A Natural Lottery Ticket Winner: Reinforcement Learning with Ordinary Neural Circuits

Ramin Hasani [1 2 *] **Mathias Lechner** [3 *] **Alexander Amini** [2] **Daniela Rus** [2] **Radu Grosu** [1]

Abstract

We propose a neural information processing system obtained by *re-purposing* the function of a biological neural circuit model to govern simulated and real-world control tasks. Inspired by the structure of the nervous system of the soil-worm, *C. elegans*, we introduce *ordinary neural circuits* (ONCs), defined as the model of biological neural circuits reparameterized for the control of alternative tasks. We first demonstrate that ONCs realize networks with higher maximum flow compared to arbitrary wired networks. We then learn instances of ONCs to control a series of robotic tasks, including the autonomous parking of a real-world rover robot. For reconfiguration of the *purpose* of the neural circuit, we adopt a search-based optimization algorithm. Ordinary neural circuits perform on par and, in some cases, significantly surpass the performance of contemporary deep learning models. ONC networks are compact, 77% sparser than their counterpart neural controllers, and their neural dynamics are fully interpretable at the cell-level.

1. Introduction

We wish to explore a new class of machine learning algorithms for robot control inspired by nature. Through natural evolution, the subnetworks within the nervous system of the nematode, *C. elegans*, structured a near-optimal wiring diagram from the *wiring economy principle*[1] perspective (White et al., 1986; Pérez-Escudero & de Polavieja, 2007). Its stereotypic brain composed of 302 neurons connected through approximately 8000 chemical and electrical synapses (Chen et al., 2006). The wiring diagram therefore, establishes a 91% sparsity and gives rise to high-degrees of controllability, to process complex chemical stimulations (Bargmann, 2006), express adaptive behavior (Ardiel & Rankin, 2010), and to control muscles (Wen et al., 2012).

This property is particularly attractive to the machine learning community that aims at reducing the size of fully-connected neural networks to sparser representations while maintaining the great output performance (LeCun et al., 1990; Hassibi & Stork, 1993; Han et al., 2015; Hinton et al., 2015; Frankle & Carbin, 2018). In this regard, the lottery ticket hypothesis (Frankle & Carbin, 2018), suggested an algorithm to find sparse subnetworks (winning tickets) within a dense, randomly initialized feedforward neural network, which can achieve comparable (and sometimes greater) performance to the original network, when trained separately (Frankle & Carbin, 2018; Zhou et al., 2019; Morcos et al., 2019). The lottery ticket hypothesis motivated us to investigate whether subnetworks (neural circuits) within the natural nervous systems are already formulation of *winning tickets* originated from the natural evolution?

To study this question fundamentally, we take a computational approach to analyze neural circuit models from the worm's nervous system. The reason is that the function of many circuits within its nervous system have been identified (Wicks & Rankin, 1995; Chalfie et al., 1985; Li et al., 2012; Nichols et al., 2017; Kaplan et al., 2019), and simulated (Islam et al., 2016; Hasani et al., 2017; Sarma et al., 2018; Gleeson et al., 2018), which makes it a suitable model organism for further computational investigations.

The general network architecture in *C. elegans* establishes a hierarchical topology from sensory neurons (source nodes) through upper interneuron and command interneurons down to motor neurons, sink nodes, (See Fig. 1A). Typically, in these neuronal circuits, interneurons establish highly recurrent wiring diagrams with each other while sensors and command neurons mostly realize feedforward connections to their downstream neurons.

*Equal contribution [1]Technische Universität Wien (TU Wien) [2]Massachusetts Institute of Technology (MIT) [3]Institute of Science and Technology (IST) Austria. Correspondence to: Ramin Hasani <rhasani@mit.edu>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020.

[1]Under proper functionality of a network, the wiring economy principle proposes that its morphology is organized such that the cost of wiring its elements is minimal.

An example of such a structure is a neural circuit shown in Fig. 1B, the Tap-withdrawal (TW) (Rankin et al., 1990), which is responsible for inducing a forward/backward locomotion reflex when the worm is mechanically exposed to touch stimulus on its body. The circuit has been characterized in terms of its neuronal dynamics (Chalfie et al., 1985). It comprises eleven neuron classes which are wired by thirty chemical and electrical synapses. Is TW a *Winning Ticket* compared to networks of the same size, from any perspective?

1.1. TW graph realizes the highest maximum flow rate

Let us first define the *maximum flow problem* (Shiloach & Vishkin, 1982):

Definition 1. *For a given graph $G(V,E)$, with $s,t \in V$ source and sink nodes, respectively:*

- *The **capacity (weight)** of an edge is the mapping $c: E \rightarrow \mathbb{R}^+$, declared by c_e,*
- *A **Flow** is a mapping $f: E \rightarrow \mathbb{R}^+$, denoted by f_e, from node u to v, if: 1) $f_e \leq c_e$ for each $e \in E$. 2) $\sum_{Inputs\ to\ v} f_e = \sum_{output\ from\ v} f_e$ for all $v \in V$ except source and sink nodes,*
- *The **flow rate** is denoted by $|f| = \sum_{s \rightarrow v} f_{sv}$, where s is the source of G. This value depicts the amount of flow passing from a source node to a chosen sink node.*
- *The **maximum flow problem** is to maximize $|f|$.*

The maximum flow problem is typically used for sparse directed networks to assess their input/output propagation performance. The TW circuit is a sparse directed network, and therefore, we chose to evaluate its propagation properties by computing the maximum flow rate.

TW realizes higher flow-rate from arbitrary chosen source to sink node, compared to randomly-wired networks of the same size. Formally, consider a directed weighted graph $G(V,E)$, with V vertices, $E \ll V^2$ edges and, $S \subset V$, $S = \{s_1,...,s_k\}$ source (sensory neurons), $T \subset V$, $T = \{t_1,...,t_n\}$ sink (motor neurons), $I \subset V$, $I = \{i_1,...,i_{N_i}\}$ interneurons, $C \subset V$, $I = \{c_1,...,c_{N_c}\}$ command neurons. Then, the highest max. flow is achievable for randomly-weighted and -wired networks, when the architecture approaches that of randomly-weighted TW.

To show this, we construct 40000 randomly-wired networks and compare their max-flow rate to randomly-weighted TW. We witnessed an enhanced max-flow rate between 1% and 17% when a network is constrained to be wired similar to TW. (See details in Section 2). Accordingly, this finding motivated us to explore the TW circuit's dynamics from a control theory perspective.

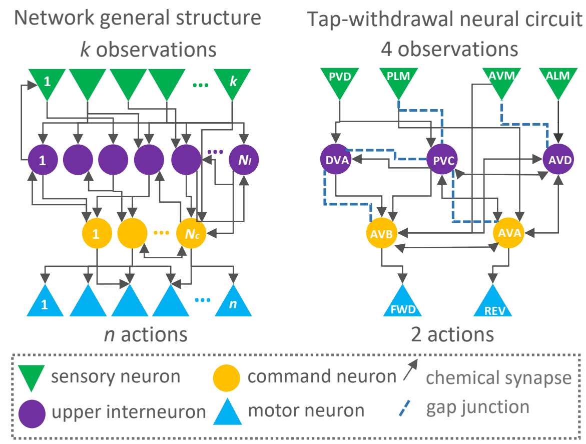

Figure 1. Left: C. elegans' general neuronal circuit structure. Right: Tap-Withdrawal (TW) neural circuit schematic. Total number of interneurons $= N_i + N_C$. We preserve the TW circuit wiring topology and deploy a search-based reinforcement learning algorithm to control robots.

