Northumbria Research Link

Citation: Grozinger, Lewis, Amos, Martyn, Gorochowski, Thomas E., Carbonell, Pablo, Oyarzún, Diego A., Stoof, Ruud, Fellermann, Harold, Zuliani, Paolo, Tas, Huseyin and Goñi-Moreno, Angel (2019) Pathways to cellular supremacy in biocomputing. Nature Communications, 10 (1). p. 5250. ISSN 2041-1723

Published by: Nature Publishing

URL: https://doi.org/10.1038/s41467-019-13232-z <https://doi.org/10.1038/s41467-019-13232-z>

This version was downloaded from Northumbria Research Link: http://nrl.northumbria.ac.uk/id/eprint/41564/

Northumbria University has developed Northumbria Research Link (NRL) to enable users to access the University's research output. Copyright © and moral rights for items on NRL are retained by the individual author(s) and/or other copyright owners. Single copies of full items can be reproduced, displayed or performed, and given to third parties in any format or medium for personal research or study, educational, or not-for-profit purposes without prior permission or charge, provided the authors, title and full bibliographic details are given, as well as a hyperlink and/or URL to the original metadata page. The content must not be changed in any way. Full items must not be sold commercially in any format or medium without formal permission of the copyright holder. The full policy is available online: http://nrl.northumbria.ac.uk/policies.html

This document may differ from the final, published version of the research and has been made available online in accordance with publisher policies. To read and/or cite from the published version of the research, please visit the publisher's website (a subscription may be required.)

Northumbria University NEWCASTLE

University Library

nature COMMUNICATIONS

PERSPECTIVE

https://doi.org/10.1038/s41467-019-13232-z OPEN

Pathways to cellular supremacy in biocomputing

Lewis Grozinger[1], Martyn Amos[2], Thomas E. Gorochowski[3,4], Pablo Carbonell[5], Diego A. Oyarzún[6,7], Ruud Stoof[1], Harold Fellermann[1], Paolo Zuliani[1], Huseyin Tas[8] & Angel Goñi-Moreno[1*]

Synthetic biology uses living cells as the substrate for performing human-defined computations. Many current implementations of cellular computing are based on the "genetic circuit" metaphor, an approximation of the operation of silicon-based computers. Although this conceptual mapping has been relatively successful, we argue that it fundamentally limits the types of computation that may be engineered inside the cell, and fails to exploit the rich and diverse functionality available in natural living systems. We propose the notion of "cellular supremacy" to focus attention on domains in which biocomputing might offer superior performance over traditional computers. We consider potential pathways toward cellular supremacy, and suggest application areas in which it may be found.

Synthetic biology[1,2] is a rapidly growing field of research which applies engineering concepts and principles to the rational engineering of living systems, such as bacteria and yeast. The promise of synthetic biology lies in its potential to provide new substrates for computation[3], production[4], pollution control[5] and medical diagnosis[6] (among many areas), and to harness the "wetware"[7] inside the living cell for human-defined purposes. Synthetic biology is set to become a significant component of the multi-billion dollar bio-economy[8], but, in addition to tangible benefits such as cheaper drug production or more precise bio-sensing, many researchers in the field believe that the very process of re-engineering life will both require and inform a reexamination of our fundamental understanding of cellular processes[9]. This position underpins the current paper.

Much existing work in synthetic biology is concerned with the construction of "circuits" of biological components (such as genes and proteins) that receive some input (either endogenous, or exogenous), perform some transformation of that input, and produce a result determined by the "rules" encoded in the circuit. This input-transform-output pipeline is entirely consistent with the notion of a computation, as traditionally defined, and it is wholly unsurprising that synthetic biology has, so far, largely adhered to the genetic "logic circuit" model (groundbreaking examples include the genetic toggle switch[10] and the "repressilator"[11]).

The roots of this biological circuit metaphor date back to the mechanistic view of the cell; the idea that living systems may be viewed as machines, which, in turn, can be traced as far back as

[1] School of Computing, Newcastle University, Newcastle Upon Tyne NE4 5TG, UK. [2] Department of Computer and Information Sciences, Northumbria University, Newcastle upon Tyne NE1 8ST, UK. [3] BrisSynBio, University of Bristol, Tyndall Avenue, Bristol BS8 1TQ, UK. [4] School of Biological Sciences, University of Bristol, Tyndall Avenue, Bristol BS8 1TQ, UK. [5] Manchester Synthetic Biology Research Centre for Fine and Speciality Chemicals (SYNBIOCHEM), Manchester Institute of Biotechnology and School of Chemistry, University of Manchester, Manchester M1 7DN, UK. [6] School of Informatics, University of Edinburgh, Edinburgh EH8 9AB, UK. [7] School of Biological Sciences, University of Edinburgh, Edinburgh EH9 3BF, UK. [8] Systems and Synthetic Biology Program, Centro Nacional de Biotecnología (CNB-CSIC), Campus de Cantoblanco, 28049 Madrid, Spain. *email: angel.goni-moreno@newcastle.ac.uk

Descartes (and beyond). Characterisations of the cell as a "factory", a "little engine", and a "chemical machine" dominated early discourse in cell biology, and the post-war development of cybernetics and computer science, combined with the emergence of molecular biology, provided significant support for this conceptual interpretation of living systems[12].

The "machine model" of the cell was further cemented by the pioneering work of Jacob and Monod, which established that the *lac* operon[13] facilitates metabolic switching within *E. coli* in a manner that may be interpreted as a simple Boolean circuit. In his highly influential popular book, *Chance and Necessity* (a chapter of which is titled "Microscopic Cybernetics"), Monod made explicit the connection between biological processes and the basis of electronic circuits:

"The logic of biological regulatory systems abides not by Hegelian laws but, like the workings of computers, by the propositional algebra of George Boole."[14]

Since the discovery of the *lac* operon, the machine model has largely dominated molecular biology[15], and the notion of the genome serving as a "genetic program" is still relatively common[16]. Of course, abstracting biological processes to the level of circuits (Boolean or otherwise) or even programs may be a necessary step in terms of making sense of their operation. Conversely, in the context of engineered biological systems, the availability of a set of well-characterised, modular and tunable components are perhaps necessary for creating controllable and predictable systems[17]. We emphasise here that we do not seek to overturn or abandon an extremely powerful and useful metaphor. However, it may give a misleading impression regarding the reliability and predictability of genetic circuits, compared to their electronic counterparts[18]. Moreover, as we argue in this paper, by restricting ourselves to the computational palette offered by (Boolean) combinatorial logic, we inherently (and seriously) limit the power and scope of synthetic biology.

It is perhaps instructive at this point to discuss the field of molecular computing[19], which many see as a conceptual precursor to synthetic biology, if not a direct predecessor. Although the notion of computing with individual atoms and molecules was attributed to Richard Feynman as early as the 1950s[20], the field was only reified in 1994, when Leonard Adleman published his groundbreaking paper on computing solutions to the Hamiltonian Circuit Problem using molecular operations with strands of DNA[21]. At the time, a commonly-held belief was that the future promise of DNA computation[22] lay in solving instances of hard computational problems that were beyond the capabilities of traditional silicon-based computers[23]. Much early work on producing general DNA-based computational models focussed on molecular emulation of the Boolean circuit model[24,25]. Discussion of the field's overall direction was, for a time, dominated by the quest for the "killer application", the application of DNA computing that would establish its superiority over existing computational substrates[26,27]. In some ways, this may be viewed as a restricted form of the search for so-called "quantum supremacy" in the quantum computing domain[28].

As the field of molecular computing evolved, it rapidly became clear that the killer application would not be found by solving large instances of computationally hard problems using any variant of Adleman's original model. The elegance of the original solution derived from the encoding used to construct all possible paths through a given graph; vertices and edges were represented as strands of DNA that self-assembled into a collection of paths, which was then rapidly filtered in parallel using molecular manipulations to gradually extract valid solutions. Crucially, this reasonable run-time required the entire solution space for a given problem instance to be represented in an initial "test tube" at the start of the computation; the problem was to find the "needle" (valid solution(s)) in a molecular "haystack". Essentially, the entire space of solutions was tested in polynomial time using molecular operations, but this came at the cost of having to represent the whole solution set in a single volume of liquid[22]. For even small problem instances, the amount of DNA required to represent all possible solutions would be astronomical; Hartmanis calculated that if Adleman's experiment were scaled up from 7 to 200 vertices, the amount of DNA required would weigh more than the Earth[29]. We note, however, that this inherent time-space tradeoff would appear to hold regardless of whether the underlying substrate is DNA, silicon, or a more exotic material (with the possible exception of substrates for quantum computers[30]).

Importantly, this assessment emphasised the fact that the killer application would probably not be derived from single-use, finite DNA-based systems (albeit ones that were large and potentially massively-parallel in nature). Subsequent influential work on DNA self-assembly[31] and strand displacement architectures[32] emulated traditional computational models such as cellular automata[33] and artificial neurons[34], but also took advantage of biochemical processes that provided power beyond mere miniaturisation. That is, rather than simply harnessing the potential for massively-parallel laboratory operations on large (but finite) numbers of molecules, these systems co-opted specific physical and chemical properties of the underlying substrate to generate the potential for autonomous operation and dynamic behaviour. In the context of synthetic biology, we suggest the killer application for engineered living systems will be similarly derived; one that will use features of the living system that are unavailable via traditional silicon-based substrates, for the purposes of applications that exploit autonomy and the ability to perform in uncertain environments.

