

Brain-Inspired Synaptic Resistor Circuits for Self-Programming Intelligent Systems

Yong Chen

Unlike artificial intelligent systems based on computers, which need to be preprogrammed for specific tasks, restricting their functions to their preprogrammed ranges, the human brain does not need to be preprogrammed, and has general intelligence to create new tactics in complex and erratic environments. The basic element in the brain, a synapse, has the function to process and learn from signals in real time by following Hebb's rule, which is a critical function missing from the transistor, the basic device in computers. In this work, a computing circuit based on synaptic resistors (synstors) with signal processing and Hebbian learning functions is modeled and analyzed. A synstor circuit emulates a neurobiological network to concurrently execute signal processing and learning algorithms in parallel mode, does not need to be preprogrammed, and has the capability to optimize and create new algorithms in complex and erratic environments with speed and energy efficiency significantly superior to those of existing computing circuits. The synstor circuit can potentially circumvent the fundamental limitations of existing computing circuits, leading to a new computing platform with real-time self-programming functionality and general intelligence in complex and erratic environments.

1. Introduction


After Turing proposed his model for a programmable computer, he also forecasted that “by modifying this computer to have an adequate storage, suitably increasing its speed of action, and providing it with appropriate programs,” the computer will “eventually compete with men in all purely intellectual fields.”^[1] Based on the Turing model, computers can be programmed to implement arbitrary algorithms on transistors—the basic devices in computers. Following Moore's law, the miniaturization of transistors has exponentially improved the scale, memory capacity, speed, and energy efficiency of computing circuits,^[2] leading

to an information revolution and artificial intelligent systems that can learn and surpass human performance in specific tasks,^[3] such as pattern recognition^[4] and the Go game.^[5] However, the computing speeds and energy efficiencies are asymptotically saturated when transistors approach the limit of their minimal sizes near the end of Moore's law.^[2,6] On the other hand, the time and energy consumption to execute learning algorithms in computers from a dataset with M -dimensional variables increase versus M exponentially,^[7] which is referred to as the “curse of dimensionality.”^[8] The exponentially increasing data volumes and explosive computing requirements for machine learning led to the development of computing circuits, such as the Summit supercomputer,^[9] graphics processing units (GPUs),^[10] tensor processing units (TPUs),^[5] field-programmable gate arrays (FPGAs),^[11] TrueNorth,^[12] and Tianjic^[13] neuromorphic circuits, with high speeds

and improved parallel computing architectures and energy efficiencies. However, the transistor-based computing circuits execute algorithms on physically separated logic and memory transistors, and the computing energy efficiency is predominantly limited by data transmissions between logic and memory transistors ($\approx 10^{-11}$ J/bit⁻¹),^[14] and computing energy efficiencies ($\approx 10^{10} - 10^{13}$ OPS/W (operations per second per watt)),^[2,5,9-13] significantly lower than that of the human brain ($\approx 10^{15}$ OPS/W).^[15] Due to the limitations on the computing speeds and power consumption, computing circuits embedded in artificial intelligent systems can hardly calculate learning algorithms on-site based on real-time sensing signals from complex environments. Artificial intelligent systems need to be preprogrammed for specific tasks by collecting “big data,” and then perform learning from the data saved in off-site computing circuits with high speeds, high power consumption, and bulky volumes,^[2,5,11-13] resulting in their failures in arbitrary and erratic environments beyond their preprogrammed domains.^[3]

The human brain has long served as the inspiration for the development of intelligent systems. The human brain concurrently executes its signal-processing and learning algorithms in massively parallel mode via a network of neurons connected by $\approx 10^{14}$ synapses.^[15] In a neural network with M presynaptic and N postsynaptic neurons connected by $M \times N$ synapses (Figure 1), a train of voltage pulses, $x_m(t)$, from the m th presynaptic neuron is processed by a synapse connecting the m th

Prof. Y. Chen
Department of Mechanical and Aerospace Engineering
University of California, Los Angeles
420 Westwood Plaza, Los Angeles, CA 90095, USA
E-mail: yongchen@seas.ucla.edu

 The ORCID identification number(s) for the author(s) of this article can be found under <https://doi.org/10.1002/aisy.202000219>.

© 2020 The Authors. Advanced Intelligent Systems published by Wiley-VCH GmbH. This is an open access article under the terms of the Creative Commons Attribution License, which permits use, distribution and reproduction in any medium, provided the original work is properly cited.

DOI: 10.1002/aisy.202000219

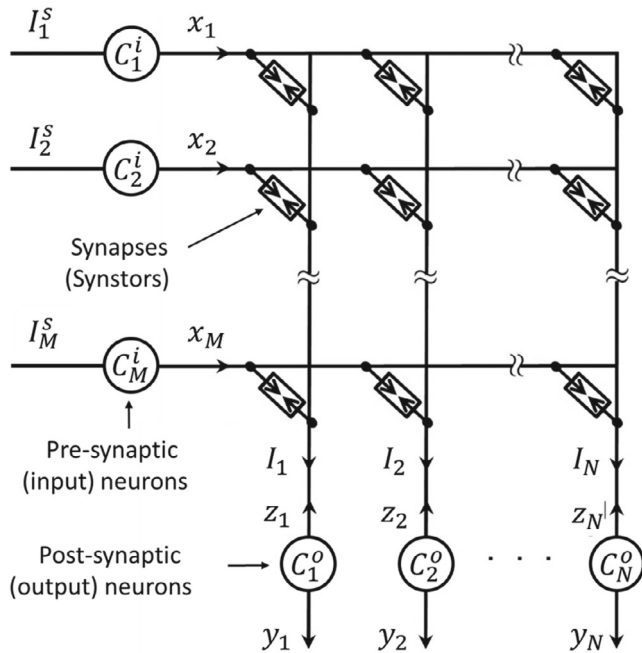


Figure 1. A circuit of $M \times N$ synapses (synstors) connected with M presynaptic (input) neurons ($C_1^i, C_2^i, \dots, C_M^i$) and N postsynaptic (output) neurons ($C_1^o, C_2^o, \dots, C_N^o$). I_m^S denotes a current flowing into the m th presynaptic (input) neuron, x_m denotes input voltage pulses triggered from the m th presynaptic (input) neuron, I_n denotes a current flowing through synapses (synstors) into the n th postsynaptic (output) neuron, y_n denotes voltage pulses from the n th postsynaptic (output) neuron, and z_n denotes feedback voltage pulses from the n th (output) neuron.

presynaptic and the n th postsynaptic neuron, and induces a current in the n th postsynaptic neuron,^[16] $I_{nm} = \kappa * (w_{nm}x_m)$, where w_{nm} denotes the synaptic weight (conductance), κ denotes a temporal kernel function, and $\kappa * (w_{nm}x_m) = \int_0^t \kappa(t-t')w_{nm}(t')x_m(t')dt'$ represents the temporal convolution between κ and $w_{nm}x_m$. For signal processing, a spatiotemporal wave of voltage pulses in M presynaptic neurons, \mathbf{x} , induces a collective current via synapses in the N postsynaptic neurons,

$$\mathbf{I} = \kappa * (\mathbf{w}\mathbf{x}) \quad (1)$$

where \mathbf{w} denotes a $(w_{nm})_{N \times M}$ matrix, \mathbf{x} denotes the $(x_m)_{M \times 1}$ vector, and \mathbf{I} denotes the $(I_n)_{N \times 1}$ vector. The current $\mathbf{I}(t)$ induces a train of voltage pulses, $\mathbf{y}(t)$, in the N postsynaptic neurons with \mathbf{y} as the $(y_n)_{N \times 1}$ vector. When a voltage pulse is fired in the postsynaptic neuron ($y_n \neq 0$), the postsynaptic current in the neuron, $I_n = 0$. x_m and y_n voltage pulses have the same amplitudes.

The learning rule in neurobiological networks was originally postulated by Hebb,^[5] and spike-timing-dependent plasticity (STDP) was discovered experimentally.^[6] A synapse is modulated by voltage pulses firing concurrently in pre- and postsynaptic neurons connected by the synapse by following the learning rule, $\dot{w}_{nm} = \alpha z_n x_m$, where α denotes the conductance modification coefficient and z_n is a function of y_n . For Hebbian learning, $z_n = y_n$, and $\dot{w}_{nm} = \alpha y_n x_m$; for anti-Hebbian learning, $z_n = -y_n$, and $\dot{w}_{nm} = -\alpha y_n x_m$. For the STDP and anti-STDP learning rules, $z_n(t)$ is a function of the timing difference

between x_m and y_n pulses (Equation S1, Supporting Information). In a neural network, the synaptic weight matrix, \mathbf{w} , is modified by the spatiotemporal waves of voltage pulses in the presynaptic neurons, \mathbf{x} , and postsynaptic neurons, \mathbf{z} , for learning.^[17]

$$\dot{\mathbf{w}} = \alpha \mathbf{z} \otimes \mathbf{x} \quad (2)$$

where $\mathbf{z} \otimes \mathbf{x}$ represents the outer product between \mathbf{z} and \mathbf{x} . Following Equation (2), when $z_n \cdot x_m = 0$ (e.g., $x_m \neq 0$ and $z_n = 0$ for signal processing), $\dot{w}_{nm} = 0$; i.e., w_{nm} remains nonvolatile for memory. By integrating the analog convolutional signal processing, learning, and memory functions in a single synapse, the brain concurrently executes the signal processing (Equation (1)) and learning (Equation (2)) algorithms in a neural network in analog parallel mode with an estimated speed ($\approx 10^{16}$ OPS)^[15] comparable to the speed ($\approx 10^{17}$ floating-point OPS) of the fastest supercomputer, Summit,^[9] but consumes much less power (≈ 30 W) than the supercomputer ($\approx 10^7$ W), and has much smaller volume ($\approx 10^{-3}$ m³) than the supercomputer ($\approx 10^4$ m³). When synapses receive and process signals, they are simultaneously modified in a parallel learning process to dynamically optimize and create new tactics in the human brain. The unique real-time learning function in each synapse facilitates self-programming capability in the human brain and its general intelligence in complex and erratic environments.

Novel neuromorphic devices, such as floating-gate transistors,^[18,19] ferroelectric transistors,^[20] memristors,^[21,22] and phase change memory,^[23,24] have been developed to emulate synapses by integrating logic and memory functions in a single device. By circumventing the data transmissions between logic and memory, neuromorphic circuits based on these devices processed signals with energy efficiencies ($\sim 10^{10} - 10^{14}$ OPS/W)^[18-20,22,23] significantly higher than those of transistor-based neuromorphic circuits ($\sim 10^{10} - 10^{12}$ OPS/W).^[2,5,10-13,25,26] Although learning algorithms such as STDP were executed on individual devices by applying tailored voltage signals,^[19,21] learning algorithms needed to be executed in external transistor-based computing circuits to derive desirable device conductances; then the devices were preprogrammed to the conductance values in iterative writing and reading processes with much lower speeds ($\sim 10^2 - 10^5$ OPS) and energy efficiencies ($< 10^{10}$ OPS/W) than those for signal processing.^[19,21] To avoid the change of conductance after the writing process, the magnitudes of voltage pulses for signal processing were decreased below the magnitudes of voltage pulses for writing. When the signal processing algorithm was executed in the circuits, the writing process was interrupted, and vice versa.^[18-22,23] Intrinsically, these neuromorphic devices lack the synaptic function to process and learn from signals with the same magnitude; therefore, the circuits based on these devices cannot compute signal processing and learning algorithms (Equation (1) and (2)) concurrently, and lack the real-time learning and self-programming functionality of the human brain and its general intelligence in complex and erratic environments.

