Annals of Biomedical Engineering manuscript No.
(will be inserted by the editor)

Stiffness analysis of cardiac electrophysiological models

Raymond J. Spiteri · Ryan C. Dean

Received: date / Accepted: date

Abstract The electrophysiology in a cardiac cell can be modelled as a system of ordinary differential equations. The efficient solution of these systems is important because they must be solved many times as sub-problems of tissue- or organ-level simulations of cardiac electrophysiology. The wide variety of existing cardiac cell models encompasses many different properties, including the complexity of the model and the degree of stiffness. Accordingly, no single numerical method can be expected to be the most efficient for every model. In this paper, we study the stiffness properties of a range of cardiac cell models and discuss the implications for their numerical solution. This analysis allows us to select or design numerical methods that are highly effective for a given model and hence outperform commonly used methods.

Raymond J. Spiteri · Ryan C. Dean

Department of Computer Science, University of Saskatchewan,

Tel.: +306-966-2909

Fax: +306-966-4884

E-mail: spiteri@cs.usask.ca · rcd144@mail.usask.ca

Keywords cardiac electrophysiology · stiff differential equation · Rush–Larsen method · Runge–Kutta methods

1 Introduction

The electrophysiological behaviour in myocardial tissue can be modelled by means of differential equations, often as a combination of ordinary and partial differential equations. Ionic currents at the myocardial cell level are described by a model consisting of ordinary differential equations (ODEs). These ionic currents are coupled via a model consisting of partial differential equations (PDEs) to describe the flow of electricity across the heart. A PDE model, such as the bidomain model, coupled with an ionic model can be used to simulate the electrical activity in the heart. For a thorough introduction, see [42] or [53].

Coupled ODE/PDE models are often solved separately via an algorithm called operator splitting. In this algorithm, the solution process alternates between solving the ODEs and the PDEs separately. In this context, the solution of each system is an important sub-problem of the overall solution process, each posing challenges to solving the overall problem efficiently. One such challenge is that the ionic model is usually a stiff, non-linear set of ODEs that must be solved for each node in the simulation. Moreover, a fine spatial discretization is required to produce useful data [57], and so another challenge is the magnitude of the system required for a realistic simulation. Inefficiencies in solving the ionic model, for example, are magnified as simulations become larger. The use of reduced, computationally inexpensive ionic models (e.g., [55]) illustrates the difficulty in efficiently performing simulations. In this study, we perform a systematic stiffness analysis of 37 verified ionic cell models from the CellML

model repository [26] and show how this analysis can lead to application or design of appropriate methods for their efficient numerical solution.

A wide range of ionic models exists. Many models are based upon the Hodgkin–Huxley (HH) model for the giant squid axon [21]. The FitzHugh–Nagumo (FHN) model [17,32], a simplification of the HH model with two ODEs, is often used as an inexpensive ionic model, albeit usually in a form modified to more accurately model cardiac action potentials; see, e.g., [2,24,43,56]. Modern models, on the other hand, usually add detail to the HH model or subsequent models; e.g., the model of Iyer et al. (2004) of human left-ventricular epicardial myocytes consists of 67 ODEs. There is also variety in the type of cardiac cell modelled, including models of cells in the atria, ventricles, sinoatrial node, and Purkinje fibres, as well as in the species being modelled, e.g., human, canine, rabbit, etc.

The wide variety of ionic models has significant implications for efficiently obtaining their numerical solution. The simplest models, such as FHN, are typically non-stiff [53]. In such cases, the use of a non-stiff numerical method is the most appropriate to obtain a solution efficiently. Cell models more detailed than FHN tend to be stiff and require the use of a stiff numerical method to obtain a solution efficiently [53]. However, stiffness is a subtle effect and challenging to definitively characterize. The distinction between stiff and non-stiff models is not clear cut in general; it depends on aspects such as the range of the time scales involved and the accuracy required of the approximate solution. We find there is a considerable difference in the degree of stiffness across the diversity of ionic models even for typical accuracy requirements. For example, [28] demonstrates the marked difference in the degree of stiffness between the models of Courtemanche et al. [11] and Winslow et al. [59]. Both are stiff, detailed, second-generation models, but the most well-suited numerical method is different in each case.

In this study, we analyse the stiffness of a wide range of cardiac electrophysiological models. We examine the eigenvalues of the Jacobian of the solutions over time for the 37 ionic cell models considered. These eigenvalues are often related to the stiffness of the model [19], and in fact a stiff initial-value problem is sometimes defined in terms of these eigenvalues, e.g., [25]. We show how this information can be used to select or design an effective numerical method for specific ionic cell models. We give special focus to numerical methods commonly used in cardiac electrophysiological simulations and demonstrate the efficiency gains that are possible through the use of stiffness data.

The rest of this paper is organized as follows. In Section 2, we give a brief outline of ionic cell models and an overview of the particular models used in this study. In Section 3, we give a brief overview of the concept of stiffness. We also present the extreme eigenvalue data generated for the models considered and discuss the implications for model stiffness. In Section 4, we use the eigenvalue data to select an effective numerical method for the model of Pandit et al. [39] as well as to design effective type-insensitive codes for several cell models. We compare the proposed numerical methods in terms of efficiency to commonly used methods. Finally, in Section 5, we summarize the conclusions reached based on our observations.

2 Cardiac electrophysiological models

2.1 Ionic current models

Small-scale processes that occur at the level of the individual cells in the heart can be modelled using ODEs. Such models can be used to simulate the behaviour of electrical activity in an isolated cell, or, when coupled with a PDE model, can be used to provide the ionic current to tissue- or organ-level simulations. The simplest of these models

are called *first-generation models*. A first-generation model contains enough detail to reproduce an action potential, but it has a simplified description of the underlying physiological details [53]. Some popular examples of first-generation models are the FHN model and the 1991 Luo–Rudy model (LR) [27]. More complex models, called *second-generation models*, contain all of the details included in first-generation models and as many fine-scaled physiological details as possible [53]. Most modern models can be classified as second-generation models because the most useful simulations tend to require details on the finest level [53]. Although first-generation models have less detail, their advantage is that they are computationally inexpensive relative to second-generation models.

The heart consists of different types of excitable cells, each having their own properties [42]. As such, most models are suited to particular type of cell in the heart. Normally, the electrical activity in the heart is initiated by a spontaneous electrical pulse emanating from specialized tissue called the sinoatrial node. From the sinoatrial node, electricity spreads to the atrial myocardium [42]. From the atria, the activation wave spreads to the atrioventricular node, the Purkinje fibres, and, finally, to the ventricles. There are models for each of these regions of the heart. In particular, there are numerous models of atrial and ventricular cells. On the other hand, there are relatively few cell models for the atrioventricular node; see, e.g., [42], for a detailed list of dozens of cardiac electrophysiological models classified in this way.

As an example of a first-generation model, we consider the LR model, sometimes called the Luo–Rudy Phase 1 model. The LR model describes guinea pig ventricular tissue and consists of 8 ODEs. For an individual cardiac cell, we have that the

transmembrane potential, V_m, satisfies [27]

$$\frac{dV_m}{dt} = -\frac{1}{C_m}(I_{ion} + I_{st}), \tag{1}$$

where C_m is the membrane capacitance, I_{ion} is the total transmembrane ionic current, and I_{st} is the stimulus current. The LR model contains six gating variables that determine the flow of current. The evolution of each gating variable y is governed by a nonlinear ODE involving rate parameters $\alpha_y = \alpha_y(V_m)$ and $\beta_y = \beta_y(V_m)$ in the general form

$$\frac{dy}{dt} = \frac{y_\infty - y}{\tau_y}, \tag{2}$$

where

$$y_\infty = \frac{\alpha_y}{\alpha_y + \beta_y} \quad \text{and} \quad \tau_y = \frac{1}{\alpha_y + \beta_y}.$$

The range of each gating variable is $[0, 1]$, representing the probability of a channel being open or closed. When $y = 0$, all channels are completely closed, allowing no current to flow. When $y = 1$, all channels are completely open, allowing current to flow without being inhibited by the gate [42]. The remaining ODE in the LR model describes calcium concentration in the cell:

$$\frac{d\,([\mathrm{Ca}]_i)}{dt} = -10^{-4} I_{\mathrm{si}} + 0.07\,(10^{-4} - [\mathrm{Ca}]_i), \tag{3}$$

where $[\mathrm{Ca}]_i$ is the intracellular calcium concentration and I_{si} is the slow inward calcium current [27]. The 6 gating equations of the form (2) are coupled with (1) and (3) to form the complete LR model. Full details of the model can be found in [27].

