PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A MATHEMATICAL, PHYSICAL & ENGINEERING SCIENCES

Analog synthetic biology

R. Sarpeshkar

Phil. Trans. R. Soc. A 2014 **372**, 20130110, published 24 February 2014

PHILOSOPHICAL TRANSACTIONS OF THE ROYAL SOCIETY A

rsta.royalsocietypublishing.org

Research

Cite this article: Sarpeshkar R. 2014 Analog synthetic biology. *Phil. Trans. R. Soc. A* **372**: 20130110.
http://dx.doi.org/10.1098/rsta.2013.0110

One contribution of 11 to a Discussion Meeting Issue 'Beyond Moore's law'.

Subject Areas:
computational biology, biochemistry, biophysics, bioenergetics, chemical engineering, electrical engineering

Keywords:
analog computation, synthetic biology, cytomorphic, logarithmic computation, probabilistic computation, bioenergetics

Author for correspondence:
R. Sarpeshkar
e-mail: rahuls@mit.edu

Analog synthetic biology

R. Sarpeshkar

Analog Circuits and Biological Systems, Research Lab of Electronics, Massachusetts Institute of Technology, Cambridge, MA 02139, USA

We analyse the pros and cons of analog versus digital computation in living cells. Our analysis is based on fundamental laws of noise in gene and protein expression, which set limits on the energy, time, space, molecular count and part-count resources needed to compute at a given level of precision. We conclude that analog computation is significantly more efficient in its use of resources than deterministic digital computation even at relatively high levels of precision in the cell. Based on this analysis, we conclude that synthetic biology must use analog, collective analog, probabilistic and hybrid analog–digital computational approaches; otherwise, even relatively simple synthetic computations in cells such as addition will exceed energy and molecular-count budgets. We present schematics for efficiently representing analog DNA–protein computation in cells. Analog electronic flow in subthreshold transistors and analog molecular flux in chemical reactions obey Boltzmann exponential laws of thermodynamics and are described by astoundingly similar logarithmic electrochemical potentials. Therefore, cytomorphic circuits can help to map circuit designs between electronic and biochemical domains. We review recent work that uses positive-feedback linearization circuits to architect wide-dynamic-range logarithmic analog computation in *Escherichia coli* using three transcription factors, nearly two orders of magnitude more efficient in parts than prior digital implementations.

1. Introduction, motivations and overview

Every living cell within us is a hybrid analog–digital supercomputer that implements highly computationally intensive nonlinear, stochastic, differential equations with 30 000 gene–protein state variables that interact

Royal Society Publishing

via complex feedback loops. The average 10 µm human cell performs these amazing computations with 0.34 nm self-aligned nanoscale DNA–protein devices, with 20 kT per molecular operation (1 ATP molecule hydrolysed), approximately 0.8 pW of power consumption (10 M ATP s^{-1}) and with noisy, unreliable devices that collectively interact to perform reliable hybrid analog–digital computation [1]. Based on a single amino acid among thousands of proteins, immune cells must collectively decide whether a given molecule or molecular fragment is from a friend or foe, and if they err in their decision by even a tiny amount, autoimmune disease, infectious disease, or cancer could originate with high probability every day [2]. Even at the end of Moore's law, we will not match such performance by even a few orders of magnitude [1,3].

The field of synthetic biology attempts to transfer engineering design principles and experimental techniques into rational biological design [4–8]. It represents the ultimate limit of Moore's law: computation with the molecules themselves at the nanoscale through the use of controlled biochemistry and biophysics. Such an approach can blend 'fabrication' and 'computation' in a seamless fashion. The self-organizing amorphous soup in a cell processes information while it destroys, repairs and rebuilds the structures needed to do so. It is remarkable that it does so through a self-aligned nanotechnology with no explicit wiring. Instead, chemical binding among specific molecules serves to 'implicitly wire' them together and causes them to interact via chemical reactions. These reactions cause transformations of state, which are necessary for computation to occur.

While significant progress has been made w.r.t. fundamentals and applications in the field of synthetic biology, it has failed to scale significantly in complexity over more than a decade [9,10]. One important reason for this failure has been its overemphasis on digital paradigms of thought: because digital design is relatively straightforward and scalable, and because molecules and atoms are discrete, it is logical to assume that engineering biology as we engineer switches and logic gates today will bear fruit. The sheer force and powerful success of digital computation over the past few decades has been impressive. Nevertheless, we must not forget that digital computation has not offered an effective paradigm for computing efficiently and precisely with noisy and unreliable devices; that multi-logic-gate computations can impose significant metabolic or toxicity burdens on cells owing to their need to use a lot of parts and power; that the fact that there are five to six orders of magnitude fewer genes per cell than digital transistors per chip means that using genes to only perform logic is likely not an efficient way to attain high complexity; that a library of 'digital parts' with good on–off ratios, low standby power consumption and low crosstalk does not exist in biology; that the computing basis functions in cells are not really logic functions and abstracting them as such compromises computational efficiency; and that logic basis functions are not the only universal computation primitives.

Nature is not purely digital. While molecules are discrete and digital, all molecular interactions that lead to computation, e.g. association, transformation and dissociation chemical reactions, have a probabilistic analog nature to them. Depending on one's point of view, computation in a cell is owing to lots of probabilistic digital events or owing to continuous analog computation with noise. Both views are equivalent. Indeed, the noise in analog systems is related to the Poisson rate of the underlying probabilistic digital events; the shot noise of thermally generated diffusion currents caused by these Poisson processes generates noise in all analog systems [1,11].

Synthetic biology has now recognized that the signals in cells are stochastic (noisy) and analog (graded) in their nature [12]. The '1's and '0's of today's digital computers are useful abstractions of the analog signals in cells, but are often an oversimplification. Furthermore, as in analog circuits, the wiring of the output of one circuit to the input of another leads to 'loading' interactions that degrade overall function and prevents simple modular digital abstractions from being effective [13]. While logic basis functions and positive-feedback loops are certainly used by cells to make irreversible decisions, to organize sequential computation and to perform signal restoration, analog computation is extremely important for the cell's incredible efficiency w.r.t. the use of energy, time and space [1,14]. The efficiency arises because analog computation can use powerful input–output basis functions in the technology for addition, multiplication,

exponentials, logarithms, power laws and spatio-temporal filtering, which are already naturally present in the differential equations of physics and chemistry. Therefore, it does not need to re-invent these input–output basis functions from scratch with logic. For example, the production fluxes of two genes that encode the synthesis of a common output protein automatically perform addition via a molecular version of Kirchoff's current law without the need for tedious logic; the binding of two molecules provides a basis function for multiplication; the binding of identical molecules in a dimer provides basis functions for computing squares or square roots; molecular degradation naturally provides basis functions for temporal filtering; diffusion naturally provides a basis function for spatial filtering.

The pioneers of digital computation, John von Neumann and Alan Turing, appreciated the great power of analog computation and were investigating it intensely to cope with the limitations of using only logic to compute. Near the end of their lives, they were working on understanding analog computation in brains [15] and in cells [16], respectively. Analog computation has long been appreciated to be important in the brain [17], but its importance in cells has been greatly underappreciated. The power of the analog wave function in quantum mechanics enables quantum computers to solve problems that no digital computer can [18].

In this article, we begin by describing analog circuit schematics for a single gene, which provide a foundation for the rest of the article and a quick introduction to the computational basics of cell biology. Then, we review the pros and cons of analog versus digital computation with a focus on computation in living cells. In particular, we extend a prior analysis [1,14] to a simple exemplary computation in *Saccharomyces cerevisae* (yeast) cells, namely that of adding two numbers in an analog versus a digital fashion. This simple example will help us to analyse general trade-offs w.r.t. energy consumption or w.r.t. the number of molecules needed to implement addition at a given level of computational precision. The analysis will show that, below a certain crossover computational precision, it is highly advantageous to compute in an analog fashion to reduce the energy, part count or number of molecules (and thus volume or space) needed for the computation. It is therefore not surprising that cells exploit analog computation to perform their moderate-precision computations. Other work has also shown that *collective analog* computation can use several moderate-precision interacting analog units to architect computations of arbitrary precision, e.g. 16-bit addition via four interacting 4-bit-precise analog computational units [1,19], on an experimental silicon chip [20]. Collective analog systems can also architect arbitrary complex computations through interactions of moderate-precision analog computational units in the brain and in the cell. Indeed, cellular and neural computations bear 13 similarities to each other from a hybrid analog–digital point of view [1].

In analog computation, every signal and every device are not reliable, but important final or decisive outputs are. To attenuate noise, analog computation invariably relies on feedback loops, the wise use of energy, time or space resources to improve precision via averaging at critical state variables and reference inputs, learning and adaptation, nonlinearities, such as thresholding or digitization, or invariants and attractors in the analog dynamical system that cause it to equilibrate to restorative discrete outputs. Though noise limits information capacity, it can sometimes be beneficial for searching intractable combinatorial spaces and for improving the detection of signals by phenomena such as stochastic resonance.

How do we transfer analog circuits in electronics to analog circuits in cells? Fortunately, there is a deep connection between electronics and chemistry, which greatly aids the design of analog circuit motifs and analog computation in synthetic biology. This deep connection arises because there are astounding similarities between the equations that describe noisy electronic flow in subthreshold transistors and the equations that describe noisy molecular flow in chemical reactions, both of which obey the laws of exponential thermodynamics [1,21]. Therefore, circuit motifs from the electronic domain are useful for creating circuit motifs in biology and vice versa in a *cytomorphic* fashion, as figure 3 shows [1]. A recent paper [22] showed that log-domain analog circuits in living *Escherichia coli* bacterial cells, which were inspired by log-domain electronic circuits, could efficiently compute logarithms, add, subtract, divide and do square-root-like computations with less than three genetic parts. By contrast, a prior *in vitro* 2-bit-precise digital

square-root computation required 130 DNA-based parts [23]. Another circuit described in [22] was able to accurately compute the logarithmic ratio of two molecular concentrations over four orders of magnitude. The latter circuit used novel positive-feedback linearization circuits, similar to those used in my laboratory in log-domain subthreshold electronic amplifiers in the past [24]. The genetic circuits described in [22] may have wide applications for wide-dynamic-range molecular sensing, complex computation with few parts in biotechnology and medicine, and for the fine control of gene expression.

