IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 6 March 2024, accepted 7 April 2024, date of publication 22 April 2024, date of current version 29 April 2024.

Digital Object Identifier 10.1109/ACCESS.2024.3391511

RESEARCH ARTICLE

CMOS Analog Simulators of Dynamical Systems

JOÃO C. G. ARAÚJO[1,2], LUIS B. OLIVEIRA[1,2], (Senior Member, IEEE), BRUNO J. GUERREIRO[1,2,3], (Member, IEEE), AND FILIPE FERREIRA DA SILVA[4]

[1]LASI, CTS-UNINOVA, NOVA School of Science and Technology, 2829-516 Caparica, Portugal
[2]Department of Electrical and Computer Engineering, NOVA School of Science and Technology, 2829-516 Caparica, Portugal
[3]Institute for Systems and Robotics, LARSyS, 1049-001 Lisbon, Portugal
[4]CEFITEC, Department of Physics, NOVA School of Science and Technology, 2829-516 Caparica, Portugal

Corresponding author: João C. G. Araújo (jc.araujo@campus.fct.unl.pt)

This work was supported by Fundação para a Ciência e a Tecnologia (FCT) under Grant 10.54499/UIDB/00066/2020, Grant 10.54499/PTDC/EEI-AUT/1732/2020, and Grant 10.54499/UIDB/00068/2020. The work of João C. G. Araújo was supported by FCT through the Ph.D. Studentship under Grant 2022.13724.BD.

ABSTRACT Analog computing is based upon using physical processes to solve formal mathematical problems. In the past, it was the predominant instrument of scientific calculations. Now, as the physical limits imposed on digital devices compel research into alternate computing paradigms, a reexamination of the potentialities of analog computing is warranted. This work studies the application of analog CMOS cells toward the simulation of dynamical systems, and, more generally, solving sets of coupled time-dependent ordinary differential equations. Following a brief review of the fundamentals of systems theory and analog computing, the main set of computing elements is introduced, each comprising analog cells designed in a 130 nm process. These are subsequently applied to the realization of practical, special-purpose analog computing modules. Illustrative systems from various fields are selected for simulation. Though by no means comprehensive, these case studies highlight the capabilities of contemporary analog computing, especially in solving nonlinear problems. Circuit simulations show good agreement with solutions obtained from high-order numerical methods, at least over a limited range of system parameters. The article concludes with a brief discussion of broader analog computing applications, offering future prospects toward further exploration of its potentialities and limitations in a wide range of domains.

INDEX TERMS Analog computers, differential equations, dynamical systems, integrated circuits.

I. INTRODUCTION

Dynamical systems compose the set of mathematical tools with which we model and analyze the world around us, and are widely found in the natural sciences and the engineering disciplines [1], [2]. These systems are commonly represented in the form of time-dependent ordinary differential equations (ODEs), the solution to which provides a description of the system's evolution through time. Generally, higher order and/or nonlinear systems defy analytical treatment, and their solution relies on application of numerical procedures, such as the Euler or Runge-Kutta methods [1], [3]. The discrete, stepwise nature of these algorithms makes them well suited for implementation in digital computers, especially given the performance of modern machines. However, these computations still present considerable workloads, demanding resources which are often not available in the current trend toward low-power, high-efficiency embedded systems. Further development of digital technology is also increasingly hampered by the physical limits imposed on transistor density and clock speed [4], [5], [6]. These issues motivate the pursuit of alternative computation paradigms. For the purposes of scientific and engineering calculations, a shift back into the analog domain has not only proven to be worthwhile, but also provided researchers with a whole new set of tools with which to deal with otherwise intractable problems.

An analog computer (AC) maps a mathematical problem onto a set of physical quantities, thereby exploiting natural processes in order to reach a solution [2]; since this procedure doesn't rely on traditional, sequential computation, the solution is obtained in real-time. Many physical representations

The associate editor coordinating the review of this manuscript and approving it for publication was Harikrishnan Ramiah.

are suitable for effective analog computation, and over the course of their history, ACs developed from mechanical, to electrical and eventually electronic apparatuses [7]. Unlike digital computers, which are mainly geared toward data manipulation, ACs are most suited to scientific computation, namely solving sets of coupled time-dependent ODEs; as explained previously, this is closely related to the problem of predicting the behavior of dynamical systems. The ubiquity of these problems in all areas of science and engineering gives a sense of the wide applicability of analog computing techniques. Though far too numerous to exhaustively list, some of the fields in which ACs have found extensive application include mechanics (both classical and quantum), biology, computer science and applied mathematics, as well as mechanical and electrical engineering [7], [8], [9], [10], [11]. ACs excel in solving/simulating nonlinear systems, which may be insurmountable using analytical or numerical methods; they can also accommodate constrained variables, as well as hybrid systems comprising both continuous and logical variables. By coupling two or more AC modules, interaction between multiple systems can also be implemented, allowing for real-time simulation of closed-loop dynamics.

Starting from the late 20th century, the booming microprocessor industry pushed ACs into the background, where they have largely remained. However, the increasing availability of mainstream, low-cost microelectronics fabrication processes has enabled a new generation of integrated analog computing devices. These most often take the form of special-purpose analog processors aimed toward solving a particular class of problems [5], [6], [12], although more general-purpose architectures, like the field-programmable analog array, have also been explored [13], [14], [15], [16], [17], [18].

The following work presents an overview of the general principles of AC operation and design in CMOS technology, aimed toward the development of practical simulators for (nonlinear) dynamical systems. While it is impossible to achieve a complete account of this broad topic in such a brief format, some typical techniques and implementations have been selected in an attempt to showcase the capabilities, and limitations, of AC technology. Furthermore, since all computing elements are realized in standard analog cells, the provided case studies can be readily generalized to suit a wide variety of problems, in whatever field may be of interest to the reader.

The remainder of the paper is structured as follows: Section II lays down the minimal theoretical groundwork in systems theory and analog computing necessary for understanding the subsequent discussion; Section III presents CMOS realizations of the principal computing elements required for analog computation; in Section IV, practical concerns and limitations of electronic analog solvers are considered; Section V showcases four analog solvers, each dedicated to solving a different family of problems; Section VI concludes the article with final remarks and future prospects. All circuits were designed in the UMC 130 nm process (1.2 V supply) within Cadence Design Systems' Virtuoso environment, and simulations were conducted with the Spectre electrical simulator using foundry-supplied device models.

II. BACKGROUND

Understanding the scope and applicability of CMOS analog solvers warrants a review of some underlying concepts. Within this framework, the current Section first outlines the basic theory of dynamical systems, insofar as it is relevant to the remainder of the paper. The focus is placed exclusively on continuous-time systems, for two reasons: 1) it is the most natural, and intuitive, description of natural phenomena; 2) discrete-time formulations are mainly useful for computations in a digital environment, and are unnecessary for analog implementations. Following this is a brief discussion of the operating principles of ACs, their history and how they differ from today's digital processors.

A. DYNAMICAL SYSTEMS

A dynamical system is characterized by an internal state whose value changes over time, either autonomously or in response to external input [19]. For an n^{th}-order system with m inputs and l outputs, the state, input and output vectors are respectively defined as

$$\mathbf{x}(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ \vdots \\ x_n(t) \end{bmatrix}, \quad \mathbf{u}(t) = \begin{bmatrix} u_1(t) \\ u_2(t) \\ \vdots \\ u_m(t) \end{bmatrix}, \quad \mathbf{y}(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ \vdots \\ y_l(t) \end{bmatrix}. \tag{1}$$

In its most general formulation, the system's evolution is a function of $\mathbf{x}$ and $\mathbf{u}$, as well as time t,

$$\dot{\mathbf{x}}(t) = \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t) \tag{2}$$

where $\mathbf{f}$ is known as the state function, and the overdot denotes the time derivative of $\mathbf{x}$. The system's output may likewise be described in terms of an output function $\mathbf{h}$,

$$\mathbf{y}(t) = \mathbf{h}(\mathbf{x}(t), \mathbf{u}(t), t). \tag{3}$$

Together, (2) and (3) form the system's state-space model. This formulation models dynamics with a first-order differential equation, though systems involving higher-order derivatives can still be converted into this form by appropriate choice of $\mathbf{x}$, $\mathbf{f}$ and $\mathbf{h}$.

It's useful to represent the system described by (2) and (3) in the form of a block diagram. To do so, we will first integrate both sides of (2) with respect to time

$$\mathbf{x}(t) = \int_{t_0}^{t} \mathbf{f}(\mathbf{x}(t), \mathbf{u}(t), t)\, dt + \mathbf{x}_0 \tag{4}$$

where $\mathbf{x}_0 = \mathbf{x}(t_0)$ is the initial state condition. With (3) and (4), we may construct the diagram of Fig. 1; this is an input/output representation of the system's behavior, i.e., it completely abstracts the system's inner workings. This

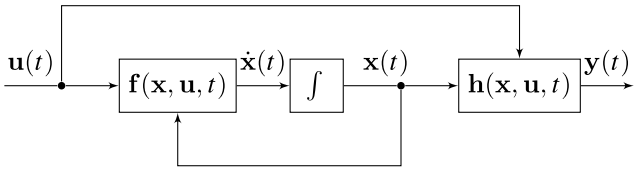


FIGURE 1. Generic block diagram of a dynamical system.

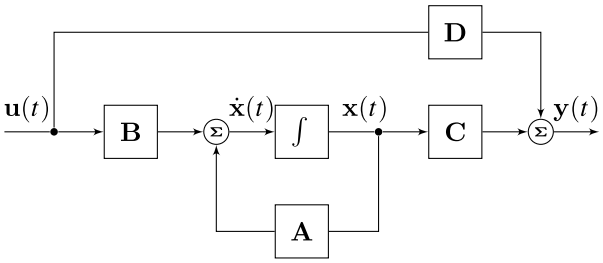


FIGURE 2. Block diagram of an LTI system.

abstraction can be quite useful, as it facilitates mapping the system's dynamics onto some physical medium, such as electronic circuits, which may be completely distinct from the system's original depiction (for instance, think of representing the position of a swinging pendulum by the voltage of an *LC* oscillator).

The model of a dynamical system provides a mathematical description of the system behavior and properties (e.g. stability, observability, controllability, modal decomposition, etc), as well as the means to predict its future behavior based on the current state and external inputs. Modeling real-world systems requires finding explicit expressions for the state and output functions which accurately capture the system's dynamics, or a suitable portion thereof; this is usually accomplished either by fitting experimental data and/or through physical principles.

For the particular case in which functions $\mathbf{f}$ and $\mathbf{h}$ are linear with $\mathbf{x}$ and $\mathbf{u}$, and do not depend explicitly on time, the state-space model reduces to

$$\dot{\mathbf{x}}(t) = \mathbf{A}\mathbf{x}(t) + \mathbf{B}\mathbf{u}(t), \quad \mathbf{y}(t) = \mathbf{C}\mathbf{x}(t) + \mathbf{D}\mathbf{u}(t) \tag{5}$$

where $\mathbf{A} \in \mathbb{R}^{n\times n}$, $\mathbf{B} \in \mathbb{R}^{n\times m}$, $\mathbf{C} \in \mathbb{R}^{l\times n}$ and $\mathbf{D} \in \mathbb{R}^{l\times m}$ are constant matrices. The model in (5) describes a linear time-invariant (LTI) system [20]. Similarly to the general case, the system's block diagram can be obtained through integration of the state equation in (5), and is illustrated in Fig. 2. The exact solution to the state equation of an LTI system is given by [19]

$$\mathbf{x}(t) = \int_{t_0}^{t} e^{\mathbf{A}(t-\tau)}\mathbf{B}\mathbf{u}(\tau)\,d\tau + e^{\mathbf{A}t}\mathbf{x}_0 \tag{6}$$

where τ is the convolution dummy variable. Many systems in engineering and industrial applications can be represented by LTI models, at least as a first approximation. Because of this, LTI systems have been the subject of in-depth study for many decades, and a wide range of tools have been developed for their analysis [20], [21].

Generally, however, functions $\mathbf{f}$ and $\mathbf{h}$ are nonlinear, i.e., they include terms which are themselves nonlinear functions of $\mathbf{x}$, $\mathbf{u}$, and/or t, with some common examples being products, polynomials and trigonometric functions [20]. It is generally impossible to obtain analytical solutions to these systems, which greatly complicates the analysis of their behavior and properties.

In order to tap into the body of knowledge concerning LTI systems, the study of nonlinear systems often begins with some attempt at linearization. The most common strategy starts by identifying the system's equilibrium points (i.e. states in which the system will remain unless acted upon), and assuming that, for sufficiently small variations around these points, the system behaves linearly [19]. While this method can provide insight into the system's local behavior, it fails to capture its global evolution, and certainly cannot account for intrinsically nonlinear phenomena, e.g. multiple modes of behavior and chaotic motion [22].

Another issue lies in the amount of computational resources required for accurate numerical simulations of these nonlinear properties. Though advances in numerical methods and the increasing availability of sophisticated computing platforms have made these simulations more viable, they are still far too demanding for many resource-constrained, low-power applications [23]. In this regard, *analog* simulations warrant further exploration, since they can replicate nonlinear behavior in real-time, often with minimal power requirements.

B. ANALOG COMPUTERS

Whereas digital computers represent data symbolically, ACs directly map a problem's variables onto continuous physical quantities, such that these are subject to the same restrictions that define the problem [7]. In other words, the AC exploits physical processes to create simulacra of abstract mathematical constructs.