1.2. TW can be trained to govern control tasks

The behavior of the TW reflexive response is substantially similar to the control agent's reaction in standard control settings such as a controller acting on driving an underpowered car, to go up on a steep hill, known as the *Mountain Car* (Singh & Sutton, 1996), or a controller acting on the navigation of a rover robot that plans to go from point A to B.

We model the TW circuit by continuous-time biophysical neuronal and synaptic models that bring about useful attributes; I) In addition to the nonlinearities expressed by the neurons' hidden state, synapses possess additional nonlinearity. This property results in realizing complex dynamics with a fewer number of neurons (Hasani et al., 2018). II) Their dynamics are set by grounded biophysical properties, which ease the interpretation of the network's dynamics.

We construct instances of the TW network obtained by learning its parameters and define these learning systems as *ordinary neural circuits* (ONC). We experimentally investigate ONC's properties in terms of their learning performance, their ability to solve tasks in different RL domains, and introduce ways to interpret their internal dynamics. For this purpose, we preserve the wiring structure of an example ONC (the TW circuit) and adopt a search-based optimization algorithm for learning the neuronal and synaptic parameters of the network.

We discover that sparse ONCs (*Natural lottery winners*) not only establish a higher maximum flow rate from any arbitrary source to sink node but also when trained in isolation for control tasks, significantly outperform randomly wired networks of the same size and in many cases contemporary deep learning models with larger capacities.

1.3. Contributions of the work

- Quantitative illustration of achieving the highest maximum flow rate for randomly wired sparse networks, when their architecture gets closer to ONCs.
- Demonstration of the performance of a compact ONC as an interpretable controller in a series of control tasks and the indication of its superiority compared to similarly structured networks and to contemporary deep learning models.
- Experiments with ONCs in simulated and physical robot control tasks, including the autonomous parking of a reak mobile robot. This is performed by equipping ONCs with a search-based RL optimization scheme.
- Interpretation of the internal dynamics of the learned policies. We introduce a novel computational method to understand continuous-time network dynamics. The technique (Definition 2) determines the relation between the kinetics of sensory/interneurons and a motor neuron's decision. We compute the magnitude of a neuron's contribution (positive or negative), of these hidden nodes to the output dynamics in determinable phases of activity, during the simulation.

2. Design Ordinary Neural Circuits

In this section, we first briefly describe the structure and dynamics of the tap-withdrawal neural circuit as an instance of ONCs. We then delve into the graph theory properties of the network to motivate the TW circuit choice as the *natural lottery winner* for control. We then introduce the mathematical neuron and synapse models utilized to build up the circuit, as an instance of ordinary neural circuits.

2.1. Tap-withdrawal neural circuit

A mechanically exposed stimulus (i.e., tap) to the petri dish in which the worm inhabits, results in the animal's reflexive response in the form of a forward or backward movement. This response has been named as the *tap-withdrawal reflex*, and the circuit identified to underlay such behavior is known as the *tap-withdrawal* (TW) neural circuit (Rankin et al., 1990). The circuit is shown in Fig. 1B. It is composed of four sensory neurons, PVD and PLM (posterior touch sensors), AVM and ALM (anterior touch sensors), five interneuron classes (AVD, PVC, AVA and AVB, DVA), and two subgroups of motor neurons which are abstracted as forward locomotory neurons, FWD, and backward locomotory neurons, REV. Interneurons recurrently synapse into each other with excitatory and inhibitory synaptic links. TW consists of 28 synapses connecting 11 neurons.

Algorithm 1 Design ONC-like random networks

S =**sensory,** T =**motor,** I =**interneuron,** C =**command,** E =**No. of synapses**
Generate E synapse weights, $W \sim Binomial(E, \rho)$
Step 1
for e in range $[1, 40\%E]$ **do**
  source = Rand($^{S}P_1$), target = Rand($^{I\&C}P_1$)
end for
connect source and target
Step 2
$E_{ic} = 53\%E$ selected from the remainder of the synapses
for e in E_{ic} **do**
  source = Rand($^{I\&C}P_1$), target = Rand($^{I\&C}P_1$)
  connect source and target
end for
Step 3
Connect $C = \{c_1, ..., c_{N_c}\}$, one-to-one to $T = \{t_1, ..., t_n\}$
Return $Random_{TW}$ Graph

2.2. Maximum flow rate in ONCs vs. other networks

The TW neural circuit, is wired with a set of network-design constraints. Formally, given V vertices and E edges:

- It realizes a 77% network sparsity.
- The structure exclusively determines four distinct layers of neurons: $S \subset V$, $S = \{s_1, ..., s_k\}$ source (sensory neurons), $T \subset V$, $T = \{t_1, ..., t_n\}$ sink (motor neurons), $I \subset V$, $I = \{I_1, ..., I_{N_i}\}$ interneurons, and $C \subset V$, $I = \{C_1, ..., C_{N_c}\}$ command neurons.
- Sensory nodes unidirectionally synapse into upper interneurons with 40% of the total number of connections.
- Interneurons and command neurons recurrently synapse into each other (without any self-connections) by 53% of the total number of connections.
- Command neurons exclusively synapse into motors by the rest of the synapses (7%).

We discovered that with the construction of randomly-wired sparse networks while applying the aforementioned TW constraints, we can achieve the highest maximum flow rate for such networks. To demonstrate this quantitatively, we developed Algorithm 1 to design random networks with a series of assumptions gradually increased to satisfy TW constraints. We then compute the ratio of the average maximum flow (computed by a tree-search max-flow algorithm (Boykov & Kolmogorov, 2004)) from sensory nodes to motor neurons of the TW circuit, to the obtained networks and report results in Table 1. The ratio approaches 1, which indicates that networks designed based on the TW constraints would benefit from a better max-flow rate than less-constrained, randomly connected networks.