As an example of a second-generation model, we mention the model of Winslow et al. [59]. This model describes canine ventricular tissue and consists of 33 ODEs.

2.2 Models used

In this study, we consider 37 different ionic models, including some that are variations of a given model. This represents a wide variety of models in terms of type of cardiac cell, species modelled, degree of stiffness, and level of detail. Model data were obtained from the CellML model repository [26]. The CellML representation of models was used to generate Matlab code via a Python script. To ensure the faithfulness of the code to the model, two specific considerations were made. First, apart from one exception, we used only models considered to be faithful to the published model according to CellML's curation guidelines, i.e., models marked with a gold star on the CellML website. The exception was for the Puglisi–Bers model [41] because we already had reliable code as part of other work [48]. Second, reference solutions were obtained with the generated Matlab code and were verified to be the same as reference solutions obtained with JSim [1], an independent software package capable of parsing CellML files.

A detailed formulation of all models is omitted for the sake of brevity. The reader is instead referred to the original model papers or to the CellML repository, which contains a concise formulation of each model. A summary of each of the models used in this study is presented in Table 1 including, for each model, the name used for it in this paper, a reference to the original paper, a brief description of the model, and the number of ODEs in the model.

For each model, the set of default parameters in CellML was used. All models contain adjustable parameters; e.g., the model of Pandit et al. (2001) has a parameter that determines if the model should take into account the influence of diabetes. Because adjusting parameters to the models requires intricate knowledge of each of the model and increases the chance of human error, we did not vary any of these parameters

Table 1 Summary of models used. Three variants (endocardial cell, epicardial cell, and M-cell) exist for each of the models marked with an asterisk.

Model	Reference	ODEs	Description
Beeler–Reuter (1977)	[6]	8	Canine ventricular model
Bondarenko et al. (2004)	[8]	41	Mouse ventricular model
Courtemanche et al. (1998)	[11]	21	Human atrial model
Demir et al. (1994)	[13]	27	Rabbit sinoatrial node model
Demir et al. (1999)	[12]	29	Rabbit sinoatrial node model
DiFrancesco–Noble (1985)	[14]	16	Mammal Purkinje fibre model
Dokos et al. (1996)	[15]	18	Rabbit sinoatrial node model
Faber–Rudy (2000)	[16]	19	Guinea pig ventricular model
FitzHugh–Nagumo (1961)	[17,32]	2	Nerve membrane model
Fox et al. (2002)	[18]	13	Canine ventricular model
Hilgemann–Noble (1987)	[20]	15	Rabbit atrial model
Hund–Rudy (2004)	[22]	29	Canine ventricular model
Jafri et al. (1998)	[23]	31	Guinea pig ventricular model
Luo–Rudy (1991)	[27]	8	Guinea pig ventricular model
Maleckar et al. (2008)	[29]	30	Human atrial model
McAllister et al. (1975)	[31]	10	Canine Purkinje fibre model
Noble (1962)	[33]	4	Mammal Purkinje fibre model
Noble–Noble (1984)	[34]	15	Rabbit sinoatrial node model
Noble et al. (1991)	[35]	17	Guinea pig ventricular model
Noble et al. (1998)	[36]	22	Guinea pig ventricular model
Nygren et al. (1998)	[37]	29	Human atrial model
Pandit et al. (2001)	[38]	26	Rat left-ventricular model
Pandit et al. (2003)	[39]	26	Rat left-ventricular model
Puglisi–Bers (2001)	[41]	17	Rabbit ventricular model
Sakmann et al. (2000)*	[45]	21	Guinea pig ventricular model
Stewart et al. (2009)	[51]	20	Human Purkinje fibre model
Ten Tusscher et al. (2004)*	[54]	17	Human ventricular model
Ten Tusscher et al. (2006)*	[55]	19	Human ventricular model
Wang–Sobie (2008)	[58]	35	Neonatal mouse ventricular model
Winslow et al. (1999)	[59]	33	Canine atrial model
Zhang et al. (2000)	[61]	15	Rabbit sinoatrial node model

unless separate CellML files were provided for variations on the same model. This was the case for the model of Sakmann et al. (2000) and the two models of Ten Tusscher et al. studied here. For these three models, endocardial, epicardial, and M-cell variants of the model were in the CellML repository and thus included in this study.

3 Stiffness

We are concerned with the characterization of stiffness in various ionic cell models. Arguably, there is no universally accepted definition of stiffness. For example, Hairer and Wanner state that "stiff equations are problems for which explicit methods don't work" [19, pg. 2]. Lambert [25] describes stiffness as a phenomenon rather than a property because the concept of property implies the requirement of a precise mathematical definition. In this paper, we say that a problem is stiff with respect to a given method when stability requirements force the method to take a smaller step size than that dictated by accuracy requirements. This is similar to the definition of stiffness used by Ascher and Petzold [4]. This can be seen as a pragmatic definition because it frames stiffness in terms of the associated computational consequences. Generally, the step size required for a non-stiff method applied to a stiff problem is much smaller than accuracy requirements dictate, resulting in a numerical solution that is much more accurate (and hence more costly) than desired. For efficiency, we would like that a step size be chosen based only the accuracy requirements. For example, second-order methods have an error of $\mathcal{O}(\Delta t^2)$. So for an error tolerance `TOL` of 10^{-4}, a step size $\Delta t \approx 10^{-2}$ might suffice. However suppose the method has a stability restriction of $\Delta t = 10^{-6}$; i.e., step sizes $\Delta t > 10^{-6}$ are unstable. Then the error produced would be

around 10^{-12}, much smaller than desired. Because Δt is restricted by stability, there is no way to stably increase Δt to make the error match the tolerance more closely.

Despite the absence of a universally accepted definition, there is a large body of knowledge regarding the suitability of particular methods for both stiff and non-stiff problems. Numerical methods for the solution of ODEs can generally be put into two groups, stiff and non-stiff. The placement of a particular method in a certain group is based on the method's relative performance on stiff and non-stiff problems. Consider, for example, the forward Euler (FE) and backward Euler (BE) methods [4, ch. 3]. One step with FE is computationally inexpensive. So when the step size is dictated by accuracy considerations, FE is more efficient than BE. However, FE has a relatively small region of absolute stability [4, ch. 3]. An attempt to solve a stiff problem with FE requires a small step size, and overall the numerical solution process is less efficient than if we use a method with a larger stability region. Hence, FE is classified as a non-stiff method. On the other hand, BE has a large stability region. Despite the fact that each step of BE is more expensive than FE because a system of nonlinear equations must generally be solved at each step, the overall performance trade-off is favourable: the step size can be increased by more than enough to offset the extra cost per step. Hence, BE is classified as a stiff method. This analysis illustrates that choosing the proper type of method to solve the ODEs can be critical to achieve good performance.

Related to the stiffness of a general initial-value problem (IVP)

$$\frac{d\mathbf{y}}{dt} = \mathbf{f}\left(t, \mathbf{y}\right), \quad \mathbf{y}(0) = \mathbf{y}_0, \quad t \in [t_0, t_f], \tag{4}$$

are the eigenvalues of the Jacobian $\mathbf{J} = \frac{\partial \mathbf{f}}{\partial \mathbf{y}}(t, \mathbf{y})$ over time. These eigenvalues can give us an indication of a problem's stiffness. In particular, eigenvalues with large negative real parts are likely to lead to stiff problems on their corresponding time intervals.

Problems that have eigenvalues with large imaginary parts also tend to be difficult to solve by standard solvers, but the highly oscillatory nature of the solutions to the associated linearized problem does not make the problems stiff according to the classical description of stiffness. In Section 3.1, we compute the eigenvalues of the Jacobian over time for each of the models introduced in Section 2.2 and discuss the implications for stiffness. In Section 3.2, we discuss numerical experiments to illustrate our discussion of stiffness.

3.1 Eigenvalue Data

For each model in Section 2.2, we found the eigenvalues of the Jacobian over time in the following way. First, a reference solution was generated using Matlab's `ode15s` [30]. This was done by lowering the error tolerances for successive approximations until two approximations were identical for at least 10 significant digits at $N^* = 100$ equally spaced output points. Matlab code representing the derivative of each model was created via automatic differentiation using AdiMat [7]. We found a value for the Jacobian at every 1 ms of simulated time using the derivative code with the reference solution, and we found the eigenvalues of each of these Jacobians with Matlab's `eig` function.