The earliest ACs were mechanical in nature, dating back to Ancient Greece, where they were used for astronomical and geometrical calculations [7]. The 19$^{\text{th}}$ century saw the development of mechanical devices capable of performing calculus, and even Fourier analysis [24], [25]. In the early 20$^{\text{th}}$ century came the first electronic ACs, though they were quite limited, being developed solely with passive elements (essentially *RLC* networks) [7]. Much more sophisticated, active designs arose with the development of the vacuum tube-based operational amplifier (op-amp), which allowed for electronic implementation of the most common mathematical operations, including integration and differentiation [2], [8], [10], [26]. These computers found practical application in artillery guidance systems during World War II, and later integrated the navigational computers of the Apollo space program [8], [27]. Concurrently, researchers were realizing the AC's potential for more general scientific and engineering calculations, namely in the study of differential

equations and dynamical systems [10], [28]. Eventually, the development of increasingly sophisticated digital machines would mean the supersedure of ACs, which were mostly considered obsolete by the 1970s [7], [29], [30]. Partly to blame is the often highly technical and specialized nature of the AC: even in later ''transistorized'' iterations, these devices were quirky, unwieldy and difficult to operate. The so-called ''patch-programming'' methods involved turning knobs and switching wires, necessitating knowledgeable personnel [17], [30].

In the intervening decades, however, research into analog computing would persevere, motivated by the development of analog very-large scale integration (VLSI) technology and the increasing interest in neural networks and neuromorphic computation [31], [32], [33], [34]. Today, the availability of low cost, scalable CMOS technology has rekindled interest in analog computation. Recent works have explored the solution/simulation of nonlinear dynamical systems using integrated analog circuitry, often coupled with digital interfaces for configuration and data sampling purposes [6], [35], [36], [37]. Results are promising, with special-purpose solvers displaying efficiency metrics far superior to those of modern digital computers, though often at the cost of lower precision [17], [18], [29], [37].

At their apex, general-purpose ACs would comprise a large set of functional blocks, namely integrators, adders, amplifiers and multipliers, which could be interconnected and configured to suit a plethora of problems [7], [38]. As evidenced throughout its history, the AC naturally lends itself to scientific computation, namely to solving ODEs, with time as the independent variable. As seen from Subsection II-A, by appropriate choice of the dependent variables, this class of problems can be reformulated as the description of dynamical systems. This interpretation is useful, as it allow us to use the language and tools of systems theory to replicate the differential equation's dynamics in another medium. In particular, block diagram representations (such as the ones illustrated in Fig. 1 and 2) can act as guides for designing/programming an AC, by simply mapping each of the diagram's blocks onto their analog equivalents. In this way, the AC ''simulates'' the dynamical system by mapping its variables/processes onto a set of physical quantities; in the case of electronic ACs, these usually consist of voltage and/or current.

Compared to numerical (digital) simulations, ACs do not rely on discrete approximations of the system's model, as they intrinsically operate with continuous variables [2]. Instead, the main source of error in an analog simulation is in the implementation of each functional block: practical realizations, whether mechanical, electrical or electronic, always carry limited precision, accuracy and operating range. Hence, minimizing the overall error is often a matter of tuning the design of each component, but also of properly conditioning the problem to the specific capabilities of the available hardware [7].

The number of functional blocks required for solving a given problem is proportional to the number of dependent variables, the maximum derivative order and the number of nonlinear terms. In particular, integrators are placed in parallel for each state variable, and cascaded with each derivative order. Thus, apart from initial transients and propagation delays, this architecture makes the ''computation'' time of an AC approximately independent of the problem's complexity (it is instead set by the circuit's time constant, as will be seen in Section IV). Because of this, the AC is especially useful for solving/simulating nonlinear systems, which may otherwise be prohibitively expensive, or outright impossible, to solve in a digital environment. Furthermore, convergence is not an issue: if a solution to the problem exists, then the AC is guaranteed to find it [18], [39]. Of course, an AC's flexibility is ultimately limited by the variety of nonlinear blocks which constitute it. In an electronic AC, multiplier blocks are easy to achieve, and can be used to implement products, roots and polynomials; coupled with function generators (or even ''arbitrary waveform generators''), multipliers can also implement time-varying terms. Other nonlinear electronic blocks include exponential, logarithmic and trigonometric functions, though these are less common and usually suffer from limited operating range and/or temperature sensitivity [40], [41], [42], [43], [44].

Analog computing is also suited to simulating closed-loop dynamics, which are the subject of extensive study in control theory. In a typical control application, two dynamical systems interact in a feedback loop, with the first attempting to drive the second to a particular state, often set by an external reference. The former system is known as the controller, and the latter as the plant [20]. Either, or both, of these systems may be simulated through an AC, enjoying the perks described previously [8], [38]. Classical architectures, such as the proportional-integral-derivative (PID) controller, are readily implemented in an analog environment by simply connecting the corresponding function blocks to a weighted sum node, configured with the proper gain values (these have to be computed *a priori*, or at least adjusted experimentally).

III. COMPUTING ELEMENTS

The following subsections describe the CMOS realization of the fundamental blocks of an analog solver, with voltage chosen as the main computing variable. The circuitry was designed and simulated for 130 nm technology, though it is highly scalable to other technology nodes. All device sizings can be found in Appendix.

A. INTEGRATOR

The integrator is the basic building block of an analog solver, as it relates a variable to its time derivative. In analog circuits, integration is implemented by exploiting the relation between current and voltage in a capacitor, which may be expressed in

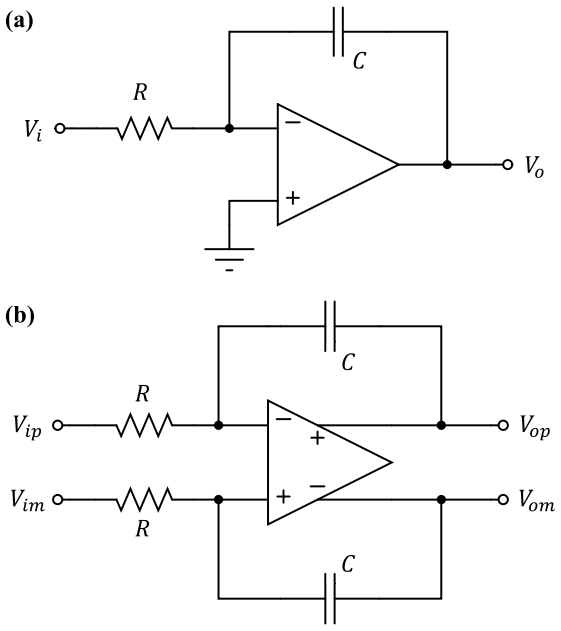


FIGURE 3. Operational amplifier realization of voltage-mode integrators. (a) Single-ended integrator op-amp configuration. (b) Fully-differential integrator topology.

the form [46]

$$V_C(t) = \frac{1}{C}\int_{t_0}^{t} I_C(t)\,dt + V_C(t_0) \tag{7}$$

where V_C and I_C respectively represent the voltage and current across a capacitance C, initially charged at $V_C(t_0)$. Though I_C may be provided by a current source, using an op-amp allows for voltage-mode integration by turning a resistor into an approximately ideal voltage-controlled current source, as in the circuit of Fig. 3(a). Assuming the amplifier has infinite gain, the virtual ground at the inverting input means I_C is simply given by the ratio between the input voltage V_i and the resistance R. Writing the output voltage as $V_o = -V_C$, (7) becomes

$$V_o(t) = -\frac{1}{RC}\int_{t_0}^{t} V_i(t)\,dt + V_o(t_0). \tag{8}$$

This principle may be extended into a fully-differential topology, as in Fig. 3(b): this configuration retains the functionality of the single-ended integrator, while offering reduced even-order harmonic distortion, greater signal-to-noise ratios and easily switchable signal polarity by cross-coupling [47]. Whether in single-ended or fully-differential configurations, the op-amp may also perform integration on multiple inputs. The virtual ground nodes act as summing junctions for the input currents. As such, the total voltage across the feedback capacitor(s) is given by the sum of the input voltages $V_n(t)$ weighted by their respective resistors R_n, such that (8) becomes

$$V_o(t) = -\frac{1}{C}\int_{t_0}^{t} \sum_n \frac{V_n(t)}{R_n}\,dt + V_o(t_0). \tag{9}$$

One problem with the integrator arrangements of Fig. 3 is that they lack stable DC operating points, as the capacitor's infinite impedance at DC effectively places the op-amp in open-loop operation; this issue is commonly mitigated by placing a large resistor across the feedback capacitor, which ensures proper biasing by turning the op-amp into an inverting amplifier at lower frequencies [46]. However, in this application, the integrator is part of a larger feedback network that guarantees a DC path between the integrator's input and output, such that no further biasing techniques are required.

TABLE 1. Op-amp performance specifications.

Parameter	Value
Open-loop DC gain	71 dB
GBW*	36 MHz
Output swing	900 mV
Phase margin*	77°
Power dissipation	570 μW

*With 10 pF load on each output.

Fast operation and high accuracy are dependent upon op-amp characteristics, namely gain-bandwidth product (GBW). High output-swing is also desirable for preventing output saturation. Among the most common CMOS op-amp architectures, the folded-cascode amplifier provides a good compromise between speed, noise and power consumption [45], [48], [49]. The schematic for the folded-cascode amplifier used for this work is shown in Fig. 4(a). Transistors $M_1 - M_4$ form a differential cascoded pair, loaded by two current mirrors formed by M_5, M_6 and $M_{15} - M_{18}$, respectively. M_{14} forms the tail current source, and $M_9 - M_{13}$ provide the necessary biasing voltages. The overall biasing is set by the reference current source I_b. The output DC level is set by the common-mode feedback circuitry, which is detailed in Fig. 4(b). Operation of the circuit is straightforward: if, for instance, the sampled common-mode is higher than the reference voltage V_{cm}, the currents through M_2 and M_3 will increase, resulting in a greater voltage drop across M_5; since this voltage sets the bias for the output cascoded mirror, the currents through the output transistors will increase, pulling the output common-mode level down [45]. Resistors R provide source degeneration to minimize the loss in output swing. $M_7 - M_{10}$ form current sources and are biased by the voltage at node V_b in the main op-amp circuit. The folded-cascode op-amp, along with the common-mode feedback circuity, were simulated in the Spectre software. The amplifier's transfer function and performance specifications are provided in Fig. 5 and Table 1, respectively. All amplifiers used throughout the remainder of this article are based on this op-amp design.

B. MULTIPLIER

Nonlinear terms and voltage-controlled gains are achieved through the use of multiplier blocks. Four-quadrant analog multipliers have traditionally been based on the Gilbert cell,

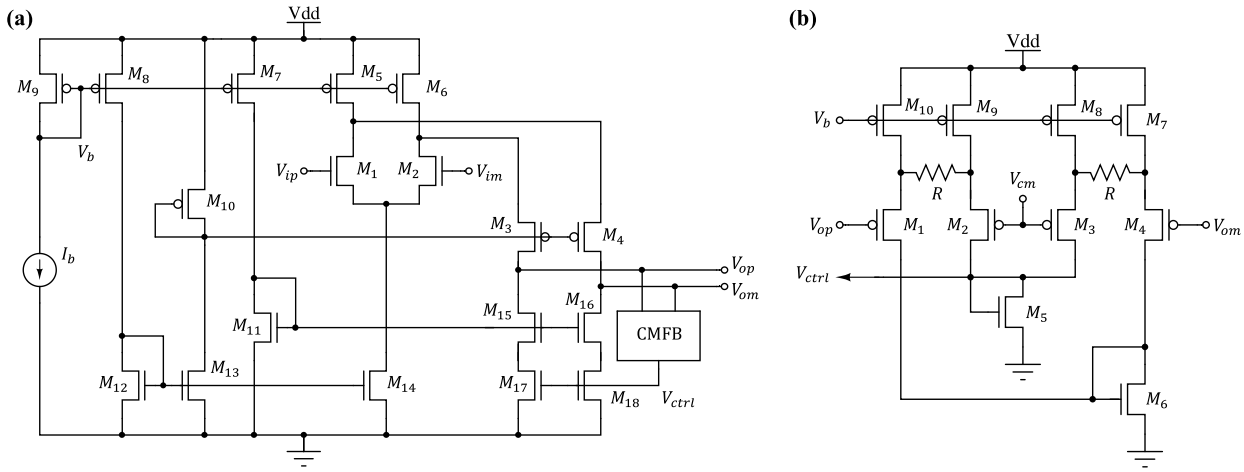


FIGURE 4. (a) Complete schematic of the folded-cascode operational transconductance amplifier [25], [26]. (b) Common-mode feedback circuitry (adapted from [45]).

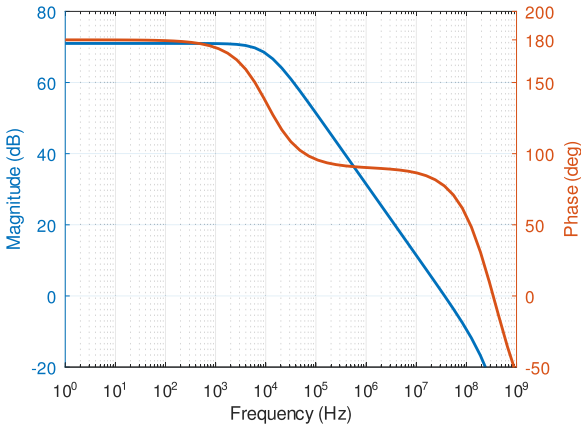


FIGURE 5. Bode plot of the folded-cascode op-amp's transfer characteristics.

which relies on the exponential relation between voltage and current in PN junctions to realize multiplication in the current domain [50]; when operating with voltage-mode signals, voltage-to-current converters are required to achieve good linearity over a reasonable range of input voltages [47]. Though the Gilbert cell may be realized with properly biased MOS devices, linearity and accuracy are often limited [51]. As such, many alternative designs have been proposed for CMOS-based multipliers [52]. Some topologies harness the square-law behavior of the drain current in the saturation region [53], [54], [55], [56], [57], while others use weak inversion to create translinear loops [58], [59], [60], [61].

