

# Model for a Population-based Microbial Oscillator

Angel Goñi-Moreno<sup>a,\*</sup>, Martyn Amos<sup>a</sup>

<sup>a</sup>*School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University, M1 5GD, UK.*

---

## Abstract

Genetic oscillators are a major theme of interest in the emerging field of synthetic biology. Until recently, most work has been carried out using *intra-cellular* oscillators, but this approach restricts the broader applicability of such systems. Motivated by a desire to develop large-scale, spatially-distributed cell-based computational systems, we present an initial design for a *population-level* oscillator which uses three different bacterial strains. Our system is based on the client-server model familiar to computer science, and uses quorum sensing for communication between nodes. Importantly, it is robust to perturbation and noise. We present the results of extensive *in silico* simulation tests, which confirm the feasibility of our design.

*Keywords:* Synthetic Biology, Bacteria, Population, Oscillator, Simulation

---

## 1. Introduction

The growing field of synthetic biology (Benner and Sismour, 2005; Purnick and Weiss, 2009; Serrano, 2007) has the potential to impact on many pressing areas of concern, such as health (Lu and Collins, 2009; Ro et al., 2006), energy (Lee et al., 2008) and the environment (Sayler et al., 2004). By engineering bacteria (and sometimes other types of cell), practitioners in the field hope to take advantage of their inherent “biological nanotechnology”. This engineering is generally achieved by modifying the natural transcriptional mechanisms and regulatory activities of the bacterium of interest. Collections of bacterial cells have recently been successfully engineered to perform

---

\*Corresponding author at: School of Computing, Mathematics and Digital Technology, Manchester Metropolitan University, M1 5GD, UK. T: +44 (0)161 247 2816

*Email address:* a.moreno@mmu.ac.uk (Angel Goñi-Moreno)

simple tasks, such as emulating light-sensitive film (Levskaya et al., 2005), or generating simple patterns (Basu et al., 2005; Sohka et al., 2009). By harnessing and controlling communication and synchronization mechanisms found in such systems, we hope to engineer scalable, robust, *fault-tolerant* bacterial devices based on the potential of using microbial consortia (Brenner et al., 2008).

A significant amount of work on synthetic biology has concerned *switches* (Gardner et al., 2000) and *oscillators*; here, we focus on the latter. Our objective is to design a multi-strain bacterial community with autonomous behaviour. We model our system on the “client-server” architecture familiar to computer science (Berson, 1996), with a single central server and two clients (one “red” and the other “green”). The task we define is that of *oscillation*; by engineering feedback between three different strains, we obtain indefinite switching between “red” and “green” outputs.

In physics, an oscillator is a system that produces a regular, periodic “output”. Familiar examples include a pendulum or a vibrating string. Linking several oscillators together in some way gives rise to *synchrony* – for example, heart cells repeatedly firing in unison, or millions of fireflies blinking on and off, seemingly as one (Strogatz, 2003). Although synthetic genetic oscillators date back to the early 1960s (Goodwin, 1963), these so-called *Goodwin oscillators* were limited to single genes. The archetype of the *multi-gene* oscillator is known as the *repressilator*, which is a “ring” of genes, each repressing its successor (Elowitz and Leibler, 2000; Müller et al., 2006). A detailed discussion of synthetic genetic oscillators is beyond the scope of this paper, but we refer the reader to a recent extensive survey (Purcell et al., 2010).

Recently, genetic clocks have been coupled to produce synchronised oscillations at the level of a cell *population* (Danino et al., 2010). Following on from earlier theoretical work (Garcia-Ojalvo et al., 2004; McMillen et al., 2002), this paper demonstrated the feasibility of engineering population-level oscillations. However, the population used was *homogenous*. In nature, there exist bacterial communities known as *biofilms* (Davies et al., 1998), in which hundreds of bacterial species form a robust and stable community through signalling and cooperation. If the potential of synthetic biology is to be fully realised, we believe that it is important to understand how to engineer communication in *mixed* groups of cells. Two recent papers (Tamsir et al., 2010; Regot et al., 2011) describe different approaches to multi-cellular computing, each using a “compartmentalized” approach which restricts particular logical operations to specific cell types. The benefits of such encapsulation are simi-

lar to those obtained by the *object-oriented* model of computer programming, and include (1) ease of implementation (only a small number of components need to be introduced into any single cell type), (2) opportunities for module reuse, and (3) the suppression of noise. As the conceptual link with computer science is clear and useful, we continue the theme in the current paper, by using the *client-server* model to illustrate our three-component oscillator.

Although the use of chemical signals leads to an inherent slowdown, thus limiting the applicability of such modular systems, they may find applications in domains where robustness or noise tolerance are important. In a commentary article (Li and You, 2010), Li and You suggest that “division of labour may be useful in metabolic engineering, in which intermediate metabolites are produced by distinct cell populations and serve simultaneously as wiring molecules. In this case, the fruits of the divided labour could be a useful chemical or protein product, whose synthesis is collectively carried out by the chemically wired populations.”

In this paper we first describe the architecture of our system. We describe *in silico* component testing results, before demonstrating, using extensive simulation studies, the feasibility of engineering multi-strain, *population-based oscillators*. Our results suggest that such distributed computations may become more common as the field of synthetic biology matures.

## 2. Models

### 2.1. A multi-strain bacterial oscillator

In this Section we describe in detail the structure of our population-based client-server oscillator. This system achieves oscillatory behaviour (switching from red to green light output) in an autonomous, synchronous fashion. The basic architecture of our system, depicted in Figure 1, is based on the “client-server” principle of modern computing, in which distributed *client* nodes communicate with a central *server* (Berson, 1996). In our system, we have one server strain and two client strains. We extend the analogy by considering the role of the *buffer*, a nutrient solution in which the cells live and grow (in computing, a buffer is a region of memory in which temporary data are stored). Signals are transmitted between client and server via this “shared memory”, through the actions of sensing and deposition. Each cell (“processor”) also has its own private internal “memory”, corresponding to the space inside the membrane where local functions are performed. As each bacterium is, in effect, an independent processor, the success of our design

relies on our ability to make all processors react *simultaneously* to external signals.

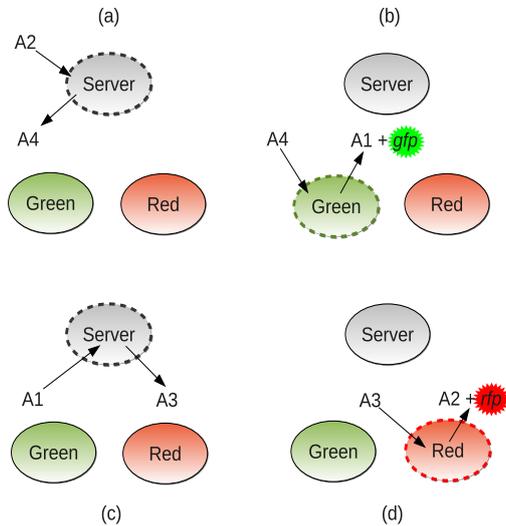


Figure 1: **Overall system architecture.** The figure shows a high-level schematic for the server and two clients. (A1...A4): autoinducers. (a) Server activated by A2. (b) Green client activated. (c) Server re-activated by A1. (d) Red client activated.

