2011 34th IEEE Software Engineering Workshop

A Change of Perspective Yields Formal Analysis

R. Grosu[1], F. Fenton[3], S. Smolka[1] and E. Bartocci[1,2]

[1]Department of Computer Science, Stony Brook University, Stony Brook, NY
[2]Department of Applied Mathematics and Statistics, Stony Brook University, Stony Brook, NY
[3]Department of Biomedical Sciences, Cornell University, Ithaca, NY

***Abstract*—In this paper we argue that a judicious use of models in science and engineering can considerably simplify the design and analysis of complex dynamic systems. To substantiate this claim, we first review the mathematical form and the role played by models in science and engineering, respectively. We then show that a change in perspective on the purpose of models in the analysis of cardiac tissue, allowed us to derive for the first time, in an automatic fashion, the parameter-ranges distinguishing between normal and abnormal behavior in cardiac cells.**

I. Introduction

Models play a central role in science and engineering. They allow us to capture the laws of nature, predict the behavior of complex dynamic systems, and devise controllers enforcing a desired behavior of a complex dynamic system.

The particular mathematical formalism used to describe such models can also be very different. For example, continuous partial differential equations (PDE), switched PDEs, finite automata (FA), discrete or continuous hidden Markov models (HMM), and (possibly stochastic) hybrid automata (HA).

Models can be constructed at various levels of abstraction, too, from very detailed to very coarse. To be useful however, the abstraction hierarchy has to relate its models in a meaningful way, through abstraction and concretization relations.

It is our conviction that every role played by models is important, and so is the abstraction level and the mathematical formalism used. However, it is even more important to be very aware of the purpose of a mathematical model. Disregarding this purpose may lead to unnecessarily complex designs whereas a conscious use of a model can considerably simplify both the design and the analysis of complex systems.

In our work we have used various models for various purposes: FAs and Büchi automata to check if a nondeterministic FA satisfies a given linear temporal logic (LTL) property [1], [2]; HMMs to predict the abnormal-behavior probability of a software system, from its time-sampled executions [3], [4]; linear time-invariant differential equations to design controllers balancing energy consumption with respect to performance [5]; and HAs to predict abnormal-behavior parameter ranges in cardiac myocytes [6]. In some of these cases we learned (identified) the models from experimental data, by assuming that the models have a particular mathematical form. In other cases, we used systematic approximations, to obtain a mathematical form amenable to formal analysis.

In this paper, we report in on our experience with models in the biological setting of cardiac myocytes. In particular, on how a careful consideration (a change in perspective) of the intended use of the models, allowed us to perform the first automated parameter-range identification for myocytes distinguishing between normal and abnormal behavior.

The rest of the paper is organized as follows. In Section II we review the various roles played by models in life sciences, engineering and computer science, and also their particular mathematical formalism. In Section III we use this review to discuss our experience with models in the biological setting.

II. The Various Uses of Models

A. Models in Natural Sciences

a) Modeling: Sciences, and in particular physics, have a long tradition in building *mathematical models* for physical systems. These models serve two main purposes:

1) Capturing the basic laws of the physical systems.
2) Predicting the behavior of the physical systems.

For example, Newton's mathematical model of *classical mechanics*, is governed by three basic laws:

1) An object at rest tends to stay at rest, and an object in motion tends to stay in motion with fixed speed, unless a nonzero resultant force acts upon it.
2) The relationship between a body mass m, its acceleration a, and an applied force is F, is $F = ma$.
3) The interaction forces between two bodies are of equal magnitude and in opposite directions.

This mechanical model allows to predict the motion of bodies in presence of gravity or any other forces.

Newton's model has been refined by Einstein's *general relativity theory*, to account for bodies moving with speeds close to the speed of light. An important part of this refinement, is the addition of a new physical law, stating that:

4) Space-time fabric is curved around bodies with mass.

This law allows to explain gravity, a question not answered by Newton's model. It is important to note however, that the predictions made by using Newton's model are identical with the predictions made with Einstein's model, as long as the bodies move with speeds much smaller than the speed of light.

The mathematical language used in the sciences to construct models such as the ones above, is the modern differential

1550-6215 2011

U.S. Government Work Not Protected by U.S. Copyright
DOI 10.1109/SEW.2011.16

calculus, usually credited to Newton and to Leibniz. These models describe the output y and the infinitesimal change in the state x of the physical system at a given time t [7]:

$$\dot{x}(t) = f(t, x(t), u(t)), \quad x(0) = x_0, \quad y(t) = g(t, x(t))$$

where u is the input and f, g are possibly nonlinear functions.

If the *next state function* f and the *output function* g are *linear*, then the physical system can be written as follows:

$$\dot{x}(t) = A(t)x(t) + B(t)u(t), \quad x(0) = x_0, \quad y(t) = C(t)x(t)$$

where A, B and C are the *next state*, the *input* and the *output matrices*, respectively. If these matrices are independent of time t, then the system is called *time invariant*.

As a consequence, ordinary differential equations, or partial differential equations if space is considered, provide a *local*, recursive view of the behavior in time of the physical system. This view is also known as the *state space* view of the system.

b) Prediction: In order to predict the behavior of the system, one needs however, a *global*, non-recursive, *input-output* view of the system, also known as the *closed-form* solution (or fixpoint) of the differential equations:

$$x(t) = \hat{f}(t, u(t)), \quad x(0) = x_0, \quad y(t) = \hat{g}(t, x(t))$$

For autonomous, time-invariant linear systems, that is for systems with no input, the solution has the following form:

$$x(t) = e^{At}x_0, \qquad y(t) = Cx(t)$$

Assuming that $u(t) = e^{Ut}u_0$, that is, that $u(t)$ is also a linear system, then the general solution for x has the form:

$$x(t) = e^{At}x_0 + \int_0^t e^{A(t-\tau)} B e^{U\tau} u_0 \, d\tau$$

In this case the closed-form solution $x(t)$ can be computed, as one knows how to compute the above integral.

Unfortunately, most physical systems do not have a linear time-invariant behavior, and for such systems it is generally not known how to compute their solution x, and correspondingly, their output y. In such cases, the (applied) sciences use two basic techniques: *simulation* and *approximation* [7].

