t)\ \mathrm{d}t = \frac{1}{T}\int_0^T \int_0^{2\pi} u \cdot \delta(x - x^*)\ \mathrm{d}x\ \mathrm{d}t.$$

Note that for this problem, if $J(u)$ was the average of u over x and t, then we would have $|J(u) - J(U)| = 0$ for all choices of S because of the periodic boundary conditions and the absence of a reaction term.

The algorithm is then defined as follows:

Initialization: Set $S = \emptyset$, so that, in this case, $\dfrac{\partial U}{\partial t} = \epsilon \dfrac{\partial^2 U}{\partial x^2}$ for all x. Compute ϕ using equation (5.12).

Iteration:

1. Compute `err(1:N,1:K)` where

$$\texttt{err}(\texttt{j},\texttt{k}) = \int_{x_{j-1}}^{x_j} \int_{t_{k-1}}^{t_k} \left| \frac{\partial U}{\partial t} + \frac{\partial U}{\partial x} - \epsilon \frac{\partial^2 U}{\partial x^2} \right| |\phi|\ \mathrm{d}x\,\mathrm{d}t. \tag{5.27}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding (x, t) regions to S.

3. Re-compute U.

Termination: Stop when

$$|J(u) - J(U)| < TOL.$$

The algorithm was implemented in Matlab. The results are shown in Figures 5.8 and 5.9. The quantity of interest was chosen to be the average of $u(\pi, t)$ over time, and the dominant effect in the problem is advection/wave propagation in the positive x-direction. Therefore, we expect that the advection term is only required in regions of the domain where the solution wave reaches $x = \pi$ before $t = T$. This is indeed the case, indicating that the method has successfully determined the region of the domain that is important for calculating the quantity of interest.

Note we have obtained results for two different domains: $x = [0, 2\pi]$, $t = [0, 0.5]$, and, $x = [0, 2\pi]$, $t = [0, 2]$. Similar results are obtained in both cases - Figures 5.8

and 5.9. In Figure 5.9, which shows the application of the algorithm when $t = [0, 2]$, the included region extends to part of $\pi < x < 2\pi$, because the periodic boundary conditions and the longer time interval enable to solution to travel to $x = \pi$ from this region. Also, for $t = [0, 2]$, there is a characteristic of the solution in the domain of influence with $\partial u/\partial x = 0$ (the maximum of the solution). In Figure 5.9 we see that the advection term is not included in a small neighbourhood of this characteristic. This makes sense because the advection term will be small in that region.

5.3.4 Application to a nonlinear advection-diffusion equation

In the previous section we analyzed a linear advection-diffusion equation. The error bound derivation in section 5.3.1 allowed for the first-order derivative term to be nonlinear. Therefore we apply the error bound to Burgers' equation [7], a model problem from fluid mechanics that contains a nonlinear first-order term. Burgers' equation is defined as,

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = \epsilon\frac{\partial^2 u}{\partial x^2}, \tag{5.28}$$

where $0 < \epsilon \ll 1$ and u is defined for $0 < x < 2\pi$, $0 < t < T$. We apply periodic boundary conditions to u, and the initial condition $u(x, 0) = \sin(x)$. The solution to equation (5.28) is shown in Figure 5.10.

We choose the following form for the reduced primal problem,

$$\begin{aligned} \frac{\partial U}{\partial t} + U\frac{\partial U}{\partial x} &= \epsilon\frac{\partial^2 U}{\partial x^2} \quad \text{for} \quad (x,t) \in S \\ \frac{\partial U}{\partial t} &= \epsilon\frac{\partial^2 U}{\partial x^2} \quad \text{otherwise,} \end{aligned} \tag{5.29}$$

where S is a region of the (x, t) domain to be determined by the algorithm, and U is defined for $0 < x < 2\pi$, $0 < t < T$. We apply the initial conditions and boundary conditions of the original problem: $U(0, t) = u(0, t) = \sin(x)$, and U has periodic boundary conditions. We require U to be continuously differentiable in x. In the reduced model the term for diffusion is included everywhere in the domain, whereas the nonlinear advection term is only included in the region S.

For the results below we choose the quantity of interest to be,

$$J(u) = \frac{1}{T}\int_0^T u(x^*, t)\ \mathrm{d}t = \frac{1}{T}\int_0^T \int_0^{2\pi} u \cdot \delta(x - x^*)\ \mathrm{d}x\ \mathrm{d}t.$$

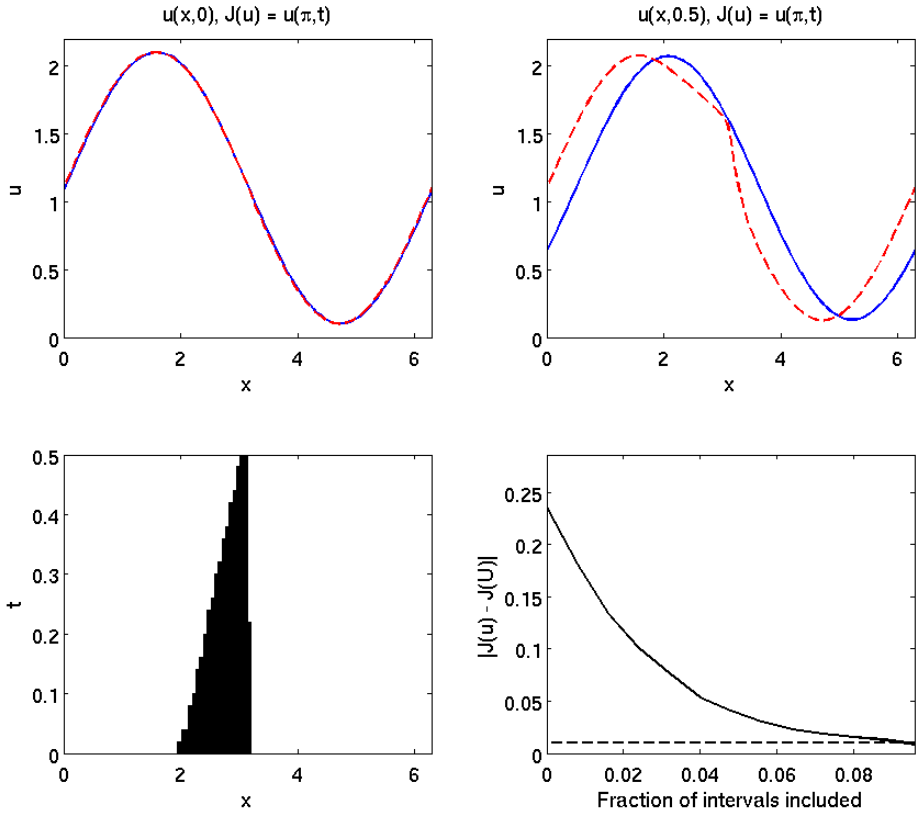


Figure 5.8: Time-dependent diffusion approximation for linear advection-diffusion equation (5.25), $T = 0.5$, $J(u) = \frac{1}{T}\int_0^T u(\pi, t)\ \mathrm{d}t$. Quantity of interest accurately computed with less than 10% of domain included in the reduced model. *Top panels*: Solid blue line is the finite element solution to (5.25) with 100 elements, $\Delta t = 0.02$. Dashed red line is solution to the reduced model, (5.26). Here $\epsilon = 0.05$, $N_{cont} = 20$ and $TOL = 0.01$. *Bottom-left panel:* Region of the domain where advection term is included in the reduced model. *Bottom-right panel:* $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