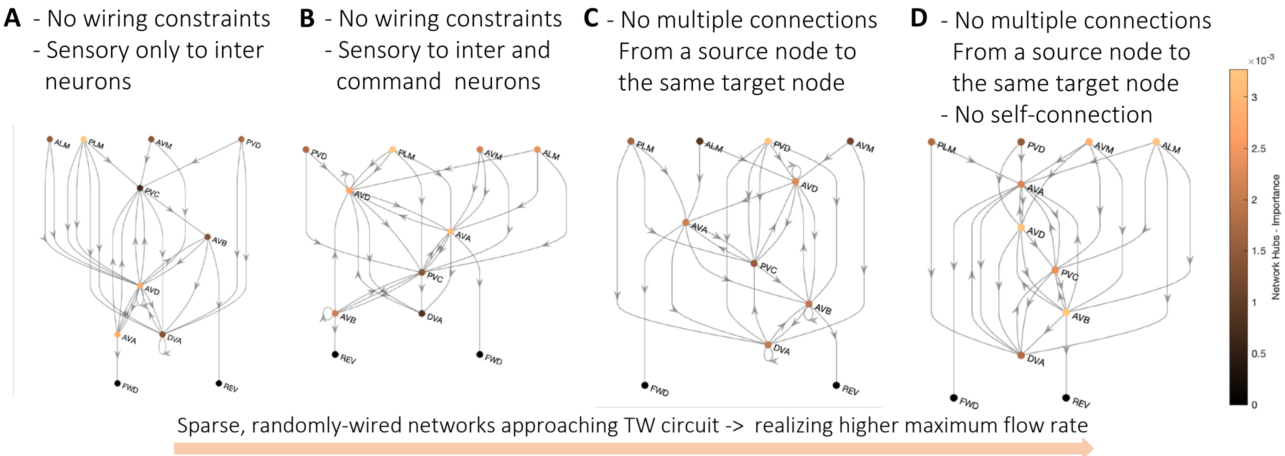

Figure 2. Sparse and randomly wired network samples. A to D indicate random neural circuits with the same number of elements as in TW, wired with modified constraints in Algorithm 1: A) In Step 1, target = Rand($^{I}P_1$) B) In Step 1, target = Rand($^{I\&C}P_1$) C) In Step 1 and Step 2, if the tuple $(source_e, target_e)$ is repeated, remove and loop again. D) In Step 1 and Step 2, if $(source_e, target_e)$ is repeated, remove and loop again. In Step 2, if $source_e = target_e$, remove the selection, and loop again. The colorbar represents network hubs – nodes with highest number of inward/outward edges.

2.3. Neuron and synapse model for ONCs

Here, we briefly describe the neuron and synapse model (Hasani et al., 2018; Lechner et al., 2019), used to design neural circuit dynamics (Hasani et al., 2020):

$$
\begin{aligned}
\dot{V}_i(t) &= \left[I_{i,L} + \textstyle\sum_{j=1}^{n} \hat{I}_{i,j}(t) + \sum_{j=1}^{n} I_{i,j}(t)\right] / C_{i,m} \\
I_{i,L}(t) &= \omega_{i,L}\left[E_{i,L} - V_i(t)\right] \\
\hat{I}_{i,j}(t) &= \hat{\omega}_{i,j}\left[V_j(t) - V_i(t)\right] \\
I_{i,j}(t) &= \omega_{i,j}\left[E_{i,j,R} - V_i(t)\right] g_{i,j}(t) \\
g_{i,j}(t) &= 1 / \left[1 + \exp(-\sigma_{i,j}(V_j(t) - \mu_{i,j}))\right]
\end{aligned} \tag{1}
$$

where $V_i(t)$ and $V_j(t)$ stand for the potential of the post and pre-synaptic neurons, respectively. $E_{i,L}$ and $E_{i,j}$ are the reversal potentials of the leakage and chemical channels. $I_{i,L}$, $\hat{I}_{i,j}$, and $I_{i,j}$ present the currents flowing through the leak channel, electric-synapse, and chemical-synapse, with conductances $\omega_{i,L}$, $\hat{\omega}_{i,j}$, and $\omega_{i,j}$, respectively. $g_{i,j}(t)$ is the dynamic conductance of the chemical-synapse, and $C_{i,m}$ is the membrane capacitance. $E_{i,j}$ determines the whether a synapse is inhibitory or excitatory.

This neural representation belongs to the continuous-time recurrent neural networks class which has recently been shown to give rise to certain computational advantages, such as adaptive computation schemes through numerical solvers of ordinary differential equations (ODEs), parameter efficiency, and strong capabilities on modeling time-series arriving at arbitrary time-steps (Chen et al., 2018; Dupont et al., 2019; Lechner & Hasani, 2020; Lechner et al., 2020).

For interacting with the environment, We introduced sensory and motor neuron models. A *sensory component* consists of two neurons S_p, S_n and an input variable, x. S_p gets activated when x has a positive value, whereas S_n fires when x is negative. The potential of the neurons S_p, and S_n, as a function of x, are defined by an affine function that maps the region $[x_{min}, x_{max}]$ of the system variable x, to a membrane potential range of $[-70mV, -20mV]$. (See the formula in Supplementary Materials Section 2). Similar to sensory neurons, a *motor component* is composed of two neurons M_n, M_p and a controllable motor variable y. Values of y is computed by $y := y_p + y_n$ and an affine mapping links the neuron potentials M_n and M_p, to the range $[y_{min}, y_{max}]$. (See supplements, Section 2). FWD and REV motor classes (Output units) in Fig. 1B, are modeled in this fashion.

For simulating neural networks composed of such dynamical models, we adopted a hybrid numerical solver (Press et al., 2007). Formally, we combined both implicit and explicit Euler's discretization method (Lechner et al., 2019).

Table 1. Ratio of the avg. max-flow of TW to variations of other networks of Fig. 2. The ratio of the max flow of the Random networks of each subcategory to the max flow of TW has been simulated for 10000 times. Total No. of networks tested = 40000.

Networks	Average $\frac{MaxFlow_{TW}}{MaxFlow}$ of FWD neuron	Average $\frac{maxflow_{TW}}{MaxFlow}$ of REV neuron
Fig. 2A	1.15 ± 0.01	1.14 ± 0.01
Fig. 2B	1.12 ± 0.005	1.10 ± 0.01
Fig. 2C	1.03 ± 0.003	1.04 ± 0.005
Fig. 2D	1.01 ± 0.001	1.01 ± 0.001

Algorithm 2 Adaptive Random Search

Input: A stochastic objective indicator f and a starting parameter θ, noise scale σ, adaption rate $\alpha \geq 1$
Output: Optimized parameter θ
$f_\theta \leftarrow f(\theta)$
for $k \leftarrow 1$ **to** maximum iterations **do**
  $\theta' \leftarrow \theta + rand(\sigma)$; $f_{\theta'} \leftarrow f(\theta')$;
  if $f_{\theta'} < f_\theta$ **then** $\theta \leftarrow \theta'$; $f_\theta \leftarrow f_{\theta'}$; $i \leftarrow 0$; $\sigma \leftarrow \sigma \cdot \alpha$ **else** $\sigma \leftarrow \sigma/\alpha$ **end if**
  $i \leftarrow i+1$
  if $i > N$ **then** $f_\theta \leftarrow f(\theta)$ **end if**;
end for
return θ

(See Supplementary Materials Section 3, for a concrete discussion on the model implementation, and the choice of parameters.) Note that the solver has to serve as a real-time control system, additionally.

For reducing the complexity, therefore, our method realizes a fixed-timestep solver. The solver's complexity for each time step Δ_t is $\mathscr{O}(|\#\text{ neurons}| + |\#\text{ synapses}|)$. In the next section, we introduce the optimization algorithm used to reparametrize the tap-withdrawal circuit.