Recently we have developed a synaptic resistor, abbreviated as synstor hereinafter, to emulate a synapse to process and learn from voltage pulses with the same magnitudes.^[27] By transmitting spatiotemporal waves of voltage pulses with the same

amplitude in a synstor circuit, the spatiotemporal convolutional signal processing (Equation (1)) and Hebbian learning algorithms (Equation (2)) can be executed concurrently in parallel analog mode. In the synstor circuit, the optimal synstor conductances are not derived by computing learning algorithms based on collected data, and iterative writing and reading processes, but achieved by a self-programming process via real-time learning. A 4×2 crossbar synstor circuit was demonstrated to execute speech signal processing and compute learning algorithms concurrently in analog parallel mode with an energy efficiency of $\approx 1.6 \times 10^{17}$ OPS/W,^[27] which is higher than the energy efficiencies of the human brain ($\approx 10^{15}$ OPS/W),^[15] computing circuits based on transistors ($\approx 10^{10} - 10^{12}$ OPS/W),^[2,5,9,11-13,26] and other neuromorphic devices ($\approx 10^{10} - 10^{14}$ OPS/W).^[18-20,22,23] In this study, we focus on a theoretical model of synstor circuits that can emulate a neurobiological network to concurrently calculate signal processing (Equation (1)) and learning algorithms (Equation (2)) in parallel analog mode. The computing mechanism, scale, speed, energy efficiency, performance density, and limitations of the synstor circuit will be analyzed in comparison with other computing circuits and neurobiological networks. A mathematical model will be established to describe and analyze the real-time learning process and self-programming function of the synstor circuits to optimize and create new algorithms in complex and erratic environments.

2. Synstor Circuit Models

2.1. Signal Processing and Learning in Synstors

Previously we have reported an electronic device, the synstor, to emulate a synapse.^[27] A synstor is composed of a semiconducting carbon nanotube (CNT) channel which forms Schottky contacts with Al input and output electrodes as a resistor, and a recessed TiO₂ charge storage layer embedded in a HfO₂ dielectric layer sandwiched between an Al reference electrode and the CNT channel as a capacitor (Figure S1, Supporting Information). The reference electrode is always electrically grounded during the operation. For signal processing (Figure S1a, Supporting Information), a synstor processes a series of voltage pulses, $x(t)$, on its input electrode by charging the capacitor during the pulses, and discharging the capacitor after the pulses, and triggering a current via the resistor, $I(t) = \kappa * (wx)$, on its grounded output electrode ($z = 0$) as a convolution of $x(t)$ and the product of its conductance, w , and a kernel function $\kappa(t)$. As shown in Figure S1b, Supporting Information, when a series of paired $x(t)$ and $z(t)$ voltage pulses with the same amplitude (i.e., $x = z$) are applied on the synstor simultaneously, w is modified by following the Hebb's learning rule, $\dot{w} = \alpha xz$, where α is a nonlinear function of the amplitudes and numbers of x and z pulses. The paired negative (positive) pulses generate a potential difference between the channel and charge storage layer to increase (decrease) the electronic charge stored in the charge storage layer, which in turn attracts (repels) the holes in the semiconducting channel to increase (decrease) its conductance with $\alpha > 0$ ($\alpha < 0$). Otherwise, when a synstor experiences x and z pulses under the condition $xz = 0$, the x or z potential mainly drops beyond the charge storage layer and reference electrode,

and the magnitudes of the potential differences between the channel and the recessed charge storage layer are below the threshold values to modify the charge stored in the charge storage layer; thus $\dot{w} = 0$ for nonvolatile memory. The analog convolutional signal processing, Hebbian learning, and nonvolatile memory functions have been integrated in a single synstor to emulate a synapse to process and learn from voltage pulses with the same magnitudes, which facilitates the concurrent signal processing and learning in synstor circuits.

2.2. Concurrent Signal Processing and Learning in Synstor Circuits

A circuit composed of $M \times N$ synstors connected with M input and N output neuron circuits is shown in Figure 1. For signal processing, sensory or output signals from the previous layer of circuits, I^s , are transmitted to M input neuron circuits, C^i , to trigger a wave of voltage pulses, x , input to the circuit. x induces a collective current flowing from synstors into the N output neuron circuits, $I = \kappa * (wx)$ (Equation (1)), where w denotes a $(w_{nm})_{N \times M}$ matrix of synstor conductances, I denotes the $(I_n)_{N \times 1}$ vector with I_n as the current flowing into n^{th} output neuron circuit, and κ denotes a convolutional kernel function of the synstors. The current $I(t)$ flows into N output neuron circuits, C^o , which generates forward-propagating output voltage pulses, $y(t)$, and back-propagating feedback voltage pulses, $z(t)$, on the N output electrodes connected with the output neuron circuits. When a voltage pulse is fired in the n^{th} output electrode ($z_n \neq 0$), the n^{th} output electrode is disconnected from the neuron circuit connected with the output electrode, and thus the current flowing into the neuron circuit, $I_n = 0$. For learning, the synstor conductance matrix, w , is modified by the spatiotemporal waves of voltage pulses, x , on the input electrodes and, z , on the output electrodes by following Equation (2), $\dot{w} = \alpha z \otimes x$. In the circuit, a synstor connected with the m^{th} input and n^{th} output neuron circuits experiences various combinations of x_m and z_n voltage pulses: 1) When $x_m \neq 0$ and $z_n = 0$, a current $I_{mn}(t) = \kappa * (w_{nm}x_m) \neq 0$ (Equation (1)) is triggered via the synstors connected with the n^{th} output neuron circuit for signal processing, and $\dot{w}_{nm} = \alpha z_n x_m = 0$ (Equation (2)) for learning. 2) When $x_m = z_n \neq 0$, $I_{mn} = 0$ for signal processing, and $\dot{w}_{nm} = \alpha z_n x_m \neq 0$ for learning. 3) Under all other conditions including $x_m = z_n = 0$, and $z_n \neq 0$ under $x_m = 0$, then $I_{mn} = 0$ for signal processing, and $\dot{w}_{nm} = 0$ for learning. Therefore, each synstor in the circuit simultaneously processes x and learns from x and z , and the signal processing algorithm $I(t) = \kappa * (wx)$ (Equation (1)) and the learning algorithm $\dot{w} = \alpha z \otimes x$ (Equation (2)) are executed concurrently in the synstor circuit without interrupting each other.

2.3. Self-Programming in Synstor Circuits

The learning algorithm, $\dot{w} = \alpha z \otimes x$ (Equation (2)), implemented in a synstor circuit can be viewed as a generic Hebb's learning rule. In principle, all major machine learning algorithms, including unsupervised, supervised, and reinforcement learning, can be implemented in synstor circuits based on $\dot{w} = \alpha z \otimes x$ by setting $z = f(w, x, y)$.^[28] The Hebbian or anti-Hebbian learning

algorithm can be implemented in the synstor circuit by setting $z = y$. When an output voltage pulse, $y_n(t) = \delta(t - t_n)$, is triggered from the n th output neuron circuit at $t = t_n$, a negative (positive) feedback pulse, $z_n(t) = \delta(t - t_n)$, is triggered from the n th output neuron circuit simultaneously at $t = t_n$. Substituting z in Equation (2) by y yields $\dot{w} = \alpha y \otimes x$, with $\alpha > 0$ ($\alpha < 0$) for Hebbian (anti-Hebbian) learning. The STDP or anti-STDP learning algorithms can be implemented in the synstor circuit by setting $z = y * \tilde{\theta}$ (Equation S1, Supporting

Information) with $\tilde{\theta}(t) = \begin{cases} -e^{t/\tau_-}/\tau_- & \text{when } t < 0 \\ 0 & \text{when } t = 0 \\ e^{-t/\tau_+}/\tau_+ & \text{when } t > 0 \end{cases}$. When an

output voltage pulse, $y_n(t) = \delta(t - t_n)$, is triggered from the n th output neuron circuit at $t = t_n$, a train of positive (negative) feedback pulses with a pulse firing rate proportional to $e^{(t-t_n)/\tau_-}/\tau_-$ under $t < t_n$ and a train of negative (positive) feedback pulses with a pulse firing rate proportional to $e^{-(t-t_n)/\tau_+}/\tau_+$ under $t > t_n$ are triggered from the n th output neuron circuit on the output electrode to implement STDP (anti-STDP) algorithms. By substituting $z = y * \tilde{\theta}$ in Equation (2),

$$\dot{w}_{nm} = \sum_{t_n} \begin{cases} -\alpha x_m(t) e^{t-t_n}/\tau_- & \text{when } t < t_n \\ 0 & \text{when } t = t_n \\ \alpha x_m(t) e^{-(t-t_n)/\tau_+} & \text{when } t > t_n \end{cases} \quad \text{with } \alpha > 0 \text{ for}$$

STDP learning algorithm, and $\alpha < 0$ for anti-STDP learning algorithm. Although more advanced machine learning algorithms can be implemented by setting $z = f(w, x, y)$, practically the computations of complex algorithms involving the iterative computation, memory, writing, and reading processes of the synstor conductance matrix, w , using external transistor-based computing circuits with low speeds and energy efficiencies^[18–22,23] should be avoided in the learning process for synstor circuits.