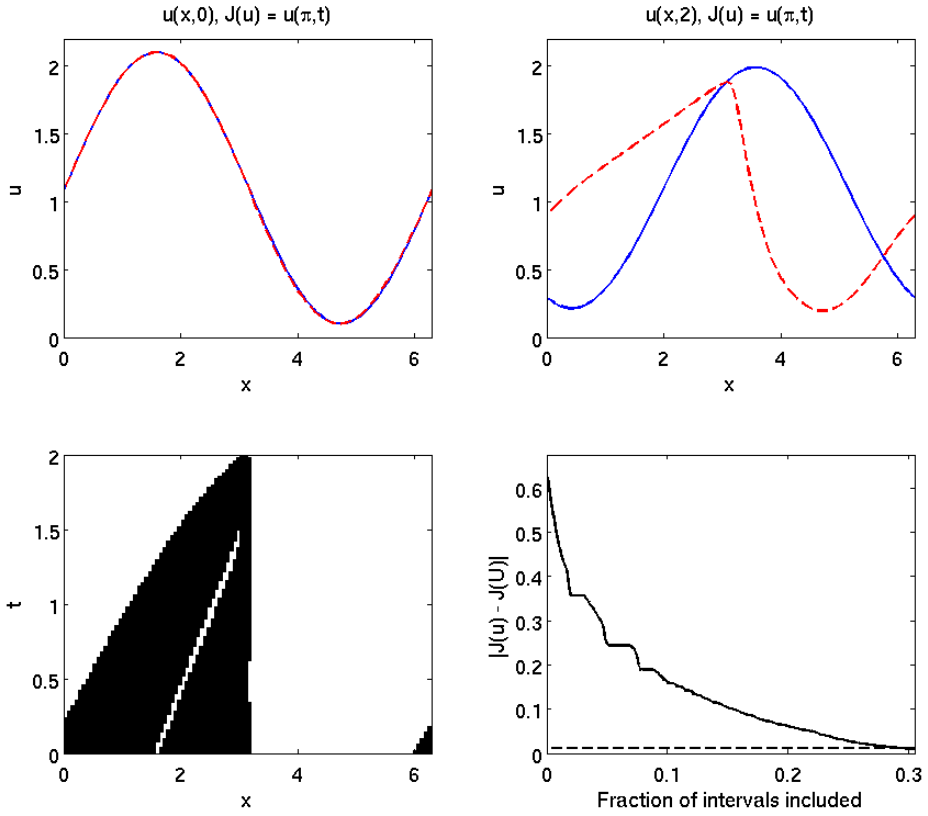


Figure 5.9: Time-dependent diffusion approximation for linear advection-diffusion equation (5.25), $T = 2$, $J(u) = \frac{1}{T}\int_0^T u(\pi, t)\ \mathrm{d}t$. Quantity of interest accurately computed with 32% of domain included in the reduced model. *Top panels*: Solid blue line is the finite element solution to (5.25) with 100 elements, $\Delta t = 0.02$. Dashed red line is solution to the reduced model, (5.26). Here $\epsilon = 0.05$, $N_{cont} = 20$ and $TOL = 0.01$. *Bottom-left panel:* Region of the domain where advection term is included in the reduced model. *Bottom-right panel:* $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

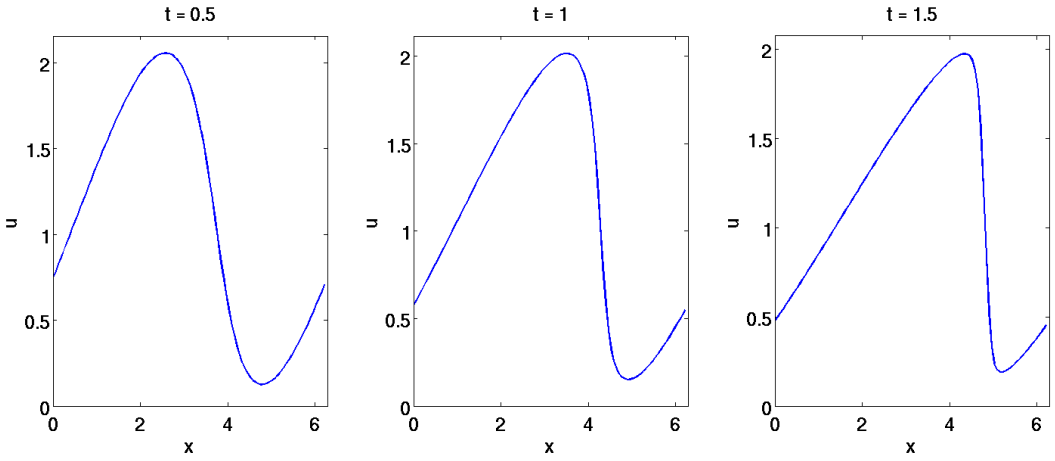


Figure 5.10: Solution to nonlinear advection-diffusion equation, (5.28), at selected time-points. Here $u(x, 0) = \sin(x)$, and u has periodic boundary conditions. The finite element method with linear basis functions was used for the x domain (100 elements) and the backwards Euler formula was used for the time domain ($\Delta t = 0.02$).

Note that as for the linear advection-diffusion PDE, if $J(u)$ was the average of u over x and t, then we would have $|J(u) - J(U)| = 0$ for all choices of S because of the periodic boundary conditions and absence of a reaction term.

The algorithm is then defined as follows:

Initialization: Set $S = \emptyset$, so that, in this case, $\dfrac{\partial U}{\partial t} = \epsilon \dfrac{\partial^2 U}{\partial x^2}$ for all x. Compute ϕ using (5.12).

Iteration:

1. Compute `err(1:N,1:K)` where

$$\texttt{err}(\texttt{j},\texttt{k}) = \int_{x_{j-1}}^{x_j} \int_{t_{k-1}}^{t_k} \left| \frac{\partial U}{\partial t} + U \frac{\partial U}{\partial x} - \epsilon \frac{\partial^2 U}{\partial x^2} \right| |\phi| \, \mathrm{d}x \, \mathrm{d}t. \tag{5.30}$$

2. Identify the largest N_{cont} elements of `err`, and add the corresponding (x, t) regions to S.

3. Re-compute U and ϕ.

Termination: User-defined. (See discussion below.)

The algorithm was implemented in Matlab. The results are shown in Figure 5.11. As in the linear case (section 5.3.3), the quantity of interest was chosen to be the average of $u(\pi, t)$ over time, and the dominant effect in the problem is advection/wave

propagation in the positive x-direction. Therefore, we expect that the advection term is only required in regions of the domain where the solution wave reaches $x = \pi$ before $t = T$. This is indeed the case, indicating that the method has successfully determined the region of the domain that is important for calculating the quantity of interest.

Note we have obtained results for two different domains: $x = [0, 2\pi]$, $t = [0, 0.5]$, and, $x = [0, 2\pi]$, $t = [0, 2]$. Similar results are obtained in both cases - Figures 5.11 and 5.12. In Figure 5.12, which shows the application of the algorithm when $t = [0, 2]$, the included region extends to part of $\pi < x < 2\pi$, as in the linear case, because the periodic boundary conditions and the longer time interval enable to solution to travel to $x = \pi$ from this region. However there is a problem in the case $x = [0, 2\pi]$, $t = [0, 2]$. In this case $J(u) - J(U)$ crosses zero, so the algorithm would stall if the algorithm is terminated with $|J(u) - J(U)| < TOL$. In order to avoid this we ran the algorithm without a stopping condition, and stopped the algorithm manually when $J(u) - J(U)$ was converging on zero (as opposed to when it was passing through zero). A question for further research is how to stop the algorithm automatically when $J(u) - J(U)$ passes through zero.

5.4 Remarks on applying *a posteriori* model reduction to quasi-linear PDEs

The problems we have considered in this chapter have all contained a diffusion term. The presence of a diffusion term simplifies the mathematics and the numerics of the problem because a second-order derivative introduces two arbitrary constants in to the problem. These constants can be used to enforce continuous differentiability in x of the model solution when the time-dependent diffusion approximation is applied in the reduced model. If there is no second-order derivative in the PDE, then some choices of reduced model lead to a solution that is a multi-valued in some regions of the domain and undefined in other regions of the domain. This difficulty can be overcome for some PDEs, generally termed quasilinear PDEs, by using the method of characteristics to convert the PDE to a system of ODEs.