3. Search-based Optimization Algorithm

In this section we formulate a *Reinforcement learning* (RL) setting for tuning the parameters of a given neural circuit to control robots. The behavior of a neural circuit can be expressed as a policy $\pi_\theta(o_i, s_i) \mapsto \langle a_{i+1}, s_{i+1} \rangle$, that maps an observation o_i, and an internal state s_i of the circuit, to an action a_{i+1}, and a new internal state s_{i+1}. This policy acts upon a possible stochastic environment $Env(a_{i+1})$, that provides an observation o_{i+1}, and a reward, r_{i+1}. The stochastic return is given by $R(\theta) := \sum_{t=1}^{T} r_t$. The objective of the RL is to find a θ that maximizes $\mathbb{E}\Big(R(\theta)\Big)$.

Simple search based RL (Spall, 2005), as suggested in (Salimans et al., 2017), in (Duan et al., 2016), and very recently in (Mania et al., 2018), can scale and perform competitively with gradient-based approaches, and in some cases even surpass their performance, with clear advantages such as skipping gradient scaling issues. Accordingly, we adopted a simple search-based algorithm to train the neuronal policies.

Our approach combines an *Adaptive Random Search* (ARS) optimization (Rastrigin, 1963), with an *Objective Estimate* (OE) function $f : \theta \mapsto \mathbb{R}^+$. The OE generates N rollouts with π_θ on the environment and computes an estimate of $\mathbb{E}(R_\theta)$ based on a filtering mechanism on these N samples. We compared two filtering strategies in this context; 1) taking the average of the N samples, and 2) taking the average of the worst k samples out of N samples.

The first strategy is equivalent to the *Sample Mean estimator* (Salimans et al., 2017), whereas the second strategy aims to avoid getting misled by high $\mathbb{E}(R_\theta)$ outliers. The objective was that a suitable parameter θ enforces the policy π_θ control the environment in a reasonable way even in challenging situations (i.e., rollouts with the lowest return). We treat this filtering strategy as a hyperparameter (see Algorithm 2).

4. Experiments

The goal of our experimentation is to answer the following questions: 1) How would an ONC with a preserved biological connectome, perform in basic standard control settings, compared to that of a randomly-wired circuit? Are ONCs *natural lottery ticket winners*? 2) When possible, how would the performance of our learned circuit compare to the other methods? 3) Can we transfer a policy from a simulated environment to a real environment? 4) How can we interpret the behavior of the neural circuit policies?

We use four benchmarks for measuring and calibrating this approach's performance, including one robot application to parking for the TW sensory/motor neurons and then deployed our RL algorithm to learn the parameters of the TW circuit and optimize the control objective. The environments include I) Inverted pendulum of Roboschool (Schulman et al., 2017), II) Mountain car of OpenAI Gym, III) HalfCHeetah from Mujoco, and IV) Parking a real rover robot with a transferred policy from a simulated environment. The code is available online. [2] The TW neural circuit (cf. Fig. 1B) allows us to incorporate four input observations and to take two output control actions. We evaluate our ONC in environments of different toolkits on a variety of dynamics, interactions, and reward settings.

4.1. How to map ONCs to environments?

The TW neural circuit is shown in Fig. 1B, contains four sensory neurons. It, therefore, allows us to map the circuit to four input variables. Let us assume we have an inverted pendulum environment which provides four observation variables The position of the cart x, together with its velocity $\dot{x}$, the angle of the pendulum φ.[3] along with its angular velocity $\dot{\varphi}$. Since the main objective of the controller is to balance the pendulum in an upward position and make the car stay within the horizontal borders, we can feed φ (positive and negative values), and x (positive and negative), as the inputs to the sensors of the TW circuit.

Control commands can be obtained from the motor neuron classes, FWD and REV. Likewise, any other control problem can be feasibly mapped to an ONC. We set up the search-

[2] Code is available online at: `https://github.com/mlech26l/ordinary_neural_circuits`

[3] Remark: The environment further splits φ into $sin(\varphi)$ and $cos(\varphi)$ to avoid the $2\pi \to 0$ discontinuity

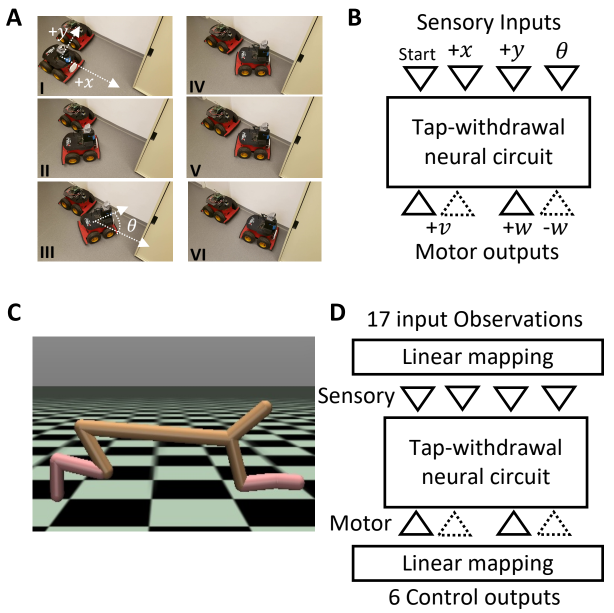

Figure 3. Mapping the environments to the TW circuit in A) Parking task, B) mapping for the parking. C) half-cheetah, and C) mapping for the half-cheetah experiment. See Table S3 in the Supplementary Material for more details.

based RL algorithm to optimize neurons' and synapses' parameters $\omega, \hat{\omega}, \sigma, C_m, E_L$ and G_L, within their corresponding range, shown in Table S2. A video of different stages of the learned ordinary neural circuit for the inverted pendulum can be viewed at `https://youtu.be/cobEtJVw3A4`

In a simulated Mountaincar experiment, the environmental variables are the car's horizontal position, x, together with its linear velocity. The control signal applies force to the car to build momentum until finally reaching the top of the hill. The TW circuit can then be learned by the search-based RL algorithm. A video illustrating the control of the car at various episodes during the optimization process can be viewed at `https://youtu.be/J7vXFsZz7EM`.

4.2. Scale the functionality of ONCs to environments with larger observation spaces

We extend the application of the TW circuit as an instance of ordinary neural circuits, to handle tasks with more observation variables. We choose the HalfCheetah-v2 test-bed of Mujoco. The environment consists of 17 input and six output variables. We add a linear layer that maps an arbitrary number of input variables to two continuous variables fed into the four sensory neurons of the TW circuit, as shown in Fig. 3D. Similarly, we add a linear layer that maps the neuron potentials of the two motor neurons to the control outputs. A video of this experiment can be viewed at `https://youtu.be/zG_L4JGOMbU`.

Table 2. ONC vs. random circuits - n=10, High standard deviations are due to the inclusion of unsuccessful attempts.

Env / Method	Random Circuit	**ONC**
Inverted Pendulum	138.1± 263.2	**866.4 ±418**
Mountain car	54 ±44.6	**91.5±6.6**
Half-Cheetah	1742.9±642.3	**2891.4±1016**

4.3. Transfer learned ONCs to control real robot

In this experiment, we generalized TW to learn a real-world control task. We let TW learn to park a rover robot on a determined spot, given a set of checkpoints on a trajectory, in a deterministic simulated environment. We then deploy the learned policy on a mobile robot in a real environment shown in Fig. 3A. The key objective here is to show the capability of the method to perform well in a transformation from a simulated environment to real. For doing this, we developed a *custom deterministic simulated RL environment*.