The analog circuits approach described in this article may enable large-scale design and analysis in synthetic and systems biology, which is faithful to how messy analog biology works, quite different from clean, well-defined digital design. It may enable several applications in synthetic biology, wherein, just as in electronics today, all applications benefit from low-part-count, low energy usage and clever analog feedback loops to improve performance. For example, synthetic biological circuits will be increasingly important in finding cures for diabetes, cancer, autoimmune, infectious and other diseases [25] wherein sophisticated synthetic feedback loops will have to compensate for error in natural feedback loops; synthetic molecular circuits are also important in biomolecule and pharmaceutical manufacturing [26] wherein efficient resource consumption and economic viability are closely tied; in energy production and environmental remediation [27,28] where efficient bacterial operation and fine control of gene expression is beneficial; in artificial fuel generation [29] where energy efficiency is paramount; in environmental pathogen or pollutant sensing where robust single-molecule analog amplification and wide-dynamic-range sensing is beneficial; in the design of 'nanorobots' and biomechanical devices [30] where analog feedback sensing and actuation loops could enable synthetic target-tracking behaviour. Analog synthetic biology can serve to make synthetic computation practical because it does not impose man-made views of what computation should be in the cell, but computes in a fashion that is similar to the way the cell itself computes.

Section 2 provides an analog circuit schematic of a gene that is useful for accurately representing steady-state, dynamic and stochastic phenomena. The success of analog circuit design arises in large part from having efficient pictorial representations that are more intuitive to humans than reams of differential equations. Intuition aids design. The analog circuit schematics nevertheless preserve most of the needed mathematical information in the equation as labelled parameters, such that analysis can aid design. Section 3 discusses the pros and cons of analog versus digital computation, first intuitively, then quantitatively. Section 4 discusses the deep similarities between electronics and chemistry, which can enable powerful cytomorphic circuit design in cells owing to the presence of a common logarithmic electrochemical potential. Section 5 provides examples of such circuit design using log-domain genetic circuits. Section 6 concludes with a summary of seven benefits of analog computation in synthetic biology.

2. Analog circuit schematic of a genetic promoter

Figure 1*a* shows an analog circuit schematic of a genetic circuit with two inputs: one activator, V_{act}, which enhances the rate of gene transcription (expression) and one repressor, V_{rep}, which suppresses the rate of gene transcription. The gene transcription initiates from a 'promoter region' on the DNA, $P_{\mathrm{act-rep}}$, near which these molecules bind and interact with the DNA reading machinery of the cell. The diamond-shaped dependent current generator, $P_{\mathrm{act-rep}}$, outputs an *mRNA* transcription current that is controlled by these 'transcription factor' inputs via an 'analogic' basis function with both linear analog and saturating digital regions of operation (the minimum and maximum expressions) [21]. The mRNA transcripts accumulate on the capacitor C (which always has $C = 1$ [1]) to create V_{mRNA}, the molecular concentration of mRNA.The resistor R_{mRNA} degrades the mRNA such that the synthesis current from the DNA promoter $P_{\mathrm{act-rep}}$ eventually balances the degradation current and V_{mRNA} equilibrates at a steady-state value. The transcription factor V_{act} is itself activated by an inducer input, $V_{\mathrm{ind}}^{\mathrm{act}}$, which binds to it via the 'multiplier' symbol, and activates it to bind DNA and stimulate transcription.

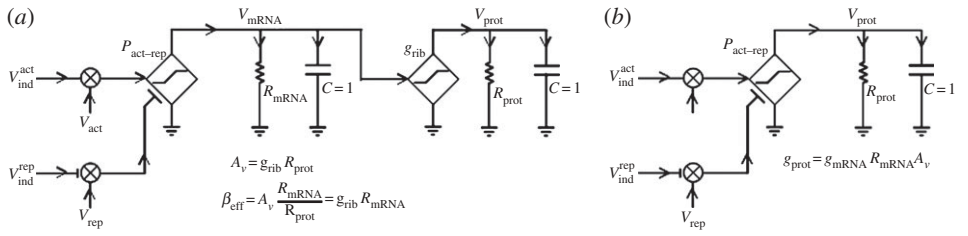

Figure 1. Analog schematic for genetic circuits. (*a*) An electrical equivalent circuit accurately represents the noise, dynamics and stochastics of DNA–protein circuits. (*b*) A simplification of the circuit in (*a*) that neglects mRNA dynamics.

Similarly, the transcription factor, V_{rep}, which normally binds DNA and represses transcription is de-repressed by V_{ind}^{rep}, such that the net effect of V_{ind}^{rep} is to also stimulate gene expression in this example. Repression is always indicated by the T-shaped symbols and activation is always indicated by arrow inputs at the border of the multiplier or at the border of the dependent current generator. Inducer inputs are typically small-molecule external inputs to the cell that diffuse into it and bind internal transcription factors within the cell. In bacteria and yeast, it is not common to have more than two controlling inputs at a DNA promoter, but mammalian cells can have several controlling inputs. The dependent generator will have as many inputs as the DNA promoter.

The dependent current generator on the right end of figure 1*a* 'translates' V_{mRNA} via ribosomal machinery in the cell to create a synthesis protein current, which accumulates on a capacitor to create V_{prot}, the molecular concentration. The resistor R_{prot} degrades the protein such that the synthesis translation current balances the protein degradation current and V_{prot} reaches a steady-state value. The equations in figure 1*a* define A_v and β_{eff}, which determine the gain from V_{mRNA} to V_{prot} and a noise factor similar to the current gain of a bipolar transistor, respectively [1]. We discuss the importance of β_{eff} for determining noise in §3. The dynamics and stochastics (noise) of the circuit of figure 1*a* faithfully capture experimental dynamic data [31] as well as noise data [32] provided that the shot noise of the molecular-dependent generators and resistors is accounted for correctly [1]: the resistor noise, unlike that in a normal resistor [11], is simply the shot noise of the net current flowing through it [1]. The current gain β_{eff} in figure 1*a* corresponds to a 'burst factor' that has been experimentally measured [32]. Compact 8-transistor analog circuits, which are current-mode circuit versions of figure 1*a*, create analogic basis functions that are mathematically exact representations of prior models [21]. They also match experimental input–output data gathered from non-pathogenic *E. coli* [21].

As R_{prot} is typically much greater than R_{mRNA}, it is common to assume that the mRNA dynamics are relatively instantaneous compared with protein dynamics [31]. Thus, the approximate circuit schematic of figure 1*b*, which is a simplified representation of figure 1*a* without m*RNA* dynamics and without explicit m*RNA* state variables, is significantly more convenient and usually accurate. It allows one to focus on a protein-input–protein-output point of view. In this article, our noise analysis in §3 is based on the accurate figure 1*a*; all other genetic circuit schematics are drawn similar to that of figure 1*b*.

3. Analog versus digital

We begin with an intuitive comparison of the pros and cons of analog versus digital computation presented in table 1.

Table 1 presents that analog computation emphasizes efficiency while digital computation emphasizes robustness. Robustness–efficiency trade-offs are ubiquitous in all engineering systems [33] and are at the heart of all good engineering design. A mathematical understanding of noise is essential for quantifying the intuition behind table 1. Therefore, we shall begin by deriving some important results regarding molecular noise in cells.

Table 1. Intuitive analog versus digital comparison.

analog	digital
(1) compute on a continuous set, e.g. [0,1], graded protein production from low to a maximum level	(1) compute on a discrete set, e.g. {0,1}, protein produced at a maximum level or not present at all
(2) the basis functions for computation arise from the physics and chemistry of the computing devices such that the amount of computation squeezed out of a single genetic, RNA or protein circuit is high	(2) the basis functions for computation arise from the mathematics of Boolean logic such that the amount of computation squeezed out of a single genetic, RNA or protein circuit is low
(3) one wire, channel or state variable represents many bits of information	(3) one wire, channel or state variable represents one bit of information
(4) computation is sensitive to the parameters of the molecular circuits	(4) computation is less sensitive to the parameters of the molecular circuits
(5) noise owing to thermal fluctuations in molecular devices	(5) noise owing to round off error and temporal aliasing
(6) signal is not restored at each stage of the computation	(6) signal is restored at each stage of the computation
(7) robust at final and decisive outputs	(7) robust in every device and signal

(a) Molecular noise in cells

There are a lot of experimental measurements and numbers available for *S. cerevisae*. Hence, we shall use it as our representative cell. For any gene, the Poisson shot noise of molecular flux that produces mRNA from genetic DNA and of molecular flux that produces protein from the mRNA is averaged over the bandwidth of an RC-like protein degradation filter to generate a mean protein copy number N_{prot} [1,31]. It also generates fluctuations around this mean with a mean square value of $\overline{n_{\mathrm{prot}}^2}$ [1,32]. In figure 1*a*, N_{prot} and $\overline{n_{\mathrm{prot}}^2}$ correspond to the mean value and fluctuations of V_{prot}, respectively.

