Graduate Training Center of Neuroscience

Computational Neuroscience

What is the neural basis of the cyclic leg movements underlying a fly's backward walking?

Laboratory report

presented by

Byoungsoo Kim

The study was supervised by

Professor Dr. Pavan P. Ramdya

Neuroengineering Laboratory, EPFL, Lausanne, Switzerland

Duration of lab rotation: 2024.10.07 - 2024.12.06

Date of submission: 2024.12.29

Abstract

The ventral nerve cord (VNC) of *Drosophila melanogaster* governs its limb movements. Descending neurons (DNs) relay commands from the brain to different parts of the VNC, enabling a wide range of complex motor behaviors. Among these, Moonwalker descending neurons (MDNs) are particularly intriguing. Upon activation, MDNs induce backward walking in flies, even in headless flies where connections between DNs are absent. Specifically, MDNs actively elicit cyclic movements of the hindlegs, which drive backward walking. To investigate the circuitry underlying MDN-driven hindleg oscillations, I utilized the recently acquired connectome of the fly's VNC. I constructed a connectome-constrained (CC) network of MDN-specific pathways responsible for hindleg movements and trained this network to reproduce desired leg joint oscillations. Analyzing the network structures revealed insights into the circuitry behind cyclic hindleg movements. The results demonstrated that even the simplest MDN-specific pathways governing the hindlegs can generate oscillations. Furthermore, simulations identified neurons that may play key roles in leg oscillations in flies in vivo. Finally, the results showed that oscillations are not solely generated by a simple balanced excitatory-inhibitory (E-I) neuron pair but rather by a larger circuit that includes this E-I pair.

Keywords: Fly backward walking, Moonwalker descending neurons, Connectome-constrained network, Computational simulation.

Acknowledgements

Neuroengineering
Laboratory

I sincerely thank Professor Pavan Ramdya and the Neuroengineering Laboratory at EPFL for the opportunity to work on this project. I am especially grateful to Femke Hurtak and Pembe Gizem Özdil for their close supervision and invaluable support throughout the project. I also thank Dr. Maite Azcorra for collecting the experimental dataset and answering my questions, and Femke Hurtak and Dr. Maite Azcorra for designing an excellent rotation project. Last but not least, I thank Professor Pavan Ramdya for his guidance and supervision.

Contents

Chapter 1

Introduction

The motor circuits in the ventral nerve cord (VNC) of *Drosophila melanogaster* play a critical role in controlling the leg parts during motor behaviors. Each leg part is primarily governed by its respective neuropil: T1, T2, and T3, which exhibit bilateral symmetry and correspond to the foreleg, midleg, and hindleg, respectively. Descending neurons (DNs) are another key component in motor control, as they connect the brain to the VNC, allowing motor commands to travel from the brain to the limbs.

A recent study have examined the role of descending neurons (DNs) in motor control[2]. In this paper, the authors reported that the activation of command-like DNs can recruit a population of DNs. However, some DNs, such as the Moonwalker descending neurons (MDNs), initiate backward walking upon activation without recruiting other DNs[3]. Experimental evidence indicates that MDNs specifically control the hindlegs during backward walking. This was demonstrated in studies where hindlegs were decoupled from mechanical ground coupling[4] or in headless flies[2], both of which confirmed MDN-specific control.

A comprehensive analysis of MDNs' role during backward walking concluded that the hindlegs drive this behavior[4]. Prior experiments from our lab further support this conclusion. Given these findings, we focused on understanding how MDNs control the hindlegs at the circuit level. Specifically, we investigated the underlying circuitry of the T3 neuropil, which MDNs directly innervate. To achieve this, we utilized the recently acquired MANC connectome[5], enabling us to construct the full connectivity graph of the MDN-T3 network. This network represents the connections between MDNs (inputs) and T3 motor neurons (outputs). Due to the bilateral symmetry of motor circuits, we analyzed the right-hand side network. Using this connectome-constrained network, we explored the circuitry responsible for cyclic hindleg movements during backward walking.

In this study, I aimed to answer the following question:

How does the T3 circuit in the VNC generate hind leg oscillation upon MDN stimulation?

To address this question, I followed these steps: (1) construct the connectome-constrained 2 hops MDN-T3 network, (2) train the network with experiment-approximated datasets and (3) investigate the trained network to identify the circuitry responsible for generating the oscillation. The 2 hops MDN-T3 network comprises a single recurrently connected interneuron layer between MDNs and the motor neurons. In this network, the shortest path between MDNs and motor neurons is one or two synapses. The rationale behind this choice is based on the hypothesis that the MDN-specific pathways within the T3 neuropil efficiently generate hindleg oscillation, as sufficient MDN stimulation immediately triggers backward walking. If a network with only one interneuron layer successfully generates the desired oscillation, it would support the hypothesis of the MDN-T3 network's efficiency.

Chapter 2

Results

2.1 Preparing the 2 hops MDN-T3 network

The first step in investigating the hind leg oscillation mechanism via computational simulations was constructing the connectome-constrained network. To do so, I extracted the connectivity graph of the 2 hops MDN-T3 network (Fig.1) from the MANC datasets[5] (Methods). The 2 hops MDN-T3 network had 112 neurons (2 MDNs, 46 interneurons, and 64 motor neurons). Among interneurons, 28 neurons were excitatory neurons and 18 neurons were inhibitory neurons.

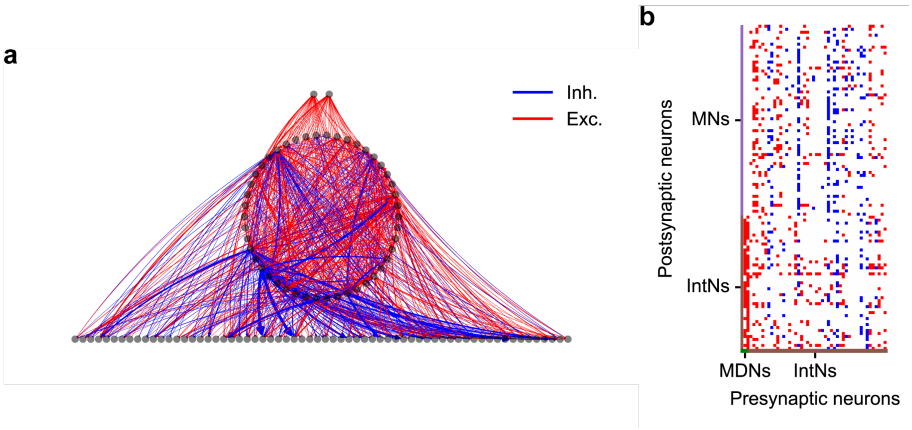


Figure 1. Visualization of the 2 hops MDN-T3 network

a, The connectivity graph of the 2 hops MDN-T3 network. Each gray dot is a neuron and each line is a connection between them. Directions for the connectivity are marked with the arrow. Two neurons on the top are MDNs, the neurons in a circle in the middle are the interneurons, and the neurons on the bottom are the motor neurons. **b**, The adjacency matrix of the 2 hops MDN-T3 network. Reds are excitatory connections, and blues are inhibitory connections. As the MDNs do not have presynaptic neurons and the motor neurons do not have postsynaptic neurons, corresponding rows and columns are omitted here.

2.2 Training the 2 hops MDN-T3 network using datasets derived from experimental observations

2.2.1 Decoding the leg joint movements from the motor neurons' neural activities

From the experiments, I could read out the angles of the leg joints, while I could only know the neural activities of the motor neurons from the 2 hops MDN-T3 network. To train the network, I transformed the motor neurons' outputs into joint movements using a decoder. First, based on the joints dynamics observed during backward walking[4], I decoded the coxa-trochanter (CxTr) and femur-tibia (FmrTi) joints. Second, due to the lack of annotation for hindleg motor neurons, I matched each motor neuron with its corresponding muscle using foreleg annotations[6].

To decode the two joint movements, I employed two linear decoders, each receiving outputs from the corresponding motor neurons. Since how each motor neuron controls each muscle is unknown, I initialized the decoders to treat all the input signals equally. To ensure that the decoders did not independently generate oscillations, I used the linear decoders (Methods). During training, the decoders were optimized to reproduce the desired joint angles.

2.2.2 The 2 hops MDN-T3 network learns naturalistic joint behaviors

For the training of the connectome-constrained network, I used curriculum learning [7] comprised of two sessions. The training datasets are derived from the experiments where we made flies walk backward by optogenetically stimulating the MDNs (Methods).

First session: learning the simple dataset

For the first session, the dataset with a simple input and output relationship was used (Methods). Since the learning process of a network depends on the initialization of its parameters, I trained 11 networks with parameters initialized using different random seeds. Each network was trained for either 10,000 or 15,000 epochs. Out of 11 networks, 9 networks passed the performance threshold (Fig.2a). Only the networks that passed the threshold went through the 2nd session of the curriculum learning.

Second session: learning the complex dataset

In the second session, the input durations and onset times were varied to create a training dataset designed to encourage the network to learn more naturalistic joint movements. After training the pre-trained 9 networks for 10,000 epochs, 7 networks achieved a loss below 0.05 (Fig.2b). These 7 networks were selected for further analysis due to their superior performance.

Testing the trained network

For the selected networks, I checked whether they can recapitulate naturalistic behaviors. Naturalistic behaviors mean whether the network only generates oscillation when there are inputs and stops without inputs. To test this, I injected two tonic pulses into the MDNs and compared the networks' outputs with the desired output (Fig.2c,d and Extended Data Fig.1). The trained network oscillates when there are inputs, and stops it almost immediately when the input stops. Also, it doesn't oscillate when there is no input. Across all trained networks, the test errors are significantly small, suggesting that the networks learned naturalistic behaviors through curriculum learning. For the joint outputs, only the CxTr joint angles are plotted, but the FmrTi joint angles also match the desired outputs.

2.3 Identifying the oscillatory mechanism of the 2 hops MDN-T3 network

The trained networks successfully generated cyclic leg movements in response to MDN stimulation, suggesting the presence of circuitry within the network that enables this behavior. Additionally, if the networks retained their original structure during training, it is likely that the same set of neurons were responsible for the oscillatory behavior. To identify this so-called "oscillatory circuit," I adopted a two-step approach: (1) verify whether training significantly altered the network structure or the role of individual neurons, and (2) identify the specific circuits responsible for the network's oscillatory behavior.

2.3.1 Training changes network structure, but not the role of individual neurons

As previously mentioned, networks can evolve differently during training based on their initial parameter configurations. To evaluate whether the 7 trained networks were similar to the original connectome-constrained network (pre-training) and to one another, I computed cosine similarities between their weight matrices (Methods). The results revealed that the global network structures changed during training. Compared to the original network, all the trained networks exhibited lower similarity than the control (Methods). Furthermore, the similarities among the trained networks themselves were also lower than those observed in the original network (Fig.3a). However, the absolute similarity values, calculated in relation to the untrained networks, remained high. This indicates that initializing connectivity with the connectome dataset helps preserve the network's structure throughout the training process.