The rover robot provides four observational variables (start signal, position (x, y) and angular orientation θ), together with two motor actions (linear and angular velocity, v and w). We mapped all four observatory variables, as illustrated in Fig. 3B, to the sensors of the TW. Note that the geometric reference of the surrounding space is set at the initial position of the robot. Therefore, observation variables are positive.

We mapped the linear velocity (which is a positive variable throughout the parking task) to one motor neuron and the same variable to another motor neuron. We determined two motor neurons for the positive and negative angular velocity. (See Table S3 in Supplementary for mapping details). This configuration implies that the command neuron, AVA, controls two motor neurons responsible for the turn-right and forward motion-primitives, and AVB to control the turn-left and also forward motor neurons.

Optimization setup for the parking task – A set of checkpoints on a pre-defined parking trajectory was determined in the custom simulated environment. For every checkpoint, a deadline was assigned. At each deadline, a reward was given due to the rover's negative distance to the current checkpoint. The checkpoints are placed to resemble a real parking trajectory composed of a sequence of motion primitives: Forward, turn left, forward, turn right, forward, and stop. We then learned the TW circuit by the RL algorithm.

The learned policy has been mounted on a Pioneer AT-3 mobile robot and performed a reasonable parking performance. The video of the TW ordinary neural circuit's performance on the parking task can be viewed at `https://youtu.be/p0GqKf0V0Ew`.

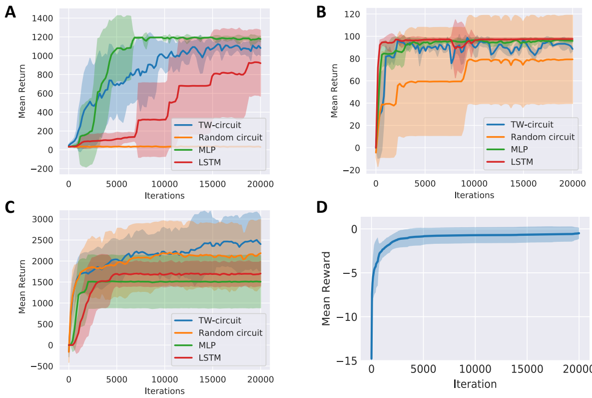

Figure 4. Learning curves. A) Inverted pendulum B) Mountain car (OpenAI Gym) C) Half-Cheetah D) The parking task. The shadows represent standard deviation. n = 10

5. Experimental Evaluation

In this section, we thoroughly assess the results of our experimentation. We qualitatively and quantitatively explain the performance of our ordinary neural circuits. We then benchmark our results with the existing methods, and describe the main attributes of our methodology. Finally, we quantitatively interpret the dynamics of the learned policies.

5.1. Do ONCs perform better than random circuits?

We performed an experiment where we designed circuits with randomly wired connectomes, with the same number of neurons and synapses used in the TW circuit. The synapses' initial polarity is set randomly (excitatory, inhibitory, or electrical synapse) with a simple rule that no synapse can be fed into a sensory neuron, which is a property of ONCs.

The random circuits were then trained over a series of control tasks described earlier, and their performance is reported in Table 2. We observe that ONCs significantly outperform the randomly wired networks, which is empirical evidence for ONCs being the lottery ticket winners.

5.2. Relation to the lottery ticket hypothesis

The Lottery ticket hypothesis (Frankle & Carbin, 2018) states that we can train sparse networks from an obtained *winning ticket* – i.e., weight initialization. Now in terms of ONCs, TW realizes a sub-circuit of 77% sparsity and more importantly, TW synapses are initialized by naturally-determined weight structures.

It is worth noting that biological weights are not simply determined by scalar weight values to be initialized. Instead, they are declared as shown in Eq. 1, by:

- different types of synapses (gap-junctions or chemical synapses see $\hat{I}_{ij}$ and I_{ij} in Eq. 1).
- different polarities i.e. excitatory/inhibitory (set by an independent variable E in I_{ij}).
- a nonlinear weight profile shown by $g_{ij}(t)$, in Eq. 1, and a maximum weight value.

Table 3. Comparison of ONC to artificial neural networks with policy gradient algorithms

Method	Inverted Pendulum	MountainCar
MLP + PPO (Schulman et al., 2017)	1187.4±51.7	94.6±1.3
MLP + A2C (Mnih et al., 2016)	1191.2±45.2	86.4±18.3
ONC + RS (ours)	1168.5±21.7	91.5±6.6

TW is one of the few circuits for which not only the sparse structure is discovered, but also their synaptic polarity and synaptic types are identified (Wicks et al., 1996). We strictly preserved such *initialization* of synaptic structures throughout our experiments and observed a better performance consistently compared to other random circuits.

5.3. Performance

The training algorithm solved all the tasks, after a reasonable number of iterations, as shown in the learning curves in Fig. 4A-D. Jumps in the learning curves of the mountain car (Fig. 4B) are the consequence of the sparse reward. For the deterministic parking trajectory, the learning curve converges in less than 5000 iterations.

ONCs' sample efficiency is highly dependent on the environment in which they are being evaluated. As shown in Fig. 4, TW compared to LSTM, is more sample efficient in Half-cheetah and the pendulum, and less in Mountain-car. It also realizes a better sampling efficiency to MLP in Half-Cheetah, a similar rate in Pendulum, and worst in Mountain-car.

5.4. How does ONC + random search compare with policy gradient-based algorithms?

ONCs + Random search algorithm demonstrates comparable performance to the state-of-the-art policy gradient RL algorithms such as Proximal Policy Optimization (PPO) (Schulman et al., 2017), and advantage actor-critic (A2C) (Mnih et al., 2016). Table 3 reports the performance of the mentioned algorithms compared to NPC+RS.

5.5. How does ONC compare to deep learning models?

The final return values for the basic standard RL tasks (provided in Table 4), matches that of conventional policies (Heidrich-Meisner & Igel, 2008), and the state-of-the-art deep neural network policies learned by many RL algorithms (Schulman et al., 2017; Berkenkamp et al., 2017).