Simulation takes advantage of the *computational meaning* of partial differential equations: A particular *while-loop program*, which computes the state of the system at time $t + dt$ (and space $s + ds$ if space is of concern, too), by knowing the state at time t (and space s). Initially $x(0) = x_0$ and $y(0) = g(0, x_0)$. The body of the loop then computes x and y at next dt:

$$x(t + dt) = x(t) + f(t, x(t), u(t))\, dt, \quad y(t) = g(t, x(t))$$

Choosing time (and space) increments dt (and ds) such that continuous behavior is recoverable, one can simulate (test) global behaviors of the system, one at a time. This interpretation of differential equations was indeed one of the motivating factors in the early development of computer science.

One simulation is one very accurate sample of the physical system's behavior, which is bounded both in time and space. By applying Monte-Carlo techniques, one can extend the simulation approach, to predictions about the global system behavior, with a given confidence and error margin. This extension is however plagued by rare events, whose accurate handling may require an exponential number of samples.

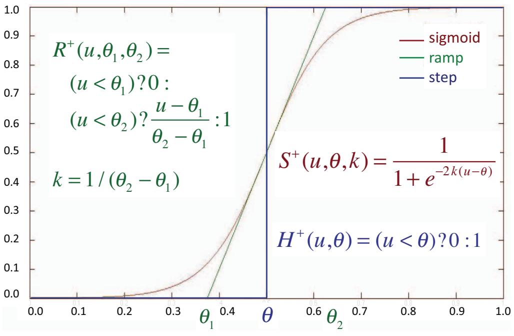


Fig. 1. Threshold-based switching functions.

With simulation alone, it is therefore impossible in general, to state with certainty that an undesired behavior will never happen, or that a desired behavior will eventually happen.

To make such statements, (applied) scientists construct asymptotic or perturbation-theory approximations, of the physical systems. These approximations result in piecewise linear differential equations, equivalent up to an approximation error ϵ to the original systems. For example, to make sure that a hydrogen bomb will not exhaust earth's atmosphere, physicists constructed such approximations, before detonation.

c) Duality and switching: One of the most intriguing aspects of physics is the particle-wave duality. At the root of this duality is the discovery that certain physical phenomena, for example the spectrum of light, do not vary in a continuous way, but have discrete jumps instead, also called quanta.

This discovery has important consequences: Physical phenomena cannot be described solely with continuous differential equations over the field structure of complex numbers. As discussed later, computer science is pushing this duality even further, distinguishing between numeric and non-numeric values. In philosophy, this duality is known as the following principle: Quantitative changes lead to qualitative changes.

Mathematically, the discontinuities in the spectrum of light can be captured with the Heaviside (or step) switching functions, shown in Figure 1. They can be understood as the limit of continuous, sigmoidal switching functions, also shown in Figure 1. Such switches occur everywhere in physical processes, for example in the description of transistors, and they are ubiquitous control functions in biological systems. Shallow sigmoidal switches can be approximated with (possibly a sequence of) ramps, as shown in Figure 1.

In conclusion, duality requires that physical and biological systems are in general described with a set of *switched* nonlinear differential equations. Switching is determined by Boolean threshold functions of the form $u(t) < \theta$ or $u(t) \geq \theta$.

B. Models in Engineering

Engineering and in particular control theory have added a new purpose to the development of models. They serve to:

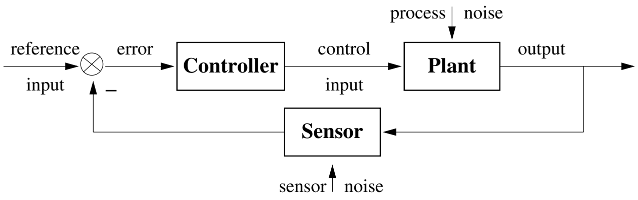


Fig. 2. Plant with feedback controller.

3) Design a controller enforcing a desired system behavior.

Given a physical system (*plant*) and a *reference* input specifying the desired output of the plant, the goal of control theory is to develop tools and techniques for the semi-automatic synthesis of a *controller*, which manipulates the plant's controlled inputs to ensure that the plant's output tracks the reference input with a negligible dynamic error. The block diagram of a feedback control system is shown in Figure 2.

The semi-automatic synthesis of a controller requires the existence of a model of the plant. Unless the physical laws governing the plant are known, this model is *learned* from an approximation of the input-output behavior of the plant. This approximation is obtained by stimulating the plant with well-chosen input signals and recording the output signals. The choice of the input signals is very important: at every moment of time, they should expose the dependency of the output not only on the input but also on the internal state of the plant.

For given input-output behavior and state-vector dimension, the choice of the state vector is not unique. Hence, in contrast to the sciences, in control theory, the model learned has a particular canonical form, which is most appropriate for either learning, observation, or control.

The plant behavior is in general nonlinear. While tools and techniques for learning nonlinear plant models exist [8], [9], the automatic design of associated controllers is much more intricate [10], [11]. Hence, in the majority of cases, one is content to learn simplified *linear* models which accurately reproduce the plant behavior around one of its equilibrium points. The learning process, where the plant model is assumed to be linear, is also known as *linear system identification*.

The continual correction performed by the controller keeps the error low even while abstracting a considerable number of states and inputs of the linear model. This abstraction can lead to a dramatic reduction in the model dimension. Hence, unlike in model-checking or static analysis [12], in control one can obtain good results with quite inaccurate models.

Common practice is to compensate for this inaccuracy by adding an uncontrolled *disturbance* input, also called *process noise*, to the plant. The sensors themselves may have limited accuracy, which is commonly captured by adding a disturbance input to the sensor, also called *sensor noise*. Disturbances are assumed to be nondeterministic or stochastic in nature.

d) Continuous numeric domains.: Control theory was developed for physical systems whose input, output, and state signals are numeric and continuous [13]. Elements of numeric domains can be added and multiplied, and both operations possess a neutral element: 0 for addition and 1 for multiplication. They also possess inverses: subtraction for addition and division for multiplication. Numeric domains are therefore field structures, with the real numbers and the complex numbers as the most important representatives [14].