The extreme values in the set of eigenvalues associated with a model are of particular interest because they offer a worst-case or extreme look at the stiffness properties. The largest stable step size for the FE method is approximately two times the reciprocal of the extreme negative real eigenvalue. At each point in time considered, we found the maximum and minimum values of both the real and complex parts of the eigenvalues. The extreme values across the solution interval of these minimum and maximum values are reported in Tables 2 and 3 along with the percentage of time at least one pair

of complex eigenvalues was present. As we can see, there is a wide range of behaviours encompassed by these models, from models such as FHN and Beeler–Reuter that do not have relatively large negative eigenvalues to models such as Wang–Sobie and that of Maleckar et al. that have moderately large negative eigenvalues to the model of Pandit et al. (2003) that has extremely large negative eigenvalues.

Although not apparent from these tables, we note that problems may not be stiff everywhere in their respective intervals of integration. Accordingly it may be possible to use different integration methods that are more appropriate on different time intervals. We also note integration methods that use a constant step size are subject to the constraints imposed by the worst-case behaviour of stiffness and hence tend to perform poorly. Graphs of how the extreme eigenvalues vary over time for two models (Pandit et al. (2003) and Winslow et al.) are given in Figures 1 and 2 below; graphs for the remaining models can be found in [49].

3.2 Numerical Experiments

3.2.1 Overview

We now discuss numerical experiments to show how the data presented in Section 3.1 can be utilized. We focus on numerical experiments for 3 integration methods for ionic cell models: FE, the Rush–Larsen (RL) [44] method, and the recently proposed second-order generalization of RL (GRL2) [52]. The primary goal of these experiments is not to find the most efficient numerical method possible but rather to illustrate how the stiffness of particular models affects the performance of different numerical methods. A secondary goal of these experiments is to demonstrate the suitability of numerical methods to particular ionic models according to their degree of stiffness.

Table 2 Extreme values of the eigenvalues for each model. The minimum real part of the set of eigenvalues is denoted $\min(Re(\lambda))$, and the maximum real part of the set of eigenvalues is denoted $\max(Re(\lambda))$. Similarly, the minimum and maximum imaginary parts are denoted $\min(Im(\lambda))$ and $\max(Im(\lambda))$. The percentage of the solution interval in which there is at least one pair of complex eigenvalues is also reported.

Model	$\min(Re(\lambda))$	$\max(Re(\lambda))$	$\min(Im(\lambda))$	$\max(Im(\lambda))$	% Complex
Beeler–Reuter (1977)	–8.20E+1	–3.968E–3	0.00E+0	0.00E+0	0
Bondarenko (2004)	–8.49E+3	4.51E+0	–2.80E+0	2.80E+0	64
Courtemanche et al. (1998)	–1.29E+2	1.87E–1	–4.50E+0	4.50E+0	82
Demir et al. (1994)	–2.24E+4	4.57E+0	–7.35E+0	7.35E+0	100
Demir et al. (1999)	–3.45E+4	4.81E+2	–6.50E+1	6.50E+1	66
DiFrancesco–Noble (1985)	–2.62E+4	8.24E+1	–3.21E+0	3.21E+0	31
Dokos et al. (1996)	–2.99E+4	2.49E–15	–6.39E+1	6.39E+1	100
Faber–Rudy (2000)	–1.83E+2	1.36E–17	0.00E+0	0.00E+0	0
FitzHugh–Nagumo (1961)	–4.38E–1	1.78E–1	–4.59E–2	4.59E–2	45
Fox et al. (2002)	–4.38E+2	4.44E–2	–4.18E–1	4.18E–1	65
Hilgemann–Noble (1987)	–2.86E+4	1.81E–14	–7.71E+1	7.71E+1	21
Hund (2004)	–1.95E+2	2.69E–2	–1.57E–3	1.57E–3	4
Jafri et al. (1998)	–1.12E+3	2.31E–7	–1.91E–2	1.91E–2	52
Luo–Rudy (1991)	–1.51E+2	7.01E–2	–4.11E–2	4.11E–2	74
Maleckar et al. (2008)	–4.16E+4	2.42E+2	–3.42E+2	3.42E+2	28
McAllister et al. (1975)	–8.18E+1	–4.79E–4	–2.85E–2	2.85E–2	100
Noble (1962)	–9.79E+3	–1.92E+0	0.00E+0	0.00E+0	0
Noble–Noble (1984)	–6.56E+3	1.35E–15	–1.36E+1	1.36E+1	100
Noble et al. (1991)	–3.88E+4	1.78E–12	0.00E+0	0.00E+0	0
Noble et al. (1998)	–3.60E+4	–6.32E–7	–8.41E+0	8.41E+0	9
Nygren et al. (1998)	–4.03E+4	1.22E–1	–3.88E+2	3.88E+2	22
Pandit et al. (2001)	–8.89E+4	2.20E–14	0.00E+0	0.00E+0	0
Pandit et al. (2003)	–3.90E+9	6.83E+0	–8.09E–5	8.09E–5	17
Puglisi–Bers (2001)	–1.67E+1	1.29E+0	–1.52E–1	1.52E–1	35
Sakmann et al. (2000) – Endocardial	–2.93E+4	6.02E–1	–5.21E+1	5.21E+1	100
Sakmann et al. (2000) – Epicardial	–2.93E+4	3.59E+1	–5.24E+1	5.24E+1	100
Sakmann et al. (2000) – M-cell	–2.93E+4	1.03E+2	–5.20E+1	5.20E+1	100
Stewart et al. (2009)	–1.38E+2	3.34E+0	–1.56E+0	1.56E+0	92

Table 3 Extreme values of the eigenvalues for each model. The minimum real part of the set of eigenvalues is denoted $\min(Re(\lambda))$, and the maximum real part of the set of eigenvalues is denoted $\max(Re(\lambda))$. Similarly, the minimum and maximum imaginary parts are denoted $\min(Im(\lambda))$ and $\max(Im(\lambda))$. The percentage of the solution interval in which there is at least one pair of complex eigenvalues is also reported.

Model	$\min(Re(\lambda))$	$\max(Re(\lambda))$	$\min(Im(\lambda))$	$\max(Im(\lambda))$	% Complex
Ten Tusscher et al. (2004) – Endocardial	–1.17E+3	1.16E–1	–4.67E+0	4.67E+0	17
Ten Tusscher et al. (2004) – Epicardial	–1.17E+3	1.12E–1	–4.73E+0	4.73E+0	18
Ten Tusscher et al. (2004) – M-cell	–1.16E+3	1.12E–1	–4.73E+0	4.73E+0	22
Ten Tusscher et al. (2006) – Endocardial	–1.26E+3	1.94E–8	0.00E+0	0.00E+0	0
Ten Tusscher et al. (2006) – Epicardial	–1.26E+3	1.92E–8	0.00E+0	0.00E+0	0
Ten Tusscher et al. (2006) – M-cell	–1.26E+3	1.92E–8	0.00E+0	0.00E+0	0
Wang–Sobie (2008)	–1.22E+2	1.23E+0	–1.23E+0	1.23E+0	46
Winslow et al. (1999)	–1.84E+4	1.53E+0	–4.22E–1	4.22E–1	63
Zhang et al. (2000)	–2.22E+4	1.29E+2	–1.00E+2	1.00E+2	89

For each numerical method, we aim to balance the requirements for accuracy and efficiency. To quantify the error in a solution, we use the *relative root mean square error* (RRMS) error of the transmembrane potential:

$$RRMS := \sqrt{\frac{1}{N^*}\frac{\sum_{i=1}^{N^*}(V_i - \hat{V}_i)^2}{\sum_{i=1}^{N^*}\hat{V}_i^2}},$$

where V_i is the numerical approximation and $\hat{V}_i$ is the reference solution at time t_i as described above. For each model and numerical method considered, we maximize the step size while producing a solution that has less than 5% RRMS error. For each combination of numerical method and model, we report the result for the step size with the least execution time and less than 5% RRMS error.

We used constant step sizes in our experiments to reflect the typical use of the cell models within a tissue-scale simulation. As mentioned, these simulations employ

operator splitting, and constant equal step sizes are often used for integrating both the PDEs and the ODEs. The use of variable step sizes for the integration of the ODEs over long time intervals can generally be expected to be more efficient than the use of constant steps [48] and hence would likely be more effective in a scenario where a fully coupled (i.e., unsplit) integration approach is used; see e.g., [60]. In the context of operator splitting, variable step sizes for the ODEs may be allowed provided they remain below the (constant) step size used for the PDEs.

We placed two additional requirements on the step size. First, the maximum step size allowed was equal to the length of time in which the stimulus current was applied. The cases for which this maximum was reached are denoted with a dagger in Tables 4 and 5. Second, the step size was adjusted at up to three points in time in order to resolve important events: the start of the application of stimulus current, the end of the application of stimulus current, and the end point of the simulation. If the integration were to step past one of these three points, the step size would be adjusted, for that step only, to land on the point exactly. This was done mainly because a discontinuous stimulus application can introduce errors that are avoidable if the points at which the discontinuities occur are resolved.