In the synstor circuit, \hat{w} is not derived by computing learning algorithms offline based on collected data, but spontaneously transformed in a self-programming process via real-time learning. In the learning process, the feedback pulses, z , are set as

$$z = (y - \hat{y}) * \tilde{\theta} \quad (3)$$

$$\text{where } \tilde{\theta}(t) = \begin{cases} -\mu \theta_-(t) & \text{when } -\tau_- < t < 0 \\ \mu \theta_+(t) & \text{when } \tau_+ > t > 0 \\ 0 & \text{when } t = 0 \text{ or } t \geq \tau_+ \text{ or } t \leq -\tau_- \end{cases}, \text{ the}$$

time constants $\tau_+ > 0$ and $\tau_- > 0$, the function $\theta_+(t) > 0$ and $\theta_-(t) > 0$, and $\mu = 1$ or -1 . The average $\tilde{\theta}$ over learning period T , $\langle \tilde{\theta} \rangle = 0$, and the average z over learning period T , $\langle z \rangle = 0$, and $\bar{z} = z - \bar{z} = z$. \hat{y} can be set in different learning algorithms. For example, $\hat{y} = 0$ in Hebbian and STDP learning. In supervised learning, \hat{y} represents the desired value of y ; in unsupervised learning, \hat{y} represents the converged value of y ; and in reinforcement learning, \hat{y} is set as zero. To generate feedback pulses with $z_n = (y_n - \hat{y}_n) * \tilde{\theta}$ and $\mu = 1$, when a y_n pulse, but no \hat{y}_n pulse, is triggered at the moment $t = t_n$, a train of positive feedback pulses with a pulse firing rate proportional to $\theta_-(t - t_n)$ within the time window $t_n - \tau_- < t < t_n$, and a train of negative feedback pulses with a pulse firing rate proportional to $\theta_+(t - t_n)$ within the time window $t_n < t < t_n + \tau_+$ are triggered from the n th output neuron circuit on the output electrode. When a \hat{y}_n pulse, but no y_n pulse, is triggered at the moment $t = t'_n$, a train of negative feedback pulses with a pulse firing rate proportional to $\theta_-(t - t'_n)$ within the time window $t'_n - \tau_- < t < t'_n$ and a train of positive

feedback pulses with a pulse firing rate proportional to $\theta_+(t - t'_n)$ within the time window $t'_n < t < t'_n + \tau_+$ are triggered from the n th output neuron circuit on the output electrode. The z pulses with the voltage polarities described above with $\mu = 1$ will lead to $\alpha < 0$ in the learning process; z pulses with their voltage polarities opposite to those described above with $\mu = -1$ will lead to $\alpha > 0$ in the learning process.

2.3.1. Self-Programming Process and Objective Functions

In an open-loop synstor circuit as shown in Figure 2a, current signals from other circuits or sensors, I^s , are processed by input neuron circuits to trigger input voltage pulses, x , which induce currents, I , flowing through the synstor circuit. I flows into output neuron circuits to generate output voltage pulses, y , and feedback voltage pulses $z = (y - \hat{y}) * \tilde{\theta}$ (Equation (3)), where \hat{y} represents the desired value of y in supervised learning, and the converged value of y in unsupervised learning. w is modified by x and z by following Equation (2), $\dot{w} = \alpha z \otimes x$ in the self-programming process. The goal of the self-programming process is to minimize an objective function (Equation S6, Supporting Information)

$$F = \frac{1}{2} (y - \hat{y})^2 \quad (4)$$

by modifying w toward $\hat{w} = \argmin_{\hat{w}} F$. By substituting z in Equation (2) by Equation (3), $\dot{w} = \alpha [\tilde{\theta} * (y - \hat{y})] \otimes x$ (Equation S2, Supporting Information). When $y = \hat{y}$, $F = 0$, $\dot{w} = 0$, w reaches an equilibrium value \hat{w} . By substituting $y - \hat{y}$ in \dot{w} as a function of $w - \hat{w}$ (Equation S4, Supporting Information), the dynamic change of w in the learning process can be expressed as a function of $w - \hat{w}$ (Equation S5, Supporting Information)

$$\langle \dot{w} \rangle = -\beta \circ (\langle w \rangle - \langle \hat{w} \rangle) + \delta w \quad (5)$$

where $\langle \dot{w} \rangle$ denotes average \dot{w} , $\langle w \rangle$ denotes average w , and $\langle \hat{w} \rangle$ denotes average \hat{w} over learning period T . $\beta \circ (\langle w \rangle - \langle \hat{w} \rangle)$ denotes the Hadamard product between β and $\langle w \rangle - \langle \hat{w} \rangle$, and β denotes a $(\beta_{nm})_{N \times M}$ matrix with $\beta_{nm} = -\alpha g_n^{y/I} \langle \tilde{\theta} * \kappa \rangle \langle x_m, x_m \rangle \geq 0$, $\alpha > 0$, $g_n^{y/I} = \left(\frac{dy_n}{dI_n} \right)_{y_n=\hat{y}_n} \geq 0$, $\langle \tilde{\theta} * \kappa \rangle = -\int_0^{\tau_+} \kappa(t) \theta_-(t) dt < 0$, and the variance of x_m , $\langle x_m, x_m \rangle \geq 0$. $\delta w = O[(w - \hat{w})^2]$ represents the higher order terms, $(w - \hat{w})^k$, with $k \geq 2$. When w approaches \hat{w} , $O[(w - \hat{w})^2]$ can be omitted. In the self-programming process, the change of w leads to the change of output signals y and objective function F defined in Equation (4). Based on Equation (5), the average change rate of F can be expressed as (Equation S7, Supporting Information)

$$\langle \dot{F} \rangle = -\beta \langle F \rangle + \delta F \quad (6)$$

where $\beta = \sum_{m,n} 2 \langle \beta_{nm} \rangle / MN = -2 \sum_{m,n} \alpha g_n^{y/I} \langle \tilde{\theta} * \kappa \rangle \langle x_m, x_m \rangle / MN \geq 0$, and where $\delta F = -g^{F/w} \circ \langle (w - \hat{w}) \circ \dot{w} \rangle + O[(w - \hat{w})^3]$ with $g^{F/w} = \langle [g^{y/I} \circ (\kappa * x)]^2 \rangle$, and $O[(w - \hat{w})^3]$ represents the higher order terms, $(w - \hat{w})^k$, with $k \geq 3$.

In a closed-loop synstor circuit as shown in Figure 2b, the synstor circuit is connected with an external system or another circuit via sensors and actuators. Output signals y from the synstor

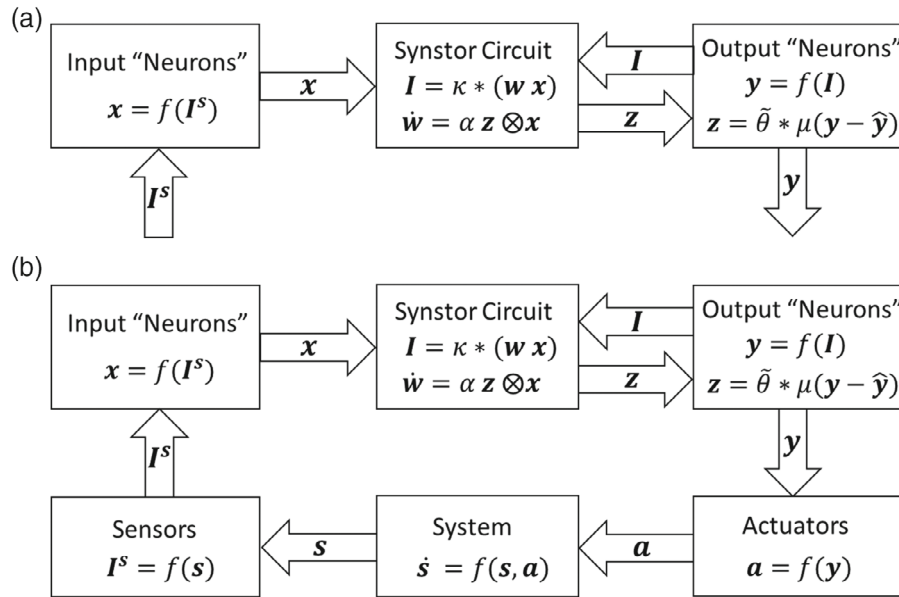


Figure 2. a) For signal processing in a synstor circuit, currents from previous layer of circuits or sensors, I^s , are transmitted to input neuron circuits to trigger input voltage pulses, x . x induces currents, I , flowing from synstors into the output neuron circuits by following Equation (1), $I = \kappa * (w x)$, where w denotes a matrix of synstor conductances. The currents I flow into output neuron circuits, which generate forward-propagating output voltage pulses, y , and back-propagating feedback voltage pulses, z , by following Equation (3), $z = (y - \hat{y}) * \tilde{\theta}$. For learning, w is modified by x and z by following Equation (2), $\dot{w} = \alpha z \otimes x$. b) A synstor circuit is integrated with a system or another circuit. Output signals, y , are transmitted to actuators to trigger actuation signals, a , which modify the states of the system, s . s is detected by sensors to generate currents, I^s , flowing into the input neuron circuits. The synstor circuit is operated in the same way as described before.

circuit are transmitted to actuators to trigger actuation signals a , which modify the states of the system s . s is detected by sensors to generate currents I^s , which are transmitted to the input neuron circuits to trigger input signals x . x induces currents I flowing through the synstor circuit and collected by output neuron circuits to generate output voltage pulses y , and feedback voltage pulses $z = y * \tilde{\theta}$ (Equation (3) with $\hat{y} = 0$). Concurrently w is modified by x and z by following Equation (2), $\dot{w} = \alpha z \otimes x$. The goal of the self-programming process is to modify s toward the desired state of the system, \hat{s} , and an objective function can be defined as

$$F = \frac{1}{2}(s - \hat{s})^2 \quad (7)$$

When $s = \hat{s}$, $F = 0$, $x = 0$, and $\dot{w} = \alpha z \otimes x = 0$ (Equation (2)), w reaches an equilibrium value \hat{w} . By substituting z in Equation (2) by Equation (3), $\dot{w} = \alpha(\tilde{\theta} * y) \otimes x$. By substituting y in \dot{w} as a function of $w - \hat{w}$ (Equation S4, Supporting Information), the dynamic change of w in the learning process can also be expressed by Equation (5) (Equation S11, Supporting Information), $\langle \dot{w} \rangle = -\beta * (\langle w \rangle - \langle \hat{w} \rangle) + \delta w$. In the self-programming processes, the change of w leads to the change of output signals y , which modifies s and the objective function F defined in Equation (7). The dynamic change of F in the self-programming process can also be described by Equation (6), $\langle \dot{F} \rangle = -\beta \langle F \rangle + \delta F$ (Equation S12, Supporting Information), with $\beta = \sum_{m,n} 2\langle \beta_{nm} \rangle / MN = -2 \sum_{m,n} \alpha g_n^{y/I} \langle \tilde{\theta} * \kappa \rangle \langle x_m, x_m \rangle / MN \geq 0$, $\delta F = -g^{F/w} * \langle (w - \hat{w}) * \dot{w} \rangle + O[(w - \hat{w})^3]$ and $g^{F/w} = \langle \{g^{x/y} * [g^{y/I} * (\kappa * x)]\}^2 \rangle$.

In a self-programming process of a closed-loop or open-loop synstor circuit, when δF satisfies $\delta F < \beta \langle F \rangle$, $\langle F \rangle = -\beta \langle F \rangle + \delta F < 0$, $\langle F(w) \rangle$ represents a Lyapunov function, and is asymptotically decreased toward its dynamic equilibrium value F_e , leading $\langle w \rangle$ to be modified toward $\langle \hat{w} \rangle$ in the self-programming process; when $\delta F = \beta \langle F \rangle$, $\langle \dot{F} \rangle = 0$, $\langle F \rangle$ reaches its dynamic equilibrium value $F_e = \delta F / \beta$ under $\langle w \rangle = \langle \hat{w} \rangle = \text{argmin}_{\langle w \rangle} \langle F \rangle$ (Theorem 1 and 2, Supporting Information).