Next, I investigated the role of individual neurons in generating the desired output. Since MDNs and motor neurons serve as the network's inputs and outputs, I focused this analysis on the interneurons. To assess their roles, I performed a silencing test on each interneuron (Methods). When silencing a neuron disrupted the output, the resulting combined loss reflected the neuron's effect on the network.

To quantify the significance of these effects, I measured the silencing impact using a silencing score (Methods). A high silencing score indicated that a neuron had a consistently high average silencing effect with minimal variance across different networks. While the combined losses varied across trained networks, each interneuron demonstrated

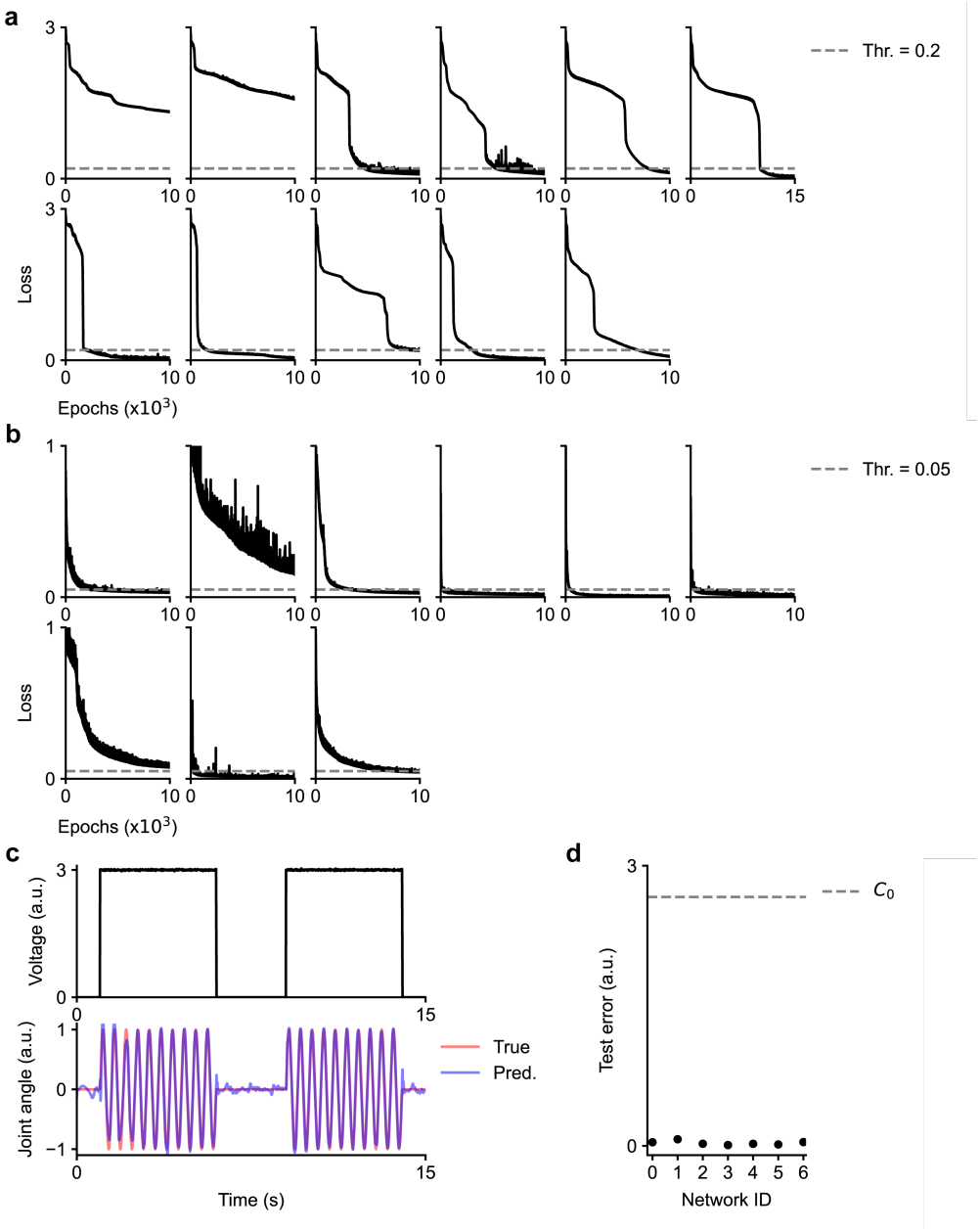


Figure 2. Training losses and the test output of the trained network

a, Training losses during the first session of curriculum learning across different networks. The gray dotted line indicates the performance threshold required for further training. **b**, Training losses during the second session of curriculum learning across different networks. The gray dotted line indicates the performance threshold required for further analysis. **c**, The test input and output of one example trained network. The desired readout and network readout of the CxTr joint are plotted. **d**, The test errors of all 7 networks. The errors here are combined losses (Methods). C_0 indicates the error when the network readout predicts the mean of the desired output.

similar silencing effects across networks (Fig.3b). Moreover, interneurons with high silencing scores were also identified as significant when compared to the null distribution (Methods), indicating non-trivial silencing effects across all trained networks. Interestingly, the silencing effects of inhibitory neurons were consistently higher than those of excitatory neurons.

I closely examined the connectivity of neurons ID 3 (neuron3) and 28 (neuron28) with other neurons. These neurons, classified as excitatory and inhibitory, respectively, demonstrated the strongest silencing effects within their groups. Notably, they were reciprocally connected and exhibited broad projections to numerous motor neurons.

Based on these results, I confirmed that (1) training altered the network's connectivity beyond chance levels, while the absolute cosine similarities remained consistently high across all trained networks, and (2) the roles of individual interneurons, particularly those with strong silencing effects, remained similar across different trained networks. Thus, I concluded that training did not significantly alter the overall network properties.

All trained networks were capable of generating oscillations, as previously confirmed (Fig.2, Extended Data

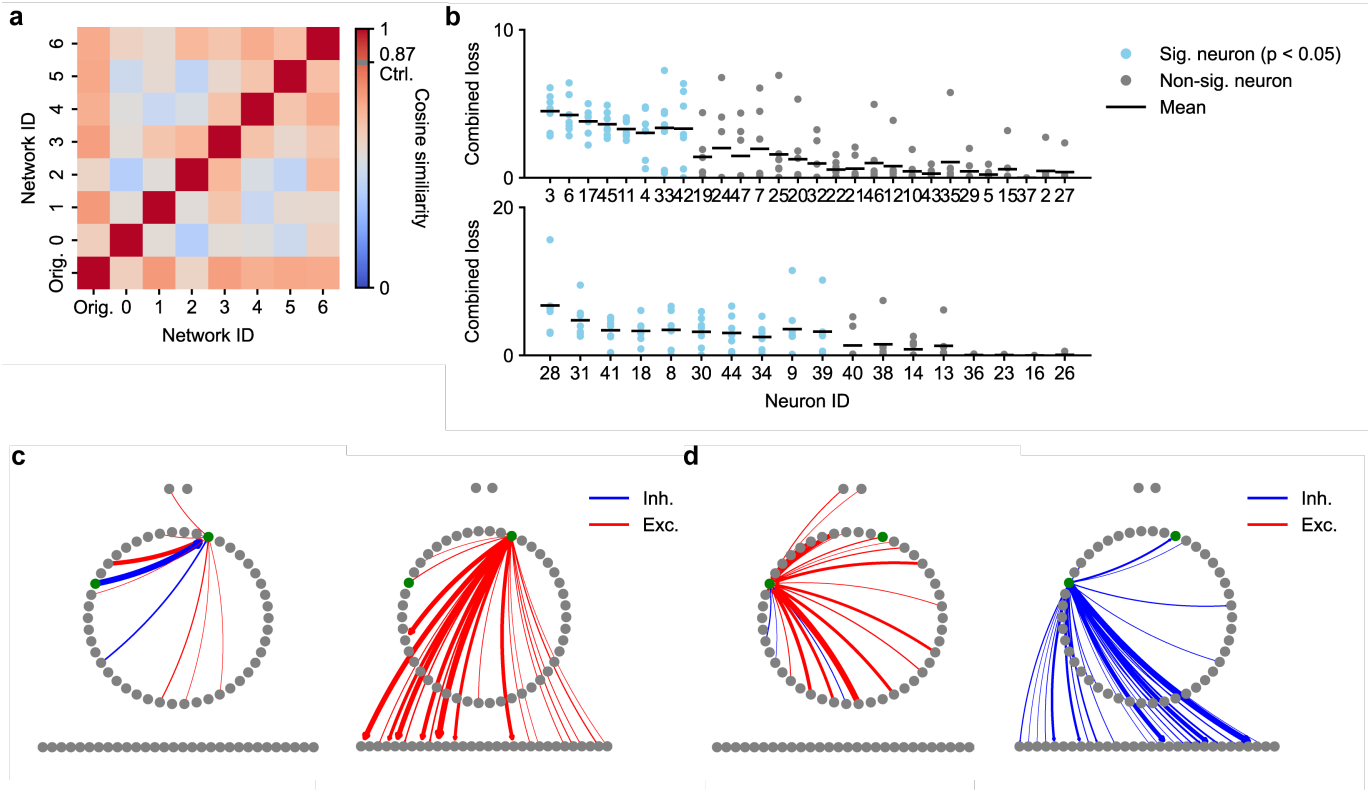


Figure 3. Impact of training on network structure and individual interneurons

a, The cosine similarity matrix between the original network and other trained networks. The similarity value between the original network and the control network (Ctrl.) is indicated by a gray line on the colorbar. **b**, Results of the silencing test for all interneurons across all trained networks. The top row shows the results for excitatory neurons, while the bottom row presents those for inhibitory neurons. Statistically significant neurons are highlighted in different colors. Combined loss represents the sum of losses from the CxTr and FmrTi joint outputs. **c**, Connectivity graphs for neuron3, with incoming connections on the left and outgoing connections on the right. The thickness of each line indicates the relative connection strength. Neuron3 and neuron28 are highlighted as green dots. The neuron layout matches that in Fig.1a. **d**, Connectivity graphs for neuron 28, formatted identically to **c**.

Fig.1). This behavior aligns with the expected dynamics of the 2 hops MDN-T3 network, the minimal structure of MDN-specific pathways within the T3 neuropil. Additionally, since training tended to preserve the impact of each interneuron, I proceeded to identify the oscillatory circuit within the network, which could be conserved across trained networks.

2.3.2 Only a handful of neurons oscillate in response to MDN stimulation

First, I analyzed the neural activities of all neurons. Strikingly, only a small subset exhibited significant 2 Hz oscillations (Fig.4a, Supplementary Video 1). Furthermore, these oscillating neurons were not in phase (Fig.4b, Extended Data Fig.2), a necessary condition for being the source of oscillation generation. Therefore, for further investigations of the oscillatory circuits, I focused on these oscillating neurons.

2.3.3 Hypothesis: Reciprocal connections between excitatory and inhibitory (E-I) neurons drive oscillation generation

I defined the group of oscillating neurons using two criteria: (1) a neuron must oscillate with a 2 Hz power greater than 5, and (2) the group must consist of no more than 8 neurons. These criteria enabled me to analyze the network of significantly oscillating neurons (oscillatory circuit) while keeping the network manageable for detailed examination.