Table 4. Compare ONC with deep learning models. numbers show the Mean, standard deviation, and success rate for 10 runs. $N = 10$

Agent	**Inverted Pendulum**	**Mountaincar**	**HalfCheetah**	Sparsity
LSTM	629.01 ± 453.1 (40.0%)	**97.5** ± 1.25 (100.0%)	1588.9 ± 353.8 (10.0%)	0% (fully connected)
MLP	**1177.49** ± 31.8 (100.0%)	95.9 ± 1.86 (100.0%)	1271.8 ± 634.4 (0.0%)	0% (fully connected)
ONC (ours)	**1168.5**± 21.7 (90.0%)	91.5 ± 6.6 (80.0%)	**2587.4** ± 846.8 (72.7%)	**77% (28 synapses)**
Random circuit	138.10 ± 263.2 (10.00%)	54.01± 44.63 (50.0%)	1743.0 ± 642.3 (50.0%)	77% (28 synapses)

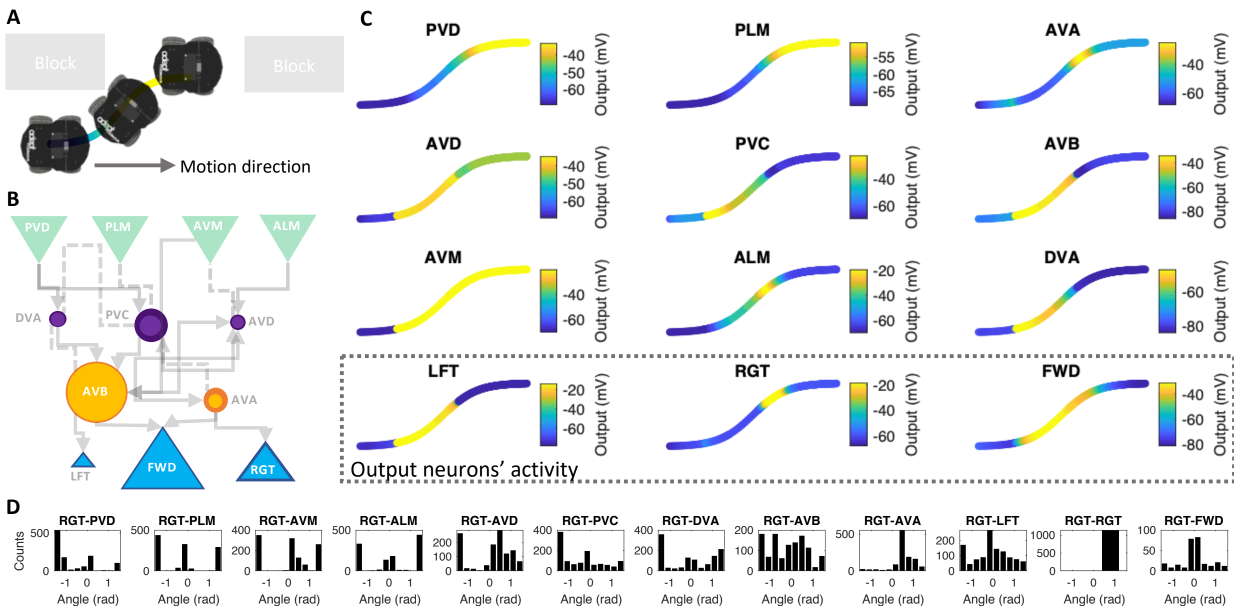

Figure 5. Interpretability analysis of the parking task. A) The parking trajectory. B) TW circuit drawn with the range of possible variations of the individual neuron's time-constants; the radius of the darker color circle for each neuron corresponds to the range within which the time-constant varies between τ_{min} and τ_{max} while the robot performs the parking. (Values in Supplementary Materials, Table S7). C) Projection of individual neuron's output over the parking trajectory. The plots demonstrate when neurons get activated while the rover is performing the parking task. D) Histogram of the slopes in manifolds' point-pair angles for a motor neuron in the parking task. (See Supplementary Materials Section 6, for full circuit's analyses, in other experiments.)

We compared the performance of the learned TW circuit to long short-term memory (LSTM) recurrent neural networks (Hochreiter & Schmidhuber, 1997), multi-layer perceptrons (MLP), and random circuits.

We tried to keep the comparison to other models as fair as possible; not only the number of neurons, their linear mapping, and their learning algorithm are the same, but also we let the trainable parameters of the other models to be larger than TW (e.g., in HalfCheetah, the total number of params for TW is 102, for MLP is 104, and for LSTM is 169) and we see TW's superior performance.

We select the same number of cells (neurons) for the LSTM and MLP networks, equal to the size of the tap-withdrawal circuit. LSTM and MLP networks are fully connected, while the TW circuit realizes a 77% network sparsity.

In simple experiments, the TW circuit performs in par with the MLP and LSTM networks, while in HalfCheetah, it significantly achieves a better performance. Results are summarized in Table 4.

5.6. Interpretability of the ordinary neural circuits

In this section, we introduce a systematic method for interpreting the internal dynamics of an ONC. The technique determines how the kinetics of sensory neurons and interneurons relate to a motor neuron's decision. Fig. 5B illustrates how various adaptive time-constants are realized in the parking environment. Interneurons (particularly PVC and AVA) change their time-constants significantly compared to the other nodes. This corresponds to their contribution to various dynamical modes and their ability to toggle between dynamic phases of an output decision.

Fig. 5C visualizes the activity of individual TW neurons (lighter colors correspond to a more activation phase) over the parking trajectory.

It becomes qualitatively explainable how individual neurons learned to contribute to performing autonomous parking.

For instance, AVA, the command neuron for turning the robot to the right-hand side (Motor neuron RGT) while moving, gets highly activated during a right-turn. Similarly, AVB and LFT neurons are excited during a left-turning phase. (See Fig. 5C).

Next, we formalize a quantitative measure of an ONC element's contribution to its output decision.

Definition 2. *Let $I = [0, T]$ be a finite simulation time of an ONC with k input neurons, N interneurons and n motor neurons, (Shown in Fig. 1), acting in an RL environment. For every neuron-pair (N_i,n_j), (N_i, N_j) and (k_i, n_j), in a cross-correlation space, let $S = \{s_1, ...s_{T-1}\}$ be the set of the gradients amongst every consecutive simulation time-points, and $\Omega = \{\arctan(s_1), ..., \arctan(s_{T-1}\}$ be the set of all corresponding geometrical angles, bounded to a range $[-\frac{\pi}{2}, \frac{\pi}{2}]$. Given the input dynamics, we quantify the way sensory neurons and interneurons contribute to motor neurons' dynamics, by computing the histogram of all Ωs, with a bin-size equal to l (i.e. Fig 5D), as follows:*

- *If sum of bin-counts of all $\Omega > 0$, is more than half of the sum of bin-counts in the $\Omega < 0$, the overall contribution of N_i to n_j is positive.*
- *If sum of bin-counts of all $\Omega < 0$, is more than half of the sum of bin-counts in the $\Omega > 0$, the overall contribution of N_i to n_j is negative,*
- *Otherwise, N_i contributes in phases (switching between antagonistic and phase-alighted) activity of n_j, on determinable distinct periods in I.*

To exemplify the use of the proposed interpretability method, let us consider the neuronal activity of a learned circuit driving a rover robot autonomously on a parking trajectory.

Fig. 5D presents the histograms computed by using Definition 1 for the RGT motor neuron dynamics (i.e., the neuron responsible for turning the robot to the right) with respect to that of other neurons. Based on Definition 1, we mark AVM, AVD, AVA as positive contributors to the dynamics of the RGT motor neuron.

We determine PVD, PLM, and PVC as antagonistic contributors. Neurons such as DVA and AVB realized phase-changing dynamics where their activity toggles between positive and negative correlations, periodically. (For the analysis of the full networks' activities visit Supplementary Materials Section 6).

