A REPORT

ON

ML APPLICATIONS IN UNDERWATER CHANNEL MODEL

By

Name of the Student	ID number
RISHABH PATRA	2018AAPS0348G

AT

MARITIME RESEARCH CENTRE, PUNE

A Practice School 1 Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE, PILANI

(JUNE, 2020)

A REPORT

ON

ML APPLICATION IN UNDERWATER CHANNEL MODEL

BY

Name of the Student	ID No	Discipline
RISHABH PATRA	2018AAPS0348G	BE(Hons) Electronics and communications

Prepared in partial fulfilment of the Practice School-I Course Nos.
BITS C221/BITS C231/BITS C241

AT

MARITIME RESEARCH CENTRE, PUNE

A Practice School 1 Station of

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI
(JUNE, 2020)

ACKNOWLEDGEMENTS -

We would like to extend our heartfelt gratitude to all those responsible for providing us interns with the opportunity to contribute to the work done at MRC. We would like to take this opportunity to thank the several entities and personalities responsible for providing us with such a platform.

Firstly we would like to thank Dr.(Cdr) Arnab Das, Director of MRC to have facilitated the collaboration of BITS Pilani University and MRC and to have been willing to train interns. We are forever indebted for this learning opportunity.

We would also like to thank Sridhar Prabhuraman for taking time out of his busy schedule to guide us through our projects, and his timely insights which makes working not only easier, but fun as well, considering the new environment we are not acquainted to.

Finally we would like to thank Dr. Anurag Nishad for his help and guidance throughout the course in terms of clarifications provided, and his availability throughout the day for any administrative or research queries.

We would finally like to thank everyone involved directly or indirectly to provide us with this opportunity

BIRLA INSTITUTE OF TECHNOLOGY AND SCIENCE PILANI
(RAJASTHAN)
Practice School Division

Station - MRC Pune **Center -** Pune
Duration - May 18 - June 18 (6 weeks) **Date of Start -** May 18, 2020
Date of Submission -
Title of the Project -
ID No. /Name(s)/Discipline(s) of the student(s):
Rishabh Patra - 2018AAPS0348G

Name(s) and designation(s) of the expert(s):
Dr. (Cdr) Arnab Das, Director, MRC
Sridhar Prabhuraman

Name(s) of PS Faculty:
Dr. Anurag Nishad

Key words: Underwater channel model, Helmholtz equation, Parabolic equation, Split step Pade solution, PyRAM, Machine Learning

Project Areas: Machine Learning, Computational modelling, Data Mining

Abstract:
The underwater channel model is a mathematical model to analyse the propagation of acoustic waves and the various interactions they go through in the underwater regions. The Range dependent Acoustic Model (RAM) based on the Parabolic equation (PE) is the state of the art to model these interactions. However, the PE-RAM is based on a numerical solution, prone to manual input errors, and huge computational time, which can be problematic while being used in a real time setting. This report looks at the application of machine learning and deep learning techniques to reduce the computational time, as well as reduce the dependence on manual inputs, while aiming to maintain the accuracy, all tailored for use in the Indian Ocean Region

Signature of PS Students **Signature of PS Faculty**
Date: 27-06-2018 **Date:**

TABLE OF CONTENTS

Introduction

The Parabolic Equation method[1] is useful in modelling range dependent wave propagation, including sound propagation in the ocean. Since ocean environments are typically large compared to the wavelength of sound being transmitted, it is imperative to come up with efficient solutions to the Parabolic equation. From these solutions, we obtain a figure of merit, the Transmission Loss, which is a very important figure in communications, be it underwater or not. The state of the art currently is a Range dependent Acoustic Model (RAM) based on the split step Pade solution[2] to the Parabolic equation. Though fairly accurate, it comes with its own errors and shortcomings, which we address in this report.

Another thing to note here would be that this RAM was built for the Western oceans, and makes some implicit assumptions about the salinity and temperature, both of which may not hold true for the Indian Ocean Region (IOR). But since the IOR is located at such a key position, from military, strategic and trade perspectives, it is imperative to find an efficient and accurate way to calculate the Transmission Loss for ships in this region.

Finally, we talk about the computational time required for the PE-RAM model. Being a numerical technique, it takes huge computational time, that scales linearly with the number of ships, which may pose problems in real time usage. Due to the recent advances in Machine Learning and Deep Learning techniques, such computations may be replaced with function approximations. This does take intense computational time, and a lot of data to learn the function mappings, but then it is immensely faster once it has learnt the mapping. Moreover, this learning has to be performed only once. This report looks at how these advancements fare compared to the SOTA, and the advantages and shortcomings of using this thereof.

Related Works

The following section goes through the existing works, and a short derivation of the Parabolic equation, from the Wave equation, and the Range dependent acoustic models currently used as SOTA.

a) The Wave equation and the parabolic equation

The wave equation is a very fundamental equation derived from the equations of state, continuity and motion. The exact form of this equation may vary based on the forces accounted for, and the accuracy required , the most general form accounting for the gravitational and the rotational effects as well[3].