Typically, mean mRNA copy numbers, N_{mRNA}, are low (0.1 to 130 in *S. cerevisae*) [34–37], while protein copy numbers are high (50–10^6 in *S. cerevisae*), and mRNA degradation time constants, τ_{mRNA}, are small (3–90 min in *S. cerevisae*) [35] while protein degradation time constants, τ_{prot}, are large (10–1000 min in *S. cerevisae*) [38]. Therefore, there is molecular gain from mRNA to protein, $A_v = N_{\mathrm{prot}}/N_{\mathrm{mRNA}}$ or $V_{\mathrm{prot}}/V_{\mathrm{mRNA}}$ in figure 1*a*, which amplifies the mRNA noise content in the protein signal. However, owing to the averaging caused by an effective bandwidth reduction factor of $\tau_{\mathrm{mRNA}}/\tau_{\mathrm{prot}}$, there is also a reduction in mRNA noise content in the protein signal in figure 1*a*. The total noise of the protein signal is owing to the net mRNA noise content as well as that owing to the intrinsic shot noise of the Poisson protein molecular flux. That is,

$$\left.\begin{aligned} \beta_{\mathrm{eff}} &= A_v \frac{\tau_{\mathrm{mRNA}}}{\tau_{\mathrm{prot}}} \\ \text{and} \qquad \overline{n_{\mathrm{prot}}^2} &= N_{\mathrm{prot}}(\beta_{\mathrm{eff}} + 1). \end{aligned}\right\} \tag{3.1}$$

The equations in figure 1*a* show circuit parameters for β_{eff} and A_{v}. These circuit parameters reproduce equation (3.1) exactly with $\tau_{\mathrm{mRNA}} = R_{\mathrm{mRNA}}C$, $\tau_{\mathrm{prot}} = R_{\mathrm{prot}}C$ and $V_{\mathrm{prot}} = N_{\mathrm{prot}}$. The output signal-to-noise ratio S_{N} is then given by

$$S_{\mathrm{N}} = \frac{N_{\mathrm{prot}}^2}{\overline{n_{\mathrm{prot}}^2}} = \frac{N_{\mathrm{prot}}}{\beta_{\mathrm{eff}} + 1}. \tag{3.2}$$

The equivalent informational bit precision [39] is given by

$$N_{\text{bits}} = \frac{1}{2}\log_2(1 + S_{\text{N}}). \tag{3.3}$$

Equations (3.2) and (3.3) reveal that, to encode an output signal at high precision (large S_{N} or large N_{bits}), the molecular protein copy number N_{prot} and consequently the mRNA copy number need to increase in proportion with S_{N}:

$$\boxed{\begin{aligned} N_{\text{prot}} &= (\beta_{\text{eff}} + 1)S_{\text{N}} \\ N_{\text{mRNA}} &= \frac{N_{\text{prot}}}{A_v}. \end{aligned}} \tag{3.4}$$

(b) Signal power consumption

Protein and mRNA molecules have to be constantly synthesized to counter their degradation, and this synthesis consumes power. The synthesis power is provided by the hydrolysis of several molecules of adenosine triphosphate (ATP), the universal energy-providing molecular currency of cells. ATP hydrolysis enables various processes in transcription and translation to move forward in a nearly irreversible fashion [2]. The hydrolysis of one molecule of ATP provides nearly 20 kT [40]. The energy cost of synthesizing an mRNA molecule in ATP is $E_{\text{mRNA}} \approx 30L_{\text{mRNA}}$ [37], where L_{mRNA} is the length of the mRNA molecule in nucleotides, with a median value of $L_{\text{mRNA}} = 1474$ nucleotides for yeast [34,36]. Similarly, the energy cost of synthesizing a protein molecule in ATP is $E_{\text{prot}} \approx 50L_{\text{prot}}$ [37], where L_{prot} is the length of the protein molecule in amino acids, with a median value of $L_{\text{prot}} = 385$ amino acids in yeast. Hence, the power consumption needed to charge or maintain a protein signal level of N_{prot} is given by

$$\left.\begin{aligned} P_{\text{prot}} &= \frac{E_{\text{mRNA}}N_{\text{mRNA}}}{\tau_{\text{mRNA}}} + \frac{E_{\text{prot}}N_{\text{prot}}}{\tau_{\text{prot}}}, \\ P_{\text{prot}} &= \frac{E_{\text{mRNA}}N_{\text{prot}}}{\tau_{\text{mRNA}}A_v} + \frac{E_{\text{prot}}N_{\text{prot}}}{\tau_{\text{prot}}}, \\ P_{\text{prot}} &= \frac{E_{\text{mRNA}}N_{\text{prot}}}{\tau_{\text{prot}}\beta_{\text{eff}}} + \frac{E_{\text{prot}}N_{\text{prot}}}{\tau_{\text{prot}}}, \\ \text{and} \quad \boxed{P_{\text{prot}} &= \tfrac{N_{\text{prot}}}{\tau_{\text{prot}}}\left(\tfrac{E_{\text{mRNA}}}{\beta_{\text{eff}}} + E_{\text{prot}}\right).} \end{aligned}\right\} \tag{3.5}$$

(c) Minimization of power consumption

If we substitute for N_{prot} in the boxed expression in equation (3.5) with its value in equation (3.4), we find another important relationship

$$\boxed{P_{\text{prot}} = \tfrac{S_{\text{N}}}{\tau_{\text{prot}}}(\beta_{\text{eff}} + 1)\left(\tfrac{E_{\text{mRNA}}}{\beta_{\text{eff}}} + E_{\text{prot}}\right).} \tag{3.6}$$

Equation (3.6) is a classic example of a *resource–precision* equation [1,14] that quantifies how power consumption (the resource) increases as the speed ($1/\tau_{\text{prot}}$) and the precision (S_{N}) of signals in a computation increase. Such resource–precision relationships are universal and can be found in neurobiological, electrical, mechanical and all systems [1]. Intuitively, to be fast and precise, a scenario that maximizes information, a system consumes power. Hence, universal information-based principles to reduce power consumption in any system from brains to electronic systems architect methods to reduce speed or precision at local state variables while maintaining system speed and precision [1,41]. For example, slow-and-parallel systems with moderate local precision lead to very power efficient systems, both in the brain and in electronics [1]. Equation (3.4) is another example of a resource–precision equation but the resource here is molecular copy number

rather than power consumption. In electronic systems, resource–precision equations relate power consumption and area consumption on chips to speed and precision [1,14].

Equation (3.6) suggests that, for a given precision (S_N) and speed ($1/\tau_{prot}$), there is an optimum value of β_{eff} that reduces power consumption. Simple differentiation reveals that this optimum occurs when

$$\beta_{eff}^{opt} = \sqrt{\frac{E_{mRNA}}{E_{prot}}}. \tag{3.7}$$

At this optimum, we can substitute for β_{eff} in equation (3.5) or in equation (3.6) to determine the power consumption

$$\left.\begin{aligned} P_{prot}^{min} &= \frac{N_{prot}}{\tau_{prot}}\sqrt{E_{prot}}\left(\sqrt{E_{mRNA}}+\sqrt{E_{prot}}\right) \\ \text{and} \quad P_{prot}^{min} &= \frac{S_N}{\tau_{prot}}\left(\sqrt{E_{mRNA}}+\sqrt{E_{prot}}\right)^2. \end{aligned}\right\} \tag{3.8}$$

From equation (3.1), for a given speed of operation, $1/\tau_{prot}$, the optimum β_{eff} can be attained either by varying the gain A_v if the time constants are fixed, or by varying τ_{mRNA} if A_v is fixed. If τ_{mRNA} is fixed, and $1/\tau_{prot}$ is increased to improve the speed of the computation, then A_v must be lowered to maintain β_{eff} at its optimal value.

(d) An analog molecular adder

To implement a molecular adder via analog computation is simple: we have the two inputs to be added; each regulates the expression of a common output protein from independent genetic promoters. During synthesis, the two molecular fluxes will automatically add to create the common output protein, which is the answer. To enable linear analog operation and avoid saturated digital operation, the input proteins must operate at a concentration that is well below K_d, the binding constant of the DNA promoter. A large K_d can be architected by using weak, mutated or multiple-binding-site promoters [22,31]. As the input and output concentrations increase, the input and output noise reduce, the precision of the computation improves, but at the price of higher molecular count and higher power consumption. The computation only requires two genetic parts, independent of precision or molecular count, which makes it practical to implement. To estimate the costs of this computation, we shall assume, for simplicity, that the input and output molecular counts are equal (promoters with a net gain of 1), which enable both the precision of the input and output to concomitantly scale with the overall needs of the computational precision. A proportionate gain scaling between the input and output is also possible but adds complexity while obscuring the key insights that we want to emphasize.

Analogous to how mRNA noise flux propagates to protein outputs and increases noise, the input noise flux propagates to the adder output. Thus, the adder will have twice the squared noise as would be expected from its output molecular count alone. Therefore, to preserve the S_N all molecular counts must be increased by a factor of 2 at the input and output, leading to a four times larger total molecular count for the same S_N. Thus, the fundamental resource–precision equations for one signal, equations (3.4) and (3.5), cause the resource–precision equations for the molecular count, N_A, and power consumption, P_A, of the analog adder to be

$$\left.\begin{aligned} N_A &= 4(\beta_{eff}+1)S_N \\ \text{and} \quad P_A &= \frac{N_A}{\tau_{prot}}\left(\frac{E_{mRNA}}{\beta_{eff}}+E_{prot}\right). \end{aligned}\right\} \tag{3.9}$$

Circuit noise-analysis techniques described in [1] can be used to derive resource–precision equations similar to equation (3.9) for arbitrarily complex analog computations.

(e) A digital molecular adder

An N-bit digital molecular adder requires 1 half-adder logic circuit (binary addition without carry for the least significant bit), $N-1$ full-adder circuits (binary addition with carry for the $N-1$ other bits) and two N-bit inputs. Appendix A presents that the total number of logic gates, each of whose outputs requires energy-consuming molecular synthesis, as well as $2N$ inputs, adds up to $17N-7$ net protein (and associated mRNA) state variables for $N \geq 2$. If $N=1$, six protein state variables are needed.

At high '1' levels, protein synthesis occurs at a high rate, and at low '0' levels, protein synthesis occurs at a lower rate. To minimize the power consumption, digital logic gates must have as small a protein copy number for high values as possible. At low values, the 'leakage' or 'basal transcription' in the genetic circuits will then limit practical implementations to 100-fold or 1000-fold lower protein copy numbers (N_{mRNA} varies from 0.1 to 130 in yeast and protein copy numbers then vary proportionately [34–37]). However, the '1' value must be sufficiently large such that it enables saturated operation, which is necessary for robust digital signal restoration in signal cascades, for a low bit error rate owing to sufficient noise margins between '1' and '0' values, for enabling robust fan-out that is not compromised by loading, and for robust operation in spite of K_{d} variations among gates. The parameter K_{d} is an equilibrium chemical binding constant that effectively behaves as a threshold voltage in the digital operation.

A reasonable design choice is to use a '1' value that is at least 10 times the K_{d} of the logic promoter such that multiple logic gates can function together in fan-out or cascade configurations with good noise margins and in spite of K_{d} variations among gates. Note that K_{d} variations among gates can be static or dynamic owing to loading and crosstalk. As in all engineering systems, a robustness–efficiency trade-off exists: a large '1' value will robustly guarantee saturation and signal restoration, good noise margins with low error rate, robust fan-out and robustness to K_{d} variations but will require higher power consumption to encode the '1'.