3.2.2 Methods

The first method used in our experiments is the explicit first-order FE method. Applied to the general IVP (4), one step of FE from $(t_{n-1}, \mathbf{y}_{n-1})$ to $(t_n, \mathbf{y}_n)$ is given by

$$\mathbf{y}_n = \mathbf{y}_{n-1} + \Delta t_n \mathbf{f}\left(t_{n-1}, \mathbf{y}_{n-1}\right),$$

$$t_n = t_{n-1} + \Delta t_n,$$

where $\mathbf{y}_n \approx \mathbf{y}(t_n)$ and $t_n = t_{n-1} + \Delta t_n$. It is a popular method in practice mainly due to the ease of implementation.

The second method is the RL method. The RL method advances the solution to the gating equations (2) associated with HH variables using

$$y_n = y_\infty + (y_{n-1} - y_\infty) e^{-\frac{\Delta t_n}{\tau_y}},$$

which represents the exact solution of (2) assuming all variables besides y are constant. FE is then used to advance the solution of the remaining equations, including those associated with non-HH state variables and Markov-type models of currents. This method is an effective stiff solver for the Luo–Rudy model [50]; i.e., the step size can be chosen based on accuracy considerations. RL is generally one of the most popular methods in practice due to its good stability properties and ease of implementation. However, this method is only first-order accurate and thus suffers from the usual inefficiencies of low-order methods when high accuracy is required.

The third method is the GRL2 method proposed by Sundnes et al. [52]. GRL2 decouples and linearizes the ODE system consisting of m ODEs around a point $\mathbf{y} = \mathbf{y}_n$ at time $t = t_n$ to obtain

$$\frac{dy_i}{dt} = f_i(\mathbf{y}_n) + \frac{\partial}{\partial y_i} f_i(\mathbf{y}_n) \left(y_i - y_{n,i}\right), \quad y_i(t_n) = y_{n,i}, \tag{5}$$

for $i = 1, 2, \ldots, m$, where the subscript i denotes component i of a vector. The exact solution of (5) is given by

$$y_i(t) = y_{n,i} + \frac{a}{b}\left(e^{b(t-t_n)} - 1\right), \quad i = 1, 2, \ldots, m, \tag{6}$$

where $a = f_i(\mathbf{y}_n)$ and $b = \partial f_i(\mathbf{y}_n)/\partial y_i$. The numerical solution $\mathbf{y}_{n+1}$ at time $t = t_{n+1}$ is then obtained in two steps:

1. Estimate the solution at time $t_{n+1/2}$ with

$$y_{n+1/2,i} = y_{n,i} + \frac{a}{b}\left(e^{b(\Delta t_n/2)} - 1\right), \quad i = 1, 2, \ldots, m.$$

2. Let $\bar{\mathbf{y}}_{n+1/2} = \mathbf{y}_{n+1/2}$ but with component i replaced by $y_{n,i}$. For each i, compute the numerical solution at time t_{n+1} from

$$y_{n+1,i} = y_{n,i} + \frac{\bar{a}}{\bar{b}}\left(e^{\bar{b}\Delta t_n} - 1\right), \quad i = 1, 2, \ldots, m,$$

where $\bar{a} = f_i(\bar{\mathbf{y}}_{n+1/2})$, $\bar{b} = \partial f_i(\bar{\mathbf{y}}_{n+1/2})/\partial y_i$ and we have used the fact that $\bar{\mathbf{y}}_{n+1/2,i} = y_{n,i}$.

GRL2 and RL treat the gating equations (2) in a similar manner. The main difference is that GRL2 also treats the non-gating equations with an exponential formula based on local linearization, whereas RL uses FE. GRL2 is verified to have second-order accuracy in [3].

We note that care must be taken in the implementation of GRL2 to ensure efficiency, in particular regarding the computation of the Jacobian $\partial\mathbf{f}/\partial\mathbf{y}$ because only the diagonal elements are needed. For example, the finite-difference approximation of $\partial f_i(\mathbf{y})/\partial y_i$ is performed via

$$\partial f_i(\mathbf{y})/\partial y_i \approx \frac{f_i(y_1, \ldots, y_{i-1}, y_i + \delta, y_{i+1}, \ldots, y_m) - f_i(\mathbf{y})}{\delta},$$

where $\delta = 10^{-8}$ for double-precision calculations. Without careful implementation, this could add another m full ODE right-hand side function evaluations per step, making the method prohibitively expensive for all but the simplest models. Moreover, we note that a full ODE right-hand-side function evaluation is not needed for each $\partial f_i(\mathbf{y})/\partial y_i$. Also in practice, if $|\partial f_i(\mathbf{y})/\partial y_i| < \delta$, the limit as $\partial f_i(\mathbf{y})/\partial y_i \to 0$ is used instead of (6):

$$y_i(t) = y_i + a(t - t_n), \quad i = 1, \ldots, m.$$

We illustrate a specific implementation of GRL2 on the subsystem describing the ryanodine-sensitive release channel from the model of Winslow et al. [59]. The ODEs considered are

$$\begin{aligned}
\frac{dP_{C_1}}{dt} &= -k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N P_{C_1} + k_a^- P_{O_1}, \\
\frac{dP_{O_1}}{dt} &= k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N P_{C_1} - k_a^- P_{O_1}, -k_b^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^M P_{O_1} \\
&\quad + k_b^- P_{O_2} - k_c^+ P_{O_1} + k_c^- P_{C_2}, \\
\frac{dP_{O2}}{dt} &= k_b^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^M P_{O_1} - k_b^- P_{O_2}, \\
\frac{dP_{C_2}}{dt} &= k_c^+ P_{O_1} - k_c^- P_{C_2},
\end{aligned}$$

where the quantities M, N, and $k_{a,b,c}^{\pm}$ are constants; their values can be found in [59]. For the purposes of this example, the subspace calcium concentration $[\mathrm{Ca}^{2+}]_{\mathrm{ss}}$ is also assumed to be constant.

One step of size Δt_n for GRL2 applied to this system starting from time t_n and state $\mathbf{y}_n = (P_{C_1,n}, P_{O_1,n}, P_{O_2,n}, P_{C_2,n})^T$ yields

$$P_{C_1,n+1/2} = P_{C_1,n} + \frac{-k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N P_{C_1,n} + k_a^- P_{O_1,n}}{-k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N} \left[e^{-k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N \Delta t_n/2} - 1\right],$$

with analogous expressions for $P_{O_1,n+1/2}, P_{O_2,n+1/2}, P_{C_2,n+1/2}$; then

$$P_{C_1,n+1} = P_{C_1,n} + \frac{-k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N P_{C_1,n} + k_a^- P_{O_1,n+1/2}}{-k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N} \left[e^{-k_a^+[\mathrm{Ca}^{2+}]_{\mathrm{ss}}^N \Delta t_n} - 1\right],$$

with analogous expressions for $P_{O_1,n+1}, P_{O_2,n+1}, P_{C_2,n+1}$.

For the results reported here for GRL2, we used a finely tuned version of the cell model code. This code had the ability to return the value of the right-hand side of any particular ODE in the system and would compute only the information required for the individual ODE. This fine tuning was necessary for GRL2 to be competitive.

3.2.3 Results

For each pair of a model given in Section 2.2 and a method given in Section 3.2.2, we find a maximum step size to 3 significant digits subject to the conditions given in Section 3.2.1. With this information, we performed an experiment to determine the best execution time possible for combinations of a model with a method. All numerical experiments were performed in Matlab on an iMac with 2.8 GHz Intel Core Duo processor with 4 GB DDR SD RAM running at 667 MHz. Except for the model of Pandit et al. (2003), we report the minimum of 100 runs of each combination of a method and model. Because the model of Pandit et al. (2003) required significant run times with the methods used, the execution time reported is the minimum of 10 runs. In all cases, we ensured that the variance of the times recorded for a given combination was small.

Results for FE, RL, and GRL2 are presented in Tables 4 and 5. The non-standard methods RL and GRL2 generally outperform FE. Of the models considered, RL is the most efficient in 18 cases, GRL2 is the most efficient in 15 cases, and FE is the most efficient in 4 cases. GRL2 can usually take a larger step than RL, which can in turn usually take a larger step than FE. However, GRL2 generally has a higher cost per step than RL, which in turn has a higher cost per step than FE. We see that generally the increases in acceptable step sizes for GRL2 and RL are large enough to more than offset the added cost per step when compared with FE. The situation for GRL2 compared with RL is less clear, with each being the most efficient on roughly an equal number of models. These results support the findings in [52] for the competitiveness of GRL2 as a method for the integration of cell models, albeit at the expense of a more involved implementation. FE was the most efficient method on four models: FHN, Noble (1962), Pandit et al. (2001), and Pandit et al. (2003). In these cases, despite the increase in

maximum step size, RL and GRL2 significantly underperform compared to FE due to their higher costs per step.

