Proceedings of the
44th IEEE Conference on Decision and Control, and
the European Control Conference 2005
Seville, Spain, December 12-15, 2005

Robustness Analysis of Biochemical Networks using μ-Analysis and Hybrid Optimisation

Jongrae Kim, Declan G. Bates, Ian Postlethwaite, Lan Ma and Pablo A. Iglesias

Abstract— **Biological systems which have been experimentally verified to be robust to significant changes in their environments require mathematical models which are themselves robust. In this context, a necessary condition for model robustness is that the model dynamics should not be extremely sensitive to small variations in the model's parameters. Robustness analysis problems of this type have been extensively studied in the field of robust control theory, and have been found to be very difficult to solve in general. This paper describes how tools from robust control theory and nonlinear optimisation can be used to analyse the robustness of a recently proposed model of the molecular network underlying adenosine 3',5'-cyclic monophosphate (cAMP) oscillations observed in fields of chemotactic** *Dictyostelium discoideum* **cells. The network model, which consists of a system of seven coupled nonlinear differential equations, accurately reproduces the spontaneous oscillations in cAMP observed during the early development of** *D. discoideum***. The analysis in this paper reveals, however, that very small variations in the model parameters can effectively destroy the required oscillatory dynamics.**

I. INTRODUCTION

In [1], a network model of interacting proteins was proposed that can account for the spontaneous oscillations in adenylate cyclase (ACA) activity that are observed in homogenous populations of *Dictyostelium* cells four hours after the initiation of development. Analyses of the numerical solutions of the nonlinear differential equations making up the model suggest that it faithfully reproduces the observed periodic changes in adenosine 3',5'-cyclic monophosphate (cAMP). In particular, periods, amplitudes and phase relations between oscillations in enzyme activities and internal and external cAMP concentrations were seen to agree well with experimental observations, [1].

In the recent literature, the issue of "robustness" in biological systems has received considerable attention, see for example [2] and references therein. In particular, it has been widely acknowledged that, since biological systems themselves are often extremely robust (loosely speaking, easily able to cope with wide variations in environmental conditions) the mathematical models developed to represent them must also reflect this reality. In this context, model robustness thus relates to how *insensitive* the dynamics of the model are to variations in the values of its parameters that correspond to realistic changes in environmental conditions.

Previous analyses of the robustness of the Laub-Loomis Model have been carried out. In [1], the authors claim that the required stable oscillatory behaviour of the model is preserved even for 25-fold changes in the values of the model parameters, and that such changes only have a minor effect on the frequency of the resulting oscillations. Note, however, that the above results are based on simulations employing "trial and error" changes in one model parameter at a time. In [3], similar results are reported for one-parameter-at-a-time variations, using a more systematic bifurcation analysis. In the same paper, simultaneous variations in all model parameters are also considered, and the results indicate that the model dynamics are insensitive to simultaneous parameter variations of up to ±8%. In this paper, we extend the analysis employed in [3] to reveal that, in fact, very small simultaneous variations in model parameters can result in dramatic changes in the model's dynamics.

II. THE OSCILLATING BIOCHEMICAL NETWORK

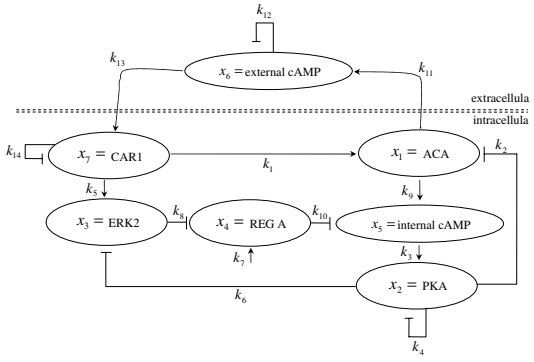

Fig. 1. Laub-Loomis biochemical network model for spontaneous oscillations in excitable cells of *Dictyostelium* [1].

The Laub-Loomis model for the oscillating biochemical network studied in this paper is shown in Figure 1. In this model, pulses of cAMP are produced when ACA is activated after the binding of extracellular cAMP to the surface receptor CAR1. When cAMP accumulates internally, it activates the protein kinase PKA. Ligand-bound CAR1 also activates the MAP kinase ERK2. ERK2 is then activated by PKA and no longer inhibits the cAMP phosphodiesterase REG A. A protein phosphatase activates REG A such that REG A can hydrolyse internal cAMP. When REG A

This work was supported by EPSRC research grant GR/S61874/01

J. Kim, D. G. Bates and I. Postlethwaite are with the Department of Engineering, University of Leicester, Leicester, LE1 7RH, United Kingdom, jrk7,dgb3,ixp@le.ac.uk

L. Ma is with the Department of Functional Genomics & Systems Biology, IBM T.J. Watson Research Center, P.O.Box 218, Yorktown Heights, NY, USA, lanma@us.ibm.com