The search for cellular supremacy

If we accept that a driving motivation for synthetic biology should be the search for a killer application, then this naturally leads to a consideration of the class of problems in which engineered living systems might offer what we call cellular supremacy (deliberately echoing the well-established notion of quantum supremacy[28,35–37]). That is, we seek a set of problem domains in which cell-based systems will offer capabilities beyond the reach of existing computers, due to cost or technological constraints. Simply put, cellular supremacy will be determined by the set of problems that biocomputers can practically solve that traditional microprocessor-based devices cannot. To identify this set of problems, we naturally focus on the "added value" that living systems offer beyond silicon-based hardware. In the following sections, we discuss a number of features of living systems that we believe give synthetic biology-based solutions a distinct "edge" over traditional computers. Importantly, we focus on aspects of computation and information transformation, rather than simply on the ability of the cell to act as a micro-scale drug precursor "production plant" or miniaturised bio-sensor. While these applications are important and valuable, they are not the main focus of the current discussion, which centres on the general computational capabilities of living systems if we go beyond the limitations of Boolean logic-based systems. In what follows, we use the term "cellular computing"[38,39] to emphasise this focus on computation.

When assessing cellular computing against traditional computing, it may be useful to frame the comparison using a scheme that emerged from an exercise on roadmapping the so-called "unconventional computation" domain. The authors identified a number of criteria or benchmarks against which emerging computational models may be measured, each accompanied by its own motivating question[40]. In what follows, we focus on the

quality criterion, which asks the following question: Does the alternative model offer qualitatively different computation to traditional models? The authors define "quality" in terms of precision, richness, stochasticity and repeatability. Quality is concerned with the ability of the system to provide an analysis of inputs that is different to that offered by a traditional computer. If we focus on problems that fall within the remit of traditional silicon-based machines, and use existing metrics, then cellular computing will inevitably fall short on some of the other criteria identified[40], such as speed and cost. However, we would argue that the real power of cell-based systems derives from the richness of their internal (and collective) architectures, and on their inherent stochasticity.

This leads us to a consideration of the specific computationally expressive mechanisms that are available to us, and which act both inside and between living cells. In what follows we focus on five (non-exclusive) features of living cells that (we believe) offer the most potential in terms of the search for cellular supremacy; these are reconfigurability (that is, the ability to change internal state/structure in response to signals), noise (the ability to not only tolerate but to harness and exploit biological "messiness"), concurrency (the multitude and richness of inter-cellular processes that facilitate massively-parallel communication and coordination), representation (the ability to use non-binary representations for signals), and evolution (the population-level ability to seek out novel solutions to problems over time). By harnessing these capabilities, we will obtain bio-computing systems that are self-organising, self-repairing, resilient, distributed, and adaptive[41].

First, we consider aspects of computational theory that suggest how cellular computers may out-perform traditional machines. We then describe a number of features of biological systems that lend themselves to non-standard computation. Taken together, these two perspectives point towards a number of possible areas for investigation in which the killer application for cellular computing may reside.

Information processing beyond digital logic

The current definition of digital computation is based on the abstract model defined by Turing in the 1930s[42], and the von Neumann architecture[43] used to implement the types of computations performed by the Turing Machine. Although Turing's model provides a framework for answering fundamental questions about computation, "…as soon as one leaves the comfort provided by the abundance of mathematical machinery used to describe digital computation, the world seems to be packed with paradoxes."[44].

The important thing to note here is that although genetic circuits may appear to behave digitally, it is only the collective behaviour of a large number of inherently analogue components that give rise to this property. The fact remains that we still currently lack any formal framework within which to argue that a cell computes, according to any understood model of computation (although recent work has started to address this[45]).

The nature of computation is not of purely theoretical interest. Mathematical models of computation and their properties inform the engineering of their physical manifestations (Fig. 1). Many such implementations may be possible, but all inherit the characteristics of their abstract counterparts—both their abilities and their limitations. The fact that the nature of computation within a given model is independent of its implementation allows the application of theoretical computer science to all kinds of physical systems, including cells. However, the cost of this generality is a semantic gap between the model and the physical processes that actually perform the computation. That is, mapping computations from an abstract model to a real system may be more or less difficult, depending on the model chosen. The cellular computing substrate is quite different from that of silicon computers, offering opportunities for implementing some models with a narrower semantic gap. Practical considerations such as these could guide future applications of cellular computing.

Conventional silicon computers are fundamentally implementations of a deterministic, centralised and digital model of computation, and they excel at computational tasks which are easily described by such models. In contrast, cellular computing has been optimised over billions of years of evolution to perform very different computational tasks, and we are unlikely to find cellular supremacy in applications such as discrete mathematics, sending emails or reading documents. However, computer science has developed models in which the nature of computation is quite different to that of the Turing Machine[46–53] and computation in cells appears to share some properties with these less

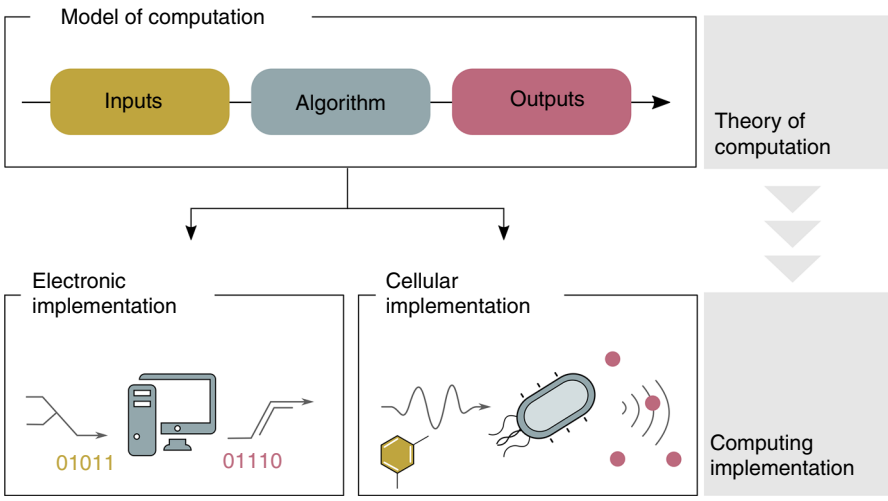

Fig. 1 The cell as a "physical" computer. A model of computation formally defines inputs and outputs, as well as how an algorithm processes inputs into outputs. Though the same theoretical model of computation can be physically implemented in many different ways, the nature of computation remains the same. Electronic implementations receive electronic data for inputs/outputs, while cells are able to sense/deliver a wide range of physical, chemical and biological inputs/outputs. The encoding of information into inputs can be done in different ways. Temperature, for instance, can be encoded as the height of mercury in a tube, the voltage of an electronic thermometer or the state of a DNA thermosensor

conventional models. With this in mind, the application of theoretical computer science to cellular computation may still present routes to cellular supremacy in those areas where the properties of cellular and traditional computation differ.

Here, we introduce a selection of theoretical aspects of computing that impact significantly upon the nature of computation, all of which seem to be exploited in the cell for processing information, but whose theoretical implications may not be familiar to those outside the field of computer science.

Computational states for computations over time

Not all models of computation are equally powerful, that is, they are not equivalent in terms of the range of computations they can possibly describe. By this metric, combinatorial logic circuits are extremely weak, since the output of a circuit is purely a function of its current inputs (that is, there is no "memory"), and this severely restricts the range of computations that are possible with a single circuit. Clearly, there are models of computation more powerful than combinatorial logic, that is, models which can describe all computations that are possible with combinatorial logic circuits and more. In this regard, two extremely powerful models of computation, the Lambda calculus[54] and the Turing Machine[42] are particularly important. These two models are both equally powerful, but quite different in their formulation—the Lambda calculus is a model of functional computation, while the Turing Machine models stateful computation. Here we focus on stateful computation, as states enable functions that operate over time, such as memory, learning and adaptation.

The output of stateful computations depends not only on the current input, but also on the current state, which encodes information about previous inputs that are no longer present. Models of computation such as finite state machines (FSMs) (shown in Fig. 2) exceed the capabilities of combinatorial logic, and are extremely powerful models of computation—the central processing units of modern digital computers are built from sequential logic circuits, which are implementations of FSMs. Even so, they cannot describe all computations that are available to the Turing Machine model, since they have only a bounded number of states. Turing machines are similar to FSMs, but have an unbounded memory (commonly conceptualized as a tape), which endows them with an unbounded computational state space (number of distinct configurations). Consequently, Turing Machines can compute anything that any real computer can, and in fact, computer science starts from the notion that Turing Machines (TMs) demarcate what is (and is not) computable[42]. From a practical perspective, these considerations of the limits of computability might be of less conceptual value than the strategy that TMs employ to decouple the complexity of the machine from the number of states on which they can operate.