Across the 7 trained networks, the oscillatory circuits did not exhibit a consistent global connectivity pattern (Fig.4c). This observation suggests that the networks did not converge on the same solution for the given task. However, at least one of neuron3 or neuron28 was an oscillating neuron in most networks (6 out of 7). Furthermore, of the 25 neurons identified as oscillating in at least one network, 19 appeared as oscillating neurons in more than one network (Fig.4d). These results confirmed that the networks recruited a similar set of neurons to learn the oscillation.

Therefore, I next explored whether the networks employed a similar mechanism to generate the oscillation using these neurons.

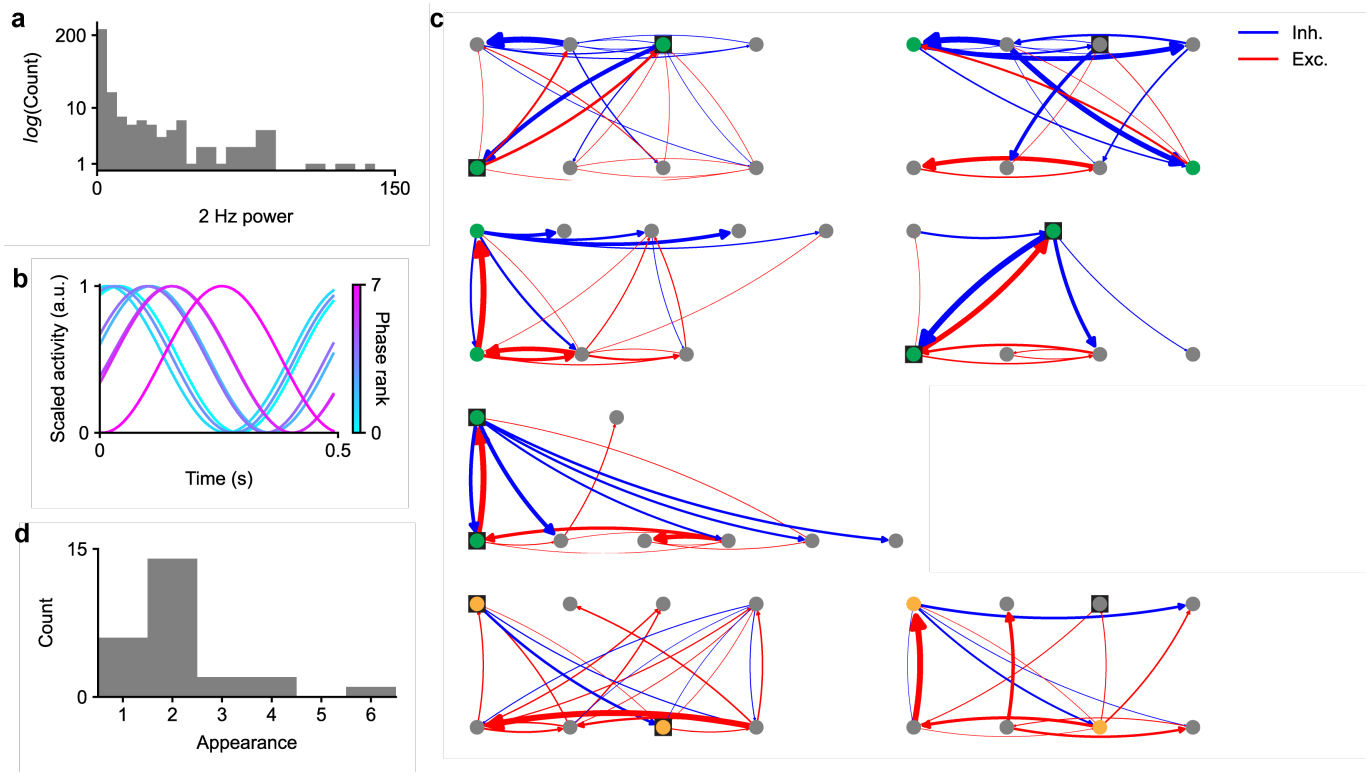


Figure 4. Identifying oscillatory circuits across the trained networks

a, Histogram of 2 Hz power across all interneurons from all trained networks (total: 323 neurons). Note that the y-axis is displayed on a logarithmic scale. **b**, Phase relationship between oscillating neurons in an example trained network. Each neuron's activity is normalized (min-max scaling) and smoothed using a Savitzky-Golay filter. Neurons are color-coded based on their oscillation onset time following MDN stimulation. **c**, Oscillatory circuits from 7 trained networks. In each circuit, inhibitory neurons are shown in the top row, and excitatory neurons in the bottom row. Each dot represents a neuron, and the line widths between neurons indicate relative connection strength. E-I pairs with direct significant reciprocal connections are marked with green dots, while those with indirect connections are marked in orange. Neuron3 and neuron28 are highlighted with a black square around their dots. **d**, Histogram showing the frequency of recurrent appearances of oscillating neurons (total: 25 neurons).

Classical circuit mechanisms for oscillation generation include reciprocal connections between two inhibitory neurons or the ability of a single neuron to burst independently[8]. However, the neural dynamics used in this study did not support single-neuron burst dynamics, nor were reciprocal inhibitory connections observed in any of the oscillatory circuits. Instead, across all networks, reciprocally connected E-I neuron pairs with relatively strong connections were identified (Fig.4c). Even in networks lacking significant direct connections between E-I neuron pairs, strong connections existed through a small number of intermediary neurons. Given the neural dynamics used, a range of parameters enables these E-I pairs to generate oscillations independently[9]. Based on this, I hypothesized that the 2-hop MDN-T3 network utilizes the coupled dynamics of E-I neuron pairs for oscillation generation and proceeded to investigate this hypothesis.

2.3.4 Single E-I Pair cannot generate oscillations alone

To test the proposed hypothesis, I followed three steps: (1) analyzing the silencing test results of the E-I neuron pairs, (2) examining the silencing test results of the connections between E-I neuron pairs, and (3) conducting a stability analysis on the coupled neural dynamics of the E-I neuron pairs (Methods). To simplify the analysis, I focused on the 5 trained networks with significant direct connections between E-I neuron pairs.

Silencing E-I pairs significantly disrupts oscillations

I silenced either an E or an I neuron in each E-I pair and compared the network's outputs to the desired outputs. Of the 10 neurons tested (one pair per network), silencing 9 completely suppressed oscillation generation upon MDN

stimulation. In one case, the decoded joint angle oscillated, but the phase progressively delayed (Fig.5a, Extended Data Fig.3a). These results suggested that the E-I neuron pairs play a significant role in oscillation generation. However, they do not prove that the oscillations were generated from each pair; rather, the one neuron without disruption in oscillation generation suggested that there might be other circuitry that gives rise to oscillations in the network.

Some E-I pairs are directly responsible for oscillations

To further investigate, I silenced the connections between each E-I pair separately in individual trials, testing the E→I and I→E connections independently. Silencing these connections aimed to determine whether the coupling of E-I neurons is directly responsible for oscillation generation, as opposed to the previous neuron silencing results, which could reflect cascading effects—where silencing one neuron indirectly suppresses others that significantly influence the network's output.

Silencing 7 of 10 connections significantly disrupted oscillation generation upon MDN stimulation. For the remaining 3 connections, the network continued to oscillate, though with differences: one connection resulted in oscillations with a lower frequency, while the other two produced oscillations with increasing phase delay (Fig.5b, Extended Data Fig.3b). These results suggest that in some trained networks (2 of 5), the E-I pair is not solely responsible for oscillation generation, indicating that multiple solutions may exist for the untrained network to converge upon during training.

In contrast, for networks where silencing caused significant disruption (3 of 5), interactions between the E and I neurons within the pair were essential for generating oscillations. Interestingly, the E-I pairs in these networks always involved neuron3 (E neuron) and neuron28 (I neuron). However, it remains unclear whether these pairs alone are sufficient to generate oscillations. A dynamical stability analysis of the E-I pair would be required to answer this question.

The E-I pairs are not solely responsible for the oscillation generation

Testing the stability of the coupled dynamics between the E and I neurons assessed whether each pair could independently generate oscillations during MDN stimulation. If the coupled system exhibited a limit cycle solution, it could generate oscillations on its own. For this analysis, I examined the E-I pairs from three networks that showed significant silencing effects (1st, 4th, and 5th rows in Fig.5a,b). The phase portraits of these systems revealed that none of the E-I pairs exhibited a limit cycle solution (Fig.5c). These results indicate that additional neurons beyond the E-I pairs are involved in driving the networks' oscillations, thus disproving the hypothesis that reciprocal connections between E and I neurons alone drive oscillation generation.

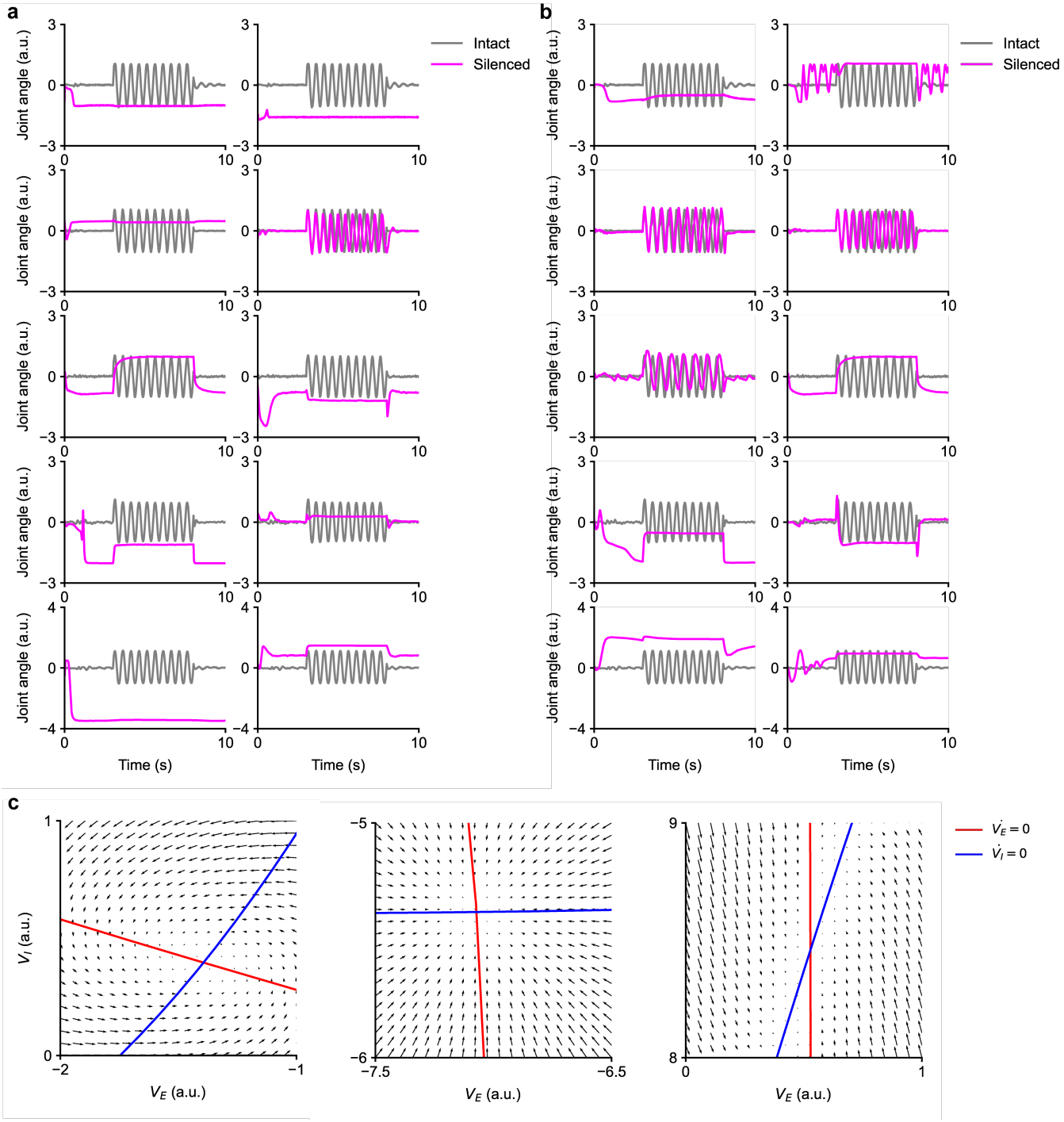


Figure 5. Joint angle outputs from silencing tests and stability analysis of E-I pairs

