CMC Project 1

Students: Nina Lahellec (327695), Marine Moutarlier (310703), Marianne Civit Ardevol (325056)

July 20, 2024

Warning: code running instructions

In order to run the plot_results.py file, please use this format to indicate for which exercise the plots are desired:

```
python plot_results.py exercise_nb
```

Where exercise_nb $\in \{0, 1, 2\}$ is the number of the exercise. The exercise 1 and 2 will run automatically the multiple plots.

Questions

At this point you can now start to work on implementing your exercises.

1. Implement the equations for a sine wave controller

Implement a sine wave swimming controller that generates a periodic traveling wave for the left and right muscle activations (M_{L_i} and M_{R_i}, for $i = 0, ...14$) to actuate the zebrafish according to equation 1.

$$\begin{aligned} M_{L_i}(t) &= 0.5 + \frac{A}{2} \cdot \sin\left(2\pi\left(f \cdot t - \epsilon \cdot \frac{i}{N_{joints}}\right)\right) \\ M_{L_i}(t) &= 0.5 - \frac{A}{2} \cdot \sin\left(2\pi\left(f \cdot t - \epsilon \cdot \frac{i}{N_{joints}}\right)\right) \end{aligned} \tag{1}$$

where ϵ is the wavefrequency, A is the amplitude and f is the frequency of the wave. Note that that equation 1 guarantee that the active stiffness component $M_{sum_i} = (M_{L_i} + M_{R_i})$ is equal to 1 at all times, and that the muscle activation difference $M_{diff_i} = (M_{L_i} - M_{R_i})$ has amplitude A.

Test the controller's ability to generate swimming locomotion for fixed values of $\epsilon \in [0, 2]$, $A \in [0, 2]$ and $f \in [1, 5]Hz$ (test different parameter combinations). Show plots of the left and right muscle activations M_{L_i} and M_{R_i}, and of the animal head trajectory in the (x,y) plane. Also, show the evolution of the joint angles. Report the performance metrics in your report. You can also upload a showing that the animal swims correctly.

Answers to question 1

In order to implement the sine wave controller behaving as described by the two equations above (equation 1), we needed to modify the code. We modified the step function in wave_controller.py such that the activation function returned followed equation 1, alternating between the left and right muscle activations.
Once this was done, we tested the controller's ability to generate swimming locomotion for fixed values of the given parameters. First of all, we tweaked the values of the controller until the zebrafish performed a movement that resembled its real motion. We found these values to be : $A = 0.5$, $f = 3$ and $\epsilon = 1$, which give the following plots:

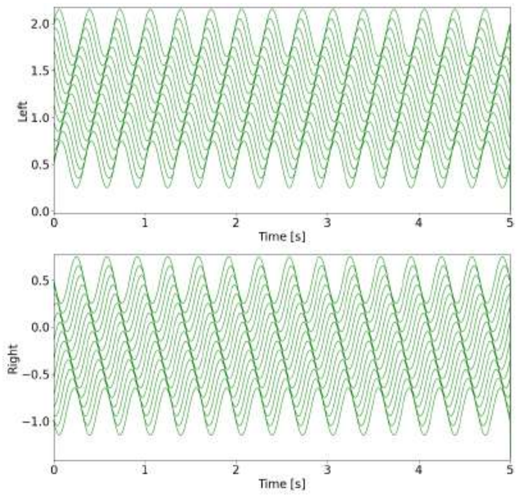


Figure 1: Left-right plot for $A = 0.5$, $\epsilon = 1$ and $f = 3Hz$

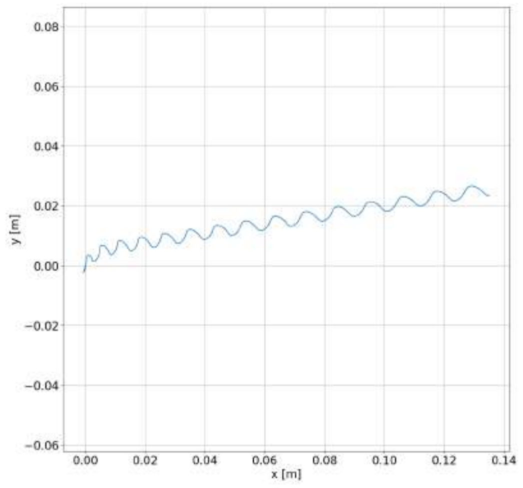


Figure 2: head trajectory for $A = 0.5$, $\epsilon = 1$ and $f = 3Hz$

We can see that the left and right muscle activations follow an oscillatory movement as expected from the equation 1 which implements a sine wave. Furthermore, their amplitudes are equal to 0.5, which is what we had set as input with $A = 0.5$. We also observe that there is a travelling wave (constant phase lag at 100%) between each joint as expected from the course information.
The head position is also oscillatory which corresponds to the main movement of the fish. It can be seen that the trajectory doesn't stay at position $y = 0$, it first turns a bit to the left and then follows a straight line. This could be due to the initial position as well as to small asymmetries in the muscle activations.

For computing the metrics, we used a simulation time of $10s$ which allows the oscillations to stabilise and therefore to get more accurate values. For these mid-range values, the computed metric values are the following:

Metric	Value
Frequency	2.9997
IPLS	0.3344
Wave Frequency	1.0030
PTCC	1.8261
Amp	1.0
Fspeed PCA	0.0378
Lspeed PCA	3.4130e-06
Fspeed Cycle	0.0377
Lspeed Cycle	0.0002
Torque	0.0123

Table 1: Computed Metrics

It can be seen that the frequency, wave frequency and amplitude values follow the inputs given. Only the amplitude is doubled because of the way that it has been defined, from peak-to-peak.

Some interesting metrics are the forward and lateral speeds computed either by PCA and by cycle. Comparing these values shows that the forwards speeds are both very similar to each other and validate the fact that the forward speed is $\approx 0.038m/s$ in average. The lateral speed is in both cases close to zero. This shows indeed that the zebrafish is moving forwards and that there is a slight lateral

movement.
We can see in figure 2 that the trajectory seems to deviate towards the left side. In the computed metrics, the lateral speed computed by PCA is much smaller than the one computed by cycle. This can be explained by the fact that the PCA model looks at the instantaneous speed of the fish whereas the cycle model computes the overall speed. Therefore, since the fish moves very slightly to the left, the PCA speed is small but looking at the end trajectory we can see that there was a real deviation to the left and we can explain the bigger lateral speed given by cycle computation.
Looking at the peak-to-through correlation (ptcc), we are able to see that the oscillations are quite stable since the value is $\sim 1.83 > 1.5$. Finally, the value of the total exerted torque is at ~ 0.012 which seems to be in correct order of magnitude and not too big to constrain the fish's movement. This will be further analysed in the next question of the report.