We consider the following linear hyperbolic problem,

$$\frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} = 0, \quad \text{for } 0 < x < 2\pi, \tag{5.31}$$

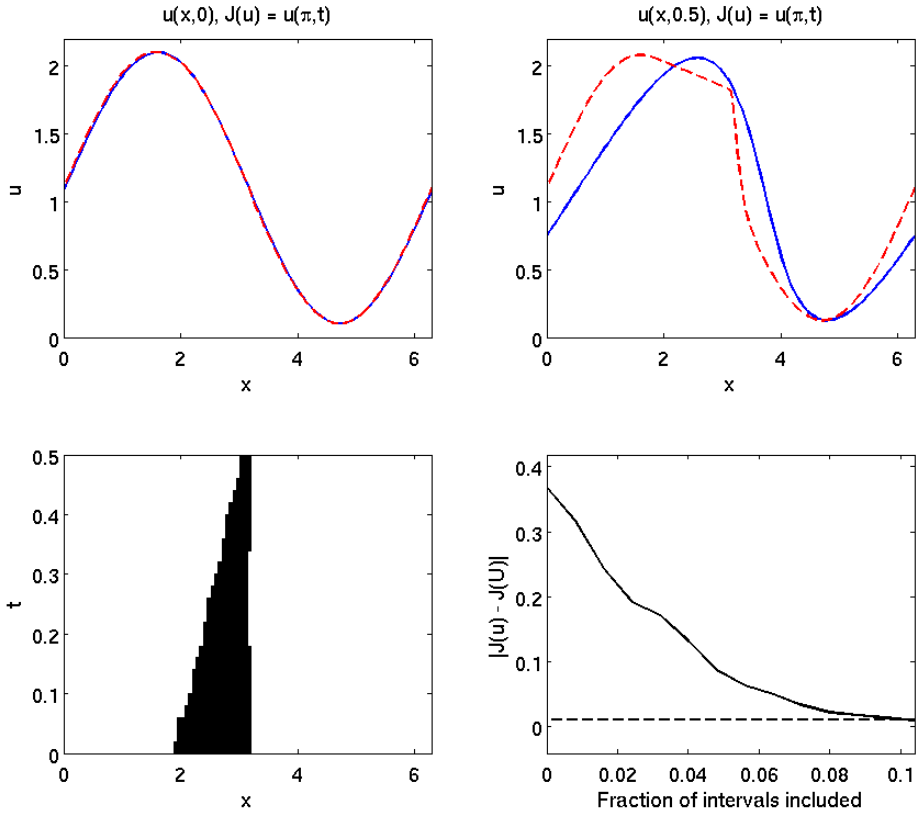


Figure 5.11: Time-dependent diffusion approximation for nonlinear advection-diffusion equation (5.28), $T = 0.5$, $J(u) = \frac{1}{T}\int_0^T u(\pi, t)\,\mathrm{d}t$. Quantity of interest accurately computed with 10.5% of domain included in the reduced model. *Top panels*: Solid blue line is the finite element solution to (5.28) with 100 elements, $\Delta t = 0.02$. Dashed red line is solution to the reduced model, (5.29). Here $\epsilon = 0.05$, $N_{cont} = 20$ and $TOL = 0.01$. *Bottom-left panel:* Region of the domain where advection term is included in the reduced model. *Bottom-right panel:* $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

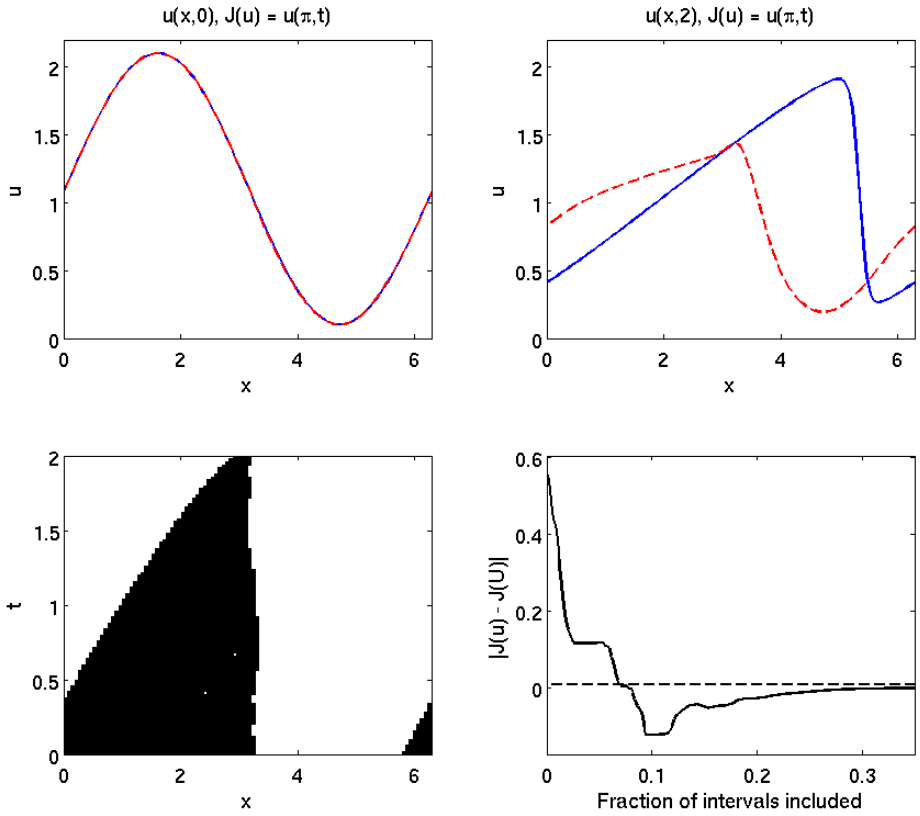


Figure 5.12: Time-dependent diffusion approximation for nonlinear advection-diffusion equation (5.28), $T = 2$, $J(u) = \frac{1}{T}\int_0^T u(\pi, t)\ \mathrm{d}t$. Quantity of interest accurately computed with 31% of domain included in the reduced model. *Top panels*: Solid blue line is the finite element solution to (5.28) with 100 elements, $\Delta t = 0.02$. Dashed red line is solution to the reduced model, (5.29). Here $\epsilon = 0.05$, $N_{cont} = 50$. *Bottom-left panel:* Region of the domain where advection term is included in the reduced model. *Bottom-right panel:* $|J(u) - J(U)|$, as a function of the fraction of the domain included in the reduced model.

with the initial condition $u(x,0) = \sin(x)$ and periodic boundary conditions. The solution to this problem is $u(x,t) = \sin(x-t)$. Suppose we define a reduced model as follows,

$$\begin{aligned} \frac{\partial u}{\partial t} + \frac{\partial u}{\partial x} &= 0, \quad \text{for } 0 < x < x^*, \\ \frac{\partial u}{\partial t} &= 0, \quad \text{for } x^* < x < 2\pi, \end{aligned}$$

with the initial condition $u(x,0) = \sin(x)$ and periodic boundary conditions. Using the method of characteristics gives $u(x,t) = \sin(x-t)$ in the set $\{(x,t) : 0 < x < 2\pi \text{ and } x - x^* < t < x\}$, and $u(x,t) = \sin(x)$ in the set $\{(x,t) : x^* < x < 2\pi\}$. These two regions overlap with each other and the two solutions do not agree with each other in the overlapping regions. Therefore the solution is multi-valued. Also in the set $\{(x,t) : 0 < x < x^* \text{ and } x < t < (x + 2\pi)\}$ the solution is undefined because no characteristics pass through this region.

The choice of reduced model above is not appropriate. In order to apply *a posteriori* model reduction to equation (5.31), we could use the method of characteristics to convert it to a system of initial-value ODEs [7, 25]. The method of characteristics can be applied to equations of the form,

$$P(u,x,t)\frac{\partial u}{\partial t} + Q(u,x,t)\frac{\partial u}{\partial x} = R(u,x,t), \tag{5.32}$$

termed quasi-linear PDEs. *A posteriori* model reduction can be applied using the theory developed in Chapter 3 for IVPs to such equations after the method of characteristics has been applied.

