Single neuron ablations in the C. elegans connectome using OpenWorm

Caleb Fedyshen 30087986

CPSC 599 Connectomics Fall 2025

Abstract / Summary

The nematode Caenorhabditis elegans has one of the most complete and reproducible documented connectomes out there, comprising 302 neurons and thousands of chemical synapses and gap junctions.[1] This compact, well characterised connectome has enabled theoretical work on network control [2] and experimental studies of locomotion following targeted neuron ablations [4]. In this project I use the OpenWorm digital worm model [6,7,8] to study how single neuron ablations in the c302 connectome affect simulated locomotion. Starting from the canonical connectome (with 279 non-pharyngeal neurons), I programmatically generate one modified network per neuron by deleting all of that neuron's outgoing chemical synapses while leaving gap junctions intact. For each modified connectome, and for an unmodified baseline, I run a short 12 ms OpenWorm simulation in identical configurations and extract midline positions from the output time series of worm positions. From these lines I compute simple locomotion metrics, lateral range of the midline, and forward velocity, and compare each ablation to the baseline. Network statistics of the original connectome, including degree distribution, clustering coefficient, and path length, are compared to Erdős–Rényi and degree preserving null models as well, to give a fuller picture. I outline how the full set of ≈280 ablations can be visualised as a "sensitivity map" over the connectome and discuss these results in light of network control based predictions of locomotor control neurons [2] and synthetic ablations [6]. Comparisons are made between the set of neurons discovered in this experiment and the preceding literature.

Research Questions

RQ1. How sensitive is simulated C. elegans locomotion, in the OpenWorm c302 model, to single neuron ablations that remove all outgoing chemical synapses from individual neurons?

RQ2. Do ablations of neurons previously predicted to be critical for muscle control by network control theory, such as AVAL/R, DA07–09, DB05–07, DD04–06, VA12, VB11, VD12–13, and PDB [2] produce larger changes in locomotion metrics than ablations of other neurons?

RQ3. Are simple metrics derived from midline coordinates (lateral range and forward velocity) sufficient to detect the impact of such ablations in these short simulations?

Introduction

The hermaphrodite Caenorhabditis elegans has 302 neurons whose synaptic wiring diagram was painstakingly reconstructed from microscope images. This connectome provided the first complete map of an animal nervous system. Since the pharyngeal neurons are largely separate from the main mass of neurons and are believed to be unrelated to the movement abilities of the worm, many studies focus on the 279 non-pharyngeal neurons connected by ≈5000 chemical synapses and hundreds of gap junctions [1]. The availability of this detailed wiring diagram has made C. elegans very important for connectomics, network neuroscience, and recently, computational modelling.

On the network side, Yan et al. [2] applied network control theory to this connectome, predicting which neuronal classes must remain intact to control body wall muscles. Their framework identified a set of 20 individual neurons whose ablation should strongly affect muscle control, including AVAL/R, AS08–11, DA07–09, DB05–07, DD04–06, VA12, VB11, VD12–13, and the previously uncharacterised interneuron PDB [2]. These predictions were validated using targeted laser ablations. Building on this, Towlson and Barabási [4] explored "synthetic ablations": pairs and triplets of neurons whose simultaneous removal is predicted to severely compromise muscle controllability, even when single ablations are relatively benign.

The OpenWorm project aims to integrate these structural and behavioural insights into an in silico model of C. elegans, combining the c302 neural simulator with the Sibernetic body and fluid dynamics engine [7,8]. In principle, this neuromechanical model allows one to test hypotheses about how changes in the connectome influence movement, by modifying the network and measuring changes in simulated locomotion.

In this project I use this to explore how single neuron ablations affect motion in the OpenWorm model. To do so I construct a pipeline that programmatically edits the c302 connectome, runs the OpenWorm simulation for each modified network, and extracts simple locomotion metrics from the resulting worm position data. I focus on three questions: the overall sensitiveness to single neuron ablations of the connectome to the motion of the worm (RQ1), differences between ablations of neurons predicted to strongly affect muscle control by Yan et al. [2] and other neurons (RQ2), and the adequacy of simple scalar metrics compared to complex ones like eigenworms (RQ3).

Dataset Description

Connectome data

The primary dataset is the adult hermaphrodite C. elegans connectome that comes packaged with the OpenWorm c302 framework [7]. In the OpenWorm Docker image this appears as a CSV file (herm_full_edgelist_MODIFIED.csv) with four columns: Source, Target, Weight, and Type (Chemical or electrical). Each row corresponds to a directed chemical synapse or an undirected gap junction. For much of the analysis I focus on the 279 non-pharyngeal neurons standard in the literature [1], excluding the ~20 pharyngeal neurons that don't strongly connect to the main part of the worm, and are safely ignored in other literature.