Formulations of derivations of acoustic propagation models start with the three-dimensional, time dependent wave equation. Depending on the assumptions and applications, the exact form may again be altered considerably. Mostly, a simplified linear hyperbolic, second order, time dependent, partial differential equation is used -

$$\nabla^2\Phi = \frac{1}{c^2}\frac{\partial^2\Phi}{\partial t^2}$$

(Eq - 1)

(Relates the second order of the laplacian to the second order rate of change of potential of the wave)

Subsequent simplifications include assuming a harmonic solution to this wave equation, giving rise to what is referred to as the Helmholtz equation -

$$\Phi = \phi\, e^{-i\omega t}$$

(The approximation) (Eq - 2)

$$\nabla^2\phi + k^2\phi = 0$$

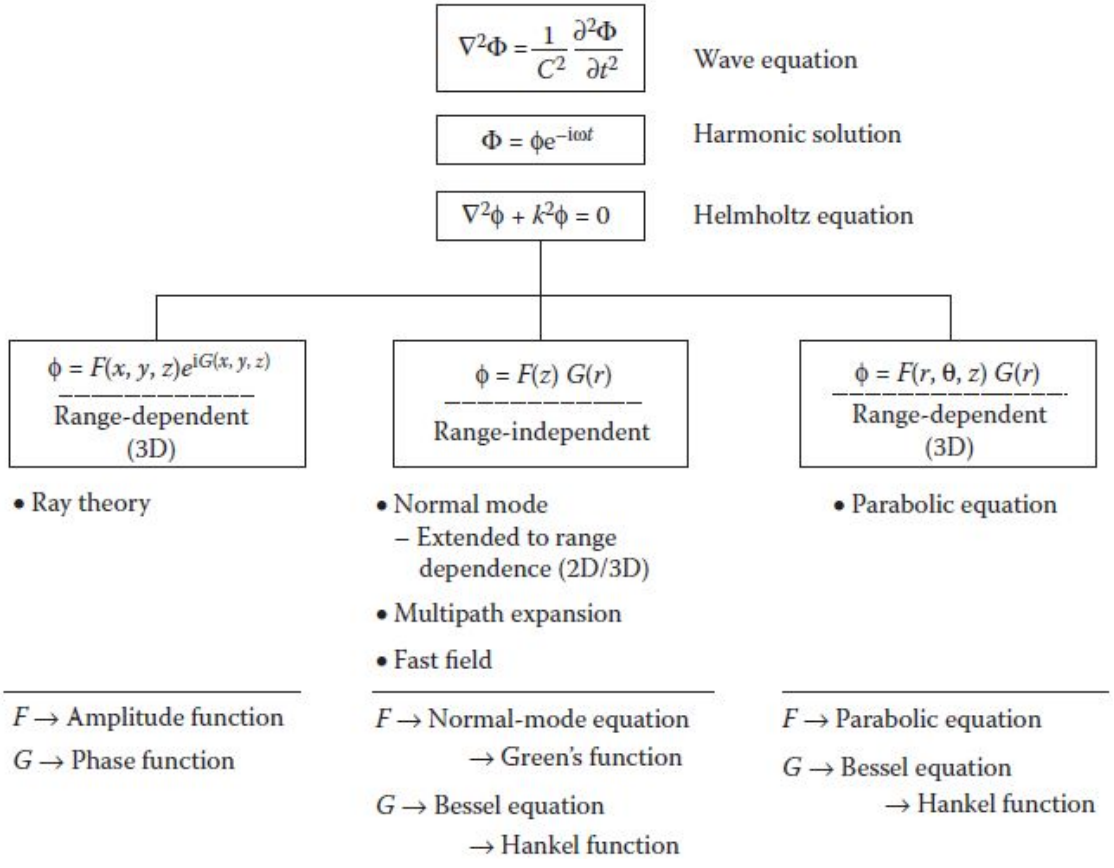
(The Helmholtz equation) (Eq - 3)

There does exist many solutions to the Helmholtz equation as well, depending on the geometrical assumptions made, and the type of solutions chosen for the potential function.

Another really important aspect to note is that this potential function normally represents the acoustic field pressure. This leads us to the practical figure of merit, the transmission loss, calculated as follows-

$$TL = 10 \log_{10}[\phi^2]^{-1} = -20 \log_{10} |\phi| \quad \text{(Eq - 4)}$$

The various commonly used solutions to the Helmholtz equation and the assumptions may be summed up as follows[4][5][6] -



The Indian ocean region is a very versatile region, with salinity and depths rapidly varying. This is why range independent models can't be used for the IOR. The ray theory approach assumes that the sediments and other objects

floating in the ocean are very small compared to the wavelength of acoustic signal - limiting our transmission wavelengths.

Hence, the most suitable form is the range dependent parabolic equation, to calculate the variation of the acoustic pressure field wrt the three cylindrical coordinates. Furthermore, the dependence on the azimuthal angle can be ignored by changing the frame of reference (specifically, tilting it as much as the azimuthal angle).
The parabolic equation then can be derived by making the following substitution -

$$\phi = \psi(r,z) \cdot S(r)$$

Which upon substituting into Eq-3 gives rise to the following -

$$\Psi\left[\frac{\partial^2 S}{\partial r^2} + \frac{1}{r}\frac{\partial S}{\partial r}\right] + S\left[\frac{\partial^2 \Psi}{\partial r^2} + \frac{\partial^2 \Psi}{\partial z^2} + \left(\frac{1}{r} + \frac{2}{S}\frac{\partial S}{\partial r}\right)\frac{\partial \Psi}{\partial r} + k_0^2 n^2 \Psi\right] = 0$$