2.3.2. Simulation of Self-Programming Dynamic Process

A closed-loop synstor circuit connected with an external system is simulated by a MATLAB software and is shown in Figure 3a as an example of the self-programming process. The goal of the self-programming process is to modify state s with an arbitrary unit in a system (Figure 3b) toward the desired state, $\hat{s} = 0$. s is detected by sensors to generate a sensory current $I^s = \begin{pmatrix} s \\ 0 \end{pmatrix}$

when $s \geq 0$, and $I^s = \begin{pmatrix} 0 \\ |s| \end{pmatrix}$ when $s < 0$. I^s triggers input pulses x with amplitudes of $\pm 1V$ and a duration of $2.5 \mu s$ (Figure 3c) from input integrate-and-fire neuron circuits (Section S4 and Figure S2, Supporting Information) with a capacitance $C_{IF} = 50 pF$, a leakage current $I_L = 1 nA$, and a threshold voltage $V_{th}^{IF} = 0.3V$. x induces currents, I (Figure 1), by following Equation (1), $I = w x$, which flows into output integrate-and-fire neuron circuits to generate feedback pulses, z (Figure 3d), with amplitudes of $\pm 1V$ and a duration of $2.5 \mu s$, and output pulses, y (Figure 3e), with an amplitude of $1V$ and a duration of $2.5 \mu s$. The output integrate-and-fire neuron circuits (Section S4 and

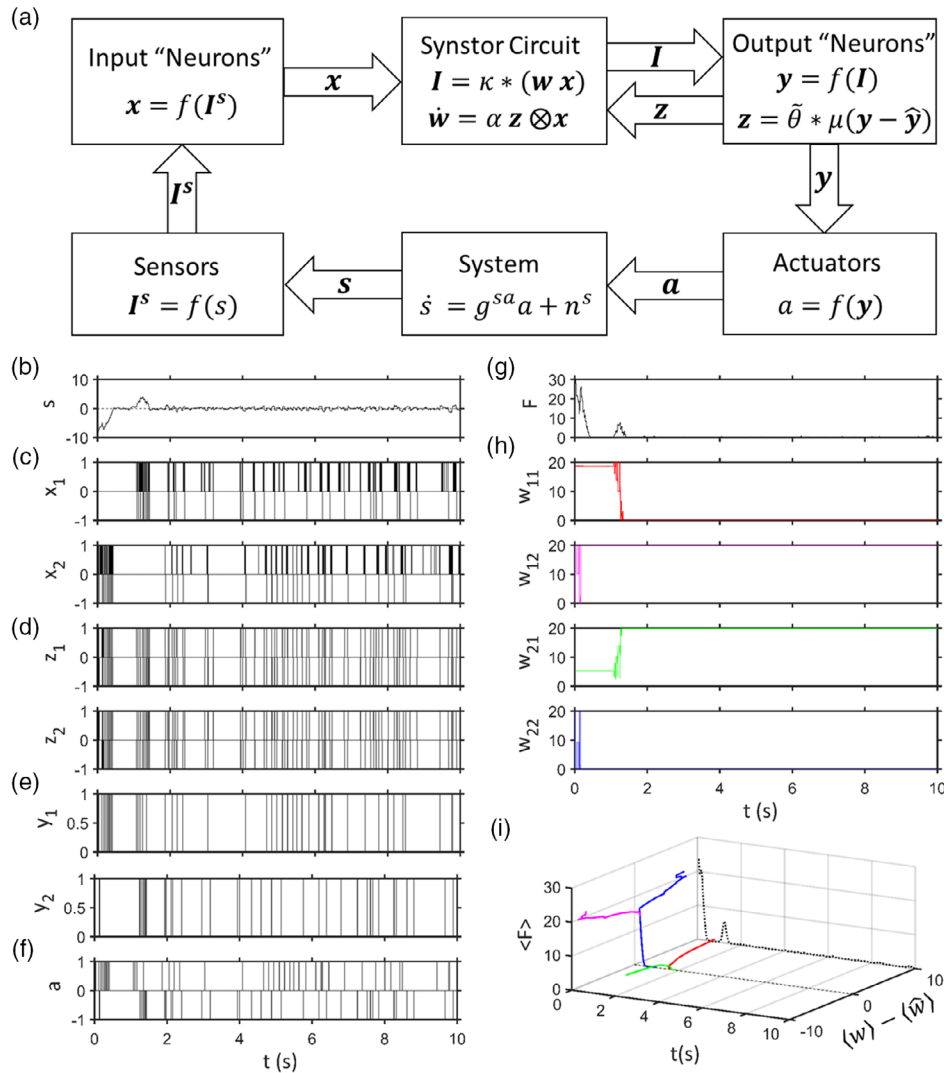


Figure 3. a) A simulated closed-loop system including a synstor circuit and an external system. In the simulated synstor circuit and system, b) s , the state of the system (arbitrary unit), c) x_1 and x_2 , the input pulses on the input electrodes 1 and 2 (unit: V), d) z_1 and z_2 , the feedback pulses on the output electrodes 1 and 2 (unit: V), e) y_1 and y_2 , the output pulses from the output neuron circuits 1 and 2 (unit: V), f) a , the actuation pulses to the system (unit: V), g) F , the objective function of the system (arbitrary unit), h) w_{11} , w_{12} , w_{21} , and w_{22} , the conductances of the synstors in the circuit (unit: nS), are shown versus time t . i) The average objective function, $\langle F \rangle$ (arbitrary unit), and the difference between the average synstor conductances and their optimal conductances, $\langle w_{11} \rangle - \langle \hat{w}_{11} \rangle$ (red solid line), $\langle w_{12} \rangle - \langle \hat{w}_{12} \rangle$ (magenta solid line), $\langle w_{21} \rangle - \langle \hat{w}_{21} \rangle$ (green solid line), and $\langle w_{22} \rangle - \langle \hat{w}_{22} \rangle$ (blue solid line), in the unit of nS is shown as a function of t . The black dotted lines show $\langle F \rangle$ changing versus time t in the background.

Figure S2, Supporting Information) have a capacitance $C_{IF} = 4$ nF, a leakage current $I_L = 0$, and a threshold voltage $V_{th}^{IF} = 0.3$ V. Following Equation (3), $z = \tilde{\theta} * y$ with $\tilde{\theta}(t) = \begin{cases} \delta(t + \tau_+) \\ -\delta(t - \tau_-) \end{cases}$, $\tau_+ = 0$, and $\tau_- = 2.5$ μ s. When a y pulse is triggered, a 1V z pulse is triggered simultaneously, and a -1 V z pulse is triggered at 2.5 μ s after the y pulse is triggered. y pulses trigger actuation pulses a (Figure 3f) with an amplitude of 1 V and a duration of 2.5 μ s by following $a = y_1 - y_2$, and the system state s is modified by a with $\dot{s} = g^{sa}a + n^s$, where g^{sa} represents a modification coefficient, and n^s represents the random perturbation from environment. When $|s| \geq 12$, it is assumed that the system reaches its boundary, is out of control, and the

self-programming process fails. When the system is modified, the objective function of the system, $F = \frac{1}{2} s^2$ (Figure 3g), is also modified accordingly. The synstor conductances are set to random values with $0 \leq w_{nm} \leq 20$ nS before the self-programming process, and the synstor conductances w_{nm} (Figure 3h) are modified by x and z following Equation (2), $\dot{w} = \alpha z \otimes x$ within the range of $0 \leq w_{nm} \leq 20$ nS in the self-programming process. In the simulation shown in Figure 3, $\alpha = 3$ nS $^{-2}$ s $^{-1}$, $g^{sa} = 1$ au, and $-0.25 \leq n^s \leq 0.25$ au.

The dynamic self-programming process shown in Figure 3 is analyzed, and the change of the average objective function $\langle F \rangle$ is shown versus time t and the average synstor conductances $\langle w_{nm} \rangle$ over a moving time window T in Figure 3i. The change of w_{nm}

leads to the change of F , which is also influenced by the random environment noise n^s . When the change of w_{nm} leads to the change of F in a learning period, the covariance between F and w_{nm} $\langle \tilde{F}, \tilde{w}_{nm} \rangle \neq 0$; when w_{nm} does not change beyond the learning period, $\langle \tilde{F}, \tilde{w}_{nm} \rangle = 0$. Learning periods within which $\langle \tilde{F}, \tilde{w}_{nm} \rangle \neq 0$ and the minimal length of moving time window T that satisfies $\langle \tilde{F} \rangle \leq 0$ or $\langle \tilde{F} \rangle \geq 0$ within each learning period are identified in the analysis. Within a learning period, if $\langle \tilde{F} \rangle < 0$, $\langle \hat{w}_{nm} \rangle$ is identified as $\langle w_{nm} \rangle$ at the end of a learning period when $\langle \tilde{F}, \tilde{w}_{nm} \rangle = 0$, $\langle \tilde{F} \rangle = 0$, and $\langle F \rangle$ approaches its minimal equilibrium value F_e (F_e may not be equal to zero due to the environment noise or other synaptic conductances $w_{n'm'}$). After $\langle \hat{w}_{nm} \rangle$ is identified, $\langle w_{nm} \rangle - \langle \hat{w}_{nm} \rangle$ and $\langle F \rangle$ within the learning period are shown versus time t in Figure 3i. Within each learning period, $\langle F \rangle$ decreases asymptotically versus time t with $\langle \tilde{F} \rangle < 0$, and $\langle w_{nm} \rangle$ is gradually modified toward $\langle \hat{w}_{nm} \rangle$; at the end of the learning period, $\langle w_{nm} \rangle = \langle \hat{w}_{nm} \rangle$, $\langle \tilde{F} \rangle = 0$, and $\langle F \rangle$ reaches an equilibrium value F_e . $\langle F \rangle$ represents a Lyapunov function within each learning period. The system changes dynamically by the random environment noise. When the system is perturbed, and F is increased from F_e above a threshold value, \hat{w} is also shifted accordingly, and a learning period will spontaneously be triggered in the self-programming process to modify w toward \hat{w} and F toward F_e in the dynamically changing environment.

The self-programming processes in the synstor circuit are simulated and analyzed statistically by setting the initial synstor conductance matrix w to random values within the range of $0 \leq w_{nm} \leq 20$ nS, and the system state s within the range of $4 \leq |s| \leq 8$ au in multiple simulations. s is also influenced by random environment noise, with $0 \leq |n^s| \leq 0.25$ au. w is modified by following the learning rule, and $\langle \hat{w} \rangle$ is extrapolated in self-programming processes. The objective function F is modified as the function of w and the random environment noise, and $\langle F \rangle$ is shown versus $\langle w \rangle - \langle \hat{w} \rangle$ in multiple self-programming processes in Figure 4a. After w is set to its initial value $w(0)$, w is modified toward \hat{w} to reduce $\langle F \rangle$ asymptotically, and $\langle F \rangle$ reaches its equilibrium value F_e when $\langle w \rangle = \langle \hat{w} \rangle$. When w_{nm} is set to an initial value that triggers actuations to decrease F toward its equilibrium value F_e , the initial w_{nm} is close to \hat{w}_{nm} , $w_{nm}(0) - \hat{w}_{nm} \approx 0$, and $\langle F \rangle$ is shown versus $w_{nm}(t) - \hat{w}_{nm}$ as lines with large slopes in Figure 4a. When w_{nm} is set to an

initial value that is far away from \hat{w}_{nm} , $|w_{nm}(0) - \hat{w}_{nm}|$ has a large value, and $\langle F \rangle$ is shown versus $w_{nm}(t) - \hat{w}_{nm}$ as lines with small slopes in Figure 4a. $\langle F \rangle$ is best fitted as a parabolic function of $|\langle w \rangle - \langle \hat{w} \rangle|$, and is shown versus $|\langle w \rangle - \langle \hat{w} \rangle|$ in Figure 4b. As shown in Figure 4b, the synstor conductance matrix w does not need to be preprogrammed, and can be spontaneously modified toward \hat{w} , decreasing $\langle F \rangle$ toward F_e in all of the 100 simulated self-programming processes.