Basic Statistics

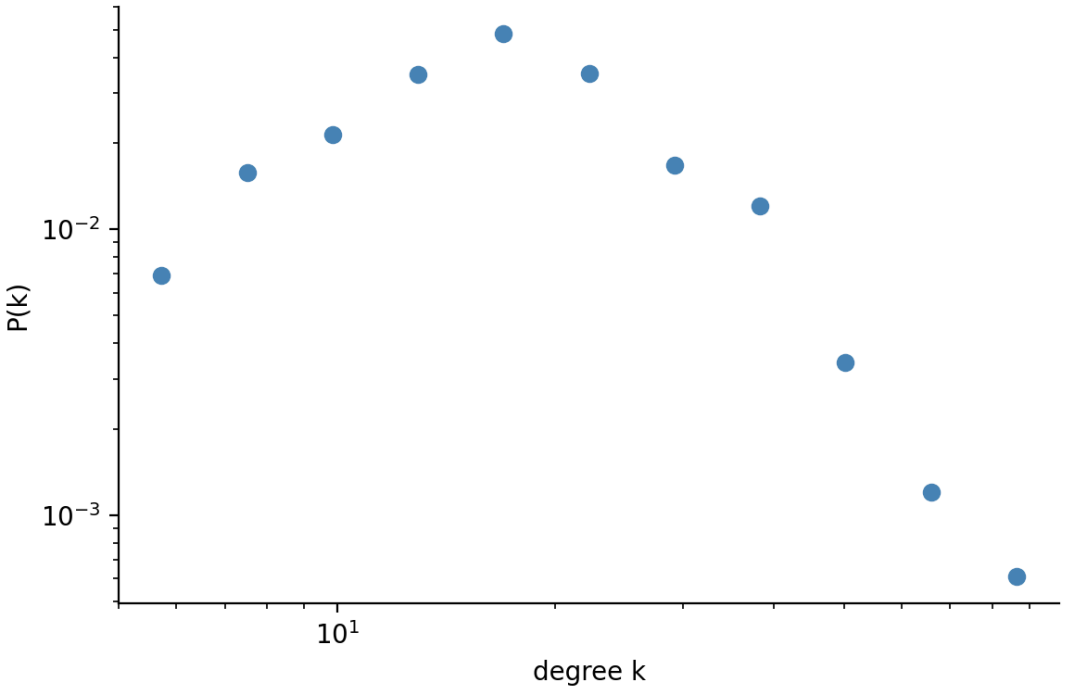
To situate the OpenWorm connectome within the broader context of network neuroscience, I computed a few graph statistics for the network and compared them against simple null models. These charts should look familiar, because I reused much of the code from assignment one to create them, substituting in the network found inside openworm instead. This should be enough to get a general feel of the dataset.

I computed the degree, weight, and strength distributions. The degree distribution has some mild turbulence at the beginning, moving high and dropping, then peaking shortly after. This indicates there are slightly less nodes with a very low degree, but the majority of the nodes are of middling degree, and there are some, but not a considerable amount of nodes on the higher degree end of the distribution. This distribution does not seem particularly surprising to me, given the nature of the network in real life, where it would be strange if it was extremely dense, given the energy cost of establishing connections and the fact that there is only so much space to connect to each neuron. The higher amount of low degree nodes makes sense in context as well. Think perhaps of motor neurons, and the routes to them. These would likely be linear with each node connecting to just a few, and terminating in a leaf node. Still yet, some of the strange shape of the beginning of the distribution can be explained by the fact that this is a rather small network, and there are simply not enough nodes to smooth out some irregularities.

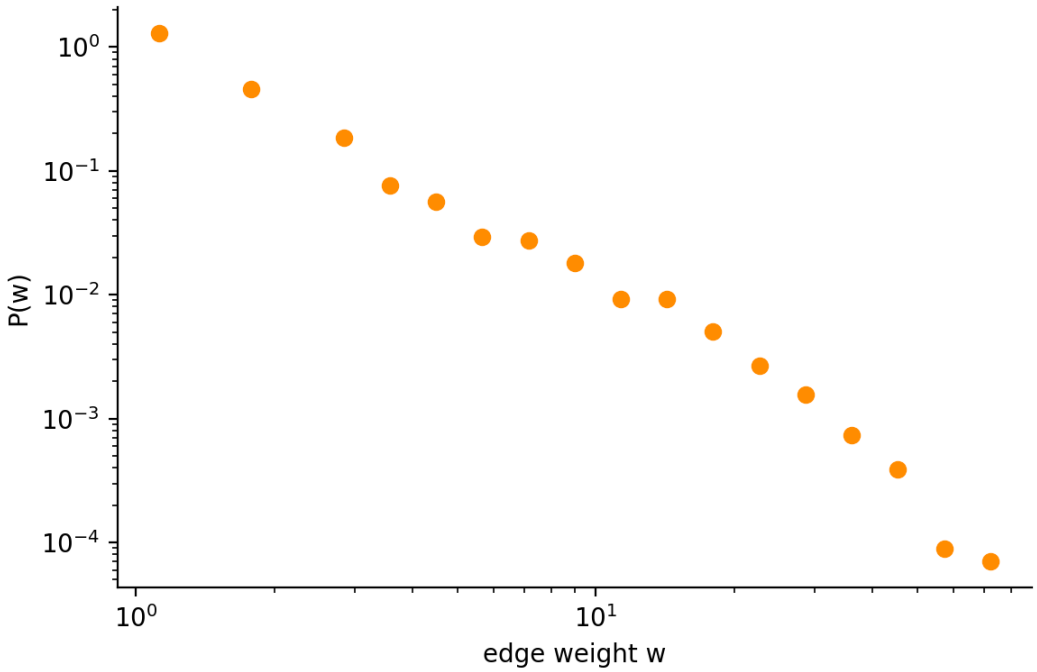
The weight distribution looks nearly linear, suggesting a heavy tailed distribution. This makes sense as well, with most connections being rather weak and a few being quite strong. In a real biological network, it seems that it would be easy (metabolically inexpensive) to have weak connections and take more effort to forge stronger ones, so there would be fewer stronger weights and many more weaker ones.