a, Outputs of the CxTr joint angles from intact and silenced networks across 5 different networks. Each row represents one network. For each trial, one neuron in the E-I pair was silenced. Silencing results for I neurons are shown on the left, and E neurons on the right. The network order corresponds to Fig.4c. **b**, Similar to **a**, but showing the silencing results of the E→I or I→E connections. Results for I→E connection silencing are on the left, and E→I on the right. **c**, Phase portraits of the E-I pair dynamics. From left to right, each panel corresponds to the pair from the 1st, 4th, and 5th rows in **a** and **b**.

Chapter 3

Discussion

In this study, I demonstrated that the simplest MDN-specific network in the T3 neuropil of the fly's VNC—the 2 hops MDN-T3 network—can be trained to generate expected cyclic joint movements. This finding supports the idea that the MDN-T3 network can efficiently drive the hindleg oscillation upon the excitation of MDNs. Additionally, training the connectome-constrained network identified key-player neurons (neuron3 and neuron28) as potential contributors to oscillation generation. While the coupling of E and I neurons plays a significant role, the E-I pair alone is not sufficient to generate cyclic leg movements in the 2 hops MDN-T3 network.

Several aspects of this study could be improved. First, the impact of training on network structure and dynamics was not fully explored. While I demonstrated certain regularities in network connectivity and the roles of individual neurons (Fig.3a,b), the observed changes in network structure exceeded chance levels. This suggests that training may alter connectivity to enforce task performance rather than leveraging the existing architecture—an assumption fundamental to the use of connectome-constrained networks. Addressing this limitation could involve (1) comparing neural recordings of the T3 neuropil with the activity outputs of the trained networks or (2) systematically analyzing how training affects network properties.

Second, the stability analyses of the E-I pairs did not account for the time dependency of their inputs. Since inputs are critical parameters influencing the dynamics of the system, incorporating this factor could significantly alter the results. Thus, a more rigorous analysis of dynamical stability is necessary.

Third, to identify the minimal subset of the network responsible for oscillation generation, an alternative silencing approach could be employed. Instead of silencing one neuron per trial, progressively increasing the number of neurons silenced could help determine the smallest network subset capable of sustaining oscillations. While this could be performed using a brute-force method, systematically selecting neurons to silence based on single-neuron silencing results would narrow the search space and improve efficiency.

Fourth, the neuronal model used in this study was rate-based, whereas neurons in the T3 neuropil are primarily spiking. Building the network with rate-based dynamics could lead to significant differences in its underlying mechanisms. Although a previous study showed that the desired function emerged in both spiking and non-spiking models on a certain network[10], employing a more biologically realistic neuronal model may still provide deeper insights into the circuitry underlying leg oscillations. However, as this approach would increase the number of parameters to train, an alternative methodology may be required.

Fifth, the decoder architecture can be improved. Due to the limited understanding of how motor neurons control individual muscles, I treated all motor neuron outputs equally in the decoder design. This approach assumes that the decoder converts motor neuron outputs into joint angles in a manner similar to how the fly's leg operates. However, this is a strong assumption with significant room for improvement. For instance, the decoder could differentiate between motor neurons that innervate extensor and flexor muscles or employ a more complex architecture than a fully connected linear decoder.

Lastly, the results I described are based on the 7 different trained networks. As it is clear from the results that the training procedure has dependency on the initialization of the parameters, we need to train more networks to investigate the consistency and variability of the trained networks.

There were multiple results that could not be fully explored due to time constraints. One intriguing finding is that inhibitory neurons exhibited stronger silencing effects than excitatory neurons (Fig.3b). This suggests that inhibitory neurons may play a more critical role in generating cyclic joint movements. Based on preliminary observations of the datasets, I hypothesize that inhibitory neurons may turn off already excited neurons upon MDN stimulation. However, this hypothesis requires further investigation.

Second, the dynamical stability analysis of the E-I pair confirmed that additional neurons should contribute to oscillation generation. A brief investigation revealed that other neurons within the oscillatory circuits, aside from those in the E-I pair, exhibited significant silencing effects (Extended Data Fig.4). This suggests that additional neurons within the oscillatory circuits play a significant role in oscillation generation. Therefore, investigating the connections and dynamical stability of these neurons, along with the E-I pairs, could help uncover the circuitry underlying cyclic joint movements.

Third, the networks exhibited two distinct behaviors for oscillation generation. Observing the neural activities across all trained networks revealed that in some networks, certain neurons oscillated independently of MDN stimulation, while in others, they did not. These differences could be inferred from the steady-state outputs (Supplementary Analysis). This analysis confirmed that some networks oscillate without any inputs, while others require stimulation, suggesting that the networks converged on different solutions to achieve the given task. Investigating the differences between these two groups of networks could uncover the circuit mechanisms underlying these distinct solutions.

Fourth, the mechanism by which the network gates oscillations remains unexplored. As previously mentioned, some networks exhibited oscillatory behavior without MDN input. Additionally, observing neural activity revealed that many neurons became active upon MDN stimulation. This suggests the existence of an underlying circuit that initiates or amplifies the oscillation and transfers it to motor neurons. An eigendecomposition of the weight matrix (Supplementary Analysis) can be useful for this analysis, as it yields eigencircuits. If an eigencircuit includes neurons from the oscillatory circuits along with other neurons that activate or deactivate upon MDN stimulation, those neurons are likely responsible for gating the joint oscillation. Therefore, looking closely at these eigencircuits can provide valuable clues to understanding the underlying mechanisms of oscillation gating.

Lastly, while I trained networks other than the 2 hops MDN-T3 network (Supplementary Analysis), I could not study them in depth. Comparing these networks with the 2 hops MDN-T3 network presents intriguing questions. In this study, I showed that the 2 hops MDN-T3 network can generate oscillations upon MDN input and investigated its oscillatory circuits. This raises subsequent questions: (1) Does the 2 hops MDN-T3 network have a specific connectivity motif that enables oscillation generation? and (2) Are these oscillatory circuits preserved in the 3 hops MDN-T3 network or even in the whole MDN-T3 network? These questions can be addressed by, for example, (1) comparing the 2 hops MDN-T3 network with the 2 hops MDN-T1 network, and (2) investigating the 3 hops MDN-T3 network and the entire MDN-T3 network, which would require training each network. Exploring these questions could further validate the underlying hypothesis that "MDN-specific pathways in the T3 neuropil can efficiently drive hindleg oscillation".

Chapter 4

Methods

4.1 Construction of the connectome-constrained networks

4.1.1 Extracting the connectivity graph and constructing the MDN-neuropil networks

Femke Hurtak helped me a lot with this process. We extracted the desired connectivity diagram based on the given annotations on the MANC v1.0 dataset[5]. From the dataset, we could extract the precise location of each neuron and its classes (e.g. descending neurons, motor neurons), along with the synaptic counts between the neurons. From the extracted connectivity data, we constructed the weight matrix. For the weight matrix, the synaptic counts between the pair of neurons are assigned as the weights for the pair's connection. In order to construct the MDN-neuropil networks, we further considered these points. First, we focused on neuropils on the right-hand side. Second, as the MDNs have axonal projections to T1, T2, and T3 neuropils, we cut them into three parts based on each neuropil. This processing eliminated the possible connections of this scenario. If you want connections from the MDNs to T1 specifically, using the raw MDNs might end up with connections such as MDN to a neuron in T2 and back to a neuron in T1. Lastly, for the MDN-neuropil networks, we only considered the connections between the neurons within the neuropil, other than the incoming projections from the MDNs to the neuropil.

4.1.2 Pruning and subsetting the networks

To prepare the network for the training, I first prune the neurons without postsynaptic neurons. However, as most of the motor neurons do not have postsynaptic neurons, I kept them without pruning. This step was necessary to lower the number of free parameters to train. After this step, I subsetted the networks based on the given interest. For example, to get the 2 hops MDN-T3 network, I started from the MDN-T3 network. Then, I subsetted the network by finding the path between the MDNs and the motor neurons in T3, connected directly or with one interneuron that is also in T3. After identifying all the interneurons that connect the MDNs and the T3 motor neurons, I reconstructed the weight matrix for this 2 hops MDN-T3 network. For this weight matrix, I considered the connections from MDNs to interneurons, all the connections between the interneurons, and the connections from the interneurons to the motor neurons. All the interneurons and motor neurons are in the T3 neuropil.

4.1.3 Assigning cell-type to each neuron

I assigned cell types to different neurons to group neurons and lower the number of free parameters during training. As the role of the interneurons within a given neuropil (e.g. T3) is not well-known, all interneurons have unique cell types. For the MDNs, I set them to have the same cell type and the same for the motor neurons.

4.2 Training the connectome-constrained networks

Pembe Gizem Özdil helped me a lot with this process. For the training, we adapted the previously published work and its Python package[11].