2.3.3. The Stability and Equilibrium States of Self-Programming Processes

The self-programming processes in the synstor circuit are simulated and analyzed statistically by setting the initial synstor conductance matrix w and the system state s to random values, as described earlier, the conductance modification coefficient $\alpha = 0.2, 1$, and 10 nSV $^{-2}$ s $^{-1}$, respectively, and the leakage current $I_L = 0.1$ nA in the output neuron circuits. The average objective $\langle F \rangle$ is shown versus $\langle w \rangle - \langle \hat{w} \rangle$ in multiple self-programming processes in Figure 5a under $\alpha = 0.2$, Figure 5b under $\alpha = 1$, and Figure 5c under $\alpha = 10$. $\langle F \rangle$ is best fitted as a parabolic function of $|\langle w \rangle - \langle \hat{w} \rangle|$, and is also shown versus $|\langle w \rangle - \langle \hat{w} \rangle|$ in Figure 5a–c. When $\alpha = 0.2$ nSV $^{-2}$ s $^{-1}$ (Figure 5a), due to the random perturbation from the environment, $\langle \tilde{F} \rangle > 0$ in some of the self-programming processes. Under these conditions, F is not stable, and gradually increases to its upper limit value (~ 72). When $\alpha = 1$ nSV $^{-2}$ s $^{-1}$ (Figure 5b) and $\alpha = 10$ nSV $^{-2}$ s $^{-1}$ (Figure 5c), $\langle \tilde{F} \rangle < 0$ in all the self-programming processes. Under these conditions, F is stable, and w is gradually modified toward \hat{w} to decrease $\langle F \rangle$ to its equilibrium values F_e .

In a self-programming process, the synstor conductance matrix w is modified by following Equation (5), $\langle \dot{w} \rangle = -\beta \cdot (\langle w \rangle - \langle \hat{w} \rangle) + \delta w$, and its solution gives $\langle w \rangle = \langle \hat{w} \rangle - [\langle w(0) \rangle - \langle \hat{w} \rangle] e^{-\int_0^t \beta(\tau) d\tau} + (\delta w - \langle \hat{w} \rangle) * e^{-\int_0^t \beta(\tau) d\tau}$, where $\langle \beta \rangle$ represents the speed to modify $\langle w \rangle$ toward $\langle \hat{w} \rangle$ in a self-programming process. $\beta_{nm} = -\alpha g_n^{y/1} \langle \tilde{\theta} * \kappa \rangle \langle x_m, x_m \rangle$ increases linearly with respect to the conductance modification coefficient α defined in Equation (2), and α increases exponentially with the magnitudes of x and z pulses.^[27] In the simulation of self-programming processes, β_{nm} is extrapolated from \dot{w}_{nm} and $w_{nm} - \hat{w}_{nm}$, $\beta_{nm} = \langle \dot{w}_{nm}, w_{nm} - \hat{w}_{nm} \rangle / (w_{nm} - \hat{w}_{nm}, w_{nm} - \hat{w}_{nm})$.

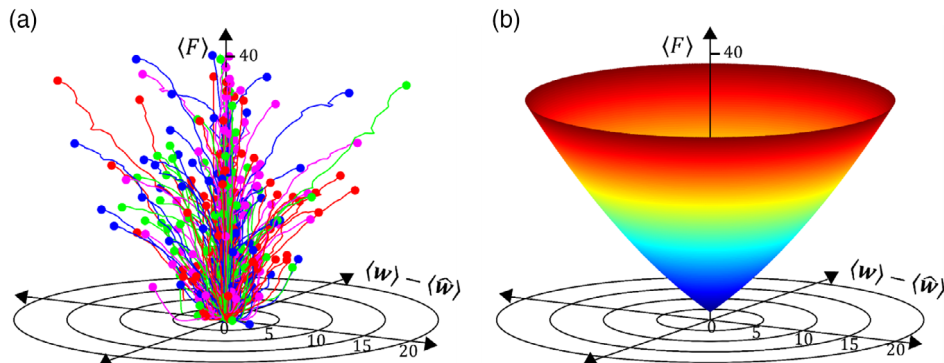


Figure 4. a) The average objective function, $\langle F \rangle$ (arbitrary unit), in multiple self-programming processes, is plotted versus the differences between the average synstor conductances and their optimal conductances, $\langle w \rangle - \langle \hat{w} \rangle$, with the unit of nS and its display azimuth proportional to the simulation number. $\langle w_{11} \rangle - \langle \hat{w}_{11} \rangle$, $\langle w_{12} \rangle - \langle \hat{w}_{12} \rangle$, $\langle w_{21} \rangle - \langle \hat{w}_{21} \rangle$, and $\langle w_{22} \rangle - \langle \hat{w}_{22} \rangle$ are shown by red, magenta, green, and blue lines, respectively, and the initial differences are marked by circles. b) $\langle F \rangle$ shown in (a) is best fitted and plotted versus $\langle w \rangle - \langle \hat{w} \rangle$.

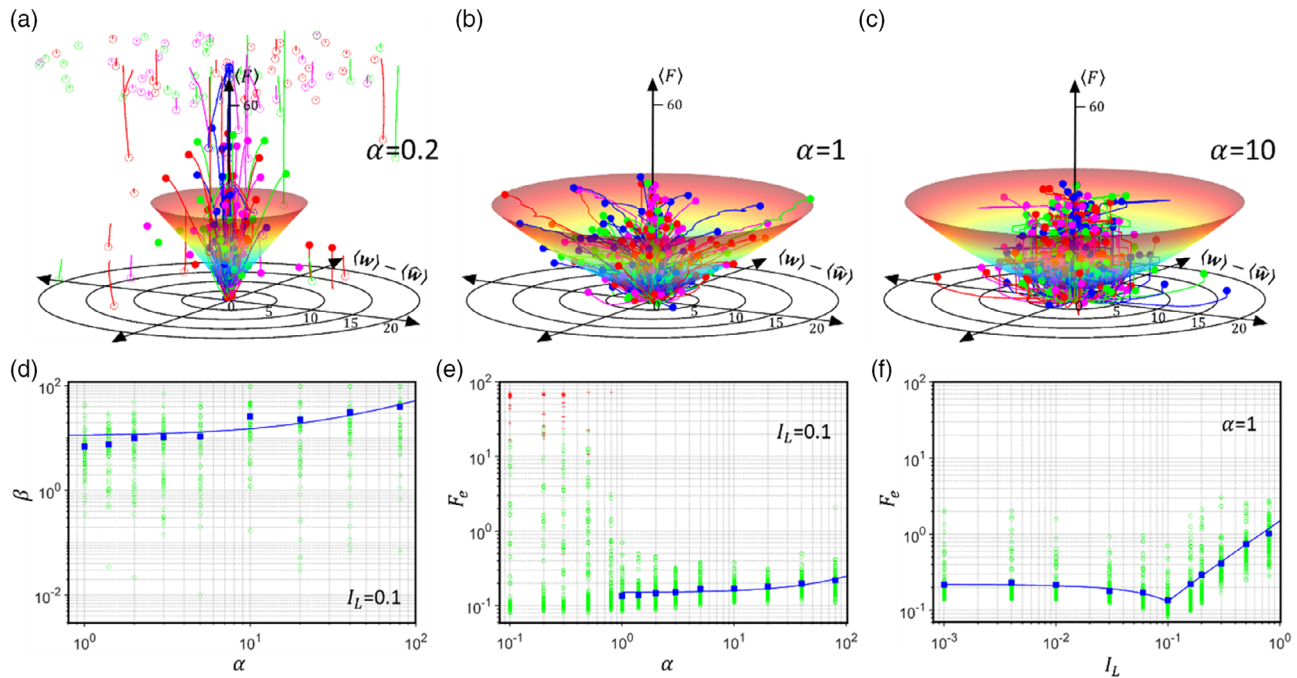


Figure 5. The average objective function, $\langle F \rangle$, in arbitrary unit (lines), and the best fit to $\langle F \rangle$ (3D surfaces) in multiple self-programming processes, is plotted versus the differences between the average synstor conductances and their optimal conductances, $\langle \mathbf{w} \rangle - \langle \hat{\mathbf{w}} \rangle$, with the unit of nS and its display azimuth proportional to the simulation number under a) $\alpha = 0.2$ nSV⁻²s⁻¹, $I_L = 0.1$ nA; b) $\alpha = 1$ nSV⁻²s⁻¹, $I_L = 0.1$ nA; and c) $\alpha = 10$ nSV⁻²s⁻¹, $I_L = 0.1$ nA. $\langle w_{11} \rangle - \langle \hat{w}_{11} \rangle$, $\langle w_{12} \rangle - \langle \hat{w}_{12} \rangle$, $\langle w_{21} \rangle - \langle \hat{w}_{21} \rangle$, and $\langle w_{22} \rangle - \langle \hat{w}_{22} \rangle$ are shown by red, magenta, green, and blue lines, respectively, and the initial differences are marked by circles. d) β representing the speed of self-programming processes is shown by green open cycles in the unit of s⁻¹ versus the conductance modification coefficient α in the unit of nSV⁻²s⁻¹ in multiple self-programming processes. The average β is shown by filled blue squares, and best fitted by $\langle \beta \rangle = g_{\beta\alpha}\alpha + \beta_0$ with $g_{\beta\alpha} \approx 0.41$ nS⁻¹V² and $\beta_0 \approx 11$ s⁻¹. The equilibrium objective function, F_e (arbitrary unit), is plotted versus e) the conductance modification coefficient α in the unit of nSV⁻²s⁻¹ under $I_L = 0.1$ nA, and f) the leakage current in the integrate-and-fire neuron circuit I_L in the unit of nA under $\alpha = 1$ nSV⁻²s⁻¹. The green open circles mark F_e , the blue filled squares mark average F_e , and the red crosses mark the initial F under unstable condition. The average F_e is fitted by $\langle F_e \rangle = g_{Fa}\alpha + F_e^0$ with $g_{Fa} \approx 10^{-3}$ (au) and $F_e^0 \approx 0.15$ (au) for 1 nSV⁻²s⁻¹ $< \alpha < 10^2$ V⁻²s⁻¹ in (e), and fitted by $\langle F_e \rangle = g_{FI}(I_L - I_L^0)$ with $g_{FI} \approx -0.89$ (au) and $I_L^0 \approx 0.67$ nA for 1 pA $\leq I_L \leq 0.1$ nA and $g_{FI} \approx 1.5$ (au) and $I_L^0 \approx 0.01$ nA for 0.1 nA $\leq I_L \leq 1$ nA in (f).

