Enhancing Whale Acoustic Signal Analysis through Reinforcement Learning

Abdelhaleem SAAD, Worachit KETRUNGSRI, Yosef GUEVARA

Reinforcement Learning
February 9, 2023

Table of Contents.

Introduction.

Whales produce sounds for communication, navigation, and echolocation by exhaling air, vibrating vocal cords, or making body movements. Human activity, such as underwater noise pollution from boats and ships, can negatively impact whale sounds and communication. Toulon bay shows a high level of whale activity after peaks of human activity in the morning and afternoon. In these situations' reinforcement Learning (RL) can be used in signal processing to de-noise environmental noise and identify whale sounds. The agent is a de-noising algorithm, which removes noise and reveals the whale sounds and receives rewards based on the difference between the denoised signal and an estimation.

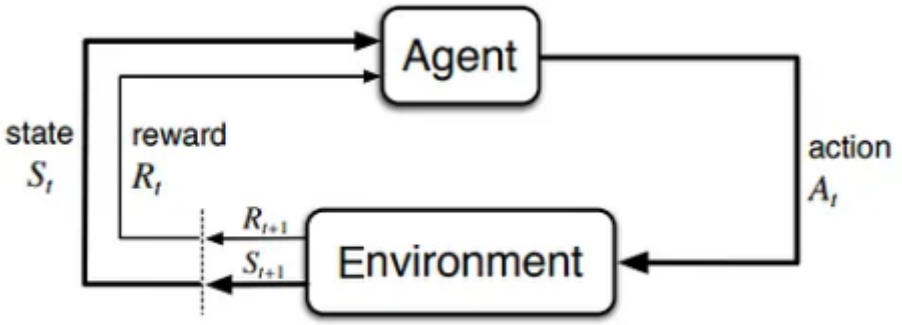

Figure 1. Planned scheme

1. The Whale moaning

Whale moans are low-frequency sounds that can range from 20 to 1,000 Hz in frequency and can last from a few seconds to several minutes. They are usually produced by baleen whales and are used for communication and navigation purposes. The exact frequency and duration of a whale moan can vary depending on the species of whale and the purpose of the moan.

Whale moans can travel long distances in the ocean, often hundreds or even thousands of kilometers, due to the properties of water as a medium for sound transmission. This long-range communication capability is particularly important for whales, as they often live in large groups and migrate over long distances.

The characteristics of whale moan, such as frequency and duration, can also provide important information about the whale's behavior and environment. For example, the frequency and duration of moans can change in response to changes in the whale's surroundings, such as the presence of other whales or the presence of underwater obstacles.

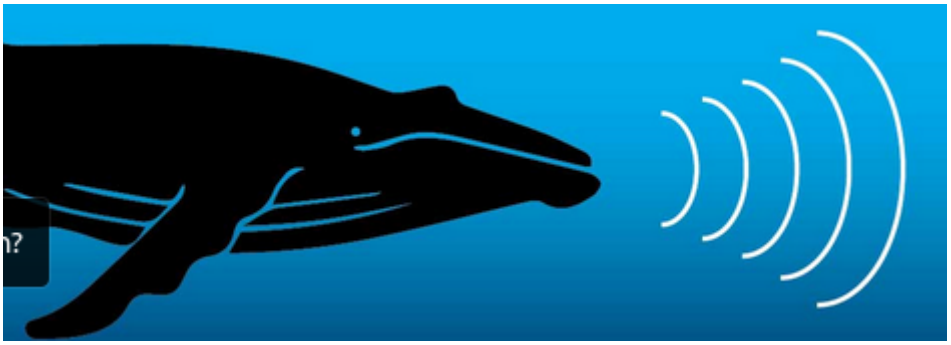

Figure 2. Whale moaning

2. Signal annotations.

Due to the environmental noise, signal annotations play a crucial role by adding labels associated with specific portions of a signal (whale moaning). For our purposes they were taken manually, using an audio editor, filtering the signal and annotating the start and end time of each pulse from a single channel and the maximum decibels from the pulse.

By using multiple hydrophones to capture whale moans, we can get valuable information about the location and behavior of whales. Hydrophones are underwater microphones that can detect and capture the sounds produced by whales. By deploying multiple hydrophones in different locations, it is possible to determine the location of the whale based on the time difference of arrival (TDOA) of the sound at each hydrophone.

For our task, 5 hydrophones with different gain were deployed. The channels 1-4 were C177 Hydrophones and the 5 channel were a SQ26 hydrophone whose signal is inverted compared to the other channels, meaning that the minimum of the recorded signal is the maximum. The hydrophones were deployed under the following scheme.