4.2.1 Neuronal dynamics and trainable parameters

Each cell-type has its own parameters for the dynamics. For the neuronal model of each cell-type, I used the leaky integrator model where the voltage dynamics $v_i(t)$ is governed by these equations:

$$\tau_i \dot{v_i}(t) = -v_i(t) + v_i^{rest} + \sum_{j=0}^{N} w_{j,i} f(v_j(t)) + I_i^{ext}(t) + \eta(t), \tag{4.1}$$

$$f(x) = \frac{1}{1+e^{-x}} \tag{4.2}$$

$$0 \leq i \leq N,\ N \text{ is the number of all cell-types}$$

Here, τ_i is the membrane time constant, v_i^{rest} is the membrane resting potential, $I_i^{ext}(t)$ is the external input, and $\eta(t)$ is the intrinsic noise sampled from $N(0, 0.02)$ with a unit mV. τ_is are initalized as 30 ms and v_i^{rest} are initialized by randomly sampling it from $N(-40, 5)$ with a unit mV. I used the sigmoid function $f(x)$ to model neurotransmitter release. The weights between cell-types are set to be:

$$w_{j,i} = \alpha_i \sigma_{j,i} \mathrm{N}_{j,i}^{syn\_count} \tag{4.3}$$

Here, α_i is the sign of the connection. Excitatory neurons have the value $+1$ and inhibitory neurons -1. If a neuron releases GABA or glutamate, I set them as inhibitory and, for acetylcholine, as excitatory. Otherwise, I set $\alpha_i = 0$. The scaling factor $\sigma_{j,i}$ is initialized as:

$$\sigma_{j,i} = \frac{0.1}{\mathrm{N}_{j,i}^{syn\_count}} \tag{4.4}$$

The synaptic counts $\mathrm{N}_{j,i}^{syn\_count}$ are based on the extracted connectivity matrix (see **Construction of the connectome-constrained networks**). The $\mathrm{N}_{j,i}^{syn\_count}$ is initialized as the natural logarithm of the mean of synaptic counts of connections from cell-types i to j. The time constant τ_i, the resting potential v_i^{rest} and the scaling factor $\sigma_{j,i}$ are trainable parameters. Specifically, each cell-type has its own time constant and resting potential, and the scaling factors are uniquely assigned to each pair of two cell-types.

4.2.2 Decoder architecture

Linear decoders were used to convert the outputs of the motor neurons into the leg joint movements. Each joint movement had its decoder. For input $\mathbf{X}$, the decoder gives output y as following:

$$y = \mathbf{a} \cdot \mathbf{X} + \mathbf{b} \tag{4.5}$$

Here, y is scalar but the rests are vectors. Throughout the training, the decoder tunes $\mathbf{a}$ and $\mathbf{b}$ to match the desired output.

4.2.3 Experiments approximated training dataset

Analyzing the experimental data

To construct the training dataset, I approximated the experimental data. Dr. Maite Azcorra collected the experimental data. The experimental data was collected by optogenetically stimulating the MDNs with three different input levels (laser powers: 10, 20, and 30 mW) while recording the leg movements of the fly. The acquired data are processed using DeepFly3d [12] and the following functions (https://github.com/NeLy-EPFL/df3dPostProcessing.git) to extract the joint angles of the right hind leg. Among all the joints, I extracted the oscillation frequency of the coxa-trochanter joint and the femur-tibia joint by Fourier transforming the time series of each joint angle. The resulting frequency was 2 Hz regardless of the inputs.

Designing the experiments approximated training dataset

I constructed one simple training dataset and one complex training dataset. For the simple dataset, the each input is 5 s tonic pulse with small noises and its intensity is randomly selected among 3 different input levels. The onset time of the input is fixed. The outputs are 2 Hz sinusoidal wave ranging from -1 to 1 during the input time. I used 30 samples for the training, each sample lasts 10 s. For the complex data set, the duration and the onset of the input time are altered. The duration can be either 1, 3 or 5 s, and the onset time was randomized. The paired output has the 2 Hz oscillation during the input time. I used 54 samples for the training, each sample lasts 10 s.

4.2.4 Training procedures

The trainable parameters were optimized every step through Backpropagation Through Time (BPTT)[13], to the direction that minimized the loss. I used the optimizer AMSGrad[14], a variance of Adam optimizer with a learning rate of 10^{-4} and batch size of 4 over 10,000 or 15,000 epochs. For the 2 hops MDN-T3 network, there were total 478 trainable parameters (96 parameters from neuronal dynamics, 382 parameters from weight scaling factors). Below is some details about the training procedures.

Loss function

During training, the loss, $\mathcal{L}$, between the desired output, y, and the decoder output, $\hat{y}$ was calculated based on this equation:

$$\mathcal{L}(y, \hat{y}) = \text{MSE}(y, \hat{y}) + 1 - \text{Corr}(y, \hat{y}) \tag{4.6}$$

Here, the loss is the combination of mean squared error and the Pearson correlation, which I called "combined loss"

Curriculum learning

For the training, I used curriculum learning[7]. During the first session of the curriculum learning, the network learned the simple training datasets for 10,000 or 15,000 epochs. If the loss reached less than 0.05, then the network went through the second session of the training, learning the complex training datasets for 10,000 epochs.

4.3 Analyzing connectome-constrained networks

4.3.1 Calculating network similarity

To compare the similarities between the given networks, I computed the cosine similarity of the weight matrices. For the weight matrices, $\mathbf{W_A}$ and $\mathbf{W_B}$, the cosine similarity was obtained as following:

$$\text{Cosine-similarity}(\mathbf{W_A}, \mathbf{W_B}) = \frac{\text{vec}(\mathbf{W_A}) \cdot \text{vec}(\mathbf{W_B})}{|\text{vec}(\mathbf{W_A})||\text{vec}(\mathbf{W_B})|} \tag{4.7}$$

Getting the network similarity of the control network

To establish a baseline for the cosine similarities between networks, I prepared a control network and computed its cosine similarity with the original (untrained) network. For the control network, scaling factors (σ) for each connection were randomly assigned values between 0 and 1.

4.3.2 Silencing test

To test the significance of a given node (neuron) or edge (connection between two neurons), I conducted silencing tests through following steps:

1. Set the weights of the target connections to 0, this procedure constructed a silenced network.
2. Give the input of 5 s tonic pulse to the silenced network and read the output.
3. Compute the combined loss (see **Loss function**) between the test output and the desired output.

Node silencing

For the node silencing, the weights of the connections going out from the given node were set to 0 at the step 1.

Edge silencing

For the edge silencing, the weight of the given connection were set to 0 at the step 1.

Silencing score

To measure the significance of the silencing of a given node i, I defined "silencing score" as follow:

For a neuron i silencing results across given seeds,

$$\text{Silencing-score}(i) = \bar{\mathcal{L}_i} - 0.5\sigma(\mathcal{L}_i) \tag{4.8}$$

$\bar{\mathcal{L}_i}$: Average combined loss,
$\sigma(\mathcal{L}_i)$: Standard deviation of combined losses

Significance test

To test the significance of an interneuron's silencing effect across trained networks, I conducted a two-sided permutation test. A null distribution was created by pooling all silencing scores from all interneurons across all networks. Random samples (equal in size to the number of trained networks) were drawn, and their silencing scores were computed to test for significance. Neurons with a p-value below 0.05 were considered significant.

4.3.3 Finding oscillatory circuits

2 Hz power

To identify the oscillating neurons, I calculated the power of 2 Hz oscillation of their neural activities. This was inspired by the fact that the desired outputs were a 2 Hz sinusoidal wave. The procedure to compute the 2 Hz power is as follows:

1. Get the time window where the input is non-zero.
2. Fourier transform the time series of the output during the obtained time window.
3. Get the power of 2 Hz from the frequency domain.

Dynamical stability analysis

To investigate the coupled dynamics of the pair of one excitatory neuron (E) and one inhibitory neuron (I), I conducted the stability analysis of that system [9] [15]. The equations that governs the dynamics of each neuron are:

$$\tau_E \dot{E}(t) = -E(t) + E^{rest} + w_{I\to E} f(I(t)) + S_E \tag{4.9}$$
$$\tau_I \dot{I}(t) = -I(t) + I^{rest} + w_{E\to I} f(E(t)) + S_I \tag{4.10}$$

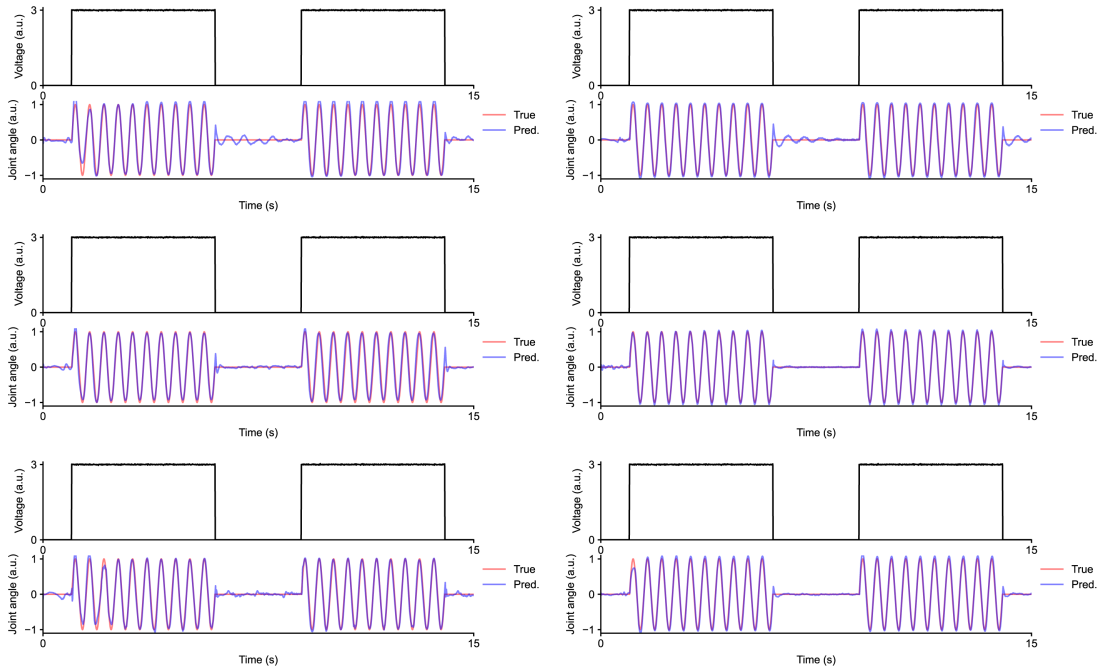
$E(t)$: the potential of the E neuron
$I(t)$: the potential of the I neuron
$w_{I\to E}$: the weight of the connection from I to E neuron
$w_{E\to I}$: the weight of the connection from E to I neuron
S_E : the input to the E neuron from all the other sources
S_I : the input to the I neuron from all the other sources

Note that these equations are adapted from above (see **Neuronal dynamics and trainable parameters**). To analyze the stability of these two neuron system, I followed these steps:

1. Get approximate S_E and S_I by computing time averaged inputs to each neuron during the time when there are the non-zero inputs to the system.
2. Draw a phase portrait of the E and I dynamics to check the system's stability.