5.5 Robustness of *a posteriori* model reduction to algorithm parameters

For all the problems considered in this chapter the algorithm parameter N_{cont} can be increased from 1 to 20 or more with negligible changes to the results. In Figures 5.13 and 5.14 we show the results of varying N_{cont} for the linear advection-diffusion and nonlinear advection-diffusion equations. In Figures 5.15 and 5.16, we show what happens to the results for the monodomain equation when N_{cont} is increased to a point where the results are considerably different. In the previous chapter we gave a rule

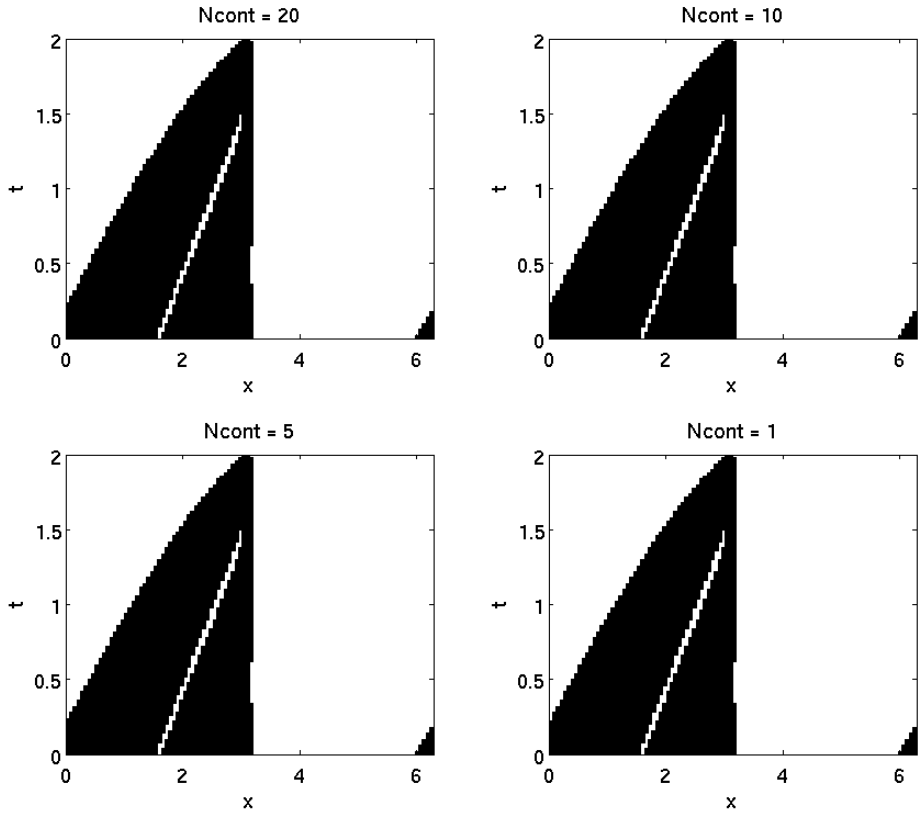


Figure 5.13: *A posteriori* model reduction with time-dependent diffusion approximation is robust to changes in N_{cont} parameter for linear advection-diffusion equation, $T = 2$. As in Figure 5.9, the black area represents the region of the domain where advection term is included in the reduced model. Apart from N_{cont}, all other parameter values are the same as in Figures 5.9.

of thumb that $MaxIter$ – the maximum number of iterations that the algorithm can take should be at least 100. Figures 5.15 and 5.16 support this rule of thumb. For the parameters we have chosen there are 80,800 elements in the computational domain that can be included in the reduced model. In order to make sure that $MaxIter$ is more than 100, N_{cont} has to be less than 808. Figures 5.15 and 5.16 show that the results are relatively similar for $N_{cont} = 50$ and $N_{cont} = 100$, but become increasingly different for $N_{cont} = 500$ and $N_{cont} = 1000$. Even when $N_{cont} = 1000$, it could be argued that the results are qualitatively the same as they are for $N_{cont} = 50$. This supports the recommendation of ensuring that $MaxIter$ is at least 100.

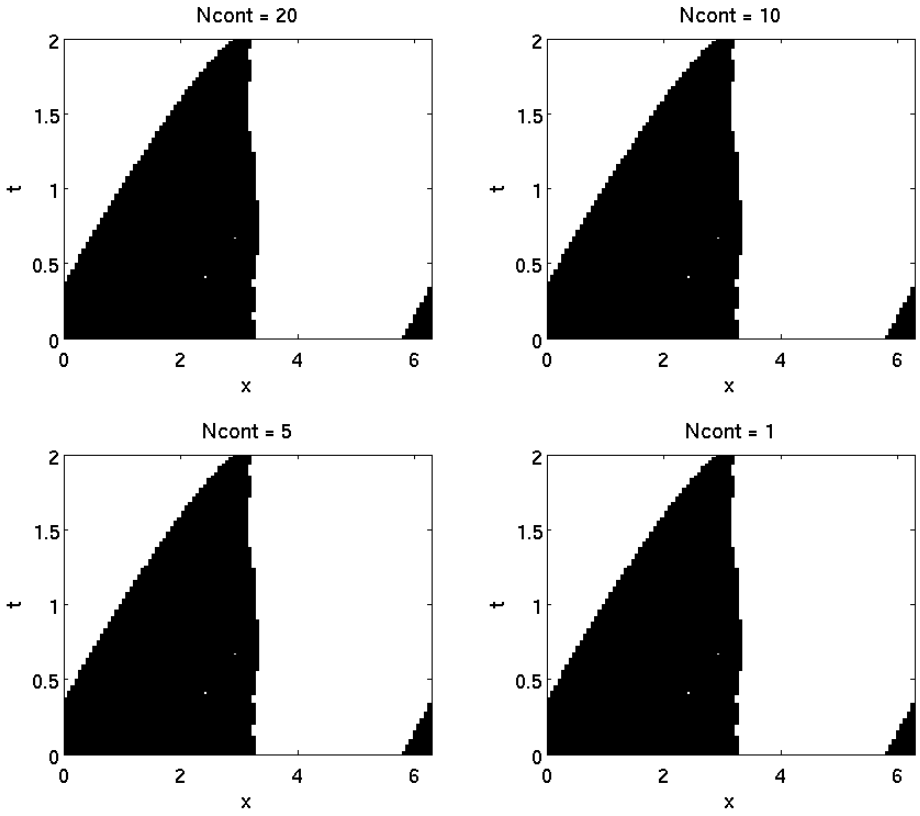


Figure 5.14: *A posteriori* model reduction with time-dependent diffusion approximation is robust to changes in N_{cont} parameter for nonlinear advection-diffusion equation, $T = 2$. As in Figure 5.12, the black area represents the region of the domain where advection term is included in the reduced model. The termination step was changed so that in each case the algorithm stopped after the advection term was included in 3,500 elements of the domain. In each case $|J(u) - J(U)|$ is less than $2 \cdot 10^{-4}$. All other parameter values apart from N_{cont} are the same as in Figures 5.12.

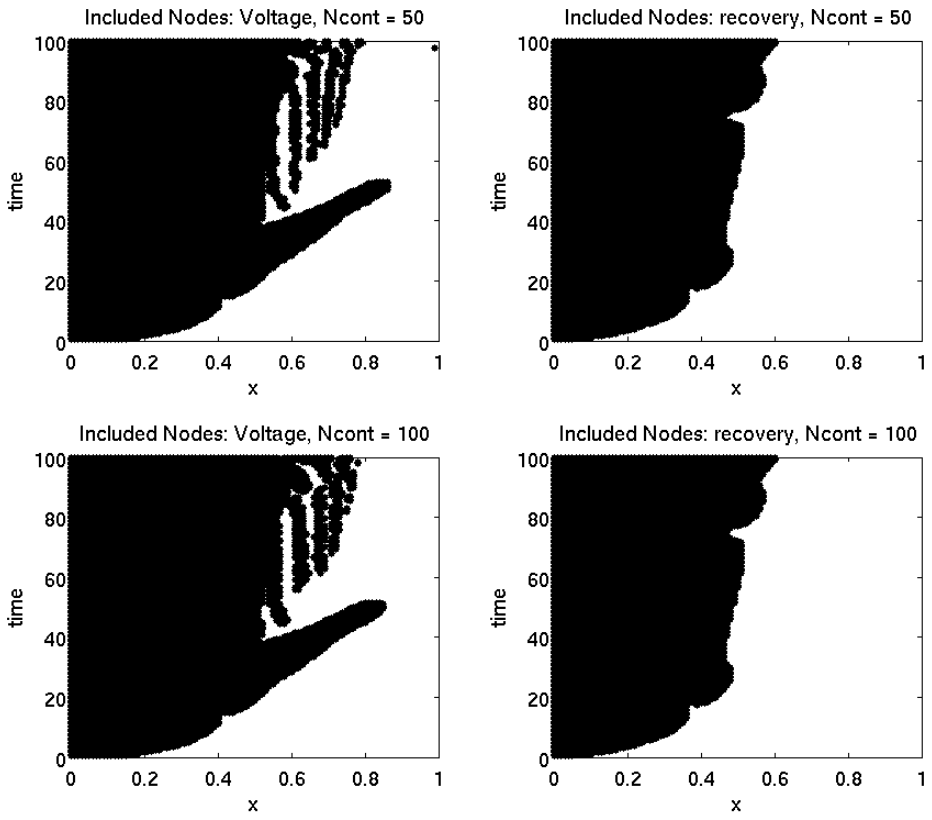


Figure 5.15: *A posteriori* model reduction with time-dependent diffusion approximation is robust to changes in N_{cont} parameter for monodomain equation, $J(\boldsymbol{u}) = ave_{|x=0.3}(V)$. As in Figure 5.6, the black area represents the region of the domain where reaction term is included in the reduced model. Apart from N_{cont}, all other parameter values are the same as in Figure 5.6.

