Report Number 09/08

Model reduction using a posteriori analysis

by

Jonathan P. Whiteley

Oxford Centre for Collaborative Applied Mathematics
Mathematical Institute
24 - 29 St Giles'
Oxford
OX1 3LB
England

Model reduction using *a posteriori* analysis

Jonathan P Whiteley[a]

a: Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, OX1 3QD, United Kingdom

Corresponding author: Dr J.P. Whiteley

Address: Oxford University Computing Laboratory, Wolfson Building, Parks Road, Oxford, OX1 3QD, United Kingdom

Telephone number: +44 1865 273858

Fax number: +44 1865 273839

E-mail address: Jonathan.Whiteley@comlab.ox.ac.uk

Abstract

Mathematical models in biology and physiology are often represented by large systems of non–linear ordinary differential equations. In many cases, an observed behaviour may be written as a linear functional of the solution of this system of equations. A technique is presented in this study for automatically identifying key terms in the system of equations that are responsible for a given linear functional of the solution. This technique is underpinned by ideas drawn from *a posteriori* error analysis. This concept has been used in finite element analysis to identify regions of the computational domain and components of the solution where a fine computational mesh should be used to ensure accuracy of the numerical solution. We use this concept to identify regions of the computational domain and components of the solution where accurate representation of the mathematical model is required for accuracy of the solution. The technique presented is demonstrated by application to a model problem, and then to automatically deduce known results from a cell–level cardiac electrophysiology model.

Keywords: Differential equation model; model simplification.

1 Introduction

Recent years have seen a dramatic increase in the availability and accuracy of experimental data in the life sciences. This increase in data has spawned a large number of complex mathematical models that have been developed to explain the mechanisms responsible for these observations. These models often consist of systems of differential equations—both deterministic and stochastic—and include an increasing number of variables and parameters to capture the large number of processes modelled. Although including more detail in the model may allow explanation of a larger number of physical phenomena, the large number of equations and parameters result in models that soon become become opaque: the key mechanisms causing an observed behaviour may then be unclear.

Many biological and physiological systems are modelled by large systems of ordinary differential equations (ODEs), for example cellular electrophysiology (Noble *et al.* 1998; Iyer *et al.* 2004), cell cycle dynamics (Chen *et al.* 2004), gene regulation (Vilar *et al.* 2002), metabolism (Beard, 2005) and coupled mechano–electrophysiology (Hunter *et al.* 1998). When solving a (possibly non–linear) system of ODEs such as these, our main interest is often a linear functional of the solution, for example: (i) the solution of a given component at a specific time t^*, $u_i(t^*)$; (ii) the weighted average of some component of the solution over a given time interval; (iii) a linear functional of the derivative of a specific component of the solution; or (iv) a combination of these simple linear functionals.

In this study we present an algorithm for automatically identifying the key mechanisms that determine a linear functional of the solution of a system of ODEs. This technique is based on the concept of *a posteriori* error analysis. *A posteriori* error analysis, when applied to the finite element technique for calculating the numerical solution of a system of ODEs, automatically identifies regions of the computational domain and components of the solution where a fine finite element mesh must be used in order to maintain accuracy (Eriksson *et al.* 1995). This technique has been utilised by several groups for developing mesh refinement strategies (see, for example, Harriman *et al.*, 2000). Rather than use this concept to automatically identify where a fine finite element mesh should be used, we instead use it to identify regions of the computational domain and components of the solution where accurate representation of the mathematical model is crucial for the linear functional to be accurate. We may use this approach to identify the key mechanisms that a mathematical model predicts is responsible an observed behaviour. This information may then be used to guide the development of a simplified model that includes all key mechanisms required to explain a given observation.

2 The theoretical basis of the technique

We assume that we are given: (i) a system of ODEs; and (ii) a linear functional of the solution that we wish to calculate. Before deriving the algorithm used in this

paper, we first write down some mathematical preliminaries that allow the theory to be developed rigorously. We then derive a bound on the error that is induced in a linear functional of the solution when a simplified ODE system—rather the full ODE system—is used to compute this linear functional. We then describe a practical implementation of this error bound that may be used to automatically identify the key mechanisms that cause a given observation.

2.1 Mathematical preliminaries

In the following sections we denote vectors using bold font, and the entries of a vector using subscripts. For example, a vector $\mathbf{v}$ of length N is written $\mathbf{v} = (v_1, v_2, \ldots, v_N)^{\mathrm{t}}$.

We denote the time interval $t_1 < t < t_2$ by (t_1, t_2).

Let V be the vector space of vector–valued functions of a prescribed size defined on the time interval $0 < t < T$ for a given $T > 0$. We may then define the inner product between $\mathbf{v}, \mathbf{w} \in V$ by

$$\langle \mathbf{v}, \mathbf{w} \rangle = \int_0^T \mathbf{v}(t) \cdot \mathbf{w}(t) \, \mathrm{d}t.$$

We also require the Riesz representation theorem (see, for example, Griffel 1981): for any linear functional $J(\mathbf{v})$, where $\mathbf{v} \in V$, there exists $\mathbf{g} \in V$ such that

$$J(\mathbf{v}) = \langle \mathbf{v}, \mathbf{g} \rangle, \qquad \forall \mathbf{v} \in V.$$

2.2 Derivation of an *a posteriori* error bound

Suppose we wish to calculate the linear functional

$$J(\mathbf{u}) = \langle \mathbf{u}, \mathbf{g} \rangle, \tag{1}$$

where $\mathbf{g}(t)$ is a prescribed vector of length N defined for $0 < t < T$, and the vector–valued function $\mathbf{u}$ (of length N) satisfies the initial value problem

$$\frac{\mathrm{d}\mathbf{u}}{\mathrm{d}t} + \mathbf{f}(t, \mathbf{u}) = \mathbf{0}, \qquad 0 < t < T, \qquad \mathbf{u}(0) = \mathbf{u_0}. \tag{2}$$

Suppose $\mathbf{U}$ is an approximation to $\mathbf{u}$ that has been calculated from a system of ODEs that is a simplification of (2). We would then be interested in the error in the linear functional (1) that is induced by using $\mathbf{U}$ instead of $\mathbf{u}$, i.e. the quantity $|J(\mathbf{U}) - J(\mathbf{u})|$. We first write down some requirements on $\mathbf{U}$ that must be satisfied in order for us to derive an error bound. We demand that $\mathbf{U}(t)$ is a vector–valued function defined on $0 < t < T$ that satisfies the following conditions:

1. $\mathbf{U}(t)$ is continuous for $0 < t < T$;

2. $\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t}$ is continuous for $0 < t < T$, except for a finite set of M points in time, $t_1, \ldots, t_M$, where $0 = t_0 < t_1 < \ldots < t_M < t_{M+1} = T$; and

3. $\mathbf{U}(t)$ satisfies the correct initial conditions of the ODE (2), i.e. $\mathbf{U}(0) = \mathbf{u_0}$.