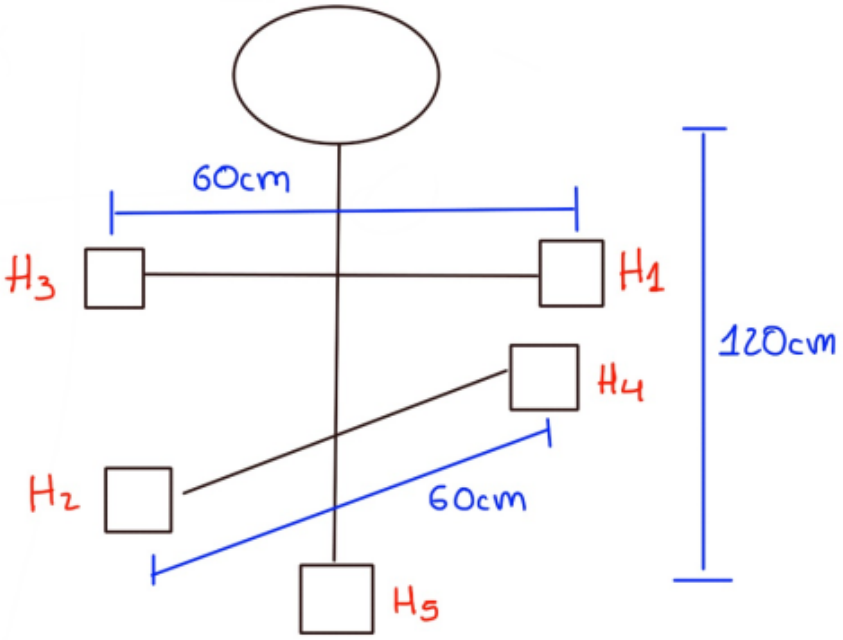

Figure 3. Hydrophones deployment schematic.

2.1. Steps to get the annotations.

The process to get the signal annotations involves 5 steps.

1. Apply the low pass filter.
2. Apply the high pass filter.
3. Amplify the signal.
4. Identify the pulses on the result signal by visual inspection.
5. Capture the start time, end time and maximum amplitude of the pulse.

   The final result must follow the nest pattern.

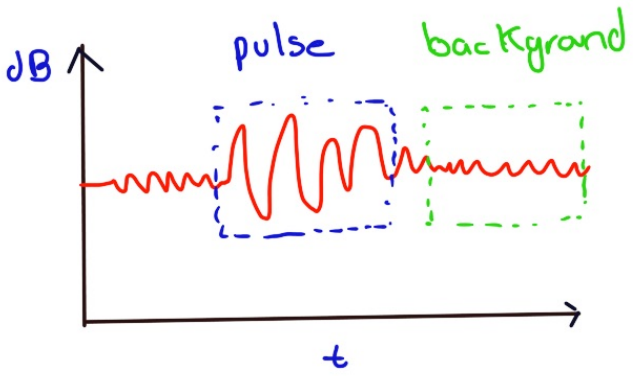

Figure 4. Expected result.

Due to time constraints, only the first channels of the20220729_040919UTC_V12.wav record.

2.1.1. Applying the low pass filter.

A low-pass filter is a signal processing tool that allows low-frequency signals to pass through while reducing or blocking high-frequency signals. It is commonly used to remove noise or isolate the low-frequency components of a signal for further analysis. For our project, we apply the following low pass filter.

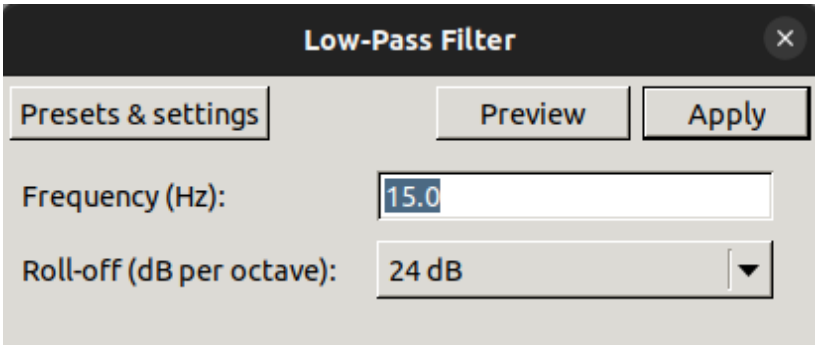

Figure 5. Low pass filter application

2.1.2. Applying the High pass filter.

A high-pass filter is a signal processing tool that allows high-frequency signals to pass through while reducing or blocking low-frequency signals. It is commonly used to remove noise or isolate the high-frequency components of a signal for further analysis. For our project.