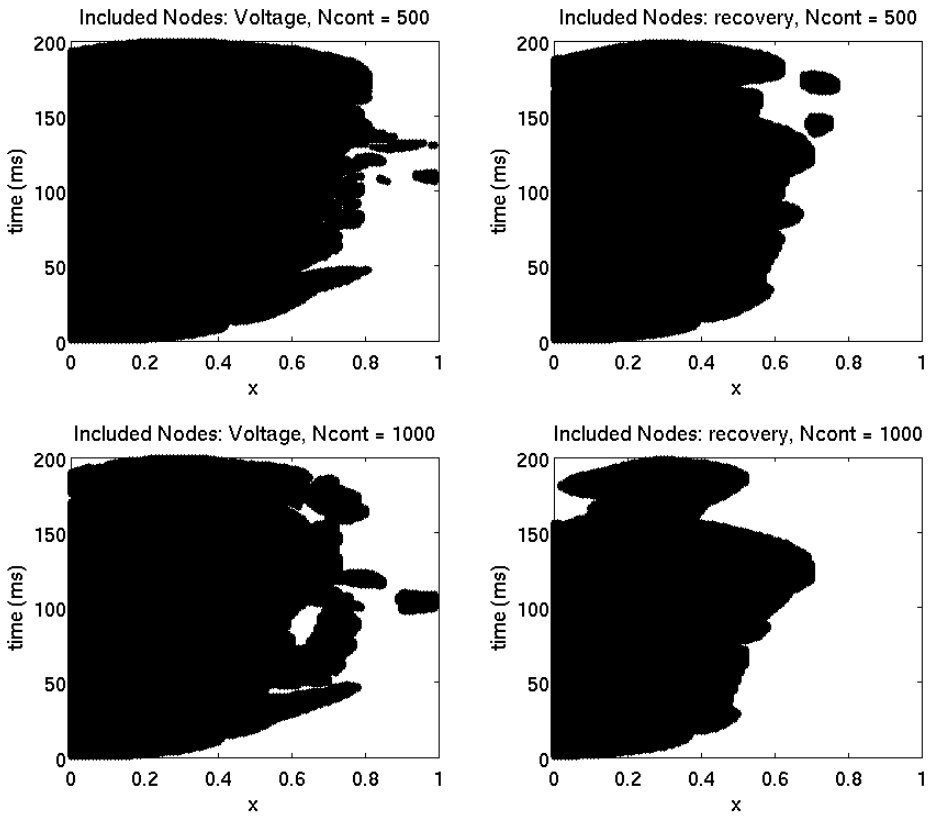


Figure 5.16: A sufficiently large increase in N_{cont} causes *a posteriori* model reduction to give different results for the monodomain equation with time-dependent diffusion approximation, $J(\boldsymbol{u}) = ave_{|x=0.3}(V)$. As in Figure 5.6, the black area represents the region of the domain where reaction term is included in the reduced model. All other parameter values apart from N_{cont} are the same as in Figure 5.6.

5.6 Computational efficiency

The algorithm used to obtain the results for the monodomain equation – Figures 5.5 and 5.6 – currently takes 15-20 minutes to run. When the functional of interest is the average transmembrane potential at a particular node, the algorithm takes less time. This is because a smaller region of the domain has to be included in order to accurately compute the quantity of interest, compared to the case when the quantity of interest is the average transmembrane potential over the whole domain.

It is possible to estimate the computational feasibility of applying the algorithm to problems with more degrees of freedom. For simplicity, we restrict ourselves to considering the monodomain equation in one spatial dimension with two components. However, this is sufficient to see that it may be difficult to apply the method to problems in two spatial dimensions. The algorithm includes a fixed number of sub-regions on each iteration. This causes an approximately linear increase in the computation time with the number of nodes in the mesh. The computation time for each iteration increases approximately quadratically with the number of nodes. This is because the reduced monodomain equation and an adjoint problem are solved on each iteration, and the rate-limiting step of solving these equations is in solving a linear system of equations. Multiplying these factors together tells us that there is an approximately cubic increase in computation time with the number of degrees of freedom in the original problem. We estimate that applying the method to 2-D problems would require over a month of simulation time, unless the current implementation can be made faster. This information is summarised in table 5.1.

Although our implementation is relatively fast, there are ways that it can be made faster. The algorithm is implemented in Chaste [57], an efficient C++ environment that is designed for solving large-scale cardiac and other tissue-level problems. A faster implementation would be obtained through using the C or Fortran languages, since features of C++ such as virtual redirection slow down the implementation. Also, the Chaste solvers calculate the solution on a uniform mesh with uniform time-stepping. More sophisticated numerical solvers, e.g., using higher-order polynomials adaptively in the FEM [2], offer a further potential speed-up. These suggestions may be more time-consuming for the user to implement than the implementation used to generate the results in this thesis.

A 10-20 fold increase in computational efficiency can be obtained by increasing the algorithm parameter N_{cont} (the number of elements added to the included region on each iteration of the algorithm). As shown in the previous section, for the problems in this chapter, the change in the results is relatively small for increases of N_{cont} from 1 to 20.

Table 5.1: Current computation times with parameter settings as above: $T = 200ms$, $dt = 0.5ms$, $N_{cont} = 50$, FitzHugh Nagumo cell model.

	Time required to do 1 iteration	Maximum number of iterations	Maximum amount of time required
100 nodes	0.90s	$8 \cdot 10^4$	24 minutes
1,000 nodes	4.82s	$8 \cdot 10^5$	21.5 hours
10,000 nodes	25s	$8 \cdot 10^6$	46.3 days

5.7 Summary

In this chapter we combined the results for IVPs and BVPs to derive an error bound for PDEs with a linear first-order time derivative term and algebraic, first order and second order x-derivative terms, any combination of which may be nonlinear. We applied *a posteriori* model reduction to two reaction-diffusion PDEs (Fisher's equation and the monodomain equation), and two advection-diffusion PDEs (a linear model problem and Burger's equation). The quantities of interest we considered were the average over time and space for a component of the model solution, and the average over time at a single point in space for a component of the model solution.

Reduced models that accurately calculate an average over time and space were generated for Fisher's equation and the monodomain equation. (Reduced models for the advection-diffusion equations were not generated for this type of quantity of interest because the average over time and space was the same regardless of whether the advection term was included.) Both Fisher's equation and the monodomain equation have travelling wave solutions. In results for Fisher's equation the reaction term is included in the reduced model only on the wave-front. The reason the reaction term is not included behind the wave-front is that the solution is close to a steady state in this region, and the reaction term is zero at the steady state. Similarly for the region ahead of the wave-front. In results for the monodomain equation the reaction term is included

in the reduced model on or behind the wave-front. The recovery variable reaction term is included in the reduced model later than the activation variable recovery term, indicating that the recovery variable reaction term is only required after activation has taken place. The reason the reaction term is included behind the wave-front is that the solution tends towards a stable limit cycle in this region, and both reaction terms are non-zero on the limit cycle.