Modern digital computers, as with any real physical system[55], lack the unbounded state space required for implementing a Turing Machine. However, their state space is so vast as to make them practically indistinguishable from implementations of Turing Machines (consider that even systems with an extremely modest 1 MB of storage have at least $2^{10^6 \times 8}$ states). It is therefore unlikely that cellular computing will outperform silicon purely on the basis of enabling more complex stateful computations. Nevertheless, synthetic biology has already engineered successful in vivo implementations of stateful computations[10,56,57], and there also exists significant theoretical work linking stateful models of computation to biological implementations[58,59]. This work is undoubtedly of considerable importance for a broad range of synthetic biology applications, and since biological systems naturally engage in stateful computation[60–62], statefulness

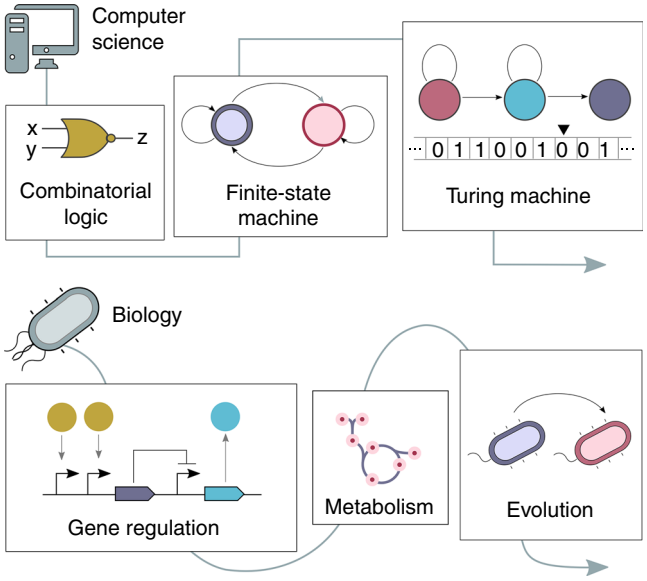

Fig. 2 Cells could provide more than logic circuits. Computer science has developed models of computation that are far more powerful than combinatorial logic, such as finite-state machines or the Turing Machine. These models are more powerful in that they allow processing of a wider range of inputs into outputs, and in many more ways, than are admissible by combinatorial logic. Similarly, living systems have evolved a variety of computational processes to allow cells to process information. A simple model, used extensively as the basis for engineering combinatorial logic circuits in cells, is the standard representation of the central dogma (CD) of molecular biology. However, this model does not incorporate core cellular mechanisms such as metabolism, or processes such as evolution, which may provide possibilities for building more powerful, but as yet unknown, models

or "memory" will likely be a key component in future engineered biocomputations of any significant complexity.

Noise permits stochastic and non-deterministic computations

Combinatorial logic circuits, FSMs and TMs model deterministic computations, which are essentially step-wise descriptions of mathematical functions that map inputs to outputs, where each step follows in a predetermined way from the previous step. Deterministic computation can be generalised to include stochastic and non-deterministic computation. Stochastic (or probabilistic, or randomised) computing refers to algorithms which can make random choices during their execution. Specifically, a given input may generate a number of different computational paths, each with some probability, and the cumulative probability of all such paths is equal to one. Probabilistic algorithms are a cornerstone of computer science[63], and are used to solve approximately, but efficiently, hard optimisation problems for which deterministic algorithms would be intractable. One of the most important algorithms in systems biology, the Gillespie algorithm, is stochastic. Gillespie showed from basic quantum mechanics that a set of biochemical reactions must be understood as a stochastic process[64], and offered an efficient way for simulating (sampling) its dynamics[65]. Therefore, cells may already be seen as "stochastic processors" that could provide a substrate for implementing probabilistic algorithms. Indeed, the utility of stochastic processors has already been recognised in the silicon world[50,66].

In non-deterministic models of computation, as in stochastic ones, there may be many computational paths from input to output, but crucially each of these computational paths is

explored simultaneously by the algorithm, with a result analogous to parallel execution of multiple distinct deterministic algorithms. While non-determinism has considerable impact on theoretical computer science, practical implementations of non-deterministic computation remain elusive. Nevertheless, algorithmic solutions to some computational tasks can be significantly easier to express as non-deterministic, and biological systems extensively exploit non-determinism, both at the population level[67,68] and at the scale of evolution[69]. Perhaps more speculatively, we might also consider the role of quantum effects in biology[70], and whether or not these may be harnessed for the purposes of non-deterministic algorithms in biological systems.

Distributed computing can exploit concurrency

For some models of computation, the definition of an algorithm is "sequential", and implies an ordering of the computational steps. Implementations of these models enforce these "sequential semantics", because sequential execution may be a strict requirement to ensure the correctness of an algorithm. However, for certain computations, algorithms exist in which the ordering of computational steps may be relaxed, or abandoned entirely—in computer science terms, the algorithm displays a level of concurrency. In such situations, if multiple computing devices may interact and coordinate their efforts as a "distributed system", concurrent parts of an algorithm may be executed in parallel, potentially speeding up computations significantly.

The utility of such distributed computing is not limited to performance advantages. For some problems, sequentiality is desirable, but the nature of the computing substrate makes it impractical (or even impossible) to enforce[71]. For instance, at a fundamental level, and even within a single cell, natural cellular computation is concurrent and distributed. Computation is performed by stochastic biochemical processes—the probabilistic interactions of spatially separated molecular computing "agents", many of which can happen in any order. Computer science has long used distributed and concurrent models of computation such as actors[49], Petri nets[51], process algebras[72] and population protocols[52] to describe computations in just these situations.

Analogue computing allows for continuous non-binary inputs

The nature of computation described so far has been digital; that is, information is represented using a set of discrete values. However, many physically relevant quantities vary continuously. Even the signals that encode binary values in electronic logic circuits are, in reality, stored as continuous-valued voltages. Such signals must be discretised in order to represent digital values. In contrast, analogue computation allows continuous signals to be used directly in the computational steps of an algorithm, and for representation of information as continuous values.

Although many cellular computations involving binary "yes/no" decisions may be interpreted as digital computations, and digital logic computations are certainly suited to applications such as biosensors, cells often exhibit graded responses to stimuli that are more appropriately viewed as analogue computations[62,73]. Furthermore, the biochemical processes responsible for cellular computations involve discrete interactions of discrete molecules, but are also inherently stochastic. Cellular computing may, therefore, be viewed as both digital and stochastic, or as analogue computation with noise[74], and the viability of the cell as a substrate for synthetic analogue computations has already been demonstrated[75].

Whilst some models of analogue computation such as the Blum-Shub-Smale machine[53] or neural networks[46], use arbitrary precision signals to perform super-Turing computation, it is not clear whether realisations of analogue computation will be able to provide more computational power than digital computation, since physical laws forbid the existence of such an idealised computer[55,76]. However, for digital models of computation, the abstraction of continuous signals into digital represents a semantic gap between the formal definition and implementation of a computation. Models of analogue computation admit continuous signals, which not only narrows this semantic gap, but also allows more powerful computational primitives to be used for defining algorithms[77]. Consequently, depending on the computation and the computing substrate, an analogue algorithm might have a more intuitive implementation, require fewer components to implement, and be considerably more energy-efficient than its digital counterpart[74,75].

Engineering complex cellular computing systems

There are certainly deep physical connections between chemistry and electronics[78], but the fact remains that the cellular environment is a radically different computing substrate than silicon. Although this difference might make cells unsuitable for computational tasks traditionally dominated by conventional computers, it could also offer opportunities to explore more unconventional models of computation. Aside from gene regulation, which has been useful for engineering biological logic circuits, a number of processes and features exist in natural systems which may offer computational capabilities. Here, we identify four such resources as promising in terms of their information-processing capabilities (Fig. 3).

Towards whole-cell computation

Synthetic cellular computing systems have generally focussed on DNA components that are isolated from the many other processes within a cell[79,80]. However, evolution has shaped intricate information-processing networks whose function emerges from the interplay between many layers of the cellular machinery[81]. For example, bacteria adapt to nutrient fluctuations by sensing extracellular cues, transducing such signals to the genetic machinery, and ultimately adapting their metabolic state to make maximal use of carbon sources. This requires a careful whole-cell orchestration of trafficking processes at the levels of membrane, cellular signalling pathways, gene expression programmes and metabolic activity.

In a similar fashion, synthetic systems that exploit whole-cell interactions offer exciting opportunities to expand the range of computational tasks that can be achieved with living systems. Metabolic circuits, in particular, produce analogue dynamics that could allow us to move beyond basic Boolean operations[82]. Circuits integrating genetic programs with metabolic activity have been successfully engineered to improve the productivity of microbial cell factories[83–85], but their potential goes far beyond simple production. Notably, retrobiosynthesis approaches based on generic representations of reactions, known as reaction rules, expand the design space of metabolic circuits by predicting novel synthetic pathways connecting metabolism and gene expression[86–88]. Machine-learning is increasingly being applied by such retrosynthesis algorithms to select the best candidate reactions in the search through the chemical space[89,90].