We will now show how the error induced in (1) by using $\mathbf{U}$—instead of the true solution $\mathbf{u}$—may be computed using the solution of an appropriate dual problem.

2.2.1 Definition of the dual problem

In this section we assume that the vector valued function $\mathbf{g}$, used to calculate the linear functional (1) takes a finite value at $t = T$. This is required for the analysis in this section. In Section 2.2.3 we explain how the analysis may be modified to allow entries of $\mathbf{g}$ to contain delta functions such as $g_1 = \delta(t - T)$.

We define the dual problem associated with (2) by

$$-\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t} + A\boldsymbol{\phi} = \mathbf{g}, \qquad \boldsymbol{\phi}(T) = \mathbf{0}, \tag{3}$$

where $\mathbf{g}$ is the vector valued function that defines the linear functional of interest in Eq. (1), and the matrix A has entries given by

$$\begin{aligned} A_{ij}(t) &= \int_0^1 \frac{\partial f_j}{\partial u_i}(t, \boldsymbol{\alpha}(s,t))\, \mathrm{d}s, \qquad i,j = 1,2,\ldots,N, \\ \text{where} \qquad \boldsymbol{\alpha}(s,t) &= s\mathbf{u}(t) + (1-s)\mathbf{U}(t), \end{aligned} \tag{4}$$

and $\mathbf{U}$ is defined in Section 2.2. Note that under these conditions the entries of A are continuous, but their derivatives w.r.t. time may have discontinuities at $t = t_1, \ldots, t_M$. Using this definition of the entries of A we may write, for $j = 1, 2, \ldots, N$,

$$\begin{aligned} \sum_{i=1}^{N} A_{ij}(t)\,(u_i(t) - U_i(t)) &= \sum_{i=1}^{N} \int_0^1 \frac{\partial f_j}{\partial u_i}(t, \boldsymbol{\alpha}(s,t))\,(u_i(t) - U_i(t))\ \mathrm{d}s, \\ &= \sum_{i=1}^{N} \int_0^1 \frac{\partial f_j}{\partial \alpha_i}\frac{\partial \alpha_i}{\partial s}\ \mathrm{d}s, \\ &= \int_0^1 \frac{\mathrm{d}f_j}{\mathrm{d}s}\ \mathrm{d}s, \\ &= f_j(t,\mathbf{u}) - f_j(t,\mathbf{U}). \end{aligned} \tag{5}$$

This result will be needed when deriving an *a posteriori* error bound.

2.2.2 Use of the dual problem to derive an *a posteriori* error bound

We can use the solution of the dual problem (3) to bound the error that is induced in the linear functional (1) when $\mathbf{u}$ is replaced by $\mathbf{U}$. We write

$$\begin{aligned}
J(\mathbf{u}) \;-\; J(\mathbf{U}) & \\
&= \langle \mathbf{u}, \mathbf{g} \rangle - \langle \mathbf{U}, \mathbf{g} \rangle \\
&= \int_0^T (\mathbf{u} - \mathbf{U}) \cdot \mathbf{g} \; \mathrm{d}t, \\
&= \sum_{i=1}^N \int_0^T (u_i - U_i)\, g_i \; \mathrm{d}t, \\
&= \sum_{i=1}^N \int_0^T (u_i - U_i) \left(-\frac{\mathrm{d}\phi_i}{\mathrm{d}t} + \sum_{j=1}^N A_{ij}\phi_j \right) \; \mathrm{d}t, \qquad \text{using (3)}, \\
&= \sum_{k=0}^M \sum_{i=1}^N \int_{t_k}^{t_{k+1}} (u_i - U_i) \left(-\frac{\mathrm{d}\phi_i}{\mathrm{d}t} + \sum_{j=1}^N A_{ij}\phi_j \right) \; \mathrm{d}t, \\
&= \sum_{k=0}^M \sum_{i=1}^N \left(-\left[(u_i - U_i)\, \phi_i \right]_{t_k}^{t_{k+1}} + \int_{t_k}^{t_{k+1}} \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^N A_{ij}\, (u_i - U_i)\, \phi_j \; \mathrm{d}t \right), \\
&= \sum_{i=1}^N \left(-\left[(u_i - U_i)\, \phi_i \right]_0^T + \int_0^T \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^N A_{ij}\, (u_i - U_i)\, \phi_j \; \mathrm{d}t \right), \qquad (6)
\end{aligned}$$

where the last equation follows because $\mathbf{u}$, $\mathbf{U}$, $\boldsymbol{\phi}$ are continuous functions. The first term on the right–hand–side of (6) is zero becuase: (i) $\mathbf{u}(0) = \mathbf{U}(0) = \mathbf{u_0}$; and (ii) the dual problem (3) has been defined with $\boldsymbol{\phi}(T) = \mathbf{0}$. We may therefore write

$$J(\mathbf{u}) - J(\mathbf{U}) = \sum_{i=1}^N \left(\int_0^T \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^N A_{ij}\, (u_i - U_i)\, \phi_j \; \mathrm{d}t \right). \qquad (7)$$

Using (2) and (5) allows us to write (7) as

$$\begin{aligned}
J(\mathbf{u}) - J(\mathbf{U}) &= \int_0^T \sum_{i=1}^N \left(-f_i(t, \mathbf{u}) - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^N \left(f_j(t, \mathbf{u}) - f_j(t, \mathbf{U}) \right) \phi_j \; \mathrm{d}t, \\
&= \int_0^T \sum_{i=1}^N -\phi_i \left(\frac{\mathrm{d}U_i}{\mathrm{d}t} + f_i(t, \mathbf{U}) \right) \; \mathrm{d}t.
\end{aligned}$$

Given a solution $\mathbf{u}$ of the ODE (2), and a vector valued function $\mathbf{U}$ satisfying the conditions listed in Section 2.2, we may write our error bound as

$$|J(\mathbf{u}) - J(\mathbf{U})| \leq \sum_{i=1}^{N}\sum_{k=0}^{M}\int_{t_k}^{t_{k+1}} |\phi_i| \left|\left(\frac{\mathrm{d}U_i}{\mathrm{d}t} + f_i(t,\mathbf{U})\right)\right| \,\mathrm{d}t. \tag{8}$$

Before describing how this error bound may be utilised when deriving simplified models, we first explain how the analysis presented may be modified when the vector $\mathbf{g}$ used in (1) contains a delta function $\delta(t-T)$, and is therefore unbounded at $t = T$.