P. A. Iglesias is with the Department of Electrical & Computer Engineering, The Johns Hopkins University, Baltimore, MD, USA, pi@jhu.edu

0-7803-9568-9/05/$20.00 ©2005 IEEE

TABLE I

THE KINETIC CONSTANTS : NOMINAL VALUES

Parameter	Nominal Value	Parameter	Nominal Value
k_1 [1/min]	2.0	k_8 [1/(Mol min)]	1.3
k_2 [1/(Mol min)]	0.9	k_9 [1/min]	0.3
k_3 [1/min]	2.5	k_{10} [1/(Mol min)]	0.8
k_4 [1/min]	1.5	k_{11} [1/min]	0.7
k_5 [1/min]	0.6	k_{12} [1/min]	4.9
k_6 [1/(Mol min)]	0.8	k_{13} [1/min]	23.0
k_7 [Mol/min]	1.0	k_{14} [1/min]	4.5

hydrolyses the internal cAMP, PKA activity is inhibited by its regulatory subunit, and the activities of both ACA and ERK2 go up. Secreted cAMP diffuses between cells before being degraded by the secreted phosphodiesterase PDE. The set of nonlinear differential equations proposed to describe the above dynamics is given by ([1], [3])

$$\begin{aligned}
\dot{x}_1 &= k_1 x_7 - k_2 x_1 x_2 \\
\dot{x}_2 &= k_3 x_5 - k_4 x_2 \\
\dot{x}_3 &= k_5 x_7 - k_6 x_2 x_3 \\
\dot{x}_4 &= k_7 - k_8 x_3 x_4 \\
\dot{x}_5 &= k_9 x_1 - k_{10} x_4 x_5 \\
\dot{x}_6 &= k_{11} x_1 - k_{12} x_6 \\
\dot{x}_7 &= k_{13} x_6 - k_{14} x_7
\end{aligned} \tag{1}$$

where $\dot{x}_i$ is the differentiation of x_i with respect to time, i.e., $\dot{x}_i = dx_i/dt$, x_1 is ACA, x_2 is PKA, x_3 is ERK2, x_4 is REG A, x_5 is internal cAMP, x_6 is external cAMP, and x_7 is CAR1. The nominal values for the kinetic constants, k_i, are given in Table I. Note that the above parameter values are slightly different to those given in [1]. This is because, as discussed in [3], the original paper contained some typographical errors. The values given above are those used in the previous analyses of [3], and are also those given on the website for the Laub-Loomis Model, http://www-biology.ucsd.edu/labs/loomis/network/laubloomis.html.

III. ROBUSTNESS ANALYSIS

Two methods for the analysis of uncertain system are used to evaluate the robustness of the biochemical network model to simultaneous variations in its parameters. The first employs the structured singular value μ, a tool developed in the field of robust control theory to measure the robustness of feedback control systems to various forms of uncertainty. The second uses a recently developed hybrid global/local optimisation algorithm to search for the smallest variation in the model parameters which drives the states of the system to a stable equilibrium point.

A. μ-Analysis

The structured singular value μ is defined as:

$$\frac{1}{\mu} \triangleq \min_{\Delta} \{\bar{\sigma}(\Delta) | \det(I - M(s)\Delta) = 0 \text{ for } \Delta \in B_\Delta\} \tag{2}$$

where $\bar{\sigma}(\cdot)$ denotes the maximum singular value, and B_Δ is a set of possibly real and/or complex uncertainties Δ, which in general has some structure. In words, the structured singular value is defined as the inverse of the smallest possible uncertainty Δ which will destabilise the closed loop system shown in Figure 2.

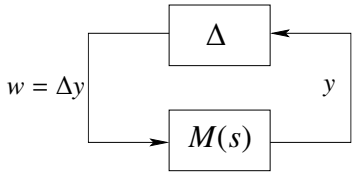

Fig. 2. M-Δ structure for μ-analysis.

For our particular problem, B_Δ will be seen to be a diagonal matrix of real scalars, since the "uncertain" parameters in our system are the kinetic constants represented by the real coefficients k_i in the model's differential equations. More details about μ-analysis can be found in [4] and [5] - here, we mention briefly only two important issues. Firstly, since μ-analysis is usually used to assess the robustness of stable linear time-invariant (LTI) systems, our oscillatory nonlinear model must be suitably transformed via a number of steps, which are described below. Secondly, the exact computation of μ is in general a non-polynomial (NP) time problem, which means that for large numbers of uncertainties we must settle for computing upper and lower bounds on μ. As discussed below, difficulties can arise in computing tight bounds on μ, depending on the number and type of uncertainties present in the system.