(Eq-5)

Now, separating the two variables, and making some assumptions (we won't go into the derivations here, one might find them in Underwater acoustic modelling and simulation, Paul C. Etter[7]). Finally, we apply a paraxial approximation -

$$\frac{\partial^2 \Psi}{\partial r^2} \ll 2k_0 \frac{\partial \Psi}{\partial r}$$ (Eq-6)

Giving rise to the following -

$$\frac{\partial^2 \Psi}{\partial z^2} + 2i\, k_0 \frac{\partial \Psi}{\partial r} + k_0^2 (n^2 - 1)\Psi = 0$$ (Eq-7)

This can be factored into two parts -

$$\left[\frac{\partial}{\partial r} + i\,k_0 - i\sqrt{k_0^2 + k_0^2(n^2-1) + \frac{\partial^2}{\partial z^2}}\right]\left[\frac{\partial}{\partial r} + i\,k_0 + i\sqrt{k_0^2 + k_0^2(n^2-1) + \frac{\partial^2}{\partial z^2}}\right]\Psi = 0$$

(Eq-8)

The left part represents the incoming wave, and the right part represents the outgoing wave. For further sections, we consider only the outgoing wave, and build up methods to solve them, numerically.

b) Split step Pade solution and the PE-RAM model

In this section, we look at the state of the art numerical solution to the 2D parabolic equation. The split step Pade solution[2] is arrived at by applying a pade approximation to approximate an exponential function -

$$\frac{\partial^2 p}{\partial r^2}+\rho\frac{\partial}{\partial z}\rho^{-1}\frac{\partial p}{\partial z}+k^2 p=0,$$

(Eq - 9)

(An equivalent representation of the PE equation, p represents the acoustic field pressure)

Factoring and keeping the outgoing factor, we obtain -

$$\frac{\partial p}{\partial r}=ik_0\sqrt{1+X}p,$$

(Eq-10)

Dropping the exponential terms from p -

$$\frac{\partial p}{\partial r}=ik_0(-1+\sqrt{1+X})p.$$

(Eq-11)

Solving the above equation analytically before applying the Pade approximation -

$$p(r+\Delta r)=\exp[i\sigma(-1+\sqrt{1+X})]p(r)$$

(Eq-12) where $\sigma=k_0\Delta r.$

Now the Pade approximation -

$$\exp[i\sigma(-1+\sqrt{1+X})]\cong 1+\sum_{j=1}^{n}\frac{a_{j,n}X}{1+b_{j,n}X}$$

(Eq-13)

(a, b can be obtained from Taylor expansions)

Substituting this in the above equation, we obtain -

$$p(r+\Delta r)=p(r)+\sum_{j=1}^{n} a_{j,n}(1+b_{j,n}X)^{-1}Xp(r). \quad \text{(Eq-14)}$$

Which is referred to the official split step Pade solution. Notice that since we perform a summation, it can be parallelised across multiple processors.

There are multiple implementations available, in MATLAB, C, and in python(referred to as PyRAM)

c) PyRAM implementation

PyRAM, as mentioned above, is an implementation of the Range dependent acoustic model based on the Split step Pade solution to the Parabolic equation. Some things to note about this implementation are as follows -

- Inputs - sound speed values, absorption, bathymetry, sea state, and bottom composition (informally stated)
- Hyperparameters - step size for range and depth, number of pade terms to include
- Output - transmission loss as a function of range
- Computations performed - Form a tridiagonal matrix based on the bathymetry and the sound speed values (based on the pade solution), solve this tridiagonal matrix for each range step, to compute the transmission loss at different range steps.

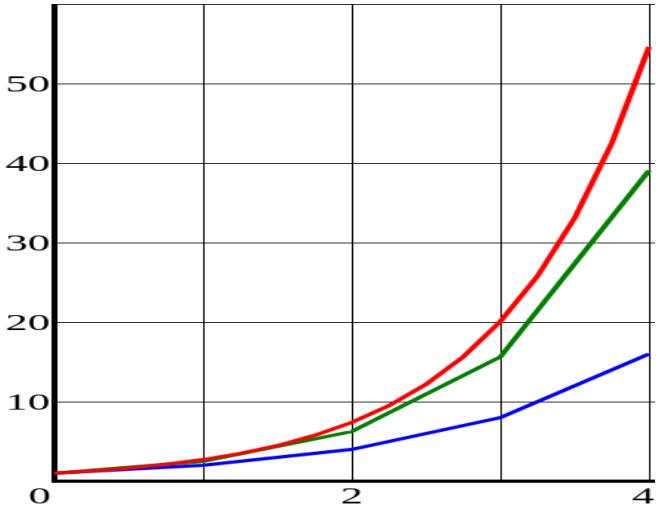
The computation involves solving a tridiagonal matrix for each range step, a computationally intensive task. Moreover, the hyperparameters mentioned need to be carefully selected. For example - the number of Pade terms - generally the accuracy will increase as this goes up, but so does the computation required. Hence, the user has to keep in mind the tradeoff between accuracy and computation in real time. The inputs are stated formally below -

```
freq: Frequency (Hz).
zs: Source depth (m).
zr: Receiver depth (m).
z_ss: Water sound speed profile depths (m), 1D array.
rp_ss: Water sound speed profile update ranges (m), 1D array.
cw: Water sound speed values (m/s),
    2D array, dimensions z_ss.size by rp_ss.size.
```

```
z_sb: Seabed parameter profile depths (m), 1D array.
rp_sb: Seabed parameter update ranges (m), 1D array.
cb: Seabed sound speed values (m/s),
    2D array, dimensions z_sb.size by rp_sb.size.
rhob: Seabed density values (g/cm3), same dimensions as cb
attn: Seabed attenuation values (dB/wavelength), same dimensions as cb
rbzb: Bathymetry (m), Numpy 2D array with columns of ranges and depths
```

Moreover, this split step Pade solution is a numerical technique. Numerical techniques, though extremely powerful, as they let us solve differential equations algebraically, they are internally prone to errors, 2 of which we shall highlight here.