2.2.3 Modified dual problem when g includes delta functions, $\delta(t-T)$

Suppose our quantity of interest is the value of component P of the solution at time $t = T$. We may then write our linear functional as

$$\begin{aligned} J(\mathbf{u}) &= u_P(T), \\ &= \int_0^T \mathbf{u}\cdot\mathbf{g}\,\mathrm{d}t, \\ &= \langle \mathbf{u}, \mathbf{g}\rangle, \end{aligned}$$

where $\mathbf{g} = \delta(t-T)\mathbf{h}$, and $\mathbf{h}$ has entries given by

$$h_i = \begin{cases} 1, & i = P, \\ 0, & i \neq P. \end{cases}$$

The vector $\mathbf{g}$ is unbounded at $t = T$ and so the dual problem and boundary conditions given by (3) are not consistent when $\mathbf{g}$ takes this form. Instead, we may use the dual problem given by

$$-\frac{\mathrm{d}\boldsymbol{\phi}}{\mathrm{d}t} + A\boldsymbol{\phi} = \mathbf{0}, \qquad \boldsymbol{\phi}(T) = \mathbf{h}(T). \tag{9}$$

Using the dual problem defined in (9), the analysis presented in Section 2.2.2 may be modified to give

$$
\begin{aligned}
0 &= \sum_{i=1}^{N} \int_0^T (u_i - U_i) \left(-\frac{\mathrm{d}\phi_i}{\mathrm{d}t} + \sum_{j=1}^{N} A_{ij}\phi_j \right) \, \mathrm{d}t, \qquad \text{using (9)}, \\
&= \sum_{k=0}^{M} \sum_{i=1}^{N} \int_{t_k}^{t_{k+1}} (u_i - U_i) \left(-\frac{\mathrm{d}\phi_i}{\mathrm{d}t} + \sum_{j=1}^{N} A_{ij}\phi_j \right) \, \mathrm{d}t, \\
&= \sum_{k=0}^{M} \sum_{i=1}^{N} \left(-\left[(u_i - U_i)\,\phi_i \right]_{t_k}^{t_{k+1}} + \int_{t_k}^{t_{k+1}} \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^{N} A_{ij} \, (u_i - U_i)\, \phi_j \, \mathrm{d}t \right), \\
&= \sum_{i=1}^{N} \left(-\left[(u_i - U_i)\,\phi_i \right]_0^T + \int_0^T \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^{N} A_{ij} \, (u_i - U_i)\, \phi_j \, \mathrm{d}t \right), \\
&= \sum_{i=1}^{N} \left(-\left(u_i(T) - U_i(T) \right) \phi_i(T) + \int_0^T \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^{N} A_{ij} \, (u_i - U_i)\, \phi_j \, \mathrm{d}t \right), \\
&= -u_P(T) + U_P(T) + \sum_{i=1}^{N} \left(\int_0^T \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^{N} A_{ij} \, (u_i - U_i)\, \phi_j \, \mathrm{d}t \right),
\end{aligned}
$$

from which we may deduce that

$$
J(\mathbf{u}) - J(\mathbf{U}) = \sum_{i=1}^{N} \left(\int_0^T \left(\frac{\mathrm{d}u_i}{\mathrm{d}t} - \frac{\mathrm{d}U_i}{\mathrm{d}t} \right) \phi_i + \sum_{j=1}^{N} A_{ij} \, (u_i - U_i)\, \phi_j \, \mathrm{d}t \right),
$$

which is identical to (7), from which the error bound (8) may be deduced using the same argument as in Section 2.2.2.

2.3 Use of the error bound to reduce a given system

In this section we explain how to construct a simplified model corresponding to the full model (2) that includes the key features required to compute the linear functional given by (1) to a given precision. We define $\mathbf{U}$ to be the solution of the

ODE given by

$$\frac{\mathrm{d}\mathbf{U}}{\mathrm{d}t} + \hat{\mathbf{f}}(t, \mathbf{U}) = \mathbf{0}, \qquad 0 < t < T, \qquad \mathbf{U}(0) = \mathbf{u_0}, \tag{10}$$

where

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

For each component of the solution $i = 1, 2, \ldots, N$, we define S_i to be a subset of the time interval $0 < t < T$ which is to be determined. Note that S_i is not necessarily a continuous time interval—we allow, for example, $S_i = \{(t_1, t_2) \cup (t_3, t_4)\}$, where $t_1 < t_2 < t_3 < t_4$. The algorithm descibed below automatically generates the sets S_i, $i = 1, 2, \ldots, N$.

The time interval $0 < t < T$ is partitioned into Q equal time intervals $t_{j-1} < t < t_j$, $j = 1, 2, \ldots, Q$. We begin with $S_i = \emptyset$, $i = 1, 2, \ldots, N$, (so that $\hat{\mathbf{f}}(t, \mathbf{U}) = \mathbf{0}$). It is assumed that $\mathbf{u}$ has already been calculated using (2). We then repeat the following process until the error $|J(\mathbf{u}) - J(\mathbf{U})|$ is sufficiently small.

1. Solve (10) to calculate $\mathbf{U}$.

2. Solve (3) or (9) to calculate $\boldsymbol{\phi}$.

3. Use $\mathbf{U}$ and $\boldsymbol{\phi}$ to calculate $|J(\mathbf{u}) - J(\mathbf{U})|$. Stop if this bound becomes acceptably small.