Varying the values of the **amplitude** (but keeping $f = 3$ and $\epsilon = 1$), gave these results:

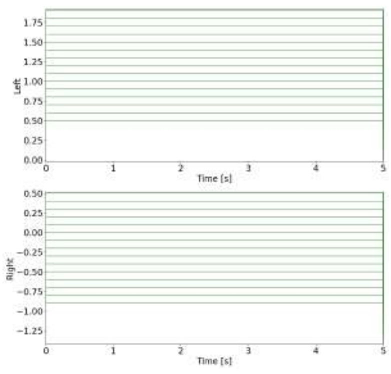


Figure 3: Left-right muscle activation plot for $A = 0$

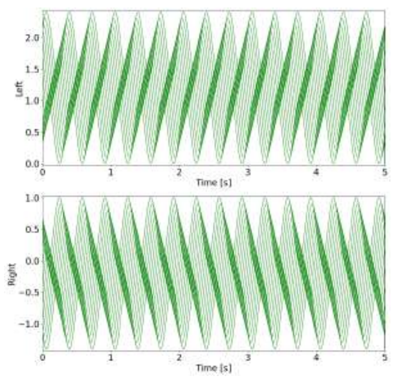


Figure 4: Left-right muscle activation plot for $A = 1$

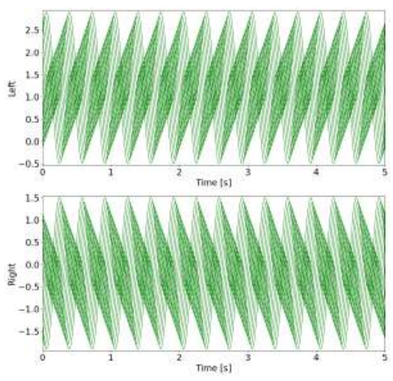


Figure 5: Left-right muscle activation plot for $A = 2$

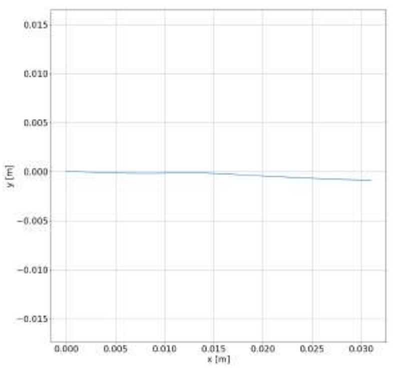


Figure 6: (x,y) head position for $A = 0$

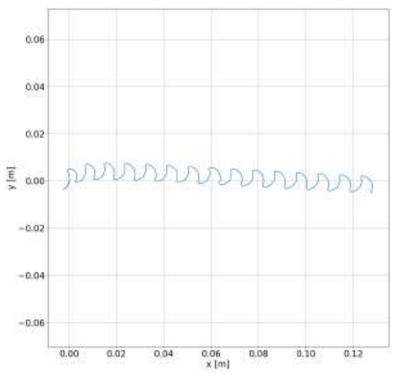


Figure 7: (x,y) head position for $A = 1$

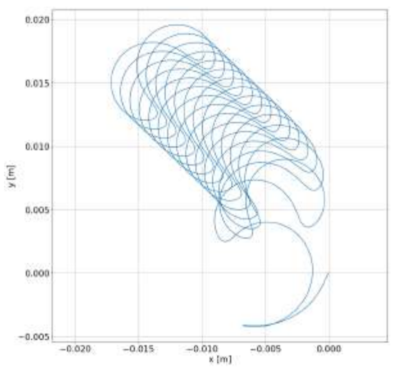


Figure 8: (x,y) head position for $A = 2$

It can be seen that with an amplitude of zero, the zebrafish is not able to swim, This behaviour could have been expected because if there is no amplitude then there is no movement and the equations 1 are constant. Furthermore, the computed metrics reflect this by giving null or almost zero values for all except for the forward speed computed by PCA (Fspeed PCA = 0.0063 m/s). Indeed, the fish does move forward but this might be due to an initial condition on the lateral speed. Interestingly, the forward speed computed by the cycle is zero which seems incorrect but this is due to the fact that the frequency of oscillations is null and therefore there are no oscillations and the speed is set to zero.
By increasing the amplitude, the movement is oscillatory and gets bigger with the amplitude.
Furthermore, increasing the amplitude too much whilst keeping the same wavefrequency, the fish

makes movements that are too big and can't move forwards as shown in figure 8. Here the (x, y) positions for $A < 2$ also seem to deviate but not as much as for the previous plot we had seen in figure 2. This can be due to the fact that the initial conditions have less impact on the fish because its amplitude is bigger and 'overrides' it. Looking at the metrics, the speed increases and then decreases for too large values of A and the torque always increases:

Metric	$A = 0.1$	$A = 0.5$	$A = 1$	$A = 2$
Fspeed PCA	0.0047	0.0378	0.0286	-0.0058
Fspeed Cycle	0.0047	0.0377	0.0286	0.0059
Torque	0.0025	0.0123	0.0246	0.0493

Table 2: Computed Metrics for varying values of A

This shows that having a big amplitude is not always the most efficient way of swimming and that there is a limit. The fact that the torque is bigger reflects the amplitude and the need of the fish to do bigger movements with stronger torques.

Next, we varied the values of the **frequency** (and kept $A = 0.5$ and $\epsilon = 1$) and obtained the following:

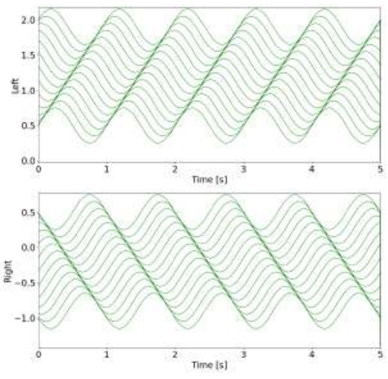


Figure 9: Left-right muscle activation plot for $f = 1$

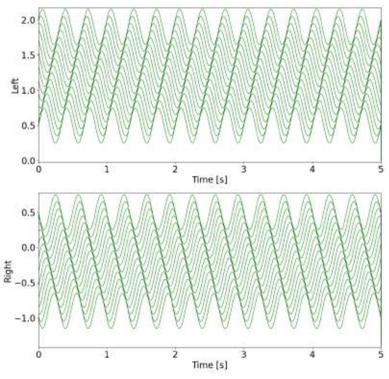


Figure 10: Left-right muscle activation plot for $f = 3$

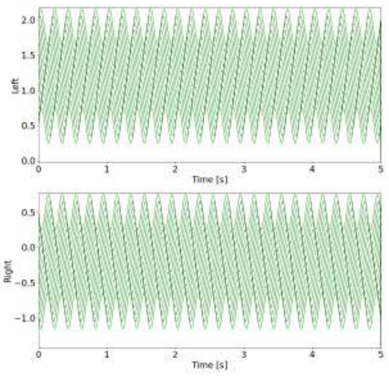


Figure 11: Left-right muscle activation plot for $f = 5$

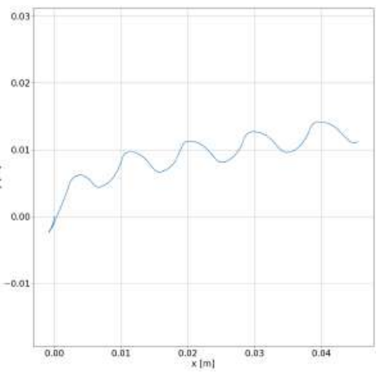


Figure 12: (x,y) head position for $f = 1$

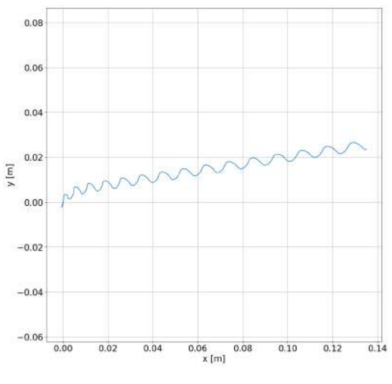


Figure 13: (x,y) head position for $f = 3$

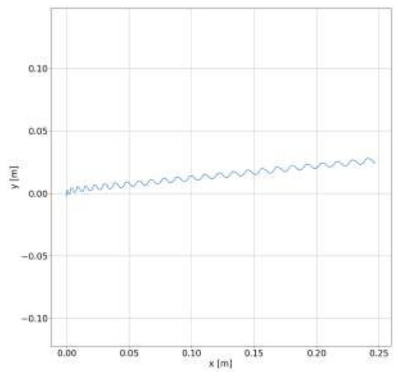


Figure 14: (x,y) head position for $f = 5$

It can be observed that increasing the frequency component yields a faster movement of the left and right muscle activations with the head position that follows. In any case, the zebrafish is able to swim.