As shown in Figure 5d, β_{nm} increases linearly with respect to α , and the average $\langle \beta \rangle$ can be best fitted by $\langle \beta \rangle = g_{\beta\alpha}\alpha + \beta_0$ with $g_{\beta\alpha} \approx 0.41$ nS⁻¹V² and $\beta_0 \approx 11$ s⁻¹ for 1 nSV⁻²s⁻¹ $\leq \alpha \leq 100$ nSV⁻²s⁻¹ under the stable conditions. The self-programming speed can be increased by increasing α , but it is limited by the dynamic response speed of the simulated system ($\approx 10^2$ s⁻¹). In a synstor circuit and system, $\langle F \rangle = -\beta\langle F \rangle + \delta F < 0$ when $\beta\langle F \rangle > \delta F$, which is the condition for the stability of the circuit and system. δF is mainly induced by the random perturbation from the environment, and β increases versus α ; therefore, $\beta\langle F \rangle$ is increased versus α , and satisfies the stability condition $\beta\langle F \rangle > \delta F$ when α is increased above a critical value ($\alpha \gtrsim 1$ nSV⁻²s⁻¹ as shown in Figure 5b–d). When α is decreased below the critical value ($\alpha \lesssim 1$ nSV⁻²s⁻¹ as shown in Figure 5a,d), $\beta\langle F \rangle < \delta F$, and $\langle F \rangle > 0$, F is unstable.

When $\beta\langle F \rangle > \delta F$ and $\langle F \rangle < 0$, $\langle F \rangle$ is asymptotically decreased to an equilibrium value F_e in a self-programming process. To understand the influence of the conductance modification coefficient α on the equilibrium state and stability, the self-programming processes in a synstor circuit are simulated when α changes between 0.1 and 100 nSV⁻²s⁻¹ and the rest of the conditions remain the same as described previously. F_e is plotted versus α in Figure 5e. When α decreases from 1 nSV⁻²s⁻¹ to

0.1 V⁻²s⁻¹, β decreases with decreasing α , leading to $\langle F \rangle > 0$ and unstable F . When α increases from 1 nSV⁻²s⁻¹ to 10² V⁻²s⁻¹, β increases with increasing α , and $\langle F \rangle = -\beta F_e + \delta F$ also decreases, leading to $\langle F \rangle < 0$ and equilibrium F_e . As shown in Figure 5e, the average F_e can be best fitted by $\langle F_e \rangle = g_{Fa}\alpha + F_e^0$ with $g_{Fa} \approx 10^{-3}$ (au) and $F_e^0 \approx 0.15$ (au) for 1 nSV⁻²s⁻¹ $< \alpha < 10^2$ V⁻²s⁻¹. F_e increases with increasing α under the stable condition $\langle F \rangle \leq 0$. β increases linearly with increasing α (Figure 5d), $\delta F = -\mathbf{g}^F/\mathbf{w} \cdot \langle (\mathbf{w} - \hat{\mathbf{w}}) \cdot \hat{\mathbf{w}} \rangle + O[(\mathbf{w} - \hat{\mathbf{w}})^3]$ is mainly induced by the environmental perturbation, and also includes the terms related to $\mathbf{w} - \hat{\mathbf{w}}$ and α^k with $k \geq 1$; therefore, $F_e = \delta F/\beta \approx F_e^0 + g_{Fa}\alpha + O(\alpha^2)$ near the equilibrium condition, as shown by the fitting line in Figure 5e.

To understand the influence of the leakage currents, I_L , in the output neurons on F_e , the self-programming processes in a synstor circuit are simulated when I_L changes between 1 pA and 1 nA and the rest of the conditions remain the same as described previously. F_e is plotted versus I_L in Figure 5f. When I_L increases from 0.1 to 1 nA, F_e increases with increasing I_L . As shown in Figure 5f, the average F_e can be best fitted by $\langle F_e \rangle = g_{FI}(I_L - I_L^0)$ with $g_{FI} \approx 1.5$ (au) and $I_L^0 \approx 0.01$ nA for 0.1 nA $\leq I_L \leq 1$ nA. When I_L decreases from 0.1 nA to 1 pA, F_e increases with decreasing I_L . As shown in Figure 5f, the

average F_e can be best fitted by $\langle F_e \rangle = -g_{FI}^2(I_L - I_L^2)$ with $g_{FI}^2 \approx 0.89(\text{au})$ and $I_L^2 \approx 0.67 \text{ nA}$ for $1 \text{ pA} \leq I_L \leq 0.1 \text{ nA}$. When I_L increases from 0.1 to 1 nA, the firing rate of output pulses y from the output neuron circuits decreases with increasing I_L , leading to the decrease of the firing rate of the actuation pulses to control the system state and the increase of F_e . When I_L decreases from 0.1 nA to 1 pA, the firing rate of output pulses y from the output neuron circuits increases with decreasing I_L , leading to the increase of the firing rate of the actuation pulses, the fluctuation of the system state, and the increase of F_e . F_e is a nonlinear function of I_L with an optimal value to minimize F_e .

F_e is also influenced by the critical parameters in self-programming processes, such as the conductance modification coefficient, α , the firing rates and amplitudes of the feedback pulses, \mathbf{z} , the upper limit of synstor conductances, transfer functions between the system and synstor circuit, g^{sa} , the gain between the firing rates of output pulses versus input currents, $g_n^{y/I}$, the capacitance of the capacitors, C_{IF} , the leakage currents, I_L , the threshold voltages, V_{th}^{IF} , and the saturated firing rates of output pulses in neuron circuits. The critical parameters that influence the equilibrium state and stability condition can be modified and optimized via learning in self-programming processes. In a simulated self-programming process, α and I_L do not have fixed values as described previously, but are modified by following the correlative learning rules similar to Equation (2), $\Delta\alpha = k_\alpha \langle \alpha, \langle F \rangle \rangle$ and $\Delta I_L = k_I \langle I_L, \langle F \rangle \rangle$, where $k_\alpha = 1 (\text{au})$ and $k_I = 10 (\text{au})$ denotes the modification coefficients for α and I_L respectively, $\langle F \rangle$ denotes the average objective function during a learning period $T = 1 \text{ s}$. $\langle \alpha, \langle F \rangle \rangle$ denotes the covariance between α and $\langle F \rangle$, and $\langle I_L, \langle F \rangle \rangle$ denotes the covariance between I_L and $\langle F \rangle$. During the self-programming process, when \mathbf{w} is continuously modified by following Equation (2), α and I_L remain at constant values within each learning period T , and are discretely modified by following the learning rules given before after the end of each learning period. In the self-programming process,

the objective function, F , is shown versus time t in Figure 6a. As shown in Figure 6b, α is set to an initial value $\alpha_0 = 0.15 \text{ nSV}^{-2} \text{ s}^{-1}$ at the beginning of the self-programming process, and is gradually modified to its equilibrium value $\alpha_e \approx 0.98 \text{ nSV}^{-2} \text{ s}^{-1}$, which is approximately equal to the optimal value of $\alpha \approx 1 \text{ nSV}^{-2} \text{ s}^{-1}$ as shown in Figure 5e. As shown in Figure 6c, I_L is set to an initial value $I_L^0 = 0.9 \text{ nA}$, and is gradually modified to its equilibrium value $I_L^e \approx 0.14 \text{ nA}$, which is equal to the optimal value of $I_L \approx 0.1 \text{ nA}$ as shown in Figure 5f, and F is reduced from its initial value $F(0) = 68 \text{ au}$ to its equilibrium value $F_e \approx 0.4 \text{ au}$ at end of the self-programming process. The simulation indicates that critical parameters that influence the objective function F can also be dynamically modified to their optimal equilibrium values in a self-programming process to minimize F .

2.4. Comparisons between Self-Programming Synstor Circuits and Programmable Computing Circuits

In comparison with computers, the equivalent computing operations in an $M \times N$ synstor circuit are approximately equal to $3MN$ to implement the signal processing algorithm ($\mathbf{I} = \kappa * (\mathbf{w}\mathbf{x})$, Equation (1), $2MN$ for multiplications between \mathbf{w} , κ , and \mathbf{x} ; MN for accumulations), and $3MN$ to implement the learning algorithm ($\dot{\mathbf{w}} = \alpha \mathbf{z} \otimes \mathbf{x}$, Equation (2), $2MN$ outer products between α , \mathbf{x} , and \mathbf{z} ; MN for \mathbf{w} modifications). The speed for the synstor circuit to implement $6MN$ equivalent computing operations for parallel signal processing and learning

$$V_c = 6MNf_c \quad (8)$$

where f_c denotes the operation frequency of the circuit.

When voltage pulses are applied on its input or output electrode of an $M \times N$ synstor circuit connected with M input and N output integrate-and-fire neuron circuits, the average power consumption in an $M \times N$ synstor circuit,

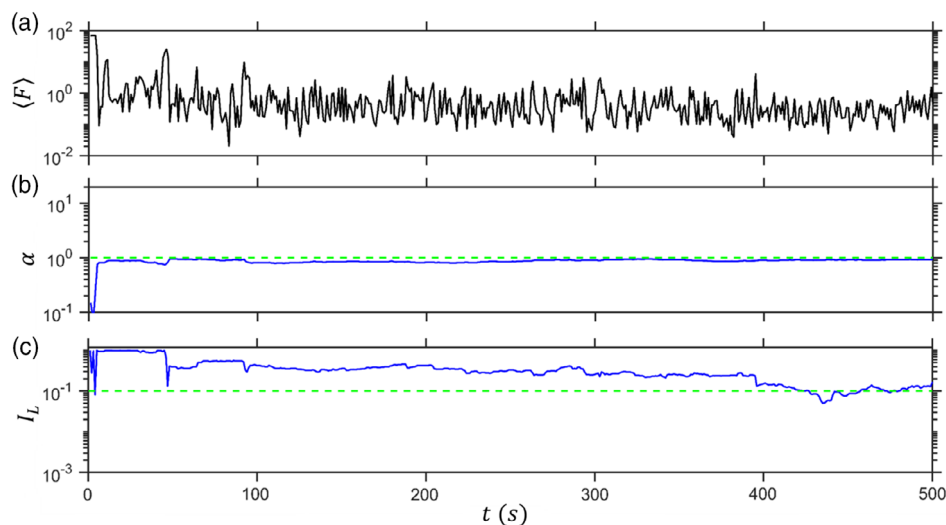


Figure 6. a) The average objective function, $\langle F \rangle$, in arbitrary unit (black line), b) the conductance modification coefficient α in the unit of $\text{nSV}^{-2} \text{ s}^{-1}$ (blue solid line), and c) the leakage current in the integrate-and-fire neuron circuit I_L in the unit of nA (blue solid line) are plotted versus time t in a self-programming process. The green dashed lines in (b) show the optimal value of $\alpha = 1 \text{ nSV}^{-2} \text{ s}^{-1}$ and the optimal value of $I_L = 0.1 \text{ nA}$.