We repeated the experiments using a tighter tolerance of 1% RRMS error. As can be expected, except for the cases when the largest allowable step size was not constrained by accuracy, the execution times were slightly higher. The only changes to the relative rankings of the methods for a given model were that FE became the most efficient on 3 more models, those of DiFrancesco–Noble, Maleckar et al,, and the endocardial variant of Sakmann et al. (2000). We also performed experiments with other explicit or implicit-explicit Runge–Kutta methods with more stages and/or higher order. However, as could have been anticipated based on the stiffness characteristics of the models and previous investigations [48], all of these method generally underperformed relative to FE and hence are not reported here. Full details can be found in [49].

4 Using stiffness data

4.1 Stiff Solvers

The model of Pandit et al. (2003) is an example of highly stiff model that cannot be integrated efficiently by any of the methods considered here thus far. These methods require between about 11 and 25 hours of execution time to integrate this model. As can be seen from Tables 2 and 3, there exists a negative real eigenvalue that is some 5 orders of magnitude larger than any of the other models considered here. Moreover, from Figure 1 we see that this large negative eigenvalue is present for most of the solution interval. Classical non-stiff methods, such as explicit Runge–Kutta methods including FE, are unlikely to perform well on such a model under normal accuracy requirements. The results presented in Section 3.2.3 demonstrate clearly that the non-

Table 4 Step size, in milliseconds, and execution time, in seconds, of the three numerical methods using the largest step size with less than 5% RRMS error.

Model	FE		RL		GRL2	
	Δt	Time	Δt	Time	Δt	Time
Beeler–Reuter (1977)	2.53E–2	5.19E–2	8.49E–1	**9.83E–3**	8.58E–1	3.09E–2
Bondarenko et al. (2004)	2.13E–4	3.81E+0	2.50E–4	4.13E+0	1.40E–2	**1.85E+0**
Courtemanche et al. (1998)	1.94E–2	4.27E+0	3.45E–1	4.79E–1	9.95E–1	**2.07E–1**
Demir et al. (1994)	5.95E–2	1.74E–2	1.53E–1	**7.48E–3**	2.86E–1	6.76E–2
Demir et al. (1999)	5.97E–2	2.13E–2	9.66E–2	**1.71E–2**	2.49E–1	8.24E–2
DiFrancesco–Noble (1985)	7.85E–2	8.59E–2	6.62E–1	**7.66E–2**	9.99E–1	2.67E–1
Dokos et al. (1996)	7.30E–2	3.44E–2	7.64E–1	**3.37E–3**	9.99E–1	8.35E–2
Faber–Rudy (2000)	1.12E–2	1.69E–1	9.51E–1	**4.60E–3**	6.35E–1	1.38E–1
FitzHugh–Nagumo (1961)	5.00E–1†	**3.93E–3**	N/A	N/A	5.00E–1†	3.19E–2
Fox et al. (2002)	4.60E–3	4.11E–1	6.37E–1	**3.11E–2**	7.74E–1	7.70E–2
Hilgemann–Noble (1987)	6.25E–2	1.69E–1	8.06E–2	9.56E–2	9.96E–2	**6.23E–2**
Hund–Rudy (2004)	1.11E–2	1.75E–1	4.17E–2	**5.35E–2**	6.20E–1	4.26E–1
Jafri et al. (1998)	5.77E–4	6.18E+0	5.89E–4	**4.42E+0**	1.59E–3	8.38E+0
Luo–Rudy (1991)	1.34E–2	3.61E–1	2.50E–1	6.94E–2	1.00E+0†	**2.31E–2**
Maleckar et al. (2008)	5.02E–2	1.09E–1	7.50E–1	**9.44E–2**	8.21E–1	3.84E–1
McAllister et al. (1975)	2.46E–2	1.21E–1	7.06E–1	**6.00E–2**	2.63E–1	2.32E–1
Noble (1962)	2.03E–1	**7.63E–3**	9.41E–2	8.82E–2	2.22E–1	7.89E–2
Noble–Noble (1984)	2.04E–1	1.82E–1	2.73E–1	1.20E–1	1.00E+0†	**4.99E–2**

standard methods RL and GRL2 are also not sufficiently stable to integrate such a model efficiently. For this model, FE in fact outperforms RL and GRL2 because they face similar extreme restrictions on their largest stable step sizes while being more expensive per step.

Table 5 Step size, in milliseconds, and execution time, in seconds, of the three numerical methods using the largest step size with less than 5% RRMS error.

Model	FE		RL		GRL2	
	Δt	Time	Δt	Time	Δt	Time
Noble et al. (1991)	5.15E–2	3.00E–2	1.53E–1	1.21E–2	6.25E–1	**9.31E–3**
Noble et al. (1998)	5.58E–2	5.25E–2	1.57E–1	**1.50E–2**	5.43E–1	2.67E–2
Nygren et al. (1998)	5.36E–2	1.02E–1	8.88E–2	8.68E–2	9.81E–1	**3.65E–2**
Pandit et al. (2001)	2.91E–4	**8.85E+0**	2.91E–4	1.01E+1	4.98E–4	2.56E+1
Pandit et al. (2003)	1.04E–7	**4.20E+4**	1.04E–7	8.37E+4	9.28E–7	8.93E+4
Puglisi–Bers (2001)	1.08E–2	4.43E–1	4.30E–1	**8.35E–2**	6.24E–1	4.18E–1
Sakmann et al. (2000) – Endocardial	6.90E–2	4.73E–2	2.36E–1	1.80E–2	9.71E–1	**1.03E–2**
Sakmann et al. (2000) – Epicardial	6.90E–2	5.11E–2	2.36E–1	1.80E–2	9.71E–1	**1.03E–2**
Sakmann et al. (2000) – M-cell	6.86E–2	5.12E–2	2.36E–1	1.80E–2	9.71E–1	**1.03E–2**
Stewart et al. (2009)	1.50E–2	4.35E–1	1.62E–1	**5.60E–2**	3.58E–2	8.61E–2
Ten Tusscher et al. (2004) – Endocardial	1.78E–3	1.75E+0	1.00E+0†	**5.43E–3**	1.00E+0†	1.30E–2
Ten Tusscher et al. (2004) – Epicardial	1.78E–3	1.75E+0	1.00E–1†	**5.43E–3**	1.00E+0†	1.30E–2
Ten Tusscher et al. (2004) – M-cell	1.76E–3	1.25E+0	2.80E–1	**1.29E–2**	9.81E–1	2.84E–2
Ten Tusscher et al. (2006) – Endocardial	1.62E–3	1.33E+0	1.52E–1	6.41E–2	7.80E–1	**9.27E–3**
Ten Tusscher et al. (2006) – Epicardial	2.14E–3	9.79E–1	2.80E–1	3.64E–2	8.39E–1	**7.06E–3**
Ten Tusscher et al. (2006) – M-cell	2.14E–3	9.79E–1	2.05E–1	4.17E–2	7.77E–1	**9.27E–3**
Wang–Sobie (2008)	1.66E–2	5.95E–2	5.26E–2	2.19E–2	4.97E–1	**1.61E–2**
Winslow et al. (1999)	1.07E–4	1.70E+1	2.80E–4	8.84E+0	1.34E–3	**2.26E+0**
Zhang et al. (2000)	9.90E–2	5.19E–2	1.00E+0†	**7.69E–3**	1.00E+0†	1.85E–1

The eigenvalue data indicate that solvers suited to stiff problems are ideal for this model. The simplest example of a stiff solver is the backward Euler (BE) method. BE

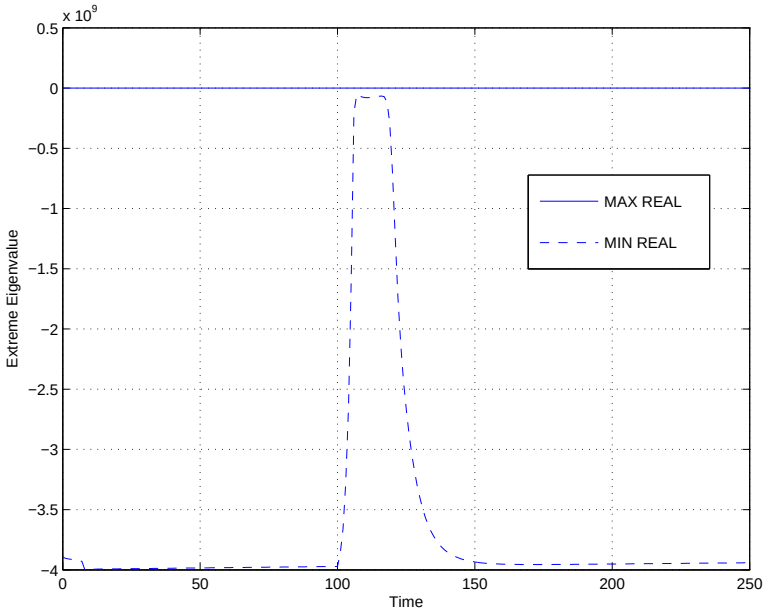

Fig. 1 Extreme eigenvalues of the model of Pandit et al. (2003).