Quorum-sensing (QS) (Atkinson and Williams, 2009) has already been studied extensively in the context of synthetic biology (Andrianantoandro et al., 2006; Balagaddé et al., 2008; Garcia-Ojalvo et al., 2004). This mechanism facilitates inter-bacterial communication via the generation and receiving of signal molecules (Fuqua et al., 1994). Most importantly, it enables a community-level response to emerge once a certain *concentration threshold* has been reached. It is this mechanism that we will use as the basis of the current study. In what follows, there exist only four different signals or AHL molecules, labelled A1, A2, A3 and A4 (we use abstract labels for now, but supply *specific* molecules in a later Section). Each cell/processor reacts not to the *absence* or *presence* of a specific AHL signal, but to the signal *level*, or concentration. For that purpose, some *threshold*,  $\Delta$ , for input responses is defined for each cell. If the output of cell  $B_i$ , when we denote by  $O(B_i)$ , activates some *other* cell  $B_j$ , and a signal level is denoted by  $|x|$ , we assert that when  $|O(B_i)| \geq \Delta B_j$  then cell  $B_j$  is activated. In this way, our model attempts to address one of the biggest problems inherent to single-cell

circuits; stochastic expression noise (Murphy et al., 2010).

Table 1: Truth table for server strain. [A1... A4]: Autoinducers; S: Global output.

A1	A2	A3	A4	S
0	0	0	0	0
0	1	0	1	1
1	0	1	0	1
1	1	0	0	0

In Figure 1, we show a high-level schematic for the server and two clients; the server is *activated* by either A1 or A2 (producing A3 and A4, respectively); the *green* client is activated by A4, producing an “end-turn” signal, A1, and green fluorescent protein, and the *red* client is activated by A3, producing an end-turn signal, A2, and red fluorescent protein. We can therefore see how this machine lies dormant until either A1 or A2 is added to the nutrient, after which the system enters a period of oscillation (either red-green-red-... or green-red-green-... respectively). This is achieved by the server cells switching “turns” between red and green client cells. The intended behaviour of the server is shown in Table 1; the important thing to note is that it acts as an XOR (exclusive OR) function, since it is only active if, and only if, it receives a *single* input (i.e., when only *one* of the clients is active). If either *both* or *none* of the clients are active, then the server is *inactive* (its internal circuit does not allow it to produce output); this is *fundamental* to the correct operation of the system.

We now describe in detail the internal structure of the client and server bacterial strains. The first stage of this is to specify a candidate set of AHL molecules, corresponding to the various signals within the system (i.e., A1-4). These are listed in Table 2.

Table 2: Specific molecules corresponding to signals.

Signal	System	Molecule
A1	LuxI/R	3OC6AHL
A2	LasI/R	3OC12AHL
A3	Rh1I/R	C4AHL
A4	SinI/R	3OC14AHL

We select four specific quorum sensing systems using three criteria: (1) Sensitivity, (2) Bacterial class, and (3) Potential conflicts. In (Pai and You,

2009) the four systems are grouped together in terms of their sensitivity; in (Lerat and Moran, 2004) the systems are all characterised as being present in particular divisions of specific Proteobacteria, and in (Steindler and Venturi, 2007) it is established that there exist *no* conflicts between the molecules sensed by the different systems. We therefore assert that the systems we have chosen are appropriate for our model, and that the possibility of error due to cross-talk may be mitigated by the system design.

### 2.1.1. Server cells

The server cells lie at the heart of the system, as they are responsible for implementing the core switching behaviour. In order to implement this, we use two *hybrid promoters* (De Boer et al., 1983). These promoters are regulated by *two* inputs (one inducer and one repressor), and careful design allows them to be combined in a single device.

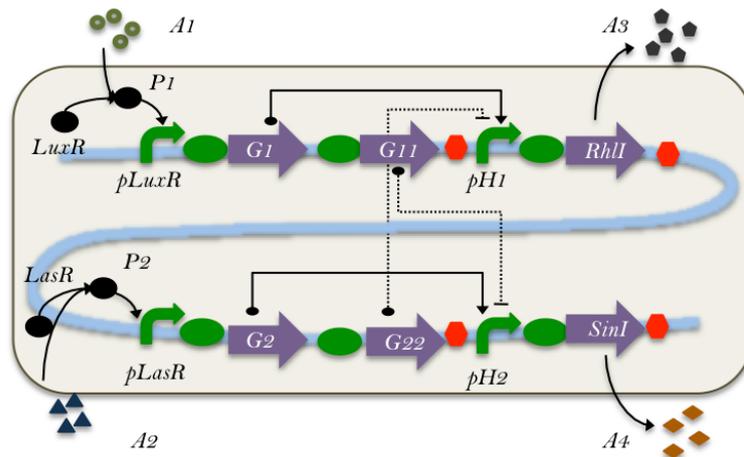


Figure 2: **Server cell internal architecture.** (A1..A4): AHL autoinducers; LuxR, LasR: Quorum Sensing receptors; P1,P2: active transcription factors; pLuxR, pLasR: inducible promoters; G1, G11, G2, G22: abstract structural genes; pH1, pH2: hybrid promoters; RhlI, SinI: structural genes involved in autoinducer production.

The detailed structure of the server is depicted in Figure 2. For clarity of description, we use the abstract molecular labels in our description. When a server bacterium detects, via its membrane, that the concentration of A1 molecules exceeds the input threshold for that particular quorum sensing (QS) system, the inducible promoter *pLuxR* is activated. As a result of this,

the two downstream structural genes are expressed. The production of  $G1$  molecules is used to stimulate a positive action in the *hybrid promoter pH1* which, in turn, manages the expression of  $A3$  molecules, using the gene  $RhlI$ . At the same time, the expression product of the second gene,  $G11$ , represses the hybrid promoter  $pH2$  so the production of  $A4$  is no longer possible due to the inhibition of  $SinI$ . This general subsystem design is duplicated in order for the server to be able to react symmetrically to each of its possible inputs.