My initial reaction to the strength distribution is that it looks like a combination of the previous two, which makes sense because it relies on both the degree and the weights of each edge. The rightmost point seems to indicate that there is a small set of nodes that are both high degree and each edge has a high weight. This could be an indication of a small number of strong hubs in the network, which also makes sense given the context, and especially given the rich club analysis given in class, where we did find several nodes that were well connected, but not with eachother, instead their connections were dispersed about the network. The resulting degree distribution is distinctly heavy tailed, a small number of neurons serve as high-degree hubs.

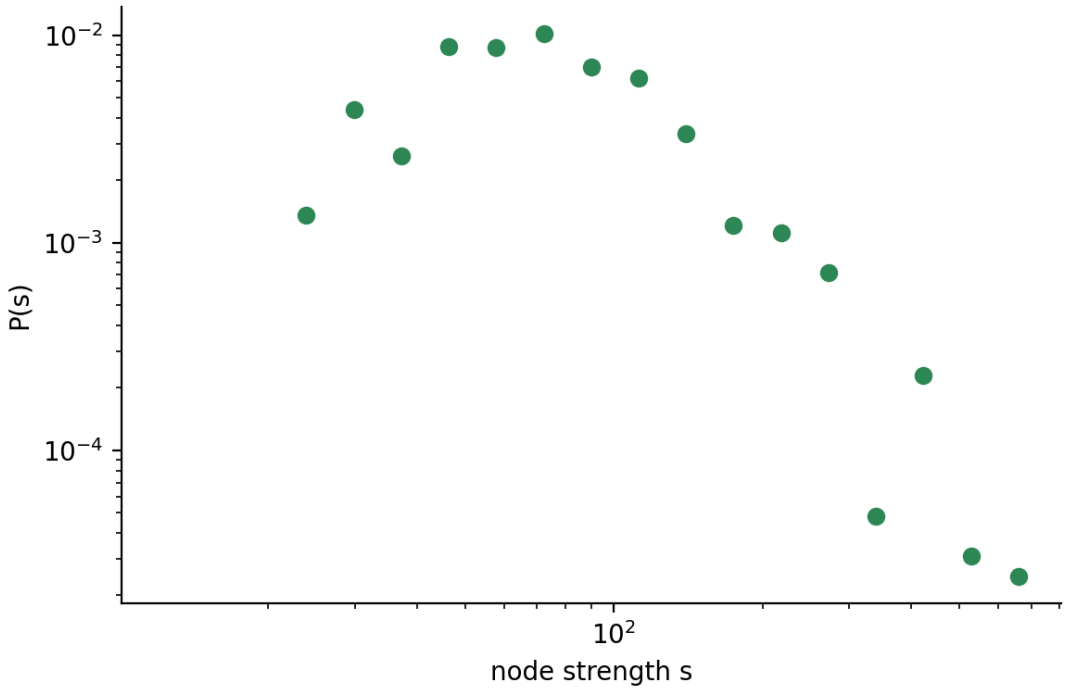
[Figure 1: Degree distribution]



[Figure 1b: Edge weight distribution]



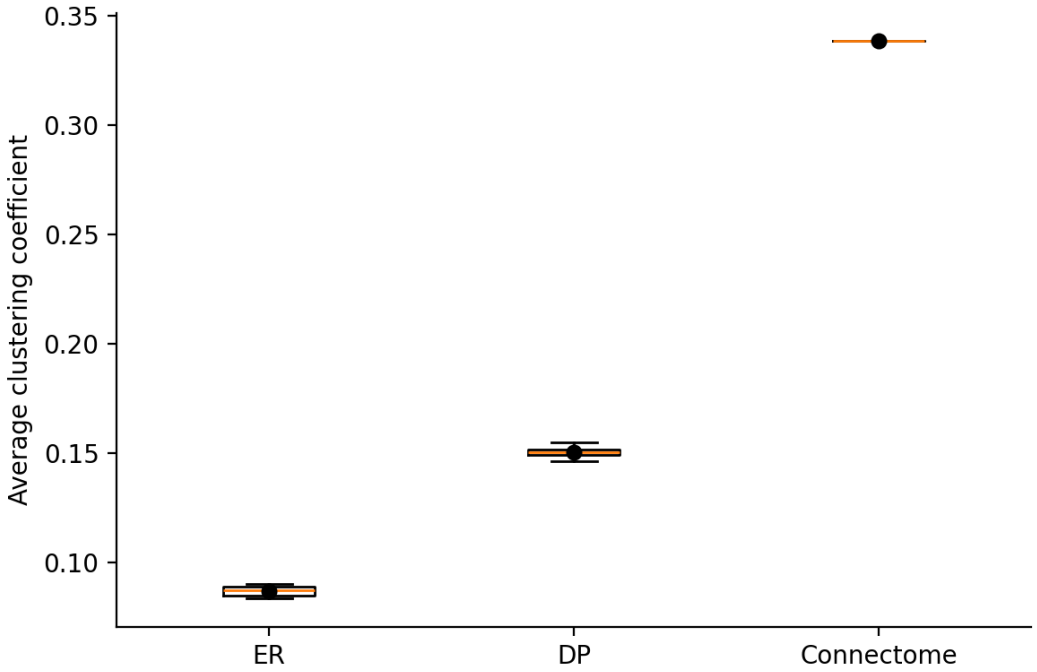
[Figure 1c: Strength distribution]