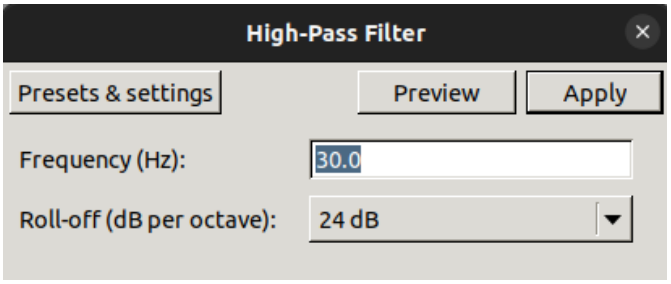

Figure 6. High pass filter application

The results show that after applying a low-pass filter to a signal emphasize the low-frequency components and reduce the high-frequency components. On the other hand, applying a high-pass filter to a signal emphasizes the high-frequency components and reduces the low-frequency components, leading to a signal with the low-frequency components removed.

2.1.3 Amplifying the signal.

After filtering the signal, we increase the magnitude of it, to help us to have an easier visual inspection to identify the pulses per channel.

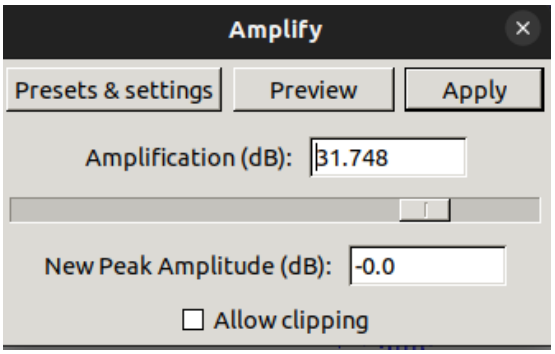

Figure 7. Signal amplification

2.1.4. Identify the signal by visual inspection.

By visually inspecting the signal, one can observe the changes in the magnitude and frequency content of the signal, to find the characteristic features of the whale sounds, such as the presence of tonal components.

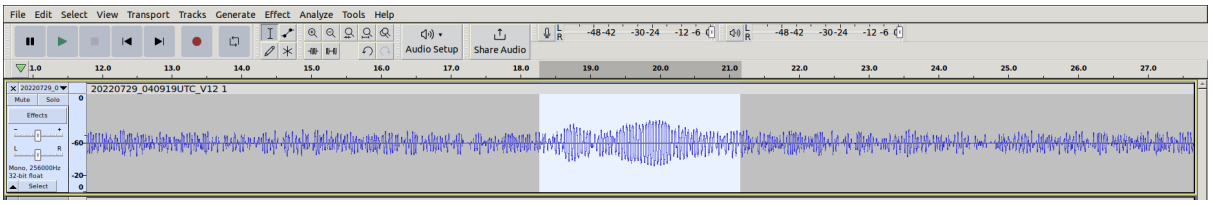

Figure 8. Filtered signal

After zooming over one of the pulse, we got that

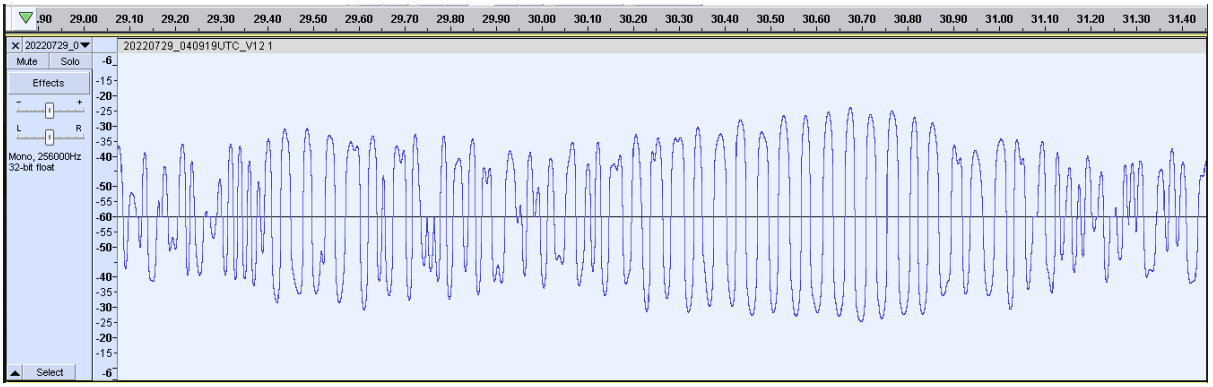

Figure 9. Zoomed signal

The idea is to get closer to a sinusoidal signal, as we can see in the following image.