### 2.1.2. Client cells

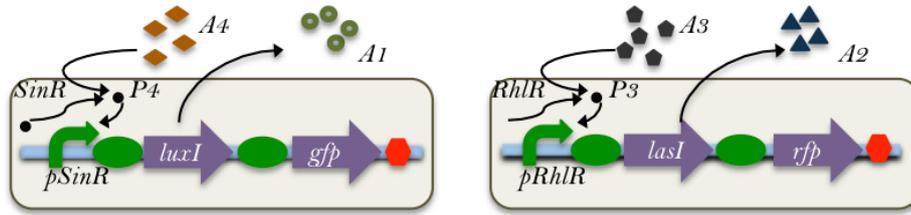


Figure 3: **Client cell internal architecture.** Left: green, Right: red. ( $A1\dots A4$ ): AHL autoinducers; SinR, RhlR: Quorum Sensing receptors; P3, P4: active transcription factors; pSinR, pRhlR: inducible promoters; LuxI, LasI: genes responsible for autoinducer production; *gfp*: green fluorescent protein; *rfp*: red fluorescent protein.

The detailed structures of the client cells are shown in Figure 3. These cells have a much simpler design, due to the lack of synchronization requirements on clients within our model. In the case of the green client, when it senses a sufficient concentration of  $A4$  molecules in the environment to raise the threshold of the corresponding QS system, it activates the internal pathway that concludes with the expression of  $A1$  molecules and the reporter, green fluorescent protein ( $GFP$ ). The first stage of this is the activation of the inducible promoter  $pSinR$  which allows the transcription and translation of the genes  $LuxI$  and  $gfp$ .  $LuxI$  is used to produce an “end-turn” signal (in this case,  $A1$ ), which is placed in the shared memory in order to notify the server that the green light that corresponds to a half-oscillation cycle has been satisfactorily expressed. The design of the red client is exactly the same, only with  $LasI$  replacing  $LuxI$ , and red fluorescent protein ( $RFP$ ) being produced instead of  $GFP$ .

## 2.2. Mathematical model

We first present the mathematical model, and then the results obtained from its computational equivalent. When simulating a genetic network, the key detail lies in the correct specification of the nature of connections between components (De Jong, 2002). Here, these connections are represented by ordinary differential equations (ODEs). In the next model, transcription and translation processes are combined into one single protein synthesis procedure.

### 2.2.1. Server equations

Equation (1) represents the formation of the active transcription factor which results from the binding, and later dimerization, of *LuxR* and *A1*. The product of the molecules is dimerized by the parameter  $\rho$ , as well as the final value is decreased by the degradation of the transcription factor. Analogously, equation (2) describes the formation of *P2*.

$$\frac{dP_1}{dt} = \rho_{P_1} \cdot [A_1]^2 \cdot [LuxR]^2 - \delta_{P_1} \cdot [P_1] \quad (1)$$

$$\frac{dP_2}{dt} = \rho_{P_2} \cdot [A_2]^2 \cdot [LasR]^2 - \delta_{P_2} \cdot [P_2] \quad (2)$$

Equation (3) represents the rate of change of *G1* over time. This expression product is controlled by the *pLuxR* promoter, which is induced by *P1*. *G1* degradation is given by the degradation rate  $\delta_{G_1}$ . Equations (4), (5) and (6) represent similar processes for the transcription factors *G11*, *G2* and *G22* respectively.

$$\frac{dG_1}{dt} = \alpha_{G_1} \cdot \frac{[P_1]^{h_1}}{K_{d_1} + [P_1]^{h_1}} - \delta_{G_1} \cdot [G_1] \quad (3)$$

$$\frac{dG_{11}}{dt} = \alpha_{G_{11}} \cdot \frac{[P_1]^{h_2}}{K_{d_2} + [P_1]^{h_2}} - \delta_{G_{11}} \cdot [G_{11}] \quad (4)$$

$$\frac{dG_2}{dt} = \alpha_{G_2} \cdot \frac{[P_2]^{h_3}}{K_{d_3} + [P_2]^{h_3}} - \delta_{G_2} \cdot [G_2] \quad (5)$$

$$\frac{dG_{22}}{dt} = \alpha_{G_{22}} \cdot \frac{[P_2]^{h_4}}{K_{d_4} + [P_2]^{h_4}} - \delta_{G_{22}} \cdot [G_{22}] \quad (6)$$

Equation (7) corresponds to the rate of change of the  $A3$  autoinducer as it is produced by gene  $RhlI$ . The upstream promoter,  $pH1$ , is induced by  $G1$  and repressed by  $G22$ . The final concentration is decreased by degradation. In the same way, equation (8) denotes the rate of change of  $A4$ .

$$\frac{dA_3}{dt} = \alpha_{A_3} \cdot \frac{[G_1]^{h_5}}{K_{d_5} + [G_1]^{h_5}} \cdot \frac{1}{1 + \left(\frac{[G_{22}]}{\beta_{G_{22}}}\right)^{h_6}} - \delta_{A_3} \cdot [A_3] \quad (7)$$

$$\frac{dA_4}{dt} = \alpha_{A_4} \cdot \frac{[G_2]^{h_7}}{K_{d_6} + [G_2]^{h_7}} \cdot \frac{1}{1 + \left(\frac{[G_{11}]}{\beta_{G_{11}}}\right)^{h_8}} - \delta_{A_4} \cdot [A_4] \quad (8)$$

### 2.2.2. Green client equations

The next three equations represent the behaviour of the Green Client. Equation (9) represents the formation of the complex  $P4$  by the binding and dimerization of  $SinR$  and  $A4$ . The concentration of the complex is affected by a degradation rate,  $\delta_{P_4}$ .

$$\frac{dP_4}{dt} = \rho_{P_4} \cdot [A_4]^2 \cdot [SinR]^2 - \delta_{P_4} \cdot [P_4] \quad (9)$$

Equation (10) represents the rate of change of  $A1$  autoinducer production over time. The promoter that controls its gene is induced by  $P4$ . The final concentration is decreased by a degradation rate,  $\delta_{A_1}$ . The expression of  $GFP$  is controlled by the same promoter, so it reacts in a similar way (equation (11)).

$$\frac{dA_1}{dt} = \alpha_{A_1} \cdot \frac{[P_4]^{h_9}}{K_{d_7} + [P_4]^{h_9}} - \delta_{A_1} \cdot [A_1] \quad (10)$$

$$\frac{dGFP}{dt} = \alpha_{GFP} \cdot \frac{[P_4]^{h_{10}}}{K_{d_8} + [P_4]^{h_{10}}} - \delta_{GFP} \cdot [GFP] \quad (11)$$

### 2.2.3. Red client equations

The next three equations represent the behaviour of the Red Client. Equation (12) represents the formation of the complex  $P3$  by the binding and dimerization of  $RhlR$  and  $A3$ . The concentration of the complex is affected by a degradation rate,  $\delta_{P_3}$ .

$$\frac{dP_3}{dt} = \rho_{P_3} \cdot [A_3]^2 \cdot [RhlR]^2 - \delta_{P_3} \cdot [P_3] \quad (12)$$

Equation (13) represents the rate of change of the  $A2$  autoinducer over time. The promoter that controls its gene is induced by  $P3$ . The final concentration is decreased by a degradation rate,  $\delta_{A2}$ . The expression of  $RFP$  is controlled by the same promoter so it reacts in a similar way (equation 14).