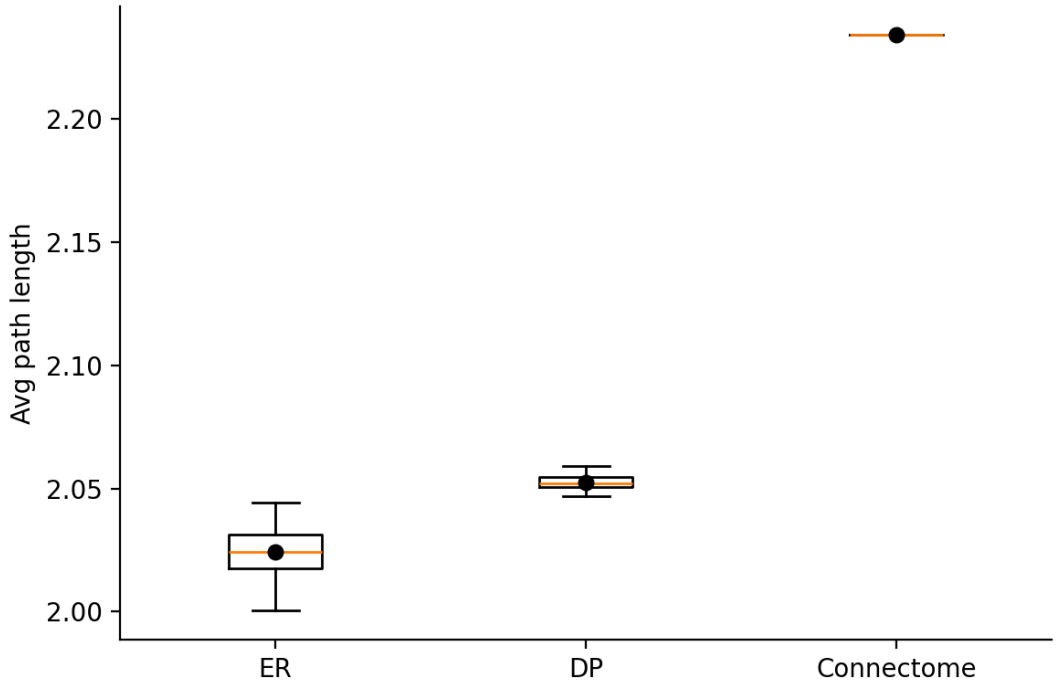
To further explore, I calculated the Average clustering coefficient, and the average path length vs two ensembles of null models. The first ensemble was of Erdos-Reyni (ER) graphs with the same number of nodes and edges, providing a fully random baseline. The second consisted of degree preserving (DP) random graphs generated via repeated double-edge swaps.

Relative to these nulls, the C. elegans connectome displays significantly higher clustering than the ER ensemble and moderately higher clustering than the degree-preserving one, while also having slightly longer path lengths. These patterns show that the network is anything but random and in fact has some underlying structure, and well, it had better because it's the brain of a worm!

[Figure 2: Average clustering coefficient vs ER and DP nulls]



[Figure 3: Average path Length vs ER and DP]



Methodology

Simulation environment: OpenWorm c302 + Sibernetic

All simulations are run using the OpenWorm Docker image, which bundles c302, a framework for generating NeuroML2 network models of the C. elegans nervous system at multiple levels of biophysical detail [7]; and Sibernetic, a physics sim modelling the worm body and its fluid environment [8].

The main script, single_ablation.py runs the ablation experiments end to end. It first loads the unmodified connectome and runs openworm_evaluator.py once to get a baseline simulation. This baseline gives a reference crawling trajectory and lets us measure things like typical lateral range and average forward speed when no neurons are ablated.

After the baseline is done, the script loops over a chosen set of target neurons. For each neuron, it creates a new connectome CSV, where that neuron's outgoing chemical synapses are set to zero. It then calls openworm_evaluator.py again, this time pointing --connectome-file at the mutated CSV and using a run tag that includes the neuron's name. The script locates the simulation output directory for that run and finds the corresponding worm_motion_log.txt, which is later used to extract motion metrics for that ablation.

The driver script openworm_evaluator.py wraps Docker to optionally mount a modified connectome file into both c302 data directories, run Sibernetic for a specified duration with fixed time step parameters, and save the outputs.

Because each OpenWorm run is computationally expensive, all experiments in this project use a short simulation duration of 12 ms, with the same values of the neural and physics time steps and a forward crawling configuration as in the default OpenWorm examples. Note that I did run some longer simulations and the magnitude of the results were larger, but it was not feasible to run say 200ms simulations for each ablation. Each of the 280 simulations starts from the same initial state and uses identical c302 and Sibernetic parameters, so the runs are basically deterministic, with repeated runs of the same simulation giving identical results at least up to the precision at which I measured. Under these conditions, the main goal is to detect whether a given ablation produces *any* reproducible deviation of the locomotion metrics from the baseline, rather than to cause and measure large changes in the generalized motion of the worm over time.