Moreover, experimental and theoretical work has shown that gene expression noise can permeate to metabolism[68,91], suggesting the possibility of processing information using metabolic heterogeneity. The interplay between gene expression and metabolism can be conceptualised in terms of memory systems employed in most computer architectures. These have a long-term static memory, conceptually similar to information stored in DNA, and a short-term volatile memory, akin to the fast dynamics of metabolism (Fig. 3a). Computer science has

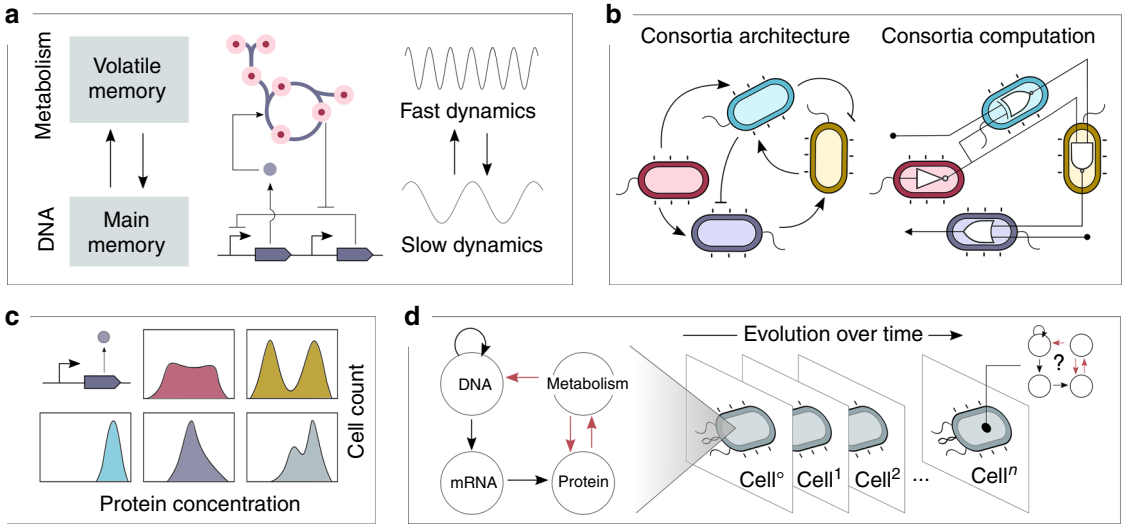

Fig. 3 Cellular information-processing fundamentals that go beyond combinatorial logic circuits. **a** Whole-cell computations, merging genetic and metabolic circuits, could achieve more ambitious goals than genetic circuits alone. Cells have evolved intricate networks that make simultaneous use of the varied features of both genetic and metabolic processes. In terms of information storage, metabolism presents a volatile memory, while DNA sequences are able to store information in a more stable fashion. Coordinating the use of different types of memory is a fundamental aspect of complex computer architectures. The dynamic difference is also a potential source of complexity if coupled; metabolic reactions operate on a faster timescale relative to genetic regulatory networks. **b** Multicellular computing (right) is currently implemented by connecting the output of one strain to the input of another. Social interactions among cells (left), such as cooperation, mutualism, competition or commensalism, are not considered in general. However, social interactions are fundamental in natural communities—they provide stable architectures executing a desired computation. **c** Gene expression noise is intrinsic to living systems; the panel figure shows different patterns for gene expression. Despite the fact that all are described as being on, there are different types of expression—thus different on/off standards. **d** The cell as a general-purpose machine. As the basis for a model of computation, the central dogma of molecular biology can be expanded to include metabolism. Evolutionary processes may also be included as major forces guiding information-processing in cells, since they allow the purpose of cellular computations to adapt over time

developed efficient techniques for managing data flows across these two memories to exploit their characteristic timescales. There is some evidence that cells also exploit such differing timescales to their benefit[62], and, ultimately, the ability to encode signals in different frequency domains may allow us to move from classic Boolean abstractions to more complex coding systems.

Whole-cell computation presents major challenges because, as the size of synthetic circuit grows, so does their footprint on the cellular host. Synthetic circuits do not work in isolation, but continuously draw resources away from vital functions of the host[92]. These resources include energy, polymerases for transcription, ribosomes for translation, enzymatic co-factors, and so on, the depletion of which has a negative impact on native processes. This is commonly known as the cellular burden, and has gathered substantial attention in the community[93]. Recent work has aimed at identifying relevant sources of burden imposed by synthetic constructs[94,95]. Together with increasingly powerful mathematical models for cell physiology[96,97], these experimental efforts are paving the way for accurate prediction[98] and control of burden[99], which will enable the construction of complex circuitry that takes full advantage of the cellular machinery[100].

The power of multicellular consortia

The burden imposed by synthetic biological constructs places a fundamental ceiling on the information processing capacity of a single cell. In order to execute larger and more complex computations, living systems harness the capabilities of multicellular ensembles, such as the vast neural networks in the brain. Distributing computational demands across a population of specialised cells lowers the burden on each individual cell, and offers a way to more easily scale the complexity of the computations being performed. Such scalability is uniquely fostered in biological systems by the inherent ability of living cells to self-replicate. Similar distributed approaches have been employed in engineered biological systems, where an algorithm is spread across a consortium of different microbes that are able to communicate and interact (Fig. 3b). Many theoretical[101–105] and experimental[106–112] examples demonstrate the benefits of distributing computation in this way, as well as a growing number of tools for their simulation[113–120] and assembly in living cells[121,122].

Although the development of engineered multicellular consortia is possible with current technologies, their structure is often fragile, and their information-processing capabilities are dwarfed by microbial communities found in nature[123]. A key difference in naturally-occurring systems is that extensive resources and diverse mechanisms are used to establish and coordinate the complex social behaviours of a community's participants[123]. Social relationships such as cooperation, mutualism, competition and commensalism are used concurrently to drive a required community structure that can potentially change over time in response to the environment. Advances in our ability to control cell-to-cell interactions, either through spatial patterning of cell populations[112] and/or orthogonal communication channels[121,124], will be crucial to future developments in this area.

A feature of multicellular consortia that pushes us beyond the capabilities of classical electronic computers is their highly flexible and re-configurable nature[125]. In computer science, a computer architecture describes the structural relationship between different components of a system (e.g. the processor and memory) and constrains how algorithms run on the system. A program compiled for one type of computer architecture may not be executable by another due to these physical differences. The architecture of most electronic computers is fixed during a computation and has been honed to perform primitive digital operations in quick succession, guided by a set of instructions (the code). While this

architecture is general enough for tackling many types of task, its static architecture makes it sub-optimal for most of them. In a cellular consortium, such a severe trade-off between flexibility and optimality (e.g. in regards to energy consumption or speed) need not exist, as the cellular community can dynamically change its physical composition and the interactions present to produce a "living architecture" that better matches the problem at hand. Furthermore, such reconfiguration is possible even during a computation itself, offering a level of control and flexibility that electronic computers are unable to match. Harnessing these unique capabilities will offer synthetic biologists new approaches to solving problems and require us to rethink what a computer architecture can be[126].

Encoding signals using gene expression noise

Combinatorial logic, which relies on well separated on/off signals to encode information, constitutes a valuable framework with which to understand and build genetic circuits. Even if molecular signals (for example, a transcription factor that down-regulates a promoter) do not have absolute and constant values, the notion of a promoter being repressed or not (thus being binary) is intuitive. Nevertheless, the use of Boolean logic for engineering cellular circuits is often challenged by the fact that molecular signals are intrinsically noisy, and therefore stochastic. Each signal will have different on/off values (Fig. 3c), and not all are compatible, which makes building a large combinatorial circuit a significant challenge[127]. Combinatorial logic is, therefore, a good example of how a given theoretical model of computation may be easily implemented with one physical toolkit (electronic), but not with another (molecular).

The voltage that runs through electronic wires is also noisy, but computer engineers overcame this problem by arranging for thresholds at the input and output of logic gates, so that analogue values could be abstracted into two unique values: 0 and 1. In contrast to this, living systems have evolved to benefit from signal variability. Stochasticity has been shown to be useful in coordinating the expression of large sets of genes[128], in producing the phenotypic variability that allows for bet-hedging strategies that promote survival in fluctuating environments[67,129], and as a key component of evolution under tight selection criteria[69]. Therefore, models of computation incorporating stochasticity would seem to be necessary to best fit the intrinsic properties of living systems.

Unlike silicon computing devices, where noise is a consequence of random signal fluctuations, cellular systems are capable of encoding important information into noise patterns. For example, the change in gene expression noise according to intracellular physical distances[130] reveals information about the structural architecture of the cell. Also, noisy patterns in the cellular machinery of *Staphylococcus aureus* (that lead to different infectious cell types) originate in upstream molecular events[131]. These examples, among others[75], highlight the fact that the shape of non-digital biological signals is used to effectively transmit information. Therefore, the challenge is to use noise for implementing computations — an area where cellular computers clearly outperform conventional ones. In the meantime, current efforts in the discretisation[132], stabilisation[133,134] and reusability[135] of noisy signals may help to understand, and harness, the dynamic complexity of molecular interactions.

Evolution as an information-processing mechanism

Evolutionary computing is a field within computer science that develops and applies algorithms inspired by Darwinian evolution[136]. Different implementations of this idea exist, such as genetic algorithms, evolutionary strategies or genetic programming, but the underlying principles are similar: a set of candidate solutions to a problem (an initial population) compete under some pre-defined fitness criteria, and the fittest individuals are selected to form the next generation (via crossover and/or mutation). An important advantage of these techniques is that they allow for the dynamic optimisation of solutions throughout potentially vast search spaces.

Both computer scientists and living systems use evolutionary algorithms to generate algorithmic solutions. Evolutionary processes have also been used to design biomolecules[137,138], libraries of biological components[139] and even to evolve non-functional genetic circuits into functional ones[140]. Despite this, evolutionary processes are often omitted when engineering computing circuits in living cells. Harnessing the power of such information-processing mechanisms for predefined functions seems to be difficult, and the rational engineering of autonomous evolutionary computing in living cells is still an overarching challenge.