$$\frac{dA_2}{dt} = \alpha_{A_2} \cdot \frac{[P_3]^{h_{11}}}{K_{d_9} + [P_3]^{h_{11}}} - \delta_{A_2} \cdot [A_2] \quad (13)$$

$$\frac{dRFP}{dt} = \alpha_{RFP} \cdot \frac{[P_3]^{h_{12}}}{K_{d_{10}} + [P_3]^{h_{12}}} - \delta_{RFP} \cdot [RFP] \quad (14)$$

### 2.3. System parameters

For all of the equations given, we use the parameter values shown in Table 3 to approximate the oscillatory behaviour. Those parameters corresponding to the deterministic simulation are also used in the stochastic simulation (adding the noise parameter  $\xi$ ). We distinguish between the parameters for which value has been directly obtained from the literature (denoted by ‘‘Ob.’’) and those whose value has been estimated (‘‘Es.’’). In both cases, we supply an appropriate reference.

## 3. Results

We now describe the results of simulation-based experiments to investigate the behaviour of both the individual components, and the client-server system as a whole.

Our simulations were implemented using Python, which provides good packages for cellular modelling (Olivier et al., 2002), and the standard ODE-PACK package (Hindmarsh, 1983) for ODE solving. In order to obtain different levels of detail we run three sets of simulations. The first (deterministic) simulation investigates *idealised* differential behaviour, the second (stochastic) simulation shows a more realistic approach, by adding noise to the equations, and the third set of experiments assesses the effects of *inter-cellular* interactions.

Table 3: Parameter values. “Ob.” = Obtained from; “Es.” = Estimated from. (\*) This concentration is fixed in the system as those proteins are constitutively expressed in cells.  $\star$ : Basu et al. (2005);  $\circ$ : Pai and You (2009);  $\diamond$ : Brenner et al. (2007);  $\bullet$ : Basu et al. (2004);  $\dagger$ : Paulsson (2004);  $\triangleleft$ : Balagaddé et al. (2008).

Parameter	Meaning	Value	Reference
Deterministic simulation			
$\alpha_{G_1}, \alpha_{G_{11}}, \alpha_{G_2}, \alpha_{G_{22}}$	Synthesis rates	$1 \mu\text{M min}^{-1}$	(Ob.) $\star$
$\alpha_{G_2}, \alpha_{G_{22}}$	Synthesis rates	$1 \mu\text{M min}^{-1}$	(Ob.) $\star$
$\alpha_{A_1}, \alpha_{A_2}, \alpha_{A_3}, \alpha_{A_4}$	Synthesis rates	$0.18 \mu\text{M min}^{-1}$	(Es.) $\circ$
$\Delta_{Det}$	Q/S Threshold	$10 \mu\text{M}$	(Es.) $\diamond \bullet$
$\rho_{P_1}, \rho_{P_2}, \rho_{P_3}$ and $\rho_{P_4}$	Dimerization coefficients	$0.5 \mu\text{M}^{-3} \text{min}^{-1}$	(Ob.) $\star$
$h_1 \cdots h_7$	Hill coefficient	1	(Ob.) $\star$
$h_6, h_8$	Hill coefficient	2	(Ob.) $\star$
$K_{d_1} \cdots K_{d_4}$	Dissociation constants	$0.01 \mu\text{M}$	(Ob.) $\star$
$K_{d_5} \cdots K_{d_{10}}$	Dissociation constants	$1 \mu\text{M}$	(Ob.) $\star$
$\delta_{P_1}, \delta_{P_2}$	Protein decay	$0.0692 \text{min}^{-1}$	(Ob.) $\star$
$\delta_{A_3}, \delta_{A_4}$	Protein decay	$0.01 \text{min}^{-1}$	(Ob.) $\star$
LuxR/LasR/SinR/RhlR $\star$	Concentration	$0.5 \mu\text{M}$	(Ob.) $\star$
$\beta_{G_{22}}, \beta_{G_{11}}$	Repression coefficients	$0.08 \mu\text{M}$	(Es.) $\star$
Stochastic simulation			
$\xi$	Noise	8%	(Es.) $\dagger$
Discrete simulation			
$\alpha_{AHL}$	AHL synthesis rate	$10^{10} \text{molecules} \cdot \text{hr}^{-1}$	(Ob.) $\triangleleft$
$\Delta_{Dis}$	Q/S Threshold	$10^{15} \text{molecules}$	(Ob.) $\triangleleft$

### 3.1. Deterministic simulation

In Figure 4(a), the concentration of output molecules is high *if and only if* the molecule  $A2$  is present at very low concentrations. If the value of  $A2$  is enough to produce a significant amount of  $P2$ , red-specific *begin turn* signals will not be produced.

This set of simulations follows exactly the equations shown in previous section. Figure 4(a) shows the Server *intra-cellular* behaviour. Since noise-attenuation inside the server is one of the most important features of the model, the graphs show the rate of change of  $A3$  over time when  $A1$  is initialised to  $0.5 \mu\text{M}$  (inside the server), while  $A2$  changes in the interval  $[0.0 \mu\text{M} \cdots 0.5 \mu\text{M}]$  (inside the server). All molecules are affected by degradation.

Figure 4(b) shows the behaviour of the Green Client when  $A4$  is equal to  $0.5 \mu\text{M}$  (inside the client). The outputs,  $GFP$  and  $A1$ , are also decreased by degradation. The different behaviour of the curves in the graph is due to different rates of those genes in the equations.

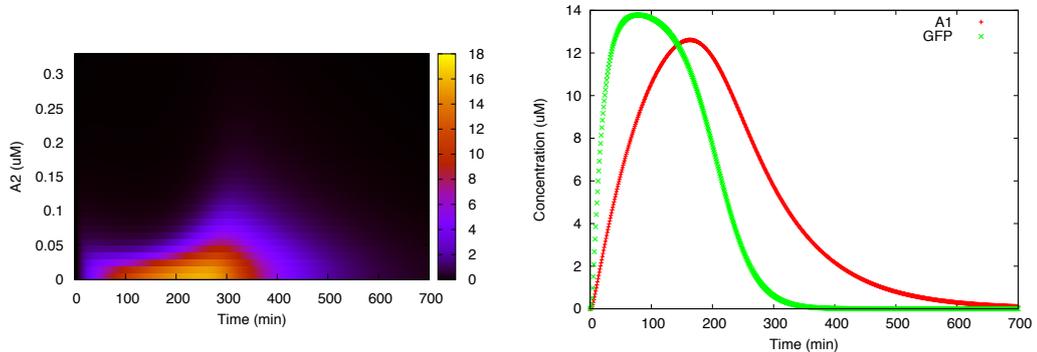


Figure 4: a) **Server *intra-cellular* behaviour. Map visualisation.** Fixed amount of  $A1 = 0.5\mu M$ . Plot depicts production of  $A3$  over time (min) while  $A2$  changes. b) **Green client.** Fixed amount of autoinducer  $A4 = 0.5\mu M$  inside the cell. All outputs are decreased by degradation.  $A1$ : AHL autoinducer;  $GFP$ : green fluorescent protein.