We now describe the process of transforming the nonlinear oscillatory model into a form which can be used for μ-analysis. Let the original nonlinear differential equations, (1), be written in compact form as

$$\dot{x} = f(x, k) \tag{3}$$

where $x = [\, x_1, x_2, \ldots, x_7 \,]^T$, $k = [\, k_1, k_2, \ldots, k_{14} \,]^T$, $f(\cdot, \cdot)$ is given by (1), and the superscript T is the transpose of a vector or matrix. With the nominal values of k, which are given in Table I, the model exhibits stable limit cycle trajectories with a period of approximately 7.31 minutes in all states. To obtain the limit cycle model, the following harmonic balance method is used: first, the solution following the limit cycle, i.e., the nominal trajectory $x_i^*(t)$, can be written as

$$x_i^*(t) = a_{0,i} + \sum_{n=1}^{\infty} a_{n,i} \cos\left(\frac{2\pi n t}{T} + \phi_{n,i}\right) \tag{4}$$

for $i = 1, 2, \ldots, 7$. Since the oscillations of the states of the system are closely approximated by simple harmonic oscillations with a constant offset, only the first two terms of the Fourier series are used. The substitution of this Fourier series into the original equations leads to a series of real algebraic equations for the coefficients, which can easily be solved using standard numerical software packages. Now,

the nonlinear differential equation can be linearised about the nominal solution, $x^*(t) = [\, x_1^*(t),\, x_2^*(t),\, \ldots,\, x_7^*(t)\,]^T$. To do this, the solution is perturbed as follows:

$$x(t) = x^*(t) + \delta x(t) \tag{5}$$

where $\delta x(t)$ is an arbitrary small perturbation away from $x^*(t)$. Differentiating both sides with respect to time yields

$$\dot{x}(t) = \dot{x}^*(t) + \delta\dot{x}(t) = f(x^*(t),\, k) + \delta\dot{x}(t) \tag{6}$$

Note that $\delta\dot{x}(t)$ is not the small perturbation of $\dot{x}(t)$ but the differentiation of $\delta x(t)$ with respect to time, which can be approximated as follows:

$$\delta\dot{x}(t) \approx \frac{\partial f(x^*(t),\, k)}{\partial x^*(t)}\, \delta x(t) \tag{7}$$

where the Jacobian, $\partial f(x^*(t),\, k)/\partial x^*(t)$, is equal to $\partial f(x(t),\, k)/\partial x(t)$ and we have substituted $x(t)$ for $x^*(t)$. We now write k as:

$$k_i = \bar{k}_i(1 + \delta_i) \tag{8}$$

where the nominal values, $\bar{k}_i$, i.e., when the δ_i are equal to zero, are given in Table I for $i = 1,\, 2, \ldots, 14$. Now, since the Jacobian is a function of k, it can be decoupled as follows:

$$\frac{\partial f(x^*(t),\, k)}{\partial x^*(t)} = A_0(x^*(t),\, \bar{k}) + B_0\tilde{\Delta}C_0(x^*(t),\, \bar{k}) \tag{9}$$

where $A_0(x^*(t),\, \bar{k})$ is the Jacobian matrix with all parameters at their nominal value, and $\tilde{\Delta}$ is a diagonal matrix containing all the uncertainties δ_i given by

$$\begin{aligned}\tilde{\Delta} = \mathrm{diag}[\, &\delta_1,\, \delta_2 I_2,\, \delta_3,\, \delta_4,\, \delta_5,\, \delta_6 I_2,\, \delta_8 I_2,\, \delta_9 \\ &\delta_{10} I_2,\, \delta_{11},\, \delta_{12},\, \delta_{13},\, \delta_{14}\,]\end{aligned} \tag{10}$$

where I_2 is 2×2 identity matrix. Note that δ_2, δ_6, δ_8, and δ_{10} are repeated two times, respectively and the dimension of $\tilde{\Delta}$ is therefore 17 even though the actual number of uncertain parameters is 13. Note also that because k_7 in (1) is not multiplied by any x_i, the Jacobian is not a function of k_7 and thus this parameter does not appear in the uncertain matrix. Expressions for $A_0(x^*(t),\, \bar{k})$, B_0, and $C_0(x^*(t),\, \bar{k})$ above can be found in [3]. In this study, we have used matrix manipulations to reduce the dimension of $\tilde{\Delta}$ to its minimal size of 13, to give:

$$\begin{aligned}\tilde{\Delta} = \mathrm{diag}[\, &\delta_1,\, \delta_2,\, \delta_3,\, \delta_4,\, \delta_5,\, \delta_6,\, \delta_8,\, \delta_9 \\ &\delta_{10},\, \delta_{11},\, \delta_{12},\, \delta_{13},\, \delta_{14}\,]\end{aligned} \tag{11}$$