is the one-stage, first-order implicit Runge–Kutta method given by

$$\mathbf{y}_n = \mathbf{y}_{n-1} + \Delta t_n \mathbf{f}\left(t_n, \mathbf{y}_n\right), \tag{7}$$

$$t_n = t_{n-1} + \Delta t_n.$$

BE is an implicit method because the unknown $\mathbf{y}_n$ appears on both sides of (7), and hence the set of nonlinear algebraic equations defined by (7) must be solved at each step to obtain the solution at time t_n. In general, these equations are solved using Newton's method or one of its variants [4, p.145]. Using the notation introduced in this paper, an outline of the BE algorithm used in our study is as follows:

Input: ODE right-hand side $\mathbf{f}(t, \mathbf{y})$ and Jacobian $\mathbf{J}(t, \mathbf{y})$, initial time t_0, final time t_f, initial condition $\mathbf{y}_0$, time step Δt, error tolerance `TOL`

Output: A matrix of approximate solution vectors $\mathbf{y}$

$t = t_0$;

$\mathbf{y}(1,:) = \mathbf{y}_0$;

Compute the number of steps needed, n_{steps};

for $n = 1$ to n_{steps} **do**

Compute the initial guess with one step of FE: $\mathbf{y}_{n+1}^{(0)} = \mathbf{y}_n + \Delta t\, \mathbf{f}(t_n, \mathbf{y}_n)$;

Initialize iteration counter $\nu = 1$;

Initialize error $e = \texttt{TOL} + 1.0$;

while $e > \texttt{TOL}$ **and** $\nu < \nu_{\max}$ **do**

Compute Newton iterate for (7): $\mathbf{y}_{n+1}^{(\nu)} = \mathbf{y}_{n+1}^{(\nu-1)} - \mathbf{J}^{-1}(t_n, \mathbf{y}_{n+1}^{(\nu-1)})\mathbf{f}(t_n, \mathbf{y}_{n+1}^{(\nu-1)})$;

$e = \|\mathbf{y}_{n+1}^{(\nu)} - \mathbf{y}_{n+1}^{(\nu-1)}\|_\infty$;

$\nu = \nu + 1$;

end while

if $\nu = \nu_{\max}$ **then**

Exit with failure;

end if

Save solution vector $\mathbf{y}(n+1,:) = \mathbf{y}_{n+1}^{(\nu)}$;

Update time $t = t + \Delta t$;

end for

In our implementation, we use $\nu_{\max} = 10$.

We performed the following experiment to compare BE to the 3 other methods used thus far in this study. As in Section 3.2, we determined the best execution time for BE when solving the model of Pandit et al. (2003) subject to the conditions on the RRMS error and stimulus duration described previously. The software we used was based on the code available at [9] but with the following 3 significant enhancements. First, the code was generalized to solving systems of equations. Second, the code was generalized

to use Matlab's `numjac` function to compute partial derivatives. Third, the original Newton solver was re-written using material from [5]. The initial guess for $\mathbf{y}_n$ used at the start of each step was provided by a FE step, and the Jacobian is recalculated and factored at each Newton iteration. The BE method was deemed to have converged at a given step when the norm of the difference between successive iterates was less than 10^{-5}. It is difficult to make many categorical statements about poor convergence of a Newton iteration; the classical causes range from an insufficiently accurate initial guess to an ill-conditioned iteration matrix. Our code lacks a few state-of-the-art options, in particular, a lack of step-size control, globalized Newton iterations, and the ability to freeze the Jacobian between iterations and/or time steps, all of which can potentially aid convergence and/or improve efficiency. Accordingly, despite these enhancements, a more sophisticated BE code should be able to produce even better results in some cases than those that we report below.

We find that BE takes less than 20 seconds to integrate this model, almost 2300 times faster than the most efficient and almost 4500 times faster than the least efficient of the methods considered. These results confirm what could have been inferred from the eigenvalue data: we need a stiff solver in order to obtain a solution of the model of Pandit et al. (2003) efficiently. Further efficiency gains for this model may be possible through the use of higher-order stiff solvers or other non-standard methods as in, e.g., [48]; such considerations are however beyond the scope of this study.

In order to investigate the efficacy of BE to other models in our study, we repeated the above experiment with the models of Courtemanche et al., Faber et al., Noble (1962), Noble et al. (1998), and Winslow et al. The stiffness data for these models encompasses a wide of behavior and none of the original methods tested (FE, RL, and GRL2) is most efficient on all models. We found that for all 5 additional models at

both 5% and 1% RRMS error levels, BE was always slower than the fastest method we used in our original study.

There can be expected to be a trade-off between model stiffness and error tolerance in any given simulation. In general, as the tolerance is lowered, the model must be increasingly stiff over significant portions of the interval of integration to benefit from a fully implicit solver such as BE with constant step sizes applied to the entire interval of integration. Cell models often do not appear to be highly stiff over much of their intervals of integration; rather they are mildly stiff in localized regions. This observation suggests that the use a combination of stiff and non-stiff solvers (i.e., a type-insensitive solver) may be an attractive approach for many cell models.

4.2 Type-insensitive methods

As discussed in Section 3, obtaining a solution efficiently for a given ODE requires the right type of method, i.e., stiff or non-stiff, and determining the right type of method can be challenging. It is also possible that an ODE is stiff in some intervals of the integration and non-stiff in others. To address these problems, *type-insensitive* solvers have been proposed. A type-insensitive solver consists of both a stiff and a non-stiff solver and a mechanism to automatically determine which method is more appropriate to use at every step. Often a type-insensitive solver estimates the largest step size that can be taken by both the stiff and non-stiff methods after analysing constraints due to stability and accuracy. This strategy can be seen in [40,46]. Another frequently used technique is to determine the stiffness of the ODE by estimating a Lipschitz constant for the problem, e.g., [10,47].

A significant cost when using a type-insensitive solver for an arbitrary ODE is determining whether and, if applicable, in which solution interval the ODE is stiff. There is also a risk that the type-insensitive solver selects the wrong method. In this case, the efficiency of the solution process is hindered by both the inappropriate choice of method as well as the cost to make the decision to use it. However, using data such as those presented in Figure 2, one can mitigate both the cost of determining stiffness as well as the risk of selecting the wrong method. With these data, it is relatively clear where each individual problem is stiff. Accordingly a type-insensitive solver can be hard-coded to always use the right type of method at each point in the solution interval, thus all but eliminating the additional costs associated with stiffness determination and risk of choosing the less-effective method.

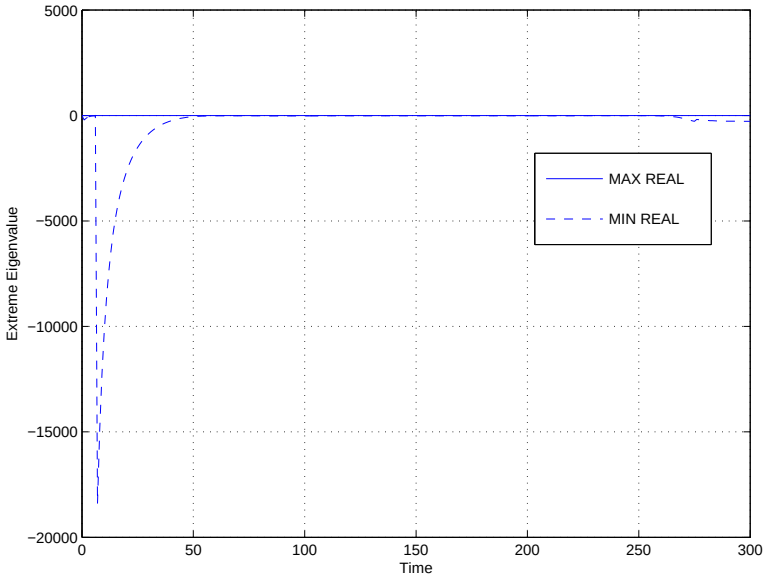

Fig. 2 Extreme eigenvalues of the model of Winslow et al.