For this application, a voltage-mode transconductor-based nonlinear cell was selected [32], [52], [62], [63], [64], [65], [66]. The complete circuit is depicted in Fig. 6. The design is based around two transconductor blocks, each comprising four matched NMOS transistors operating in the triode (linear) region. The input quad is driven by the differential signals $V_x = V_{xp} - V_{xm}$ and $V_y = V_{yp} - V_{ym}$, outputting a differential current I given by [32]:

$$I = -\mu_n C_{ox} \left(\frac{W}{L}\right)_i V_x V_y \tag{10}$$

where μC_{ox} is the NMOS intrinsic conductance and $(W/L)_i$ is the input devices' aspect ratio. Nonlinearities are effectively cancelled by the transconductor's symmetrical design. The output quad is driven by $V_z = V_{zp} - V_{zm}$ and the op-amp's output, and acts as a voltage-controlled resistor R given by [32]:

$$R = \frac{1}{\mu_n C_{ox} \left(\frac{W}{L}\right)_o V_z} \tag{11}$$

with $(W/L)_o$ being the output devices' aspect ratio. The op-amp is effectively configured as a transimpedance amplifier, having I as the input current and R as the feedback resistor. Hence, the op-amp's output voltage $V_o = V_{op} - V_{om}$ is simply given by:

$$V_o = -IR. \tag{12}$$

Combining (10) and (11) into (12), yields:

$$V_o = -\left[\frac{-\mu_n C_{ox} \left(\frac{W}{L}\right)_i V_x V_y}{\mu_n C_{ox} \left(\frac{W}{L}\right)_o V_z}\right] = \frac{\left(\frac{W}{L}\right)_i V_x V_y}{\left(\frac{W}{L}\right)_o V_z}. \tag{13}$$

Thus, this cell is capable of performing both multiplication and division; furthermore, as demonstrated in [32], it can also compute the square root of a signal, by applying an appropriate DC offset to V_z. In this application, however, V_z will only be used as a DC control voltage, setting the gain of the $V_x V_y$ product; the gain tuning range is tailored by sizing the input and output transistors' aspect ratios.

The main condition for linear operation is that all transistors remain in the triode region, which requires proper DC biasing [63]. While the DC level V_y is kept at the op-amp's common-mode, those of V_x and V_z must be raised to account for input gate threshold voltages [37]; a positive

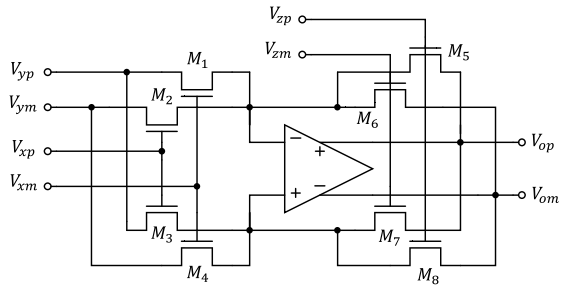


FIGURE 6. Schematic of the voltage-mode, fully-differential multiplier.

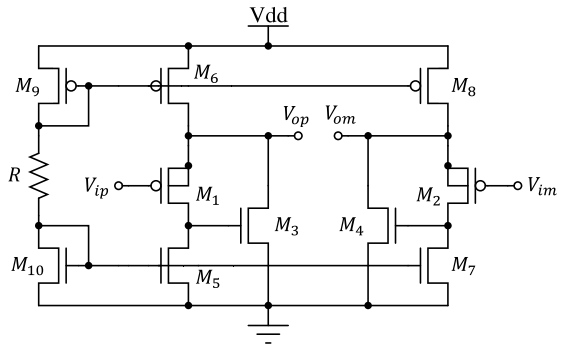


FIGURE 7. Analog level-shifting circuit, based on super source-follower design.

shift of around 300 mV was found to provide a suitable compromise between multiplier linear range and input/output swing. A level-shifting circuit was designed for this purpose: it comprises two so-called "super source followers", one for each differential branch. The complete circuit is shown in Fig. 7. The super source follower employs negative feedback through an NMOS device to enhance the otherwise poor linearity and unity-gain error of the classical source follower topology [47]. To further improve gain accuracy, the body terminals of the two input PMOS were tied to their respective sources, thereby avoiding body effect. Transistors $M_5 - M_{10}$ provide the biasing currents, which are set by resistor R. The circuit was sized to provide minimal signal attenuation ($\sim$ 0.2 dB) and the required 300 mV DC shift. Though using two single-ended source followers may introduce matching errors between the two differential signal paths, this solution is preferable to using a single fully-differential source follower [67], [68], which displayed overall worse performance in simulations, namely regarding distortion and gain error.

The simulated DC and AC characteristics of the complete multiplier (including level-shifters) are shown in Fig. 8 and Fig. 9, respectively. For both simulations, the multiplier was loaded with 10 pF on each output. The previously described folded-cascode amplifier was reused as the op-amp in this design. As this amplifier is not well-suited to driving resistive loads, ratios of $^{W}/_{L} = 1/10$ were chosen for both the input and output NMOS quads, to produce relatively high equivalent resistances; the long channel lengths also improve

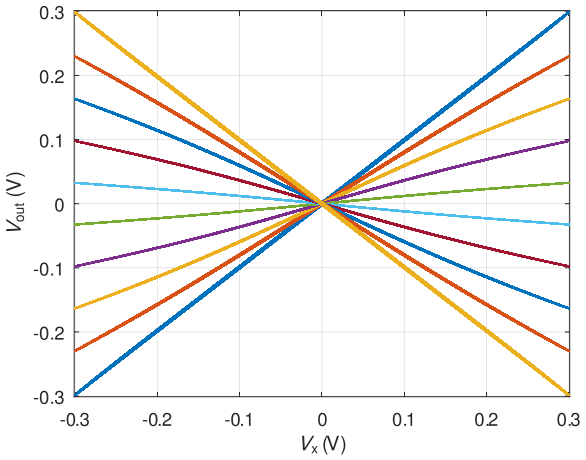


FIGURE 8. Multiplier DC transfer curves. Obtained by applying a 300 mV, 100 kHz sine wave to the V_X input and varying V_Y between -300 mV and 300 mV; V_Z was set to 300 mV.

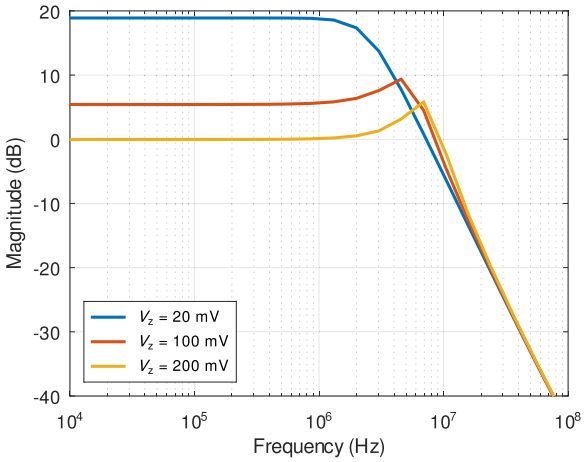


FIGURE 9. Multiplier transfer function for multiple values of V_Z; V_X was fixed at 20 mV and V_Y set to 200 mV.

the accuracy of the strong inversion square-law model [47]. Fig. 9 shows some peaking occurring at high frequencies ($> 10^6$ Hz); this is thought to result from coupling between the op-amp's output impedance and the capacitances from the output quad and load. This resonance may also account for the phase shift observed at the output, as seen by the slight elliptical shape of the traces in Fig. 8. Simulations also showed that the linear range of the multiplier is highly dependent upon the gain control voltage, V_z. Namely, decreasing V_z (hence, increasing overall multiplier gain) will quickly degrade the input-referred total harmonic distortion (THD). Therefore, if a gain greater than $\sim$10 is required, it may be necessary to distribute it between the multiplier and a further amplification stage, to ensure reasonable accuracy and linear range from the multiplier.

IV. FUNCTIONALITY

The previous section described the building blocks of a CMOS analog solver in a general context. It is now prudent

to look at some of the ways in which these circuits can be configured to suit a wide variety of conditions, in the context of the capabilities and limitations of electronic ACs.

A. INITIAL CONDITIONS

The value of the state variables at the start of the computation is determined by the initial voltage across the integrators' capacitors, as seen in (8). To set this value, an arrangement like that of Fig. 10(a) can be used. When switch S_1 is closed, the capacitor C is charged by voltage source V_C through resistor R_C. The charging time is set by the time constant $R_C C$, and should be as small as is allowed by V_C's driving capability. As long as S_1 is closed, the integrator's output V_o is locked at $-V_C$, and the computation is kept on hold. When S_1 is opened, the integrator resumes regular operation, starting at $V_o = -V_C$. Thus, S_1 also behaves as a reset switch, which can be toggled at any point during the computation to force the system to a set of initial conditions. A transient simulation is shown in Fig. 10(b), where the integrator is reset to its initial conditions by closing the switch at $t = 0.3$ ms; normal computation is resumed when the switch is released, at $t = 0.4$ ms. Though illustrated in a single-ended configuration, this arrangement is easily adapted to a fully-differential topology. For an integrated solution, the switch and voltage source might be included on-chip, in which case they could be set via external programming logic.

B. HOLD

Besides resetting, some applications demand that the computation be held on the current value for an indeterminate amount of time, e.g., for sampling by another circuit. This can be accomplished by placing track-and-hold amplifiers at the integrators' outputs, as shown in Fig. 11(a). While switch S_h is closed, the voltage on capacitor C_h tracks integrator A_1's output; when S_h opens, C_h retains its voltage at that moment, which is then buffered by the output amplifier A_2. To prevent the computation from proceeding when S_h is open, the held output is fed back into the integrator through switch S_{set}, forcing it to retain the current output value. A transient simulation is shown in Fig. 11(b): the computation is put on hold at $t = 0.3$ ms when S_h opens and S_{set} closes, then resumes at $t = 0.4$ ms when both switch states.

C. SCALING

Since an AC attempts to map a problem onto physical quantities, it is necessary to establish the mathematical relation between the problem's variables and their physical representations. This is accomplished through the use of scaling factors, in both amplitude and time [7], [9], [10].

Amplitude scaling (also known as magnitude scaling) relates a system variable to its internal representation in the computer. For example, when solving a heat transfer model, a temperature difference in the system might be proportional to a voltage difference in the solver. The relation between the two variables is given by a scaling factor, s (in the previous example, with units volts/kelvin). More generally,

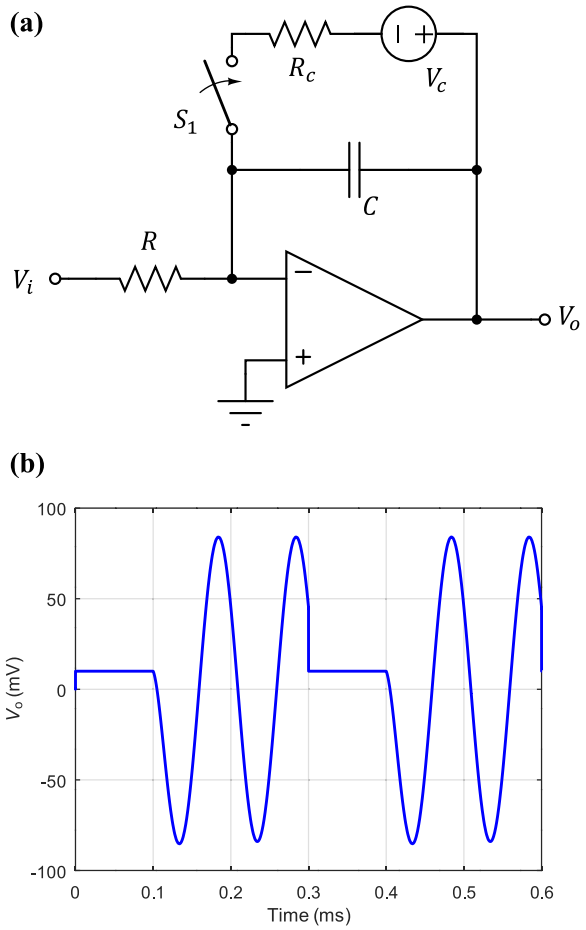


FIGURE 10. (a) Integrator with setting and resetting functionality. (b) Transient simulation of resettable integrator.

for a system variable x, the analog solver will compute the quantity $x' = sx$ [7]. The choice of s is dependent upon the characteristics of both the simulated system and the AC, itself. A large s will mean greater sensitivity to small changes in the problem variable, improving the precision of the computation; however, the solver may then not be able to accommodate larger variations, due to saturation concerns. On the other hand, a small s will improve the solver's dynamic range but sacrifice resolution due to the increased effect of noise, offsets and other circuit non-idealities. For a low-power CMOS analog solver, given the restrictions placed on voltage (and current), s will often need to be a small number, perhaps in the order of 1/10 to 1/1000 circuit unit per system unit. The optimal scaling needs to be determined for each problem individually, often requiring some previous knowledge concerning the maximum expected values for each system variable. Numerical simulations can be helpful in this regard.