Connectome mutation and neuron ablation

The file connectome_brain_mutation.py is a small library that loads the C. elegans connectome and lets you change it by weakening or removing the chemical synapses of specific neurons.

It starts by reading the main CSV file, herm_full_edgelist_MODIFIED.csv. The helper function connectome_rows(base_path) loads every row of the connectome into memory and makes sure the file has the correct columns ("Source," "Target," "Weight," and "Type"). This gives us a clean list of all known connections.

The actual mutation happens in write_mutated_connectome(rows, output_path, scale_by_neuron, brain_targets=None). This function writes a new copy of the connectome where each neuron's outgoing chemical synapses are scaled by a factor, this is because I reuse the code I originally wrote for the genetic algorithm experiment I was planning on, where each of the edges would get a small randomized scale factor to try and mutate the connectome in a productive way. In our case, if a neuron is being ablated, its factor is zero. If it stays normal, its factor is one. As in, In this model, "ablating" a neuron means removing all of its outgoing chemical synapses. This gives a clean and controlled way to simulate the loss of a single neuron in the network.

Motion metrics

To quantify locomotion I used two simple measures derived from the worm_motion_log.txt file that Sibernetic produces for each run. Each line of this log contains a time stamp and then the x and y coordinates of a set of points along the worm's body, tagged with x and y.

For each time step I first reconstruct a "midline" position by averaging the coordinates across all body points. In practice, I take the mean of all Y values to get a single lateral position and the mean of all x values to get a single forward position for that frame.

The lateral range of a run is then defined as the maximum minus the minimum of these mean y positions over the 12 ms simulation. Intuitively, it measures how far the body's midpoint swings side to side during the run. The forward velocity is computed as the net change in the mean x position between the start and end of the run, divided by the number of time steps. This gives the average forward displacement per simulation step, which is proportional to the worm's crawling speed in the model.

Both metrics are expressed in the same arbitrary coordinate units that OpenWorm uses internally. I do not attempt to convert them into physical distances or speeds (such as millimetres or mm/s); instead, I treat them as dimensionless simulation units and only compare values across different runs.

For each ablation and for the unmodified baseline I record the lateral range and forward velocity. I then compute the absolute difference between each ablation and the baseline, and the percent change, defined as 100 times the difference divided by the baseline value. All metrics are written to a single metrics.csv file, which I use for the summary statistics and plots in the Results section.

Relation to network-control predictions

To connect to the work of Yan et al. [2] and Towlson & Barabási [4], I mark the 20 individual neurons predicted to strongly affect muscle controllability: AS08–AS11, AVAL, AVAR, DA07–DA09, DB05–DB07, DD04–DD06, VA12, VB11, VD12, VD13, PDB within my set of ~280 ablated neurons. This allows a direct comparison between the locomotion metric changes for control neurons and the rest of them. Additionally, it lets me show them in the visualizations cleanly.

Results

Baseline locomotion

A baseline crawling simulation with the unmodified connectome produced the following locomotion metrics over the 12 millisecond run, the lateral range was approximately 1.55×10^{-2} whereas forward velocity is approximately 2.82×10^{-4}

These values are small in absolute terms, reflecting the short duration of the simulation, but they serve as the reference point for all ablations. In the metrics.csv file, this baseline run appears as the row with run_name = baseline and zero change in all “delta” and “percent_change” columns.

Single neuron ablation effects

I then analysed 279 single-neuron ablations, each corresponding to removal of all outgoing chemical synapses from the neuron. For each ablation, the script computes the change in lateral range, and the change in forward velocity, both as absolute differences and as percent changes relative to the baseline run. Note that the magnitude numbers here are for a specific part in the natural oscillatory motion of the worm and, if the worm were to have been started in a different position, or a different 12ms snapshot were calculated, it’s likely that the magnitudes could be much different or even possibly left to right flipped. For this reason, I don’t consider the magnitude to be the most important factor, but simply the presence of a change at all given the ablation of that neuron. With that said, across these 279 ablations, the changes are small:

In the Lateral range, the average absolute percent change is about 0.009 percent with the max at about 0.41 percent, for neuron VB04. For forward velocity the average change is about 0.025 percent with the max at about 0.95 percent, for neuron VD06

These small absolute effect sizes are partly a consequence of the experimental design described above and as stated earlier in this paper, the simulations are very short and the precision and reproducibility of the results is very high compared to what's possible in real animals. Thresholds at 0.01 %, 0.10 %, and 0.50 % absolute percent change provide a reference point to compare them by. For lateral range, 32 neurons exceed 0.01 percent, 9 neurons exceed 0.10 percent, none exceed 0.50 percent. For forward velocity: 27 neurons exceed 0.01 percent, 9 neurons exceed 0.10 percent, and 7 neurons exceed 0.50 percent, with the largest changes for VD06, VD05, VD07, DB02, VB03, and VB04.