For all of these different values of frequency, the torque metric remain more or less constant to 0.0123 and the forward speed increases. Indeed, it can be observed in the (x, y) head position plots that the fish is able to go further with the same amount of time when the frequency increases. The same happens for the lateral speed. Numerically, the values of the forward speed computed by PCA are the following:

	$f = 1$	$f = 3$	$f = 5$
PCA Fspeed	0.0113	0.0378	0.0632

Table 3: Fspeed PCA for different values of the frequency

Once again, the trajectory of the fish is diagonal indicating that the fish was facing a bit towards the left at the beginning.

Finally, varying only the **wavefrequency** component whilst keeping $A = 0.5$ and $f = 3Hz$, gives the following:

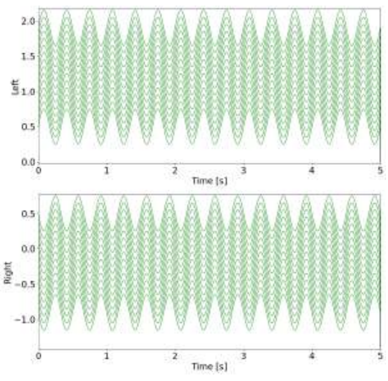


Figure 15: Left-right muscle activation plot for $\epsilon = 0$

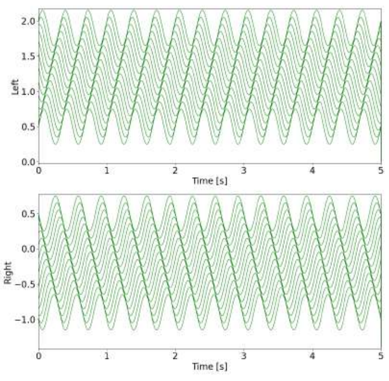


Figure 16: Left-right muscle activation plot for $\epsilon = 1$

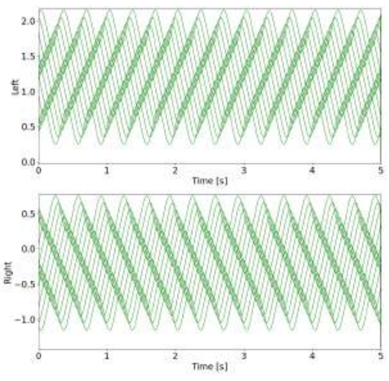


Figure 17: Left-right muscle activation plot for $\epsilon = 2$

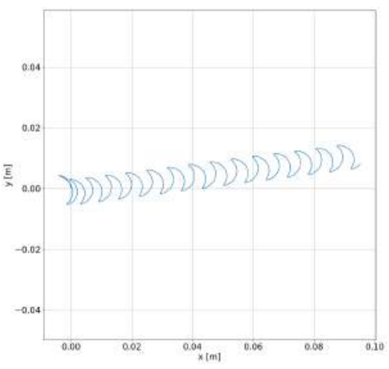


Figure 18: (x,y) head position for $f\epsilon = 0$

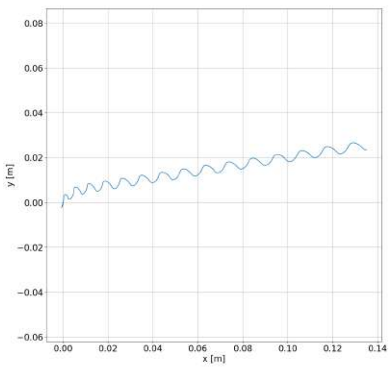


Figure 19: (x,y) head position for $\epsilon = 1$

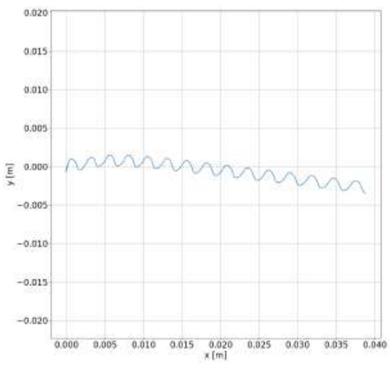


Figure 20: (x,y) head position for $\epsilon = 2$

The value of the wavefrequency has a great impact on the ability of the zebrafish to swim. When the wavefrequency is null, there is no phase lag for the left and right muscle activations and therefore the fish moves forwards with difficulty, when increasing the amplitude, the fish can't move forwards anymore. Furthermore, when $\epsilon = 2$, the phase lag seems to overlap for different joints not efficient in its swimming, we can see that its body does a double wave motion.
All of these parameters are further analysed in the next part of the report, giving a more quantitative analysis.

2. Study the performance of when varying the wavefrequency and amplitude

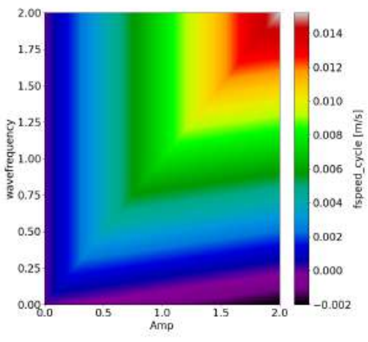
Run a parameter search of the model when varying the wavefrequency and amplitude of the sine wave controller 1. Use the metrics provided (or your own metrics) to study the performance of the model. What speeds can the model achieve? What is the optimal wavefrequency and amplitude?

Answers to question 2

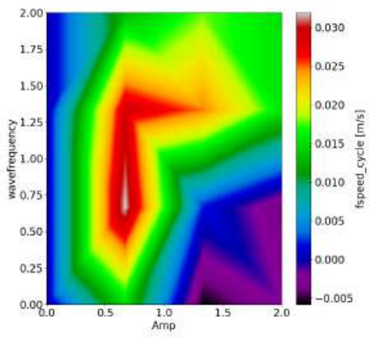
The wavefrequency and the amplitude are important spatial parameters which define the sine wave movement of the fish. Modifying these parameters has an impact on the kinematic and dynamic performances. The parameter search will be done for $\epsilon \in [0, 2]$ and $A \in [0, 2]$, since these ranges are sufficient for analysing local extremum. The number of iterations for all simulations is chosen to 10s, in order to diminish the effect of initial conditions.
The time frequency is chosen at 3Hz for this part.