In order to start addressing this challenge, it will be important to formalise the effects of evolutionary dynamics on the flow of genetic information. A potential initial step would be to expand on the standard representation of the central dogma (CD) of molecular biology to include evolutionary dynamics, in an attempt to describe the model of computation of a living cell (Fig. 3d). Expansion of the CD-based model is not a new idea; for example, the inclusion of metabolic processes has previously been suggested[141], since, as evidenced above in "Towards whole-cell computation", metabolism is a crucial omission from the point of view of information processing. A complete model of the flow of genetic information at a given point in time needs to include both genetic and metabolic processes, and dynamic models of computation inside living cells should include evolutionary processes that operate on genetic information.

Theoretical models of computation may find valuable and perhaps novel implementations within this dynamic representation of the CD. Recent developments in optogenetics are promising to this end, since these show how genetic elements dynamically shape their response in relation to external signals[142]. Importantly, these studies linking environmental signals to intracellular responses in a predictable way, as well as directed evolution efforts[143], are undertaken in tightly controlled systems. However, the mechanisms that allow a cell to thrive in challenging and dynamic environments are largely unpredictable. For problems like bio-remediation which intrinsically involve ever-changing environments, natural cellular computers clearly show superiority compared to conventional ones.

Applications of cellular computing systems

As already discussed, cellular computing is unlikely to compete with conventional computers in domains for which the latter are specifically engineered. Although future biological systems may offer competitive performance in related areas such as data storage[144], the benefits are largely limited to the exploitation of material factors such as miniaturisation and longevity of components.

The identification of promising applications for cellular computing has been largely guided by the observation that living systems operate best in application domains that are not easily reachable by conventional computers. Successes in bio-remediation[145], bio-production[146] and targeted therapeutics[147] indicate that this is indeed a fruitful line of inquiry. Nevertheless, the implementation of conventional computations with unconventional computing substrates does not, in and of itself, constitute cellular supremacy. Rather, supremacy must be derived from the fact that the type of computation performed by a conventional computer is qualitatively different to that executed by living systems. Current implementations of computations within

living cells are generally based on combinatorial or sequential logic circuits mapped onto equivalent genetic circuits; while these are useful metaphors for both understanding and engineering living systems, digital logic is not easily implemented using a cellular substrate. Furthermore, logic circuits offer a relatively bland computational palette compared to the richness of biology. For this reason, future developments in cellular computing should focus on models of computation that both accommodate and exploit the natural abilities of the cell, and avoid forcing biological systems into artificial (and often unsuitable) architectures[148].

This does not mean that cellular computing must necessarily "reinvent the wheel" when it comes to models of computation. Living systems share many characteristics with in vitro platforms for computation, and as well as the considerable body of theoretical work concerning molecular computing, cellular computing may draw on cell-free systems as a practical resource for faster prototyping with a greater degree of control[149–151]. Furthermore, the relationship between computer science and natural computing is often synergistic. As described earlier, models of computation that transcend logic circuits are available which have yet to be explored by cellular computing to any great extent. In return, biological systems offer computational features that are (and are expected to remain) unavailable to silicon implementations of existing computational models. It is at this intersection that exciting opportunities for cellular supremacy emerge. For example, in the past decade it has become increasingly clear that a number of biological processes show quantum mechanical properties[152,153]. In particular, there is strong experimental evidence that long-lived quantum coherence is involved in photosynthesis[154], and that quantum tunnelling is active in enzyme catalysis[155]. Since most quantum computing devices are built and run under stringent environmental conditions (at temperatures approaching absolute zero), the opportunity to control quantum effects in a biological system that "runs" at room temperature, through the emergence of a "quantum synthetic biology", could turn out to be a game changer in the quantum supremacy race. More to the point, realising models of quantum computation[156] using quantum biology could yield a cellular computer capable of a radically different kind of computation than silicon[153].

Although we have, until now, focussed primarily on bacterial cells, there does, of course, exist a multiplicity of cell types that possess rich and complex computational capabilities. For example, plants can respond to environmental signals and stresses in a way that is reminiscent of signal processing in neural networks (the so-called "plant perceptron")[157]. Cultured neuronal cells have been used to control flight simulators[158] and robots[159], and slime mould has a rich repertoire of spatial computing capabilities[160]. Additionally, we do not dismiss the possibility of "hybrid" architectures, in which cellular systems are interfaced to more traditional, silicon-based machines. Such connections may allow for subtle and sensitive adjustments to cell state, for example[161], but these hybrid systems will still fundamentally rely on the properties of the living material.

As we have already argued, natural cellular computing operates at vast scales, in a distributed manner, and in the presence of considerable noise. Consequently, biological metaphors have served as inspirations for models of amorphous computation, for which cellular computing is a promising implementation technology[162], especially at scales inaccessible to silicon. Robotics has also drawn inspiration from biological computation, particularly in relation to morphological computing, which takes advantage of the physical properties of computing agents in order to achieve more efficient computations[163]. By using intrinsic physical properties of the computational substrate to "outsource" parts of the computation, increasingly complex computations can be carried out whilst maintaining relatively simplistic control structures[164]. Living systems are an ideal implementation technology for morphological computation, since they not only compute solutions to individual instances of problems, but continuously compute and adapt in order to embody an efficient general solution. As we have previously argued, conventional silicon computers have inflexible architectures by comparison, which must sacrifice efficiency for generality. These qualitative differences between cellular and conventional computing suggest that applications such as terraforming and smart material production may remain beyond the reach of silicon computers, but in contrast, strategies for both applications based on living technologies have already been proposed[165,166].

In this perspective, we have championed the notion of cellular supremacy in an attempt to focus attention on the high-impact areas in which synthetic biology can have a truly transformational effect. We call on the community to consider cellular supremacy as a framing device for future work, and to explore in a systematic fashion how it may be established. By accepting the idea of cellular supremacy, we naturally acknowledge the richness and power of living systems. And by ceding a degree of control to biology, we may yet open up a much wider range of applications and perspectives on information processing in nature.

Received: 8 July 2019; Accepted: 29 October 2019;
Published online: 20 November 2019

References

1. Andrianantoandro, E., Basu, S., Karig, D. K. & Weiss, R. Synthetic biology: new engineering rules for an emerging discipline. *Mol. Syst. Biol.* **2**, 2006.0028 (2006).
2. Ausländer, S., Ausländer, D. & Fussenegger, M. Synthetic biology—the synthesis of biology. *Angew. Chem. Int. Ed.* **56**, 6396–6419 (2017).
3. Amos, M. & Goñi-Moreno, A. Cellular computing and synthetic biology. In *Computational Matter* (eds Stepney, S., Rasmussen, S. & Amos, M.) 93–110 (Springer, 2018).
4. Chubukov, V., Mukhopadhyay, A., Petzold, C. J., Keasling, J. D. & Martín, H. G. Synthetic and systems biology for microbial production of commodity chemicals. *npj Syst. Biol. Appl.* **2**, 16009 (2016).
5. de Lorenzo, V. et al. The power of synthetic biology for bioproduction, remediation and pollution control: The UN's Sustainable Development Goals will inevitably require the application of molecular biology and biotechnology on a global scale. *EMBO Rep.* **19**, e45658 (2018).
6. Slomovic, S., Pardee, K. & Collins, J. J. Synthetic biology devices for in vitro and in vivo diagnostics. *Proc. Natl. Acad. Sci.* **112**, 14429–14435 (2015).
7. Bray, D. *Wetware: A Computer in Every Living Cell* (Yale University Press, 2009).
8. Clarke, L. J. Synthetic biology – pathways to commercialisation. *Eng. Biol.* **3**, 2–5 (2019).
9. Condon, A. et al. Will biologists become computer scientists? *EMBO Rep.* **19**, e46628 (2018).
10. Gardner, T. S., Cantor, C. R. & Collins, J. J. Construction of a genetic toggle switch in *Escherichia coli*. *Nature* **403**, 339 (2000). ***Foundational demonstration of a simple bistable device in bacteria, allowing for the possibility of engineered cellular memory***.
11. Elowitz, M. B. & Leibler, S. A synthetic oscillatory network of transcriptional regulators. *Nature* **403**, 335 (2000). ***The influential "represillator", which demonstrated the ability to transmit engineered states across bacterial generations***.
12. Nicholson, D. J. Is the cell really a machine? *J. Theor. Biol.* **477**, 108–126 (2019).
13. Jacob, F. & Monod, J. Genetic regulatory mechanisms in the synthesis of proteins. *J. Mol. Biol.* **3**, 318–356 (1961). ***The first explicit framing of genetic regulation in terms of Boolean logic***.
14. Monod, J. *Chance and Necessity: An Essay on the Natural Philosophy of Modern Biology* (Alfred A. Knopf, 1971).
15. Alberts, B. The cell as a collection of protein machines: Preparing the next generation of molecular biologists. *Cell* **92**, 291–294 (1998).
16. Planer, R. J. Replacement of the "genetic program" program. *Biol. Philos.* **29**, 33–53 (2014).