For this $\tilde{\Delta}$, the corresponding $A_0(x^*(t),\, \bar{k})$ remains the same as before, i.e., the Jacobian of (1) evaluated at $x(t) = x^*(t)$. However, the B_0 and $C_0(x^*(t),\, \bar{k})$ matrices are changed to:

$$B_{0_{(i,j)}} = 1 \tag{12}$$

for (i, j) equal to $(1, 1)$, $(2, 3)$, $(3, 5)$, $(5, 8)$, $(6, 10)$, $(7, 12)$,

$$B_{0_{(i,j)}} = -1 \tag{13}$$

for (i, j) equal to $(1, 2)$, $(2, 4)$, $(3, 6)$, $(4, 7)$, $(5, 9)$, $(6, 11)$, $(7, 13)$, and otherwise $B_{0_{(i,j)}}$ is equal to zero, where $B_{0_{(i,j)}}$ is the i-th row j-th column element of B_0 and the size of B_0 is 7×13. $C_0(x^*(t), \bar{k})$ is a 13×7 matrix, which is given by

$$\begin{aligned}
&C_{0_{(1,7)}} = \bar{k}_1,\ C_{0_{(2,1)}} = \bar{k}_2 x_2^*(t),\ C_{0_{(2,2)}} = \bar{k}_2 x_1^*(t), \\
&C_{0_{(3,5)}} = \bar{k}_3,\ C_{0_{(4,2)}} = \bar{k}_4,\ C_{0_{(5,7)}} = \bar{k}_5, \\
&C_{0_{(6,2)}} = \bar{k}_6 x_3^*(t),\ C_{0_{(6,3)}} = \bar{k}_6 x_2^*(t), \\
&C_{0_{(7,3)}} = \bar{k}_8 x_4^*(t),\ C_{0_{(7,4)}} = \bar{k}_8 x_3^*(t), \\
&C_{0_{(8,1)}} = \bar{k}_9,\ C_{0_{(9,4)}} = \bar{k}_{10} x_5^*(t), \\
&C_{0_{(9,5)}} = \bar{k}_{10} x_4^*(t),\ C_{0_{(10,1)}} = \bar{k}_{11}, C_{0_{(11,1)}} = \bar{k}_{12}, \\
&C_{0_{(12,6)}} = \bar{k}_{13},\ C_{0_{(13,7)}} = \bar{k}_{14},
\end{aligned} \tag{14}$$

and otherwise $C_{0_{(i,j)}}(x^*(t), \bar{k})$ is equal to zero. As a result, the linear time varying differential equation for the perturbation is given by

$$\delta\dot{x}(t) = A_0(x^*(t), \bar{k})\delta x(t) + B_0\tilde{w}(t) \tag{15a}$$

$$\tilde{y}(t) = C_0(x^*(t), \bar{k})\delta x(t) \tag{15b}$$

where $\tilde{w}(t) = \tilde{\Delta}\tilde{y}(t)$. Note that A_0 and C_0 are state dependent, i.e., time varying matrices. Because $1/\mu$ is defined as the smallest norm of $\tilde{\Delta}$ such that (15) is destabilised, the smallest destabilising perturbation in k is sought in μ-analysis. Then, whenever the perturbation in k is inside this norm bound, the perturbation $\delta x(t)$ goes to zero as time increases. As a result, the original limit cycle will be stable and robust with respect to perturbations in k. However, to apply standard μ-analysis tools, all the matrices in (15) must be constant. To this end, the following further steps are required. Using a zero-order hold with a sampling time, h equal to T/n, where n is initially chosen equal to eight, (15) can be discretised as follows:

$$\delta x(\kappa + 1) = A_d(\kappa, \bar{k})\delta x(\kappa) + B_d(\kappa, \bar{k})\tilde{w}(\kappa) \tag{16a}$$

$$y(\kappa) = C_d(\kappa, \bar{k})\delta x(\kappa) \tag{16b}$$

where

$$A_d(\kappa, \bar{k}) = \Phi((\kappa + 1)h, \kappa h) \tag{17a}$$

$$B_d(\kappa, \bar{k}) = \int_{\kappa h}^{(\kappa+1)h} \Phi((\kappa + 1)h, t)B_0\, dt \tag{17b}$$

$$C_d(\kappa, \bar{k}) = C_0(x^*(\kappa h), k) \tag{17c}$$

and the state transition matrix, $\Phi(\cdot, \cdot)$ is given by

$$\Phi((\kappa + 1)h, \kappa h) = e^{\int_{\kappa h}^{(\kappa+1)h} A_0(x^*(\kappa h), \bar{k})\, dt} \tag{18}$$

Since A_d, B_d and C_d are function of $x^*(\kappa h)$ and $\bar{k}$, and $x^*(\kappa h)$ is periodic, the matrices A_d, B_d and C_d are also periodic. Now, using a techniques called "lifting", [6], these periodic matrices can be written as constant matrices, as follows. Without loss of generality, assume that $\kappa = 1$, then

$$\delta x(2) = A_d(1, \bar{k})\delta x(1) + B_d(1, \bar{k})\tilde{w}(1) \tag{19a}$$

$$\tilde{y}(1) = C_d(1, \bar{k})\delta x(1) \tag{19b}$$

and at $\kappa = 2$

$$\delta x(3) = A_d(2, \bar{k})\delta x(2) + B_d(2, \bar{k})\tilde{w}(2) \tag{20}$$