4. Identify the largest N_{cont} contributions—over both i and k—to the error bound (8), and include these time intervals in the appropriate S_i.

5. Repeat steps 1–5.

As an analytic solution is not available for a general system of ODEs, (2), (3), (9) are usually solved numerically. In the numerical results later in this study we use an explicit Euler approximation to calculate the numerical solution (see, for example, Süli and Mayers, 2003). Numerical integration is also required for calculating the entries of A, as described in (4). This numerical integration may be performed using Gaussian quadrature (Süli and Mayers, 2003).

3 Application to a model problem

We now illustrate the use of the algorithm by application to a model problem. The model problem used is the system of two equations given by

$$\frac{\mathrm{d}u_1}{\mathrm{d}t} = -u_1, \qquad 0 < t < T, \tag{11}$$

$$\frac{\mathrm{d}u_2}{\mathrm{d}t} = u_1 + u_2, \qquad 0 < t < T, \tag{12}$$

combined with initial conditions $u_1(0) = u_2(0) = 1$. This system has solution

$$u_1 = \mathrm{e}^{-t}, \qquad u_2 = \frac{1}{2}\left(3\mathrm{e}^{t} - \mathrm{e}^{-t}\right). \tag{13}$$

We now apply the algorithm presented in Section 2.3 to a selection of linear functionals, (1), with different choices for $\mathbf{g}$.

3.1 Deducing that u_1 is independent of u_2

We see from the system of equations (11), (12) that u_1 is entirely independent of u_2. We therefore expect the model reduction algorithm to deduce that any linear functional depending only on u_1—i.e. where $\mathbf{g} = (g_1(t), 0)^{\mathrm{t}}$—may be calculated accurately by only solving (11), and ignoring (12). We can verify that this is true by considering the matrix A that is used in the dual problems, (3) and (9). Using (4) we see that

$$A = \begin{pmatrix} -1 & 1 \\ 0 & 1 \end{pmatrix}. \tag{14}$$

Whether dual problem (3) or dual problem (9) is used, when $\mathbf{g} = (g_1(t), 0)^{\mathrm{t}}$ and the matrix A is given by (14), we may deduce that ϕ_2 satisfies the differential equation

$$-\frac{\mathrm{d}\phi_2}{\mathrm{d}t} + \phi_2 = 0, \qquad \phi_2(T) = 0,$$

which has solution $\phi_2(t) = 0$ for $0 < t < T$. Substitution of $\phi_2(t) = 0$ in the error bound (8) allows us to deduce that the error calculated using this bound is independent of the solution of (12). Hence the linear functional is completely independent of the solution of (12), and the model reduction algorithm may be used to deduce that u_1 is independent of u_2 as expected.

3.2 Computing u_1 to a given accuracy

Suppose we set the final time, $T = 10$, and want to calculate the linear functional given by

$$J(\mathbf{u}) = \frac{1}{T}\int_0^T u_1(t)\ \mathrm{d}t, \tag{15}$$

to within a tolerance of 10^{-5}. By normalising the integral in (15) with the factor $1/T$ we may think of this linear functional as being the average value of u_1 on the time interval $0 < t < T$. By inspection of the true solution (13) we see that u_1 decays exponentially to zero with time: hence we expect that $\mathrm{d}u_1/\mathrm{d}t = 0$ should be a good approximation for t sufficiently large. Furthermore, based on the discussion in Section 3.1, we expect that the algorithm will predict that this may be calculated independently of the second equation, (12). To demonstrate this, we apply the algorithm with data $\mathbf{g} = (1/T, 0)^{\mathrm{t}}$ in (3). As described in Section 2.3, we solve (2) and (3) numerically with a timestep 10^{-2}. We deduce that we can calculate the linear functional $J(\mathbf{U})$ defined in (15), such that $|J(\mathbf{u}) - J(\mathbf{U})| \leq 10^{-5}$, by using the model

$$\begin{aligned}
\frac{\mathrm{d}U_1}{\mathrm{d}t} &= \begin{cases} -U_1, & 0 < t < 8.68, \\ 0, & 8.68 < t < 10, \end{cases} \\
\frac{\mathrm{d}U_2}{\mathrm{d}t} &= 0, \qquad 0 < t < 10.
\end{aligned}$$

Note that the time predicted by the algorithm for u_1 to decay sufficiently that the simplification $\mathrm{d}u_1/\mathrm{d}t = 0$ gives a good approximation is 8.68. This is slightly different to the true value of 8.836 which may be deduced from (13). This slight

difference is because: (i) there are small errors induced by calculating the numerical solution of (2) and (3); and (ii) we only include time intervals in the reduced model that are multiples of the time step used to solve the equations numerically. The time predicted by the reduced model approaches the true value as the time step is reduced to zero.

3.3 The dependence of u_2 on u_1

We see in (13) that u_1 decays exponentially to zero with time, whilst u_2 increases exponentially with time. As such, for t sufficiently large, we have $u_2 \gg u_1$, and would expect the u_1 term in (12) to be negligible compared to the u_2 term. This is indeed detected by the algorithm: and may be demonstrated using the linear functional given by

$$J(\mathbf{u}) = \frac{1}{T}\int_0^T u_2(t)\ \mathrm{d}t, \tag{16}$$

which is the average of u_2 over the time interval $0 < t < T$. Again taking final time $T = 10$, we see that the true value of $J(\mathbf{u})$ given by (16) is 33038. Demanding that this linear functional is computed to within 0.33 of the true value—i.e. an absolute error of 0.001%—we may use the simplified model

$$\begin{aligned} \frac{\mathrm{d}U_1}{\mathrm{d}t} &= \begin{cases} -U_1, & 0 < t < 4.05, \\ 0, & 4.05 < t < 10, \end{cases} \\ \frac{\mathrm{d}U_2}{\mathrm{d}t} &= U_1 + U_2, \qquad 0 < t < 10, \end{aligned}$$

thus demonstrating that, for the linear functional given by (16), the ODE (11) is redundant for t sufficiently large.