First - The dependence on manual parameters, namely the step size. Eq-14 showcases the differential equation converted to an algebraic form, with Δr as the step size, to be chosen manually. Then, this equation is solved at step sizes of Δr and interpolated between the 2 steps r and $r + \Delta r$. A problem which immediately strikes one's mind, is the selection of this step size, the smaller the better, since the actual solution would be the limiting case where $\Delta r \rightarrow 0$. But the computational time scales linearly with the number of steps (inversely with the step size). But choosing a larger step size may fail to capture the entire dynamics, as shown by the figure below.

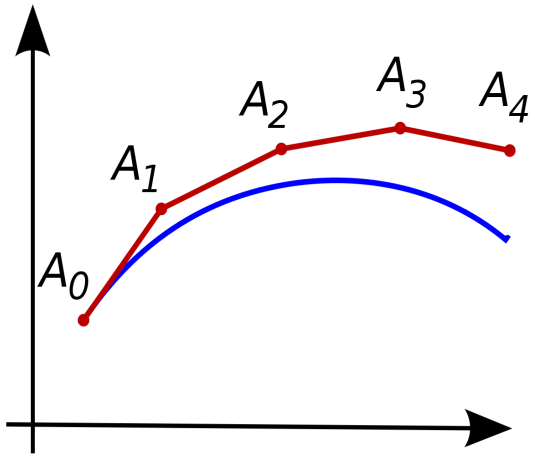


(Fig-1)

Above, the red line represents the actual solution (to a simple problem, not the parabolic equation). The green line with an arbitrary small step size, and the blue line with an even larger step size.

Hence, choosing a step size is a tradeoff between accuracy and computational time, which may be a problematic and tough decision while in use in real time settings.

Second - Numerical techniques are prone to a problem called accumulation of errors. This is best highlighted by the figure below -



(Fig-2)

Since the chosen Δr is finite, the first step solution A_1 differs from the original solution, by say δ_1. Now in the calculation of A_2 from A_1, the error of δ_1 is taken into account, leading to an error of say, δ_2. Now, the error of calculation of A_3, say δ_3 takes into consideration both δ_1 and δ_2. Same for A_4. In essence, the errors in calculation of steps keep accumulating, and may turn out to be huge if the number of step sizes are large.

Proposed Solution

With the advances in computing technology and power, machine learning and deep learning techniques have become widely popular in learning and approximating complex functions. We propose to use such a technique to approximate the Transmission loss from the transmitter to the receiver.

Machine Learning[8] is broadly defined as the study of algorithms that learn through experience. This experience is largely referred to as the train data, and this is collected, and patterns learnt from it, in order to make predictions or decisions later on, perhaps in real time usage.

A standard model has 3 components -

- Training - Given some input data, and the corresponding labels, the mathematical function is constructed, mapping from the inputs to the outputs. This may require several iterations of combing through the dataset, and updates, hence is one of the most time consuming process
- Loss computation and optimisation - The objective of training is to build a model that can minimize a given loss. The loss can be defined as the dissimilarity, or inaccuracy of the function mapping learnt. This then can be restated as an optimization problem, find the function mapping so as to reduce the loss. Using numerical nonlinear optimization techniques[9], we may arrive at the global minima of the loss function space, signifying the maximum achievable accuracy has been achieved by the best possible function mapping
- Prediction - This is the last and final step, which takes place after training and loss optimization has been completed. Prediction makes use of function mapping learnt, to predict (or rather approximate) the value of the label in case real time usage, given input data it has not seen before, but presumably, having the same distribution as the data it has been trained on.

The training step, as mentioned, is the most computationally intensive step in this case. It takes several iterations to effectively learn the function mapping