$$P_c \approx MN\langle w \rangle V_a^2 D_p + ME_p r_{in} + NE_p r_{out} \quad (9)$$

where $\langle w \rangle$ denotes the average conductance of the synstors, V_a denotes the magnitude of pulses, D_p denotes the average duty-cycle of the pulses, E_p denotes the average energy consumption to trigger a pulse from integrate-and-fire neuron circuits, and r_{in} and r_{out} denote the average firing rates of pulses from input and output neuron circuits, respectively. The computing energy efficiency of a synstor circuit is equal to its computing speed (V_c) divided by its power consumption (P_c)

$$E_c = 6f_c / (\langle w \rangle V_a^2 D_p + E_p r_{in} / N + E_p r_{out} / M) \quad (10)$$

The computing energy efficiencies of $10^3 \times 10^3$ and $10^4 \times 10^4$ synstor circuits operated with operation frequencies f_c ranging from 100 Hz to 1 GHz, average synstor conductances $\langle w \rangle$ ranging from 1 pS to 100 nS, $V_a = 1$ V, $D_p = 0.01$, $E_p = 1$ pJ, $r_{in} = r_{out} = 0.01 f_c$, and energy efficiencies $E_c \gtrsim 5 \times 10^{16}$ OPS W^{-1} are shown versus their computing speeds and power consumptions in Figure 7a. The computing energy efficiencies, speeds, and power consumptions of a 4×2 synstor circuit,^[27] the human brain,^[15] the Summit supercomputer,^[9] Volta V100 GPUs from Nvidia,^[10] TPUs from Google,^[5] a Stratix 10 field-programmable gate array (FPGA) from Intel,^[11] a TrueNorth neuromorphic circuit from IBM,^[12] memristor circuits from UMass/HP,^[21] Tianjic neuromorphic

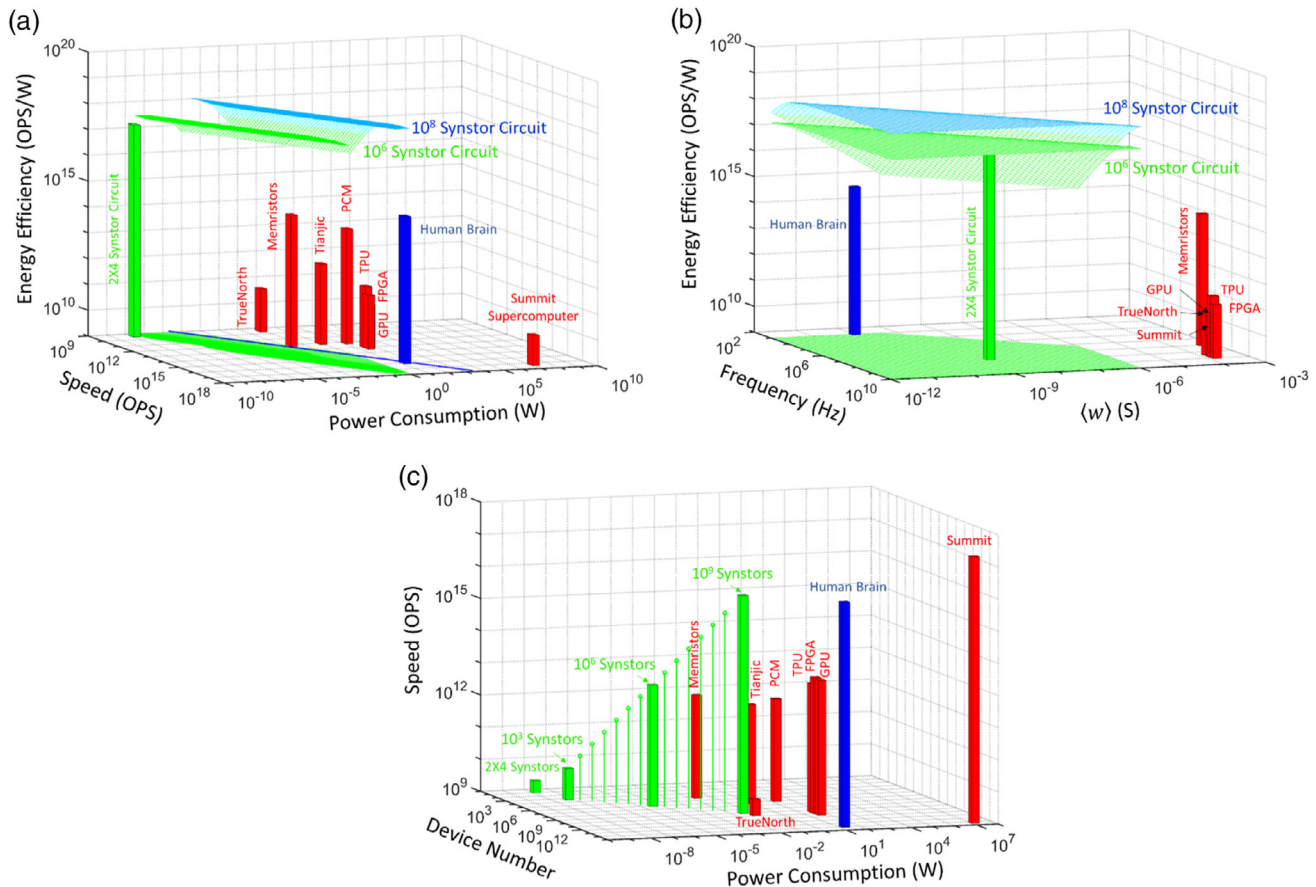


Figure 7. a) The computing energy efficiencies of $10^3 \times 10^3$ (green) and $10^4 \times 10^4$ (blue) synstor circuits operated with operation frequencies ranging from 100 Hz to 1 GHz, average synstor conductances ranging from 1 pS to 100 nS, and energy efficiencies $E_c \gtrsim 5 \times 10^{16}$ OPS W^{-1} are shown versus circuit computing speeds and power consumptions. b) The computing energy efficiencies are shown versus circuit operation frequency and average synstor conductance. c) The computing speeds of synstor circuits are shown versus the number of synstors and power consumption in integrated circuits with the number of synstors ranging from 10^2 to 10^9 . The energy efficiencies, computing speeds, power consumption, operation frequencies, average device conductances, and number of devices in a 2×4 synstor circuit (green), the human brain (blue), the Summit supercomputer, Volta V100 GPUs from Nvidia, TPUs from Google, a Stratix 10 FPGA from Intel, a TrueNorth neuromorphic circuit from IBM, memristor circuits from UMass/HP (signal processing only, learning excluded), Tianjic neuromorphic circuits from Tsinghua University, and a phase change memory circuit from IBM (signal processing only, learning excluded) are also shown in (a), (b), and (c). The energy efficiency of the brain is benchmarked by a blue line at 5×10^{14} OPS W^{-1} in the speed–power plane in (a). The circuits with energy efficiencies less than the energy efficiency of the brain (5×10^{14} OPS W^{-1}) are marked in red. The ranges of the speeds and power consumption of synstor circuits that operate with energy efficiencies higher than the energy efficiency of the brain are marked by the green areas in the speed–power plane in (a) and the frequency–conductance plane in (b). The ranges of the frequency and the average device conductance of synstor circuits that operate with energy efficiencies higher than the energy efficiency of the brain are marked by the green area in the frequency–conductance plane in (b).

circuits from Tsinghua University,^[13] and a phase change memory circuit from IBM^[24] are also shown in Figure 7a for comparison. The speed of the Summit supercomputer exceeds the speed of the human brain, but consumes much more power ($\approx 10^7$ W) than the human brain (≈ 30 W), and has a much larger volume ($\approx 10^4$ m³) than the human brain ($\approx 10^{-3}$ m³). It is not practical to send a large amount of data to the supercomputer to calculate a learning algorithm remotely in real time; thus, the supercomputer needs to collect “big data” and then calculate the learning algorithm off-site with significant delay with respect to the dynamic changes of the systems and environments. To calculate learning algorithms on-site in real time, a computing circuit needs to be embedded in intelligent systems, consume less power than the human brain ($\lesssim 30$ W), have a computing speed exceeding the speed of the human brain ($\gtrsim 10^{16}$ OPS), and have a computing energy efficiency exceeding the efficiency of the human brain ($\gtrsim 5 \times 10^{14}$ OPS W⁻¹). However, the energy efficiencies of the existing electronic computing circuits have not exceeded the efficiency of the human brain (Figure 7a). A synstor circuit can execute spatiotemporal signal processing (Equation (1)) and correlative learning (Equation (2)) algorithms concurrently with high energy efficiency by circumventing the fundamental computing limitations in existing electronic circuits such as physically separated logic and memory units, data transmission between memory and logic, the execution of the inference and learning algorithms in serial mode in different circuits, and the signal transmissions between the circuits. The equivalent computing energy efficiency of the 4×2 synstor circuit is 1.6×10^{17} OPS W⁻¹,^[27] which exceeds the efficiency of the human brain ($\approx 5 \times 10^{14}$ OPS W⁻¹). As shown in Figure 7a, when synstor circuits are scaled up to 10^8 synstors, the computing energy efficiencies of the synstor circuits are increased up to 5×10^{18} OPS W⁻¹ due to the decrease of average power consumptions to trigger pulses from neuron circuits ($E_p r_{in}/N$ and $E_p r_{out}/M$ in Equation (10)). Based on Equation (10), the computing energy efficiency of synstor circuits also increases with increasing circuit frequency f_c and decreasing average device conductance $\langle w \rangle$. As shown in Figure 7b, the average conductances of synstors ($10^{-3} - 10^3$ nS) and synapses ($10^{-3} - 10^{-1}$ nS) are much lower than the average conductances of transistors and memristors ($10^4 - 10^6$ nS) in existing computing circuits, resulting in the computing energy efficiencies of synstor circuits and the human brain surpassing the energy efficiencies of the existing computing circuits. Computers are operated in serial mode, and the total latency of the serial computations is equal to the sum of the latency of individual computations. The latency is proportional to the individual transistor resistance and needs to be reduced to extremely small values by decreasing the transistor resistance. Synstor circuits and human brains are operated in parallel mode, and the latency of the parallel analog computation is proportional to the total resistance of synstors and synapses connected in parallel with each neuron. The latency can be reduced by increasing the numbers of synstors and synapses connected in parallel (M in a synstor circuit), while keeping the high resistance of synstors and synapses to reduce the power consumption. Based on Equation (10), the computing energy efficiency of synstor circuits can also be increased by increasing the circuit frequency f_c from 100 Hz to 1 GHz (Figure 7b). Due to the limit of ion transportation speed in the human brain, it

operates at the frequency of ≈ 1 kHz. The operation of synstor circuits is based on electron transportation; therefore, the operation frequency and energy efficiency of synstor circuits can be significantly higher than those of the human brain. However, the average power consumption to trigger pulses from neuron circuits ($E_p r_{in}/N$ and $E_p r_{out}/M$ in Equation (10)) also increases with increasing operation frequency, leading to the saturation of the energy efficiency of synstor circuits at high operation frequency (Figure 7b).