Amplitude scaling can be set by tuning the internal gains of the analog solver circuit, namely in the integrators and multipliers. Since a solver relies on global negative feedback (due to the inherent relation between state variables and their

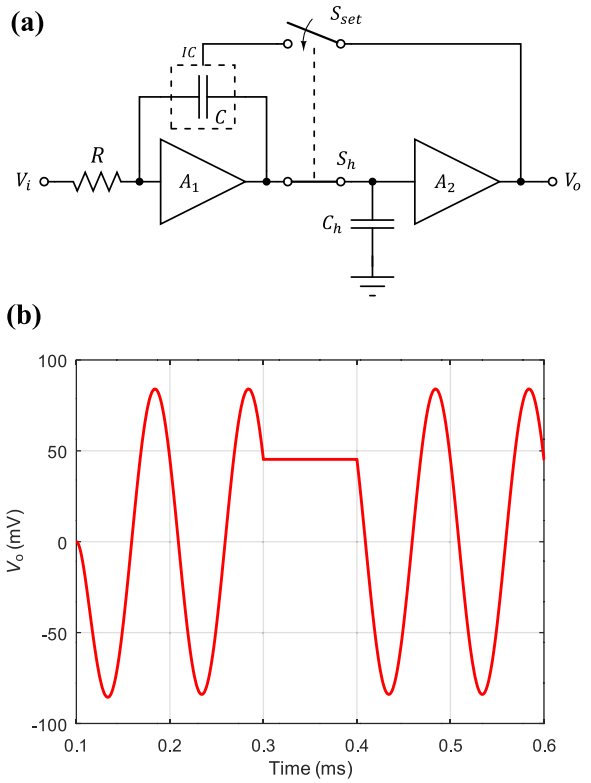


FIGURE 11. (a) Simplified schematic of an integrator with output track-and-hold amplifier. The dashed box around *C* represents a "reset" architecture similar to that of Fig. 10(a), where the "initial condition" (*IC*) is set to the current value of V_o. (b) Transient simulation of the "hold" mode.

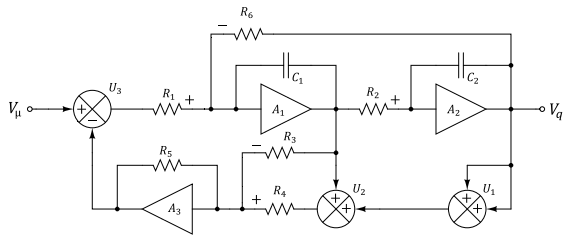


FIGURE 12. Simplified schematic of the van der Pol analog solver. "Plus" and "minus" signs denote signal polarity.

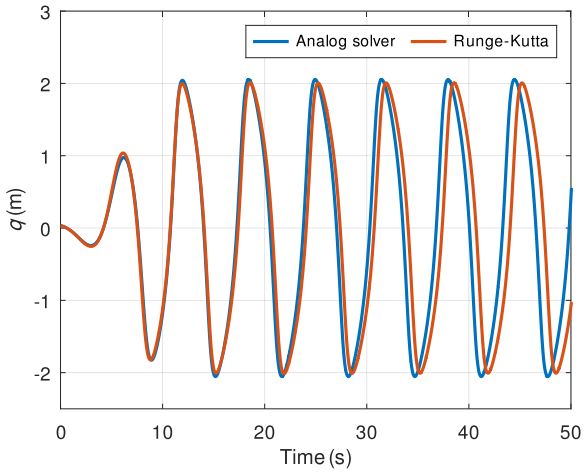


FIGURE 13. Time plot of the scaled output from the van der Pol analog solver for $\mu = 1$, compared to solution provided by a Runge-Kutta algorithm.

derivatives), larger gains translate to smaller scale factors. To accommodate a wide range of problems and configurations, it is useful that these gains be easily adjustable. The multiplier described in Section III is especially suitable for this since its gain can be set through a control voltage. On the other hand, the integrators' gain is set by the input resistors, which makes adjustment less straightforward. One option is to include a set of resistors that can be interchanged through control logic. Another option is to implement voltage-controlled resistors using MOS devices, albeit with limited linear range. Finally, frequency-controlled resistors can be implemented using switched capacitors, though their application is limited in continuous-time circuits due to clock feedthrough [49].

Time scaling is analogous to amplitude scaling, though much simpler in practice, being set by the integrators' *RC* time constant. For instance, a time constant of 1 s means the solver will simulate a system in real time. Faster computation times can be achieved by decreasing the integrators' feedback capacitors and/or input resistors, as long as the operating frequency remains well below the op-amps' GBW.

V. APPLICATIONS

The building blocks described in Section III can be employed to solve a wide range of problems in mathematics, science and engineering. Four nonlinear case studies were selected for implementation in fully-differential CMOS analog solvers/simulators. The most relevant results are presented and discussed from a qualitative approach. These problems are meant to illustrate some the capabilities of the analog computing paradigm, though they are by no means exhaustive. Each example should also be interpreted not just as a concrete system, but as an entire subset of systems, since the analog solvers presented herein are, in each case, reprogrammable and easily expandable to accommodate a number of problems. For the sake of compact notation, all variables are assumed to be functions of time, unless otherwise stated. Furthermore, circuit design values and simulation parameters are included in Appendix.

A. VAN DER POL OSCILLATOR

The van der Pol oscillator was first proposed by the Dutch electrical engineer B. van der Pol, while studying triode circuits in the 1920s [69]. Since then, this equation has found application in a wide range of areas; among them, it has been used to model cardiac oscillations, neuron action potentials, deterministic chaos and radio-frequency electronic oscillators [1], [70], [71], [72]. Let $x = \begin{bmatrix} q & v \end{bmatrix}^T$ be the state vector, with q being the position, and v the velocity. Then, the dynamics of the classical van der Pol oscillator are

described by the nonlinear state-space model [3]

$$\dot{q} = v \tag{14a}$$

$$\dot{v} = \mu \left(1 - q^2\right) v - q \tag{14b}$$

where μ is a positive parameter denoting the magnitude of the nonlinear damping. For our purposes, it is more convenient to describe the system solely in terms of q: by combining (14a) and (14b), we can write

$$\ddot{q} = \mu \left(\dot{q} - q^2 \dot{q}\right) - q. \tag{15}$$

Integrating (15) twice with respect to time yields

$$q = \int\int_{t_0}^{t} \mu \left(\dot{q} - q^2 \dot{q}\right) - q \, dt^2 + v_0 t + q_0 \tag{16}$$

where q_0 and v_0 respectively denote the initial displacement and velocity, at time t_0. The model is now suitable for implementation in an analog solver. A simplified schematic of the circuit is shown in Fig. 12. The output voltage V_q is related to q by a scale factor in units mV/m. Amplifiers A_1 and A_2 act as (inverting) integrators, while A_3 is configured as an inverting, summing amplifier. The integrators were designed with 1 MΩ input resistors and 10 pF feedback capacitors, producing a time constant of 10 μs, i.e., the system is simulated $10^5\times$ faster than real-time. The resistors' high value allows for a wide range of integrator gain tunability, without severely compromising the stage's input impedance. Multiplier U_3 acts as a variable-gain amplifier and allows μ to be set via an external DC voltage, V_μ. Another way to implement voltage control of μ would be through the multiplier gain control voltages. However, the current solution provides better adjustment range and linearity, albeit by using additional devices. The circuit dissipates a total of 4.0 mW.

Transient simulations were conducted for multiple values of μ. The results were compared to those obtained from the MATLAB function `ode45`, which employs the Dormand-Prince pair of 4$^{\text{th}}$- and 5$^{\text{th}}$- order Runge-Kutta methods to numerically integrate differential equations [3]. While `ode45` is not among the most precise numerical solvers available (even within MATLAB's suite), it is still more sophisticated than the numerical methods that are often available in real-world resource-constrained applications. Because analog solvers are particularly attractive in these scenarios, since they can produce moderately accurate solutions with minimal hardware requirements, `ode45` constitutes a fitting benchmark for the accuracy and precision of the analog-derived solutions. Time and phase-space plots are presented in Fig. 13 and Fig. 14, respectively. In both cases, the solver's output was converted to the original problem's magnitude and time scales by multiplying it by the appropriate factors, so it could be directly compared to the numerical solution.

The analog solver matches the numerical integrator within 3% error in both amplitude and frequency, however, as evidenced by the phase plots, the circuit's precision

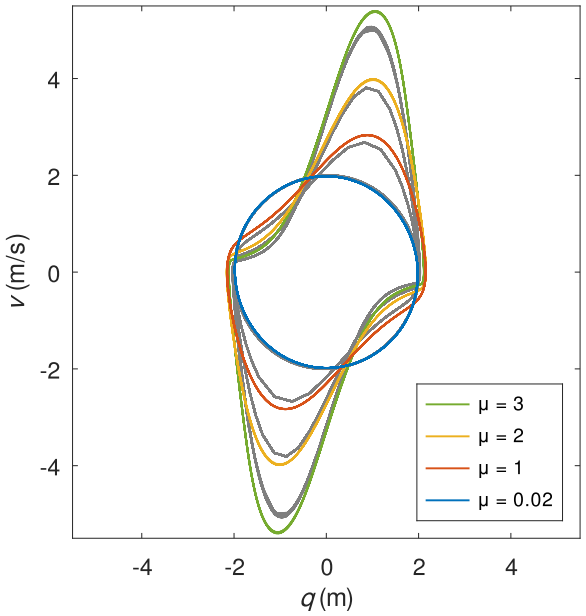


FIGURE 14. Phase-space plot of the van der Pol solver, for various values of μ. The grey traces correspond to the solutions obtained from function `ode45`.

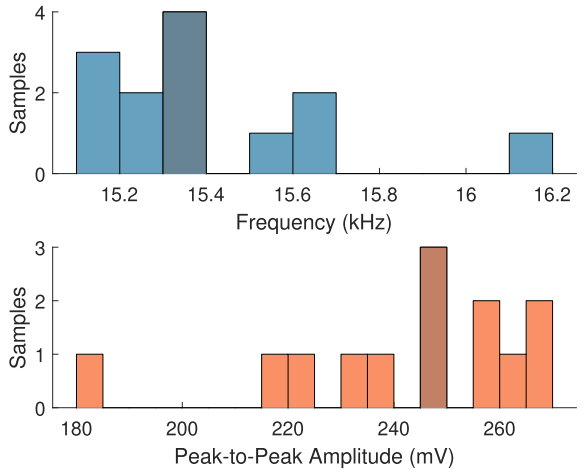


FIGURE 15. Simulation results of frequency and amplitude variation in PVT corners, for the van der Pol solver. The darker colors indicate the nominal specification.

degrades for higher values of μ. This loss of precision is concomitant with the increase in signal amplitude, which is especially prominent in the velocity component. These results typify the compromises inherent in the design of an analog solver, namely the trade-off between operating range and precision. That is, in order to accommodate larger signal swings (thereby increasing the computer's dynamic range), small scale factors are needed. However, these require large gains, which in turn strain the circuit's linear operating range, harming precision. The balance between these two factors should be considered for each individual case, according to the designer's priorities and the solver's requirements.

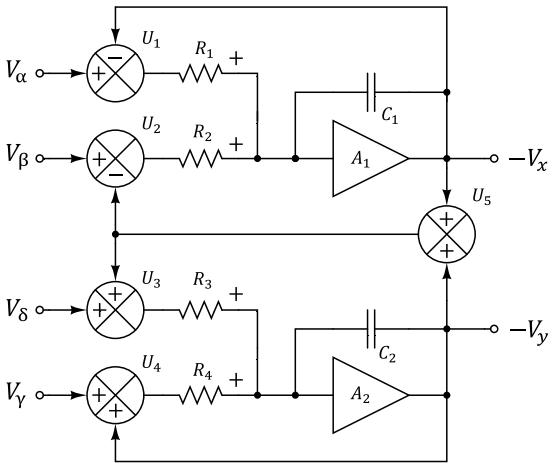


FIGURE 16. Simplified schematic of the Lotka-Volterra analog solver.

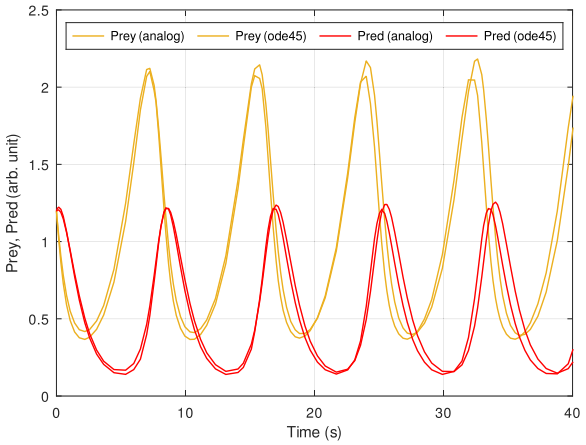


FIGURE 17. Time plot of the outputs from the Lotka-Volterra solver, compared to solutions obtained by `ode45`.

To assess the circuit's robustness to process, supply voltage and temperature (PVT) variations, corner scenarios were also examined. Typical tolerances on process and operating conditions were considered: the transistors were replaced by worst-case variants with ±25% typical speed, the supply voltage was made to vary by ±5% of its nominal value of 1.2 V, and the temperature ranged from 0 to 100° C. Overall, 12 different corners, plus the nominal case, were analyzed. Noise was also included in these simulations, up to a frequency of 100 MHz. The circuit's robustness was evaluated with respect to frequency and amplitude, and the results are summarized in Fig. 15. While the frequency remains within 5% of its nominal specification, the amplitude declines by 27% in the case of low voltage coupled with slower devices. However, these values are within the expected range of variation, and do not compromise the circuit's function. Furthermore, peak-to-peak noise amplitude was found to be < 400 μV, and therefore to have negligible impact on overall precision. Since subsequent systems were built using the same analog cells as the van der Pol solver, it is reasonable to assume these PVT results hold for those circuits as well, thus obviating further analysis of the topic.