These simulation results confirm, deterministically and in principle, the correct functioning of the individual system components. We now describe the results of *full-system simulations* to assess the overall behaviour of our client-server model. First, we simulate the system comprising a *single* cell from each strain, so that the level of AHL produced by one cell is enough to fire the action of the next (threshold  $\Delta_{Det}$ ). The results are shown in Figure 5(a).

Although this system is formed by only three bacteria - one for each component - we may consider those results to be representative of a community in which  $n$  bacteria of each strain are perfectly mixed. We observe the expected pattern of oscillation, starting with the red client. The emerging behaviour is understood in terms of the individual component simulations performed before. At the beginning of computation, a significant amount of  $A1$  is introduced in the system, so that the server cell begins its computation. Then, the sequence of autoinducers  $[A3, A2, A4, A1]$  is repeated as the cycle *RFP* - *GFP* continues. In Figures 5(a-b), the initial amount of  $A1$  is not shown (only  $A1$  synthesised by *luxI*).

In Figure 5(b) we induce unusual system behaviour by changing one of the most important parameters. We modify the degradation rate of the *red off* signal molecules so that it is removed one order of magnitude more slowly from the system than the other signal molecules. The robustness of the system when altering these decay parameters is crucial, as they *will* be different in an *in vivo* experiment.

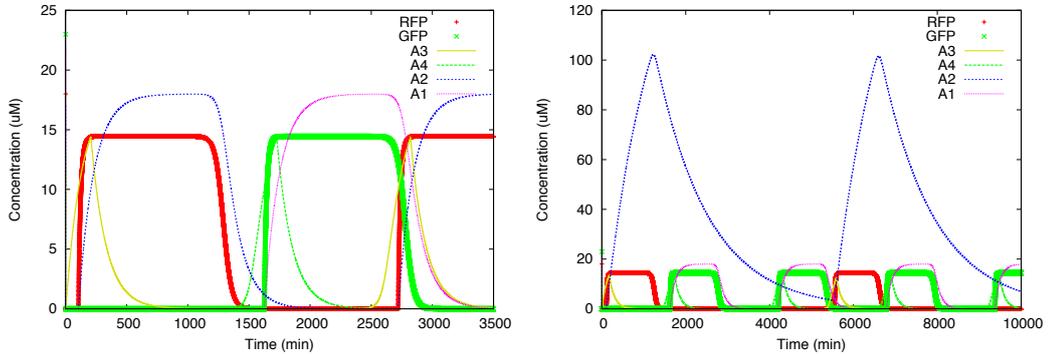


Figure 5: a) **Full-system simulation where 3 unique cells (one per each strain) interact.** The level of AHL produced by one cell is enough to fire the action of the next. b) **Full-system simulation. Decay of signalling molecules A2 initialised to 0.001.**; In both figures: (A1..A4): AHL autoinducers; RFP: red fluorescent protein (red client output); GFP: green fluorescent protein (green client output).

As we observe in Figure 5(b), the red client finishes its turn, but the *end-turn* molecules degrade more slowly, and are therefore still present in the shared memory. The server notices this red *end-turn*, and *repeatedly yields* the turn to the green client *until A2* molecules disappear. The system can therefore adapt its behaviour to this new situation. We do not observe a green-red-green-red... pattern, but instead see red-green-green-red... Importantly, the system *dynamically reconfigures* the oscillation pattern in a manner that is completely consistent with correct architectural behaviour.

### 3.2. Stochastic simulation

We now provide stochastic simulation results. Beyond our deterministic results, which may be described as the ideal behaviour, we need also take into account the inevitable fact of random noise in genetic networks. In the resulting SDEs (Stochastic Differential Equations), a Gaussian noise parameter ( $\xi$ ) with the same value in every equation is added to the previous value of the signal in each step of the integration (so that it is accumulative - this error is also applied to the concentration of autoinducers in the shared memory). Future bench experiments will help to determine a specific noise value for each reaction, based on the elements selected.

Figure 6(a) show the behaviour of the server with the new system of SDEs. Its profile is more “fuzzy” than in the deterministic system, but it still clearly reacts as an XOR gate. It is important to note, as this will

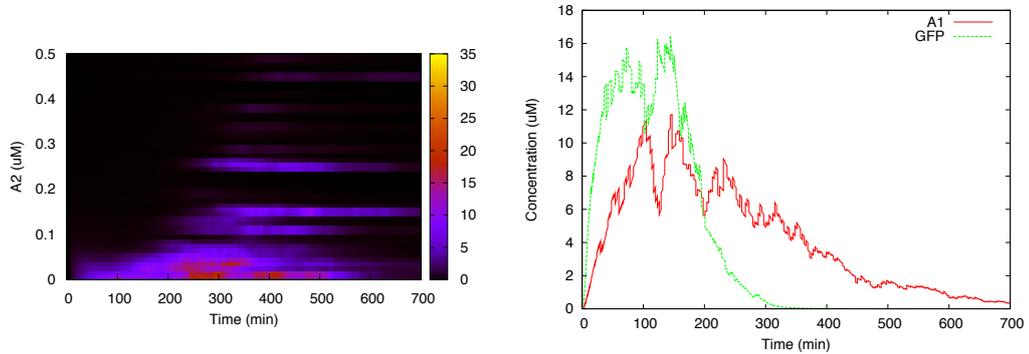


Figure 6: a) **Stochastic server simulation. Map visualisation.** Fixed amount of  $A1 = 0.5\mu M$ . Plot depicts production of  $A3$  over time (min) while  $A2$  changes. b) **Green Client. Stochastic simulation.** Fixed amount of autoinducer  $A4 = 0.5\mu M$  inside the cell. All outputs are decreased by degradation.  $A1$ : AHL autoinducer; GFP: green fluorescent protein.

be the key of the stochastic simulation, that the significant noise occurs at the moment in which the signals are almost null. Every fluctuation there will cause a quick change in the inputs of the server, and, as a consequence, noise can occur. That is why the correct selection of genes  $G1$ ,  $G11$ ,  $G2$  and  $G22$  and the promoters  $H1$  and  $H2$  is crucial. The main criteria to apply in selecting those components must be the following: as the repression forces in promoters are stronger, we will achieve a better deterministic behaviour but, at the same time, we will make noise fluctuations more frequent in the server. These are crucial considerations when designing our promoters. The clients react as expected, and the results are depicted in Figure 6(b).