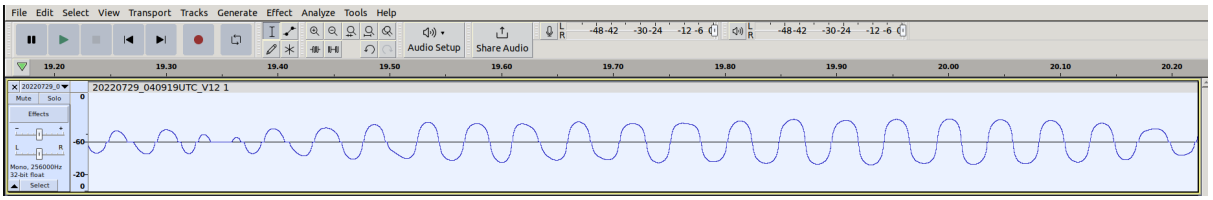

Figure 10. Sinusoidal signal

2.2 Results of applying the signal processing

After processing the signals, we were successful in locating the pulses (whale sounds) in the recordings. For each channel, we were able to identify multiple pulses with their corresponding start and end times, as well as the maximum (peak) of the signal.

Pulse #	Channel	Start Time (s)	End Time (s)	Peak (db)
1	1	19.1	20.15	-23
2	1	30.09	31.15	-23
3	1	40.25	41.25	-24
4	1	53.02	53.83	-33
5	1	64.5	65.96	-25
6	1	75.58	76.57	-26
7	1	102.7	103.55	-29
8	1	114.17	115.28	-28
9	1	140	141.1	-29.5
10	1	150.9	151.9	-29
11	1	159.7	160.5	-32
1	1	20.15	30.09	-37
2	1	31.15	40.25	-35
3	1	41.25	53.02	-35
4	1	53.83	64.5	-37
5	1	65.96	75.58	-37
6	1	76.57	102.7	-37
7	1	103.55	114.17	-40
8	1	115.28	140	-35
9	1	141.1	150.9	-38
10	1	151.9	159.7	-37
11	1	160.5	200	-38

3. Signal processing using python

As was mentioned previously, due to the noise in the records the whale vocalizations have been contaminated by unwanted sounds from human activities, such as ship traffic, industrial activities, or other sources of underwater noise. These sounds can interfere with the quality of the recording and make it difficult to accurately detect and analyze the whale vocalizations. Therefore, signal preprocessing is necessary to improve the signal quality before the application of any pulse detection agent, this can be achieve by apply a low pass filter and a high pass filter.

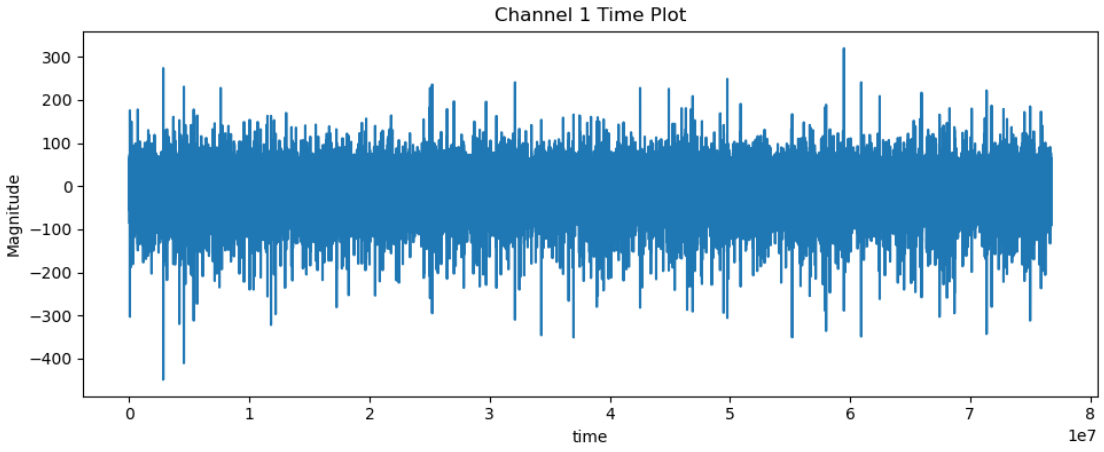

Figure 11. Signal before the preprocessing.

3.1. Energy detection

3.1.1. Mapping the frequency of the signal.

A time-frequency representation, using a continuous wavelet transform, breaks down a signal into a set of time-frequency coefficients. The magnitude of these coefficients is used to create the scalogram, which is a two-dimensional image. The scalogram will helps to identify features in the signal and visualize its dynamics over time.

By using the continuous wavelet transform (CWT) to analyze a channel, 1. We specify, the length 2000 to set the longitude of the segments of the signal that will be analyzed, using a Morlet wavelet. After that, the signal is multiplied by a Hanning window to reduce spectral leakage.