B. LOTKA-VOLTERRA EQUATIONS

The Lotka-Volterra equations describe the nonlinear dynamics of biological predator-prey interactions [3]. This model was originally proposed for the study of chemical reactions [73], and has since proved relevant in other fields, such as economics [74]. The system comprises a set of two nonlinear, first-order differential equations [1]

$$\dot{x} = \alpha x - \beta xy \tag{17a}$$
$$\dot{y} = \delta xy - \gamma y \tag{17b}$$

where x and y represent the prey and predator populations, respectively, and α, β, δ and γ are positive constants relating to the death and growth rate of both species. Following the previous procedure, integrating both equations with respect to time yields the integral equations

$$x = \int_{t_0}^{t} \alpha x - \beta xy \, dt + x_0 \tag{18a}$$
$$y = \int_{t_0}^{t} \delta xy - \gamma y \, dt + y_0 \tag{18b}$$

with x_0 and y_0 denoting the initial populations. The corresponding analog solver is shown in Fig. 16. The two output signals, V_x and V_y, are appropriately scaled representations of the corresponding populations, while DC voltages V_α, V_β, V_δ and V_γ set the matching constants. Time scaling is identical to the previous example. The solver's total power consumption is 4.8 mW.

Transient simulations were conducted for various values of α, β, δ and γ. Fig. 17 and Fig. 18, respectively, show time and phase-space plots for parameters $\alpha = 2/3$, $\beta = 4/3$ and $\delta = \gamma = 1$, the latter for various initial conditions; both plots were scaled in magnitude and time to match the original system. The time plot includes a comparison with the solutions computed by `ode45`, with coefficients and initial conditions identical to those set on the analog solver. The solver displays good accuracy, though worse precision than in the previous example, as both signals' amplitudes stray significantly from the numerical solutions over the course of the simulation.

This particular circuit presented many challenges for successful simulation. Results varied extensively based on choice of numerical integration algorithm, or convergence criteria. As such, it was not always possible to make meaningful comparisons with the numerical solutions. In particular, the phase plot of Fig. 18 should be taken as merely indicative of the effect of initial conditions on the analog solver's trajectory, rather than an accurate representation of the original system's behavior. These difficulties may be inherent to the application of numerical methods to nonlinear systems, even in the case of analog circuit simulators. Further investigation on the effects of different numerical algorithms on nonlinear behavior, especially if compared to physical

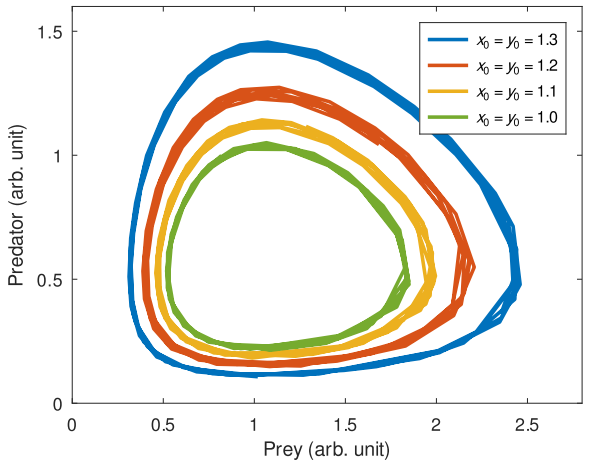


FIGURE 18. Phase-space plot of the Lotka-Volterra solver, for various initial conditions.

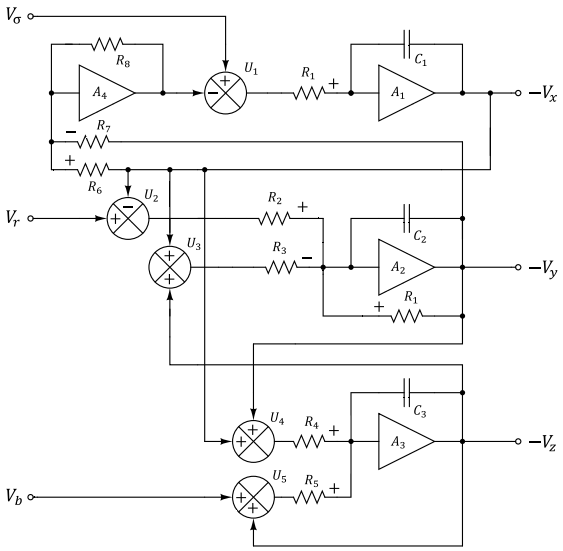


FIGURE 19. Simplified schematic of the Lorenz analog solver.

implementations of analog solvers, could lend some insight into the subject, and is a topic left for future research.

C. LORENZ SYSTEM

The Lorenz system was originally developed by meteorologist Lorenz in the study of convection phenomena in the atmosphere [75], and is described by three nonlinear first order equations [1]

$$\dot{x} = \sigma\,(y - x) \tag{19a}$$
$$\dot{y} = x\,(r - z) - y \tag{19b}$$
$$\dot{z} = xy - bz \tag{19c}$$

where x is proportional to the convective motion, y is proportional to the temperature gradient between opposing currents and z denotes the nonlinearity of the temperature profile; σ, b and r are constant system parameters [75]. For specific values of σ, b and r the system displays

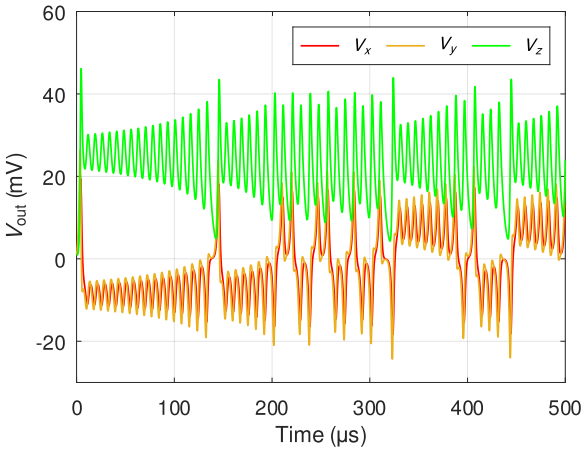


FIGURE 20. Time plot of the Lorenz solver's three outputs.

deterministic chaotic behavior, with the three-dimensional phase-space trajectory spiraling around two points, known as the attractors [1]. These trajectories are highly sensitive to initial conditions, and never repeat themselves. Integrating all three equations with respect to time (and rearranging some terms) yields the integral equations

$$x = \int_{t_0}^{t} \sigma\,(y - x)\,dt + x_0 \tag{20a}$$
$$y = \int_{t_0}^{t} rx - xz - y\,dt + y_0 \tag{20b}$$
$$z = \int_{t_0}^{t} xy - bz\,dt + z_0 \tag{20c}$$

where x_0, y_0 and z_0 represent the initial conditions. Based on (20), the analog solver shown in Fig. 19 was designed. The voltages V_σ, V_r and V_b control the corresponding parameters, which were set to those originally proposed by Lorenz: $\sigma = 10$, $b = 8/3$ and $r = 28$. The integrators' time constant was once more set to 10 μs. The circuit's power dissipation is 6.1 mW. The plot from a time-domain simulation of the V_x, V_y and V_z signals is shown in Fig. 20; the $x - z$ projection of a portion of the phase-space trajectory is shown in Fig. 21.

The chaotic nature of these signals makes them difficult to compare to equivalent numerical solutions, as small variations in initial conditions or system parameters can produce vastly different outcomes. Thus, such a comparison is not presented for this example. Nevertheless, it is still noteworthy how a relatively simple circuit can replicate such complex behavior.

D. ROTATIONAL MOTION IN TWO DIMENSIONS

The rotational motion of a rigid body can be described by its orientation relative to an inertial frame of reference. For motion in $\mathbb{R}^2$, let (u, v) be the body frame, (x, y) be the inertial frame, and u_x be the coordinates of u relative to x (and so forth for the remaining permutations). Then, the rotation matrix is

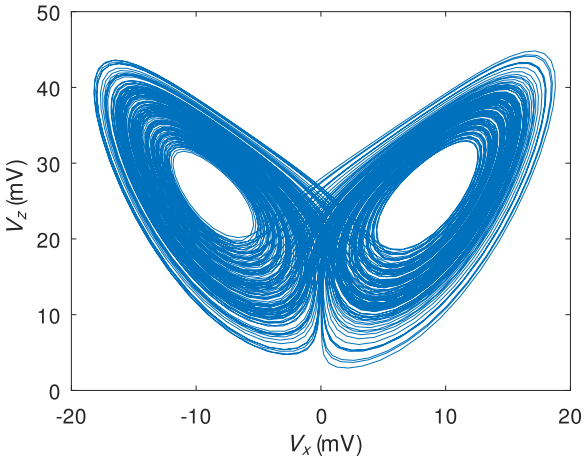


FIGURE 21. $x - z$ projection of the Lorenz solver's phase-space trajectory, between 8 ms and 10 ms simulation times.

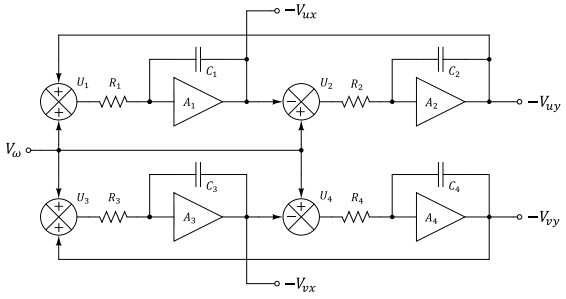


FIGURE 22. Simplified schematic of the rotational kinematics analog solver.

defined as [76]

$$\mathbf{R} = \begin{bmatrix} u_x & v_x \\ u_y & v_y \end{bmatrix}. \tag{21}$$

It follows from its construction that $\mathbf{R}$ is orthogonal, that is

$$\mathbf{R}^T\mathbf{R} = \mathbf{I} \tag{22}$$

which also implies

$$\text{tr}(\mathbf{R}^T\mathbf{R}) = 2. \tag{23}$$

Furthermore, if we assume that all frames are right-handed, it can be shown that [76]

$$\det \mathbf{R} = 1. \tag{24}$$

The kinematics of rigid rotation in $\mathbb{R}^2$ are described by [77]

$$\dot{\mathbf{R}} = \mathbf{S}(\omega)\mathbf{R} \tag{25}$$

where $\mathbf{S}(\omega)$ is a skew-symmetric matrix constructed from the time-dependent angular velocity ω,

$$\mathbf{S}(\omega) = \begin{bmatrix} 0 & -\omega \\ \omega & 0 \end{bmatrix}. \tag{26}$$

For constant ω, the exact solution to (25) is given by [77]

$$\mathbf{R} = \begin{bmatrix} \cos(\omega t) & -\sin(\omega t) \\ \sin(\omega t) & \cos(\omega t) \end{bmatrix}. \tag{27}$$

The solution to the general case where ω is a function of time can be approximated through discretization, i.e., computing the system's behavior one step at a time, during each of which ω is considered to be constant. The discrete-time form of (25), for constant ω, is given by the difference equation [76]

$$\mathbf{R}(k+1) = e^{\mathbf{S}(\omega)\Delta t}\mathbf{R}(k) \tag{28}$$

where k is an instant of time and Δt is the sampling period (interval between consecutive steps). By computing (28) with sampled values of a time-varying ω, a general solution to the dynamics of $\mathbf{R}$ can be approximated, with an error proportional to the size of Δt.

In contrast, an analog implementation solves the general case without any further approximations, as it can naturally accommodate a time-dependent ω. To construct the circuit, we'll first note that (25) can be expanded into a set of four ODEs, one for each element of $\mathbf{R}$,

$$\dot{u}_x = -\omega u_y \tag{29a}$$
$$\dot{v}_x = -\omega v_y \tag{29b}$$
$$\dot{u}_y = \omega u_x \tag{29c}$$
$$\dot{v}_y = \omega v_x \tag{29d}$$

which can be interpreted as describing a dynamical system with one input (ω) and four outputs (one for each element of $\mathbf{R}$). Following the previous examples, integrating both sides of (29) yields

$$u_x = -\int_{t_0}^{t} \omega u_y \, dt + u_{x_0} \tag{30a}$$
$$v_x = -\int_{t_0}^{t} \omega v_y \, dt + v_{x_0} \tag{30b}$$
$$u_y = \int_{t_0}^{t} \omega u_x \, dt + u_{y_0} \tag{30c}$$
$$v_y = \int_{t_0}^{t} \omega v_x \, dt + v_{y_0} \tag{30d}$$

where the "0" subscript denotes initial conditions. The system (30) is now readily mapped onto an analog solver, shown in Fig. 22. The resultant circuit comprises two independent solvers, each bearing resemblance to a voltage-controlled, double-integrator oscillator, with quadrature sine outputs and frequency set by V_ω [72], [78]. The integrators run with a time constant of approximately 1.4 μs, and the complete circuit dissipates a total of 5.4 mW.

Two transient conditions were considered for simulation: in the first, V_ω was set to a constant voltage, and in the second to a low-frequency sine wave. In both cases, the circuit's outputs were compared to the solutions computed by `ode45`, as well as those provided by (28). For the latter case, the sampling period was chosen to be very small (as fine as Spectre's own simulation timestep, < 0.1 s), in order to best approximate an exact solution. As `ode45` cannot directly account for ω being a function of time, an approach similar to that described for the discrete-time solution had to be undertaken, where the total integration time was divided

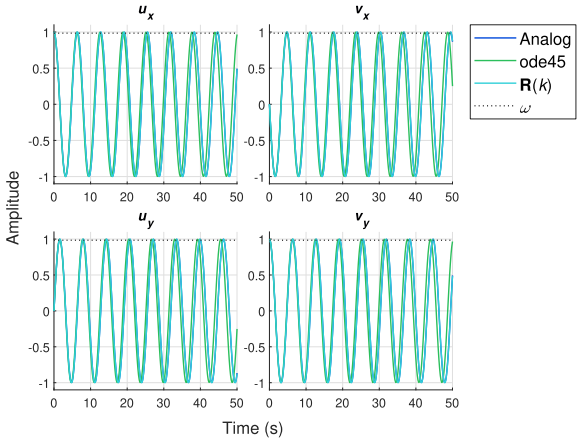


FIGURE 23. Time plot of the four elements of R for constant ω, generated by the analog solver, `ode45` and the discrete solution.