In this setting, the time-invariant linear model of a plant can be expressed as discussed in the previous section, where in addition, $v(t)$ and $w(t)$ are white noise processes, and matrices K and L capture their relative contribution:

$$\dot{x}(t) = Ax(t) + Bu(t) + Kv(t), \quad y(t) = Cx(t) + Lw(t)$$

An important concern in controller design is the observability and the controllability of the system, for the considered output and control input, respectively. If the system is not observable, then it contains states that do not contribute to the output and if the system is not controllable, then it contains states that cannot be reached with the given input. In these cases, the system can be reduced to an observable and controllable form, which is unique, modulo a linear transformation.

e) Discrete numeric domains.: Before the advent of computerized tool support, controllers were designed manually [13]. This was a tedious activity which was considerably simplified by assuming that plants had a single input and a single output (SISO). This assumption was adequate only if the model of the plant could reasonably be decomposed into an array of SISO components with very weak interdependencies.

The powerful techniques and tool support nowadays available, for example, as Matlab identification and controller design tool-boxes, removed this limitation [13], [15], [16]. Linear, multiple input and multiple output (MIMO) models are now routinely learned (identified). While controller synthesis for linear MIMO models does not yet have the same automation as for SISO models, individual tools and techniques are available, and associated tools can be implemented. For example, we have developed Matlab scripts supporting such automation. MIMO controller design is more general than SISO design, and it allows one to quantify the tradeoff between the various desired dynamic characteristics[17], [18].

The proliferation of computer systems spurred the transition from *analog* to *digital* controllers. In this setting, the continuous outputs of the plant are sampled every T time units by an analog-to-digital converter, and the digital outputs of the controller are held for every T time units by a digital-to-analog converter. This approach lead to the development of *digital control theory*, where the linear model of a plant:

$$x(n{+}1) = Ax(n) + Bu(n) + Kv(n), \; y(n) = Cx(n) + Lw(n)$$

is expressed analogously to the continuous case, using *difference equations* in place of differential equations, and where the natural number n represents the time $t = nT$.

Techniques for the identification of linear, time-invariant differential and difference equations are well understood [16], as is the associated controller-synthesis problem [15]. Moreover, Matlab tool support for linear, time-invariant system-identification and controller synthesis is available [16], [15].

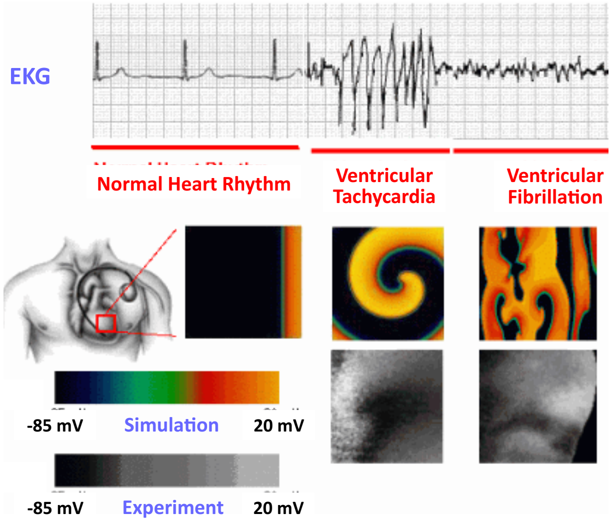


Fig. 3. Emergent behavior in cardiac-cell networks. Top: Electrocardiogram. Middle and bottom: Simulation and experimental mappings of voltage waves occurring in a small rectangular area on the surface of the heart.

C. Models in Computer Science

Since switched nonlinear differential equations can be represented as while-loop programs, every model developed in the natural sciences or in engineering, is also a computer science model. In addition, computer science contains models that do not necessarily represent physical systems.

In particular, many computer systems have inputs, states, and outputs of a non-numeric nature. They generalize the Boolean value of the switching functions. Addition, subtraction, multiplication, and division make no sense for non-numeric values. Instead, using regular-expression operations (literal constants, union, and concatenation), non-numeric values give rise to *regular languages*. The alphabet Σ of these languages is the set of non-numeric values themselves. The set of all such languages forms a semi-ring structure.

While fields are also semi-rings, a commonality that can be exploited to import results from the numeric domain to the non-numeric one, there are key differences. Union is idempotent, whereas addition is generally not. Multiplication is commutative, whereas concatenation is not. Numeric values possess a distance between them, whereas languages do not. Languages, by contrast, possess a canonical partial order. Such a canonical order cannot exist in fields [19], [20].

Similarly to the life sciences and engineering models, the analysis and control of non-numeric models is decidable for a subclass, namely of linear systems over semirings.

f) Non-numeric domains and nondeterminism.: In this setting, the role of numeric linear systems over fields (without Gaussian noise) is assumed by *finite automata*. Although not generally known, finite automata are linear systems, albeit over the semi-ring structure of languages [21]. This is an important analogy as it allows one to import concepts from the better-understood theory of numeric control to the control of systems with non-numeric parameters [21], [20].

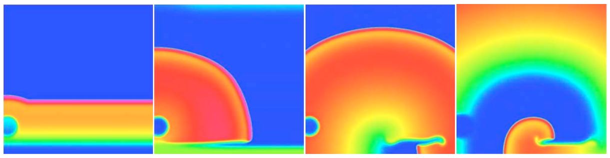


Fig. 4. Tachycardic wave induced by a disc of unexcitable myocytes.

This importing, has to be done with care; e.g. the right-hand side $Ax + Bu$ of the state equation (without noise) does not make sense in case of finite automata, since plus over regular languages means “choice” rather than addition. As a consequence, the input term Bu is dropped, and A is converted from a matrix of constants to a matrix of input symbols. Matrix A corresponds to the transition relation of the finite automaton.