- **Forward cycle speed**

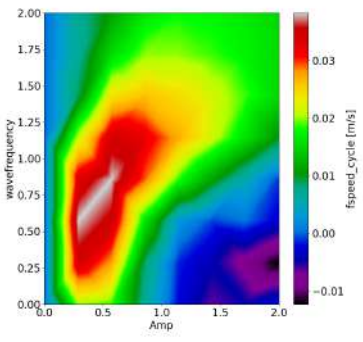
The first metric to be analysed is the forward cycle speed, since it computes the forward speed in the case of oscillatory behaviour.
In order to find the optimal forward cycle speed when varying the wavefrequency and the amplitude, a heat map is plotted as a result of multiple simulations, for a set of different amplitudes and wavefrequencies.
Improving the precision of the parameter search can be done in two ways: increasing the resolution (the number of simulations), or focusing on smaller range of wavefrequency and amplitude.
Firstly, the number of simulations is increased, keeping the range [0,2] for the wavefrequency and the amplitude (Figure 21).



(a) 4 simulations



(b) 16 simulations



(c) 64 simulations

Figure 21: Cycle forward speed with respect to amplitude ($A \in [0, 2]$) and wavefrequency ($\epsilon \in [0, 2]$), for different resolutions.

The result for 4 simulations is unsatisfactory: a grid is drawn from only 4 different values of speeds, which hides all local extrema. With 16 simulations, a specific behaviour appears: the speed tends to increase around a specific region. With 64 simulations, it becomes clear that there is a region of local maximum for $\epsilon \in [0.2, 0.8]$ and $A \in [0.25, 1.25]$.
It can be interesting to focus on these ranges to get a more precise value of the maximum forward speed (Figure 22). The resolution is there fixed to 16 simulations, which is enough since the range is progressively reduced.

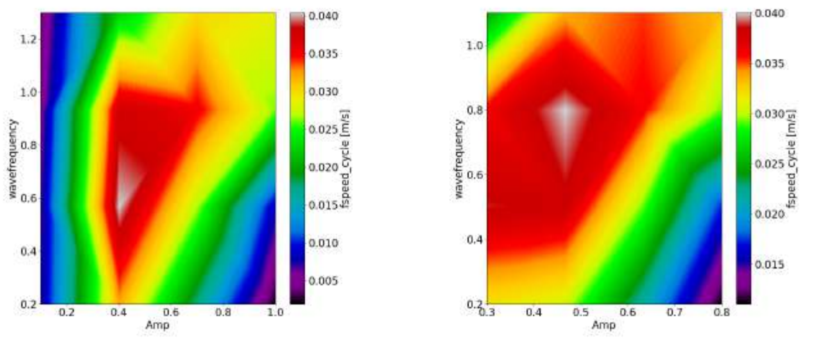

(a) Range 1 (b) Range 2

Figure 22: Cycle forward speed with respect to amplitude and wavefrequency, decreasing their ranges, and with 16 simulations.

The region of maximum speed is found to be around $0.040 m.s^{-1}$, for an amplitude of around 0.45 and a wavefrequency of 0.80.
This bell-shaped curve indicates that there is an optimal balance between wave frequency and amplitude that maximizes the fish's forward speed. Beyond the optimal point, further increases in either wave frequency or amplitude leads to decreasing performances.

As the fish's tail oscillates, it pushes water backwards, generating thrust, according to Newton's third law of motion. Increasing the wave frequency of the sine wave controller increases the rate at which the tail oscillates, resulting in faster forward movement, while increasing the amplitude of the increases the magnitude of the tail movements, generating more powerful thrust and accelerating the fish. But after an optimal combination of amplitude/wavefrequency, the oscillations may generate larger and more turbulent waves. This increased turbulence can create additional hydrodynamic resistance, effectively slowing down the fish's forward speed, or even making it follow a backward motion. Indeed there is a region in Figure 21 of negative speeds, when the amplitudes are high ($A > 1$) and the wave frequencies very small ($\epsilon < 0.7$).

- **Lateral cycle speed**

The same procedure is applied to identify the region of maximum lateral speed: first the resolution is progressively increased to select the region of local maxima (Figure 23), then the range of amplitude and wavefrequency is decreased to get more precise results (Figure 24).

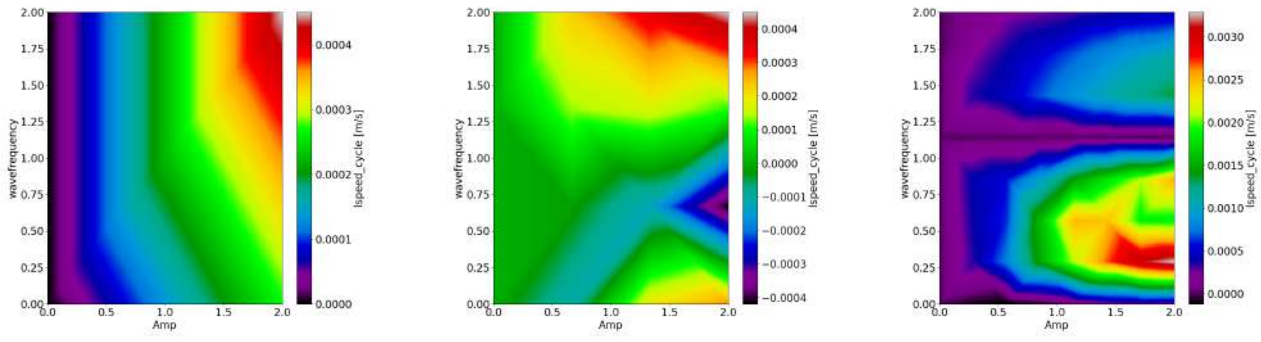
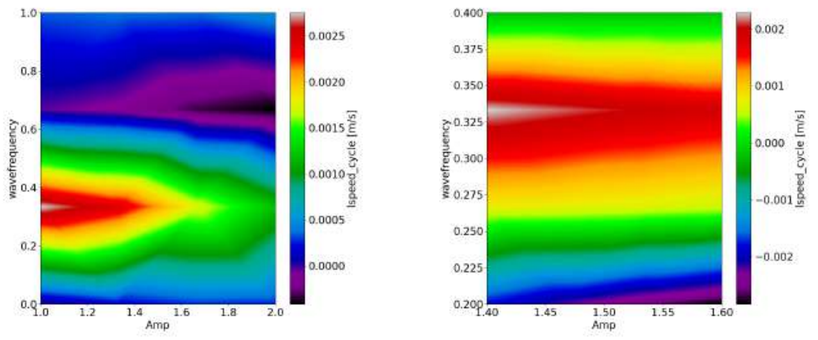

(a) 4 simulations (b) 16 simulations (c) 64 simulations

Figure 23: Cycle lateral speed with respect to amplitude ($A \in [0, 2]$) and wavefrequency ($\epsilon \in [0, 2]$), for different resolutions



(a) Range 1 *(b) Range 2*

Figure 24: Cycle lateral speed with respect to amplitude and wave frequency, decreasing their ranges, and with 16 simulations.

The lateral speed seems to be slightly more dependent on the wavefrequency than on the amplitude of the signal. However, this metric is not relevant for this kind of controller where the fish only goes forward. Indeed, as can be seen on the scale of the plots the lateral speed is negligible compared to the forward speed seen previously.

- **Torque consumption**

For any number of simulations, or precision of range for the wave frequency and the amplitude, the torque consumption is found to always have the same behaviour: it increases with respect to the amplitude, and is almost invariant to the wave frequency (Figure 25).