Now, substituting (19a) into (20) we get

$$\begin{aligned}\delta x(3) = &\left(\prod_{\kappa=1}^{2} A_d(\kappa,\bar{k})\right)\delta x(1) \\ &+ \left[A_d(2,\bar{k})B_d(1,\bar{k}), \quad B_d(2,\bar{k})\right]\tilde{w}_1^2\end{aligned} \tag{21}$$

where

$$\prod_{\kappa=1}^{2} A_d(\kappa) = A_d(2)A_d(1) \tag{22a}$$

$$\tilde{w}_1^2 = \left[\tilde{w}^T(1), \tilde{w}^T(2)\right]^T \tag{22b}$$

Note that the sequence of multiplication cannot be changed in general. Now, $\tilde{w}_1^2$ is equal to $\tilde{\Delta}_2\tilde{y}_1^2$, $\tilde{\Delta}_2 = \mathrm{diag}\left[\tilde{\Delta}, \tilde{\Delta}\right]$, and we define the accumulated output as follows:

$$\tilde{y}_1^2 = \begin{bmatrix} C_d(1) \\ C_d(2)A_d(1) \end{bmatrix}\delta x(1) + \begin{bmatrix} 0 & 0 \\ C_d(2)B_d(1) & 0 \end{bmatrix}\tilde{w}_1^2 \tag{23}$$

Repeating this procedure until κ equals n, which is the number of sample points in one period of the limit cycle, the following equations are obtained:

$$\delta x(\kappa+n) = \tilde{A}_d\,\delta x(\kappa) + \tilde{B}_d\,\tilde{w}_\kappa^{\kappa+n} \tag{24a}$$

$$\tilde{y}_\kappa^{\kappa+n} = \tilde{C}_d\,\delta x(\kappa) + \tilde{D}_d\,\tilde{w}_\kappa^{\kappa+n} \tag{24b}$$

where

$$\begin{aligned}\tilde{w}_\kappa^{\kappa+n} = &\left[\tilde{w}^T(k), \tilde{w}^T(k+1), \ldots,\right. \\ &\left.\ldots, \tilde{w}^T(k+n-1), \tilde{w}^T(k+n)\right]^T,\end{aligned} \tag{25}$$

$\tilde{w}_\kappa^{\kappa+n}$ is equal to $\tilde{\Delta}_n\tilde{y}_\kappa^{\kappa+n}$, and $\tilde{\Delta}_n$ is the block diagonal matrix with $\tilde{\Delta}$ repeated n-times. Note that all the matrices, $\tilde{A}_d$, $\tilde{B}_d$, $\tilde{C}_d$, and $\tilde{D}_d$, are time-invariant. Since the same $\tilde{\Delta}$ is repeated n-times, we can rearrange the $\tilde{\Delta}$ as follows:

$$\begin{aligned}\Delta = \mathrm{diag}[\,&\delta_1 I_n,\ \delta_2 I_n,\ \delta_3 I_n,\ \delta_4 I_n,\ \delta_5 I_n,\ \delta_6 I_n, \\ &\delta_8 I_n,\ \delta_9 I_n,\ \delta_{10} I_n,\ \delta_{11} I_n,\ \delta_{12} I_n,\ \delta_{13} I_n,\ \delta_{14} I_n\,]\end{aligned} \tag{26}$$

by defining the row re-ordering matrix F such that

$$F\tilde{\Delta}_n = \Delta F \tag{27}$$

where $F^T F = I_{17n}$. Thus, Equation (24) is transformed into:

$$\delta x(\kappa+n) = \tilde{A}_d\,\delta x(\kappa) + \tilde{B}_d F^T\,w_\kappa^{\kappa+n} \tag{28a}$$

$$y_\kappa^{\kappa+n} = F\tilde{C}_d\,\delta x(\kappa) + F\tilde{D}_d\,F^T w_\kappa^{\kappa+n} \tag{28b}$$

where $y_\kappa^{\kappa+n} = F\tilde{y}_\kappa^{\kappa+n}$, and $w_\kappa^{\kappa+n}$ is equal to $\Delta\,y_\kappa^{\kappa+n}$. Finally, using a zero-order hold or some other sampling methods, Equation (28) is transformed back to the continuous time domain with the sampling time T, to give

$$\delta\dot{x}(t) = A\,\delta x(t) + B\,w(t) \tag{29a}$$

$$\tilde{y}(t) = C\,\delta x(t) + D\,w(t) \tag{29b}$$

where $w(t)$ is equal to $\Delta\,y(t)$. Thus, a linear time-invariant system is obtained in the standard form so that μ-analysis techniques can be applied. $M(s)$ in Figure 2 is given by