Reduced models that accurately calculate an average over time at a single point in space were generated for all four of the PDEs analyzed in this chapter. In results for Fisher's equation, the reaction term was included in a band that travelled across the domain with the wave-front until it reached $x = x^*$. For the monodomain equation the reaction terms are required over a larger region of the domain. In both cases, the included region is smaller than when the quantity of interest is an average over time and space, indicating that less information from the original problem is required when the domain of integration for the quantity of interest is restricted to a point in space. For the two advection-diffusion equations, results show that the method identifies the domain of influence in the problem, i.e., the advection term is only included in regions of the domain where the solution wave reaches the point of interest before $t = T$. For larger values of T the advection term is included over a greater region of the domain as the solution wave travels further during the simulation.

An issue that we faced in applying the method to Burger's equation was that $J(u) - J(U)$ passes through 0 before it converges on zero. This is not a problem if the user is only interested in finding a model that accurately calculates the quantity of interest. However, when an average is chosen as a quantity of interest the user will often be interested in accurately reproducing the solution of the original problem over the entire region over which the average is taken. For Burger's equation the method finds a reduced model such that $J(u) - J(U)$ is close to zero but the solution to the reduced model is inaccurate almost everywhere on the computational domain. This is because large positive and negative contributions to $J(u) - J(U)$ cancel each other out. We resolved this issue for Burger's equation by removing the termination condition, $|J(u) - J(U)| < TOL$, and instead running the algorithm for a fixed number of iterations, saving data on included regions periodically. This made it possible to manually find out when $J(u) - J(U)$ starts converging on zero. An area for further work would be

to implement the algorithm with an improved stopping criterion to avoid premature termination of the algorithm.

In this chapter we have considered a single type of reduced model – the time-dependent diffusion approximation. In this case the reduced model contains only a time derivative and a diffusion term in regions of the domain where the original differential operator is not applied. An area for further work is to consider different types of approximation for PDEs. For reaction-diffusion PDEs, a straightforward extension of the reaction approximation method introduced in Chapter 4 would give a reduced model where the diffusion term is only included in automatically identified regions of the domain. Similarly, for advection-diffusion PDEs an alternative approximation could be made by only including the diffusion term in automatically identified regions of the domain. There are a couple of potential issues that need to be considered in this case. First, if we are solving a reduced model where an advection-diffusion equation is solved in one region, and an advection equation is solved elsewhere, it will not be possible to ensure that the solution to the reduced model is continuously differentiable. As in the case of the reaction approximation, this is because there are not enough arbitrary constants in the problem to enforce continuous differentiability. Second, the region where the advection equation is solved would have to coincide with characteristics of the advection equation. If the boundary between the included and non-included region was parallel to the t-axis, as it is in all of the reduced models in this chapter, then the solution to the reduced model would not in general be well-defined.

The work in this chapter could be continued by analyzing a time-dependent form of the chemotaxis equations from section 4.3.3 and [27]. Another potential application area is PDE models of crawling cells [43, 55]. The models in [43, 55] couple hyperbolic, parabolic and elliptic PDEs. The error bound derived in section 5.3.1 can be applied to both the chemotaxis and the crawling cell models, and many other PDEs in systems biology that take account of effects such as diffusion, chemical kinetics, wave propagation, and viscosity.

6

Discussion

The aim of this thesis is to contribute to the development of the *a posteriori* model reduction methodology for model analysis in systems biology. Mathematical modelling in biology promises to deliver predictive models of physiological and pathophysiological systems. However a major challenge for achieving this goal is that the dynamic behaviour of many mathematical models in systems biology is in general not very well understood. One reason for this is that mathematical models in systems biology are increasingly large-scale and multiphysics. This means that it is difficult to apply many existing methods for model analysis that have been developed in other areas of science. Two approaches that have been successful for model analysis in systems biology are sensitivity analysis and perturbation methods. Sensitivity analysis is suited to answering questions such as – "Which parameter values in the model equations have the greatest effect on a model output of interest?" However, as discussed in Chapter 2, the question – "Which terms in the model equations have the greatest effect on a model output of interest?" – has not been answered using sensitivity analysis methods for systems biology models. Perturbation methods can answer the second of these questions. However perturbation methods require specialist mathematical skills to apply. Furthermore perturbation methods become increasingly unwieldy as the number of components in the model increases. There is therefore a need for a methodology that applies to large-scale multiphysics models and can answer the question – "Which terms in the model equations have the greatest effect on a model output of interest?" A relatively new method, *a posteriori* model reduction, is able to answer this question through application of a computational algorithm. The method can be used with limited mathematical

knowledge, and is suited for application to large-scale multiphysics models.

Previous work on *a posteriori* model reduction has developed the method for removing algebraic terms from nonlinear coupled IVPs in regions of the domain where those terms are not required for accurately computing some user-defined quantity of interest [75]. In this thesis we have developed the method to automatically remove,

- the first-order derivative term from nonlinear coupled IVPs resulting in a quasi-steady state approximation,
- either the algebraic or the second-order derivative terms from nonlinear coupled BVPs,
- either algebraic or first-order x-derivative terms from time-dependent nonlinear coupled PDEs,

in regions of the domain where those terms are not required for accurately computing some user-defined quantity of interest.

We have applied the new variants of the method to a number of model problems including the Michaelis-Menten equations – a nonlinear ODE system; a chemotaxis PDE model containing a nonlinear second-order term; the monodomain equation – a time-dependent coupled reaction-diffusion equation with nonlinear reaction term; and Burger's equation – a nonlinear advection-diffusion equation. Summaries of the results for all model problems can be found at the end of Chapters 3, 4, and 5.

An overall conclusion that can be drawn in this thesis is that the method automatically constructs informative reduced models for a wide range of problems. The method seems to work better for linear problems in the sense that for some of the nonlinear problems analyzed, such as the chemotaxis model, we expected the included region of the domain to be smaller. Understanding why this is the case is an area for further work.

A feature of the method that has been preserved in the development of the method in this thesis is that it is automatic. The user has to specify the quantity of interest and the choice of the reduced model. Because of the simplicity of the algorithm, almost all of the user's work is in solving the primal problem, and the adjoint problem as it has been defined in this thesis. If the user is able to accurately calculate the solution to

the primal problem, solving the adjoint problem is also possible and no more difficult technically than solving the primal problem.

Another area for further work would be to extend the automaticity of the method so that the choice of reduced model is automatically generated. For example many IVPs in systems biology contain a number of different reaction terms. It may be desirable to identify which of these reaction terms is important for accurately calculating $J(\cdot)$. This could be done, for example, by further decomposing the expression for the error into contributions from each of the individual reaction terms. Note that automatically generating the choice of reduced models for BVPs and PDEs that contain several different types of term may be more challenging, as it can be difficult to identify whether the reduced model will always have a solution.

The question of computational efficiency is a pertinent one as the method scales approximately cubicly with the number of elements in the computational domain. This would make it difficult to apply the method to problems with more than two independent variables, unless an order of magnitude decrease in the computation time for the differential equation solvers was obtained. This may be possible by using C or Fortran, and or by using more sophisticated numerical solvers, e.g., using higher-order polynomials adaptively in the FEM [2]. We have investigated whether the method can be made more computationally efficient by increasing the algorithm parameter N_{cont} (the number of elements added to the included region on each iteration of the algorithm). Results show that the effect of increasing N_{cont} by a factor of 10 or 20 has a negligible effect for the the IVPs and majority of BVPs and PDEs considered in this thesis. However an exception to this is the chemotaxis model where differences were observed between resulting reduced models for different values of N_{cont}. This warrants further investigation.

The areas for further work outlined above aim to extend the applicability of the method to a wider range of problems. Another important area for further work is to apply the method in its existing form to other models that are currently not well-understood in order to gain insight into the importance of different terms in the equations, and therefore potential insight into the important physical effects in the system that the model is approximating. An obvious starting place would be to apply the method to other large-scale IVPs. Previously the method has been applied to a large-scale cardiac electrophysiology model [75]. There are many other IVPs that could be

analyzed, for example [10, 22, 32, 45, 53]. The variant of the method that makes a quasi-steady state approximation is novel, and has thus far only been applied to the two model problems presented in Chapter 3. There are also many mixed-type PDEs in systems biology to which the general error bound derived in Chapter 5 can be applied. For example, there are models for cell motility that couple hyperbolic, parabolic and elliptic PDEs [43, 55]. The quantity of interest in these models is the speed at which the cell moves, and the equations are of the general form of the PDEs in Chapter 5. These equations are therefore an obvious candidate for application of the method. Applying *a posteriori* model reduction to any of these problems can be done using the algorithms developed in this thesis.