The coefficients for the channel are stored in separate variable and combined into a single list called coefs. By plotting the results, we can appreciate the distribution of energy in the signal in the time-frequency domain. The x-axis represents time, the y-axis represents frequency.

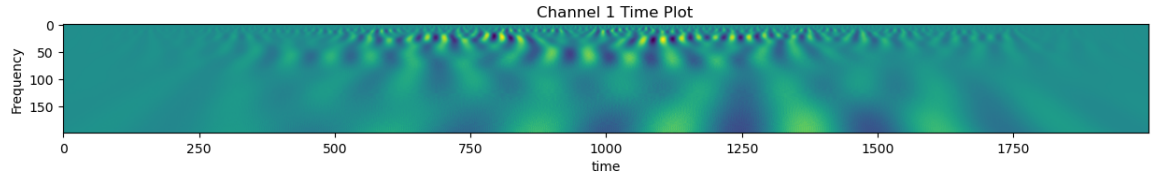

Figure 12. Signal scalogram

The scalogram image displays the magnitude of frequency components of a signal at different times. The bright, high-intensity regions in the image correspond to high-magnitude frequency components, which can be identified as the whale moans. By observing these bright regions, it is possible to determine which frequencies are most active in the signal at different times. On the other hand, dark or low-intensity regions in the image represent low-magnitude frequency components. For channel one, the image shows that the high frequencies are located between 0-50 Hz.

3.2.1. Continuous Wave transform (CWT)

Also, the energy of the signal can be calculated by squaring the coefficients obtained from the CWT and summing them up. The "energy" function takes as input the signal at the start time "t", the sample rate, and the lower and upper bounds "LH" for the frequency range to be considered. The signal is multiplied by a Hanning window to reduce spectral leakage, and then transformed using the Morley wavelet. The squared coefficients are then summed up to calculate the energy.

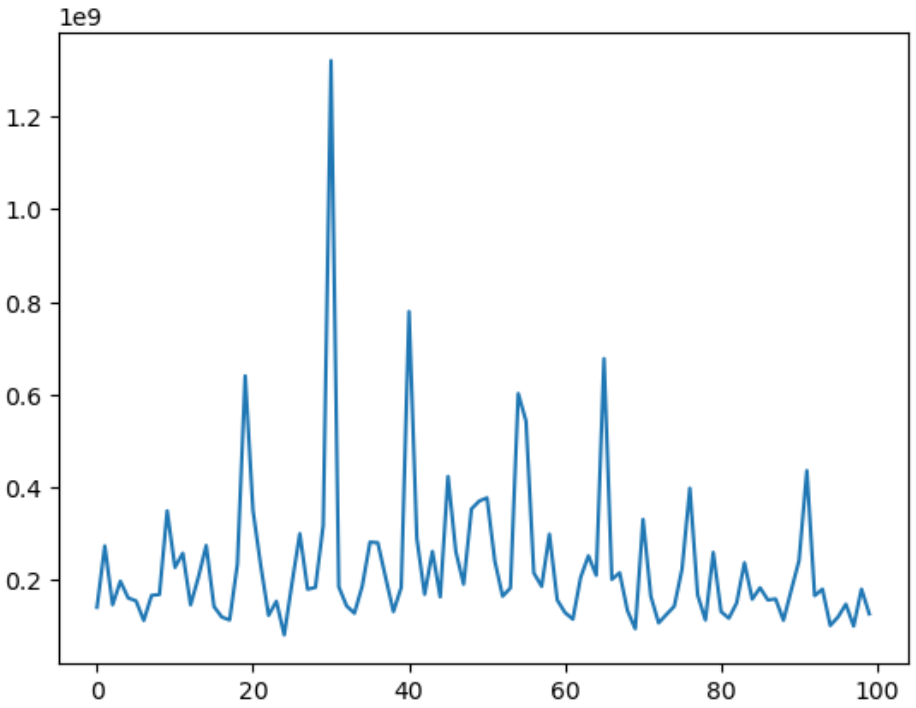

Figure 13. Signal continuous wave transforms total energy over time

The previous image displays the moments when the energy of the signals is at its highest, which can be identified as the times when the whale moaning take place. These peaks decrease with time, and the highest peaks indicate when the whale was closest to the hydrophone. Conversely, the lower peaks correspond to when the whale was farther away. This information can be combined with other methods to reconstruct the whale's trajectory over time.