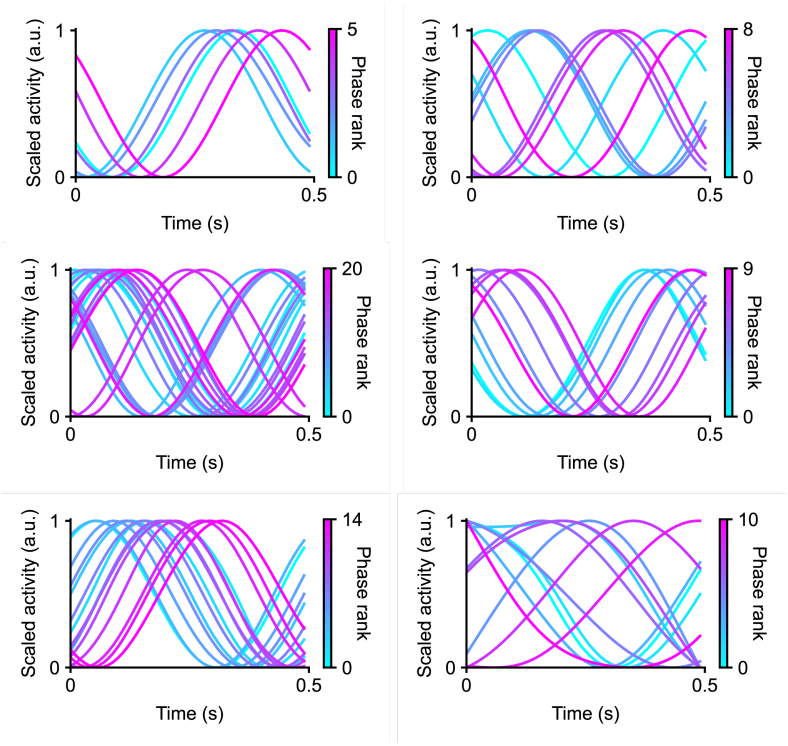
Appendix A

Extended Figures



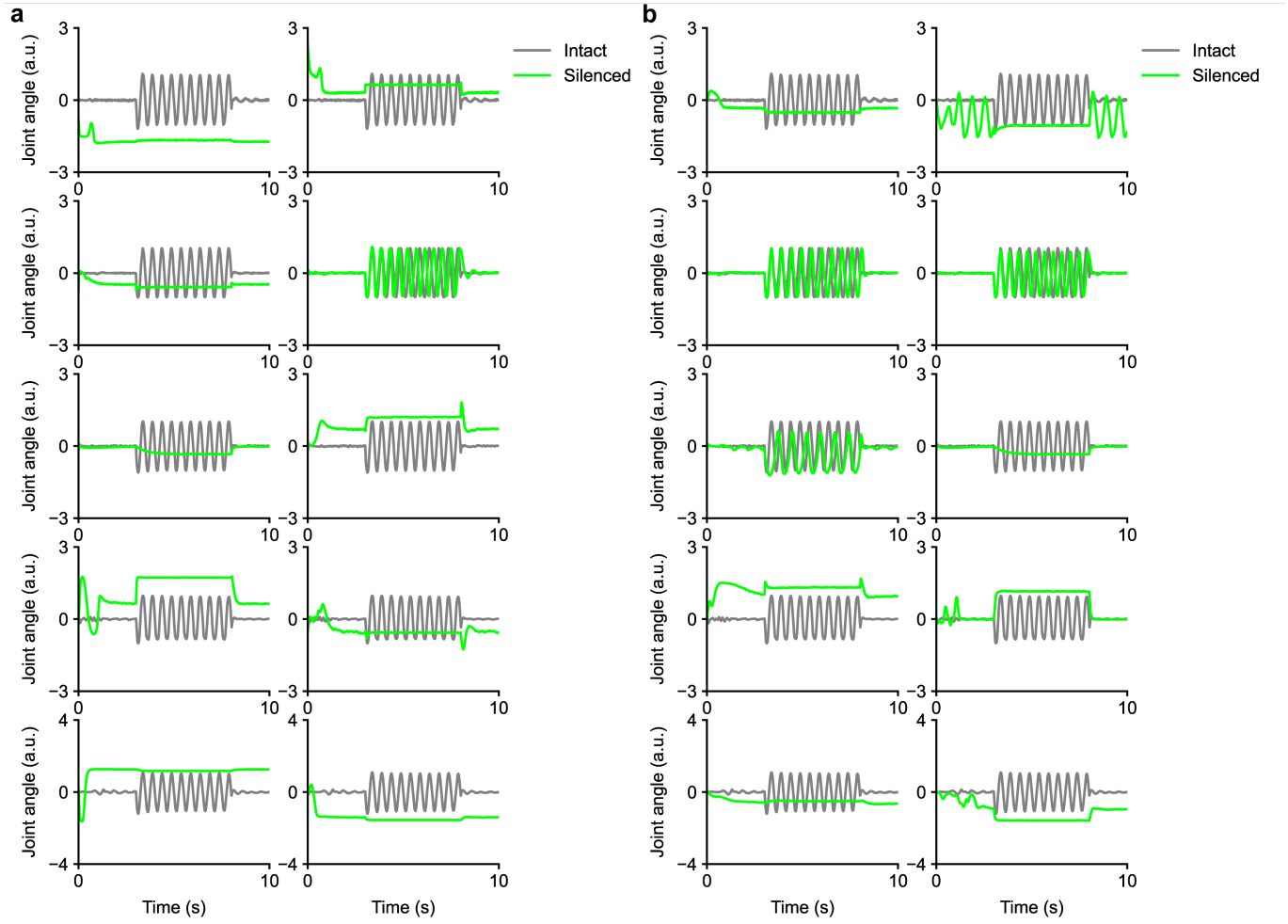
Extended Figure 1. Test inputs and outputs of the trained networks

The black line is the input to the MDNs, the red line is the desired output from the network, and the blue line is the output from the network at each panel. The test results for all the networks other than the one at Fig.2c are plotted here.



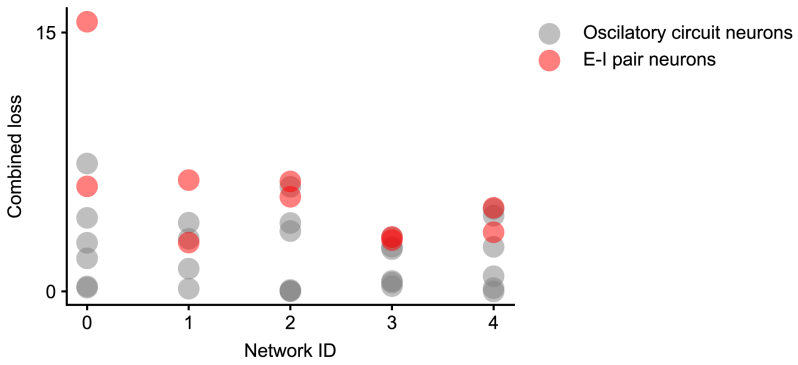
Extended Figure 2. Phase relationships between oscillating neurons across 6 trained networks.

Similar to 4b, but showing oscillating neurons from networks not included there.



Extended Figure 3. Outputs of FmrTi joint angles from intact and silenced networks across 5 trained networks

a, Similar to 5a, but showing the FmrTi joint angles. **b**, Similar to 5b, but showing the FmrTi joint angles.



Extended Figure 4. Silencing test result of oscillatory circuits 5 trained networks

Each dot represents the combined loss from the silencing test of a neuron. In each trial, a neuron within the oscillatory circuit was silenced. Neurons belonging to E-I pairs are marked in red, while others are marked in gray.

Appendix B

Supplementary Analysis

B.1 Steady-state output

I observed from the neural activities of all neurons that the trained networks could be divided into two groups: those with interneurons that oscillate without MDN input and those with interneurons that oscillate only with MDN input. If these two distinct behaviors of the interneurons influence the network's dynamics, they should be reflected in the decoder outputs. To investigate this, I provided no input to the trained networks for 15 seconds and recorded their steady-state outputs. Despite the absence of input, the networks produced non-zero outputs due to intrinsic noise. Upon the onset of simulations, these outputs stabilized within a short time (Supplementary Fig.1a,b).

From the stabilized outputs, I observed that some networks exhibited strong oscillatory components, while others did not. To quantify this, I computed the autocorrelation of each CxTr joint output (Supplementary Fig.1c) over lags from 0 to 5 seconds. The results clearly demonstrated two distinct network behaviors: networks that oscillate without input and networks that oscillate only when input is present.

B.2 Eigendecomposition of the weight matrix[1]

To understand how the network dynamics evolve over time, the network can be viewed as a function that takes an input $\mathbf{r_t}$ and produces an output $\mathbf{r_{t+1}}$ iteratively. Here, $\mathbf{r}$ represents the vector of all neurons' activities. The relationship between the input and output can then be expressed as:

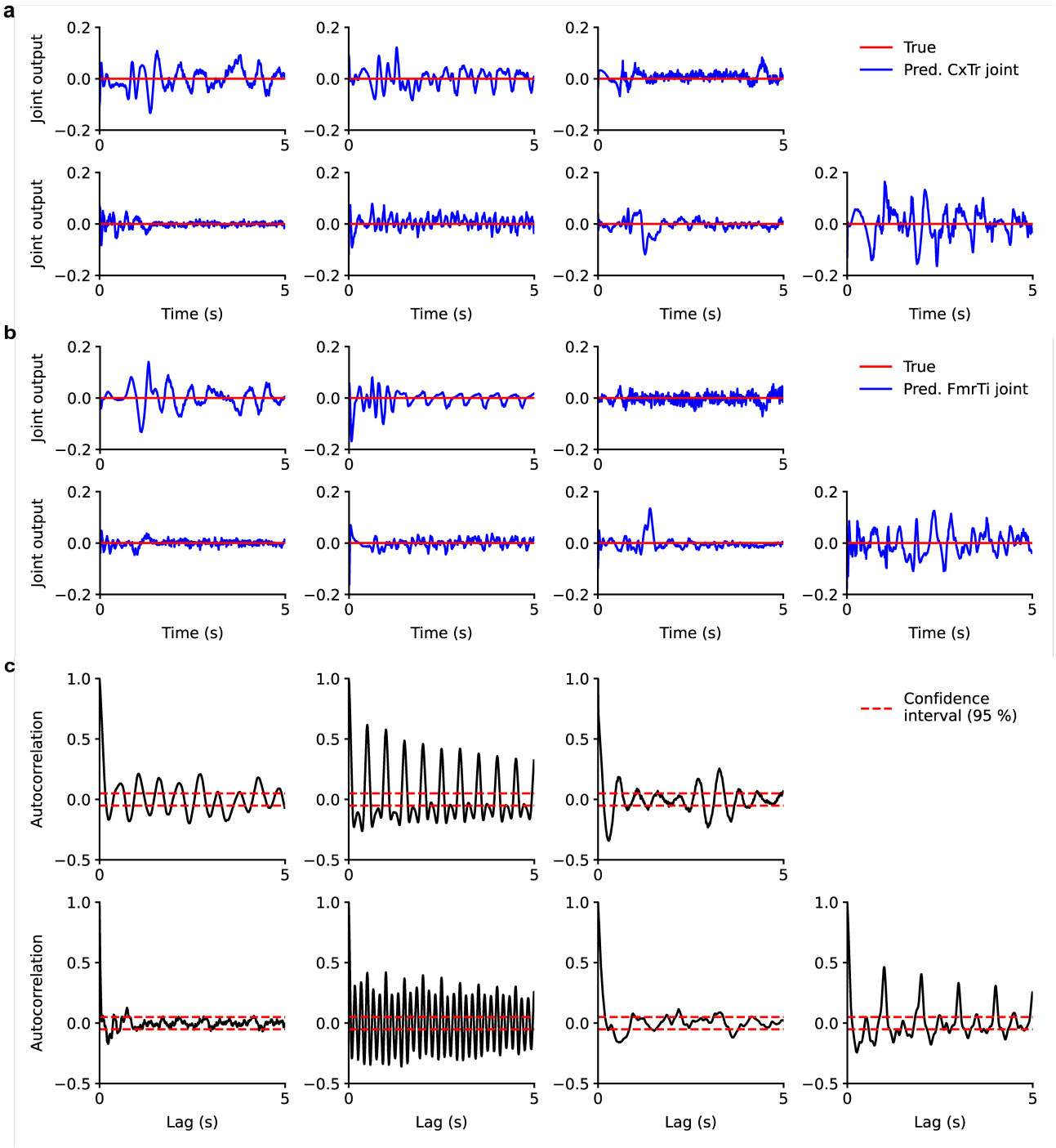
$$\mathbf{r_{t+1}} = \mathbf{W} \cdot \mathbf{r_t} \tag{B.1}$$

, where $\mathbf{W}$ represents the weight matrix of the network. The eigenvectors of $\mathbf{W}$ with significantly large eigenvalues indicate traces of neural activity that strongly influence the network dynamics. From the entries with significant loadings in an eigenvector, we can identify groups of neurons, termed "eigencircuits," as each entry corresponds to a neuron within the network. If these eigencircuits are activated simultaneously at an adequate level, they can drive the network dynamics in the direction of their corresponding eigenvector.