Further, A can be decomposed into a set of Boolean transition relations $A(a)$, one for each non-numeric symbol a in the finite alphabet Σ, such that $A(a)(i,j) = 1$ only if there is an a-transition from state i to state j [22], [21]. With these transformations, states $x_i(n)$ become Booleans, true if the input sequence so far is accepted when starting in x_i, and false otherwise. The next state and output equations become:

$$x(n{+}1) = A(u(n))\, x(n), \qquad y(n) = Cx(n)$$

One can rewrite this time-triggered difference-equations system to an event-triggered difference-equations system, by using a branching notion of time, whose paths consist of sequences s of symbols (events) a in the input alphabet Σ:

$$x(as) = A(a)\, x(s), \quad x(\epsilon) = x_0, \quad y(s) = Cx(s)$$

Techniques for learning (identifying) finite automata from their input-output behavior are based on, and extend, Angluin's work [23], [24]. The larger the number of states of the system, the harder it is to accurately identify the system. The control problem for finite automata is based on the work of Ramadge and Wonham, and it is collectively known as supervisory control [25], [26]. The reference input in this case becomes a language, and the error notion is dealt with in an order-theoretic fashion (minimal, maximal, etc).

g) Non-numeric domains and stochasticity.: The noise term Kv of the right-hand side of the state equation is treated analogously to Bu: Kv is dropped, and noise is added to the matrix A by defining for each non-numeric input symbol i, a stochastic matrix $A(i)$. The RHS $Cx + Lw$ of the output equation is treated similarly. For each non-numeric output symbol o, the output matrix $C(o)$ is a stochastic matrix.

$$x(is) = A(i)\, x(s), \qquad y(s)(o) = C(o)x(s)$$

One can readily see that the resulting stochastic system is a Hidden Markov Model (HMM) [27], [28].

HMMs have primarily been used for monitoring and recognition. As such, they do not have input (or, equivalently, the input alphabet contains only one symbol), and they therefore possess only one next-state matrix. One of the classic questions in this setting is: What is the probability that a particular output sequence is generated by a given HMM?

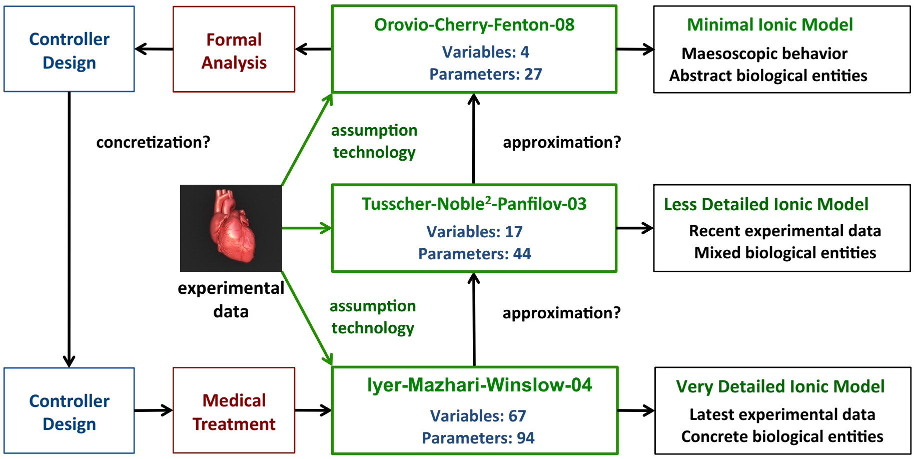


Fig. 5. Towers of abstraction for the analysis of cardiac abnormalities.

There is no reason, however, for not using full-fledged HMMs, with inputs and outputs, to address the controller-synthesis problem. Such a treatment is discussed in [28], which also addresses the problem of learning HMMs (see also [27]). The reference input and the input error in this setting have to be treated in a probabilistic fashion. Finally, the combination of nondeterminism and probability is addressed under the name of *hidden Markov processes* [29].

h) Mixed numeric and non-numeric domains.: Similarly to switched nonlinear differential equations, many computer science models consist of a mixture of non-numeric and numeric domains. For example, for system software energy and performance are numeric parameters, whereas the choice of a compression algorithm and compression level are non-numeric. The challenge in such models is to properly integrate numeric field structures with non-numeric semi-ring structures, for example in the design of modern controllers.

Such integration has begun in the form of *hybrid automata*, where non-numeric values, called *modes*, define a non-numeric linear system (a finite automaton), and where each mode has an associated invariant and numeric linear system with Gaussian noise. Within a mode, numeric states evolve according to the mode's differential or difference equations. A jump may occur to another mode, when the associated transition becomes enabled. A transition is forced to occur if remaining in the mode would cause the values of the numeric variables to violate the invariant associated with the mode.

The learning (or identification) problem for hybrid automata is discussed, for example, in the proceedings of the *Hybrid Systems: Computation and Control* conference, and their analysis and control problems are discussed, for example, in [30]. An approach to the control of computer systems that also leads to hybrid automata, although without explicit mention of this fact, the so-called gain scheduled control, is discussed in [31].

III. Models for Cardiac Myocytes

i) Myocyte models: The recent emergence of high-throughput experimental-data acquisition methods has dramatically changed biology, from a purely lab-based science to also an engineering and information science. The identification of a mathematical model, from data experimentally collected from a biological system, and the use of this model to predict and control the system's behavior, are now days indispensable tools in biology's arsenal [32], [33].

In the context of a mathematical model, the distinction between normal (or desired) system behavior and abnormal (or undesired) system behavior reduces to the identification of the parameter ranges for which the model accurately reproduces the normal and abnormal system behavior, respectively. Moreover, finding a treatment strategy, reduces to synthesizing a controller, which enforces desired behavior.

A fundamental question in the treatment of cardiac disorders, such as tachycardia and fibrillation [33] (see Figure 3), is under what circumstances does such a disorder arise? Cardiac contraction is electrically regulated by particular cells, known as myocytes. For each electric stimulus originating in the sino-atrial node of the heart (its natural pace-making unit), the myocytes propagate this stimulus and enforce this way the contraction of cardiac muscle, known as a heart beat.

In certain circumstances, myocytes can partially or completely loose excitability, that is their ability to propagate and reinforce an electric stimulus. A region of such myocytes can be responsible for ventricular tachycardia or fibrillation, as shown in Figure 4: the region becomes an obstacle in the way of a propagating electric wave, triggering a spiral rotation of the wave (tachycardia); the spiral may then break up into a disordered collection of spirals (fibrillation). Once such an abnormal behavior is triggered, it may last for a long time, due to the excitability of healthy myocytes.

The past decades have witnessed the development of increasingly sophisticated partial differential equations models (DEM) for myocytes, reflecting the technological advances. These DEMs are similar in spirit to the mechanical models used in physics. Their main purpose is to elucidate the biological laws governing the electric behavior of cardiac myocytes that is their underlying cellular and ionic processes.