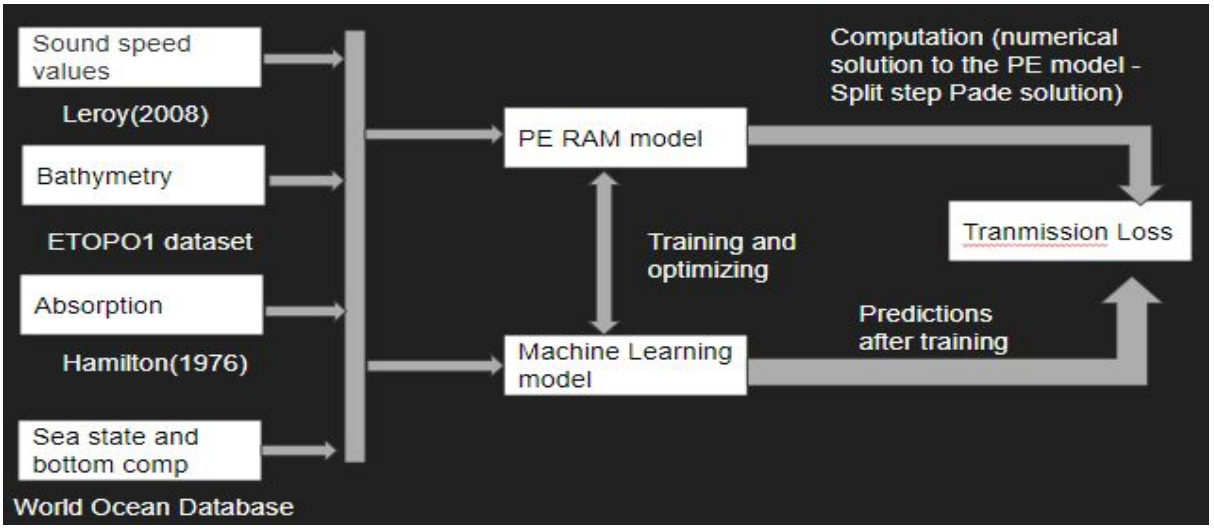
(1000-2000 is not uncommon, depending on the complexity of the space). The only advantage being that this has to be performed only once. In all subsequent real time usage, the prediction can be done almost instantly (Will be highlighted later in the results section).

We propose using a Neural Network for approximating this transmission loss. Furthermore, we propose a system where the user needs to input only the sender and receiver latitude and longitude. Computer programs then comb through the relevant databases to collect the relevant bathymetry, bottom composition, and sound speed profiles for the given coordinates, which then are sent as inputs to the model, and out comes the transmission loss.

Methodology

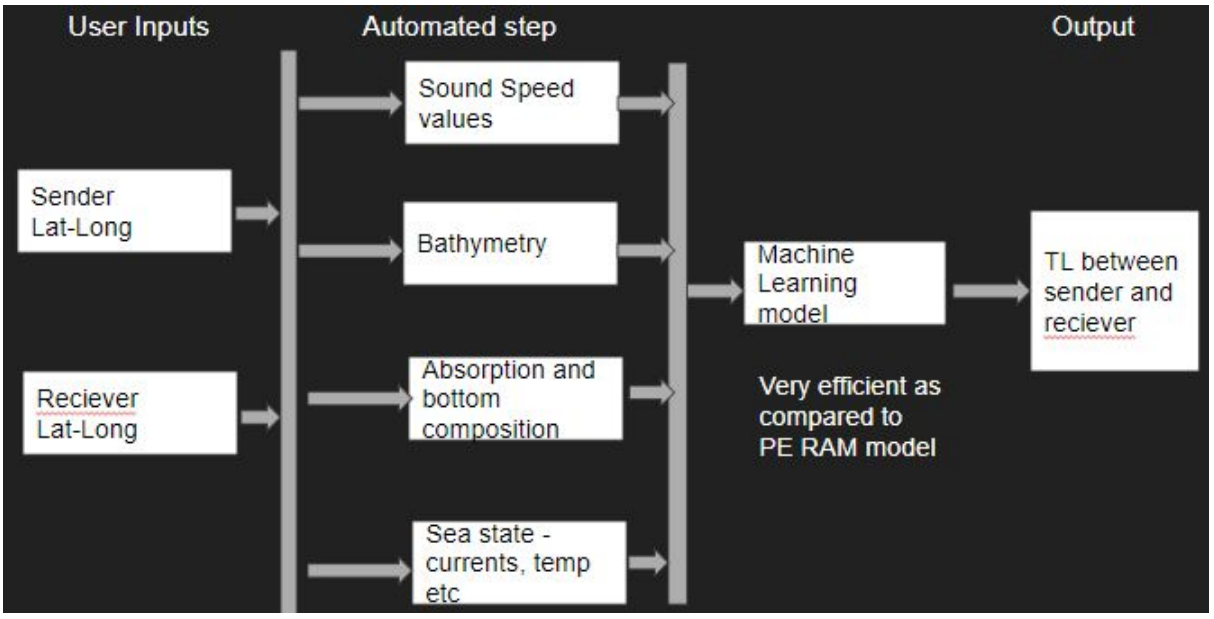
We used AIS data to create a list of sender receiver pairs, whose distance was less than 200km, since the transmission loss beyond this distance is too much, hence the purpose of transmission is defeated. These pairs of senders and receivers signify that we consider actual ships present in the seas for the purpose for calculating the transmission loss. A total of 145000 pairs of senders and receivers were extracted from this data. The depths chosen were 15m for both the senders and the receivers, since this is the average draught of the ship. We use the sound speed equation[10] to calculate the values of the sound speed profile, taking the temperature and salinity data from April - June for the IOR. The bathymetry data is obtained from the ETOPO1 dataset[11], the absorption and attenuation parameters from Hamilton[12] and the sea state and bottom composition from World ocean Database[13]. To reduce any discrepancies, we use the same inputs to train the ML model as the PyRAM takes.