4 Application to a cardiac electrophysiology model

Having demonstrated the use of the technique for a model problem, we now apply the algorithm to a cell–level ODE model arising in cardiac electrophysiology. We use the model described by Noble *et al.* (1998) which is a system of 22 ODEs for the transmembrane potential V_m, the concentration of various species, and various gating variables that control components of the transmembrane ionic current. The quantities that are modelled are summarised in Table 1. This model is an excitable system: if a sufficiently large stimulating current is applied across the cell membrane an action potential is generated. The primary quantity of interest to many researchers is the transmebrane potential during this action potential—a plot of this quantity against time during an action potential is shown in Figure 1.

Eq	Quantity
1	Activator fraction for calcium release
2	Product fraction for calcium release
3	Fast sodium current h gate
4	Fast sodium current m gate

5	Concentration of calcium bound to calmodulin
6	Concentration of calcium in diadic subspace
7	Intracellular calcium concentration
8	Calcium concentration in junctional sarcoplasmic reticulum release store
9	Concentration of calcium bound to troponin
10	Calcium concentration in network sarcoplasmic reticulum uptake store
11	Intracellular potassium concentration
12	Intracellular sodium concentration
13	L type calcium channel d gate
14	L type calcium channel f gate
15	L type calcium channel f_2 gate
16	L type calcium channel f_2 diadic space gate
17	Transmembrane potential
18	Rapid delayed rectifier potassium current x_{r1} gate
19	Rapid delayed rectifier potassium current x_{r2} gate
20	Slow delayed rectifier potassium current x_s gate
21	Transient outward current r gate
22	Transient outward current s gate

Table 1: The state variables in the electrophysiology model described by Noble *at al.* (1998).

4.1 Implementation of the algorithm

The equations used are those originally published by Noble *at al.* (1998). Parameter values and initial conditions used are those listed by Whiteley *et al.* (2007). Code was automatically generated using Cellular Open Resource (COR), described by Garny *at al.* (2003). The equations were solved using the explicit Euler method with a timestep of 2×10^{-5} s.

We will concentrate on computing the transmembrane potential V_m accurately, as this is the quantity that is of most importance to experimental workers and plays a central role in determining the transmembrane ionic current. We see in Table 1 that the transmembrane potential is the 17th component of the solution, and so the linear functional that will be used to drive the model reduction will contain solution component u_{17}. We see in Figure 1 that the solution for V_m returns to its steady state after the action potential. If we were to use $u_{17}(T)$ as our linear functional then, for T sufficiently large, the error in the action potential would be small enough if we were to solve the system $\mathrm{d}\mathbf{U}/\mathrm{d}t = \mathbf{0}$ for $0 < t < T$. As such, our functional of interest is given by

$$J(\mathbf{u}) = \frac{1}{T}\int_0^T u_{17}(t)\ \mathrm{d}t. \tag{17}$$

This functional can be interpreted as the avergage of V_m over the interval $0 < t < T$. In all simulations below, we demand $|J(\mathbf{u}) - J(\mathbf{U})| < 1$ mV.

4.2 Use of the algorithm

Our first simulations generate reduced systems of ODEs that calculate the linear functional given in (17), to within a tolerance of 1 mV, for $T = 0.003$ s, $T = 0.005$ s and $T = 0.01$ s. The transmembrane potentials, calculated using the reduced model predicted from the algorithm, are shown in Figure 2(a) for $T = 0.003$ s, in Figure 3(a) for $T = 0.005$ s, in Figure 4(a) for $T = 0.01$ s. In all of these plots the solid line represents V_m calculated using the reduced model, whilst the true plot of V_m calculated from the full model is shown by the dotted line. We see good agreement between the solution of the full model and the solution of the reduced model. For these three simulations, the reduced model solution was computed by solving equations only for three quantities out of those listed in Table 1, specifically the equations for V_m, the fast sodium current h gate, and the fast sodium current m gate. The precise forms of the reduced model are given below.

1. $T = 0.003$ **s.** Reduced model satisfies:

$$\begin{aligned}
\frac{\mathrm{d}U_4}{\mathrm{d}t} + f_4(t, \mathbf{U}) = 0, &\qquad 0.0012 \text{ s} < t < 0.00232 \text{ s}, \\
\frac{\mathrm{d}U_{17}}{\mathrm{d}t} + f_{17}(t, \mathbf{U}) = 0, &\qquad 0 \text{ s} < t < 0.00238 \text{ s}, \quad 0.00248 \text{ s} < t < 0.00268 \text{ s}, \\
\frac{\mathrm{d}U_i}{\mathrm{d}t} = 0, &\qquad \text{otherwise.}
\end{aligned}$$

2. $T = 0.005$ **s.** Reduced model satisfies:

$$\begin{aligned}
\frac{\mathrm{d}U_3}{\mathrm{d}t} + f_3(t, \mathbf{U}) = 0, \qquad & 0.0022 \text{ s} < t < 0.00232 \text{ s}, \quad 0.00242 \text{ s} < t < 0.00414 \text{ s}, \\
\frac{\mathrm{d}U_4}{\mathrm{d}t} + f_4(t, \mathbf{U}) = 0, \qquad & 0.0011 \text{ s} < t < 0.00234 \text{ s}, \quad 0.00338 \text{ s} < t < 0.00340 \text{ s}, \\
& 0.00364 \text{ s} < t < 0.00366 \text{ s}, \\
\frac{\mathrm{d}U_{17}}{\mathrm{d}t} + f_{17}(t, \mathbf{U}) = 0, \qquad & 0 \text{ s} < t < 0.00274 \text{ s}, \quad 0.00284 \text{ s} < t < 0.003 \text{ s}, \\
& 0.0033 \text{ s} < t < 0.00336 \text{ s}, \quad 0.00356 \text{ s} < t < 0.00366 \text{ s}, \\
& 0.00374 \text{ s} < t < 0.00386 \text{ s}, \\
\frac{\mathrm{d}U_i}{\mathrm{d}t} = 0, \qquad & \text{otherwise.}
\end{aligned}$$