The first myocyte DEM was the Luo-Rudi (LR) guinea-

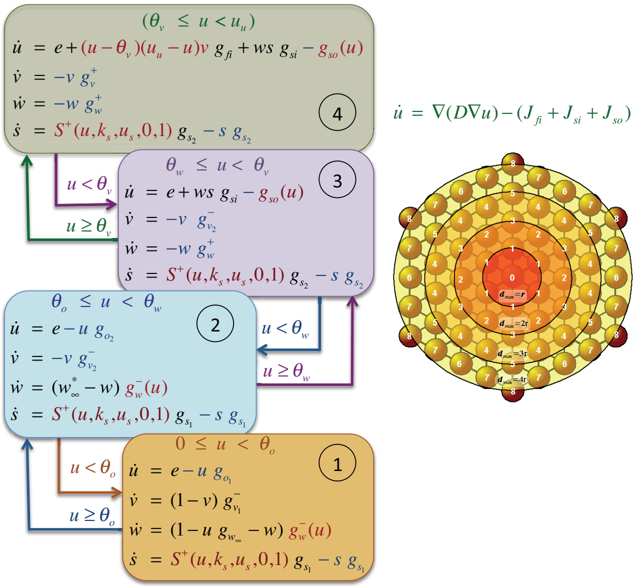

$(\theta_v \leq u < u_u)$
$\dot{u} = e + (u-\theta_v)(u_u - u)v\, g_{fi} + ws\, g_{si} - g_{so}(u)$
$\dot{v} = -v\, g_v^+$
$\dot{w} = -w\, g_w^+$
$\dot{s} = S^+(u,k_s,u_s,0,1)\, g_{s_2} - s\, g_{s_2}$
4
$u < \theta_v$
$u \geq \theta_v$
$\theta_w \leq u < \theta_v$
3
$\dot{u} = e + ws\, g_{si} - g_{so}(u)$
$\dot{v} = -v\, g_{v_2}^-$
$\dot{w} = -w\, g_w^+$
$\dot{s} = S^+(u,k_s,u_s,0,1)\, g_{s_2} - s\, g_{s_2}$
$\theta_o \leq u < \theta_w$
2
$\dot{u} = e - u\, g_{o_2}$
$\dot{v} = -v\, g_{v_2}^-$
$\dot{w} = (w_\infty^* - w)\, g_w^-(u)$
$\dot{s} = S^+(u,k_s,u_s,0,1)\, g_{s_1} - s\, g_{s_1}$
$u < \theta_w$
$u \geq \theta_w$
$0 \leq u < \theta_o$
1
$u < \theta_o$
$u \geq \theta_o$
$\dot{u} = e - u\, g_{o_1}$
$\dot{v} = (1-v)\, g_{v_1}^-$
$\dot{w} = (1 - u\, g_{w_\infty} - w)\, g_w^-(u)$
$\dot{s} = S^+(u,k_s,u_s,0,1)\, g_{s_1} - s\, g_{s_1}$
$\dot{u} = \nabla(D\nabla u) - (J_{fi} + J_{si} + J_{so})$

Fig. 6. Canonical hybrid automaton model for human ventricular myocytes.

pig ventricular model in [34]. Refining this model to human myocytes led to the Tusscher-Noble^2-Panfilov (TNNP) DEM in [35]. This DEM has 17 state variables and 44 parameters. Similarly to LR DEM, TNNP DEM is an electric model, with the myocyte's membrane modeled as a capacitor, and the ionic channels modeled as resistances with nonlinear behavior.

The Iyer-Mazhri-Winslow (IMW) DEM in [36] is one of the most detailed cellular and ionic models to date. It contains 65 state variables and 94 parameters. This DEM is based on most recent experimental data, and it reflects the latest understanding of the biological entities regulating the electric behavior of human myocytes. In contrast to the TNNP DEM, the equations of the IMW DEM are written in a chemical reaction (law of mass action) style. As a consequence, most of the differential equations have a simpler form then in TNNP. Their dynamics is mostly affine, multiaffine or polynomial. In multiaffine dynamics, products of variables are allowed, as long as these products are linear in each variable. In polynomial dynamics, arbitrary products are allowed.

The problem with the LR, TNNP and IMW DEMs, and many other DEMs developed in the same spirit, is that they often contain too many parameters to be reliably and robustly identified from experimental data. Moreover, their dimensionality (for example 65 in IMW) is too large to render their formal analysis or even simulation tractable. If one is only interested in the transmembrane potential and its propagation in cardiac tissue, such models are unnecessarily complicated.

j) The minimal model: Motivated by these observations, Orovio-Cherry-Fenton developed a very versatile electric DEM in [37], involving only four state variables and 27 parameters. This DEM reproduces with great accuracy the mesoescopic AP behavior of myocyte tissue. The model was called by its authors the minimal model (MM). The dynamics of this model is nonlinear and it contains both step and sigmoidal switches. Moreover, its sigmoidal switches occur both as numerators and as denominators.

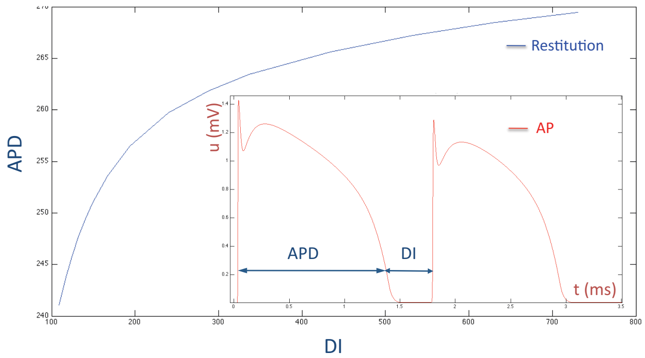


Fig. 7. Action potential (AP), AP duration (APD), diastolic interval (DI), and restitution curve at 10% of the maximum value of the AP.

In [6] we showed that one can bring the MM DEM to a canonical sigmoidal form, where the sigmoids occur only as numerators. Moreover, by expanding the step switches, one obtains the nonlinear hybrid automaton (MM HA) shown in Figure 6, where each mode corresponds to a particular phase of the myocyte's response to an external stimulus e.