Measured K_{d}'s for signalling proteins are usually in the 30 nM–10 µM range [42], and some sources report about 10 nM [43]. We shall assume that 10 nM is practical for all of our logic gates, and that all operate at a '1' value that is $10\,K_{\mathrm{d}}$ and at a '0' value that is at least 200 times lower. As yeast is approximately 200 $\mu\mathrm{m}^3$ in volume [31,43], 200 molecules correspond to 1 nM and 2000 molecules correspond to 10 nM (near the median copy number of most proteins in yeast [34–37]). Thus, a '1' value corresponds to 20 000 molecules. If $N_{\mathrm{digHI}} = 20\,000$, then $N_{\mathrm{digLO}} = 20\,000/200 = 100$, which is about two times the lowest measured copy numbers in yeast [34–37]. The exact value of N_{digLO} does not matter too much as long as it is sufficiently low.

Many signalling proteins require at least dimerization to bind, which doubles molecular count and power consumption for the same K_{d}; but, we could reduce the '1' value to operate at $5K_{\mathrm{d}}$ owing to higher digital slopes, thus halving molecular count and power consumption. Thus, we shall, for simplicity, ignore dimerization and polymerization effects that are, to first order, neutral w.r.t. resource consumption in our analysis. For simplicity, we shall also assume that any logic gate operates at saturated values that are the same as that of its inputs such that logic gates can be easily composed and cascaded similar to that in today's electronic systems. Otherwise, digital implementations will require molecular gain and attenuation between logic stages. So, our assumption is generous to digital implementations.

Appendix A presents that the delay of the digital adder is approximately $(N+2)\tau_{\mathrm{prot}}$. Therefore, for the same speed as a parallel analog adder, each logic stage of the digital adder must decrease its degradation time constant, τ_{digN}, according to

$$\tau_{\mathrm{digN}} = \frac{\tau_{\mathrm{prot}}}{N+2}. \tag{3.10}$$

From equation (3.1), to operate at the optimal β_{eff} given by equation (3.7), the gain A_v can be adjusted to $A_{v\mathrm{digN}}$ according to

$$A_{v\mathrm{digN}} = \beta_{\mathrm{eff}} \frac{\tau_{\mathrm{digN}}}{\tau_{\mathrm{mRNA}}}. \tag{3.11}$$

Table 2. Parameters used for molecular addition.

parameter	value
τ_{mRNA}	3 min
τ_{prot}	1000 min
N_{digHI}	20 000
N_{digLO}	100
p_{hi}	0.5
p_{lo}	0.5
$\beta_{eff} = \beta_{eff}^{opt}$	2.49

If the probabilities of low and high values in the adder are p_{low} and p_{high}, respectively, with corresponding values of N_{digHI} and N_{digLO} for the protein state variables, then the resource–precision equations for the molecular count, N_D, and the power consumption, P_D, of the digital adder can be computed to be

and

$$\left.\begin{aligned} N_{dig} &= p_{hi}N_{digHI} + p_{lo}N_{digLO}, \\ N_D &= (17N - 7)N_{dig} \\ P_D &= \frac{N_D(N+2)}{\tau_{prot}}\left(\frac{E_{mRNA}}{\beta_{eff}} + E_{prot}\right). \end{aligned}\right\} \tag{3.12}$$

For most computations, $p_{hi} = p_{lo} = 0.5$ if we average across all possible inputs. Random additive inputs to our adder will generate equal probabilities of ones and zeros.

(f) Analog addition versus digital addition

The resource–precision equations for analog addition are described by equation (3.9), while those for digital addition are described by equation (3.12). To compare the costs of analog versus digital addition at the same speed ($1/\tau_{prot}$) and precision (S_N for analog and N for digital), we use equation (3.3) to convert S_N to an equivalent number of bits N_{bits}. With the analog and digital precision now in the same units (N for digital and N_{bits} for analog), we can compare analog and digital computation w.r.t. their resource consumption of molecules or power consumption. Table 2 lists a set of parameters for yeast used to make this comparison [34–38,44], which were chosen to minimize power consumption in the analog and digital domains.

Figure 2*a* reveals that the costs of analog power consumption in ATP s^{-1} are significantly less than those of digital power consumption even at nearly 10 bits of precision. At this crossover precision, the power consumption is a staggering 10 million ATP s^{-1}, so high that a yeast cell will be using almost its entire power budget to add two numbers at 10-bit precision! It is not surprising then that cells do not waste their energy doing highly accurate digital computation as computers do. The figure also implies that a synthetic biological circuit that just has 163 simple two-input deterministically robust logic gates is not practical. Thus, deterministic digital approaches to synthetic computation have barely scaled over a decade [9] and seem unlikely to in the future. Digital computation will likely have to be probabilistic and therefore behave more as noisy analog computation with a lower value of N_{digHI}. A lower value of N_{digHI} will necessarily make digital computation less robust and more probabilistic as we discuss in §3*g*. Probabilistic 'dithering' converts digital black-and-white pixels in an image to greyscale pixels. Thus probabilistic digital operation will have similarities to noisy analog operation, which is in fact noisy precisely because many probabilistic digital Poisson events constitute its value [1,11].

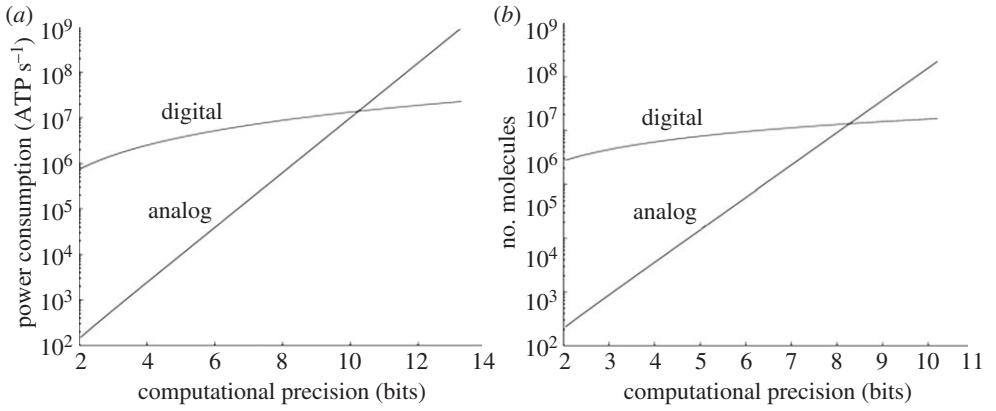

Figure 2. Analog versus digital computation in cells. (*a*) The figure shows the power costs for doing addition in cells with a genetic circuit. (*b*) The figure shows the molecular protein number required for doing addition in cells with a genetic circuit.

Figure 2*b* shows that the costs of analog addition in total protein molecular number are also significantly less than those of digital addition even at nearly 8 bits of precision. At this crossover precision, a staggering 1 million protein molecules are involved in addition. This number is about 1 in every 1000 protein molecules in yeast [31], which may be possible in the analog approach and harder in the digital approach (because 129 logic gates are needed). However, from [44], this 1/1000 change is about three orders of magnitude higher than the change needed to impact evolution in a yeast population. Therefore, the molecular-count costs of adding at 8-bit precision in a cell are still very expensive. At the 8-bit crossover in figure 2*b*, figure 2*a* shows that the corresponding analog power consumption is almost an order of magnitude less than digital power consumption.

High molecular copy numbers and high power consumption are characteristic of high-precision computation. Both lead to toxicity in the cell: high protein copy numbers can cause inadvertent binding of high-copy-number molecules to molecules in essential pathways and interfere with their function via competitive binding. High power consumption generates reactive oxygen species that lead to oxidative stress, cellular damage and cell death. So, high power consumption is not just an undesirable feature in cells as it is in some electronic systems: it actually leads to death. In some electronic systems, especially digital ones, high power consumption can lead to catastrophic failure of portions of a chip via thermal runaway effects.

(g) Analog versus digital

Do the lessons learned from the example of addition generalize to other computations? Are these lessons sensitive to our parameter values and assumptions? All computations have analog–digital crossover curves similar to those in figure 2*a*,*b* but the exact crossover precision depends on the computational task. Tasks that can be performed with fewer analog parts push the crossover point to higher precision [1,14]. Feedback loops in analog computation that correct for error push the crossover to higher precision [1]. Parameter changes can certainly shift the location of the crossover point and are different in different technologies—in electronics, in neurobiology and in cells [1,14].

As an example of how parameters can shift the analog–digital crossover point, let us assume that $N_{\mathrm{digHI}} = 2\,\mathrm{K}$ instead of 20 K such that digital power consumption is reduced by nearly 10 times. The analog–digital crossover point for power consumption will then be near 7 bits rather than 10 bits for addition, which is still very precise for a cell. In practice, there may be several problems associated with this lower value of N_{digHI}: with $N_{\mathrm{digHI}} = 2\,\mathrm{K}$, we require operation at $K_{\mathrm{d}} = 1\,\mathrm{nM}$ for all gates in the digital adder. The low K_{d} requires significant DNA–protein

engineering to improve chemical binding for all adder molecules, and also leads to a significant slowing of transcription-factor binding to DNA analogous to similar trade-offs seen in translation [45]. Higher noise caused by a lower N_{digHI} increases the bit error rate in each channel, making the overall digital computation more probabilistic. As the median protein copy number in yeast is nearly 2000 [34–37], the lower N_{digHI} may also lead to significant crosstalk owing to competitive binding with background molecules. As β_{eff} is not likely at its optimal value for all digital signal variables as we have assumed, each of them will be noisier, such that the lower N_{digHI} may significantly increase bit error rate. Finally, even with N_{digHI} at a 10 times lower value, nearly 10% of the cell's power budget would just be spent on 7-bit addition, compromising its other functions severely. At a 100 times lower value of $N_{\mathrm{digHI}} = 200$, digital operation will be highly probabilistic owing to insufficient noise margins and it would require significant engineering to reduce N_{digLO} to values near 20 to achieve on–off ratios of just 10: protein copy numbers below 50 are extremely rare in yeast [34–37]. So our choice of $N_{\mathrm{digHI}} = 20\,\mathrm{K}$ to get yeast cells to perform digital addition with 163 gates in a deterministic and robust digital fashion appears to be reasonable. By contrast, as the analog approach only requires two genes, it is significantly easier to engineer and optimize.