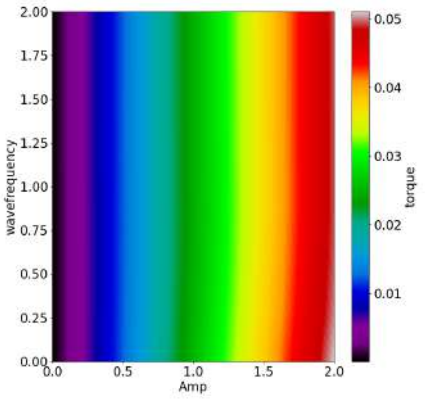


Figure 25: Torque consumption with respect to amplitude and wavefrequency.

The torque consumption is computed as:

$$T_{tot} = \sum_t \sum_{seg} ||\tau_{seg}(t)|| \tag{2}$$

Where $\tau_{seg}(t)$ is the active term of the output torque, it is proportional to:

$$M_{diff_i} = (M_{L_i} - M_{R_i}) = Asin(2\pi ft - \frac{\epsilon i}{N})$$

The high dependency to the amplitude compared to the wave frequency suggests that the fish's motion is primarily influenced by the extent of deviation from its equilibrium position rather than the speed of oscillations. Indeed, the amplitude of the sine wave determines the maximum deviation from the fish's equilibrium position. The greater the deviation, the more torque is required to move the fish back to its desired trajectory.

- **Cost of transport**

As a metric of performance optimization, it can be interesting to introduce a measure of cost of transport: the higher power consumption, the higher the cost, and the higher the forward speed, the lower the cost (because we go further). This measure would allow to get a sense of trade off in locomotion optimization.
A metric of total power consumption is firstly computed, using the active torque and the angular joint velocities, summing this product across all joint and for all time iterations:

$$P_{tot} = \sum_t \sum_{seg} ||\tau_{seg}(t)\dot{\theta}_{seg}(t)||$$

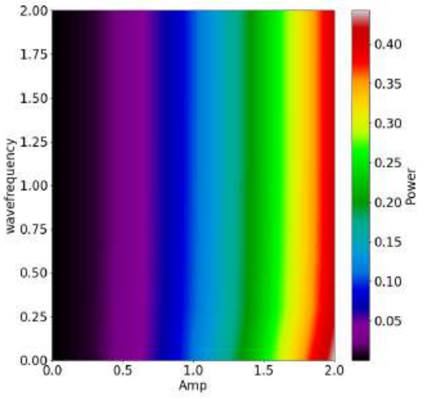


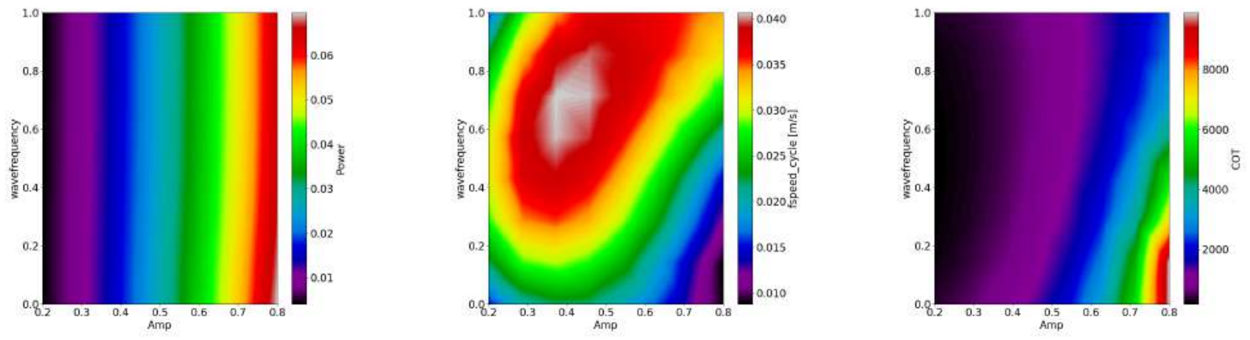
Figure 26: Power consumption with respect to amplitude and wavefrequency, for 64 simulations points.

As seen in Figure 26, the power consumption behaves as expected: it increases with the amplitude, and is invariant to the wave frequency. Indeed, adding the angular joints velocities will not introduce a new dependence on the wave frequency. It is coherent that the higher the amplitude, the more power is consumed, because again more energy is needed when the spatial deviation gets bigger.

Now let's introduce the cost of transport, by dividing the power consumption by the mass, the gravitational acceleration, and the forward constant speed:

$$COT = \frac{P_{tot}}{mgv}$$

Since we are always comparing the same animal, the mass of the fish can be set to a constant approximation of 80mg. Indeed, the mass doesn't vary in the simulations and using a default value for it is useful.
As for the speed, it is the forward speed cycle computed in the metrics provided.
To get a coherent plot of the COT with respect to the amplitude and wave frequency, a specific range for A and ϵ is chosen, to prevent negative or null forward speeds, which would alter the range of results. Indeed, a null speed would result in an infinite COT, and a negative speed does not have a lot of sense here: the cost should get bigger when the speed gets bigger in the negatives, which is not what is computed here.



(a) Power consumption(W) *(b) Forward cycle speed(m/s)* *(c) COT()*

Figure 27: Power consumption, forward speed, and COT drawn from 64 simulations points, with respect to amplitude and wave frequency

The plot of the COT shown is Figure 27 looks coherent: the cost is minimum when the speed is maximum, and the power consumption is minimal. The COT increases with the amplitude, which was predictable given the behaviour of the power consumption. The values of the costs are extremely large, due to small values on the denominator. It is clear here that a small speed results in a huge cost effect. There is there a need to perform a trade off, the power consumption needs to be minimized, but choosing a too small amplitude (less than 0.2) would result in an infinite cost since the speed would be null.

3. Implement and study a non-sinusoidal controller

In you implemented a sinusoidal propagating wave to actuate the model. In reality, the left-right muscle activations in a real animal might are less stereotypical than perfect sines. Modify equation 1 to implement a square wave controller, that modifies the signals to make them similar to a square wave. You should implement a gain function similar to the ones used in Lab4, and have a parameter that can control how steep the square wave can be. Test different steepness values and plot the performance of the model.

Answers to question 3

In order to implement the square wave controller we needed to modify the code. We modified the step function in `wave_controller.py`. We created a new variable, *self.mode* in `parameter_simulations.py`. This variable is set to 'sine' by default, but would be changed to 'square' when running the `exercise2.py` file. It would do so such that the activation function returned followed equation 13, if the mode 'sinus' is set, and the new equations if the mode 'square' is set.
The way that the square controller was implemented was to use a sigmoid function. When a sigmoid function is applied to a sine wave, it creates a square-like appearance due to the sigmoid's tendency to flatten or clip the peaks and troughs of the sine wave. This effect occurs especially with steep sigmoid functions that rapidly transition between their maximum and minimum values, resulting in squared-off shapes. The sigmoid function is defined like this:

$$\sigma = \frac{1}{1 + \exp(-\text{steepness} \cdot (x - theta))} \tag{3}$$

with

1. x: The input to the sigmoid function. It could be a scalar, vector, or matrix. Here it is our sinus function that was computed before.

2. Steepness: A parameter that controls the steepness of the sigmoid curve. Higher values result in a steeper curve, while lower values result in a flatter curve.
3. θ: The threshold parameter of the sigmoid function. It determines the midpoint of the curve where the output is approximately 0.5. The theta parameter (θ) in the context of a sigmoid function represents the phase shift of a sinusoidal waveform. it should be $\theta = -\frac{b}{a}$ for $f(x) = \sin(ax + b)$.
4. σ: The sigmoid function itself. It maps the input x to a value between 0 and 1, following an S-shaped curve.