As a concrete example, we look at the model of Winslow et al. The plot of the extreme eigenvalues over time in Figure 2 demonstrates that the degree of stiffness over time varies considerably over the solution interval. As a simple characterization, we consider the problem to be stiff in the interval $[0, 50]$ and non-stiff in $[50, 300]$. Using this information, we design a set of simple type-insensitive solvers as follows: Apply RL, GRL2, or BE on $[0, 50]$ and FE on $[50, 300]$. We constructed type-insensitive solvers based on a similar analysis of the extreme eigenvalues of the models of Jafri et al., Bondarenko et al., and Pandit et al. (2001); see [49] for the complete figures. We note that practical type-insensitive software for realistic simulations is likely to require an automatic method for choosing the appropriate solver; see, e.g., [47]. However, a full discussion of this issue is beyond the scope of this study.

We performed an experiment as described in the previous section to compare this type-insensitive method to the other 3 methods discussed in this study. For each model in this experiment we have two step sizes to consider, one for each method. Accordingly we found the pair of step sizes that minimized execution time while producing a solution with less than 5% RRMS error and present the results in Table 6. We find that the type-insensitive method that combines BE with FE is always the best performing, with improvements ranging from 10% to 10 times faster than the next most efficient single method; recall Tables 4–5 for the results of the single methods. Similar results hold at 1% RRMS error. The type-insensitive methods presented here are mainly for proof-of-concept; further use of these eigenvalue data or more elaborate type-insensitive strategies could lead to more efficient type-insensitive methods.

Table 6 Stiffness intervals and execution time, in seconds, of type-insensitive methods.

Model	Stiff Interval	Non-stiff Interval	RL-FE	GRL2-FE	BE-FE
			Time		
Bondarenko et al. (2004)	[0 10]	[10 75]	6.72E–1	5.90E–1	2.64E–1
Jafri et al. (1998)	[0 50]	[50 300]	8.47E–1	9.11E–1	5.38E–1
Pandit et al. (2001)	[105 125]	[0 105]; [125 250]	9.86E+0	1.42E+1	7.81E+0
Winslow et al. (1999)	[0 50]	[50 300]	1.07E+0	7.18E–1	2.30E–1

5 Conclusions

In this paper, we examined the eigenvalues of the Jacobian of a variety of cardiac electrophysiological models. In particular, we examined the consequences for stiffness that arise from these eigenvalues and demonstrated how numerical methods could be selected or designed to overcome stiffness.

The extremes of these eigenvalues were presented in Section 3.1. The wide range of cardiac electrophysiological models led to considerable differences in the eigenvalues from one model to another. One can infer from these data that there is a large variation in the stiffness properties among the models. At one extreme is the non-stiff FitzHugh–Nagumo model; at the other is the highly stiff model of Pandit et al. (2003). The data suggest that most cell models are moderately stiff for the typical accuracies required.

Numerical experiments with 3 integration methods (FE, RL, and GRL2) for cell models were discussed in Section 3.2. The non-standard methods GRL2 and RL were found to outperform FE on 33 of the 37 models considered. Individually, each of RL and GRL2 was the most efficient method on roughly an equal number of models. This observation lends support to the effectiveness of the recently proposed GRL2

method, albeit at the cost of a non-trivial implementation. With its good stability, RL seems to generally strike a good balance between efficient integration and ease of implementation. Nonetheless, these methods did not perform satisfactorily on all models. In particular, FE, RL, and GRL2 all face severe step-size restrictions due to stiffness for the models of Winslow et al. and Pandit et al. (2003).

In the cases where FE, RL, and GRL2 produce a solution inefficiently due to step-size restrictions, eigenvalue data can be used to give insight into the cause as well as potential solutions. In Section 4.1, we observed that the model of Pandit et al. (2003) is highly stiff for most of the solution interval. This suggests the use of a stiff solver, such as BE. BE is not commonly used by researchers in cell modelling; however the data and experiments presented here make a compelling case for its use on such models. We compared the solution with BE to the 3 other methods in this study and found that BE was able to produce a solution with at most 5% RRMS error almost 2300 times faster than the next most efficient method (FE). In Section 4.2, we observed that the model of Winslow et al. is stiff only in a portion of the overall solution interval. The same was true for the models of Bondarenko et al., Jafri et al., and Pandit et al. (2001). As a way to improve efficiency, we proposed the use of a type-insensitive method that uses different numerical methods in the stiff and non-stiff sub-intervals. Specifically we combined RL, GRL2, and BE as stiff solvers with FE as the non-stiff solver. The BE-FE type-insensitive solver was able to produce an acceptable solution from about 10% to 10 times times faster than the next most efficient standard method. We note that the goal of our studies was to demonstrate how the eigenvalue data can be used to improve the efficiency of the integration rather than to lay claim to the most efficient method for a given model.

References

1. http://bioeng.washington.edu/jsim/
2. Aliev, R.R., Panfilov, A.V.: A simple two-variable model of cardiac excitation. Chaos, Solitions and Fractals **7**(3), 293–301 (1996)
3. Artebrant, R., Sundnes, J., Skavhaug, O., Tveito, A.: A second order method of Rush-Larsen type. Technical report, Simula Research Laboratory (2008)
4. Ascher, U.M., Petzold, L.R.: Computer methods for ordinary differential equations and differential-algebraic equations. Society for Industrial and Applied Mathematics (SIAM), Philadelphia, PA (1998)
5. Atkinson, K., Han, W.: Elementary Numerical Analysis, third edn. John Wiley & Sons Ltd, Hoboken, NJ (2005)
6. Beeler, G.W., Reuter, H.: Reconstruction of the action potential of ventricular myocardial fibres. J Physiol **268**(1), 177–210 (1977)
7. Bischof, C.H., Bücker, H.M., Vehreschild, A.: A macro language for derivative definition in ADiMat. In: Automatic differentiation: applications, theory, and implementations, *Lect. Notes Comput. Sci. Eng.*, vol. 50, pp. 181–188. Springer, Berlin (2006)
8. Bondarenko, V.E., Szigeti, G.P., Bett, G.C.L., Kim, S.J., Rasmusson, R.L.: Computer model of action potential of mouse ventricular myocytes. Am J Physiol Heart Circ Physiol **287**(3), H1378–403 (2004)
9. Burkardt, J.: http://orion.math.iastate.edu/burkardt/math2071/be_newton.m
10. Butcher, J.C.: Order, stepsize and stiffness switching. Computing **44**(3), 209–220 (1990)
11. Courtemanche, M., Ramirez, R.J., Nattel, S.: Ionic mechanisms underlying human atrial action potential properties: insights from a mathematical model. Am. J. Physiol. **275**(1), H301–H321 (1998)
12. Demir, S.S., Clark, J.W., Giles, W.R.: Parasympathetic modulation of sinoatrial node pacemaker activity in rabbit heart: a unifying model. Am J Physiol **276**(6 Pt 2), H2221–44 (1999)
13. Demir, S.S., Clark, J.W., Murphey, C.R., Giles, W.R.: A mathematical model of a rabbit sinoatrial node cell. Am J Physiol **266**(3 Pt 1), C832–52 (1994)
14. DiFrancesco, D., Noble, D.: A model of cardiac electrical activity incorporating ionic pumps and concentration changes. Philos Trans R Soc Lond B Biol Sci **307**(1133), 353–98 (1985)

15. Dokos, S., Celler, B., Lovell, N.: Ion currents underlying sinoatrial node pacemaker activity: a new single cell mathematical model. J Theor Biol **181**(3), 245–72 (1996)

16. Faber, G.M., Rudy, Y.: Action potential and contractility changes in [Na(+)](i) overloaded cardiac myocytes: a simulation study. Biophys J **78**(5), 2392–404 (2000)

17. FitzHugh, R.: Impulses and Physiological States in Theoretical Models of Nerve Membrane. Biophys. J. **1**(6), 445–466 (1961)

18. Fox, J.J., McHarg, J.L., Gilmour Robert F., J.: Ionic mechanism of electrical alternans. Am J Physiol Heart Circ Physiol **282**(2), H516–530 (2002)

19. Hairer, E., Wanner, G.: Solving Ordinary Differential Equations II: Stiff and Differential-Algebraic Problems, second edn. Springer-Verlag, Berlin (1996)

20. Hilgemann, D.W., Noble, D.: Excitation-contraction coupling and extracellular calcium transients in rabbit atrium: reconstruction of basic cellular mechanisms. Proc R Soc Lond B Biol Sci **230**(1259), 163–205 (1987)