17. Wang, B., Kitney, R. I., Joly, N. & Buck, M. Engineering modular and orthogonal genetic logic gates for robust digital-like synthetic biology. *Nat. Commun.* **2**, 508 (2011). ***Important demonstration of the construction of a number of genetic logic gates in a modular fashion***.
18. de Lorenzo, V. Beware of metaphors: Chasses and orthogonality in synthetic biology. *Bioengineered Bugs* **2**, 3–7 (2011).
19. Conrad, M. Molecular computing. In *Advances In Computers*, vol 31, 235–324 (Elsevier, 1990).
20. Richard, F. There's plenty of room at the bottom. In *Feynman and Computation*, 63–76 (CRC Press, 2018).
21. Adleman, L. M. Molecular computation of solutions to combinatorial problems. *Science* **266**, 1021–1024 (1994).
22. Amos, M. *Theoretical and Experimental DNA Computation* (Springer, 2005).
23. Lipton, R. J. Using DNA to solve NP-complete problems. *Science* **268**, 542–545 (1995).
24. Ogihara, M. & Ray, A. Simulating Boolean circuits on a DNA computer. *Algorithmica* **25**, 239–250 (1999).
25. Amos, M., Dunne, P. E. & Gibbons, A. DNA simulation of Boolean circuits. In *Proceedings of 3rd Annual Genetic Programming Conference*, 679–683 (1997).
26. Rubin, H. Looking for the DNA killer app. *Nat. Struct. Biol.* **3**, 656–658 (1996).
27. Landweber, L. F. & Lipton, R. J. DNA^2 DNA computations: A potential "killer app"? In *International Colloquium on Automata, Languages, and Programming*, 56–64 (Springer, 1997).
28. Harrow, A. W. & Montanaro, A. Quantum computational supremacy. *Nature* **549**, 203 (2017).
29. Hartmanis, J. On the weight of computations. *Bull. Eur. Assoc. Theor. Computer Sci.* **55**, 136–138 (1995).
30. Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2**, 79 (2018).
31. Rothemund, P. W. Folding DNA to create nanoscale shapes and patterns. *Nature* **440**, 297 (2006).
32. Qian, L. & Winfree, E. Scaling up digital circuit computation with DNA strand displacement cascades. *Science* **332**, 1196–1201 (2011).
33. Rothemund, P. W. K., Papadakis, N. & Winfree, E. Algorithmic self-assembly of DNA Sierpinski triangles. *PLoS Biol.,* **2**, e424 (2004).
34. Qian, L., Winfree, E. & Bruck, J. Neural network computation with DNA strand displacement cascades. *Nature* **475**, 368 (2011).
35. Preskill, J. Quantum computing and the entanglement frontier. *arXiv preprint arXiv:1203.5813*, 2012. https://arxiv.org/abs/1203.5813. ***First usage of the term "quantum supremacy" to denote problems that can be solved using quantum computers in a way that significantly out-performs classical machines***.
36. Boixo, S. et al. Characterizing quantum supremacy in near-term devices. *Nat. Phys.* **14**, 595 (2018).
37. Neill, C. & Roushan, P. et al. A blueprint for demonstrating quantum supremacy with superconducting qubits. *Science* **360**, 195–199 (2018).
38. Amos, M. (ed.) *Cellular Computing* (Oxford University Press, 2004).
39. Teuscher, C. Cellular computing. In *Computational Complexity: Theory, Techniques, and Applications*, 465–478 (Springer, 2012).
40. Stepney, S. & Hickinbotham, S. J. In *Computational Matter* (eds Stepney, S., Rasmussen, S. & Amos, M.) (Springer, 2018).
41. Amos, M., Dittrich, P., McCaskill, J. & Rasmussen, S. Biological and chemical information technologies. *Procedia Computer Sci.* **7**, 56–60 (2011).
42. Turing, A. M. On computable numbers, with an application to the Entscheidungsproblem. *Proc. Lond. Math. Soc.* **s2–42**, 230–265 (1937).
43. Von Neumann, J. First draft of a report on the EDVAC. *IEEE Ann. Hist. Comput.* **15**, 27–75 (1993).
44. Konkoli, Z. et al. Philosophy of computation. In *Computational Matter* (eds Stepney, S., Rasmussen, S. & Amos, M.), 153–184 (Springer, 2018).
45. Horsman, D., Kendon, V., Stepney, S. & Young, J. P.W. Abstraction and representation in living organisms: when does a biological system compute? In *Representation and Reality in Humans, Other Living Organisms and Intelligent Machines*, 91–116 (Springer, 2017).
46. Siegelmann, H. T. & Sontag, E. D. Analog computation via neural networks. *Theor. Computer Sci.* **131**, 331–360 (1994).
47. MacLennan, B. J. Natural computation and non-Turing models of computation. *Theor. Computer Sci.* **317**, 115–145 (2004).
48. Shannon, C. E. Mathematical Theory of the Differential. *Analyzer. J. Math. Phys.* **20**, 337–354 (1941).
49. Agha, G. ACTORS: A model of concurrent computation in distributed systems (1986).
50. Alaghi, A. & Hayes, J. P. On the functions realized by stochastic computing circuits. In *Proceedings of the 25th Edition on Great Lakes Symposium on VLSI*, GLSVLSI '15, 331–336, (ACM, New York, NY, USA, 2015). ACM.
51. Peterson, J. L. Petri Nets. *ACM Computing Surveys* **9**, 223–252 (1977).
52. Angluin, D., Aspnes, J., Diamadi, Z., Fischer, M. J. & Peralta, R. Computation in networks of passively mobile finite-state sensors. *Distrib. Comput.* **18**, 235–253 (2006).
53. Blum, L., Shub, M. & Smale, S. On a theory of computation over the real numbers; NP completeness, recursive functions and universal machines. In *Proceedings of the 29th Annual Symposium on Foundations of Computer Science*, 387–397 (1988).
54. Church, A. An unsolvable problem of elementary number theory. *Am. J. Math.* **58**, 345–363 (1936).
55. Bekenstein, J. D. Universal upper bound on the entropy-to-energy ratio for bounded systems. *Phys. Rev. D.* **23**, 287–298 (1981).
56. Friedland, A. E. et al. Synthetic gene networks that count. *Science* **324**, 1199–1202 (2009).
57. Lou, C. et al. Synthesizing a novel genetic sequential logic circuit: a push-on push-off switch. *Mol. Syst. Biol.* **6**, 350 (2010).
58. Oishi, K. & Klavins, E. Framework for engineering finite state machines in gene regulatory networks. *ACS Synth. Biol.* **3**, 652–665 (2014).
59. Soloveichik, D., Cook, M., Winfree, E. & Bruck, J. Computation with finite stochastic chemical reaction networks. *Nat. Comput.* **7**, 615–633 (2008).
60. Lambert, G. & Kussell, E. Memory and fitness optimization of bacteria under fluctuating environments. *PLoS Genet.* **10**, e1004556 (2014).
61. Hoffer, S. M., Westerhoff, H. V., Hellingwerf, K. J., Postma, P. W. & Tommassen, J. Autoamplification of a two-component regulatory system results in "learning" behavior. *J. Bacteriol.* **183**, 4914–4917 (2001).
62. Vladimirov, N. & Sourjik, V. Chemotaxis: How bacteria use memory. *Biol. Chem.* **390**, 1097–1104 (2009).
63. Motwani, R. & Raghavan, P. *Randomized Algorithms* (Cambridge University Press, 1995).
64. Gillespie, D. T. A rigorous derivation of the chemical master equation. *Phys. A: Stat. Mech. its Appl.* **188**, 404–425 (1992).
65. Gillespie, D. T. Exact stochastic simulation of coupled chemical reactions. *J. Phys. Chem.* **81**, 2340–2361 (1977).
66. Alaghi, A. & Hayes, J. P. Survey of stochastic computing. *ACM Trans. Embedded Comput. Syst.* **12**, 92:1–92:19 (2013).
67. Ana Solopova, J. et al. Bet-hedging during bacterial diauxic shift. *Proc. Natl Acad. Sci. USA* **111**, 7427–7432 (2014).
68. Mona, K. et al. A Oyarzún. Stochastic modelling reveals mechanisms of metabolic heterogeneity. *Commun. Biol.* **2**, 108 (2019).
69. Eldar, A. & Elowitz, M. B. Functional roles for noise in genetic circuits. *Nature* **467**, 167–173 (2010).
70. Marais, A. et al. The future of quantum biology. *J. R. Soc. Interface* **15**, 20180640 (2018).
71. Lamport, L. Time, clocks, and the ordering of events in a distributed system. *Commun. ACM* **21**, 558–565 (1978).
72. Ciocchetta, F. & Hillston, J. Bio-PEPA: A framework for the modelling and analysis of biological systems. *Theor. Computer Sci.* **410**, 3065–3084 (2009).
73. Scialdone, A. et al. *Arabidopsis* plants perform arithmetic division to prevent starvation at night. *eLife* **2**, e00669 (2013).
74. Sarpeshkar, R. Analog synthetic biology. *Philosophical transactions. Series A, Mathematical, physical, and Engineering sciences*, 372, 2014.
75. Daniel, R., Rubens, J. R., Sarpeshkar, R. & Lu, T. K. Synthetic analog computation in living cells. *Nature* **497**, 619–623 (2013). ***Robust demonstration of the power of non-digital representations in synthetic biology***.
76. Heisenberg, W. Über den anschaulichen Inhalt der quantentheoretischen Kinematik und Mechanik. *Z. f.ür. Phys.* **43**, 172–198 (1927).
77. Sarpeshkar, R. Analog versus digital: Extrapolating from electronics to neurobiology. *Neural Comput.* **10**, 1601–1638 (1998).
78. Woo, S. S., Kim, J. & Sarpeshkar, R. A digitally programmable cytomorphic chip for simulation of arbitrary biochemical reaction networks. *IEEE Trans. Biomed. Circuits Syst.* **12**, 360–378 (2018).
79. Goñi-Moreno, A. & Nikel, P. I. High-performance biocomputing in synthetic biology—integrated transcriptional and metabolic circuits. *Front. Bioeng. Biotechnol.* **7**, 40 (2019).
80. Oyarzún, D. A. & Stan, G.-B. V. Synthetic gene circuits for metabolic control: Design trade-offs and constraints. *J. R. Soc. Interface*, **10**, 20120671 (2013).
81. Chavarría, M., Goñi-Moreno, Á., de Lorenzo, V. & Nikel, P. I. A metabolic widget adjusts the phosphoenolpyruvate-dependent fructose influx in *Pseudomonas putida*. *mSystems* **1**, e00154–16 (2016).
82. Pandi, A. et al. Metabolic perceptrons for neural computing in biological systems. *Nat. Commun.* **10**, 1–13 (2019).
83. Zhang, F., Carothers, J. M. & Keasling, J. D. Design of a dynamic sensor-regulator system for production of chemicals and fuels derived from fatty acids. *Nat. Biotechnol.* **30**, 354–9 (2012).
84. Xu, P., Li, L., Zhang, F., Stephanopoulos, G. & Koffas, M. Improving fatty acids production by engineering dynamic pathway regulation and metabolic control. *Proc. Natl Acad. Sci. USA* **111**, 11299–11304 (2014).
85. Liu, D., Mannan, A. A., Han, Y., Oyarzún, D. A. & Zhang, F. Dynamic metabolic control: Towards precision engineering of metabolism. *J. Ind. Microbiol. Biotechnol.* **45**, 535–543 (2018).