As shown in Figure 7c, the computing speed and power consumption of an $M \times N$ synstor (synapse) circuit increases with increasing scale of the circuit (i.e., MN , the numbers of parallel input/output electrodes; Equation (8) and (9)). With the energy efficiency of the 4×2 circuit demonstrated experimentally (1.6×10^{17} OPS W⁻¹)^[27] and an operation frequency $f_c = 1$ MHz, a $10^3 \times 10^3$ synstor circuit has a speed of 6×10^{12} OPS (Figure 7c), which is comparable with the speeds of TPU, GPU, and FPGA transistor-based circuits ($\approx 10^{12} - 10^{14}$ OPS); a power consumption of $\approx 40 \mu$ W, much less than the power consumptions of the transistor-based circuits (≈ 40 W); and the number of synstors $\approx 10^6$, much less than the number of transistors in the transistor-based circuits ($\approx 10^9 - 10^{11}$). A $10^5 \times 10^4$ synstor circuit has a speed of 6×10^{15} OPS, comparable with the speeds of the human brain ($\approx 10^{16}$ OPS) and the Summit supercomputer ($\approx 10^{17}$ OPS); a power consumption of ≈ 40 mW, much less than the power consumptions of the human brain (≈ 30 W) and Summit supercomputer ($\approx 10^7$ W); and the number of synstors $\approx 10^9$, much less than the number of devices in the human brain ($\approx 10^{14}$) and Summit supercomputer ($\approx 10^{14}$). Synstor circuits with high speed, low power consumption, high energy efficiency, and small circuit scale/volume could be embedded in intelligent systems and powered by batteries to compute signal processing and learning algorithms in the self-programming process.

3. Conclusion

We have modeled and analyzed a synstor circuit that emulates a neurobiological network to execute signal processing and learning algorithms concurrently in parallel mode. The synstor conductance matrix \mathbf{w} in the circuit does not need to be preprogrammed, and can be spontaneously modified toward the optimal matrix $\hat{\mathbf{w}}$, minimizing the average objective function $\langle F \rangle$ in a self-programming process to optimize and create new algorithms in complex and erratic environments. The stability, learning speed, and equilibrium state of the self-programming process are influenced by critical parameters such as the conductance modification coefficient, α , and neuron circuit leakage currents, I_L , which can also be optimized in the self-programming process. The computing speed and power consumption of the synstor circuit can be improved by scaling up the circuit, increasing its operational frequency, and reducing synstor conductances. A circuit of 10^6 synstors will have a speed (6×10^{12} OPS) comparable with the speeds ($\approx 10^{12} - 10^{14}$ OPS) of TPU, GPU, and FPGA circuits with $\approx 10^9 - 10^{11}$ transistors, and consume much less power ($\approx 40 \mu$ W) than the transistor-based circuits (≈ 40 W). A circuit of 10^9 synstors will have a speed (6×10^{15} OPS) comparable with the speeds of the human brain ($\approx 10^{16}$ OPS) and the Summit supercomputer ($\approx 10^{17}$ OPS), and consumes much less power

(~40 mW) than the human brain (≈ 30 W with $\approx 10^{14}$ synapses) and the Summit supercomputer ($\approx 10^7$ W with $\approx 10^{14}$ transistors). There is “plenty of room at the bottom” to scale up synstors circuits with high speed, low power consumption, high energy efficiency, and small circuit scale/volume for a new computing platform that can self-program in real time in arbitrary and erratic environments, embedded in intelligent systems, and powered by batteries.

Supporting Information

Supporting Information is available from the Wiley Online Library or from the author.

Acknowledgements

The author acknowledges the support of this work by the Air Force Office of Scientific Research (AFOSR) under a grant number AFOSR-FA9550-19-0213, titled “Brain Inspired Networks for Multifunctional Intelligent Systems in Aerial Vehicles” and under a grant number AFOSR-FA9550-20-1-0230, titled “Multi-layer Self-Programming Neuromorphic Integrated Circuits for Deep Learning.”

Conflict of Interest

The author declares no conflict of interest.

Keywords

intelligent systems, neuromorphic computation, self-programming, synaptic resistors

Received: October 1, 2020
Revised: November 8, 2020
Published online:

- [1] A. M. Turing, *Mind* **1950**, 49, 28.
- [2] M. M. Waldrop, *Nature* **2016**, 530, 144.
- [3] C. D. James, J. B. Aimone, N. E. Miner, C. M. Vineyard, F. H. Rothganger, K. D. Carlson, S. A. Mulder, T. J. Draelos, A. Faust, M. J. Marinella, *Biol. Inspired Cognit. Archit.* **2017**, 19, 49.
- [4] J. Schmidhuber, *Neural Netw.* **2015**, 61, 85.
- [5] D. Silver, J. Schrittwieser, K. Simonyan, I. Antonoglou, A. Huang, A. Guez, T. Hubert, L. Baker, M. Lai, A. Bolton, Y. T. Chen, T. Lillicrap, F. Hui, L. Sifre, G. van den Driessche, T. Graepel, D. Hassabis, *Nature* **2017**, 550, 354.
- [6] X. Xu, Y. Ding, S. X. Hu, M. Niemier, J. Cong, Y. Hu, Y. Shi, *Nat. Electron.* **2018**, 1, 216.
- [7] A. J. G. Hey, R. P. Feynman, *Feynman and Computation: Exploring the Limits of Computers*, Westview Press/Perseus Books, Cambridge, MA **2002**.
- [8] R. Bellman, Rand Corporation, *Dynamic Programming*, Princeton University Press, Princeton **1957**.
- [9] Oak Ridge National Laboratory, America's newest and smartest supercomputer, <https://www.olcf.ornl.gov/summit/> (accessed: June 2018).
- [10] D. Wysocki, R. O'Shaughnessy, J. Lange, Y.-L. L. Fang, *Phys. Rev. D* **2019**, 99, 084026.
- [11] E. Nurvitadhi, G. Venkatesh, J. Sim, D. Marr, R. Huang, J. Ong Gee Hock, Y. T. Liew, K. Srivatsan, D. Moss, S. Subhaschandra, *presented at Proc. of the 2017 ACM/SIGDA Int. Symp. on Field-Programmable Gate Arrays*, Monterey, CA, February 2017.
- [12] P. A. Merolla, J. V. Arthur, R. Alvarez-Icaza, A. S. Cassidy, J. Sawada, F. Akopyan, B. L. Jackson, N. Imam, C. Guo, Y. Nakamura, B. Brezzo, I. Vo, S. K. Esser, R. Appuswamy, B. Taba, A. Amir, M. D. Flickner, W. P. Risk, R. Manohar, D. S. Modha, *Science* **2014**, 345, 668.
- [13] J. Pei, L. Deng, S. Song, M. Zhao, Y. Zhang, S. Wu, G. Wang, Z. Zou, Z. Wu, W. He, *Nature* **2019**, 572, 106.
- [14] a) R. S. Williams, *Comput. Sci. Eng.* **2017**, 19, 7; b) V. Zhironov, R. Cavin, L. Gammaitoni, in *ICT-Energy-Concepts Towards Zero-Power Information and Communication Technology*, IntechOpen, London, UK **2014**.
- [15] R. Kurzweil, *The Singularity Is Near: When Humans Transcend Biology*, Penguin, New York, NY **2005**.
- [16] W. Gerstner, W. M. Kistler, *Spiking Neuron Models: Single Neurons, Populations, Plasticity*, Cambridge University Press, Cambridge **2002**.
- [17] D. O. Hebb, *The Organization of Behavior: A Neuropsychological Theory*, Wiley, New York, **1949**.
- [18] C. Diorio, P. Hasler, A. Minch, C. A. Mead, *IEEE Trans. Electron. Dev.* **1996**, 43, 1972.
- [19] Q. Lai, L. Zhang, Z. Li, W. F. Stickle, R. S. Williams, Y. Chen, *Adv. Mater.* **2010**, 22, 2448.
- [20] Y. Kaneko, Y. Nishitani, M. Ueda, *IEEE Trans. Electron. Dev.* **2014**, 61, 2827.
- [21] Z. Wang, S. Joshi, S. Savel'ev, W. Song, R. Midya, Y. Li, M. Rao, P. Yan, S. Asapu, Y. Zhuo, *Nat. Electron.* **2018**, 1, 137.
- [22] a) Q. Xia, J. J. Yang, *Nat. Mater.* **2019**, 18, 309; b) M. Hu, C. E. Graves, C. Li, Y. Li, N. Ge, E. Montgomery, N. Davila, H. Jiang, R. S. Williams, J. J. Yang, *Adv. Mater.* **2018**, 30, 1705914; c) X. Yan, J. Zhao, S. Liu, Z. Zhou, Q. Liu, J. Chen, X. Y. Liu, *Adv. Funct. Mater.* **2018**, 28, 1705320; d) X. Yan, Y. Pei, H. Chen, J. Zhao, Z. Zhou, H. Wang, L. Zhang, J. Wang, X. Li, C. Qin, *Adv. Mater.* **2019**, 31, 1805284.
- [23] S. B. Eryilmaz, D. Kuzum, R. Jayasingh, S. Kim, M. BrightSky, C. Lam, H.-S. P. Wong, *Front. Neurosci.* **2014**, 8, 205.
- [24] B. L. Jackson, B. Rajendran, G. S. Corrado, M. Breitwisch, G. W. Burr, R. Cheek, K. Gopalakrishnan, S. Raoux, C. T. Rettner, A. Padilla, *ACM J. Emerg. Technol. Comput. Syst.* **2013**, 9, 1.
- [25] J. Yin, S. Gahlot, N. Laanait, K. Maheshwari, J. Morrison, S. Dash, M. Shankar, *presented at 2019 IEEE/ACM Third Workshop on Deep Learning on Supercomputers (DLS)*, Denver, CO, November 2019.
- [26] D. Silver, A. Huang, C. J. Maddison, A. Guez, L. Sifre, G. Van Den Driessche, J. Schrittwieser, I. Antonoglou, V. Panneershelvam, M. Lanctot, S. Dieleman, D. Grewe, J. Nham, N. Kalchbrenner, I. Sutskever, T. Lillicrap, M. Leach, K. Kavukcuoglu, T. Graepel, D. Hassabis, *Nature* **2016**, 529, 484.
- [27] C. D. Danesh, C. M. Shaffer, D. Nathan, R. Shenoy, A. Tudor, M. Tadayon, Y. Lin, Y. Chen, *Adv. Mater.* **2019**, 31, 1808032.
- [28] Z. Chen, S. Haykin, J. J. Eggermont, S. Becker, *Correlative Learning: A Basis for Brain and Adaptive Systems*, John Wiley & Sons, New York **2007**.