$$M(s) = C\,(sI - A)^{-1}\,B + D \tag{30}$$

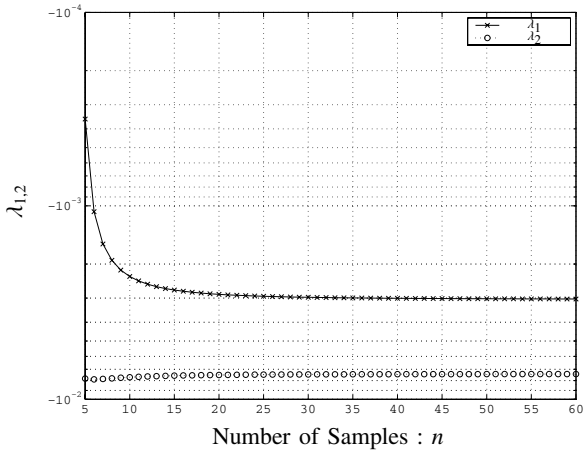

Fig. 3. Distances of the two closest eigenvalues from the imaginary axis as a function of n.

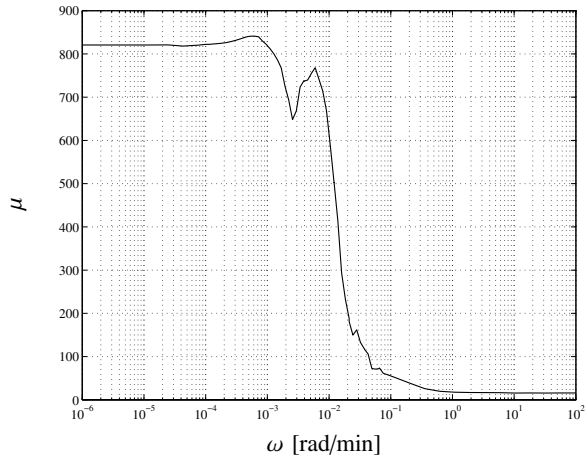

Fig. 4. μ upper bound calculated using μ-toolbox at each ω.

Now, the final issue to be resolved concerns the effect of varying the number of samples in one period, i.e., n, on the final LTI system. Figure 3 shows the variation in the distance of the two closest eigenvalues of the nominal system from the imaginary axis with respect to the number of samples in one period. For example, for n equal to eight the eigenvalues of the nominal system, i.e., the eigenvalues of A, are given by

$$\lambda_1 = -0.0019,\ \lambda_2 = -0.0078,\ \lambda_{3,4,5,6,7} = -0.2703 \tag{31}$$

However, for n equal to 39, the eigenvalues are given by

$$\lambda_1 = -0.0030,\ \lambda_2 = -0.0074,\ \lambda_{3,4,5,6,7} = -0.2703 \tag{32}$$

Since, as can be seen from Figure 3, the position of the eigenvalues is almost completely unchanged for $n \geq 39$, this value was used for the subsequent μ bound computations. Application of the standard algorithms for computing bounds on μ, [4], to the above system produced the results shown in Figure 4. The inverse of the peak of the upper

bound on μ provides a maximum allowable level of uncertainty for which stable oscillations in the original nonlinear system are guaranteed to persist. From the figure, however, this corresponds to a maximum allowable percentage variation in the parameters k_i of only 1/842 = 0.12%, indicating (possibly) very poor robustness indeed. Unfortunately, due to the large number of repeated real parameters in the Δ matrix, the μ lower bound algorithms fail to converge and for the frequency range in the figure are all zero everywhere. It is thus not possible to definitively establish from this analysis whether the indicated lack of robustness is in fact true (μ is close to its upper bound), or not (μ is much smaller than the computed upper bound, i.e. the upper bound is conservative). It is of particular interest to resolve this issue since these results differ significantly from the analysis of [3], which proposed maximum allowable percentage variations in the parameters k_i of 8.3%. In the next section we resolve this issue by using a recently developed hybrid global/local optimisation algorithm to search for the smallest variation in the model parameters which drives the states of the nonlinear model to a stable equilibrium point. As will be seen, the possible lack of robustness indicated by the above analysis is in fact very close to the true situation.