Summary

Model analysis in systems biology is important for development of detailed biophysical models that accurately reproduce the underlying physiology. Because models in systems biology are increasingly large-scale and multiphysics, model analysis poses a significant challenge. The long term aim of this project is the development of the *a posteriori* model reduction methodology for identifying which terms in a model have the greatest effect on a model output of interest, and where in the computational domain these terms are required in the model. The contributions of this thesis have been to further develop the methodology for IVPs and to extend the methodology to BVPs and PDEs. We have considered differential equations with algebraic terms, first order terms, and second order terms any combination of which may be nonlinear. We have given several examples of how a reduced model can be constructed such that any one of these terms is removed in automatically identified regions of the domain. The methodology can now be applied to other problems that are less well understood than the model problems analyzed in this thesis. Moreover there is scope to further develop the method to widen its applicability.

References

[1] J. Anderson, Y.C. Chang, and A. Papachristodoulou. Model decomposition and reduction tools for large-scale networks in systems biology. *Automatica*, 47(6):1165–1174, 2011.

[2] C.J. Arthurs, M.J. Bishop, and D. Kay. Efficient simulation of cardiac electrical propagation using high order finite elements. *Journal of Computational Physics*, 231(10), 2012.

[3] I. Babuška and W.C. Rheinboldt. Error estimates for adaptive finite element computations. *SIAM Journal on Numerical Analysis*, 15(4):736–754, 1978.

[4] I. Babuška and W.C. Rheinboldt. *A posteriori* error estimates for the finite element method. *International Journal for Numerical Methods in Engineering*, 12(10):1597–1615, 1978.

[5] R. Becker and R. Rannacher. An optimal control approach to *a posteriori* error estimation in finite element methods. *Acta Numerica*, 10:1–102, 2001.

[6] O. Bernus, R. Wilders, C.W. Zemlin, H. Verschelde, and A.V. Panfilov. A computationally efficient electrophysiological model of human ventricular cells. *American Journal of Physiology – Heart and Circulatory Physiology*, 282(6):H2296, 2002.

[7] J. Billingham and A.C. King. *Wave Motion*, volume 24 of *Cambridge Texts in Applied Mathematics*. Cambridge University Press, 2000.

[8] G.E. Briggs and J.B.S. Haldane. A note on the kinetics of enzyme action. *Biochemical Journal*, 19(2):338, 1925.

[9] M. Buzby, D. Neckels, M.F. Antolin, and D. Estep. Analysis of the sensitivity properties of a model of vector-borne bubonic plague. *Journal of the Royal Society Interface*, 5(26):1099–1107, 2008.

[10] K.C. Chen, L. Calzone, A. Csikasz-Nagy, F.R. Cross, B. Novak, and J.J. Tyson. Integrative analysis of cell cycle control in budding yeast. *Molecular Biology of the Cell*, 15(8):3841, 2004.

[11] M. Courtemanche, W. Skaggs, and A.T. Winfree. Stable three-dimensional action potential circulation in the FitzHugh-Nagumo model. *Physica D: Nonlinear Phenomena*, 41(2):173–182, 1990.

[12] G.E. Dullerud and F.G. Paganini. *A Course in Robust Control Theory: A Convex Approach*, volume 36 of *Texts in Applied Mathematics*. Springer Verlag, 2000.

[13] H.C. Elman, D.J. Silvester, and A.J. Wathen. *Finite Elements and Fast Iterative Solvers: with Applications in Incompressible Fluid Dynamics*. Oxford University Press, USA, 2005.

[14] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. Introduction to adaptive methods for differential equations. *Acta Numerica*, 4:105–158, 1995.

[15] K. Eriksson, D. Estep, P. Hansbo, and C. Johnson. *Computational Differential Equations*. Cambridge University Press, 1996.

[16] D. Estep and D. Neckels. Fast and reliable methods for determining the evolution of uncertain parameters in differential equations. *Journal of Computational Physics*, 213(2):530–556, 2006.

[17] D. Estep and D. Neckels. Fast methods for determining the evolution of uncertain parameters in reaction-diffusion equations. *Computer Methods in Applied Mechanics and Engineering*, 196(37-40):3967–3979, 2007.

[18] D.A. Fell. *Understanding the Control of Metabolism.* Portland Press Ltd., 1997.

[19] R.J. Field and R.M. Noyes. Oscillations in chemical systems. IV. Limit cycle behavior in a model of a real chemical reaction. *Journal of Chemical Physics*, 60:1877–1884, 1974.

[20] R. Fitzhugh. Thresholds and plateaus in the Hodgkin-Huxley nerve equations. *The Journal of General Physiology*, 43(5):867–896, 1960.

[21] R. Fitzhugh. Impulses and physiological states in theoretical models of nerve membrane. *Biophysical Journal*, 1(6):445–466, 1961.

[22] D.B. Forger and C.S. Peskin. A detailed predictive model of the mammalian circadian clock. *Proceedings of the National Academy of Sciences USA*, 100(25), 2003.

[23] M.B. Giles and E. Süli. Adjoint methods for PDEs: *a posteriori* error analysis and postprocessing by duality. *Acta Numerica*, 11:145–236, 2002.

[24] K. Glover. All optimal hankel-norm approximations of linear multivariable systems and their L^∞-error bounds. *International Journal of Control*, 39(6):1115–1193, 1984.

[25] R.B. Guenther and J.W. Lee. *Partial Differential Equations of Mathematical Physics and Integral Equations*. Dover Publications, 1996.

[26] R. Heinrich and S. Schuster. *The Regulation of Cellular Systems*. Springer, 1996.

[27] T. Hillen and K.J. Painter. A user's guide to PDE models for chemotaxis. *Journal of Mathematical Biology*, 58(1):183–217, 2009.

[28] E.J. Hinch. *Perturbation Methods*, volume 6 of *Cambridge Texts in Applied Mathematics*. Cambridge University Press, 1991.

[29] R. Hinch, J.L. Greenstein, A.J. Tanskanen, L. Xu, and R.L. Winslow. A simplified local control model of calcium-induced calcium release in cardiac ventricular myocytes. *Biophysical Journal*, 87(6):3723–3736, 2004.

[30] A.L. Hodgkin and A.F. Huxley. A quantitative description of membrane current and its application to conduction and excitation in nerve. *The Journal of Physiology*, 117(4):500–544, 1952.

[31] A. Hoffmann, A. Levchenko, M.L. Scott, and D. Baltimore. The IκB-NF-κB signaling module: temporal control and selective gene activation. *Science*, 298(5596):1241, 2002.

[32] A. Hoffmann, A. Levchenko, M.L. Scott, and D. Baltimore. The IκB-NF-κB signaling module: temporal control and selective gene activation. *Science*, 298(5596):1241, 2002.

[33] F. Hynne, S. Danø, and P.G. Sørensen. Full-scale model of glycolysis in *Saccharomyces cerevisiae*. *Biophysical Chemistry*, 94(1):121–163, 2001.

[34] A.E.C. Ihekwaba, D.S. Broomhead, R.L. Grimley, N. Benson, and D.B. Kell. Sensitivity analysis of parameters controlling oscillatory signalling in the NF-κB pathway: the roles of IKK and IκBα. *Systems Biology*, 1(1):93, 2004.

[35] J.A. Pojman I.R. Epstein. *An Introduction to Nonlinear Chemical Dynamics: Oscillations, Waves, Patterns, and Chaos*. Oxford University Press, USA, 1998.

[36] V. Iyer, R. Mazhari, and R.L. Winslow. A computational model of the human left-ventricular epicardial myocyte. *Biophysical Journal*, 87(3):1507–1525, 2004.

[37] E.M. Izhikevich. *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting*. The MIT Press, 2007.

[38] C. Johnson, S. Larsson, V. Thomée, and L.B. Wahlbin. Error estimates for spatially discrete approximations of semilinear parabolic equations with nonsmooth initial data. *Mathematics of Computation*, 49:331–357, 1987.