3. $T = 0.01$ **s.** Reduced model satisfies:

$$\begin{aligned}
\frac{\mathrm{d}U_3}{\mathrm{d}t} + f_3(t, \mathbf{U}) = 0, \qquad & 0.00224 \text{ s} < t < 0.00676 \text{ s}, \\
\frac{\mathrm{d}U_4}{\mathrm{d}t} + f_4(t, \mathbf{U}) = 0, \qquad & 0.00124 \text{ s} < t < 0.00232 \text{ s}, \quad 0.00242 \text{ s} < t < 0.00526 \text{ s}, \\
\frac{\mathrm{d}U_{17}}{\mathrm{d}t} + f_{17}(t, \mathbf{U}) = 0, \qquad & 0 \text{ s} < t < 0.0024 \text{ s}, \quad 0.00244 \text{ s} < t < 0.00282 \text{ s}, \\
& 0.00294 \text{ s} < t < 0.00296 \text{ s}, \quad 0.00306 \text{ s} < t < 0.00316 \text{ s}, \\
& 0.0032 \text{ s} < t < 0.00322 \text{ s}, \quad 0.00338 \text{ s} < t < 0.0035 \text{ s}, \\
& 0.00356 \text{ s} < t < 0.00358 \text{ s}, \quad 0.00378 \text{ s} < t < 0.00382 \text{ s}, \\
& 0.00394 \text{ s} < t < 0.00396 \text{ s}, \quad 0.0042 \text{ s} < t < 0.00522 \text{ s}, \\
& 0.00636 \text{ s} < t < 0.00638 \text{ s}, \quad 0.00682 \text{ s} < t < 0.00684 \text{ s}, \\
& 0.00716 \text{ s} < t < 0.00718 \text{ s}, \\
\frac{\mathrm{d}U_i}{\mathrm{d}t} = 0, \qquad & \text{otherwise.}
\end{aligned}$$

Plots of the fast sodium current h gate, and the fast sodium current m gate are given in Figure 2(b) for $T = 0.003$ s, in Figure 3(b) for $T = 0.005$ s, in Figure 4(b) for $T = 0.01$ s. In all cases the solid line represents the h gate calculated using the reduced model, the broken line represents the m gate calculated using the reduced model, and the dotted lines represent h and m calculated using the full model.

We have seen in Figures 2–4 that the initial stages of the action potential shown in Figure 1—i.e. the rapid upstroke—may be computed reasonably accurately by using a model containing only three dependent variables: the transmembrane potential V_m; the fast sodium current h gate; and the fast sodium current m gate. This is in agreement with known physiology, as the steep upstroke of the action potential is caused by the fast sodium current.

We now turn our attention to computing V_m accurately over the whole of the action potential shown in Figure 1, and therefore use $T = 0.25$ s in (17). The plot of V_m calculated using the reduced model generated is shown by the solid line in Figure 5(a), whilst the solution of the whole model is shown by the dotted line. We see that the algorithm has generated a model that gives an accurate representation of V_m over the course of a whole action potential. Far more of the equations listed in Table 1 are included in the reduced model for this value of T than was the case for the reduced models that generated the upstroke of the action potential accurately. This is shown in Figure 5(b), where a solid line indicates the time intervals for which a given equation is included in the model, with the equation numbers corresponding to those given in Table 1. We see that almost all the

equations listed in Table 1 play a role in the accurate computation of the action potential at some point, with the only exceptions being the equations representing intracellular potassium concentration and the L type calcium channel f_2 gate (equations 11 and 15 in Table 1). This confirms accepted physiological knowledge, namely that the whole action potential depends on many interrelated processes.

5 Discussion

We have presented a model reduction algorithm in this paper, and have demonstrated its application to both a model problem and a reasonably large system of differential equations that has been used extensively to model cell level cardiac electrophysiology. In both cases the algorithm behaved as expected, and known features of these two systems of equations were elucidated automatically. This has an obvious application in the development of mathematical models, as the key mechanisms that the model predicts are responsible for a given observed behaviour can be compared with known physiology. This will aid the iterative model development between experimental workers and mathematicians, and facilitate some checking of any extensions to a mathematical model against known biology or physiology.

A further application of the algorithm presented here is in the design of experiments. Given an experimental protocol, we can use the algorithm presented here to identify the key mechanisms that are responsible for the observed

experimental data. Before carrying out an experiment that is designed with a mathematical model in mind, we may first check what components of the model are predicted to have a significant effect on the behaviour that will be observed. This will help design experiments that reap as much benefit from the data obtained as possible.

In this study we have restricted ourselves to applying the algorithm presented to models represented by systems of ODEs. There is no reason why this algorithm cannot be applied to the wide range of partial differential equation (PDE) models that have been derived in biology and physiology (Murray, 2003; Keener and Sneyd, 1998). All that is required is the derivation of a suitable adjoint problem for the governing PDE, which should not pose significant difficulties.

Acknowledgements

This publication is based on work supported by Award No. KUK-C1-013-04, made by King Abdullah University of Science and Technology (KAUST).

References

Beard, D.A., 2005. A biophysical model of the mitochondrial respiratory system and oxidative phosphorylation. PLOS Comput. Biol. 1, 252–264.

Chen, K.C., Calzone, L., Csikasz–Nagy, A., Cross, F.R., Novak, B., Tyson, J.J., 2004. Integrative analysis of cell cycle control in budding yeast. Mol. Biol. Cell 15, 3841–3862.

Eriksson, K., Estep, D., Hansbo, P., Johnson, C., 1995. Introduction to adaptive methods for differential equations. Acta Numerica 1995, 105–158.

Garny, A., Kohl, P., Noble, D., 2003. Cellular open resource (COR): a public CellML based environment for modeling biological function. Int. J. Bif. Chaos 13, 3579–3590.

Griffel, D.H., 1981. Applied Functional Analysis. John Wiley and sons, New York.

Harriman, K., Gavaghan, D.J., Houston, P., Süli, E., 2000. Adaptive finite element simulation of currents at microelectrodes to a guaranteed accuracy. First–order EC' mechanism at inlaid and recessed discs. Electrochem. Commun. 2, 163–170.

Hunter, P.J., McCulloch, A.D., ter Keurs, H.E.D.J., 1998. Modelling the mechanical properties of cardiac muscles. Prog. Biophys. Mol. Biol. 69, 289–331.

Iyer, V.I., Mazhari, R., Winslow, R.L., 2004. A computational model of the human left–ventricular epicardial myocyte. Biophys. J. 87, 1507–1525.