Using normal parameter values, as seen in Figure 7(a), the stochastic simulation shows the expected behaviour. The difference we observe in this graph, compared to the previous set of results, is that the oscillations become quicker and the intersection between both colours bigger. A more unexpected behaviour is shown in Figure 7(b), where the decay rate of  $A2$  is again initialised to 0.001. Different runs will, of course, produce different results, but the behaviour of Figure 5(b) is hard to find, although oscillation hardly ever stops (approximately 1/10 times). In this graph we highlight the ability of the system to overcome a *difficult* situation and return to a normal oscillation: the intersection of lights at  $t \approx 3000$ .

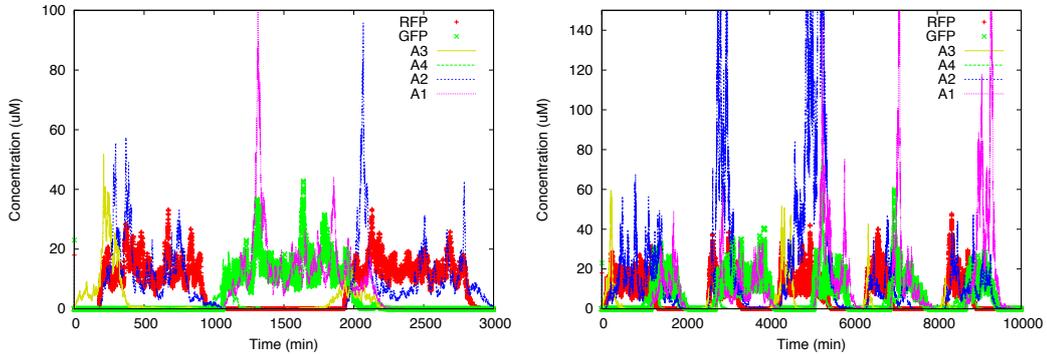


Figure 7: a) **Stochastic system simulation. 3 unique cells (one per each strain) interact.** b) **Stochastic system simulation. Decay of signalling molecules A2 initialised to 0.001.**; In both figures: (A1...A4): AHL autoinducers; RFP: red fluorescent protein (red client output); GFP: green fluorescent protein (green client output).

### 3.3. Perturbation analysis

In order to demonstrate the robustness of the model, we perform a *perturbation analysis* by altering the initial values of the system and assessing the impact (or otherwise) on system behaviour.

Figure 8 shows a collection of simulations of the system (10 deterministic, 10 stochastic). In each collection, a different initial set of parameters is tested. Given a specific parameter  $X_0$  (from table 3), we obtain the new altered parameter  $X_1$  by randomly choosing a new value between  $X_0 - 0.2 * X_0$  and  $X_0 + 0.2 * X_0$ . That is to say, we include flexibility in the initial parameters of 20% of their original value. The parameters that do *not* change are: *LuxR*, *LasR*, *SinR*, *RhlR* (constitutively expressed) and  $\Delta_{Det}$  (will vary only qualitative behaviour, depending on the size of the population).

As we conclude from Figure 8, the *quantitative* behaviour of the system changes depending on the initial conditions, but *synchronisation is still achieved*. We emphasise that the main objective of the model is to demonstrate the ability to synchronise “turns” of expression via a client/server architecture. Thus, the amplitude, phase and period of the oscillatory cycles may well vary from one implementation to another, but this is not quantitatively important in terms of the correct functioning of the device.

### 3.4. Discrete space simulation

The final set of simulations consider discrete spatial effects. We investigate how the multi-strain machine behaves when the three strains ( $10^4$  cells

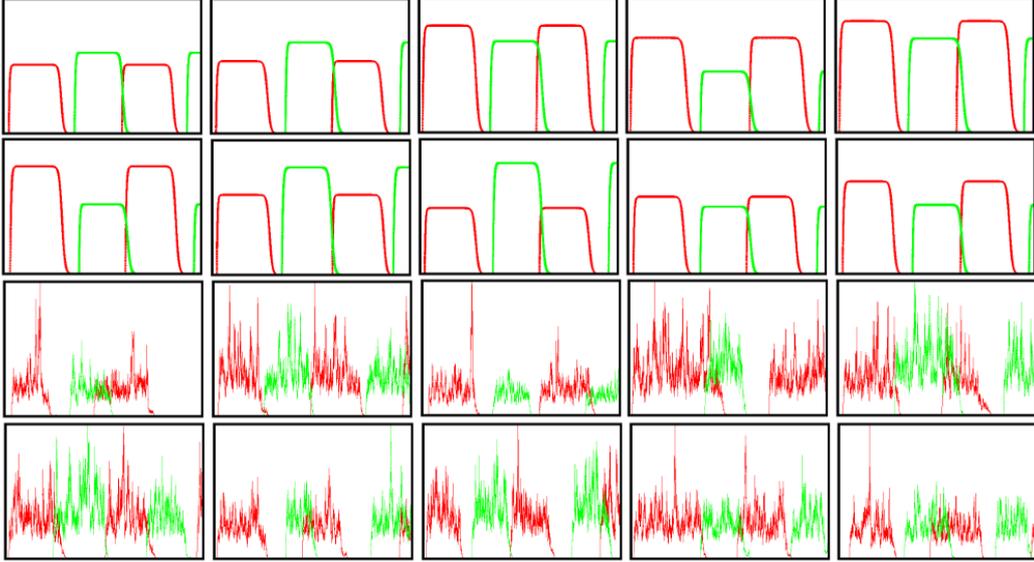


Figure 8: **Perturbation analysis.** Top two rows: deterministic simulations. Bottom two rows: stochastic simulations. Only variations over time of RFP (red) and GFP (green) are shown. X axis: time; Y axis: concentration.

per strain) are physically placed at different initial points. We therefore perform an agent-based simulation (Parunak et al., 1998), where each bacterium is modelled as a mobile “agent” that can move in a liquid environment, so that the three cell types may be found at different relative levels in different parts of the *in-silico* world (represented by a 2D grid). Note that this simulation considers only the production and degradation of AHL signalling molecules (and not intra-cellular processes).

The dimension of the matrix is  $100 \times 100$ , with each “square” big enough to allocate several cells at the same time (initially,  $\simeq 10$  cells). The signalling molecules are not explicitly represented, as this would be computationally expensive. Instead, the values of the four different AHL concentrations are stored in four buffers, and the area covered by  $\text{AHL}_i$  is considered to be the same as the area covered by strain  $i$ .

Initially, the server cells are placed randomly within the region of the matrix bounded by  $x=[0..30]$  to  $y=[0..30]$ , the green client cells in the region bounded by  $x=[70..99]$  to  $y=[0..30]$ , and the red client cells in the

region bounded by  $x=[35..65]$  to  $[70..99]$ . In order to simplify matters, the community is grown in a continuous culture.