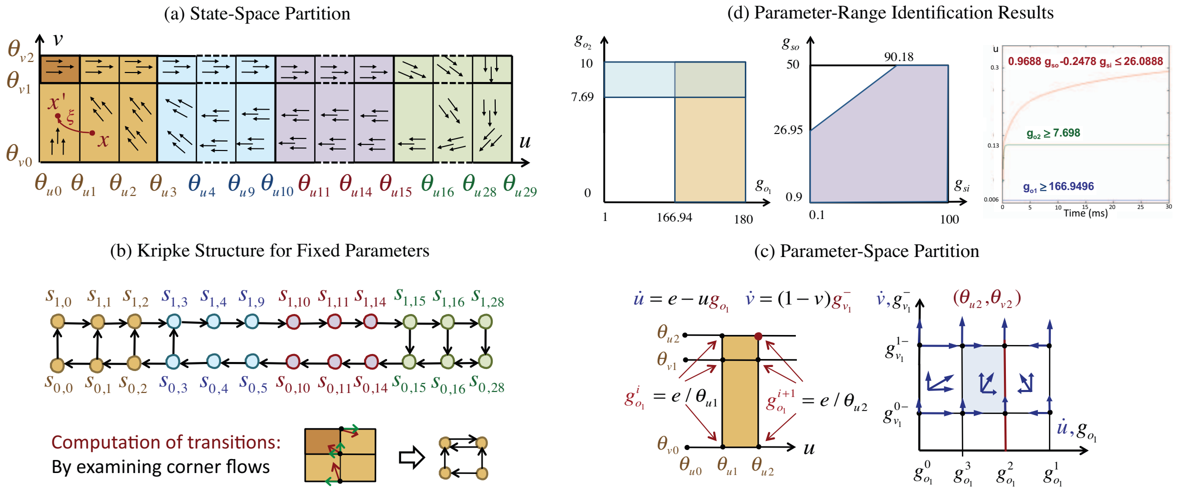
This HA contains four state variables: the myocyte's transmembrane potential u, and the myocyte's gating variables v, w and s which regulate the ionic flows through the myocyte's membrane channels. Intuitively, a healthy membrane acts like a capacitor. In the absence of an external stimulus e, it stays charged to -80mV, shown as 0-potential in the scaled Figure 7. A stimulus causes the ion-inflow channels to open which leads to a depolarization of the membrane, that is, to a raise of u in Figure 7. Subsequently these channels close, and the ion-outflow channels open, which leads to a repolarization of the membrane, that is, to a drop of u in Figure 7. The raise and drop of u is known as an action potential (AP). The parameters g_k of the MM HA denote conductances.

If the stimulus is weak, u never raises above threshold θ_v, and one talks about a failed AP-initiation. Otherwise, the AP raises to an approximately constant value, and the AP has the characteristic shape shown in Figure 7.

Recharging takes time. Hence, the more time the membrane has to recharge, the greater (and longer) the AP. The AP duration (APD), the diastolic interval (DI), and the AP restitution curve (dependence of the APD on the DI) measured at 10% of the maximum AP value, are shown in Figure 7. They are important properties of myocyte behavior.

k) Spatial models: A rectangular tissue is modeled as an $N \times N$ array of MMs. They communicate with each other through diffusion, expressed as $\nabla(D\nabla u)$ where ∇ is the gradient operator and D is the diffusion coefficient. In isotropic tissue D is constant, and diffusion reduces to $D \triangle u$. In the experiments of this paper, we considered isotropic tissue, only.

Adding $D \triangle u$ to the stimulus, transforms the ordinary differential equations of the MM HA into partial differential equations. Our implementation of diffusion is based on finite differences: To compute $D \triangle u$, an MHA polls the potential


(a) State-Space Partition

(b) Kripke Structure for Fixed Parameters

(c) Parameter-Space Partition

(d) Parameter-Range Identification Results

θ_{v2}
θ_{v1}
θ_{v0}
v
u
x'
ξ
x
θ_{u0} θ_{u1} θ_{u2} θ_{u3} θ_{u4} θ_{u9} θ_{u10} θ_{u11} θ_{u14} θ_{u15} θ_{u16} θ_{u28} θ_{u29}
$s_{1,0}$ $s_{1,1}$ $s_{1,2}$ $s_{1,3}$ $s_{1,4}$ $s_{1,9}$ $s_{1,10}$ $s_{1,11}$ $s_{1,14}$ $s_{1,15}$ $s_{1,16}$ $s_{1,28}$
$s_{0,0}$ $s_{0,1}$ $s_{0,2}$ $s_{0,3}$ $s_{0,4}$ $s_{0,5}$ $s_{0,10}$ $s_{0,11}$ $s_{0,14}$ $s_{0,15}$ $s_{0,16}$ $s_{0,28}$
Computation of transitions:
By examining corner flows
$\dot{u} = e - ug_{o_1}$
$\dot{v} = (1-v)g^-_{v_1}$
θ_{u2}
θ_{v1}
$g^i_{o_1} = e/\theta_{u1}$
$g^{i+1}_{o_1} = e/\theta_{u2}$
θ_{v0}
θ_{u0} θ_{u1} θ_{u2}
$\dot{v}, g^-_{v_1}$
$(\theta_{u2}, \theta_{v2})$
$g^{1-}_{v_1}$
$g^{0-}_{v_1}$
$\dot{u}, g_{o_1}$
$g^0_{o_1}$ $g^3_{o_1}$ $g^2_{o_1}$ $g^1_{o_1}$
g_{o_2}
10
7.69
0
1
166.94
180
g_{o_1}
g_{so}
50
90.18
26.95
0.9
0.1
100
g_{si}
u
0.3
0.13
0.006
0.9688 g_{so}-0.2478 g_{si} ≤ 26.0888
g_{o2} ≥ 7.698
g_{o1} ≥ 166.9496
0 5 10 15 20 25 30
Time (ms)

Fig. 8. Steps in computing the parameter ranges

u_n of its neighbours in each direction and takes the difference to its own potential u. Hence, the overall e is computed by summing up this difference with any external stimulus.

An AP-wave, is the potential u across the MM HA array. Now consider an 800×800 array with a disc of unexcitable HAs on its middle left area. Applying a train of stimuli to the bottom cells of the area, triggers successive waves, that hit the unexcitable cells. Due to the restitution property (recharging delay) of HAs, the unexcitable region eventually acts as a pace-making unit, and triggers the rotation of a spiral wave, as shown in Figure 4. Such a spiral may last in principle forever, or it may break down chaotically into sustained smaller spirals.

l) Parameter ranges for abnormal behavior: In terms of the MM HA model, lack of excitability means that mode four of the MM HA is never reached. This can be formulated in linear temporal logic (LTL) by requiring that globally (always) $u < \theta_v$. In LTL notation $G\,(u < \theta_v)$.