B. Hybrid Optimisation

In robustness analysis, optimisation algorithms can be used to search for particular combinations of parameters in the "uncertain parameter space" which maximise or minimise a particular cost function. Local optimisation methods, e.g. sequential quadratic programming (SQP), [7], that use gradient information are computationally efficient but can, of course, easily get locked into local optima in the case of multimodal search spaces. Global optimisation methods such as Genetic Algorithms (GAs) [8], on the other hand, use stochastic search and evolutionary principles to try to approach the true global optimum, albeit at the cost of vastly increased computation times. In the recent literature, several researchers have proposed combining the two approaches, ([9], [10]), and in [11], some guidelines are provided on designing hybrid GAs, along with experimental results and supporting mathematical analysis. In the present application, a probabilistic switching scheme based on that proposed in [11] is used. The scheme is based on the idea of associating a reward (or gain) with each method which reflects the effectiveness of that method at each iteration. The reward associated with each method then determines the probability of that method being chosen at the next iteration. A simple way to assign a reward is with a weighted geometric average [11]:

$$W^{k+1}_{\text{GA or Local}} = W^{k}_{\text{GA or Local}}(1-c) + cR^{k}_{\text{GA or Local}} \tag{33}$$

where W^k denotes the weighted reward, R^k is the reward at the iteration k, and c is a constant in [0, 1], which is a design parameter. The resulting switching algorithm is summarised in Table II. To avoid getting trapped in local optima, at the beginning of the optimisation the GA should have a higher probability to be chosen than the local algorithm. Hence, initially the weights for the GA and the local algorithm are given as 0.9 and 0.1, respectively. The HGA scheme starts from a randomly generated population of candidates. The initial guess for the local algorithm is taken from the population depending on the calculation mode. There are two modes in the algorithm, search and confirm. In search mode, the initial guess is chosen from the two best in the population. In confirm mode, the initial guess is chosen from a subset of the population, chosen to be far away from the current best. From here onwards the decision-making is done based on probability matching depending on the rewards associated with each of the optimisation schemes. The probability of selecting the GA at any iteration can be calculated from the following equation [11]:

$$P^{k}_{\text{GA}} = W^{k}_{\text{GA}} / \left(W^{k}_{\text{GA}} + W^{k}_{\text{Local}} \right) \tag{34}$$

A random number generator simulates a coin toss and depending on the result one of the optimisation schemes is chosen and proceeded with. If the scheme chosen is global optimisation, it proceeds with only one generation. If the local scheme is chosen, then the optimisation runs until it either converges or reaches the defined maximum number of cost function evaluations. At the end of a run of either of the optimisation schemes, the improvement achieved above the value of the best solution prior to the optimisation run is checked. The reward for a particular, local or global, optimisation is assigned, the probabilities are updated and the sequence is repeated until no improvement occurs from either of the two methods. To apply the hybrid algorithm to

TABLE II
HYBRID GENETIC ALGORITHM

1) Initialize $W^0_{\text{GA}} = 0.9$, $W^0_{\text{Local}} = 0.1$, $c = 0.3$, $k = 1$, set the calculation mode "Search", the number of confirmation zero, and generate initial population for GA
2) While the confirmation number is less than a certain number (e.g. 20)
   a) Calculate P^k_{GA}, (34)
   b) (Flip Coin) = a random number between zero and one
   c) If (Flip Coin) < P^k_{GA} then run GA and update W^k_{GA}, (33)
   d) else choose the local algorithm with the following initial guess
      i) If the calculation mode is "Search", choose one randomly from two best in the population,
      ii) else choose one randomly from the subset of population where the distance of each element from the current best is out of 1σ (standard deviation of the population from the current best)
      iii) Update W^k_{Local}, (33)
   e) If the cost does not improve,
      i) Initialize the following every five confirmation: population, $W^0_{\text{GA}} = 0.5$, $W^0_{\text{Local}} = 0.5$, $c = 0.6$ and set calculation mode equal to "Confirm"
      ii) Increase the number of confirmation
   f) else set the number of confirmation equal to zero
3) end of While

test the robustness of the model's limit cycle the following cost function is defined to be minimized:

$$\min_{\delta\in\Delta} J = \frac{1}{2}\int_{t=t_0}^{t=t_f} \dot{x}_1^2\, dt \qquad (35)$$

where $\delta = [\delta_1, \delta_2, \ldots, \delta_{14}]^T$, $\Delta = (p_\delta/100)\text{diag}[\delta]$, $\delta_i \in [-1, 1]$ for $i = 1, 2, \ldots, 14$, p_δ is in the range of $[0, \infty)$, and t_0 and t_f are chosen as 600 and 1200 minutes, respectively. Note that δ now includes all δ_i from $i = 1$ to 14 unlike in the μ-analysis where δ_7 could not be included. The reason for this choice of cost function is that the state derivative has to be zero whenever the limit cycle does not exist. The nonzero initial integration lower bound, $t_0 = 600$ minutes, is chosen to reduce the effect of initial transient responses on the cost function optimisation. Hence, the hybrid algorithm tries to find a δ combination for the given boundary of p_δ, which minimises the cost function. After the minimum is found by the algorithm, it should be checked whether the state converges to an equilibrium point or not by integrating the nonlinear differential equations with the given values for the k_i for a number of different initial conditions. Depending on the existence of a limit cycle, p_δ is then increased or decreased until the minimum p_δ, denoted p_δ^*, is found to whatever desired accuracy. Results of the application of the hybrid optimisation algorithm are shown in Figure 5, which shows ACA trajectories with the optimal combination of uncertainties inside the set Δ for three different values of p_δ. For all three cases, the optimal δ minimising the cost J occurs at the same boundary point, i.e.,