Yan-predicted "muscle control" neurons versus others

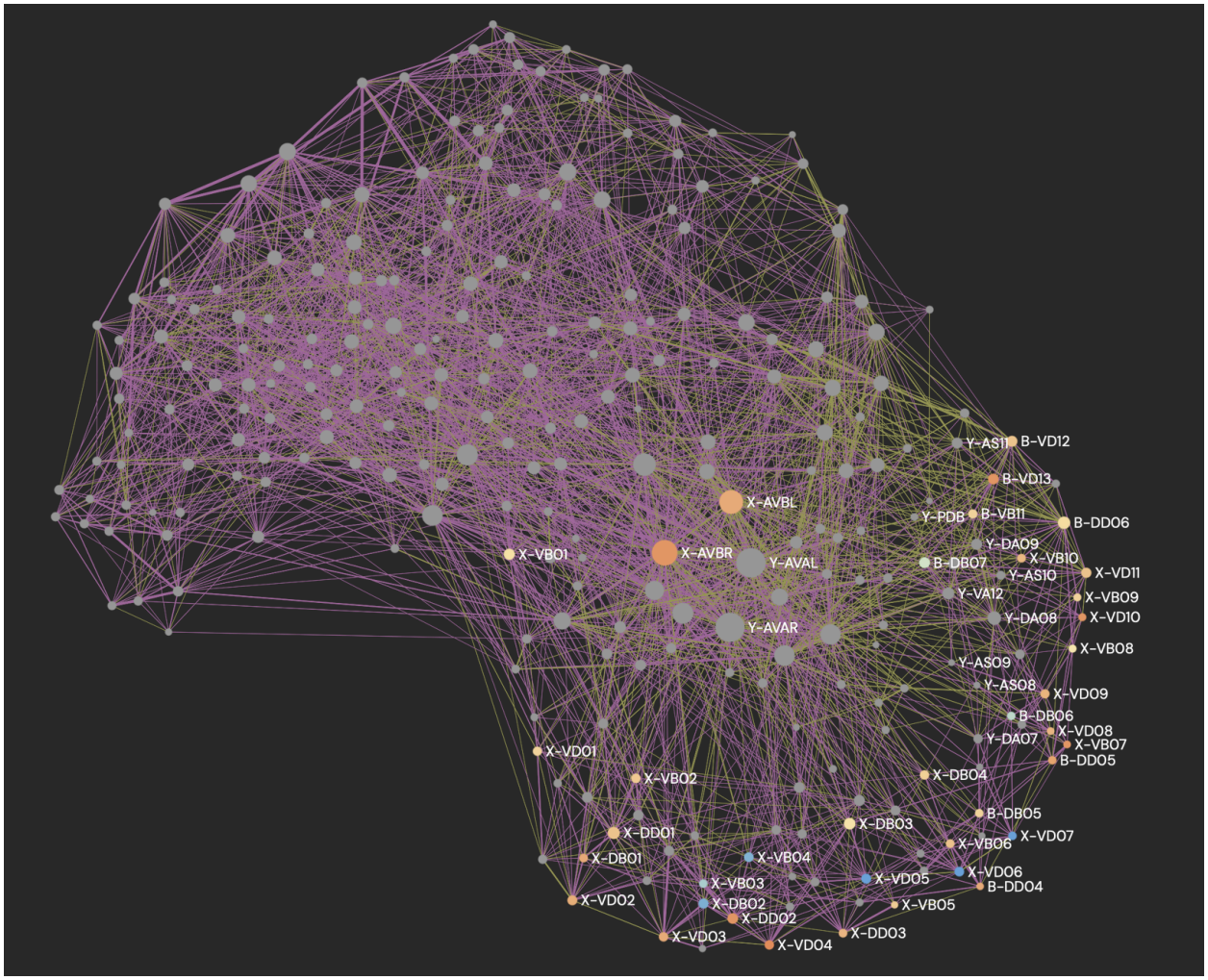
Yan et al. predicted 20 individual neurons whose integrity should be particularly important for muscle control: AS08 to AS11, AVAL, AVAR, DA07 to DA09, DB05 to DB07, DD04 to DD06, VA12, VB11, VD12, VD13, and PDB. All 20 of these neurons appear in my ablation set. Within the Yan set, the neurons with the largest effects are mainly DB05, DB06, DB07, VB11, and VD12, with DB05, DB06, and DB07 showing lateral range changes up to roughly 0.26 percent and DB06 and DB07 also exhibiting the largest forward-velocity changes, around 0.5 percent in magnitude.

However, the very largest forward-velocity changes overall (up to about 0.95 percent) occur in several ventral cord motor neurons outside the Yan set, notably VD05 to 07, DB02, VB03, and VB04. Again note that these magnitudes were observed in a short duration test from an identical baseline position. The results might seem small, but given that the run is so short and tightly controlled, even sub-percent changes correspond to big differences in how the worm is moving.

I found eight nodes that occurred in my experimental result set, i.e. the subset of nodes with differing metrics from baseline, and the Yan set, VD12, VD13, DD05 to 07 and DB05 to 07. Which is a significant overlap, furthering the idea that these two analyses are exploring a similar space. In summary, the Yan neurons are modestly more sensitive on average than the rest of the network, especially in forward velocity and lateral range. Several non-predicted motor neurons produce comparable or larger changes in these metrics.