Here is the code implementation of how this was done:

```
activation_func = np.zeros(30)
        j = 0
        i = 0
        while j != 2*self.pars.n_joints:
            activation_func[j] = self.typeofcontroller_l(iteration, timestep, i)
            activation_func[j+1] = self.typeofcontroller_r(iteration, timestep, i)
            j = j+2
            i = i+1

        self.state[iteration,:] = activation_func

        return activation_func
```

```
    def typeofcontroller_l(self, iteration, timestep, j):
        if self.pars.mode == "square":
            x= 0.5 + self.pars.amp/2 *math.sin(math.pi*2*(self.pars.frequency*timestep*iteration - (
                ↪ self.pars.wavefrequency * j /self.pars.n_joints)))
            return 1/(1 + np.exp(-self.pars.steepness * (x - self.pars.theta)))

        elif self.pars.mode == "sinus":
            return 0.5 + self.pars.amp/2 * math.sin(math.pi*2*(self.pars.frequency*timestep*iteration -
                ↪  (self.pars.wavefrequency * j /self.pars.n_joints)))


    def typeofcontroller_r(self, iteration, timestep, j):
        if self.pars.mode == "square":
            x = 0.5 - self.pars.amp/2 *math.sin(math.pi*2*(self.pars.frequency*timestep*iteration - (
                ↪ self.pars.wavefrequency * j /self.pars.n_joints)))
            return 1/(1 + np.exp(-self.pars.steepness * (x - self.pars.theta)))
        elif self.pars.mode == "sinus":
            return 0.5 - self.pars.amp/2 * math.sin(math.pi*2*(self.pars.frequency*timestep*iteration -
                ↪  (self.pars.wavefrequency * j /self.pars.n_joints)))
```

Once this was done, we tested the controller's ability to generate swimming locomotion for fixed values of the given parameters. We tested the controller with certain values of the parameters. With $A = 2$, $\epsilon = 1$, $f = 2Hz$, $\theta = 0.5$, and steepness $= 50$.

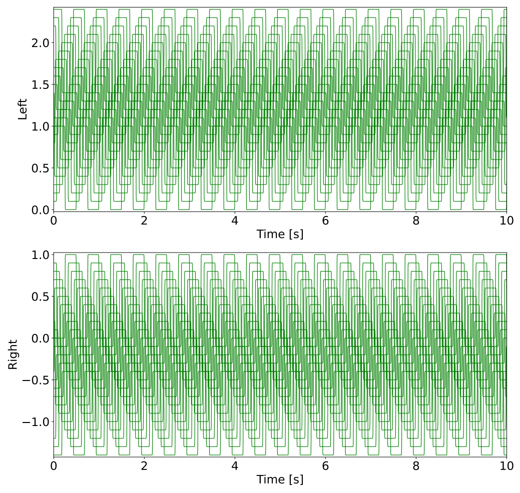
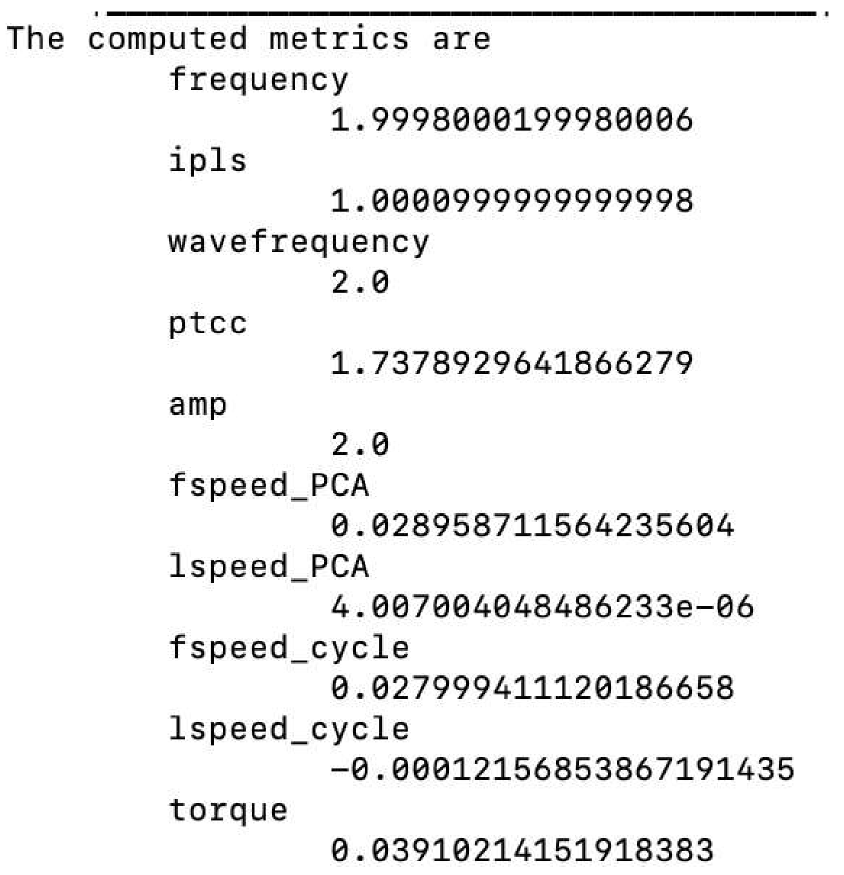


Figure 28: Left-right plot for $A = 2$, $\epsilon = 1$, $f = 2Hz, \theta = 0.5$, and steepness = 50

We can see that the left and right muscle activations follow a square movement. Furthermore, their amplitudes are equal to 2, which is what we had put as input with $A = 2$ and that there is a travelling wave (constant phase lag) between each joint.
For these values, the computed metric values are the following:



```
The computed metrics are
        frequency
                1.9998000199980006
        ipls
                1.0000999999999998
        wavefrequency
                2.0
        ptcc
                1.7378929641866279
        amp
                2.0
        fspeed_PCA
                0.028958711564235604
        lspeed_PCA
                4.007004048486233e-06
        fspeed_cycle
                0.027999411120186658
        lspeed_cycle
                -0.00012156853867191435
        torque
                0.03910214151918383
```

Figure 29: head trajectory for $A = 2$, $\epsilon = 2$, $f = 2Hz, \theta = 0.5$, and steepness = 50

It can be seen that the frequency, wave frequency and amplitude values follow the inputs given. The ptcc measures the stability of the signals based on the difference between the maximum and minimum of the autocorrelogram of oscillatory signals. Here, the computed value is approximately 1.73789, indicating stable oscillations.