Such analysis is generalizable to the other environments too. (See Supplementary Materials Section 6). In that case, the algorithm determines principal neurons in terms of neuron's contribution to a network's output decision in computable intervals within a finite simulation time.

6. Scope and limitations

Scalability We emphasize that the field of connectome-analysis, although being in its infancy, is rapidly growing (Sarma et al., 2018; Gleeson et al., 2018; Cook et al., 2019). For instance, the discovery of the mapping of fruit fly's brain (Xu et al., 2020), in combination with our method, constructs an exciting prospective line of research. As our knowledge about connectomes grows, we are confident that our proposed approach emerges as a significant viewpoint casting on network-design paradigms in deep learning and deep RL, in more general domains.

Moreover, instead of solely scaling our experiments to larger problems, we diversified them to multiple settings, establishing a solid foundation for ONCs on well-established environments, and thus enabling the machine learning community to build over this new line of research. In this regard, our experiments included benchmarking RL tasks, sim-to-real robotics, a general framework for efficient network design, and higher dimensional observation/action spaces to the degree compatible with the natural neural circuit.

Network design and dynamical systems Design principles provided in this work are ad-hoc, although we made sure to provide statistically significant evidence to support our quantitative findings. Moreover, applying advanced but solely graph theory analysis to connectomes (Varshney et al., 2011) misses the control and dynamical systems aspect. Thus, an ideal platform would take both measures into account. This is an exciting line of research with very few proposals (Towlson & Barabási, 2020), including ours.

7. Conclusions

We showed the performance of ONCs in control environments as the natural lottery winner networks. We quantitatively demonstrated that the sub-networks taken directly from the small species' nervous system realize an attractive max-flow rate and, when trained in isolation, perform significantly better than randomly-wired circuits, as well as contemporary deep learning models in simulated and real-life tasks.

We experimentally demonstrated the interpretable control performance of the learned circuits in action and introduced a quantitative method to explain networks' dynamics. The proposed method can also be utilized as a building block for the interpretability of recurrent neural networks, despite a couple of fundamental studies (Karpathy et al., 2015; Chen et al., 2016; Olah et al., 2018; Hasani et al., 2019), is still a grand challenge to be addressed (Hasani, 2020).

Finally, we open-sourced our methodologies to encourage other researchers to further explore the attributes of ONCs and apply them to other control and RL domains.

Acknowledgements

RH and RG are partially supported by Horizon-2020 ECSEL Project grant No. 783163 (iDev40), Productive 4.0, and AT-BMBFW CPS-IoT Ecosystem. ML was supported in part by the Austrian Science Fund (FWF) under grant Z211-N23 (Wittgenstein Award). AA is supported by the National Science Foundation (NSF) Graduate Research Fellowship Program. RH and DR are partially supported by The Boeing Company and JP Morgan Chase. This research work is partially drawn from the PhD dissertation of RH.

References

Ardiel, E. L. and Rankin, C. H. An elegant mind: learning and memory in caenorhabditis elegans. *Learning & memory*, 17(4):191–201, 2010.

Bargmann, C. I. Chemosensation in c. elegans. *WormBook*, pp. 1–29, 2006.

Berkenkamp, F., Turchetta, M., Schoellig, A., and Krause, A. Safe model-based reinforcement learning with stability guarantees. In *Advances in Neural Information Processing Systems 30*, pp. 908–919. Curran Associates, Inc., 2017.

Boykov, Y. and Kolmogorov, V. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *IEEE transactions on pattern analysis and machine intelligence*, 26(9):1124–1137, 2004.

Chalfie, M., Sulston, J., White, J., Southgate, E., Thomson, J., and Brenner, S. The neural circuit for touch sensitivity in *Caenorhabditis elegans*. *Journal of Neuroscience*, 5 (4):956–964, 1985.

Chen, B. L., Hall, D. H., and Chklovskii, D. B. Wiring optimization can relate neuronal structure and function. *Proceedings of the National Academy of Sciences of the United States of America*, 103(12):4723–4728, 2006.

Chen, T. Q., Rubanova, Y., Bettencourt, J., and Duvenaud, D. K. Neural ordinary differential equations. In *Advances in neural information processing systems*, pp. 6571–6583, 2018.

Chen, X., Duan, Y., Houthooft, R., Schulman, J., Sutskever, I., and Abbeel, P. Infogan: Interpretable representation learning by information maximizing generative adversarial nets. In *Advances in Neural Information Processing Systems*, pp. 2172–2180, 2016.

Cook, S. J., Jarrell, T. A., Brittin, C. A., Wang, Y., Bloniarz, A. E., Yakovlev, M. A., Nguyen, K. C., Tang, L. T.-H., Bayer, E. A., Duerr, J. S., et al. Whole-animal connectomes of both caenorhabditis elegans sexes. *Nature*, 571 (7763):63–71, 2019.

Duan, Y., Chen, X., Houthooft, R., Schulman, J., and Abbeel, P. Benchmarking deep reinforcement learning for continuous control. In *International Conference on Machine Learning*, pp. 1329–1338, 2016.

Dupont, E., Doucet, A., and Teh, Y. W. Augmented neural odes. In *Advances in Neural Information Processing Systems*, pp. 3134–3144, 2019.

Frankle, J. and Carbin, M. The lottery ticket hypothesis: Finding sparse, trainable neural networks. *arXiv preprint arXiv:1803.03635*, 2018.

Gleeson, P., Lung, D., Grosu, R., Hasani, R., and Larson, S. D. c302: a multiscale framework for modelling the nervous system of caenorhabditis elegans. *Philosophical Transactions of the Royal Society B: Biological Sciences*, 373(1758):20170379, 2018.

Han, S., Pool, J., Tran, J., and Dally, W. Learning both weights and connections for efficient neural network. In *Advances in neural information processing systems*, pp. 1135–1143, 2015.

Hasani, R. *Interpretable Recurrent Neural Networks in Continuous-time Control Environments*. PhD dissertation, Technische Universität Wien, 05 2020.

Hasani, R., Amini, A., Lechner, M., Naser, F., Grosu, R., and Rus, D. Response characterization for auditing cell dynamics in long short-term memory networks. In *2019 International Joint Conference on Neural Networks (IJCNN)*, pp. 1–8. IEEE, 2019.

Hasani, R., Lechner, M., Amini, A., Rus, D., and Grosu, R. Liquid time-constant networks. *arXiv preprint arXiv:2006.04439*, 2020.

Hasani, R. M., Fuchs, M., Beneder, V., and Grosu, R. Non-associative learning representation in the nervous system of the nematode caenorhabditis elegans. *arXiv preprint arXiv:1703.06264*, 2017.

Hasani, R. M., Lechner, M., Amini, A., Rus, D., and Grosu, R. Liquid time-constant recurrent neural networks as universal approximators. *arXiv preprint arXiv:1811.00321*, 2018.

Hassibi, B. and Stork, D. G. Second order derivatives for network pruning: Optimal brain surgeon. In *Advances in neural information processing systems*, pp. 164–171, 1993.

Heidrich-Meisner, V. and Igel, C. Variable metric reinforcement learning methods applied to the noisy mountain car problem. In *European Workshop on Reinforcement Learning*, pp. 136–150. Springer, 2008.