[39] H. Kacser, J.A. Burns, and D.A. Fell. The control of flux. *Biochemical Society Transactions*, 23(2):341–366, 1995.

[40] J.P. Keener and J. Sneyd. *Mathematical Physiology: Cellular Physiology*. Springer Verlag, 2009.

[41] S. Kharche, N. Lüdtke, S. Panzeri, and H. Zhang. A global sensitivity index for biophysically detailed cardiac cell models: a computational approach. *Functional Imaging and Modeling of the Heart*, pages 366–375, 2009.

[42] S. Krishna, M.H. Jensen, and K. Sneppen. Minimal model of spiky oscillations in NF-κB signaling. *Proceedings of the National Academy of Sciences USA*, 103(29):10840, 2006.

[43] E. Kuusela and W. Alt. Continuum model of cell adhesion and migration. *Journal of Mathematical Biology*, 58(1):135–161, 2009.

[44] C. Lanczos. *Linear Differential Operators*. Van Nostrand, 1961.

[45] J.C. Leloup and A. Goldbeter. Toward a detailed computational model for the mammalian circadian clock. *Proceedings of the National Academy of Sciences USA*, 100(12):7051, 2003.

[46] P.K. Maini, M.R. Myerscough, K.H. Winter, and J.D. Murray. Bifurcating spatially heterogeneous solutions in a chemotaxis model for biological pattern generation. *Bulletin of Mathematical Biology*, 53(5):701–719, 1991.

[47] S. Marino, I.B. Hogue, C.J. Ray, and D.E. Kirschner. A methodology for performing global uncertainty and sensitivity analysis in systems biology. *Journal of Theoretical Biology*, 254(1):178–196, 2008.

[48] B. Moore. Principal component analysis in linear systems: Controllability, observability, and model reduction. *Automatic Control, IEEE Transactions on*, 26(1):17–32, 1981.

[49] M.D. Morris. Factorial sampling plans for preliminary computational experiments. *Technometrics*, pages 161–174, 1991.

[50] J.D. Murray. *Mathematical Biology*. Springer, 2003.

[51] M.R. Myerscough, P.K. Maini, and K.J. Painter. Pattern formation in a generalized chemotactic model. *Bulletin of Mathematical Biology*, 60(1):1–26, 1998.

[52] J. Nagumo, S. Arimoto, and S. Yoshizawa. An active pulse transmission line simulating nerve axon. *Proceedings of the Institute of Radio Engineers*, 50(10):2061–2070, 1962.

[53] D.E. Nelson, A.E. Ihekwaba, M. Elliott, J.R. Johnson, C.A. Gibney, B.E. Foreman, G. Nelson, V. See, C.A. Horton, D.G. Spiller, et al. Oscillations in NF-κB signaling control the dynamics of gene expression. *Science*, 306(5696):704, 2004.

[54] D. Noble, A. Varghese, P. Kohl, and P. Noble. Improved guinea-pig ventricular cell model incorporating a diadic space, IKr and IKs, and length-and tension-dependent processes. *The Canadian Journal of Cardiology*, 14(1):123, 1998.

[55] J.M. Oliver, J.R. King, K.J. McKinlay, P.D. Brown, D.M. Grant, C.A. Scotchford, and J.V. Wood. Thin-film theories for two-phase reactive flow models of active cell motion. *Mathematical Medicine and Biology*, 22(1):53–98, 2005.

[56] P. Pathmanathan, M.O. Bernabeu, R. Bordas, J. Cooper, A. Garny, J.M. Pitt-Francis, J.P. Whiteley, and D.J. Gavaghan. A numerical guide to the solution of the bidomain equations of cardiac electrophysiology. *Progress in Biophysics and Molecular Biology*, 102(2):136–155, 2010.

[57] J. Pitt-Francis, P. Pathmanathan, M.O. Bernabeu, R. Bordas, J. Cooper, A.G. Fletcher, G.R. Mirams, P. Murray, J.M. Osborne, A. Walter, et al. Chaste: a test-driven approach to software development for biological modelling. *Computer Physics Communications*, 180(12):2452–2471, 2009.

[58] T.P. Prescott and A. Papachristodoulou. Guaranteed error bounds for structured complexity reduction of biochemical networks. *Journal of Theoretical Biology*, 304:172–182, 2012.

[59] H. Rabitz, M. Kramer, and D. Dacol. Sensitivity analysis in chemical kinetics. *Annual Review of Physical Chemistry*, 34(1):419–461, 1983.

[60] D.A. Rand. Mapping global sensitivity of cellular network dynamics: sensitivity heat maps and a global summation law. *Journal of the Royal Society Interface*, 5(Suppl 1):S59, 2008.

[61] D.A. Rand. Network control analysis for time-dependent dynamical states. *Dynamics, Games and Science II*, pages 1–24, 2011.

[62] J.N. Reddy. *An Introduction to the Finite Element Method*. McGraw-Hill New York, 1993.

[63] J.N. Reddy. *An Introduction to Nonlinear Finite Element Analysis*. Oxford University Press, USA, 2004.

[64] L. Romero, E. Pueyo, M. Fink, and B. Rodríguez. Impact of ionic current variability on human ventricular cellular electrophysiology. *American Journal of Physiology – Heart and Circulatory Physiology*, 297(4):H1436, 2009.

[65] W. Rudin. *Real and Complex Analysis*. McGraw-Hill, 1966.

[66] A. Saltelli, M. Ratto, S. Tarantola, and F. Campolongo. Sensitivity analysis for chemical models. *Chemical Reviews*, 105(7):2811–2828, 2005.

[67] A.A. Sher, M.T. Cooling, B. Bethwaite, J. Tan, T. Peachey, C. Enticott, S. Garic, D.J. Gavaghan, D. Noble, D. Abramson, and E.J. Crampin. A global sensitivity tool for cardiac cell modeling: Application to ionic current balance and hypertrophic signaling. In *Engineering in Medicine and Biology Society (EMBC), 2010 Annual International Conference of the IEEE*, pages 1498–1502. IEEE, 2010.

[68] J.L. Snoep, F. Bruggeman, B.G. Olivier, and H.V. Westerhoff. Towards building the silicon cell: a modular approach. *Biosystems*, 83(2):207–216, 2006.

[69] E.A. Sobie. Parameter sensitivity analysis in electrophysiological models using multivariable regression. *Biophysical Journal*, 96(4):1264–1274, 2009.

[70] E. Süli. Finite Element Methods for Partial Differential Equations. `people.maths.ox.ac.uk/suli/fem.pdf`, 2012.

[71] T. Sumner, E. Shephard, and I.D.L. Bogle. A methodology for global-sensitivity analysis of time-dependent outputs in systems biology modelling. *Journal of the Royal Society Interface*, 9(74):2156–2166, 2012.

[72] B. Teusink, J. Passarge, C.A. Reijenga, E. Esgalhado, C.C. van der Weijden, M. Schepper, M.C. Walsh, B.M. Bakker, K. van Dam, H.V. Westerhoff, and J.L. Snoep. Can yeast glycolysis be understood in terms of in vitro kinetics of the constituent enzymes? Testing biochemistry. *European Journal of Biochemistry*, 267(17):5313–5329, 2000.

[73] T. Turányi. Sensitivity analysis of complex kinetic systems. tools and applications. *Journal of Mathematical Chemistry*, 5(3):203–248, 1990.

[74] J.P. Whiteley. An efficient numerical technique for the solution of the monodomain and bidomain equations. *Biomedical Engineering, IEEE Transactions on*, 53(11):2139–2147, 2006.

[75] J.P. Whiteley. Model reduction using *a posteriori* analysis. *Mathematical Biosciences*, 225(1):44–52, 2010.

[76] A.M. Zhabotinsky. A history of chemical oscillations and waves. *Chaos*, 1(4):379–386, 1991.

[77] K. Zhou, J.C. Doyle, and K. Glover. *Robust and Optimal Control*. Prentice Hall Upper Saddle River, NJ, 1996.

Declaration

I herewith declare that I have produced this paper without the prohibited assistance of third parties and without making use of aids other than those specified; notions taken over directly or indirectly from other sources have been identified as such. This paper has not previously been presented in identical or similar form to any other UK or foreign examination board.

Oxford