These 145000 pairs are first fed into the PyRAM, and the transmission loss is calculated between them, and is used as a baseline. These pairs are divided into 3 groups - 60% used for training the model, 20% for validation after each training iteration, to check for overfitting errors, and 20% for testing when the model is finally trained. As a flowchart, the following highlights the key steps in the process



(Fig-3)

The final deliverable of the project, including the automation step can be incorporated as follows -



(Fig-4)

Moving on, we use a Neural network as a model for learning this function approximation. This stems from the fact that deep neural networks are good at complex function approximation[14][17], even with little to no feature engineering[14], since the neural network can learn different complex features on its own as it goes deeper. We compare 3 different architectures, a 4-layered network, a 5-layered network, and a 6-layered network, with different neurons per layer (which was selected manually after several iterations of training and

testing). The architectures and the neurons per layer will be shown in a figure below.

A fully connected layer is a multidimensional linear layer, that given the inputs from the previous layer (for the first one being the inputs) and a weight matrix and a bias vector, performs the following operation -

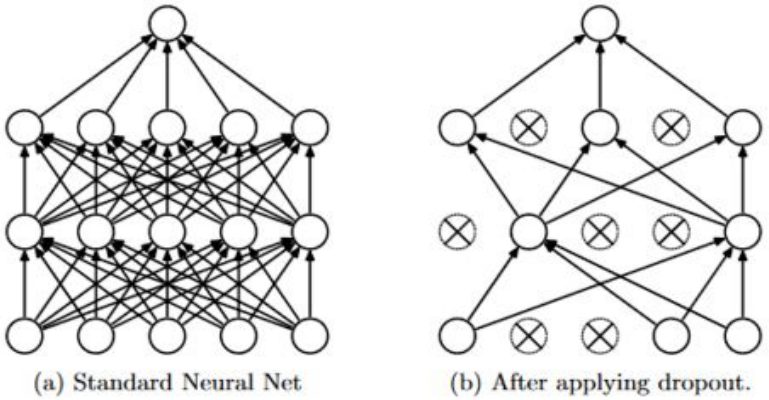
$$Z_i = W_{(i-1,\,i)}A_{i-1} + B_i \quad \text{(Eq-15)}$$

Where A_{i-1} is the input to the i^{th} layer from the $(i-1)^{th}$ layer, the matrix $W_{(i-1,\,i)}$ is the matrix connecting the i^{th} and the $(i-1)^{th}$ layer, and B_i being the bias vector. This is a matrix multiplication, which is less computationally intensive than performing a matrix decomposition, hence promises to lead to improvement in computational time.

The other facts to note are that we use the Rectified Linear Unit (ReLU) non-linearity. This comes with the assumption that any nonlinear function can be approximated by several combinations of linear layers and ReLU units[15]. The ReLU unit can be described as follows -

$$ReLU(x) = max(0,\, x) \quad \text{(Eq-16)}$$

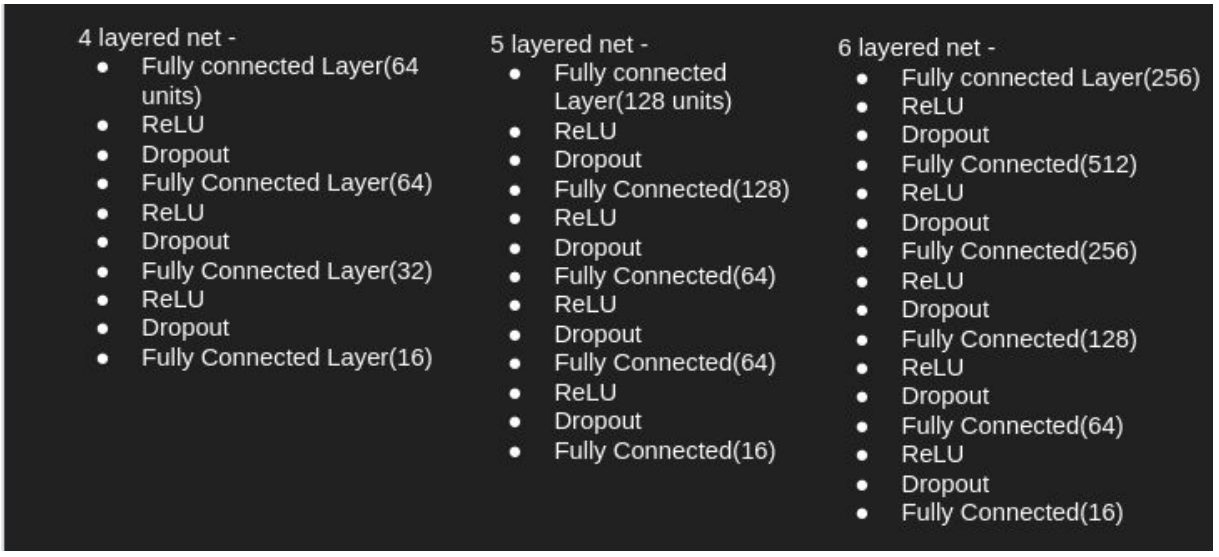
Another important fact is that we use dropout to reduce overfitting and heavy reliance on one neuron[16]. This works by randomly switching off neurons in a given layer with a probability p (here p = 0.25 is chosen).



(a) Standard Neural Net (b) After applying dropout.