Unfortunately, the MM HA cannot be formally analysed as it contains nonlinear terms. We therefore linearized these terms, in an optimal way in each mode, by using dynamic programming techniques [6]. This resulted in 29 modes and 30 thresholds θ_{ui} for the voltage. The initial value of v and w is one, and in order to work with regions, we added the thresholds $\theta_{v0} = \theta_{w0} = 0$, $\theta_{v1} = \theta_{w1} = 0.95$ and $\theta_{v2} = \theta_{w2} = 1$, for v and w, respectively. Since s is initially zero we also added the thresholds $\theta_{s0} = 0$, $\theta_{s1} = 0.001$ and $\theta_{s2} = 1$ for s.

All these transformations brought the MM HA to the form of multi-affine HA (MHA), whose differential equations have the form of a genetic regulatory network:

$$\dot{x}_i = f_i(x,p) = \sum_{j\in P_i} \kappa_{ij}\, r_{ij}(x) - \sum_{j\in D_i} \gamma_{ij}\, r_{ij}(x)\, x_i \qquad (1)$$

where x_i is the i-th component of the state vector, κ_{ij} and γ_{ij} are production and degradation rate parameters, and r_{ij} are continuous piecewise multiaffine functions (PMA) arising from products of ramp functions r^+ and r^- (see Figure 9). Assuming that x_i does not regulate its own degradation, that is, for $j \in D_i$, the protein x_i does not occur in $r_{ij}(x)$, function $f = (f_1, \ldots, f_n)$ is multiaffine in x and affine in p.

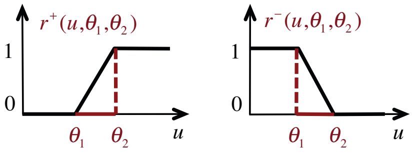


Fig. 9. Ramps in the multi-affine hybrid automaton (MHA)

The thresholds determine a partitioning of the MHA state space as shown for the $u\,v$ plane in Figure 8(a). Moreover, assuming that the atomic propositions of the LTL formula have the form $u < \theta$ or $u \geq \theta$, each hyperrectangle in this partition is an equivalence class with respect to the LTL formula. As a consequence, the quotient of the MHA with respect to the state-space-partition equivalence relation, results in a finite Kripke structure (KS) as shown in Figure 8(b).

The states of this KS are fully determined. For a fixed parameter value v_p, the function $f(x, v_p)$ is multiaffine, and the transitions of the KS are therefore fully determined by examining the value of $f(c_r, v_p)$ in the corners c_r of each hyperrectangle, as shown in Figure 8(b).

Unfortunately, there are an infinite number of parameter values v_p. Fortunately, the function $f(c_r, p)$ is affine in each corner c_r. Solving these affine equations in all corners c_r of the state space, results in a partitioning of the parameter space as shown in Figure 8(c). As $f(x, p)$ is affine in p, it is enough to construct a KS for each corner c_p in the parameter space, and check if the KS satisfies the property, to determine if all parameters inside a parameter space polyhedron satisfy the property. This algorithm can be further improved, but this improvement is omitted in this paper.

Using the approach described above, we obtained the first automatic inference of the parameter ranges for which the myocyte is not excitable. The results are shown in Figure 8(d), where we also show a simulation with a parameter value chosen from each of the computed ranges. The parameters of interest were the conductances g_{o_1}, g_{o_2}, g_{si} and g_{so}. We allowed these "uncertain" parameters to vary within the ranges $[0, 180]$, $[0, 10]$, $[0.1, 100]$ and $[0.9, 50]$, respectively, which were determined in an experimental way.

The MHA parameters are not directly related to biological entities, and as a consequence, they cannot be directly used in bio-chemical treatment strategies. Moreover, there is currently no abstraction or concretization relation among the DEMs for myocytes. To remedy this situation, we have started a systematic investigation of the abstractions necessary in order to reduce the IMW DEM to the MM HA. This will allow us to relate the parameters in these two models, and therefore develop an adequate bio-chemical treatment strategy.

It important to note that, as the voltage is a state variable of the MM HA, electric control strategies can be developed based on this HA. Of great importance in this respect is the real time simulation speed of the MM HA, which we developed on GPU architectures [38]. This was used to develop and implement the first low energy control strategy for both atria and ventricles, and shown to work both in vitro and in vivo [39].

Finally, also note that all the discussed DEMs were identified from experimental data. What distinguishes them is the amount of differentiation provided by the data, and the assumptions made about the form of the DEMs. The assumptions for the very detailed DEMs reflect the current knowledge about the underlying cellular and the ionic processes. The assumptions for the MM, reflect the desire for abstraction.

References

[1] R. Grosu and S. Smolka, "Quantitative model checking," in *Proc. of ISoLA'04, the 1st International Symposium on Leveraging Applications of Formal Methods*, Pahphos, Cyprus, November 2004, pp. 165–174.

[2] ——, "Monte Carlo model checking," in *Proc. of TACAS'05, the 11th International Conference on Tools and Algorithms for the Construction and Analysis of Systems*, ser. LNCS, vol. 3440. Edinburgh, Scotland: Springer Verlag, April 2005, pp. 271–286.

[3] S. Callanan, D. Dean, M. Gorbovitski, R. Grosu, J. Seyster, S. Smolka, and E. Zadok, "Software monitoring with bounded overhead," in *Proc. of NGS'08, the Next Generation Software Workshop at IPDPS*. Miami, Florida, USA: IEEE Press, April 2008, pp. 1–8.

[4] S. D. Stoller, E. Bartocci, J. Seyster, R. Grosu, K. Havelund, S. A. Smolka, and E. Zadok, "Runtime verification with state estimation," in *Proc. 2nd International Conference on Runtime Verification (RV 2011)*, ser. Lecture Notes in Computer Science. Springer-Verlag, Sep. 2011, (**Won best student paper award**).