Given all this, I created the Visualization below, where the nodes in the Yan set are labelled beginning with Y, the eXperimental set are labelled with an X, and those in Both groups are labelled with a B, with the names of the nodes only in those sets labelled. The node's colour is its magnitude of change over baseline, whereas its size is its degree. Edge colour is edge type, (chemical or electrical), and its width is the weight of the edge. Visually, the Yan and X group

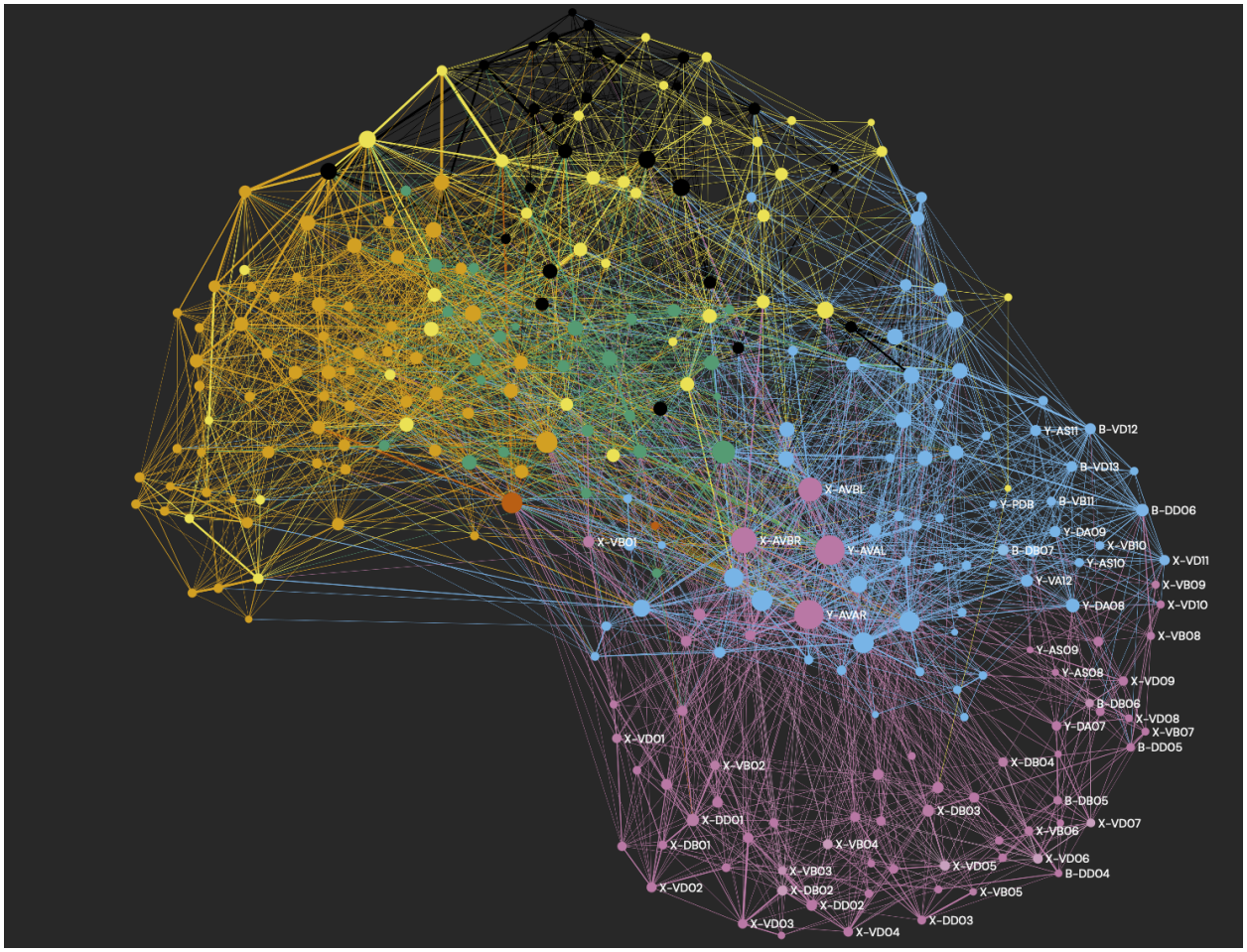
form a cohesive picture, all located in a small subset of the neuron groups of the worm.



Community detection

Next, using the Louvain community detection algorithm at a resolution of 1.3, seven communities were found. Interestingly, all of the nodes in the X and Y sets are located entirely within just 2 of these communities. Especially in the community coloured pink in the visualization below, where a majority of the nodes are in the sets described earlier. Note that the colour palette show below is a colour blind friendly collection of shades described in Bang

W. 2011 [9]



Conclusions / Discussion

This project develops an entire pipeline for testing how connectome perturbations affect simulated locomotion in the OpenWorm model. Using the c302 representation of the C. elegans somatic connectome, I procedurally ablate individual neurons by removing their outgoing chemical synapses, run short (12 ms) neuromechanical simulations, and compute simple geometric and kinematic metrics from midline trajectories.

In addressing RQ1, the full ablation run shows that simulated forward crawling, as captured by lateral range and forward velocity, is remarkably robust to single neuron ablations over this time window in general. However, for roughly 14% of neurons, ablating them creates measurable (non-zero) differences in the worm's motion. More neurons than expected, as compared to those discovered in the Yan set.