$$\delta^* = [-1, -1, 1, 1, -1, 1, 1, -1, 1, 1, -1, 1, -1, 1] \qquad (36)$$

From the figure, it can be clearly seen that even for p_δ equal to 0.6 (corresponding to ± 0.6% variations in the parameters) the optimisation algorithm is able to find a parameter combination that destroys the limit cycle in the network model. As the allowable variation in the model parameters is increased, the rate of decay of the oscillations becomes even more rapid - for a ±2% variation the oscillations have completely ceased in less than 6 hours, whereas in laboratory experiments *Dictyostelium* cells were observed to oscillate with a constant amplitude for 12-18 hours before they formed a spore. Thus our results confirm the poor robustness properties indicated in the previous μ-analysis, i.e. extremely small changes in the values of the model's parameters can destroy the required oscillatory behaviour.

IV. CONCLUSIONS

The observed ability of many biological systems to cope with large changes in their environments requires that the mathematical models proposed to represent their dynamics should do so robustly, i.e. their dynamics should not be extremely sensitive to small changes in the values of the model's parameters. In this paper, the robustness of a recently proposed model of the molecular network underlying adenosine 3',5'-cyclic monophosphate (cAMP) oscillations observed in fields of chemotactic *Dictyostelium discoideum* cells was evaluated. The analysis of the model revealed an extreme lack of robustness to variations in the values of its parameters. In the original paper proposing the model, the authors provide biochemical explanations for the choice of nominal values for only four of the fourteen parameters in the model - the nominal values of all other parameters were chosen by trial and error in order to match the model dynamics with the experimental data. An interesting open question is, therefore, whether a different set of nominal values for the model parameters could be found which would replicate the required oscillatory dynamics *and* also provide the required level of robustness.

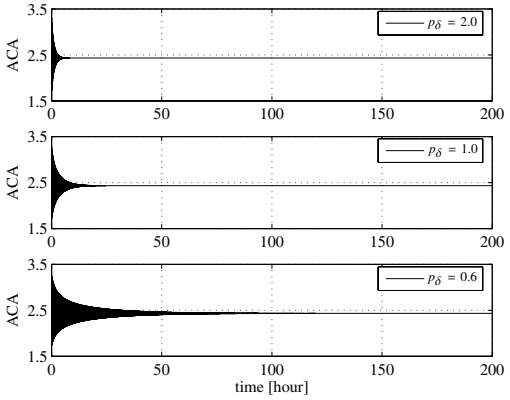

Fig. 5. Effect of different levels of parameter variation on oscillatory behaviour

Acknowledgments

This work was carried out under EPSRC research grant GR/S61874/01.

References

[1] M. T. Laub and W. F. Loomis, "A molecular network that produces spontaneous oscillations in excitable cells of dictyostelium," *Molecular Biology of the Cell*, vol. 9, pp. 3521–3532, 1998.

[2] M. Morohashi, A. E. Winn, M. T. Borisuk, H. Bolouri, J. Doyle, and H. Kitano, "Robustness as a measure of plausibility in models of biochemical networks," *Journal of Theoretical Biology*, vol. 216, pp. 19–30, 2002.

[3] L. Ma and P. A. Iglesias, "Quantifying robustness of biochemical network models," *BMC Bioinformatics*, vol. 3, no. 38, 2002.

[4] G. J. Balas, J. C. Doyle, K. Glover, A. Packard, and R. Smith, *μ-Analysis and Synthesis Toolbox User's Guide, Version 3*, The Mathworks, 2001.

[5] S. Skogestad and I. Postlethwaite, *Multivariable Feedback Control*. New York: John Wiley, 1996.

[6] Chen and B. A. Francis, *Optimal Sampled-Data Control Systems*. Springer-Verlag, 1995.

[7] *MATLAB Optimization Toolbox User's Guide, Version 2*, The Mathworks, 2000.

[8] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*. Reading, MA: Addison-Wesley, 1989.

[9] L. Davis, *Handbook of Genetic Algorithms*. New York: Van Nostrand Reinhold, 1991.

[10] J. Yen, J. C. Liao, D. Randolph, and B. Lee, "A hybrid approach to modeling metabolic systems using genetic algorithm and simplex method," in *Proc. of the 15th World Congress of Int. Federation of Automat. Contr.*, 1995.

[11] F. G. Lobo and D. E. Goldberg, "Decision making in a hybrid genetic algorithm," IlliGAL Report No. 96009, Tech. Rep., 1996.