All cells are simulated sequentially; every iteration corresponds to 1 hour, the cells producing at every iteration the corresponding amount of AHL (only if it is their turn) according to the previously-stated ratios ( $\alpha_{AHL}$  and  $\Delta_{Dis}$ ) from Table 3). During this step, every cell also moves across the world by choosing randomly between the eight squares around it. In this way, square is the distance they travel in an hour. As different motility speeds can be obtained using different liquids, the relation between time/space is not a key aspect of the simulation. The pseudo-code for a 700-hour simulation is:

```

iteration ← 0
WHILE iteration < 700
    FORALL bacteriai
        Move()
        CheckEnvironment()
        IF thresholdi = TRUE
            ProduceAHL()
        ENDIF
    ENDFOR
    DegradeAHL()
    iteration ++
ENDWHILE

```

where the degradation function uses the parameter shown in Table 3.

The results of the simulation are shown in Figure 9. When the server bacteria produces  $A_4$ , those green clients that enter the area of influence of the server bacteria sense the threshold of  $A_4$  autoinducers. At that moment,  $A_1$  is produced only by those client cells, and only the server cells in the area of the latter will sense the threshold of  $A_1$  when the concentration of those molecules exceeds the predefined value. The number of cells involved in this threshold sensing will increase during the simulation, as the strains are increasingly mixed.

The initial number of cells placed in the grid is much higher than the number needed to produce the enough AHL molecules to fire any threshold. Therefore, the machine does not need to be perfectly assembled (i.e., components are perfectly mixed) in order to start functioning.

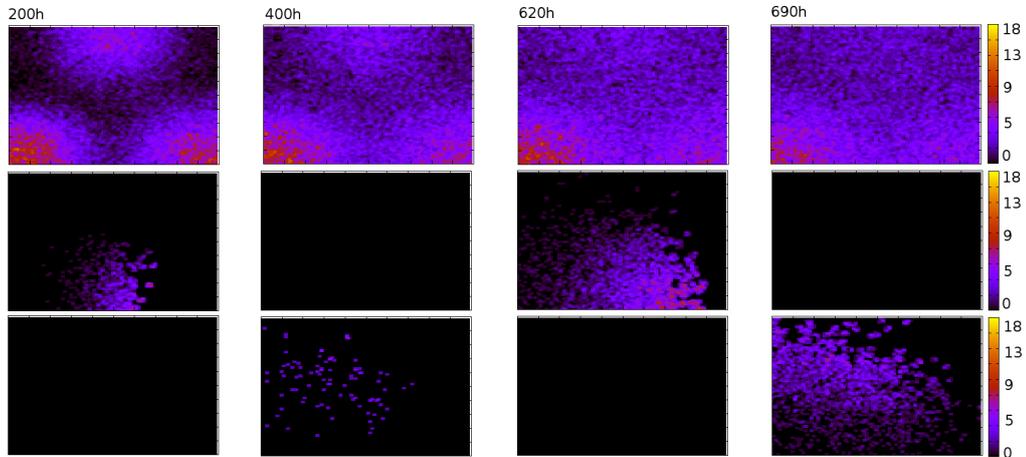


Figure 9: **The colony at several stages of the simulation.** (Top) All cells. Red client is central, server bottom-left, green client bottom-right. (Middle) Cells that express GFP. (Bottom) Cells that express RFP.

One remarkable feature of this simulation is the speed, or frequency, of oscillation cycles in the community. Nothing happens in the first 200 hours, until the green client cells make contact with the server cells (initially  $A_2$  molecules are introduced into the community). It is important to notice that the captures shown in Figure 9 simply represent the evolution of the community, not the period of the oscillatory cycles (between those steps there are several cycles).

While the simulation runs, the cycles occur much more often until the point around 600-700h, where the community is almost perfectly mixed, and the results - in terms of period of oscillation - are equivalent to those shown in previous simulations. This is due to the fact that when the population is mixed there are more cells expressing autoinducers, and thresholds are reached sooner.

As the local density of cells changes in the simulation, we can clearly see the effect of colony size in the behaviour of the system. Basically, it affects the time that the colony needs to fill the *buffer* in a specific local area. Along with the synthesis rate of autoinducers, it will determine the period of the cycles.

## 4. Discussion

In this paper we presented a design for a population-based cellular oscillator, which uses quorum sensing-based signalling within a client-server model. Simulation studies of our design suggest that it is realistic and robust to fluctuations in environmental conditions. Such systems will become increasingly important for synthetic biology, as the field seeks applications in (for example) distributed bio-sensing or tissue engineering. As more multi-strain devices appear, this kind of synchronisation will become increasingly important. The light oscillations in the proposed design are a proof of principle to test the client-server architecture. Future work will focus on refinements of the model, as well as its experimental validation.

## 5. Acknowledgments

We thank several anonymous reviewers for helpful comments on an earlier draft of this paper. This work was supported by the European Commission FP7 Future and Emerging Technologies Proactive initiative: Bio-chemistry-based Information Technology (CHEM-IT) (ICT-2009.8.3), project reference 248919 (BACTOCOM).

## References

- Andrianantoandro, E., Basu, S., Karig, D.K., Weiss, R., 2006. Synthetic biology: new engineering rules for an emerging discipline. *Mol. Syst. Biol.* 2:2006.0028 doi:10.1038/msb4100073.
- Atkinson, S., Williams, P., 2009. Quorum sensing and social networking in the microbial world. *Journal of The Royal Society Interface* 6(40):959.
- Balagaddé, F.K., Song, H., Ozaki, J., Collins, C.H., Barnet, M., Arnold, F.H., Quake, S.R., You, L., 2008. A synthetic escherichia coli predator-prey ecosystem. *Mol. Syst. Biol.* 4:187 doi:10.1038/msb.2008.24.
- Basu, S., Mehreja, R., Thiberge, S., Chen, M.T., Weiss, R., 2004. Spatiotemporal control of gene expression with pulse-generating networks. *Proceedings of the National Academy of Sciences of the United States of America* 101(17):6355 – 6360.