For RQ2, the key comparison is between neurons identified by Yan et al. [2] as critical for muscle control and the rest of the network. In my simulations, most of these 20 neurons show changes in the behaviour of the worm, especially in forward velocity. Several non-predicted neurons (e.g. VD05–07, DB02, VB03–04) exhibit the largest individual changes. This suggests that, at least for short episodes of straight crawling in the OpenWorm C2 configuration, the mapping from network control predictions to observable changes is pretty nuanced, diverging from the previous literature, while still remaining somewhat cohesive with it. As in, the neurons highlighted by my experiment are in similar groupings to the ones previously discovered.

Regarding RQ3, the project highlights a trade off between simplicity and sensitivity. The simple metrics used here are easy to compute from existing OpenWorm logs, and easily understandable. On this time scale, it seems there is not enough variance in the body shapes of the worms to use more advanced metrics like eigenworms [5]. And simple metrics are very capable of showing which neurons have an effect on the worm's motion in this sense.

Future work could therefore project the same trajectories into eigenworms, or other more complex metrics and figure out whether ablations in the computational model converge with the pre existing research over longer simulation runs. Or examine longer simulations with varied initial conditions to further understand the magnitude of change occurring instead of focusing on simply whether it is occurring like in my paper. Future work could also move beyond single neuron ablations to double and triple synthetic ablations in the spirit of Towlson & Barabási [4], which are known to produce stronger predicted results, this however expands the combinatorial space greatly and would require a significant amount of computing power.

From a broader connectomics perspective, this work illustrates both the promise and the difficulty of closing the loop from structure to function. Even for the relatively small C. elegans connectome, with a detailed wiring diagram and an ambitious in silico model, predicting behavioural consequences of neural perturbations remains challenging. The pipeline developed here is a step towards systematically exploring that space, enabling detailed in silico ablation studies that complement theoretical analysis and experimental work.

Code and Data Information

The scripts described above are included with this paper in a zip folder, along with the metrics file and some of the visualizations used. The outputs of the simulations are far too large to share here (up to gigabytes per run). All other code, including OpenWorm, Sibernetic, and c302 can be found at the openworm website [8]

Addendum / Postscript

Overall, this research project underwent several course corrections, from building a genetic algorithm, to attempting to run mass simulations in parallel on ARC, there were many dead ends and detours, I learned something from each of them. From the GA, I learned to consider the required scale of a computation before attempting to run it. Given 279 neurons, and choosing a few to modify in very incremental amounts, in this case maybe by increasing the weight of say four of the chemical synapses by tiny increments results in a monstrous search space, and initial experiments were taking far too long to be feasible in this time frame. Running a nine hour set of simulations every night to get 2.25 seconds of simulation split across just a handful of connectomes, just to retune the parameters of the simulation and try again the next night in hopes I could find ones that worked in order to scale up wasn't cutting it. From ARC I learned to come up with a run size that is achievable with the tools you already have working, and, if time permits, expand from there. Small scale first then large scale when you've settled on a direction or tuned the parameters well enough. I explored the Eigenworm classification paper, and began calculating the eigenvalues of the worm based on the data I had, but without an intuitive understanding of what I was doing, interpretation of those details was out of my reach. I should really re-learn Linear algebra again. I had a good time in this class, and feel a spark of curiosity that will keep me exploring on my own for quite some time. There is much, much more to learn.

References

[1] WormWiring project, "Adult hermaphrodite wiring diagram," based on White et al. 1986. Accessed 2025. https://www.wormwiring.org

[2] G. Yan, P. E. Vértes, E. K. Towlson et al., "Network control principles predict neuron function in the Caenorhabditis elegans connectome," Nature, vol. 550, pp. 519–523, 2017. https://www.nature.com/articles/nature24056

[3] Towlson EK, Vertes PE, Yan G, Chew YL, Walker DS, Schafer WR, Barabasi A-L. 2018 Caenorhabditis elegans and the network control framework—FAQs. Phil. Trans. R. Soc. B 373: 20170372. http://dx.doi.org/10.1098/rstb.2017.0372 https://royalsocietypublishing.org/doi/pdf/10.1098/rstb.2017.0372

[4] E. K. Towlson and A.-L. Barabási, "Synthetic ablations in the C. elegans nervous system," Network Neuroscience, vol. 4, no. 1, pp. 200–216, 2020. https://pmc.ncbi.nlm.nih.gov/articles/PMC7055645/

[5] G. J. Stephens, B. Johnson-Kerner, W. Bialek, and W. S. Ryu, "Dimensionality and dynamics in the behavior of C. elegans," PLoS Computational Biology, vol. 4, e1000028, 2008 https://journals.plos.org/ploscompbiol/article?id=10.1371%2Fjournal.pcbi.1000028

[6] G. P. Sarma et al., "OpenWorm: overview and recent advances in integrative biological simulation of Caenorhabditis elegans," Philosophical Transactions of the Royal Society B, vol. 373, 2018 . https://royalsocietypublishing.org/doi/10.1098/rstb.2017.0382

[7] P. Gleeson et al., "c302: a multiscale framework for modelling the nervous system of C. elegans," 2018 .https://pmc.ncbi.nlm.nih.gov/articles/PMC6158223/

[8] OpenWorm documentation and project pages, accessed 2025. https://docs.openworm.org/projects/

[9] Wong B "Points of view: Color blindness" https://www.nature.com/articles/nmeth.1618