Hinton, G., Vinyals, O., and Dean, J. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015.

Hochreiter, S. and Schmidhuber, J. Long short-term memory. *Neural computation*, 9(8):1735–1780, 1997.

Islam, M. A., Wang, Q., Hasani, R. M., Balún, O., Clarke, E. M., Grosu, R., and Smolka, S. A. Probabilistic reachability analysis of the tap withdrawal circuit in caenorhabditis elegans. In *2016 IEEE International High Level Design Validation and Test Workshop (HLDVT)*, pp. 170–177. IEEE, 2016.

Kaplan, H. S., Thula, O. S., Khoss, N., and Zimmer, M. Nested neuronal dynamics orchestrate a behavioral hierarchy across timescales. *Neuron*, 2019.

Karpathy, A., Johnson, J., and Fei-Fei, L. Visualizing and understanding recurrent networks. *arXiv preprint arXiv:1506.02078*, 2015.

Lechner, M. and Hasani, R. Learning long-term dependencies in irregularly-sampled time series. *arXiv preprint arXiv:2006.04418*, 2020.

Lechner, M., Hasani, R., Zimmer, M., Henzinger, T. A., and Grosu, R. Designing worm-inspired neural networks for interpretable robotic control. In *2019 International Conference on Robotics and Automation (ICRA)*, pp. 87–94. IEEE, 2019.

Lechner, M., Hasani, R., Rus, D., and Grosu, R. Gershgorin loss stabilizes the recurrent neural network compartment of an end-to-end robot learning scheme. In *2020 International Conference on Robotics and Automation (ICRA)*. IEEE, 2020.

LeCun, Y., Denker, J. S., and Solla, S. A. Optimal brain damage. In *Advances in neural information processing systems*, pp. 598–605, 1990.

Li, Z., Li, Y., Yi, Y., Huang, W., Yang, S., Niu, W., Zhang, L., Xu, Z., Qu, A., Wu, Z., et al. Dissecting a central flip-flop circuit that integrates contradictory sensory cues in c. elegans feeding regulation. *Nature communications*, 3(1):1–8, 2012.

Mania, H., Guy, A., and Recht, B. Simple random search provides a competitive approach to reinforcement learning. *arXiv preprint arXiv:1803.07055*, 2018.

Mnih, V., Badia, A. P., Mirza, M., Graves, A., Lillicrap, T., Harley, T., Silver, D., and Kavukcuoglu, K. Asynchronous methods for deep reinforcement learning. In *International conference on machine learning*, pp. 1928–1937, 2016.

Morcos, A., Yu, H., Paganini, M., and Tian, Y. One ticket to win them all: generalizing lottery ticket initializations across datasets and optimizers. In *Advances in Neural Information Processing Systems*, pp. 4933–4943, 2019.

Nichols, A. L., Eichler, T., Latham, R., and Zimmer, M. A global brain state underlies c. elegans sleep behavior. *Science*, 356(6344):eaam6851, 2017.

Olah, C., Satyanarayan, A., Johnson, I., Carter, S., Schubert, L., Ye, K., and Mordvintsev, A. The building blocks of interpretability. *Distill*, 3(3):e10, 2018.

Pérez-Escudero, A. and de Polavieja, G. G. Optimally wired subnetwork determines neuroanatomy of caenorhabditis elegans. *Proceedings of the National Academy of Sciences*, 104(43):17180–17185, 2007.

Press, W. H., Teukolsky, S. A., Vetterling, W. T., and Flannery, B. P. *Numerical Recipes 3rd Edition: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 3 edition, 2007.

Rankin, C. H., Beck, C. D., and Chiba, C. M. Caenorhabditis elegans: a new model system for the study of learning and memory. *Behavioural brain research*, 37(1):89–92, 1990.

Rastrigin, L. A. About convergence of random search method in extremal control of multi-parameter systems. *Avtomat. i Telemekh.*, 24:1467–1473, 1963.

Salimans, T., Ho, J., Chen, X., Sidor, S., and Sutskever, I. Evolution strategies as a scalable alternative to reinforcement learning. *arXiv preprint arXiv:1703.03864*, 2017.

Sarma, G. P., Lee, C. W., Portegys, T., Ghayoomie, V., Jacobs, T., Alicea, B., Cantarelli, M., Currie, M., Gerkin, R. C., Gingell, S., et al. Openworm: overview and recent advances in integrative biological simulation of caenorhabditis elegans. *Philosophical Transactions of the Royal Society B*, 373(1758):20170382, 2018.

Schulman, J., Wolski, F., Dhariwal, P., Radford, A., and Klimov, O. Proximal policy optimization algorithms. *arXiv preprint arXiv:1707.06347*, 2017.

Shiloach, Y. and Vishkin, U. An o (n2log n) parallel maxflow algorithm. *Journal of Algorithms*, 3(2):128–146, 1982.

Singh, S. P. and Sutton, R. S. Reinforcement learning with replacing eligibility traces. *Recent Advances in Reinforcement Learning*, pp. 123–158, 1996.

Spall, J. C. *Introduction to stochastic search and optimization: estimation, simulation, and control*, volume 65. John Wiley & Sons, 2005.

Towlson, E. K. and Barabási, A.-L. Synthetic ablations in the c. elegans nervous system. *Network Neuroscience*, 4 (1):200–216, 2020.

Varshney, L. R., Chen, B. L., Paniagua, E., Hall, D. H., and Chklovskii, D. B. Structural properties of the caenorhabditis elegans neuronal network. *PLoS computational biology*, 7(2):e1001066, 2011.

Wen, Q., Po, M. D., Hulme, E., Chen, S., Liu, X., Kwok, S. W., Gershow, M., Leifer, A. M., Butler, V., Fang-Yen, C., et al. Proprioceptive coupling within motor neurons drives c. elegans forward locomotion. *Neuron*, 76(4): 750–761, 2012.

White, J. G., Southgate, E., Thomson, J. N., and Brenner, S. The structure of the nervous system of the nematode *Caenorhabditis elegans*. *Philosophical Transactions of the Royal Society of London B: Biological Sciences*, 314 (1165):1–340, 1986.

Wicks, S. and Rankin, C. Integration of mechanosensory stimuli in *Caenorhabditis elegans*. *Journal of Neuroscience*, 15(3):2434–2444, 1995.

Wicks, S. R., Roehrig, C. J., and Rankin, C. H. A dynamic network simulation of the nematode tap withdrawal circuit: Predictions concerning synaptic function using behavioral criteria. *Journal of Neuroscience*, 16(12):4017–4031, 1996.

Xu, C. S., Januszewski, M., Lu, Z., Takemura, S.-y., Hayworth, K., Huang, G., Shinomiya, K., Maitin-Shepard, J., Ackerman, D., Berg, S., et al. A connectome of the adult drosophila central brain. *BioRxiv*, 2020.

Zhou, H., Lan, J., Liu, R., and Yosinski, J. Deconstructing lottery tickets: Zeros, signs, and the supermask. In *Advances in Neural Information Processing Systems*, pp. 3592–3602, 2019.