4. Implementation of RL agent and the environment

Reinforcement learning can be used in audio signal enhancement, where the agent is an audio enhancement algorithm that takes actions to remove noise from the audio signal and the environment is the noisy audio signal. The reward signal is a quality metric such as signal-to-noise ratio (SNR) or perceptual quality for our purposes will be:

$$Reward = |Annotation - estimation|$$

And the agent can be defined by:

$$Agent = \begin{cases} Convolution\ with\ K_i \\ High\ pass\ Hj \\ Low\ pass\ Lp \\ Interpulse\ Interval \\ Duration\ of\ \ pulse \end{cases}$$

Reinforcement learning offers a trial-and-error based framework for optimizing signal processing algorithms and control systems. By learning from experience, the agents can improve their performance over time.

These are steps, we followed to define the environment, observation space, and action space for your reinforcement learning problem which is implementing an RL agent to find the optimal range of frequencies at which the pulse of the whale occurs.

4.1. Define the Environment.

The environment can be a simulation where the agent has access to the sound wave file to extract the useful information for the state, such as frequency information and computing energy function.

The signal-to-noise ratio from the given window that will be using from now on is computed from the energy of the pulse divided by the area of the window to normalized this energy. This can be computed from the equation

$$D_p = \frac{E_p}{A_p}$$

Where:

D_p Is the signal-to-noise ratio of the pulse given the current filter.

E_p Is the energy of the pulse given the current filter.

A_p Is the area of the window given the current filter.

```
def energyDensity(self, t, interval, channel=0):
    LH[0] = interval*20
    LH[1] = LH[0] + 19
    Ep = energy(self.signal, channel, t*self.T, self.samplerate, LH, self.T)/((LH[1]-LH[0])*self.T)
    Ea = energy(self.signal, channel, t*self.T, self.samplerate, None, self.T)

    Dp = Ep/((LH[1]-LH[0])*self.T)
    Db = (Ea-Ep)/((199-(LH[1]-LH[0]))*self.T)
    return Dp, Db
```

Figure 14. Observation space

4.2. Observation space.

Observation space can be defined as the input to the RL agent, which is the energy of the signal given the current filter. This can be represented as an array that contains this information about the energy from the current window. This energy is discretizing into 1000 state.

```
self._N_state = 1000
self.observation_space = gym.spaces.Discrete(self._N_state)
```

Figure 15. Observation space

4.3. Action space.

The action space can be defined as the set of possible actions that the RL agent can take. For this problem, the action could be the selection of a specific frequency range, which the agent believes is optimal for detecting the pulse. We defined our action space to be continuous as the frequency of the signal ranges from 0 to 199 Hz . So, we decided to divide the range of frequency into 10 intervals, each interval is assigned to an action.

The agent can choose to select one of these intervals as the optimal range of frequencies for detecting the pulse. The reward function can then evaluate the agent's performance based on its selection of frequency intervals.

```
self.action_space = gym.spaces.Discrete(10) # 10 frequency intervals
self.frequency_intervals = ["0-19", "20-39", "40-59", "60-79", "80-99.9",
                            "100-119", "120-139", "140-159", "160-179", "180-199"]
```

Figure 16. Action space

4.4. Reward model

The reward function defines the rewards the agent receives for its actions. This will guide the agent towards the desirable state and action. We thought about different ways to define our reward function and we decided to select one. Where our reward function is:The reward function defines the rewards the agent receives for its actions. This will guide the agent towards the desirable state and action. We thought about different ways to define our reward function and we decided to select one. Where our reward function is:The reward function defines the rewards the agent receives for its actions. This will guide the agent towards the desirable state and action. We thought about different ways to define our reward function and we decided to select one. Where our reward function is:

$$reward(s, a) = D_p$$

The reward model is the signal-to-noise ratio, to measure how good the current filter is.

4.5. Environment Implementation

```
def reset(self):
    # randomly select annotation idx from the annotation
    self.annotation_idx = np.random.randint(0, len(self.annotation)-1)
    # Randomly generate a new state for the environment
    t = self.annotation["Start Time (s)"][self.annotation_idx]
    Dp, Db = self.energyDensity(t, self.action_space.sample())
    self._state = int(np.tanh(self.energy_scale*Dp/Db)*self._N_state)
    return self._state
```

Figure 17. Function to compute reset the environment.

For every step, the signal is randomly selected from the given annotation and the environment will be done if the change in the signal-to-noise is less than the tolerance

```python
def step(self, action):
    # frequency_interval = self._frequency_intervals[action]
    # Calculate the change in energy caused by the selected frequency interval
    t = self.annotation["Start Time (s)"][self.annotation_idx]
    Dp, Db = self.energyDensity(t, action)
    next_state = int(np.tanh(self.energy_scale*Dp/Db)*self._N_state)
    # calculate the reward from the change in energy
    energy_change = (next_state - self._state)/self._N_state

    # Calculate the reward based on the change in energy
    reward = 1 if energy_change > 0 else -0.25
    # if the energy is change less than tolerance then change done is true
    done = False if np.abs(energy_change) > self.tol else True
    self._state = next_state
    # Return the new state, reward, and done flag
    return self._state, reward, done, {}
```

Figure 18. Function to compute the next step of the environment.