The fundamental reason for the analog–digital crossover does not lie in issues having to do with parameters and numbers. It really lies in information theory: information is coded across many 1-bit-precise interacting computational channels in the digital approach but on 1 multi-bit computational channel in the analog approach [14]. At low informational precision, logic basis functions simply cannot compete with the richer basis functions of analog computation that can process all the bits at once in parallel and just automatically solve the task, e.g. by using Kirchoff's current law for addition or chemical binding for multiplication. At high precision, however, the fundamental laws of large numbers and thermodynamics require the noise per computational channel to be too low in analog computation: the low-noise requirement drastically increases the costs of energy (more energy enables more molecules to be produced), time (more time allows for more noise reduction via averaging) or space (molecular count or parallel and redundant architectures cost volume) for analog computation. Digital computation wisely distributes the information and information processing over many channels that interact with each other to preserve information, e.g. in the case of addition via a carry. No one channel is required to perform heroically in digital computation. Thus, the polynomial scaling of resource costs with $N = N_{\mathrm{bits}}$ in digital computation, e.g. in equation (3.12) or equivalently its logarithmic scaling with S_{N} in equation (3.3), eventually always outcompete analog computation's polynomial scaling with S_{N}, e.g. in equation (3.9).

From analyses such as this one and others, we have concluded that the most efficient and scalable form of computation for arbitrary precision and complexity is a hybrid of analog and digital computation termed as *collective analog* computation [1,14,19,20]. In this form, several moderate-precision analog computational units interact to preserve and process information as many interacting analog neurons in the brain do. Certainly, one often uses digital signals for decision-making, signal restoration and communication within and among the analog units. For example, the first 16-bit (and scalably) precise analog adder in electronics works in this fashion on an experimental silicon chip [20]. Given current experimental data, it is likely that cells use many interacting 2- to 5-bit-precise analog computations to compute. Neurons also use distributed computation with moderate-precision analog components to reduce their significant energy costs [14,46,47].

Hybrid analog–digital architectures termed as *hybrid-state machines* (HSMs) [1] are useful for describing computations in cells in development or cell cycle pathways [2]. These machines provide a methodology for gene–protein networks to compute in a sequential analog fashion and generalize the notion of finite state machines to the hybrid analog–digital domain. HSMs also provide a good framework for describing spiking neuronal computation [19,20]. Indeed, there are 13 similarities between gene–protein and neural computation from a hybrid analog–digital point of view [1], and *in vitro* molecular computation has attempted to engineer neural-network-like computations [48].

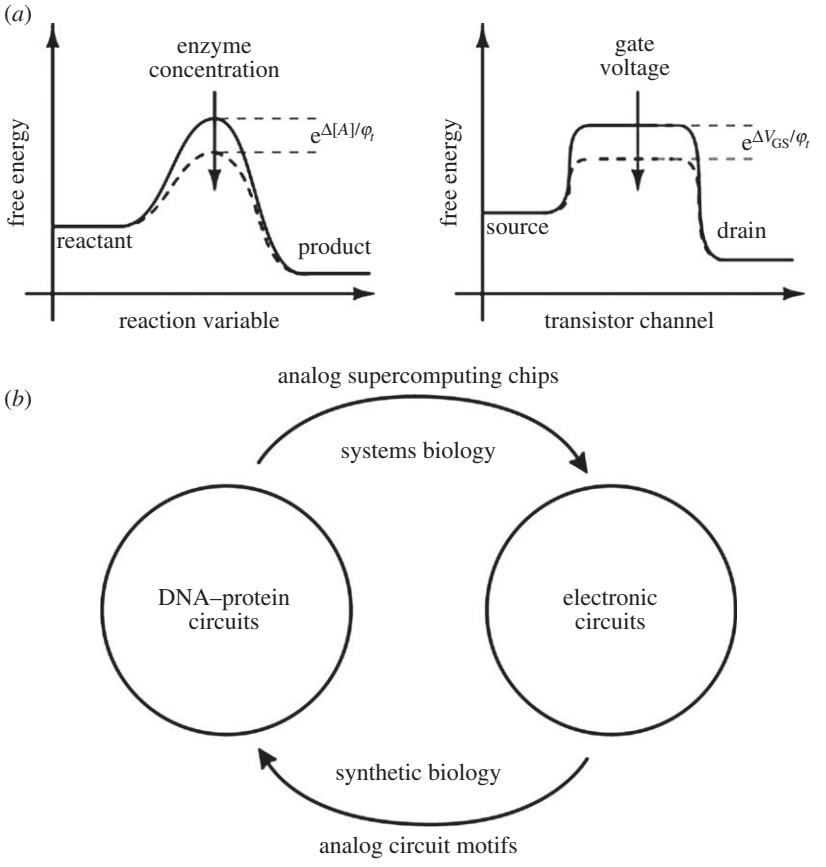

Figure 3. The cytomorphic mapping. (*a*) The figure shows the deep connections between electronic flow in transistors and molecular flow in chemical reactions caused by their obeying the same laws of thermodynamics. (*b*) The figure shows that this similarity enables circuits in both domains to be mapped to each other.

4. Deep connections between electronics and chemistry

There are striking similarities between chemical-reaction dynamics (figure 3*a*) and electronic current flow in the subthreshold regime of transistor operation (figure 3*b*): electron concentration at the source is analogous to reactant concentration; electron concentration at the drain is analogous to product concentration; forward and reverse current flows in the transistor are analogous to forward and reverse reaction rates in a chemical reaction; the forward and reverse currents in a transistor are exponential in voltage differences at its terminals analogous to reaction rates being exponential in the free-energy differences in a chemical reaction; increases in gate voltage lower energy barriers in a transistor increasing current flow analogous to the effects of enzymes or catalysts in chemical reactions that increase reaction rates; and the stochastics of the Poisson shot noise in subthreshold transistors are analogous to the stochastics of molecular shot noise in reactions. Some of these analogies have been listed in the table 3.

The logarithmic dependence of the electrochemical potential in chemical concentration or of current enables one to map log-domain analog transistor circuit motifs in electronics to log-domain analog molecular circuit motifs in cells and vice versa.

If we examine the classic Michaelis–Menten enzyme–substrate binding basis function, we find that

$$\frac{(x/K_{\mathrm{d}})}{1+(x/K_{\mathrm{d}})} = \frac{1}{1+\mathrm{e}^{-\ln(x/K_{\mathrm{d}})}}. \tag{4.1}$$

Table 3. Chemistry and electronics.

chemical-reaction dynamics	electron flow in transistor
reactant concentration	electron concentration at the source
product concentration	electron concentration at the drain
forward and reverse reaction rates in chemical reaction	forward and reverse current flows in the transistor
forward and reverse chemical reaction rates are exponential in the free-energy difference between transition state and reactant/product, respectively	forward and reverse electronic currents are exponential in the voltage difference between transistor channel and source/drain, respectively
enzymes or catalysts in chemical reactions increase reaction rates	increases in gate voltage lower energy barriers in a transistor, increasing current flow
stochastics of molecular Poisson processes in chemical reactions [1,11]	stochastics of electronic Poisson processes in subthreshold transistors [1,11]
flux balance analysis	Kirchoff's current law
chemical energy conservation	Kirchoff's voltage law
chemical concentration	current
electrochemical potential: log(concentration) + energy	electrochemical potential: log(current) + voltage

Therefore, if we look at the left-hand side, the basis function may be viewed as being approximately linear and proportional to (x/K_d) for $x < K_d$ and saturating after $x = 10\,K_d$; or, if we look at the right-hand side, it may be viewed as being log-linear over the range $0.1K_d < x < 10K_d$ where the sigmoid is linear. Hence, this basis function has a log-linear analog regime of operation over the central portion of the sigmoid and a saturating digital regime at the extremes of the sigmoid. Note that $\ln(K_d)$ is proportional to the free energy of the binding reaction. Highly compact eight-transistor log-domain analog differential-pair circuits generate almost exactly identical mathematical basis functions in either the linear current [21] (left-hand side of equation (4.1)) or log-linear voltage domain [1] (right-hand side of (4.1)). Log-domain analog circuits can generate any polynomially linear or polynomially nonlinear dynamical system in both electronics and in chemistry [1]. In particular, they can generate dynamical systems of the form

$$\left.\begin{aligned} \frac{d\mathbf{x}}{dt} &= \mathbf{C} + \mathbf{D}\mathbf{x} + \mathbf{E}(\mathbf{x} \otimes \mathbf{x}) + \mathbf{F}\mathbf{u} + \mathbf{G}(\mathbf{x} \otimes \mathbf{u}) \\ \text{and} \qquad \mathbf{y} &= \mathbf{H}\mathbf{x} + \mathbf{K}\mathbf{u}, \end{aligned}\right\} \tag{4.2}$$

where $\mathbf{x}$ and $\mathbf{u}$ are vector state and input variables, respectively, the matrix coefficients are determined by chemical-reaction parameters, and the multiply operations refer to outer products derived from two-molecule chemical binding. The limitation to two-molecule binding is for practicality and leads to no loss of generality [1]: two molecules can bind to generate a complex, and the complex can then bind to the third as is true in most chemical reactions.

Figure 3*b* shows that the *cytomorphic* mapping between electronics and chemistry outlined in figure 3*a* and table 3 enables one to map from electronic circuits to DNA–protein circuits and vice versa. In this article, we focus on the mapping from electronic circuits to molecular circuits, which is most useful in synthetic biology. The other direction of mapping has been extensively discussed in [1] and is useful for the design and simulation of synthetic biological circuits or the ultrafast stochastic simulation of large-scale systems-biology circuits with supercomputing chips.