- Basu, S., Gerchman, Y., Collins, C.H., Arnold, F.H., Weiss, R., 2005. A synthetic multicellular system for programmed pattern formation. *Nature* 434(7037):1130–1134. doi:10.1038/nature03461.
- Benner, S.A., Sismour, A.M., 2005. Synthetic biology. *Nat. Rev. Genet.* 6(7):533–543. doi:10.1038/nrg1637.
- Berson, A., 1996. Client/server architecture. McGraw-Hill, Inc. New York, NY, USA.
- Brenner, K., Karig, D.K., Weiss, R., Arnold, F.H., 2007. Engineered bidirectional communication mediates a consensus in a microbial biofilm consortium. *Proceedings of the National Academy of Sciences of the United States of America* 104(44):17300–17304.
- Brenner, K., You, L., Arnold, F.H., 2008. Engineering microbial consortia: a new frontier in synthetic biology. *Trends. Biotechnol.* 26(9):483-9
- Danino, T., Mondragón-Palomino, O., Tsimring, L., Hasty, J., 2010. A synchronized quorum of genetic clocks. *Nature* 463(7279):326–330. doi:10.1038/nature08753.
- Davies, D.G., Parsek, M.R., Pearson, J.P., Iglewski, B.H., Costerton, J.W., Greenberg, E.P., 1998. The involvement of cell-to-cell signals in the development of a bacterial biofilm. *Science* 280(5361):295.
- De Boer, H.A., Comstock, L.J., Vasser, M., 1983. The tac promoter: a functional hybrid derived from the trp and lac promoters. *Proceedings of the National Academy of Sciences of the United States of America* 80(1):21.
- De Jong, H., 2002. Modeling and simulation of genetic regulatory systems: a literature review. *Journal of computational biology* 9(1):67–103.
- Elowitz, M.B., Leibler, S., 2000. A synthetic oscillatory network of transcriptional regulators. *Nature* 403(6767):335–3–38. doi:10.1038/35002125.
- Fuqua, W.C., Winans, S.C., Greenberg, E.P., 1994. Quorum sensing in bacteria: the LuxR-LuxI family of cell density-responsive transcriptional regulators. *Journal of Bacteriology* 176(2):269.

- Garcia-Ojalvo, J., Elowitz, M.B., Strogatz, S.H., 2004. Modeling a synthetic multicellular clock: repressilators coupled by quorum sensing. *Proc. Natl. Acad. Sci. (USA)* 101(30):10955–10960. doi:10.1073/pnas.0307095101.
- Gardner, T.S., Cantor, C.R., Collins, J.J., 2000. Construction of a genetic toggle switch in *Escherichia coli*. *Nature* 403:339–342.
- Goodwin, B.C., 1963. Temporal organization in cells. A dynamic theory of cellular control processes. Academic Press London.
- Hindmarsh, A.C., 1983. ODEPACK, a systematized collection of ODE solvers. pages 55–64. North-Holland, Amsterdam.
- Lee, S.K., Chou, H., Ham, T.S., Lee, T.S., Keasling, J.D., 2008. Metabolic engineering of microorganisms for biofuels production: from bugs to synthetic biology to fuels. *Curr. Opin. Biotechnol.* 19(6):556–663. doi:10.1016/j.copbio.2008.10.014.
- Lerat, E., Moran, N.A., 2004. The evolutionary history of quorum-sensing systems in bacteria. *Molecular Biology and Evolution* 21(5):903.
- Levskaya, A., Chevalier, A.A., Tabor, J.J., Simpson, Z.B., Lavery, L.A., Levy, M., Davidson, E.A., Scouras, A., Ellington, A.D., Marcotte, E.M., Voigt, C.A., 2005. Synthetic biology: engineering *Escherichia coli* to see light. *Nature* 438(7067):441–442 doi:10.1038/nature04405.
- Li, B., You, L., 2010. Division of logic labour. *Nature* 469:171–172, 2010.
- Lu, T.K., Collins, J.J., 2009. Engineered bacteriophage targeting gene networks as adjuvants for antibiotic therapy. *Proc. Natl. Acad. Sci. (USA)* 106(12):4629–4634. doi:10.1073/pnas.0800442106.
- McMillen, D., Kopell, N., Hasty, J., Collins, J.J., 2002. Synchronizing genetic relaxation oscillators by intercell signaling. *Proceedings of the National Academy of Sciences of the United States of America* 99(2):679.
- Müller, S., Hofbauer, J., Endler, L., Flamm, C., Widder, S., Schuster, P., 2006. A generalized model of the repressilator. *Journal of Mathematical Biology* 53(6):905–937.

- Murphy, K.F., Adams, R.M., Wang, X., Balázsi, G., Collins, J.J., 2010. Tuning and controlling gene expression noise in synthetic gene networks. *Nucleic acids research* 38(8):2712.
- Olivier, B.G., Rohwer, J.M., Hofmeyr, J.H.S., 2002. Modelling cellular processes with python and scipy. *Molecular Biology Reports* 29(1-2):249–254.
- Pai, A., You, L., 2009. Optimal tuning of bacterial sensing potential. *Molecular Systems Biology* 5(1).
- Paulsson, J., 2004. Summing up the noise in gene networks. *Nature* (427):415–418.
- Purcell, O., Savery, N.J., Grierson, C.S., di Bernardo, M., 2010. A comparative analysis of synthetic genetic oscillators. *Journal of The Royal Society Interface* doi:10.1098/rsif.2010.0183.
- Purnick, P.E.M., Weiss, R., 2009. The second wave of synthetic biology: from modules to systems. *Nat. Rev. Mol. Cell Biol.* doi:10.1038/nrm2698.
- Regot, S., Macia, J., Conde, N., Furukawa, K., Kjellen, J., Peeters, T., Hohmann, S., de Nadal, E., Posas, F. Solé, R., 2011. Distributed biological computation with multicellular engineered networks. *Nature* 469, 207–211.
- Ro, D.K., Paradise, E.M., Ouellet, M., Fisher, K.J., Newman, K.L., Ndungu, J.M., Ho, K.A., Eachus, R.A., Ham, T.S., Kirby, J., Chang, M.C.Y., Withers, S.T., Shiba, Y., Sarpong, R., Keasling J.D., 2006. Production of the antimalarial drug precursor artemisinic acid in engineered yeast. *Nature* 440(7086):940–943 doi:10.1038/nature04640.
- Sayler, G.S., Simpson, M.L., Cox, C.D., 2004. Emerging foundations: nano-engineering and bio-microelectronics for environmental biotechnology. *Curr. Opin. Microbiol* 7(3):267–273. doi:10.1016/j.mib.2004.04.003.
- Serrano, L., 2007. Synthetic biology: promises and challenges. *Molecular Systems Biology* 3. doi:10.1038/msb4100202.
- Sohka. T., Heins, R.A., Ostermeier, M., 2009. Morphogen-defined patterning of *escherichia coli* enabled by an externally tunable band-pass filter. *J. Biol. Eng.* 3(10) doi:10.1186/1754-1611-3-10.

- Steindler, L., Venturi, V., 2007. Detection of quorum-sensing N-acyl homoserine lactone signal molecules by bacterial biosensors. *FEMS Microbiology Letters*. 266(1):1–9.
- Strogatz, S., 2003. *Sync: The Emerging Science of Spontaneous Order*. Penguin.
- Tamsir, A., Tabor, J.J., Voigt, C.A., 2010. Robust multicellular computing using genetically encoded NOR gates and chemical wires. *Nature* doi:10.1038/nature09565.
- Parunak, H.V.D., Savit, R., Riolo, R.L., 1998. Agent-based modeling vs. equation-based modeling: A case study and users guide. In *Multi-Agent Systems and Agent-Based Simulation*, pages 10–25. Springer.