Varying the values of the **steepness** (whilst keeping the other values fixed), gave these results:

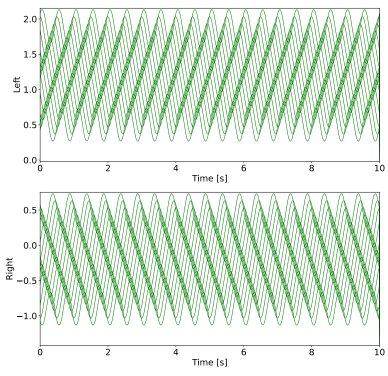


Figure 30: Left-right muscle activation plot for Steepness = 1

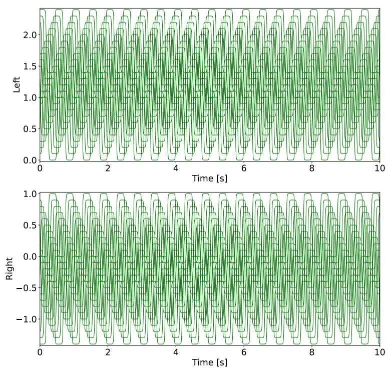


Figure 31: Left-right muscle activation plot for Steepness = 10

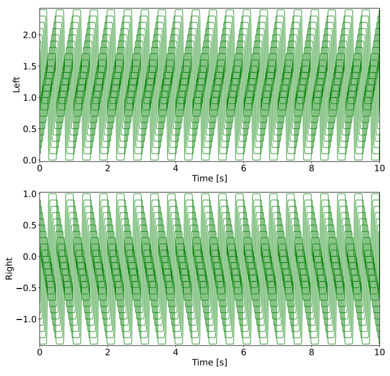


Figure 32: Left-right muscle activation plot for Steepness = 20

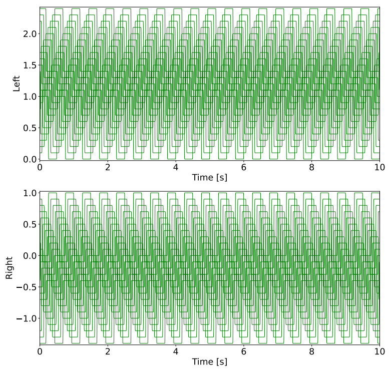


Figure 33: Left-right muscle activation plot for Steepness = 50

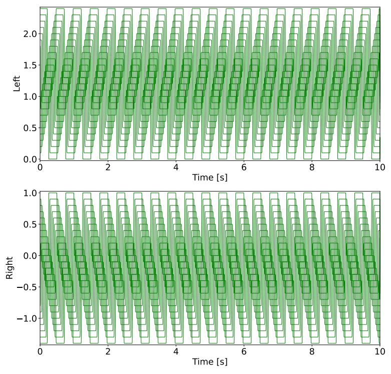


Figure 34: Left-right muscle activation plot for Steepness = 100

The steepness is a measure of how sharp the edges of the controller are. Here it can be seen that for a low value of the steepness (example 1) the edges are not sharp and the controller looks almost exactly like a sinus wave. The more the value increases, the sharper it gets. Already at around 50/100, it reaches the sharpest shape.

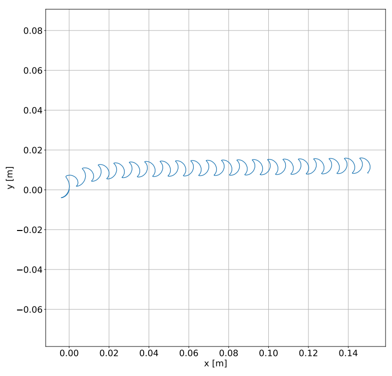


Figure 35: Trajectory plot for Steepness = 10

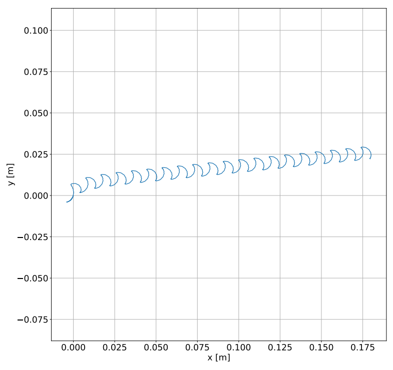


Figure 36: Trajectory plot for Steepness = 20

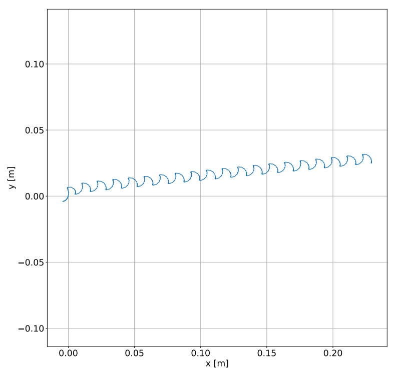


Figure 37: Trajectory plot for Steepness = 50

Increasing the steepness parameter for the sigmoid function in our fish movement model, makes the model more sensitive to variations in the input. As a result, even small deviations from the original trajectory, such as slight fluctuations or errors in the input data, can cause significant changes in the output trajectory. When the steepness increases, the sigmoid function becomes steeper, making the curve transition more abruptly between values. Consequently, minor perturbations or errors in the input data will be amplified in the output trajectory, leading to deviations from the original path.

Then, we plotted graphs for different values of θ. These were plotted for A = 2, frequency f = 2Hz, wavefrequency $\epsilon = 1$ and steepness = 50.

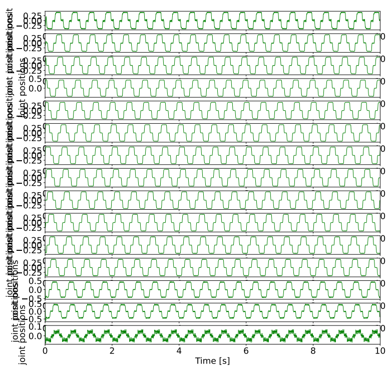


Figure 38: joints positions for $\theta = 0$

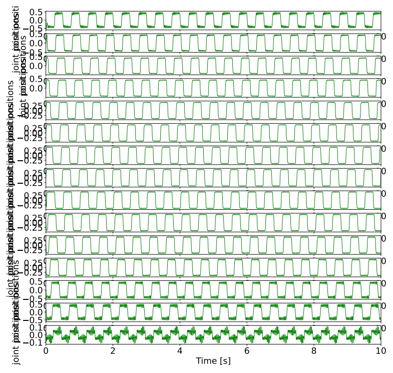


Figure 39: joints positions for $\theta = 0.5$

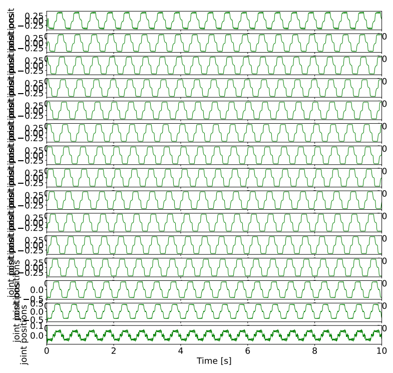


Figure 40: joints positions for $\theta = 1$

The plots above show that varying the value of θ changes the phase shift from a sinusoidal to a square. For the provided parameter values, the shift is in phase when $\theta = 0.5$. It is out of phase for the other values of θ. In the provided equation, the threshold parameter θ is in phase with the oscillation at $x = 0.5$ because at this point, the output of the sigmoid function reaches its midpoint (0.5). This occurs because the sine wave, with an amplitude of 1 (A=2/2) and an offset of 0.5 ($\epsilon * i/N_{joints}$), oscillates between 0 and 1, causing the input to the sigmoid function to vary symmetrically around θ, resulting in a sigmoid output of 0.5 at $x = \theta$, thus appearing in phase with the oscillation.