- First, we compute the current signal-to-noise ratio using the action (the lower and upper bound of the filter).

- Then, the next state is normalized and discretize using the **Tanh** function (since the input to this function is only positive real number, the output of the **Tanh** function will be between 0 and 1) after that we discretize this into N state (_N_state=1000).

- Then, we compare the energy from the previous time step using with the current time step and get the energy change, which will be used to compute the reward for taking the action.

- Lastly, if the energy change is less than the tolerance which indicate that the agent fully learn the filter for this energy, the episode is done.

4.6. Training the RL agent

After defining the environment, observation space, action space, rewards, and state space, our next step is to train the RL agent to select the optimal range of frequencies.

4.6.1 Q-learning Algorithm

We decided to use Q-Learning algorithm is estimated the optimal action-value function, which assigns a value to each action at each state, and represents the expected reward of taking a certain action at a certain state and then following the optimal policy.

In Q-learning, the action-value function is represented by a table, and the algorithm updates the table based on the observed rewards and the estimated values of the next states. The algorithm starts with an initial estimate of the action-value function and updates it over time as it explores the state-action space. The updates are performed using the Q-Learning rule, which states that the value of a state-action pair is equal to the expected reward of the current state plus the maximum expected value of the next state. The Q-Learning algorithm updates the action-value function, Q(s,a), for a given state, s, and action, a, using the following formula:

$$Q(s,a) = Q(s,a) + \alpha\,(r + \gamma\ max_a(Q(s',a')) - Q(s,a))$$

where:

- $Q(s,a)$ Is the current estimate of the action-value function for state s and action a
- α Is the learning rate, which determines the step size of the update
- r Is the reward received after taking action a in state s
- γ Is the discount factor, which determines the importance of future rewards relative to immediate rewards
- $max_a Q(s', a')$ Is the maximum estimated action-value function over all possible actions a' in the next state, s'

4.6.2 Epsilon-greedy policy

Epsilon-greedy policy is a common exploration strategy that is used in Q-Learning. The policy defines a probability of selecting the greedy action, which is the action with the highest estimated value, and a probability of selecting a random action, which allows the algorithm to explore the state-action space. The epsilon parameter controls the balance between exploration and exploitation, and it typically decreases over time.

Q-Learning with an epsilon-greedy policy is a suitable choice for selecting the optimal range of frequencies for whale pulse detection because it can learn the optimal policy based on the observed reward. Additionally, the epsilon-greedy policy allows the algorithm to explore the state-action space and find the optimal range of frequencies, even if the initial estimates are not accurate.

5. Results

After implementing the algorithm we mentioned earlier, we plotted the maximum reward obtained in each episode. We conducted 100 episodes in this experiment and recorded the rewards.

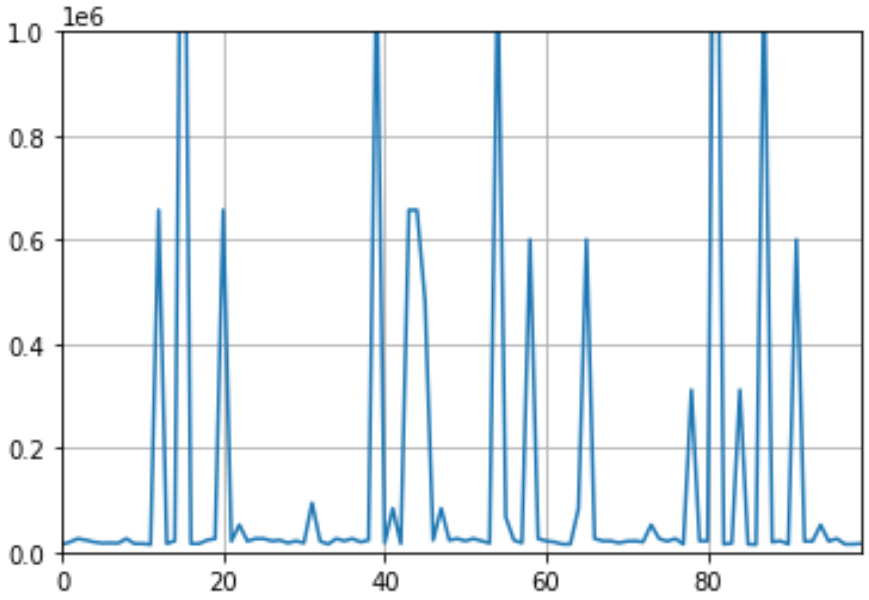

Figure 19. Reward after training for 100 episodes

It's possible that the oscillations in the reward function were caused by the fixed range of frequencies that the agent was allowed to choose from. If the frequency interval was too narrow, the agent may have struggled to identify the optimal frequency range for detecting whale vocalizations, resulting in suboptimal decision-making and a fluctuating reward signal.