Therefore, if an eigenvector corresponds to an eigencircuit composed of neurons from the oscillatory circuits, studying this eigencircuit could provide insights into the circuit mechanisms underlying oscillation generation. However, due to time constraints, I only demonstrated there are a few eigenvectors with eigenciruits comprised of multiple neurons. Considering the degenerate eigenvalues, there are approximately 60 unique eigenvalues and corresponding eigenvectors (Supplementary Fig.2a). Among these, only a few eigenvectors involve more than two neurons with significant loadings (Supplementary Fig.2b). Interestingly, eigenvectors with multiple neurons showing significant loadings have sparse distributions, resulting in eigencircuits of interpretable sizes.

B.3 Training results of 2 hops MDN-T1 network

Here, I constructed and trained 2 hops MDN-T1 networks with different initializations on a simple training dataset. The results showed that the 2 hops MDN-T1 network generally failed to learn the desired oscillation, and when it did, the training required significantly more time (over 15,000 epochs) (Supplementary Fig.3a). I then trained three networks where the loss dropped significantly on the complex training dataset. Among these, the network with the highest loss on the simple dataset failed to learn the complex dataset, whereas the others succeeded (Supplementary Fig.3b). These results suggest that the 2 hops MDN-T1 network is not inherently designed to generate oscillations upon MDN stimulation. However, further investigation is needed to confirm this.



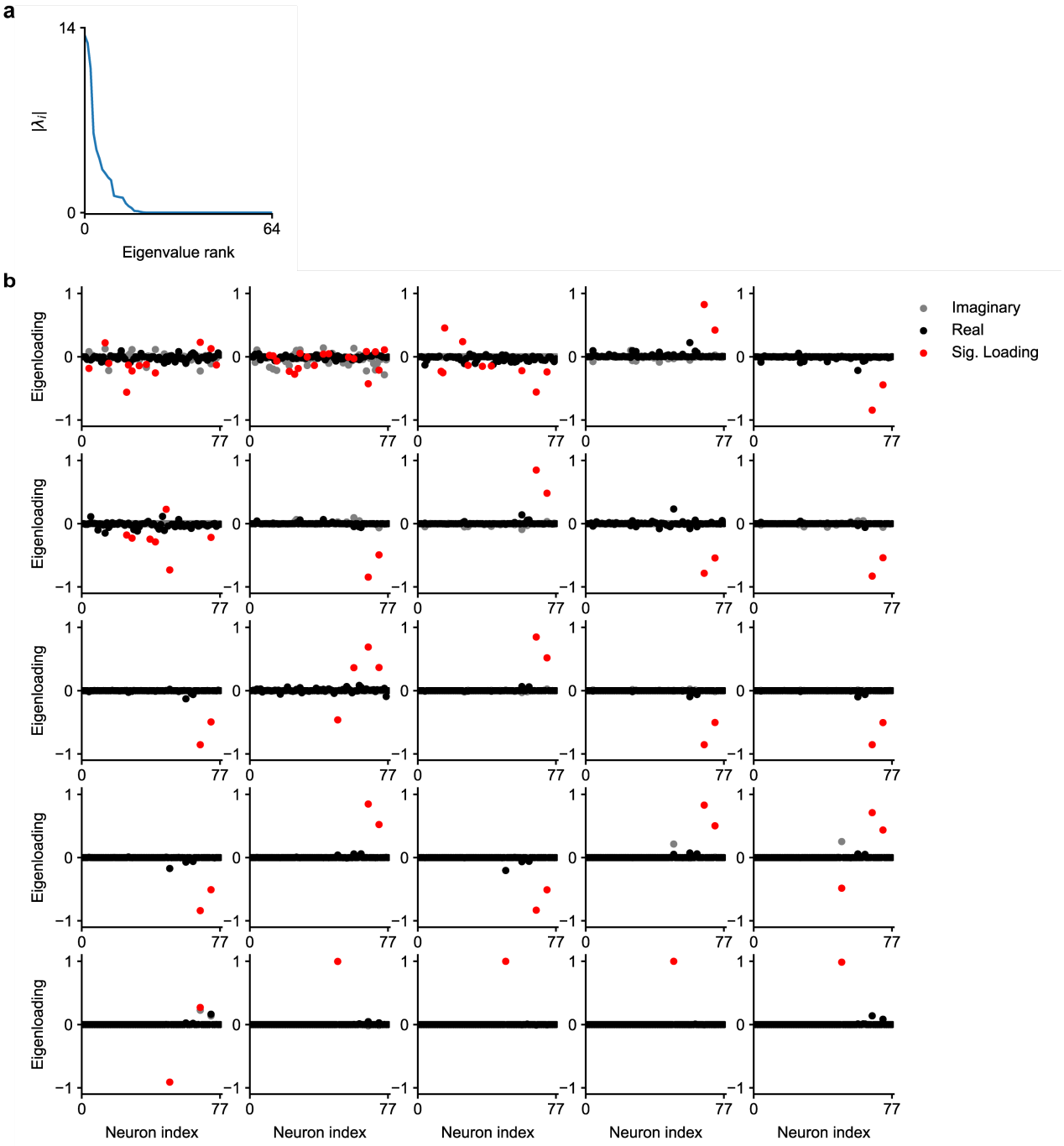
Supplementary Figure 1. Steady-state outputs of the networks and their autocorrelation

a, Steady-state outputs from the CxTr joint decoders of the 7 trained networks, plotted for the time range of 0 to 5 seconds. **b**, Similar to **a**, but showing steady-state outputs from the FmrTi joint decoders. **c**, Autocorrelation of the steady-state outputs from the CxTr joints with lags ranging from 0 to 5 seconds. The 95% confidence interval is indicated on each graph.

B.4 Training results of 3 hops networks

Here, I constructed and trained 3 hops MDN-T3 and MDN-T1 networks with different initializations on a distinct training dataset. This dataset consisted of tonic pulse inputs with durations of 1, 3, or 5 seconds, all at the same amplitude. The onsets of the inputs were randomized between 0 and 10 seconds, and the outputs were 2 Hz oscillations during the input period. The dataset contained a total of 54 samples, each sample lasted for 10 seconds. In all cases, the networks successfully learned the task within 6,000 epochs (Supplementary Fig.4).

Given the sizes of these networks (T3: 613 neurons with 23,000 synapses; T1: 405 neurons with 11,364 synapses), it is expected that they could perform the given task. Furthermore, since the T1 neuropil is also involved in generating cyclic foreleg movements, it would be intriguing to investigate whether the neurons driving oscillation



Supplementary Figure 2. Eigenspectrum and eigenvectors of an example weight matrix

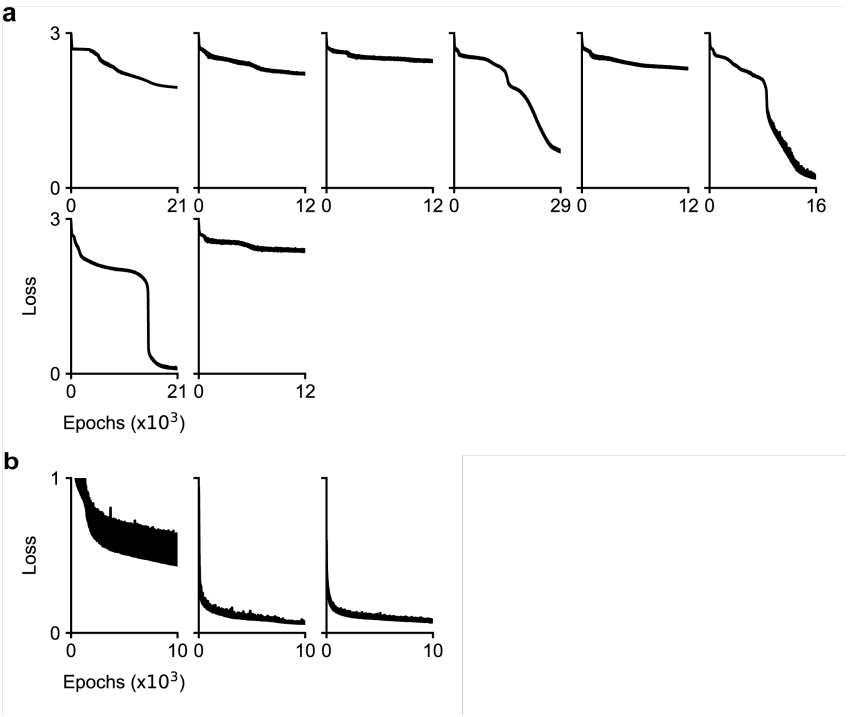
a, Eigenvalue spectrum of an example weight matrix. Eigenvalues are ranked based on their absolute values. **b**, Eigenvectors of an example weight matrix with non-zero eigenvalues. Significant loadings are identified as follows: neurons are ranked by their absolute values, and loadings are considered significant (marked with red dots) if their cumulative sum, starting from the highest loading, explains more than 85%.

generation for this task overlap with those involved in other cyclic foreleg movements, such as antennal grooming.

B.5 Training results of full networks

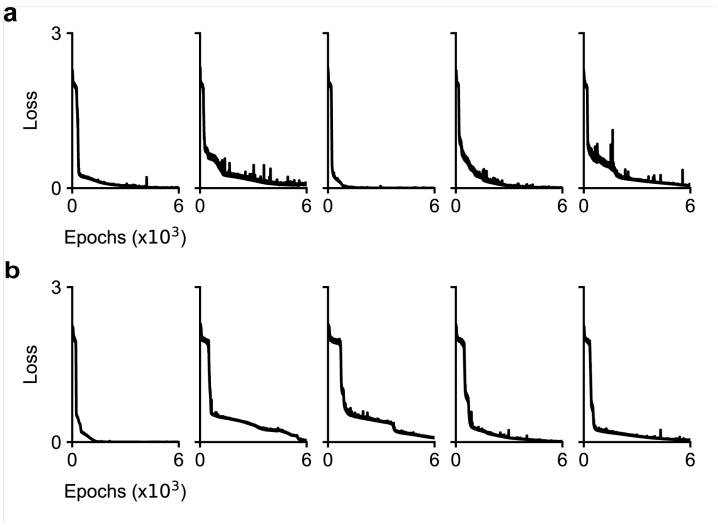
Training on the toy dataset

Here, I constructed the full MDN-T3 and MDN-T1 networks and trained them on a simpler toy dataset. The purpose of this experiment was to evaluate whether the networks could learn oscillations from tonic inputs. The training dataset consisted of 2 seconds of input sequences—a 1 second tonic pulse padded by 0.5 second zero periods at the beginning and end—and 2 seconds of output sequences represented as sinusoidal waves. The frequencies of the sinusoidal waves were proportional to the input amplitudes, and the dataset contained a total of 320 samples.