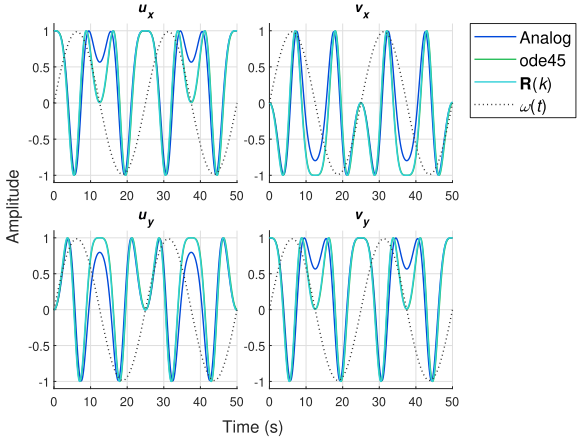


FIGURE 24. Time plot of the four elements of R for sinusoidal ω, generated by the analog solver, `ode45` and the discrete solution.

into smaller intervals of constant ω. Decreasing the length of each interval, thereby increasing the total amount of "steps", improves the approximation, albeit at the expense of computational resources; a step of 0.1 s was chosen, to match that of the discrete-time solution. Comparisons between the circuit, numerical integrator and discrete solution, for both constant and time-varying ω, are plotted in Fig. 23 and Fig. 24, respectively. The initial conditions were set to the identity matrix, that is, $u_{x0} = v_{y0} = 1$ and $u_{y0} = v_{x0} = 0$. The analog solver shows very good agreement with the numerical and discrete solutions in the first case, but differs somewhat in the second, as it fails to match the correct amplitude at every other half-period of $\omega(t)$.

The accuracy of the analog and numerical solutions can also be evaluated by how well properties (23) and (24) are respected throughout the simulation time. Fig. 25(a) and (b) plot the value of $\mathrm{tr}(\mathbf{R}^T\mathbf{R})$ and $\det \mathbf{R}$, respectively, for both solvers. The results are presented for constant ω only, since the approximation required for employing `ode45` to a varying ω essentially prevents any significant error

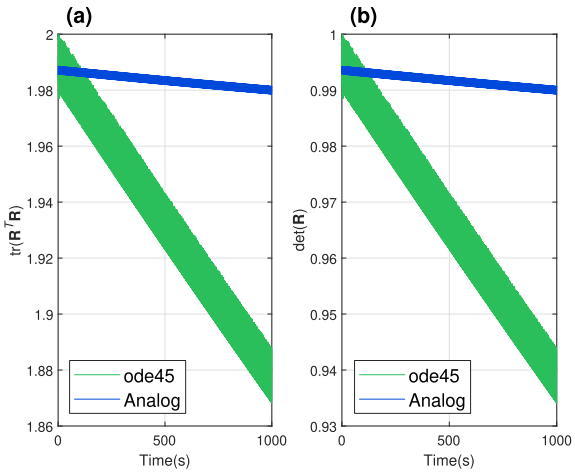


FIGURE 25. Evolution of (a) the orthogonality and (b) the determinant of R throughout 1000 s of simulation time, for constant ω. Results from both solvers are compared.

accumulation, precluding meaningful comparison to the analog solver. Nonetheless, for constant ω, the circuit presents a significant improvement over the numerical integrator. Though there is an offset at the start (due to inaccuracies in setting the initial conditions), both the orthogonality and the determinant of the analog solution stay within 1% of their initial values. This represents a 6× improvement over the numerical solution. It is important to consider, also, that the analog solver's error is a combination of both the circuit's intrinsic errors, and the simulator's inaccuracies. Like the example from Subsection V-B, this circuit's behavior proved to be highly dependent on simulator parameters and selection of integration algorithm. The extent to which these choices impact the accuracy and precision of the solver's simulated behavior is yet to be quantitatively described.

VI. CONCLUSION AND OUTLOOK

This work shows how a set of ODEs can be mapped onto standard CMOS cells for realization of practical analog solvers, aimed at the simulation of nonlinear dynamics. When feasible, the results obtained from circuit simulations were compared to those from numerical methods. The former tend to lack in precision relative to the latter, and this can be attributed to the accumulation of many small errors throughout the circuits. One source of error is the op-amp's finite gain, which limits the precision of the integrator and multiplier modules; additionally, given the prominence of resistive loads throughout the circuits, it may be worth investigating alternate op-amp topologies, with better driving capabilities. The multipliers' gain also introduces some distortion, especially in larger signals, as the NMOS devices leave the linear operating region. Yet another source of error lies in the level-shifters' slight attenuation, though this could be compensated by downstream amplification stages, if necessary. Finally, intrinsic MOS capacitances introduce phase shifts in the signal paths, impacting the overall precision of

the circuit. Some error sources which were not accounted for in the simulations include imbalance between the differential signal paths due to mismatches in the amplifiers and level-shifters, and parasitic capacitances from physical layout. This latter issue, which concerns the physical realization of the circuits, also warrants discussion. Since all simulations were conducted in an industry-standard environment with foundry-supplied device models, the results presented throughout this work should constitute accurate representations of real circuit behavior, despite the lack of a validated physical prototype. Time and financial concerns place integrated circuit fabrication outside the scope of this contribution. Also, while the proposed circuits could be adapted for discrete, commercially available components and realized in printed circuit-board format, their specifications would necessarily differ so significantly from those presented herein as to constitute entirely distinct entities, thus adding little value to this discussion. Nevertheless, the design and testing of prototypes for analog solvers, whether discrete or integrated, will be pursued in future work.

As other works have shown, despite generally lower precision, analog computations can be orders of magnitude faster and more power efficient than those on even the best digital computers [17], [18], [30], [37]. This is largely attributable to the parallel architecture of ACs, which allows for multiple computations to be performed concurrently; this feature contrasts with the inherently sequential algorithms executed on digital devices, where parallel computing (i.e. running multiple processing units simultaneously) may be of limited benefit in solving multidimensional problems with a large degree of inter-variable coupling [31], [36], [37].

While the present work followed a traditional approach to analog computation, this comprises only a small subset of the possibilities allowed by contemporary technology. For instance, besides multiplication, different nonlinear terms, such as trigonometric functions and logarithms, can be realized in integrated circuit by various approximations [40], [41], [43]. Likewise, constraints in state or input/output variables can be enforced by using nonlinear components, such as diodes, or even through amplifier saturation [10]. Hybrid dynamical systems with continuous- and logical- variables might also be implemented by employing comparators and other digital circuitry [7]. With suitable modifications, the presented topologies may also be adapted to discrete-time switched capacitor designs [12], or even current-mode operation [5], [17], [18], [79], both of which might produce higher precision solvers. Finally, the use of ACs as analog ''co-processors'' or ''accelerators'' as been explored in the past [5], [17], [18]. In these systems, computations are performed by using a low-precision analog solution as the starting point for a high-order numerical solver. Such architectures attempt to harness the best of both technologies to achieve very precise solutions using less resources than a purely digital processor, but also requires a meticulous synchronization scheme to ensure optimal performance. These hybrid arrangements also raise the question of analog-digital interfaces, and how digital structures like memory and data converters can be exploited, not only to interact with external devices, but also enhance analog performance and capabilities.

The scope of AC applicability, which may initially seem limited to the study of physical systems, is greatly expanded if one begins to consider the multitude of problems in all areas of science and engineering which can be reformulated as time-dependent ODEs. For instance, it has been shown that the nonperiodic Toda lattice equations can produce a set of ODEs describing a ''smooth'' sorting problem, i.e., a dynamical system whose outputs settle on a sorted arrangement of the inputs [80], [81]. Optimization, too, has been reframed as the solution to a set of ODEs, encompassing linearly-constrained, quadratic programming problems [82], [83], [84], [85], [86]. Other, non-dynamical, formulations of linear and quadratic programming have also been proven suitable for implementation in analog circuitry [39], [84], [87], [88], [89], [90], [91], [92], [93], [94].

Continuous optimization architectures have already found application in some areas, including optimal and predictive control. An optimal controller attempts to find the plant trajectory that will minimize (or maximize) a performance criterion over a given time horizon, while subject to a set of constraints defined by the plant's model and any existing physical limitations [19], [95], [96]. Predictive control extends this idea by incorporating known information about upcoming external stimuli to solve a ''receding-horizon'' optimal control problem at each time instant, allowing it to anticipate (and compensate for) future events [97], [98]. In recent years, predictive control has garnered attention for its performance in constrained multi-input, multi-output systems [23], but its high computational demand still limits the scope of possible applications. Recent works have attempted to apply analog optimization strategies to design fast and efficient discrete-time optimal/predictive controllers [93], [94], [99], [100], [101], [102], [103], [104], [105]. Results are promising, but remain largely academical. A pervasive issue with these controller designs is the very large number of components necessary for implementing any reasonable prediction horizon, all the more so for higher order systems. Analog computing techniques may play a role in simplifying, and indeed extending, these designs toward more widespread application. Still, the interaction between analog-simulated systems and analog predictive controllers appears to remain largely unexplored, particularly in the nonlinear domain. It is especially interesting to consider how an analog system model might be incorporated directly into a controller's design, perhaps in a closed-loop configuration, so as to automate gain calculation and adjustment. Continuous-time optimal, and predictive, controllers are a natural extension of this rationale, and also warrant closer attention in future research. All in all, analog techniques and tools undoubtedly offer many exciting possibilities

TABLE 2. Device parameters for the folded-cascode amplifier of Fig. 4(a).

Device	Width (μm)	Length (μm)
M_1, M_2	158	0.59
M_3, M_4	118	0.6
M_5, M_6	188	0.51
M_7, M_8, M_9	18.8	0.51
M_{10}	5.13	1.2
M_{11}	2.9	3.8
M_{12}, M_{13}	2	0.76
M_{14}, M_{17}, M_{18}	20	0.76
M_{15}, M_{16}	29	0.7

Obs.: $I_b = 200\ \mu$A.

for the future of fast, low-power and portable control applications.

As device scaling approaches fundamental physical limits, it becomes increasingly important to search for alternatives to the modern computing paradigm. The mathematical description of dynamical systems, namely through differential equations, provides fertile ground for the exploration of analog computers in scientific and engineering applications. Reinvigorated by VLSI design, the AC has proven to still have a role to play in contemporary technology. The body of knowledge compiled throughout the 20$^{\text{th}}$ century, coupled with more recent innovations, provides a compelling foundation for further progress. Yet, the AC's capabilities seem to remain underdeveloped. This neglect may be due to a general lack of exposure outside the electronics research community. A common thread throughout this article has been the framing of analog computing as a cross-disciplinary instrument, highlighting its application in a variety of subjects. While the issue of what problems are, and aren't, computable by analog means remains an open question, it is clear that the scientific research community only stands to benefit from raised awareness of these devices, whose long history and well-established principles still remain relevant today.

APPENDIX
SIZINGS AND SIMULATION PARAMETERS

Tables 2 and 3 contain the sizing values for the folded-cascode amplifier and common-mode feedback circuit, respectively, from Fig. 4. Table 4 lists the nominal device sizings for the multiplier of Fig. 6, though these were tuned on a case by case basis. Table 5 holds the sizings for the level-shifter of Fig. 7, and Table 6 indicates component values for all four analog solvers of Section V. Finally, Table 7 summarizes the Spectre configurations used in Section V; any unlisted parameters were kept at their default values. The "Error Preset" and "Relative Tolerance" parameters provide the software with an indication of the number of steps and the convergence criteria required for each simulation. Meanwhile, step-size is dynamically adjusted by the numerical integration method according to circuit behavior, in order to keep the error within tolerance specifications.

TABLE 3. Device parameters for the common-mode feedback circuit of Fig. 4(b).

Device	Width (μm)	Length (μm)
M_1, M_2, M_3, M_4	11	0.36
M_5	9	1
M_6	9.1	1
M_7, M_8, M_9, M_{10}	18.8	0.51

Obs.: $R = 100\ \Omega$.

TABLE 4. Nominal device parameters for the input and output quads of the multiplier circuit of Fig. 6.

Device	Width (μm)	Length (μm)
$M_1 - M_4$	0.4	4
$M_5 - M_8$	0.4	4

TABLE 5. Device parameters for the level-shifting circuit of Fig. 7.

Device	Width (μm)	Length (μm)
M_1, M_2	20	0.36
$M_3 - M_{10}$	10	0.36

Obs.: $R = 50\ \text{k}\Omega$.

TABLE 6. Design specifications for the four analog solvers.

Circuit	R (kΩ)	C (pF)
van der Pol	1000	10
Lotka-Volterra	1000	10
Lorenz	32*	10
2D rotation	141.5	10

*Except for $R_1, R_7, R_8, R_9 = 1\ \text{M}\Omega$.

TABLE 7. Simulator parameters for the four analog solvers.

Circuit	Analysis (time)	Error preset	Method	Relative tolerance
van der Pol	dc, trans (1 ms)	cons	gear2	10^{-3}
Lotka-Volterra	dc, trans (0.5 ms)	mod	trapgear2	10^{-4}
Lorenz	dc, trans (10 ms)	cons	traponly	10^{-5}
2D rotation	dc, trans (1 ms)	cons	traponly	10^{-6}

Legend: trans = transient, mod = moderate, cons = conservative.

REFERENCES

[1] D. Kaplan and L. Glass, *Understanding Nonlinear Dynamics* (Texts in Applied Mathematics) vol. 19. New York, NY, USA: Springer, 1995.

[2] J. S. Small, "General-purpose electronic analog computing: 1945–1965," *IEEE Ann. Hist. Comput.*, vol. 15, no. 2, pp. 8–18, Apr. 1993.