(Fig-5)

The specifications of the architectures used are as follows -



(Fig-6)

The loss function used is the Mean Squared Error Loss (MSE). The mean absolute error loss (MAE) is also a suitable candidate, but the MSE loss penalises off target predictions more than the MAE, hence it was a suitable choice.

$$MSE = \Sigma_i (y_i - \hat{y_i})^2 \quad \text{(Eq-17)}$$

$$MAE = \Sigma_i |y_i - \hat{y_i}|^2 \quad \text{(Eq-18)}$$

where y_i represents the actual label and $\hat{y_i}$ represents the predicted labels.

We then use the Stochastic Gradient Descent (SGD) algorithm[9] to update the parameters, so as to reduce the loss. An SGD update is as follows -

$$\theta \leftarrow \theta - \alpha \nabla_\theta L(y, \theta) \quad \text{(Eq-19)}$$

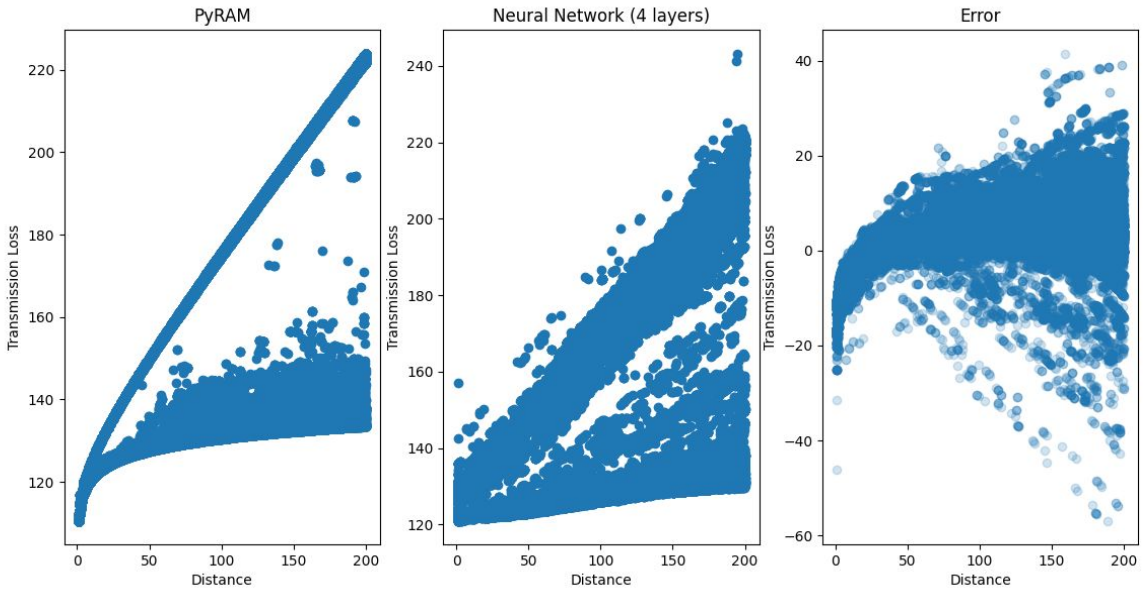
Where θ are the network parameters, α is the learning rate (usually set to 0.001) and $L(y, \theta)$ is the loss function to be minimised.

All these put together form our neural network. All the three architectures are trained for a total of 150 iterations, and their MSE and MAE values are analysed in the next section.

Experimental Results

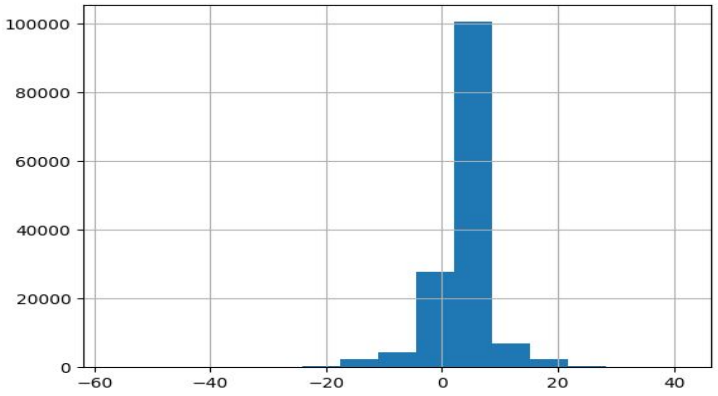
After training the neural network of all three architectures for 150 iterations, and using MSE loss with SGD updates, the following are the actual TL from the PyRAM, the TL calculated by the neural network, and the errors plotted against distance -

4-layered Neural network -



(Fig-7)

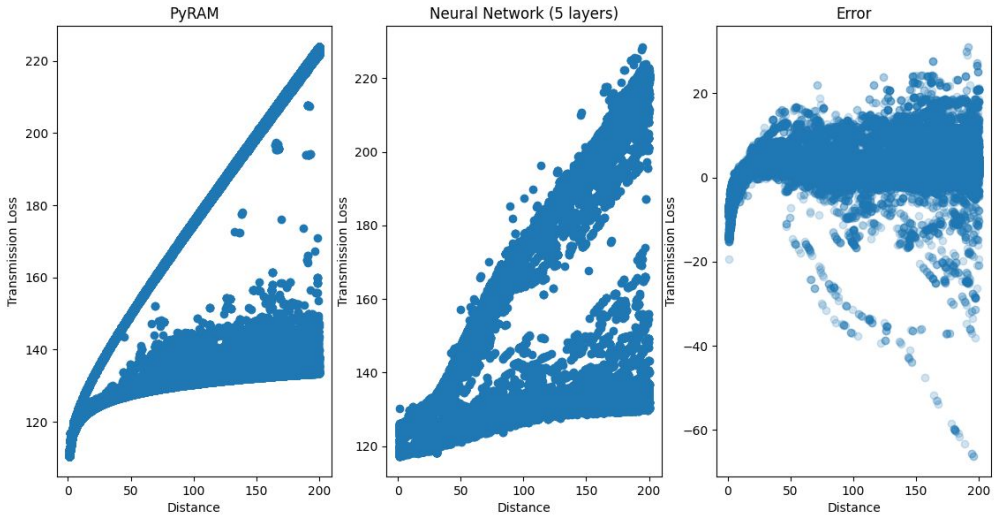
The MSE comes out to be 35.622, and the MAE 4.55dB. The Errors plotted as a histogram is shown below -