5. Logarithmic analog computation in living cells

To widen the dynamic range of input operation of an inducer, it is useful to have a wide log-linear range. Logarithmic transduction affords advantages such as constant-precision sensing at

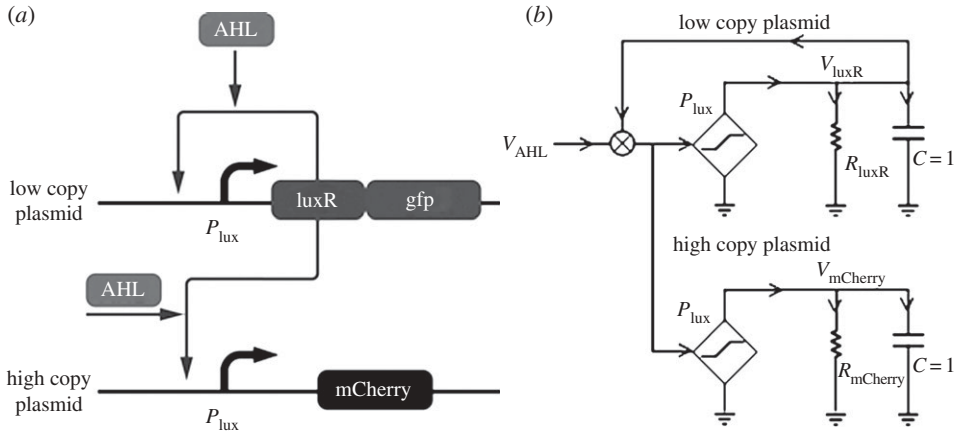

Figure 4. A positive-feedback linearization circuit. (*a*) A genetic circuit that linearizes inducer operation over a wide log-linear dynamic range is shown. (*b*) An analog circuit schematic represents this circuit.

any intensity (Weber's law) and is seen in many natural systems, including audition, vision and in cells [49].

If the concentration of a transcription factor is fixed, as the inducer increases in value, it will eventually be bound to all the available transcription-factor molecules and saturate the number of bound inducer–transcription-factor complexes that are possible. In addition, if the number of DNA-binding sites for a complex is limited, these sites will eventually all be bound by complexes and gene expression will saturate. These two sources of saturation limit the dynamic range of inducer operation. Figure 4*a* shows a genetic circuit motif [22] and an associated circuit schematic in figure 4*b* that simultaneously alleviates both these saturation problems to widen the dynamic range.

In figure 4*a*, a positive-feedback loop increases the generation of transcription factor by catalysing its own production on a low-copy plasmid. As the amount of inducer increases, more transcription factors are created alleviating the saturation of complexes. The high-copy plasmid has several binding sites that also shunt away the complexes, thus alleviating the saturation of DNA-binding sites on the low-copy plasmid. The high-copy plasmid also serves to control the gain of the positive-feedback loop by altering the number of complexes available to the low-copy plasmid [22]. Such control exploits 'the fan-out loading' of the downstream high-copy plasmid on the upstream low-copy plasmid as a desirable feature in our analog circuit. Fan-out is often a problem in molecular digital circuits. Figure 5*a* shows that the log-linear dynamic range is almost four orders of magnitude.

The genetic circuit of figure 4*a* actually exploits an idea from log-domain analog circuit design shown in figure 5*b*: expansive *sinh*-based linearization of compressive and saturating *tanh*-based differential-pair circuits causes them to widen their linear range of operation [24]. The function of the *sinh* is achieved by positive feedback in figure 4*a* and the function of the *tanh* is analogous to that of the saturation of biochemical binding. Indeed, in the future, other linearization circuit motifs from log-domain circuit design, e.g. those in [1], could also be exported to cells. The supplementary section of [22] models details how, analogous to the case in electronics [24], the widest dynamic range in the circuit motif of figure 4*a* is achieved at an optimal value of positive feedback.

Figure 6*a* and its associated schematic in figure 6*b* show how to use logarithms to divide by using $\log(A) - \log(B) = \log(A/B)$: two linearized positive-feedback circuit motifs enable two inducers, Arab and AHL, to effectively control the expression of the mCherry fluorescent protein on the high-copy plasmid. The Arab input controls mCherry by *activating* its production. The AHL input generates LacI on the low-copy plasmid, which then *represses* mCherry production. The IPTG inducer fine-tunes this repression gain. The RBS1 and RBS2 translation gains (analogous

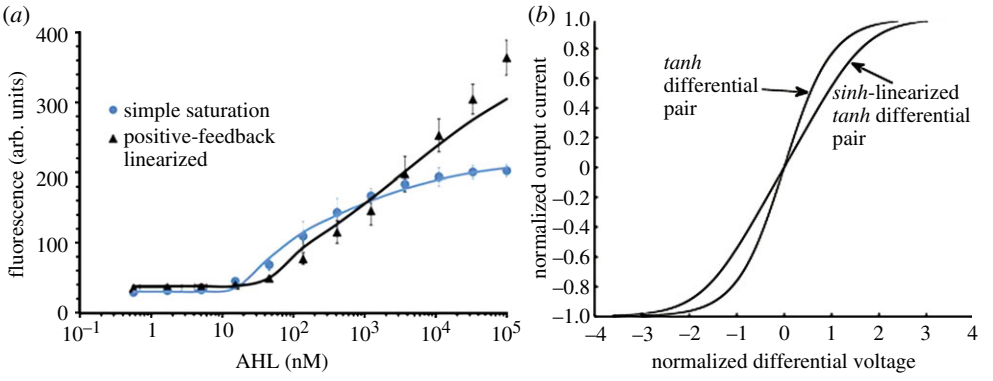

Figure 5. Linearization data. (*a*) The effect of linearization on the biological input is clearly seen. (*b*) A *sinh-linearized tanh* differential-pair circuit in electronics bears similarity to the genetic circuit motif. It also architects linearization by having expansive and compressive (saturating) nonlinearities interact. (Online version in colour.)

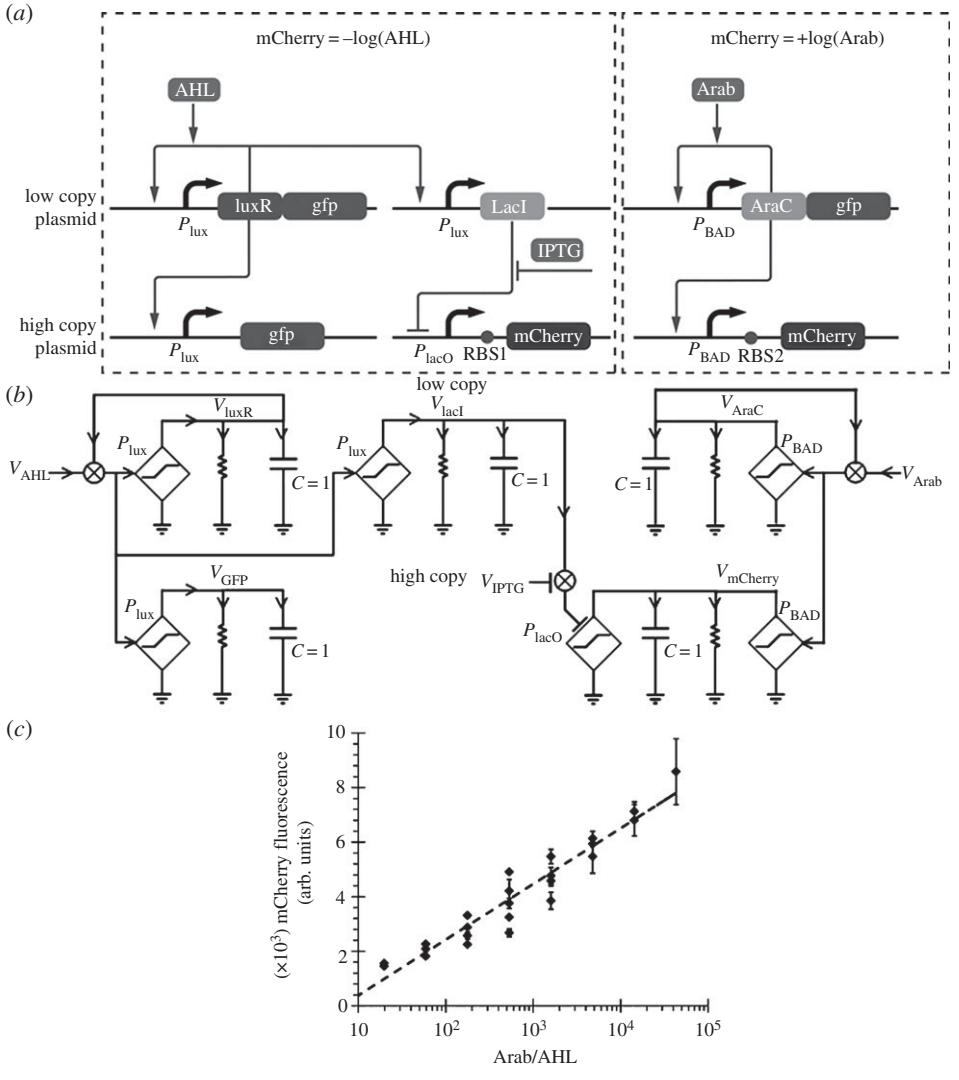

Figure 6. A pRATIO circuit. (*a*) The genetic circuit computes the logarithmic ratio of its two molecular inputs. (*b*) An associated electrical schematic equivalent is shown. (*c*) The ratio is computed over four orders of magnitude in the genetic circuit.