[3] A. Quarteroni and F. Saleri, *Scientific Computing With MATLAB and Octave* (Texts in Computational Science and Engineering), vol. 2. New York, NY, USA: Springer, 2006.

[4] V. V. Zhirnov, R. K. Cavin, J. A. Hutchby, and G. I. Bourianoff, "Limits to binary logic switch scaling—A gedanken model," *Proc. IEEE*, vol. 91, no. 11, pp. 1934–1939, Nov. 2003.

[5] Y. Huang, N. Guo, M. Seok, Y. Tsividis, and S. Sethumadhavan, "Analog computing in a modern context: A linear algebra accelerator case study," *IEEE Micro*, vol. 37, no. 3, pp. 30–38, Jun. 2017.

[6] N. Udayanga, S. I. Hariharan, S. Mandal, L. Belostotski, L. T. Bruton, and A. Madanayake, "Continuous-time algorithms for solving Maxwell's equations using analog circuits," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 66, no. 10, pp. 3941–3954, Oct. 2019.

[7] B. J. MacLennan, "A review of analog computing," Dept. Elect. Eng. Comput. Sci., Univ. Tennessee, Knoxville, Tennessee, Tech. Rep., UT-CS-07-601, 2007.

[8] K. H. Lundberg, "The history of analog computing: Introduction to the special section," *IEEE Control Syst.*, vol. 25, no. 3, pp. 22–25, Jun. 2005.

[9] V. P. Kodali, "A study of the applications of analog computers," *IEEE Trans. Ind. Electron. Control Instrum.*, vol. IECI-14, no. 1, pp. 1–7, Apr. 1967.

[10] J. R. Ragazzini, R. H. Randall, and F. A. Russell, "Analysis of problems in dynamics by electronic circuits," *Proc. IRE*, vol. 35, no. 5, pp. 444–452, May 1947.

[11] S. Mandal and R. Sarpeshkar, "Log-domain circuit models of chemical reactions," in *Proc. IEEE Int. Symp. Circuits Syst.*, May 2009, pp. 2697–2700.

[12] J. Liang, N. Udayanga, A. Madanayake, S. I. Hariharan, and S. Mandal, "A switched-capacitor-based analog computer for solving the 1-D wave equation," in *Proc. IEEE Int. Symp. Circuits Syst.*, Seville, Spain, Oct. 2020, pp. 1–5.

[13] E. K. F. Lee and P. G. Gulak, "A CMOS field-programmable analog array," *IEEE J. Solid-State Circuits*, vol. 26, no. 12, pp. 1860–1867, Dec. 1991.

[14] J. Becker, F. Henrici, S. Trendelenburg, M. Ortmanns, and Y. Manoli, "A field-programmable analog array of 55 digitally tunable OTAs in a hexagonal lattice," *IEEE J. Solid-State Circuits*, vol. 43, no. 12, pp. 2759–2768, Dec. 2008.

[15] A. Basu, S. Brink, C. Schlottmann, S. Ramakrishnan, C. Petre, S. Koziol, F. Baskaya, C. M. Twigg, and P. Hasler, "A floating-gate-based field-programmable analog array," *IEEE J. Solid-State Circuits*, vol. 45, no. 9, pp. 1781–1794, Sep. 2010.

[16] C. R. Schlottmann and P. E. Hasler, "A highly dense, low power, programmable analog vector-matrix multiplier: The FPAA implementation," *IEEE J. Emerg. Sel. Topics Circuits Syst.*, vol. 1, no. 3, pp. 403–411, Sep. 2011.

[17] G. E. R. Cowan, R. C. Melville, and Y. P. Tsividis, "A VLSI analog computer/digital computer accelerator," *IEEE J. Solid-State Circuits*, vol. 41, no. 1, pp. 42–53, Jan. 2006.

[18] N. Guo, Y. Huang, T. Mai, S. Patil, C. Cao, M. Seok, S. Sethumadhavan, and Y. Tsividis, "Energy-efficient hybrid analog/digital approximate computation in continuous time," *IEEE J. Solid-State Circuits*, vol. 51, no. 7, pp. 1514–1524, Jul. 2016.

[19] D. G. Luenberger, *Introduction to Dynamic Systems: Theory, Models, and Applications*. New York, NY, USA: Wiley, 1979.

[20] K. J. Åström and R. M. Murray, *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton, NJ, USA: Princeton Univ. Press, 2008.

[21] W. J. Rugh, *Linear System Theory* (Prentice-Hall Information and System Sciences Series), 2nd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 1996.

[22] H. K. Khalil, *Nonlinear Systems*, 3rd ed. Upper Saddle River, NJ, USA: Prentice-Hall, 2002.

[23] M. Schwenzer, M. Ay, T. Bergs, and D. Abel, "Review on model predictive control: An engineering perspective," *Int. J. Adv. Manuf. Technol.*, vol. 117, nos. 5–6, pp. 1327–1349, 2021.

[24] W. Thomson, "VI. Mechanical integration of the general linear differential equation of any order with variable coefficients," *Proc. Roy. Soc. London*, vol. 24, pp. 271–275, Dec. 1876.

[25] W. Thomson, "IV. Harmonic analyzer," *Proc. Roy. Soc. London*, vol. 27, pp. 371–373, Dec. 1878.

[26] C. A. Lovell, "Electrical computing system," U.S. Patent 2 476 747, Jul. 19, 1949.

[27] J. A. Lawrence and H. E. Smith, "The role of JSC engineering simulation in the Apollo program," *Simulation*, vol. 57, no. 1, pp. 9–16, Jul. 1991.

[28] C. E. Shannon, "Mathematical theory of the differential analyzer," *J. Math. Phys.*, vol. 20, nos. 1–4, pp. 337–354, Apr. 1941.

[29] N. Udayanga, A. Madanayake, S. I. Hariharan, J. Liang, S. Mandal, L. Belostotski, and L. T. Bruton, "A radio frequency analog computer for computational electromagnetics," *IEEE J. Solid-State Circuits*, vol. 56, no. 2, pp. 440–454, Feb. 2021.

[30] Y. Tsividis, "Not your father's analog computer," *IEEE Spectr.*, vol. 55, no. 2, pp. 38–43, Feb. 2018.

[31] R. J. Douglas, M. A. Mahowald, and K. A. C. Martin, "Hybrid analog-digital architectures for neuromorphic systems," in *Proc. IEEE Int. Conf. Neural Netw. (ICNN)*, Orlando, FL, USA, 1994, pp. 1848–1853.

[32] N. I. Khachab and M. Ismail, "A nonlinear CMOS analog cell for VLSI signal and information processing," *IEEE J. Solid-State Circuits*, vol. 26, no. 11, pp. 1689–1699, Nov. 1991.

[33] C. Mead, "Neuromorphic electronic systems," *Proc. IEEE*, vol. 78, no. 10, pp. 1629–1636, Oct. 1990.

[34] S. R. Zarabadi, M. Ismail, and C.-C. Hung, "High performance analog VLSI computational circuits," *IEEE J. Solid-State Circuits*, vol. 33, no. 4, pp. 644–649, Apr. 1998.

[35] M. Bucolo, A. Buscarino, L. Fortuna, and M. Frasca, "Towards analog computing devices for matrix algebraic problems," in *Proc. IEEE 12th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Arequipa, Peru, Feb. 2021, pp. 1–4.

[36] J. Liang, N. Udayanga, A. Madanayake, S. I. Hariharan, and S. Mandal, "An offset-cancelling discrete-time analog computer for solving 1-D wave equations," *IEEE J. Solid-State Circuits*, vol. 56, no. 9, pp. 2881–2894, Sep. 2021.

[37] H. Malavipathirana, S. I. Hariharan, N. Udayanga, S. Mandal, and A. Madanayake, "A fast and fully parallel analog CMOS solver for nonlinear PDEs," *IEEE Trans. Circuits Syst. I, Reg. Papers*, vol. 68, no. 8, pp. 3363–3376, Aug. 2021.

[38] A. L. Fitch, H. H. C. Iu, and D. D. C. Lu, "An analog computer for electronic engineering education," *IEEE Trans. Educ.*, vol. 54, no. 4, pp. 550–557, Nov. 2011.

[39] L. Chua and G.-N. Lin, "Nonlinear programming without computation," *IEEE Trans. Circuits Syst.*, vol. CS-31, no. 2, pp. 182–188, Feb. 1984.

[40] B. Gilbert, "A monolithic microsystem for analog synthesis of trigonometric functions and their inverses," *IEEE J. Solid-State Circuits*, vol. JSSC-17, no. 6, pp. 1179–1191, Dec. 1982.

[41] E. Seevinck and G. Renkema, "A 4-quadrant cosine-synthesis circuit," in *Proc. IEEE Int. Solid-State Circuits Conference. Dig. Tech. Papers*, Feb. 1982, pp. 40–41.

[42] M. Zareie, M. Hosseinnejad, and S. J. Azhari, "Ultra-Low-power, wide dynamic range front-end logarithmic amplifier for biomedical applications," in *Proc. 5th Conf. Knowl. Based Eng. Innov. (KBEI)*, Feb. 2019, pp. 548–552.

[43] K. Koli and K. Halonen, "A 2.5 V temperature compensated CMOS logarithmic amplifier," in *Proc. Custom Integr. Circuits Conf.*, 1997, pp. 79–82.

[44] K. M. Al-Tamimi and M. A. Al-Absi, "A 6.13 μW and 96 dB CMOS exponential generator," *IEEE Trans. Very Large Scale Integr. (VLSI) Syst.*, vol. 22, pp. 2443–2447, Nov. 2014.

[45] T. C. Carusone, D. Johns, K. W. Martin, and D. Johns, *Analog Integrated Circuit Design*, 2nd ed. Hoboken, NJ, USA: Wiley, 2012.

[46] P. Horowitz and W. Hill, *The Art of Electronics*, 3rd ed. New York, NY, USA: Cambridge Univ. Press, 2022.

[47] P. R. Gray, *Analysis and Design of Analog Integrated Circuits*, 5th ed. New York, NY, USA: Wiley, 2009.

[48] K. B. Pandya and K. A. Shah, "Performance evaluation of different types of CMOS operational transconductance amplifier," *Int. J. Sci. Res.*, vol. 2, no. 3, pp. 91–96, 2013.

[49] B. Razavi, *Design of Analog CMOS Integrated Circuits*, 2nd ed. New York, NY, USA: McGraw-Hill, 2017.

[50] B. Gilbert, "A precise four-quadrant multiplier with subnanosecond response," *IEEE Solid-State Circuits Newslett.*, vol. 12, no. 4, pp. 29–37, Dec. 2007.

[51] J. N. Babanezhad and G. C. Temes, "A 20-V four-quadrant CMOS analog multiplier," *IEEE J. Solid-State Circuits*, vol. JSSC-20, no. 6, pp. 1158–1168, Dec. 1985.

[52] G. Han and E. Sanchez-Sinencio, "CMOS transconductance multipliers: A tutorial," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 45, no. 12, pp. 1550–1563, Dec. 1998.

[53] S. M. Rezaul Hasan, "A novel wide-swing wide-bandwidth scalable low-voltage analog CMOS multiplier for communication signal processing," in *Proc. IEEE Region Conf.*, Nov. 2005, pp. 1–4.

[54] S. A. Mahmoud, "Low voltage low power wide range fully differential CMOS four-quadrant analog multiplier," in *Proc. 52nd IEEE Int. Midwest Symp. Circuits Syst.*, Cancun, Mexico, Aug. 2009, pp. 130–133.

[55] A. Satapathy, S. K. Maity, and S. K. Mandal, "A flipped voltage follower based analog multiplier in 90 nm CMOS process," in *Proc. Int. Conf. Adv. Comput. Eng. Appl.*, Ghaziabad, India, Mar. 2015, pp. 628–631.

[56] A. J. S. De Sousa, F. M. Cardoso, K. K. C. F. Nunes, F. S. De Andrade, G. C. Goncalves, E. P. Santana, and A. I. A. Cunha, "A very compact CMOS analog multiplier for application in CNN synapses," in *Proc. IEEE 10th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Armenia, Colombia, Feb. 2019, pp. 241–244.

[57] A. J. S. De Sousa, F. De Andrade, H. D. Santos, G. Goncalves, M. D. Pereira, E. Santana, and A. I. Cunha, "CMOS analog four-quadrant multiplier free of voltage reference generators," in *Proc. 32nd Symp. Integr. Circuits Syst. Design*, Aug. 2019, pp. 1–6.

[58] F. M. Cardoso, M. C. Schneider, and E. P. Santana, "CMOS analog multiplier with high rejection of power supply ripple," in *Proc. IEEE 9th Latin Amer. Symp. Circuits Syst. (LASCAS)*, Feb. 2018, pp. 1–4.

[59] A. Cracan, G. Bonteanu, and R.-G. Bozomitu, "A weak-inversion CMOS analog multiplier/divider circuit," in *Proc. IEEE 24th Int. Symp. Design Technol. Electron. Packag.*, Oct. 2018, pp. 261–264.

[60] S. Hiratkar, A. Tijare, and P. Dakhole, "VLSI design of analog multiplier in weak inversion region," in *Proc. Int. Conf. Commun. Signal Process. (ICCSP)*, Tamilnadu, India, Apr. 2016, pp. 0832–0835.

[61] S. Nikseresht, S. Javad Azhari, and M. Danesh, "High bandwidth four-quadrant analog multiplier," in *Proc. Iranian Conf. Electr. Eng. (ICEE)*, Tehran, Iran, May 2017, pp. 210–215.

[62] Z. Czarnul, "Modification of Banu–Tsividis continuous-time integrator structure," *IEEE Trans. Circuits Syst.*, vol. CS-33, no. 7, pp. 714–716, Jul. 1986.