[5] Z. Li, R. Grosu, P. Sehgal, S. Smolka, S. Stoller, and E. Zadok, "On the energy consumption and performance of systems software," in *In Proc. of SYSTOR'11, the 4th nnual International Systems and Storage Conference*. Haifa, Israel: ACM, June 2011.

[6] R. Grosu, G. Batt, F. Fenton, J. Glimm, C. L. Guernic, S. Smolka, and E. Bartocci, "From cardiac cells to genetic regulatory networks," in *In Proc. of CAV'11, the 23rd International Conference on Computer Aided Verification*, ser. LNCS. Cliff Lodge, Snowbird: Springer Verlag, July 2011.

[7] C. Bender and S. Orszag, *Advanced Mathematical Methods for Scientists and Engineers: Asymptotic Mathods and Perturbation Theory*. Springer, 1999.

[8] J. Durbin and S. Koopman, *Time Series Analysis by State Space Methods*. Oxford University Press, 2008.

[9] W. Wei, *Time Series Analysis: Univariate and Multivariate Methods*. Addison Wesley, 2006.

[10] S. Sastry, *Nonlinear Systems: Analysis, Stability and Control*. Springer, 1999.

[11] H. Khalil, *Nonlinear Systems*. Prentice Hall, 2002.

[12] E. Clarke, O. Grumberg, and D. Peled, *Model Checking*. MIT Press, 2000.

[13] G. Franklin, J. Powell, and A. Emami-Naeini, *Feedback Control of Dynamic Systems, Sixth Edition*. Prentice Hall, 2009.

[14] F. Szidarovsky and A. Bahill, *Linear Systems Theory*. CRC Press, 1992.

[15] G. Franklin, J. Powell, and M. Workman, *Digital Control of Dynamic Systems, Third Edition*. Addison Wesley Longman, Inc., 1998.

[16] L. Ljung, *System Identification (2nd ed.): Theory for the User*. Prentice Hall, 1999.

[17] I. Ross, *A Primer on Pontryagin's Principle in Optimal Contro*. Collegiate Publishers, 2009.

[18] A. Bryson and Y. Ho, *Applied Optimal Control: Optimization, Estimation and Control*. Taylor and Francis, 1975.

[19] M. Gondran and M. Minoux, *Graphs, Dioids and Semirings: New Models and Algorithms*. Springer, 2008.

[20] R. Grosu, "The Cayley-Hamilton theorem for noncommutative semirings," in *Proc. of CIAA'10, the 15th International Conference on Implementation and Application of Automata*, ser. LNCS, vol. 5469. Winnipeg, Canada: Springer Verlag, August 2010, pp. 194–208.

[21] ——, "Finite automata as time-invariant linear systems: Observability, reachability and more," in *Proc. of HSCC'09, the 12th International Conference on Hybrid Systems: Computation and Control*, ser. LNCS, vol. 5469. San Francisco, USA: Springer Verlag, April 2009, pp. 194–208.

[22] W. Kuich and A. Salomaa, *Semirings, Automata and Languages*. Springer, 1986.

[23] D. Angluin, "Learning regular sets from queries and counterexamples," *Information and Computation*, vol. 75, pp. 87–106, 1987.

[24] F. Howar, B. Steffen, and M. Merten, "Automata learning with automated alphabet abstraction refinement," *Proc. of VMCAI'11, the 12th Int. Conf. on Verification, Model Checking and Abstract Interpretation*, vol. 6538, pp. 263–277, January 2011.

[25] R. Ramadge and W. Wonham, "Supervisory control of a class of discrete event processes," *Journal on Control and Optimization*, vol. 25, no. 1, pp. 206–230, 1987.

[26] R. Kumar and V. Garg, *Modeling and Control of Logical Discrete Event Systems*. Springer, 1995.

[27] L. R. Rabiner, "A tutorial on hidden Markov models and selected applications in speech recognition," *Proceedings of the IEEE*, vol. 77, no. 2, pp. 257–286, 1989.

[28] R. Elliott, L. Aggoun, and J. Moore, *Hidden Markov Models: Estimation and Control*. Springer, 1995.

[29] Y. Ephraim and N. Merhav, "Hidden Markov processes," *IEEE Transactions on Information Theory*, vol. 48, pp. 1518–1569, Jun. 2002.

[30] P. Tabuada, *Verification and Control of Hybrid Systems*. Springer, 2010.

[31] J. L. Hellerstein, Y. Diao, S. Parekh, and D. M. Tibury, *Feedback Control of Computing Systems*. Wiley-IEEE Press, 2004.

[32] C. Myers, *Engineering Genetic Circuits*. CRC Press, 2010.

[33] E. M. Cherry and F. H. Fenton, "Visualization of spiral and scroll waves in simulated and experimental cardiac tissue," *New Journal of Physics*, vol. 10, p. 125016, 2008.

[34] C. H. Luo and Y. Rudy, "A dynamic model of the cardiac ventricular action potential. I. simulations of ionic currents and concentration changes," *Circulation Research*, vol. 74, no. 6, pp. 1071–1096, Jun. 1994.

[35] K. Ten Tusscher, D. Noble, P. Noble, and A. Panfilov, "A model for human ventricular tissue," *American Journal of Physiology*, no. 286, pp. H1573–H1589, 2004.

[36] V. Iyer, R. Mazhari, and R. Winslow, "A computational model of the human leftventricular epicardial myocytes," *Biophysical Journal*, no. 87, pp. 1507–1525, 2004.

[37] A. Bueno-Orovio, M. Cherry, and F. Fenton, "Minimal model for human ventricular action potentials in tissue," *Journal of Theoretical Biology*, no. 253, pp. 544–560, 2008.

[38] E. Bartocci, E. M. Cherry, J. Glimm, R. Grosu, S. A. Smolka, and F. Fenton, "Toward real-time simulation of cardiac dynamics," in *In Proc. of CMSB'11, the 9th International Conference on Computational Methods in Systems Biology*. Paris, France: ACM, September 2011.

[39] L. energy control of electrical turbulence in the heart, "S. luther and f.h. fenton and b.g. kornreich and a. squires and p. bittihn and d. hornung and m. zabel and j. flanders and a. gladuli and l. campoy and e.m. cherry and g. luther and g. hasenfuss and v.i. krinsky and a. pumir and r.f. gilmour and e. bodenschatz," *Nature*, vol. 475, no. 7355, pp. 235–239, 2011.