86. Delépine, B., Libis, V., Carbonell, P. & Faulon, J. -L. SensiPath: Computer-aided design of sensing-enabling metabolic pathways. *Nucleic Acids Res.* **44** (W1), W226–31 (2016).
87. Delépine, B., Duigou, T., Carbonell, P. & Faulon, J. -L. RetroPath2.0: a retrosynthesis workflow for metabolic engineers. *Metab. Eng.* **45**, 158–170 (2018).
88. Lin, G.-M., Warden-Rothman, R. & Voigt, C. A. Retrosynthetic design of metabolic pathways to chemicals not found in nature. *Curr. Opin. Syst. Biol.* **14**, 82–107 (2019).
89. Segler, M. H. S., Preuss, M. & Waller, M. P. Planning chemical syntheses with deep neural networks and symbolic AI. *Nature* **555**, 604–610 (2018).
90. Liu, B. et al. Retrosynthetic reaction prediction using neural sequence-to-sequence models. *ACS Central. Science* **3**, 1103–1113 (2017).
91. Kotte, O., Volkmer, B., Radzikowski, J. L. & Heinemann, M. Phenotypic bistability in *Escherichia coli*'s central carbon metabolism. *Mol. Syst. Biol.* **10**, 736 (2014).
92. Nikolados, E.-M., Weiße, A. Y., Ceroni, F. & Oyarzún, D. A. Growth defects and loss-of-function in synthetic gene circuits. *ACS Synth. Biol.* **8**, 1231–1240 (2019).
93. Cardinale, S. & Arkin, A. P. Contextualizing context for synthetic biology - identifying causes of failure of synthetic biological systems. *Biotechnol. J.* **7**, 856–866 (2012).
94. Ceroni, F., Algar, R., Stan, G.-B. & Ellis, T. Quantifying cellular capacity identifies gene expression designs with reduced burden. *Nat. Methods* **12**, 415–418 (2015).
95. Gyorgy, A. et al. Isocost lines describe the cellular economy of genetic circuits. *Biophysical J.* **109**, 639–646 (2015).
96. Gorochowski, T. E., Avcilar-Kucukgoze, I., Bovenberg, R. A. L., Roubos, J. A. & Ignatova, Z. A minimal model of ribosome allocation dynamics captures trade-offs in expression between endogenous and synthetic genes. *ACS Synth. Biol.* **5**, 710–720 (2016).
97. Weiße, A. Y., Oyarzún, D. A., Danos, V. & Swain, P. S. Mechanistic links between cellular trade-offs, gene expression, and growth. *Proc. Natl Acad. Sci. USA* **112**, E1038–E1047 (2015).
98. Borkowski, O. et al. Cell-free prediction of protein expression costs for growing cells. *Nat. Commu. 9*, 1457 (2018).
99. Francesca Ceroni, A. et al. Burden-driven feedback control of gene expression. *Nat. Methods* **15**, 387–393 (2018).
100. Ceroni, F., Blount, B. A. & Ellis, T. Sensing the right time to be productive. *Cell Syst.* **3**, 116–117 (2016).
101. Macia, J., Vidiella, B. & Solé, R. V. Synthetic associative learning in engineered multicellular consortia. *J. R. Soc. Interface* **14**, 20170158 (2017).
102. Sardanyés, J., Bonforti, A., Conde, N., Solé, R. & Macia, J. Computational implementation of a tunable multicellular memory circuit for engineered eukaryotic consortia. *Front. Physiol.* **6**, 281 (2015).
103. Macia, J. & Sole, R. How to make a synthetic multicellular computer. *PLoS One* **9**, 1–13 (2014).
104. Goñi-Moreno, A., Redondo-Nieto, M., Arroyo, F. & Castellanos, J. Biocircuit design through engineering bacterial logic gates. *Nat. Comput.* **10**, 119–127 (2011).
105. Goñi-Moreno, A., Amos, M. & de la Cruz, F. Multicellular computing using conjugation for wiring. *PLoS One* **8**, e65986 (2013).
106. Chen, Y., Kim, J. K., Hirning, A. J., Josic, K. & Bennett, M. R. Emergent genetic oscillations in a synthetic microbial consortium. *Science* **349**, 986–989 (2015).
107. Fiore, G. et al. In-silico analysis and implementation of a multicellular feedback control strategy in a synthetic bacterial consortium. *ACS Synth. Biol.* **6**, 507–517 (2017).
108. Urrios, A. et al. A synthetic multicellular memory device. *ACS Synth. Biol.* **5**, 862–873 (2016).
109. Regot, S. et al. Distributed biological computation with multicellular engineered networks. *Nature* **469**, 207 (2011).
110. Danino, T., Mondragón-Palomino, O., Tsimring, L. & Hasty, J. A synchronized quorum of genetic clocks. *Nature* **463**, 326–330 (2010).
111. Tabor, J. J. et al. A synthetic genetic edge detection program. *Cell* **137**, 1272–1281 (2009).
112. Tamsir, A., Tabor, J. J. & Voigt, C. A. Robust multicellular computing using genetically encoded NOR gates and chemical 'wires'. *Nature* **469**, 212–215 (2011).
113. Gorochowski, T. E. Agent-based modelling in synthetic biology. *Essays Biochem.* **60**, 325–336 (2016).
114. Rudge, T. J., Steiner, P. J., Phillips, A. & Haseloff, J. Computational modeling of synthetic microbial biofilms. *ACS Synthetic. Biology* **1**, 345–352 (2012).
115. Jang, S. S., Oishi, K. T., Egbert, R. G. & Klavins, E. Specification and simulation of synthetic multicelled behaviors. *ACS Synthetic. Biology* **1**, 365–374 (2012).
116. Gorochowski, T. E. et al. BSim: An agent-based tool for modeling bacterial populations in systems and synthetic biology. *PLoS ONE* **7**, 1–9 (2012).
117. Goni-Moreno, A. & Amos, M. DiSCUS: A simulation platform for conjugation computing. In *International Conference on Unconventional Computation and Natural Computation*, 181–191 (Springer, 2015).
118. Naylor, J. et al. Simbiotics: A multiscale integrative platform for 3D modeling of bacterial populations. *ACS Synth. Biol.* **6**, 1194–1210 (2017).
119. Montagna, S. & Viroli, M. A computational framework for modelling multicellular biochemistry. In *2009 IEEE Congress on Evolutionary Computation*, 2233–2240 (2009).
120. Kang, S., Kahan, S., McDermott, J., Flann, N. & Shmulevich, I. Biocellion: accelerating computer simulation of multicellular biological system models. *Bioinformatics* **30**, 3101–3108 (2014).
121. Kylilis, N., Tuza, Z. A., Stan, G.-B. & Polizzi, K. M. Tools for engineering coordinated system behaviour in synthetic microbial consortia. *Nat. Commun.* **9**, 2677 (2018).
122. Ji, W. et al. A formalized design process for bacterial consortia that perform logic computing. *PLoS One* **8**, 1–9 (2013).
123. McCarty, N. S. & Ledesma-Amaro, R. Synthetic biology tools to engineer microbial communities for biotechnology. *Trends Biotechnol.* **37**, 181–197 (2019).
124. Scott, S. R. & Hasty, J. Quorum sensing communication modules for microbial consortia. *ACS Synth. Biol.* **5**, 969–977 (2016).
125. Amos, M. Population-based microbial computing: a third wave of synthetic biology? *Int. J. Gen. Syst.* **43**, 770–782 (2014).
126. Brenner, K., You, L. & Arnold, F. H. Engineering microbial consortia: a new frontier in synthetic biology. *Trends Biotechnol.* **26**, 483–489 (2008). ***Foundational discussion of how we might harness the power of multicellular consortia for the purposes of production and computation***.
127. Nielsen, A. A. K. et al. Genetic circuit design automation. *Science,* **352**, aac7341–aac7341 (2016).
128. Cai, L., Dalal, C. K. & Elowitz, M. B. Frequency-modulated nuclear localization bursts coordinate gene regulation. *Nature* **455**, 485–490 (2008).
129. Salek, M. M., Carrara, F., Fernandez, V., Guasto, J. S. & Stocker, R. Bacterial chemotaxis in a microfluidic T-maze reveals strong phenotypic heterogeneity in chemotactic sensitivity. *Nat. Commun.* **10**, 1877 (2019).
130. Goñi-Moreno, Á., Benedetti, I., Kim, J. & de Lorenzo, V. Deconvolution of gene expression noise into spatial dynamics of transcription factor– promoter interplay. *ACS Synth. Biol.* **6**, 1359–1369 (2017).
131. García-Betancur, J.-C. et al. Cell differentiation defines acute and chronic infection cell types in *Staphylococcus aureus*. *Elife* **6**, e28023 (2017).
132. Lu, T. K., Khalil, A. S. & Collins, J. J. Next-generation synthetic gene networks. *Nat. Biotechnol.* **27**, 1139–1150 (2009).
133. Tomazou, M. & Stan, G.-B. Portable gene expression guaranteed. *Nat. Biotechnol.* **36**, 313 (2018).
134. Aoki, S. K., Lillacci, G., Gupta, A., Baumschlager, A. & Schweingruber, D. A universal biomolecular integral feedback controller for robust perfect adaptation. *Nature* **570**, 533–537 (2019).
135. Goñi-Moreno, A. & Amos, M. A reconfigurable NAND/NOR genetic logic gate. *BMC Syst. Biol.* **6**, 126 (2012).
136. Eiben, A. E. et al. *Introduction to Evolutionary Computing*, vol 53 (Springer, 2003).
137. Esvelt, K. M., Carlson, J. C. & Liu, D. R. A system for the continuous directed evolution of biomolecules. *Nature* **472**, 499–503 (2011).
138. Zhao, H., Giver, L., Shao, Z., Affholter, J. A. & Arnold, F. H. Molecular evolution by staggered extension process (StEP) in vitro recombination. *Nat. Biotechnol.* **16**, 258–261 (1998).
139. Brödel, A. K., Jaramillo, A. & Isalan, M. Engineering orthogonal dual transcription factors for multi-input synthetic promoters. *Nat. Commun.* **7**, 13858 (2016).
140. Yokobayashi, Y., Weiss, R. & Arnold, F. H. Directed evolution of a genetic circuit. *Proc. Natl. Acad. Sci. USA* **99**, 16587–16591 (2002).
141. de Lorenzo, V. From the selfish gene to selfish metabolism: revisiting the central dogma. *Bioessays* **36**, 226–235 (2014).
142. Chait, R., Ruess, J., Bergmiller, T., Tkacik, G. & Guet, C. C. Shaping bacterial population behavior through computer-interfaced control of individual cells. *Nat. Commun.* **8**, 1–11 (2017).
143. Cobb, R. E., Sun, N. & Zhao, H. Directed evolution as a powerful synthetic biology tool. *Methods*, **60**, 81–90, 2013.
144. Goldman, N. et al. Towards practical, high-capacity, low-maintenance information storage in synthesized DNA. *Nature* **494**, 77 (2013).
145. Dvorák, P., Nikel, P. I., Damborsky, J. & de Lorenzo, V. Bioremediation 3.0: engineering pollutant-removing bacteria in the times of systemic biology. *Biotechnol. Adv.* **35**, 845–866 (2017).
146. TerAvest, M. A., Li, Z. & Angenent, L. T. Bacteria-based biocomputing with cellular computing circuits to sense, decide, signal, and act. *Energy Environ. Sci.* **4**, 4907–4916 (2011).
147. Chen, Y. Y. & Smolke, C. D. From DNA to targeted therapeutics: bringing synthetic biology to the clinic. *Sci. Transl. Med.* **3**, 106ps42–106ps42 (2011).