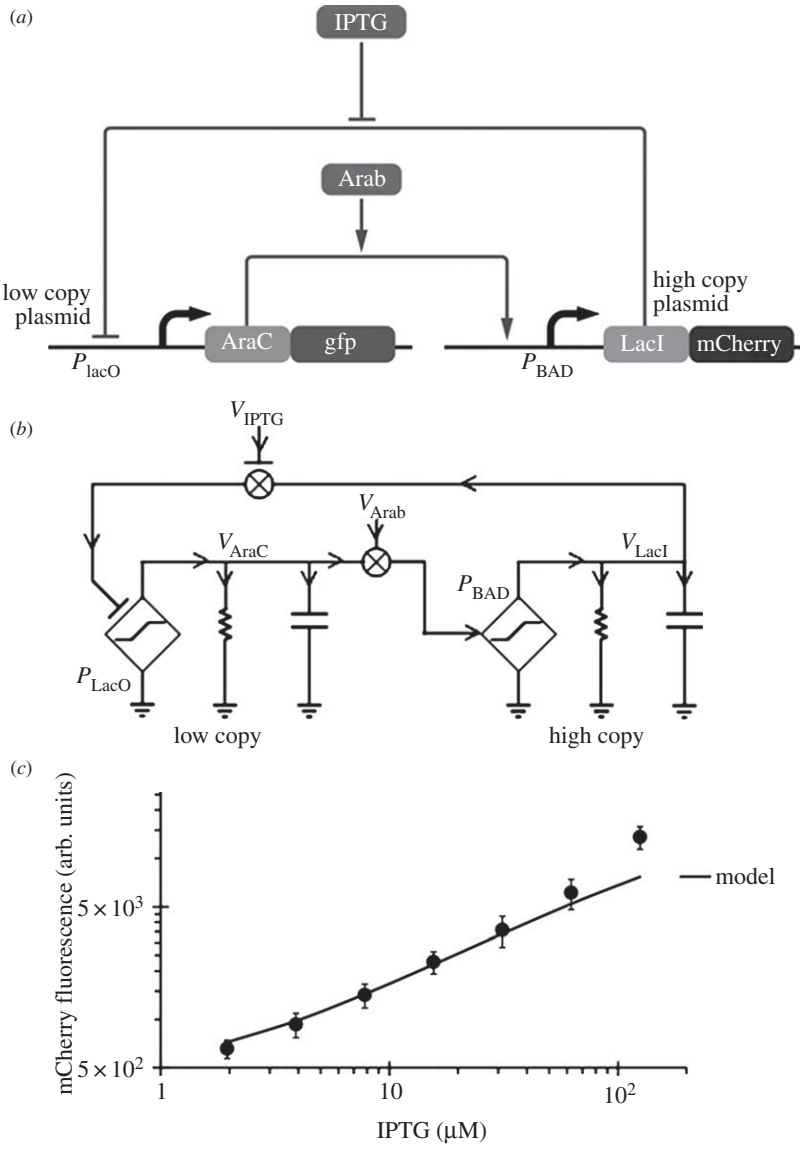

Figure 7. A power-law circuit. (*a*) A genetic power-law circuit. (*b*) An associated electrical equivalent schematic is shown. (*c*) The experimental data for the genetic circuit.

to g_{rib} in figure 1*a*) also serve to fine-tune the gain of the positive logarithm w.r.t. the gain of the negative logarithm. The net result is that the logarithmic ratio of the two inducer molecules in figure 6*c* is obtained over almost four orders of magnitude. This 'pRATIO' circuit generalizes the concept of pH, a logarithmic ratio used to measure H^+ concentration w.r.t. a reference, to any arbitrary ratio of two inputs w.r.t. one another. It may have applications for the wide-dynamic-range sensing of biomolecules by serving as a log differential amplifier.

In log-linear systems, the computational basis functions needed for universality besides the logarithm itself are addition, subtraction and scaling. Daniel *et al.* [22] shows how to architect logarithmic addition by merely summing up fluxes as in our analog adder. We have already shown how to subtract two logarithmic inputs in figure 6*a*,*b*. Scaling is also implicit in figure 6*a*,*b* (via the IPTG, RBS1 and RBS2 gains). The equivalent of addition, subtraction and scaling with logarithms corresponds to multiplication, division and power laws in the linear domain, much as in the operation of slide rules.

Figure 7*a* and its associated circuit schematic in figure 7*b* show how to compute a power law without any explicit use of logarithms: the high-copy plasmid produces a high level of LacI that strongly represses AraC production with $\mathrm{AraC} = g_{\mathrm{AraC}}(K_{\mathrm{df}}/\mathrm{LacI})(\mathrm{IPTG}/K_m)^h$; IPTG controls the strength of its de-repression according to the inherent power law h, and g_{AraC} is the gain of the first-stage amplifier in figure 7*b*. As the low-copy production of AraC does not saturate the binding sites on the high-copy plasmid, the falling AraC reduces the level of LacI in a relatively linear fashion according to $\mathrm{LacI} = g_{\mathrm{LacI}}\,(\mathrm{AraC}/K_{\mathrm{d}})$, with g_{LacI} being the gain of the second-stage amplifier of figure 7*b*. The net consequence of the negative feedback is then that LacI must equilibrate at a consistent value, where

$$\left.\begin{aligned} \mathrm{LacI} &= g_{\mathrm{LacI}}\frac{g_{\mathrm{AraC}}(K_{\mathrm{df}}/\mathrm{LacI})(\mathrm{IPTG}/K_m)^h}{K_{\mathrm{d}}} \\ \text{and}\quad \mathrm{LacI} &= \sqrt{g_{\mathrm{LacI}}g_{\mathrm{AraC}}\left(\frac{K_{\mathrm{df}}}{K_{\mathrm{d}}}\right)\left(\frac{\mathrm{IPTG}}{K_m}\right)^h}. \end{aligned}\right\} \tag{5.1}$$

The net power law w.r.t. IPTG is then $h/2$, which in figure 7*c* is found to be about 0.7. The supplementary section in [22] provides further details. This power-law circuit only requires two transcription factors to implement, in contrast to an *in vitro* circuit, which required 130 DNA parts to implement a square-root circuit with 2 bits of output precision [23].

6. Summary

We summarize by reviewing seven benefits of analog computation in cells, which are important for synthetic (and systems) biology.

(1) *Digital computation is a subset of analog computation* that only operates at its saturated high or low extremes. Therefore, by allowing operation over the whole range of signal levels and by allowing exploitation of all the basis functions of biochemistry and biophysics, not just logic, *horizons for computation are expanded*. Digital computation will continue to be important for decision-making, signal restoration, communication and sequential operation.

(2) *At the moderate precision of computation seen in cells, analog computation is significantly more efficient in its use of energy, time and space* (molecular count and part count) than digital computation. Therefore, analog and collective analog computation is *likely to be more practical and scalable for synthetic biology* than purely digital computation.

(3) *Probabilistic digital computation operates with analog probabilities, and behaves as noisy analog computation*. As even digital computation in cells may be forced to be probabilistic to cope with constraints on part count, molecular count and energy, analog computation may be inevitable.

(4) *Problems such as loading and fan-out*, which are limiting the scaling of synthetic biology, *are part of the design process of analog computation*, which uses concepts of input and output impedance and feedback design to cope with such problems. It can even exploit them for computation (figure 4*a*,*b*).

(5) *Analog design*, which has been developed over several decades, *provides accurate pictorial motifs with abstraction* (figure 1*b*) unlike pure differential-equation design, but these abstractions *are not oversimplified* as in molecular logic design.

(6) The great similarities between neural and cellular computation suggest that *analog computation in the cell may have predated and pioneered analog computation in the brain*.

(7) The *deep cytomorphic connection between electronics and chemistry* via the common electrochemical potential *can enable log-domain analog electronic circuits to inspire new analog circuit motifs in cells* and vice versa. Therefore, future work in synthetic biology or in systems biology will be likely to benefit from the cytomorphic mapping (figure 3*a*,*b*).

Acknowledgements. I thank R. Daniel for providing several figures pertinent to §5, which were modified and adapted for this article. I thank M. Sarpeshkar for drawing all of the circuit schematics in this article. Electronic supplementary material relevant to this publication can be found at http://www.rle.mit.edu/acbs/.

Funding statement. This work was financially supported in part by grants from the National Science Foundation under CCF 1124247 and CCF 1348519 and by a grant from the Semiconductor Research Corporation under Proposal P16730.

Appendix A. Costs of *N*-bit digital molecular addition

An N-bit digital adder uses one 'half-adder' logic circuit (addition of two binary numbers A_0 and B_0 with no carry) at its least significant bit (LSB) and $N-1$ 'full-adder' logic circuits (addition of two binary numbers A_n, B_n, and the carry C_{n-1} from the previous bit) at all other $N-1$ bits. This architecture outputs local binary answers at each bit stage and also propagates carries to neighbouring bit stages. The carries propagate such that the most significant bit (the 'MSB' stage at bit $N-1$ with LSB being at bit 0) is the very last stage to output its answer. This scheme of addition is really just a binary version of classic decimal addition. More complex schemes, for example Brent–Kung adders, can reduce carry propagation delays but are too complex in a molecular context and do not alter any of the conclusions of this paper. Therefore, we shall only consider simple addition of two positive numbers via the classic method of addition described above. We shall also not delve into two's complement arithmetic because we shall assume that all numbers are positive.

The half-adder stage implements the following logic for the local sum S_0 and carry C_0:

$$S_0 = A_0\overline{B}_0 + B_0\overline{A}_0$$

and

$$C_0 = A_0B_0.$$

These logic functions can be implemented with two genetic promoter circuits that each implement a half-sided XOR (such circuits are common as a repressor and an activator often interact in genetic promoter circuits), one genetic OR promoter circuit, and one genetic AND promoter circuit. All of these two-input genetic circuits have been reported in the literature and are possible to synthesize [31]. We shall refer to the half-sided XOR circuit as an H-XOR. Three-input genetic circuits have thus far not been robust or easy to synthesize, so we shall not use them.

The full-adder stage at the nth bit implements the following logic for the local sum S_n and carry C_n:

$$S_n = (A_n\overline{B}_n)\overline{C}_{n-1} + (B_n\overline{A}_n)\overline{C}_{n-1} + (C_{n-1}\overline{A}_n)\overline{B}_n + (A_nB_n)C_{n-1}$$

and

$$C_n = A_nB_n + (A_n + B_n)C_{n-1}.$$

Thus, it requires 3×2 H-XORs that need to be composed in cascade (for S_n), four ANDs (two cascaded for S_n and two for C_n) and five ORs. In our scheme of architecting the digital-adder circuit, no signal inversions are necessary, which saves gates. However, it does assume a combinatorial set of promoters that respond to inverted and non-inverted logical signals as needed. Such promoters cost no power and add no molecular count in our accounting. Thus, we are likely to be underestimating digital resource consumption compared with an actual implementation.

Hence, if N is at least 2 (the 1-bit case is degenerate because there are no full-adder circuits) the net number of gates, each of whose molecular outputs leads to an energy-consuming synthesis operation, and the net number of molecular inputs, which also lead to energy-consuming synthesis operations, can be computed from table 4.

Table 4. Resource consumption in an *N*-bit digital adder.