To address this issue, a new approach was developed to create an adaptable frequency window that could be adjusted based on the feedback provided by the environment. This was achieved using a function that could dynamically expand or contract the frequency range, providing the agent with a more flexible and adaptive decision-making framework. By allowing the agent to select from a broader range of frequencies, this new approach aimed to enhance the agent's ability to accurately identify the optimal frequency range and ultimately improve its performance.

$$a =< L, h >$$

Where:

L is the lower bound of the

h is the high of the frequency window

The lower bound and upper bound can be computed using

$$< lower, upper >=< L, max(L + h, F_{max}) >$$

where:

F_{max} is the possible maximum frequency, which is 199 in this data

After we applied this method and increase the training epoch to 500, this is the result that we got

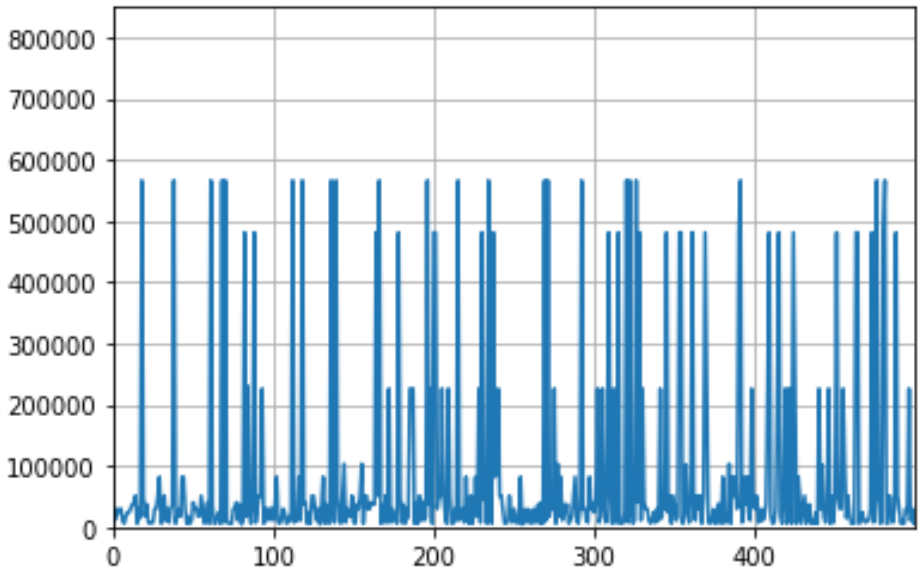

Figure 20. Reward of the modified action environment after training for 500 episodes

The results of the latest iteration of the reinforcement learning agent show a slight improvement over the previous version, which could be attributed to two factors: an increase in training time and a modified reward function. Despite these enhancements, however, the agent was still unable to accurately identify the correct frequency window. This may be due in part to the reward function, which was observed to oscillate and fail to steadily increase, indicating that the agent's decision-making policy was not converging towards an optimal solution.

6. Conclusion

This project aimed to use reinforcement learning to develop an agent capable of identifying the optimal frequency range for detecting whale vocalizations. The agent was trained using the Q-learning algorithm to learn the relationship between the signal-to-noise ratio (SNR) and the corresponding frequency interval. While the agent was able to accurately predict the correct frequency window in some instances, the overall results were not consistently effective in identifying the correct interval.

There are several factors that could have contributed to the agent's limited performance. These include insufficient training time, limited exploration of the environment, a sparse reward function, or a lack of information about the system's state, such as the SNR. If the SNR varied significantly over time or was difficult to estimate accurately, the agent may have struggled to identify the optimal frequency window.

Another challenge was that the frequency interval that maximizes the reward depends on the current state of the environment. This made it difficult for the agent to generalize effectively to new environments, resulting in narrow results. While the Q-learning algorithm is a robust and widely-used approach in reinforcement learning, it may require significant time and exploration to learn effectively and generalize to new environments.

Overall, Q-learning is a robust and effective algorithm that can be used for a wide range of reinforcement learning problems, including the problem of identifying the optimal frequency range for detecting whale vocalizations.

7. Code implementation.

- whale_final.ipynb - Colaboratory (google.com)