148. Paton, R. C. Some computational models at the cellular level. *BioSystems* **29**, 63–75 (1993).
149. Niederholtmeyer, H. et al. Rapid cell-free forward engineering of novel genetic ring oscillators. *eLife* **4**, e09771 (2015).
150. Swank, Z., Laohakunakorn, N. & Maerkl, S. J. Cell-free gene-regulatory network engineering with synthetic transcription factors. *Proc. Natl. Acad. Sci. USA* **116**, 5892–5901 (2019).
151. Lehr, F. X. et al. Cell-free prototyping of AND-logic gates based on heterogeneous RNA activators. *ACS Synth. Biol.* **8**, 2163–2173, 2019.
152. Lambert, N. et al. Quantum biology. *Nat. Phys.* **9**, 10–18 (2013).
153. Adriana, M. et al. The future of quantum biology. *J. R. Soc. Interface* **15**, 20180640 (2018).
154. Engel, G. S. et al. Evidence for wavelike energy transfer through quantum coherence in photosynthetic systems. *Nature* **446**, 782–786 (2007).
155. R. K. Allemann & Scrutton, N.S. *Quantum Tunnelling in Enzyme-Catalysed Reactions* (Royal Society of Chemistry, 2009).
156. David, D. & Roger, P. Quantum theory, the Church–Turing principle and the universal quantum computer. *Proc. R. Soc. Lond. A. Math. Phys. Sci.* **400**, 97–117 (1985).
157. Scheres, B. & Van Der Putten, W. H. The plant perceptron connects environment to development. *Nature* **543**, 337 (2017).
158. DeMarseT.B. & Dockendorf, K. P. Adaptive flight control with living neuronal networks on microelectrode arrays. In *Proceedings of the IEEE International Joint Conference on Neural Networks*, vol 3, 1548–1551 (IEEE, 2005).
159. Warwick, K., Nasuto, S. J., Becerra, V. M. & Whalley, B. J. Experiments with an in-vitro robot brain. In *Computing with Instinct*, 1–15 (Springer, 2011).
160. Adamatzky, A. *Advances in Physarum machines: Sensing and Computing with Slime Mould*, vol 21 (Springer, 2016).
161. Tschirhart, T. et al. Electronic control of gene expression and cell behaviour in *Escherichia coli* through redox signalling. *Nat. Commun.* **8**, 14030 (2017).
162. Abelson, H. et al. Amorphous computing. *Commun. ACM* **43**, 74–82 (2000).
163. Gordana D.-C. The info-computational nature of morphological computing (ed. Müller, V. C.), *Philosophy and Theory of Artificial Intelligence*, Studies in Applied Philosophy, Epistemology and Rational Ethics, 59–68 (Springer Berlin Heidelberg, Berlin, Heidelberg, 2013).
164. Umedachi, T., Takeda, K., Nakagaki, T., Kobayashi, R. & Ishiguro, A. Fully decentralized control of a soft-bodied robot inspired by true slime mold. *Biol. Cybern.* **102**, 261–269 (2010).
165. Solé, R. Bioengineering the biosphere? *Ecol. Complex.* **22**, 40–49 (2015). ***Discussion of a potentially high-impact application area of synthetic biology, of which engineered cellular information processing is a fundamental component***.
166. Armstrong, R. Systems architecture: A new model for sustainability and the built environment using nanotechnology, biotechnology, information technology, and cognitive science with living technology. *Artif. Life* **16**, 73–87 (2010).

Acknowledgements

A.G.-M. was supported by the SynBio3D project of the UK Engineering and Physical Sciences Research Council (EP/R019002/1) and the European CSA on biological standardization BIOROBOOST (EU grant number 820699). T.E.G. was supported by a Royal Society University Research Fellowship (grant UF160357) and BrisSynBio, a BBSRC/EPSRC Synthetic Biology Research Centre (grant BB/L01386X/1). P.Z. was supported by the EPSRC Portabolomics project (grant EP/N031962/1). P.C. was supported by SynBioChem, a BBSRC/EPSRC Centre for Synthetic Biology of Fine and Specialty Chemicals (grant BB/M017702/1) and the ShikiFactory100 project of the European Union's Horizon 2020 research and innovation programme under grant agreement 814408.

Author contributions

L.G., M.A. and A. G.-M. conceived the study. L.G., M.A., T.E.G., P.C., D.A.O., R.S., H.F., P.Z., H.T. and A.G.-M. contributed to the writing and editing of the paper.

Competing interests

The authors declare no competing interests.

Additional information

Correspondence and requests for materials should be addressed to A.Gñi-M.

Peer review information *Nature Communications* thanks Guy-Bart Stan and the other, anonymous, reviewer(s) for their contribution to the peer review of this work.

Reprints and permission information is available at http://www.nature.com/reprints

Publisher's note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.

Open Access This article is licensed under a Creative Commons Attribution 4.0 International License, which permits use, sharing, adaptation, distribution and reproduction in any medium or format, as long as you give appropriate credit to the original author(s) and the source, provide a link to the Creative Commons license, and indicate if changes were made. The images or other third party material in this article are included in the article's Creative Commons license, unless indicated otherwise in a credit line to the material. If material is not included in the article's Creative Commons license and your intended use is not permitted by statutory regulation or exceeds the permitted use, you will need to obtain permission directly from the copyright holder. To view a copy of this license, visit http://creativecommons.org/licenses/by/4.0/.

© The Author(s) 2019