LSB sum and carry gates	$(N-1)$ MSB sum gates	N MSB carry gates	N-bit inputs	gate	total
2	$6(N-1)$	0	0	H-XOR	$6N-4$
1	$2(N-1)$	$2N$	0	AND	$4N-1$
1	$3(N-1)$	$2N$	0	OR	$5N-2$
0	0	0	$2N$	input (no gate)	$2N$
				total molecules (no. of gate outputs plus inputs)	$17N-7$

As the final MSB output requires N stages of carry propagation, and the local sum and carry generation stages incur two logic-gate delays, the worst-case delay of the adder is approximately $(N+2)\tau_{\mathrm{prot}}$, where τ_{prot} is protein degradation time constant of any logic gate.

For simplicity, we have assumed that there is no crosstalk between gates, which for a large number of gates may be difficult to architect in practice.

References

1. Sarpeshkar R. 2010 *Ultra low power bioelectronics: fundamentals, biomedical applications, and bio-inspired systems*. Cambridge, UK: Cambridge University Press.
2. Lodish HF. 2008 *Molecular cell biology*, 6th edn. New York, NY: W.H. Freeman.
3. Zhirnov VV, Cavin RK. 2013 Future microsystems for information processing: limits and lessons from the living systems. *IEEE J. Electron Devices Soc.* **1**, 29–47. (doi:10.1109/JEDS.2013.2258631)
4. Chen YY, Galloway KE, Smolke CD. 2012 Synthetic biology: advancing biological frontiers by building synthetic systems. *Genome Biol.* **13**, 240. (doi:10.1186/gb-2012-13-2-240)
5. Baker D, Church G, Collins J, Endy D, Jacobson J, Keasling J, Modrich P, Smolke C, Weiss R. 2006 Engineering life: building a FAB for biology. *Sci. Am.* **294**, 44–51. (doi:10.1038/scientificamerican0606-44)
6. Elowitz MB, Leibler S. 2000 A synthetic oscillatory network of transcriptional regulators. *Nature* **403**, 335–338. (doi:10.1038/35002125)
7. Auslander S, Auslander D, Muller M, Wieland M, Fussenegger M. 2012 Programmable single-cell mammalian biocomputers. *Nature* **487**, 123–127. (doi:10.1038/nature11149)
8. McMillen D, Kopell N, Hasty J, Collins JJ. 2002 Synchronizing genetic relaxation oscillators by intercell signaling. *Proc. Natl Acad. Sci. USA* **99**, 679–684. (doi:10.1073/pnas.022642299)
9. Purnick PEM, Weiss R. 2009 The second wave of synthetic biology. *Nat. Rev. Mol. Cell Biol.* **10**, 410–422. (doi:10.1038/nrm2698)
10. Cardinale S, Arkin AP. 2012 Contextualizing context for synthetic biology—identifying causes of failure of synthetic biological systems. *Biotechnol. J.* **7**, 856–866. (doi:10.1002/biot.201200085)
11. Sarpeshkar R, Delbruck T, Mead CA. 1993 White noise in MOS transistors and resistors. *IEEE Circuits Devices* **9**, 23–29. (doi:10.1109/101.261888)
12. Sauro HM, Kim K. 2013 Synthetic biology: it's an analog world. *Nature* **497**, 572–573. (doi:10.1038/nature12246)
13. Del Vecchio D, Ninfa AJ, Sontag ED. 2008 Modular cell biology: retroactivity and insulation. *Mol. Syst. Biol.* **4**, 161. (doi:10.1038/msb4100204)
14. Sarpeshkar R. 1998 Analog versus digital: extrapolating from electronics to neurobiology. *Neural Comput.* **10**, 1601–1638. (doi:10.1162/089976698300017052)
15. Von Neumann J. 1958 *The computer and the brain*. New Haven, CT: Yale University Press.
16. Turing AM. 1990 The chemical basis of morphogenesis. *Bull. Math. Biol.* **52**, 153–197. (doi:10.1007/BF02459572)
17. Tank DW, Hopfield JJ. 1987 Collective computation in neuronlike circuits. *Sci. Am.* **257**, 104–114. (doi:10.1038/scientificamerican1287-104)

18. Nielsen MA, Chuang I. 2011 *Quantum computation and quantum information*, 10th edn. Cambridge, UK: Cambridge University Press.
19. Halloran MO, Sarpeshkar R. 2002 Scalable hybrid computation with spikes. *Neural Comput.* **14**, 2003–2038. (doi:10.1162/089976602320263971)
20. Woo SS, Sarpeshkar R. 2013 A spiking-neuron collective analog adder with scalable precision. In *Proc. 2013 IEEE Symp. on Circuits and Systems* (*ISCAS 2013*), pp. 1620–1623. (doi:10.1109/ISCAS.2013.6572172)
21. Daniel R, Woo S, Turicchia L, Sarpeshkar R. 2011 Analog transistor models of bacterial genetic circuits. In *Proc. 2011 IEEE Biomedical Circuits and Systems Conf.*, pp. 333–336. (doi:10.1109/BioCAS.2011.6107795)
22. Daniel R, Rubens JR, Sarpeshkar R, Lu TK. 2013 Synthetic analog computation in living cells. *Nature* **497**, 619–623. (doi:10.1038/nature12148)
23. Qian L, Winfree E. 2011 Scaling up digital circuit computation with DNA strand displacement cascades. *Science* **332**, 1196–1201. (doi:10.1126/science.1200520)
24. Tavakoli M, Sarpeshkar R. 2005 A sinh resistor circuit and its application to tanh linearization. *IEEE J. Solid State Circuits* **40**, 536–543. (doi:10.1109/JSSC.2004.841015)
25. Ruder WC, Lu T, Collins JJ. 2011 Synthetic biology moving into the clinic. *Science* **333**, 1248–1252. (doi:10.1126/science.1206843)
26. Withers ST, Shiba Y, Sarpong R, Keasling JD. 2006 Production of the antimalarial drug precursor artemisinic acid in engineered yeast. *Nature* **440**, 940–943. (doi:10.1038/nature04640)
27. Lovley DR. 2012 Electromicrobiology. *Annu. Rev. Microbiol.* **66**, 391–409. (doi:10.1146/annurev-micro-092611-150104)
28. Logan BE. 2009 Exoelectrogenic bacteria that power microbial fuel cells. *Nat. Rev. Microbiol.* **7**, 375–381. (doi:10.1038/nrmicro2113)
29. Xu L, Brilman DWF, Withag JAM, Brem G, Kersten S. 2011 Assessment of a dry and a wet route for the production of biofuels from microalgae: energy balance analysis. *Bioresour. Technol.* **102**, 5113–5122. (doi:10.1016/j.biortech.2011.01.066)
30. Martel S. 2012 Bacterial microsystems and microrobots. *Biomed. Microdevices* **14**, 1033–1045. (doi:10.1007/s10544-012-9696-x)
31. Uri A. 2007 *An introduction to systems biology: design principles of biological circuits*. Boca Raton, FL: Chapman & Hall/CRC.
32. Ozbudak EM, Thattai M, Kurtser I, Grossman AD, van Oudenaarden A. 2002 Regulation of noise in the expression of a single gene. *Nat. Genet.* **31**, 69–73. (doi:10.1038/ng869)
33. Csete ME, Doyle JC. 2002 Reverse engineering of biological complexity. *Science* **295**, 1664–1669. (doi:10.1126/science.1069981)
34. Arava Y *et al.* 2003 Genome wide analysis of mRNA translation profiles in *Saccharomyces cerevisae*. *Proc. Natl Acad. Sci. USA* **100**, 3889–3894. (doi:10.1073/pnas.0635171100)
35. Wang Y, Liu CL, Storey JD, Tibshirani RJ, Herschlag D, Brown PO. 2002 Precision and functional specificity in mRNA decay. *Proc. Natl Acad. Sci. USA* **99**, 5860–5865. (doi:10.1073/pnas.092538799)
36. Ghaemmaghami S *et al.* 2003 Global analysis of protein expression in yeast. *Nature* **425**, 737–741. (doi:10.1038/nature02046)
37. Wagner A. 2005 Energy constraints on the evolution of gene expression. *Mol. Biol. Evol.* **22**, 1365–1374. (doi:10.1093/molbev/msi126)
38. Belle A, Tanay A, Bitincka L, Shamir R, O'Shea EK. 2006 Quantification of protein half lives in the yeast proteome. *Proc. Natl Acad. Sci. USA* **103**, 13 004–13 009. (doi:10.1073/pnas.0605420103)
39. Shannon CE, Weaver W. 1949 *The mathematical theory of communication*. Urbana, IL: University of Illinois Press.
40. Berg JM, Tymoczko JL, Stryer L. 2006 *Biochemistry*, 6th edn. New York, NY: Freeman.
41. Sarpeshkar R. 2012 Universal principles for ultra low power and energy efficient design. *IEEE Trans. Circuits Systems* **59**, 193–198. (doi:10.1109/TCSII.2012.2188451)
42. Maerkl SJ, Quake SR. 2007 A systems approach to measuring the binding energy landscape of transcription factors. *Science* **315**, 233–237. (doi:10.1126/science.1131007)
43. Moran U, Phillips R, Milo R. 2010 Snapshot: key numbers in biology. *Cell* **141**, 1262–1262.e1. (doi:10.1016/j.cell.2010.06.019)
44. Wagner A. 2007 Energy costs constrain the evolution of gene expression. *J. Exp. Zool. Mol. Dev. Evol.* **308B**, 322–324. (doi:10.1002/jez.b.21152)

45. Hopfield JJ. 1974 Kinetic proofreading: a new mechanism for reducing errors in biosynthetic processes requiring high specificity. *Proc. Natl Acad. Sci. USA* **71**, 4135–4139. (doi:10.1073/pnas.71.10.4135)
46. Laughlin S. 1998 The metabolic costs of neural information. *Nat. Neurosci.* **1**, 36–41. (doi:10.1038/236)
47. Attwell D, Laughlin SB. 2001 An energy budget for signaling in the grey matter of the brain. *J. Cereb. Blood Flow Metab.* **21**, 1133–1145. (doi:10.1097/00004647-200110000-00001)
48. Kim J, Hopfield JJ, Winfree E. 2004 Neural network computation by in-vitro transcription circuits. *Adv. Neural Inf. Process. Syst.* **17**, 681–688.
49. Ferrell JE. 2009 Signaling motifs and Weber's law. *Mol. Cell* **36**, 724–727. (doi:10.1016/j.molcel.2009.11.032)