[63] M. Ismail, "Four-transistor continuous-time MOS transconductor," *Electron. Lett.*, vol. 23, no. 20, p. 1099, 1987.

[64] M. Ismail, R. Brannen, S. Takagi, R. Khan, O. Aaserud, N. Fujii, and N. Khachab, "A configurable CMOS multiplier/divider for analog VLSI," in *Proc. IEEE Int. Symp. Circuits Syst.*, Chicago, IL, USA, 1993, pp. 1085–1088.

[65] B. Song, "CMOS RF circuits for data communications applications," *IEEE J. Solid-State Circuits*, vol. JSSC-21, no. 2, pp. 310–317, Apr. 1986.

[66] Y. Tsividis, M. Banu, and J. Khoury, "Continuous-time MOSFET-C filters in VLSI," *IEEE Trans. Circuits Syst.*, vol. CS-33, no. 2, pp. 125–140, Feb. 1986.

[67] A. R. Bonaccio, "Differential source follower with body effect compensation," U.S. Patent 5 874 840, Feb. 23, 1999.

[68] I. Hwang, D. Kim, J.-H. Roh, M.-K. Lee, S.-P. Nah, and M. Song, "An 8-b cascaded folding A/D converter with a new fully differential source follower," in *Proc. Eur. Conf. Circuit Theory Design (ECCTD)*, Dresden, Germany, Sep. 2013, pp. 1–4.

[69] B. Van der Pol, "LXXXVIII. On 'relaxation-oscillations,'" *London, Edinburgh, Dublin Philos. Mag. J. Sci.*, vol. 2, pp. 978–992, Nov. 1926.

[70] B. van der Pol and J. van der Mark, "LXXII. The heartbeat considered as a relaxation oscillation, and an electrical model of the heart," *London, Edinburgh, Dublin Phil. Mag. J. Sci.*, vol. 6, no. 38, pp. 763–775, Nov. 1928.

[71] K. Yi Chen, P. D. Biernacki, A. Lahrichi, and A. Mickelson, "Analysis of an experimental technique for determining van der pol parameters of a transistor oscillator," *IEEE Trans. Microw. Theory Techn.*, vol. 46, no. 7, pp. 914–922, Jul. 1998.

[72] J. Casaleiro, L. B. Oliveira, and I. M. Filanovsky, "Quadrature error of two-integrator oscillators," in *Proc. IEEE 57th Int. Midwest Symp. Circuits Syst. (MWSCAS)*, College Station, TX, USA, Aug. 2014, pp. 483–486.

[73] A. J. Lotka, "Contribution to the theory of periodic reactions," *J. Phys. Chem.*, vol. 14, no. 3, pp. 271–274, Mar. 1910.

[74] H.-C. Hung, Y.-C. Chiu, and M.-C. Wu, "A modified Lotka–Volterra model for diffusion and substitution of multigeneration DRAM processing technologies," *Math. Problems Eng.*, vol. 2017, pp. 1–12, Jan. 2017.

[75] E. N. Lorenz, "Deterministic nonperiodic flow," *J. Atmos. Sci.*, vol. 20, no. 2, pp. 130–141, Mar. 1963.

[76] R. M. Murray, Z. Li, and S. S. Sastry, *A Mathematical Introduction to Robotic Manipulation*, 1st ed. Boca Raton, FL, USA: CRC Press, Dec. 2017.

[77] K. M. Lynch and F. C. Park, *Modern Robotics: Mechanics, Planning, and Control*. Cambridge, U.K.: Cambridge Univ. Press, 2017.

[78] A. M. Soliman, "Two integrator loop quadrature oscillators: A review," *J. Adv. Res.*, vol. 4, no. 1, pp. 1–11, Jan. 2013.

[79] M. A. Al-Absi, A. Hussein, and M. T. Abuelma'atti, "A low voltage and low power current-mode field programmable analog computational unit," in *Proc. 2nd Middle East Conf. Biomed. Eng.*, Doha, Qatar, Feb. 2014, pp. 51–54.

[80] R. W. Brockett, "Dynamical systems that sort lists, diagonalize matrices, and solve linear programming problems," *Linear Algebra its Appl.*, vol. 146, pp. 79–91, Feb. 1991.

[81] R. W. Brockett, "A rational flow for the Toda lattice equations," in *Operators, Systems and Linear Algebra*, L. Arkeryd, H. Engl, A. Fasano, R. M. M. Mattheij, P. Neittaanmaki, H. Neunzert, U. Helmke, D. Prätzel-Wolters, and E. Zerz, Eds. Wiesbaden, Germany: Vieweg+Teubner Verlag, 1997, pp. 33–44.

[82] J. Platt and A. Barr, "Constrained differential optimization," in *Proc. Neural Inf. Process. Syst.*, 1987, pp. 1–10.

[83] J. Platt, "Analog circuits for constrained optimization," in *Proc. Adv. Neural Inf. Process. Syst.*, vol. 2. San Francisco, CA, USA: Morgan Kaufmann, 1989, pp. 1–8.

[84] S. Zhang and A. G. Constantinides, "Lagrange programming neural networks," *IEEE Trans. Circuits Syst. II, Analog Digit. Signal Process.*, vol. 39, no. 7, pp. 441–452, Jul. 1992.

[85] K. L. Moore and S. Naidu, "Linear quadratic regulation using neural networks," in *Proc. IJCNN-91-Seattle Int. Joint Conf. Neural Netw.*, Seattle, WA, USA, 1991, pp. 735–739.

[86] R.-L. Yang and C.-P. Wu, "Neural networks for optimal control of nonlinear systems with bound constraints," in *Proc. Int. Conf. Neural Netw.*, Seattle, WA, Australia, 1995, pp. 2342–2347.

[87] H.-C. Cheng, S. Su, M. Palaria, Q. Zhang, C. Yang, S. Hossain, R. Bena, B. Chen, Z. Liu, J. Liu, R. Rasul, Q. Nguyen, W. Wu, and M. S.-W. Chen, "A memristor-based analog accelerator for solving quadratic programming problems," in *Proc. IEEE Custom Integr. Circuits Conf. (CICC)*, San Antonio, TX, USA, Apr. 2023, pp. 1–2.

[88] A. Gangopadhyay, O. Chatterjee, and S. Chakrabartty, "Continuous-time optimization using sub-threshold current-mode growth transform circuits," in *Proc. IEEE 61st Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Windsor, ON, Canada, Aug. 2018, pp. 246–249.

[89] K. M. Deems, "High speed analog circuit for solving optimization problems," Ph.D. thesis, EECS Dept., Univ. California, Berkeley, CA, USA, 2015.

[90] M. P. Kennedy and L. O. Chua, "Neural networks for nonlinear programming," *IEEE Trans. Circuits Syst.*, vol. 35, no. 5, pp. 554–562, May 1988.

[91] S. Vichik, "Quadratic and linear optimization with analog circuits," Ph.D. thesis, Dept. Mech. Eng., UC Berkeley, Berkeley, CA, USA, 2015.

[92] D. Tank and J. Hopfield, "Simple 'neural' optimization networks: An A/D converter, signal decision circuit, and a linear programming circuit," *IEEE Trans. Circuits Syst.*, vol. CS-33, no. 5, pp. 533–541, May 1986.

[93] S. Vichik and F. Borrelli, "Fast solution of linear and quadratic programs with an analog circuit," in *Proc. Amer. Control Conf.*, Portland, OR, USA, Jun. 2014, pp. 2954–2959.

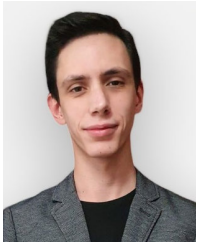
[94] S. Vichik and F. Borrelli, "Solving linear and quadratic programs with an analog circuit," *Comput. Chem. Eng.*, vol. 70, pp. 160–171, Nov. 2014.

[95] M. Athans and P. L. Falb, *Optimal Control: An Introduction to the Theory and Its Applications*. New York, NY, USA: Dover, 2007.

[96] D. S. Naidu, *Optimal Control Systems* (Electrical Engineering Textbook Series), Boca Raton, FL, USA: CRC Press, 2003.

[97] L. Wang, *Model Predictive Control System Design and Implementation Using MATLAB* (Advances in Industrial Control). London, U.K.: Springer, 2009.

[98] F. Borrelli, A. Bemporad, and M. Morari, *Predictive Control for Linear and Hybrid Systems*. Cambridge, U.K.: Cambridge Univ. Press, 2017.
[99] J. M. Quero, E. F. Camacho, and L. G. Franquelo, "Neural network for constrained predictive control," *IEEE Trans. Circuits Syst. I, Fundam. Theory Appl.*, vol. 40, no. 9, pp. 621–626, Sep. 1993.
[100] J. Quero and E. Camacho, "Neural generalized predictive self-tuning controllers," in *Proc. IEEE Int. Conf. Syst. Eng.*, Pittsburgh, PA, USA, Aug. 1990, pp. 160–163.
[101] J. N. Bruno, F. D. Moran, H. I. Khajanchi, and A. A. Adegbege, "Analog solver for embedded model predictive control with application to quadruple tank system," in *Proc. Amer. Control Conf. (ACC)*, New Orleans, LA, USA, May 2021, pp. 4680–4685.
[102] T. Skibik and A. A. Adegbege, "An architecture for analog VLSI implementation of embedded model predictive control," in *Proc. Annu. Amer. Control Conf. (ACC)*, Milwaukee, WI, USA, Jun. 2018, pp. 4676–4681.
[103] R. M. Levenson and A. A. Adegbege, "Analog circuit for real-time optimization of constrained control," in *Proc. Amer. Control Conf. (ACC)*, Boston, MA, USA, Jul. 2016, pp. 6947–6952.
[104] T. Skibik and A. A. Adegbege, "Rapid prototyping of constrained linear quadratic optimal control with field programmable analog array," in *Proc. IEEE Conf. Decis. Control (CDC)*, Miami Beach, FL, USA, Dec. 2018, pp. 1864–1869.
[105] M.-S. Lan and S. Chand, "Solving linear quadratic discrete-time optimal controls using neural networks," in *Proc. 29th IEEE Conf. Decis. Control*, Honolulu, HI, USA, Dec. 1990, pp. 2770–2772.



JOÃO C. G. ARAÚJO received the B.S. and M.S. degrees in physics engineering from NOVA University Lisbon, in 2019 and 2022, respectively. He is currently pursuing the Ph.D. degree in electrical and computer engineering with CTS-UNINOVA, NOVA School of Science and Technology, Portugal.

LUIS B. OLIVEIRA (Senior Member, IEEE) was born in Lisbon, Portugal, in 1979. He received the degree in electrical and computer engineering and the Ph.D. degree from Instituto Superior Técnico, Technical University of Lisbon, in 2002 and 2007, respectively.

Since 2001, he has been a member of the Analog and Mixed-Signal Circuits Group, INESC-ID. Although his research work has been done mainly at INESC-ID, he has had intense collaboration with TUDelft, The Netherlands, and the University of Alberta, Canada. Since 2007, he has been with the Department of Electrical and Computer Engineering, NOVA School of Science and Technology, where he is currently an Associate Professor. He is currently a Researcher with CTS-UNINOVA. He has more than 100 publications in international journals and conferences and is the coauthor of two books: "*Analysis and Design of Quadrature Oscillators*" (Springer, 2008) and "*Quadrature RC-Oscillators: The van der Pol Approach*" (Springer 2018). His current research interests include RF oscillators, PLLs, LNAs, mixers, and other building blocks for RF integrated transceivers.

Prof. Oliveira has been a member of the IEEE CASS Analog Signal Processing Technical Committee, ASPTC, (the largest within CASS), since 2008. He was the Chair of the Broadcast Technology Society, the Circuits and Systems Society, the Consumer Electronics Society, and the IEEE Joint Chapter, between 2016 and 2019. He was an Associate Editor of IEEE TRANSACTIONS ON CIRCUITS AND SYSTEMS—II: EXPRESS BRIEFS, between 2018 and 2023.

BRUNO J. GUERREIRO (Member, IEEE) received the B.S. and Ph.D. degrees in electrical and computer engineering from Instituto Superior Técnico, Lisbon, Portugal, in 2004 and 2013, respectively.

He is currently an Assistant Professor with the NOVA School of Science and Technology, NOVA University Lisbon (FCT/UNL), and also a Researcher with CTS/Uninova/LASI and ISR-Lisbon/LARSYS. He is the coauthor of more than 46 publications, 12 of which are in Q1-SCOPUS journals, with more than 515 citations and an H-index of 13 (Google Scholar). He was a team member in more than 14 national, EU, and Macanese funded research and innovation projects, and coordinator in two of them, participating in the development of ten autonomous vehicles prototypes (two marine and eight aerial). His research interests include theoretical, practical, and technological challenges related to the use of aerospace robotics for civilian applications, with particular emphasis on sensor-based SLAM, model-based predictive control, trajectory planning and tracking, linear and nonlinear control, automatic LIDAR calibration, and modeling and instrumentation of aerospace and marine vehicles

FILIPE FERREIRA DA SILVA received the B.S. and M.S. degrees in chemical engineering from NOVA University Lisbon, the Ph.D. degree in physics from the University of Innsbruck, Austria, in 2009, and the Habilitation degree in molecular physics, in 2022.

From 2009 to 2018, he was a Postdoctoral Researcher with the University of Paris-Sud and NOVA University Lisbon. He is an Associate Professor with the Physics Department, NOVA School of Science and Technology, NOVA University Lisbon. He is the author of three chapter books and more than 125 articles, leading to a H-index of 22. His research interests include electron interactions with isolated molecules in the gas phase, electron scattering processes, mass spectrometry, and innovation for electron interactions and mass spectrometry applications.