Keener, J.P., Sneyd, J., 1998. Mathematical Physiology. Springer–Verlag, New York.

Murray, J.D., 2003. Mathematical Biology. Springer, New York.

Noble, D., Varghese, A., Kohl, P., Noble, P., 1998. Improved guinea–pig ventricular cell model incorporating a diadic space, IKr and IKs, and length– and tension–dependent processes. Cand. J. Cardiol. 14, 123–134.

Süli, E., Mayers, D.F., 2003. An Introduction to Numerical Analysis. Cambridge University Press, Cambridge, UK.

Vilar, J.M.G., Kueh, H-Y., Barkai, N., Leibler, S., 2002. Mechanisms of noise–resistance in genetic oscillators. Proc. Natl. Acad. Sci. USA 99, 5988–5992.

Whiteley, J.P., Bishop, M.J., Gavaghan, D.J., 2007. Soft tissue modelling of cardiac fibres for use in coupled mechano–electric simulations. Bull. Math. Biol. 69, 2199–2225.

Figure Legends

Figure 1. An action potential calculated using the model described by Noble *et al.* (1998).

Figure 2. Application of the algorithm to the model described by Noble *et al.* (1998) for the time interval $0 < t < 0.003$ s. (a) The transmembrane potential V_m (solid line represents reduced model, dotted line represents full model). (b) The gating variables h (solid line represents reduced model, dotted line represents full model) and m (broken line represents reduced model, dotted line represents full model).

Figure 3. Application of the algorithm to the model described by Noble *et al.* (1998) for the time interval $0 < t < 0.005$ s. (a) The transmembrane potential V_m (solid line represents reduced model, dotted line represents full model). (b) The gating variables h (solid line represents reduced model, dotted line represents full model) and m (broken line represents reduced model, dotted line represents full model).

Figure 4. Application of the algorithm to the model described by Noble *et al.* (1998) for the time interval $0 < t < 0.01$ s. (a) The transmembrane potential V_m (solid line represents reduced model, dotted line represents full model). (b) The gating variables h (solid line represents reduced model, dotted line represents full model) and m (broken line represents reduced model, dotted line represents full model).

Figure 5. Application of the algorithm to the model described by Noble *et al.* (1998) for the time interval $0 < t < 0.01$ s. (a) The transmembrane potential V_m (solid line represents reduced model, dotted line represents full model). (b) The inclusion of a given equation in the model—a solid line indicates that the equation is included at a given time. Equation numbers correspond to those given in Table 1.

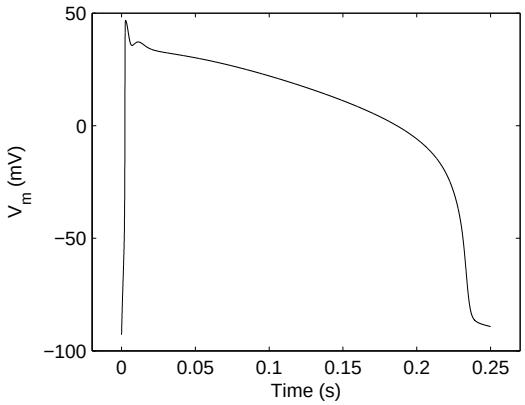
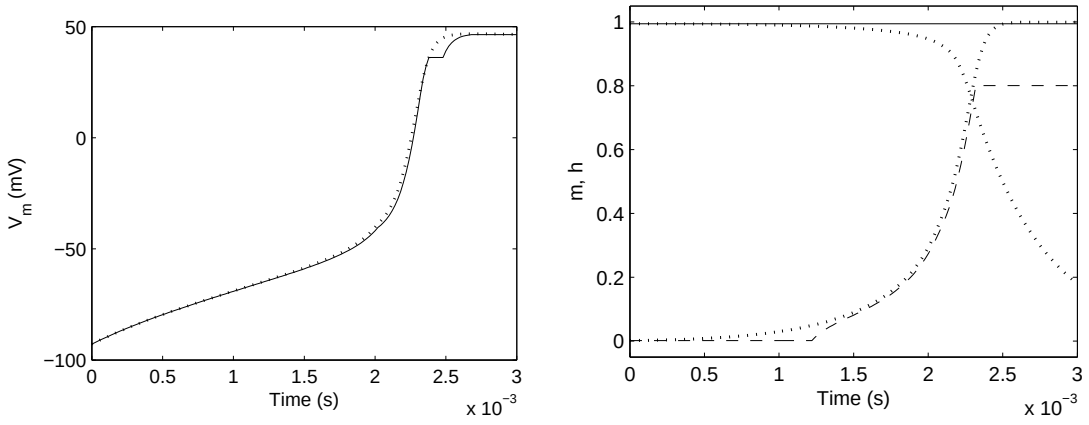
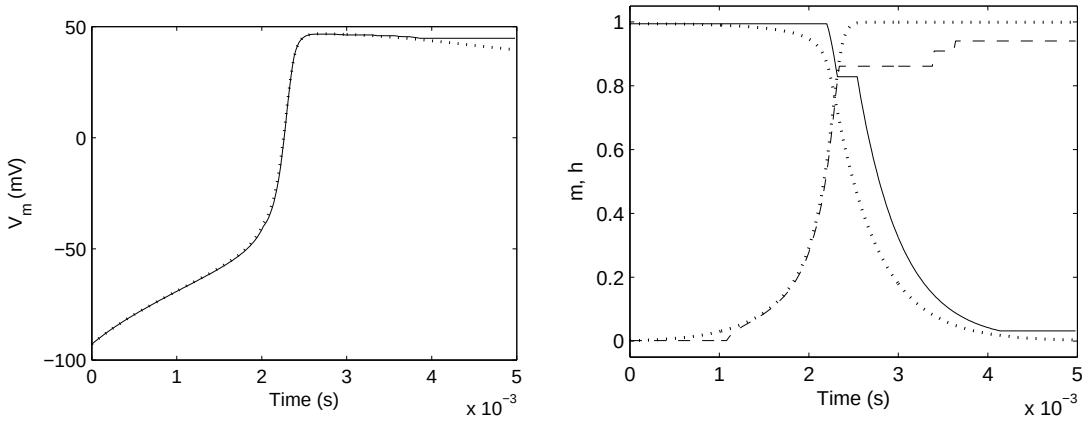