To better visualize the performance we also decided to plot the speed for different values of the steepness when changing the values of the amplitude and wavefrequency. The frequency was set to f = 1Hz:

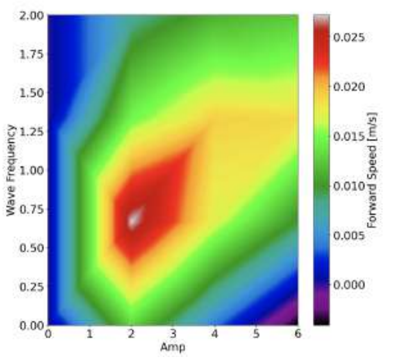


Figure 41: Speed distribution for Steepness = 1

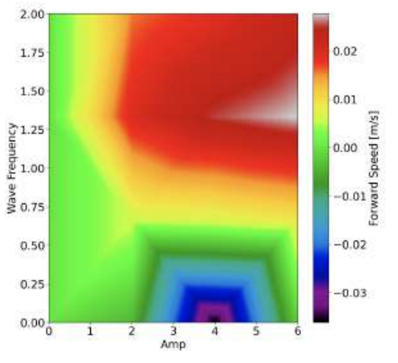


Figure 42: Speed distribution for Steepness = 10

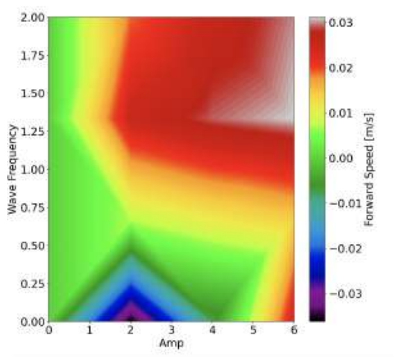


Figure 43: Speed distribution for Steepness = 20

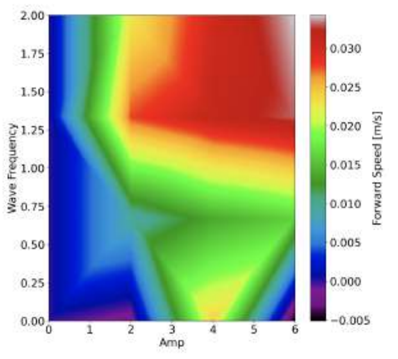


Figure 44: Speed distribution for Steepness = 30

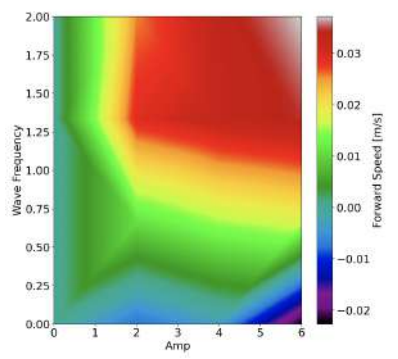


Figure 45: Speed distribution for Steepness = 40

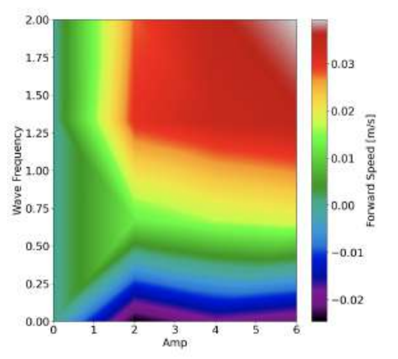


Figure 46: Speed distribution for Steepness = 50

We explored how the steepness of a sigmoid function impacts the performance of a system. The steepness parameter determines how abruptly the system transitions between its minimum and maximum states. When the steepness is low, there is a gentle transition, such that the controller is the less square it can be. There is a concentrated area where the system achieves higher speeds at specific amplitude and frequency settings.
When the steepness increases, making the transitions more sudden, the regions denoting higher speeds become more bigger across the amplitude-wavefrequency space. This means that the system's optimal conditions vary significantly with the steepness. In practical terms, this means that the system can tolerate a wider range of operating conditions as the steepness increases. The red regions denoting big speeds becomes bigger and bigger. Indeed, for a given amplitude and wavefrequency, the speed of the fish itself increases with the steepness, going up to 0.03 m/s when the steepness is at 50, for these parameters.

In order to find the best performance values for the parameters (i.e the parameters that would provide the biggest speed and displacement), we run a grid search and slowly narrow down the values to find the best ones:

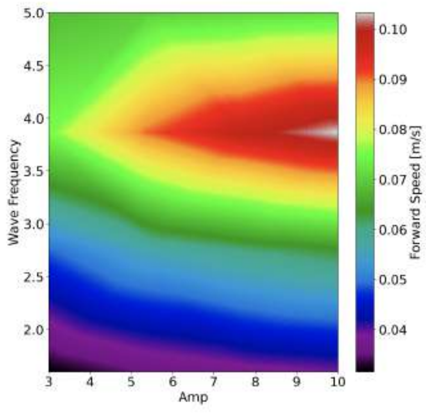


Figure 47: Speed distribution for the optimal values of the exercise 2

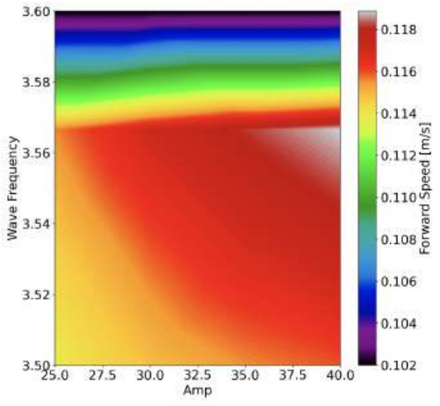


Figure 48: Speed distribution for the optimal values of the exercise 2

Here, the model was pushed to a very high amplitude, which is unusual but this allowed us to see what the highest values of speed would be. From these values it can be seen that the speed reaches a maximum of 0.118 m/s for these ranges. To further analyse the behaviour and see how far it goes in one episode, the trajectory can be plotted:

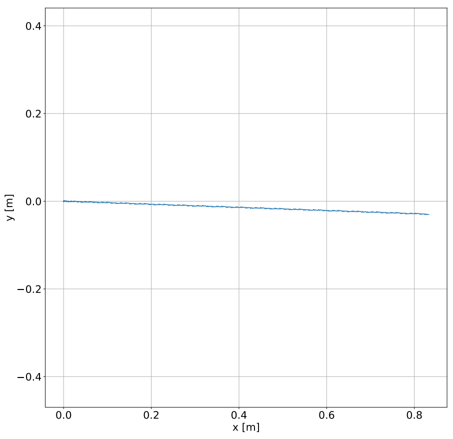


Figure 49: Trajectory for the optimal values of the exercise 2

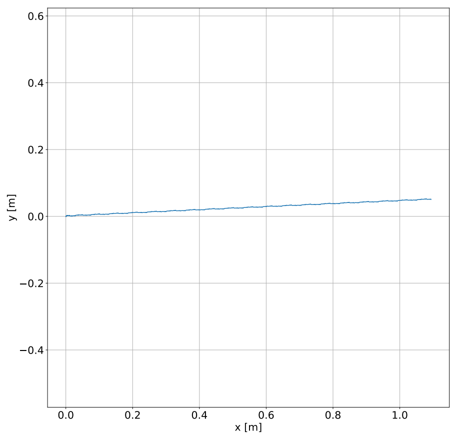


Figure 50: Trajectory for the optimal values of the exercise 2

These plots indicate that the distance travelled by the fish with the optimal parameters is almost 1.2m for a simulation time of 10s. This confirms the previous speed of 0.118m/s that we had found because $0.118\,[m/s] * 10\,[s] = 1.18\,[m] \approx 1.2m$.