21. Hodgkin, A., Huxley, A.: A quantitative description of membrane current and its application to conduction and excitation in nerve. J. Physiol. (Lond) **117**(4), 500–544 (1952)

22. Hund, T.J., Rudy, Y.: Rate dependence and regulation of action potential and calcium transient in a canine cardiac ventricular cell model. Circulation **110**(20), 3168–74 (2004)

23. Jafri, M.S., Rice, J.J., Winslow, R.L.: Cardiac Ca2+ dynamics: the roles of ryanodine receptor adaptation and sarcoplasmic reticulum load. Biophys J **74**(3), 1149–68 (1998)

24. Kogan, B.Y., Karplus, W.J., Billett, B.S., Pang, A., Karagueuzian, H.S., Kahn, S.S.: The simplified FitzHugh–Nagumo model with action potential duration restitution: effects on 2-D wave propagation. Physica D **50**, 327–340 (1991)

25. Lambert, J.D.: Numerical methods for ordinary differential systems. John Wiley & Sons Ltd., Chichester (1991). The initial value problem

26. Lloyd, C.M., Lawson, J.R., Hunter, P.J., Nielsen, P.F.: The CellML model repository. Bioinformatics **24**(18), 2122–2123 (2008)

27. Luo, C., Rudy, Y.: A model of ventricular cardiac action potential. Circ. Res. **68**(6), 1501–1526 (1991)

28. Maclachlan, M.C., Sundnes, J., Spiteri, R.J.: A comparison of non-standard solvers for ODEs describing cellular reactions in the heart. Comput Methods Biomech Biomed Engin **10**(5), 317–26 (2007)

29. Maleckar, M.M., Greenstein, J.L., Trayanova, N.A., Giles, W.R.: Mathematical simulations of ligand-gated and cell-type specific effects on the action potential of human atrium. Prog Biophys Mol Biol **98**(2-3), 161–70 (2008)

30. MathWorks Inc., The: Matlab R2008b. `http://www.mathworks.com/` (2008)

31. McAllister, R.E., Noble, D., Tsien, R.W.: Reconstruction of the electrical activity of cardiac Purkinje fibres. J Physiol **251**(1), 1–59 (1975)

32. Nagumo, J., Arimoto, S., Yoshizawa, S.: An active pulse transmission line simulating nerve axon. Proceedings of the IRE **50**(10), 2061–2070 (1962)

33. Noble, D.: A modification of the Hodgkin–Huxley equations applicable to Purkinje fibre action and pace-maker potentials. J Physiol **160**(NIL), 317–52 (1962)

34. Noble, D., Noble, S.J.: A model of sino-atrial node electrical activity based on a modification of the DiFrancesco-Noble (1984) equations. Proc R Soc Lond B Biol Sci **222**(1228), 295–304 (1984)

35. Noble, D., Noble, S.J., Bett, G.C., Earm, Y.E., Ho, W.K., So, I.K.: The role of sodium-calcium exchange during the cardiac action potential. Ann N Y Acad Sci **639**(NIL), 334–53 (1991)

36. Noble, D., Varghese, A., Kohl, P., Noble, P.: Improved guinea-pig ventricular cell model incorporating a diadic space, IKr and IKs, and length- and tension-dependent processes. Can J Cardiol **14**(1), 123–34 (1998)

37. Nygren, A., Fiset, C., Firek, L., Clark, J.W., Lindblad, D.S., Clark, R.B., Giles, W.R.: Mathematical model of an adult human atrial cell: the role of K+ currents in repolarization. Circ Res **82**(1), 63–81 (1998)

38. Pandit, S.V., Clark, R.B., Giles, W.R., Demir, S.S.: A mathematical model of action potential heterogeneity in adult rat left ventricular myocytes. Biophys J **81**(6), 3029–3051 (2001)

39. Pandit, S.V., Giles, W.R., Demir, S.S.: A mathematical model of the electrophysiological alterations in rat ventricular myocytes in type-I diabetes. Biophys J **84**(2 Pt 1), 832–841 (2003)

40. Petzold, L.: Automatic selection of methods for solving stiff and nonstiff systems of ordinary differential equations. SIAM J. Sci. Statist. Comput. **4**(1), 137–148 (1983)

41. Puglisi, J.L., Bers, D.M.: Labheart: an interactive computer model of rabbit ventricular myocyte ion channels and ca transport. Am. J. Physiol. Cell Physiol. **281**(6), C2049–C2060 (2001)

42. Pullan, A.J., Buist, M.L., Cheng, L.K.: Mathematically Modelling the Electrical Activity of the Heart: From Cell to Body Surface and Back Again. World Scientific, New Jersey (2005)

43. Rogers, J., McCulloch, A.: A collocation–galerkin finite element model of cardiac action potential propagation. IEEE Trans. Biomed. Eng. **41**(8), 743–757 (1994)

44. Rush, S., Larsen, H.: A practical algorithm for solving dynamic membrane equations. IEEE Trans. Biomed. Eng. **BME-25**(4), 389–392 (1978)

45. Sakmann, B.F., Spindler, A.J., Bryant, S.M., Linz, K.W., Noble, D.: Distribution of a persistent sodium current across the ventricular wall in guinea pigs. Circ Res **87**(10), 910–4 (2000)

46. Shampine, L.F.: Type-insensitive ODE codes based on extrapolation methods. SIAM J. Sci. Statist. Comput. **4**(4), 635–644 (1983)

47. Shampine, L.F.: Stiffness and the automatic selection of ode codes. J. Comput. Phys. **54**, 74–86 (1984)

48. Spiteri, R.J., Dean, R.C.: On the performance of an implicit-explicit Runge–Kutta method in models of cardiac electrical activity. IEEE Trans. Biomed. Eng. **55**(5), 1488–1495 (2008)

49. Spiteri, R.J., Dean, R.C.: Stiffness analysis of cardiac electrophysiological models. Technical report, Department of Computer Science, University of Saskatchewan (2009)

50. Spiteri, R.J., MacLachlan, M.C.: An efficient non-standard finite difference scheme for an ionic model of cardiac action potentials. J. Difference Equ. Appl. **9**(12), 1069–1081 (2003)

51. Stewart, P., Aslanidi, O.V., Noble, D., Noble, P.J., Boyett, M.R., Zhang, H.: Mathematical models of the electrical action potential of Purkinje fibre cells. Philos Transact A Math Phys Eng Sci **367**(1896), 2225–55 (2009)

52. Sundnes, J., Artebrant, R., Skavhaug, O., Tveito, A.: A second-order algorithm for solving dynamic cell membrane equations. IEEE Trans. Biomed. Eng. **56**(10), 2546–2548 (2009)

53. Sundnes, J., Lines, G.T., Cai, X., Nielsen, B.F., Mardal, K.A., Tveito, A.: Computing the electrical activity in the heart. Springer-Verlag, Berlin (2006)

54. Ten Tusscher, K.H.W.J., Noble, D., Noble, P.J., Panfilov, A.V.: A model for human ventricular tissue. Am J Physiol Heart Circ Physiol **286**(4), H1573–89 (2004)

55. Ten Tusscher, K.H.W.J., Panfilov, A.V.: Cell model for efficient simulation of wave propagation in human ventricular tissue under normal and pathological conditions. Phys. Med. Biol. **51**(23), 6141–6156 (2006)

56. Van Capelle, F.J.L., Durrer, D.: Computer simulation of arrhythmias in a network of coupled excitable elements. Circ. Res. **47**, 454–466 (1980)

57. Vigmond, E.J., dos Santos, R.W., Prassl, A.J., Deo, M., Plank, G.: Solvers for the cardiac bidomain equations. Prog Biophys Mol Biol **96**(1-3), 3–18 (2008)

58. Wang, L.J., Sobie, E.A.: Mathematical model of the neonatal mouse ventricular action potential. Am J Physiol Heart Circ Physiol **294**(6), H2565–2575 (2008)

59. Winslow, R.L., Rice, J., Jafri, S., Marbán, E., O'Rourke, B.: Mechanisms of altered excitation-contraction coupling in canine tachycardia-induced heart failure, ii model studies. Circ Res. **84**(5), 571–586 (1999)

60. Ying, W., Rose, D.J., Henriquez, C.S.: Efficient fully implicit time integration methods for modeling cardiac dynamics. IEEE Trans. Biomed. Eng. **55**(12), 2701–2711 (2008)

61. Zhang, H., Holden, A.V., Kodama, I., Honjo, H., Lei, M., Varghese, T., Boyett, M.R.: Mathematical models of action potentials in the periphery and center of the rabbit sinoatrial node. Am J Physiol Heart Circ Physiol **279**(1), H397–421 (2000)