Supplementary Figure 3. Training losses of the 2 hops MDN-T1 network

a, Training losses of 8 different networks trained on the simple training dataset. **b**, Training losses of 3 different network trained on the complex training dataset. The 4th, 6th and 7th networks from **a** are used.



Supplementary Figure 4. Training losses of the 3 hops network

a, Training losses of the 5 different 3 hops MDN-T3 networks. **b**, Training losses of the 5 different 3 hops MDN-T1 networks.

Moreover, to reduce the number of free parameters during training, I clustered the neurons using the DBSCAN algorithm based on their pre- and postsynaptic connectivity patterns.

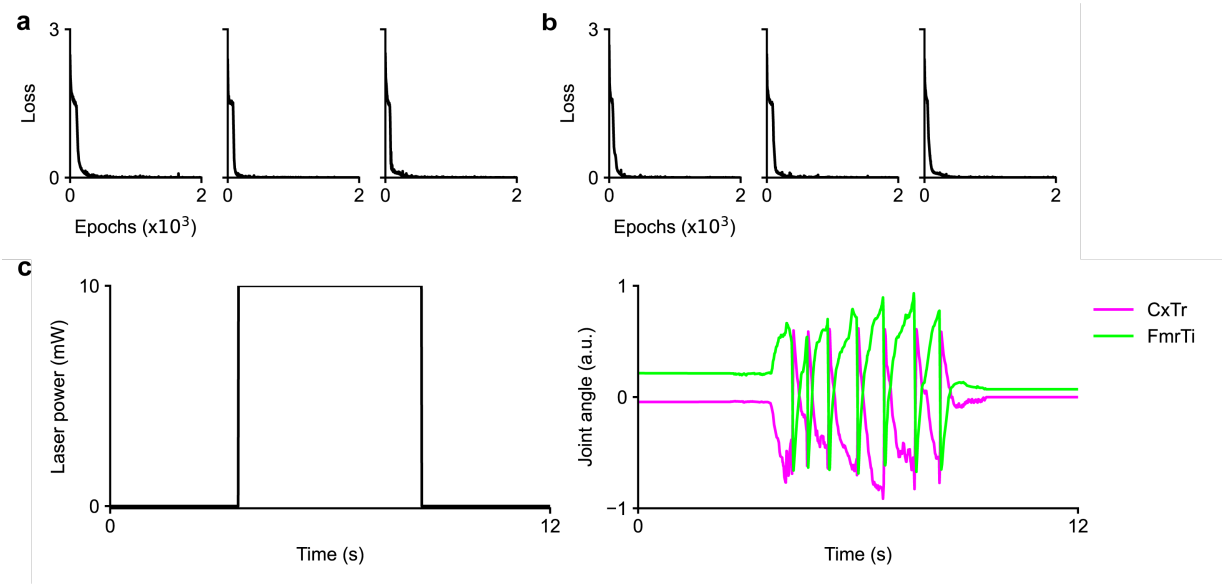
Both networks successfully learned the dataset within 2,000 epochs (Supplementary Fig.5a,b). These results are expected given the complexity of the networks (T3: 1,892 neurons with 50,954 synapses; T1: 2,072 neurons with 58,574 synapses).

Training on the experimental dataset

I also attempted to train the same networks using an experimental dataset. In the experiment, we optogenetically stimulated the MDNs at varying input levels and processed the resulting joint movements to extract the corresponding oscillations (Methods). The experimental dataset consisted of six different tonic pulse input levels and their corresponding joint oscillations (Supplementary Fig.5c). However, both networks failed to learn this dataset (data

not shown).

This failure is likely due to the limitations of the simple linear decoders, which were insufficient to generate the desired joint movements. Leg joint movements are not simple affine transformations of motor neuron activity; instead, neural activity undergoes nonlinear transformations through muscles and mechanical couplings between leg segments. Therefore, to successfully train the networks with the desired dataset, more realistic decoders need to be constructed.



Supplementary Figure 5. Training losses of full networks and an example experimental dataset

a, Training losses of the 3 different full MDN-T3 networks. **b**, Training losses of the 3 different full MDN-T1 networks. **c**, One sample from the experimental dataset. The optogenetic input to MDNs is shown on the left, while the CxTr and FmrTi joint movements are plotted in different colors on the right.

References

[1] Dean A. Pospisil, Max J. Aragon, Sven Dorkenwald, Arie Matsliah, Amy R. Sterling, Philipp Schlegel, Szi-chieh Yu, Claire E. McKellar, Marta Costa, Katharina Eichler, Gregory S. X. E. Jefferis, Mala Murthy, and Jonathan W. Pillow. The fly connectome reveals a path to the effectome. *Nature*, 634(8032):201–209, October 2024.

[2] Jonas Braun, Femke Hurtak, Sibo Wang-Chen, and Pavan Ramdya. Descending networks transform command signals into population motor control. *Nature*, 630(8017):686–694, June 2024.

[3] Salil S. Bidaye, Christian Machacek, Yang Wu, and Barry J. Dickson. Neuronal Control of Drosophila Walking Direction. *Science*, 344(6179):97–101, April 2014.

[4] Kai Feng, Rajyashree Sen, Ryo Minegishi, Michael Dübbert, Till Bockemühl, Ansgar Büschges, and Barry J. Dickson. Distributed control of motor circuits for backward walking in Drosophila. *Nature Communications*, 11(1):6166, December 2020.

[5] Shin-ya Takemura, Kenneth J. Hayworth, Gary B. Huang, Michal Januszewski, Zhiyuan Lu, Elizabeth C. Marin, Stephan Preibisch, C. Shan Xu, John Bogovic, Andrew S. Champion, Han SJ Cheong, Marta Costa, Katharina Eichler, William Katz, Christopher Knecht, Feng Li, Billy J. Morris, Christopher Ordish, Patricia K. Rivlin, Philipp Schlegel, Kazunori Shinomiya, Tomke Stürner, Ting Zhao, Griffin Badalamente, Dennis Bailey, Paul Brooks, Brandon S. Canino, Jody Clements, Michael Cook, Octave Duclos, Christopher R. Dunne, Kelli Fairbanks, Siqi Fang, Samantha Finley-May, Audrey Francis, Reed George, Marina Gkantia, Kyle Harrington, Gary Patrick Hopkins, Joseph Hsu, Philip M. Hubbard, Alexandre Javier, Dagmar Kainmueller, Wyatt Korff, Julie Kovalyak, Dominik Krzemiński, Shirley A. Lauchie, Alanna Lohff, Charli Maldonado, Emily A. Manley, Caroline Mooney, Erika Neace, Matthew Nichols, Omotara Ogundeyi, Nneoma Okeoma, Tyler Paterson, Elliott Phillips, Emily M. Phillips, Caitlin Ribeiro, Sean M. Ryan, Jon Thomson Rymer, Anne K. Scott, Ashley L. Scott, David Shepherd, Aya Shinomiya, Claire Smith, Natalie Smith, Alia Suleiman, Satoko Takemura, Iris Talebi, Imaan FM Tamimi, Eric T. Trautman, Lowell Umayam, John J. Walsh, Tansy Yang, Gerald M. Rubin, Louis K. Scheffer, Jan Funke, Stephan Saalfeld, Harald F. Hess, Stephen M. Plaza, Gwyneth M. Card, Gregory SXE Jefferis, and Stuart Berg. A Connectome of the Male Drosophila Ventral Nerve Cord. *eLife*, 13, May 2024.

[6] Anthony Azevedo, Ellen Lesser, Jasper S. Phelps, Brandon Mark, Leila Elabbady, Sumiya Kuroda, Anne Sustar, Anthony Moussa, Avinash Khandelwal, Chris J. Dallmann, Sweta Agrawal, Su-Yee J. Lee, Brandon Pratt, Andrew Cook, Kyobi Skutt-Kakaria, Stephan Gerhard, Ran Lu, Nico Kemnitz, Kisuk Lee, Akhilesh Halageri, Manuel Castro, Dodam Ih, Jay Gager, Marwan Tammam, Sven Dorkenwald, Forrest Collman, Casey Schneider-Mizell, Derrick Brittain, Chris S. Jordan, Michael Dickinson, Alexandra Pacureanu, H. Sebastian Seung, Thomas Macrina, Wei-Chung Allen Lee, and John C. Tuthill. Connectomic reconstruction of a female Drosophila ventral nerve cord. *Nature*, 631(8020):360–368, July 2024.

[7] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proceedings of the 26th Annual International Conference on Machine Learning*, ICML '09, pages 41–48, New York, NY, USA, June 2009. Association for Computing Machinery.

[8] Eve Marder and Dirk Bucher. Central pattern generators and the control of rhythmic movements. *Current Biology*, 11(23):R986–R996, November 2001.

[9] Peter Dayan and Laurence F. Abbott. *Theoretical Neuroscience: Computational and Mathematical Modeling of Neural Systems*. Computational Neuroscience. MIT Press, Cambridge, Mass., 2001.

[10] Kyobi S. Kakaria and Benjamin L. de Bivort. Ring Attractor Dynamics Emerge from a Spiking Model of the Entire Protocerebral Bridge. *Frontiers in Behavioral Neuroscience*, 11, February 2017.

[11] Janne K. Lappalainen, Fabian D. Tschopp, Sridhama Prakhya, Mason McGill, Aljoscha Nern, Kazunori Shinomiya, Shin-ya Takemura, Eyal Gruntman, Jakob H. Macke, and Srinivas C. Turaga. Connectome-constrained networks predict neural activity across the fly visual system. *Nature*, 634(8036):1132–1140, October 2024.

[12] Semih Günel, Helge Rhodin, Daniel Morales, João Campagnolo, Pavan Ramdya, and Pascal Fua. DeepFly3D, a deep learning-based approach for 3D limb and appendage tracking in tethered, adult Drosophila. *eLife*, 8:e48571, October 2019.

[13] Paul Werbos. Backpropagation through time: What it does and how to do it. *Proceedings of the IEEE*, 78:1550–1560, November 1990.

[14] Sashank J. Reddi, Satyen Kale, and Sanjiv Kumar. On the Convergence of Adam and Beyond, April 2019.

[15] Eugene M. Izhikevich. *Dynamical Systems in Neuroscience: The Geometry of Excitability and Bursting*. The MIT Press, July 2006.