Figure 1:


50
0
−50
−100
V_m (mV)
0
0.5
1
1.5
2
2.5
3
Time (s)
x 10^−3
1
0.8
0.6
0.4
0.2
0
m, h
0
0.5
1
1.5
2
2.5
3
Time (s)
x 10^−3

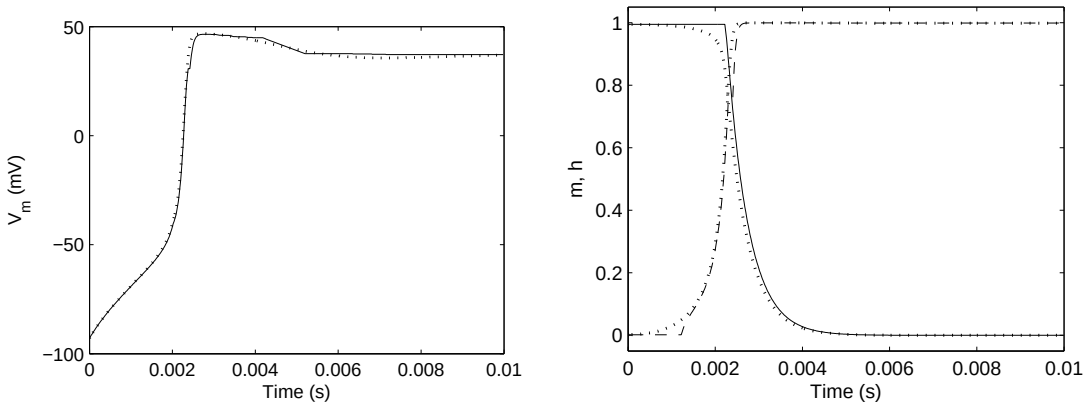
(a) (b)

Figure 2:


50
0
−50
−100
V_m (mV)
0
1
2
3
4
5
Time (s)
x 10^−3
1
0.8
0.6
0.4
0.2
0
m, h
0
1
2
3
4
5
Time (s)
x 10^−3

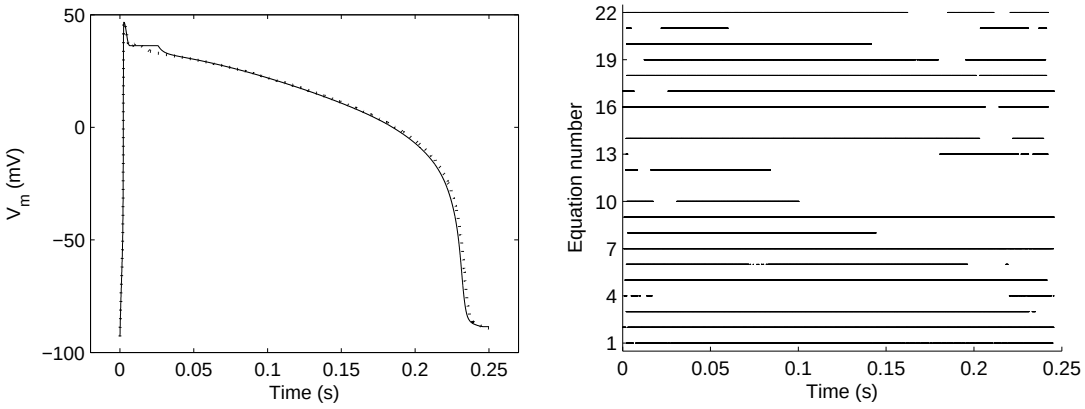
(a) (b)

Figure 3:


50
0
−50
−100
V m (mV)
0
0.002
0.004
0.006
0.008
0.01
Time (s)
1
0.8
0.6
0.4
0.2
0
m, h
0
0.002
0.004
0.006
0.008
0.01
Time (s)

(a) (b)

Figure 4:


50
0
−50
−100
V_m (mV)
0
0.05
0.1
0.15
0.2
0.25
Time (s)
22
19
16
13
10
7
4
1
Equation number
0
0.05
0.1
0.15
0.2
0.25
Time (s)

(a) (b)

Figure 5:

RECENT REPORTS

2009

01/09	A Mass and Solute Balance Model for Tear Volume and Osmolarity in The Normal And The Dry Eye	Gaffney Tiffany Yokoi Bron
02/09	Diffusion and permeation in binary solutions	Peppin
03/09	On the modelling of biological patterns with mechanochemical models: insights from analysis and computation	Moreo Gaffney Garcia-Aznar Doblare
04/09	Stability analysis of reaction-diffusion systems with timem-dependent coefficients on growing domains	Madzvamuse Gaffney Maini
05/09	Onsager reciprocity in premelting solids	Peppin Spannuth Wettlaufer
06/09	Inherent noise can facilitate coherence in collective swarm motion	Yates *et al.*
07/09	Solving the Coupled System Improves Computational Efficiency of the Bidomain Equations	Southern Plank Vigmond Whiteley
08/09	Model reduction using a posteriori analysis	Whiteley
09/09	Equilibrium Order Parameters of Liquid Crystals in the Laudau-De Gennes Theory	Majumdar
10/09	Landau-De Gennes theory of nematic liquid crystals: the Oseen-Frank limit and beyond	Majumdar Zarnescu
11/09	A Comparison of Numerical Methods used for Finite Element Modelling of Soft Tissue Deformation	Pathmanathan Gavaghan Whiteley
12/09	From Individual to Collective Behaviour of Unicellular Organisms: Recent Results and Open Problems	Xue Othmer Erban
13/09	Stochastic modelling of reaction-diffusion processes: algorithms for bimolecular reactions	Erban Chapman
14/09	Chaste: a test-driven approach to software development for physiological modelling	Pitt-Francis *et al.*

15/09	Block triangular preconditioners for PDE constrained optimization	Rees Stoll
16/09	From microscopic to macroscopic descriptions of cell migration on growing domains	Baker Yates Erban

Copies of these, and any other OCCAM reports can be obtained from:

Oxford Centre for Collaborative Applied Mathematics
Mathematical Institute
24 - 29 St Giles'
Oxford
OX1 3LB
England
www.maths.ox.ac.uk/occam