(Fig-8)

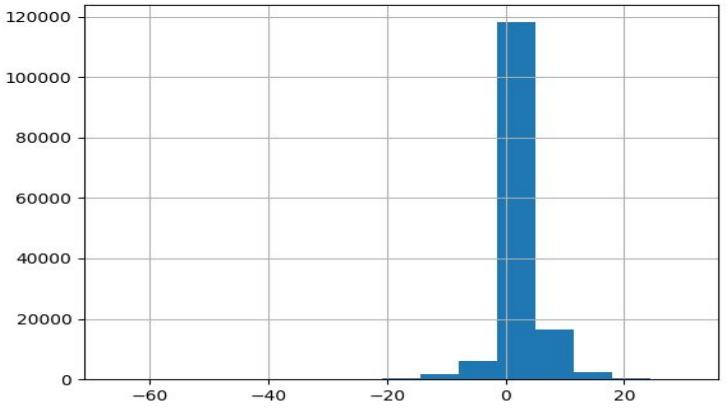
It can be seen that the majority of the errors are populated close to 0, a good sign, implying that a 4-layered neural network comes close to approximating the solution to the differential equation. Even this network can be further improved to include more neurons per layer, and be trained for more iterations, to get a better result than the one obtained above.

Similar results for the 5-layered network is as follows -



(Fig-9)
MSE = 20.55
MAE = 3.41 dB

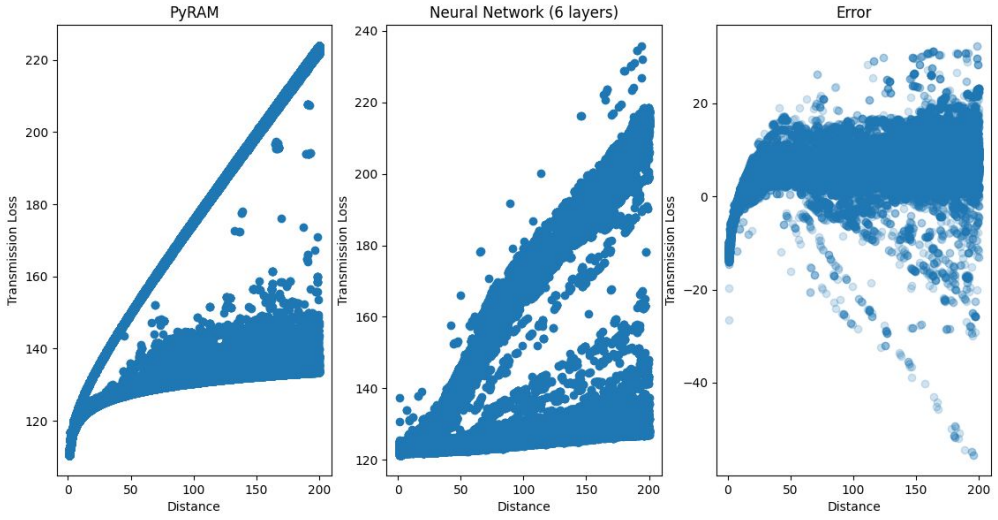
Error histogram -



(Fig-10)

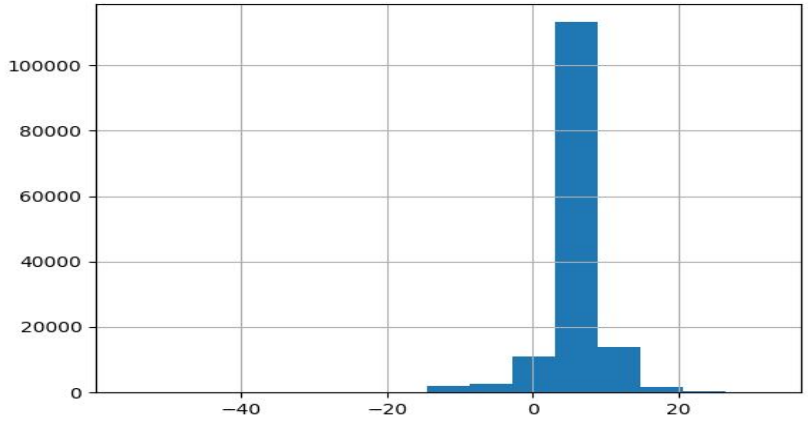
The same comments apply here, for the 5-layered network as well, relating to changing the neurons per layer and iterations.

And the 6 layered network -



(Fig-11)
MSE = 47.68
MAE = 6.27dB

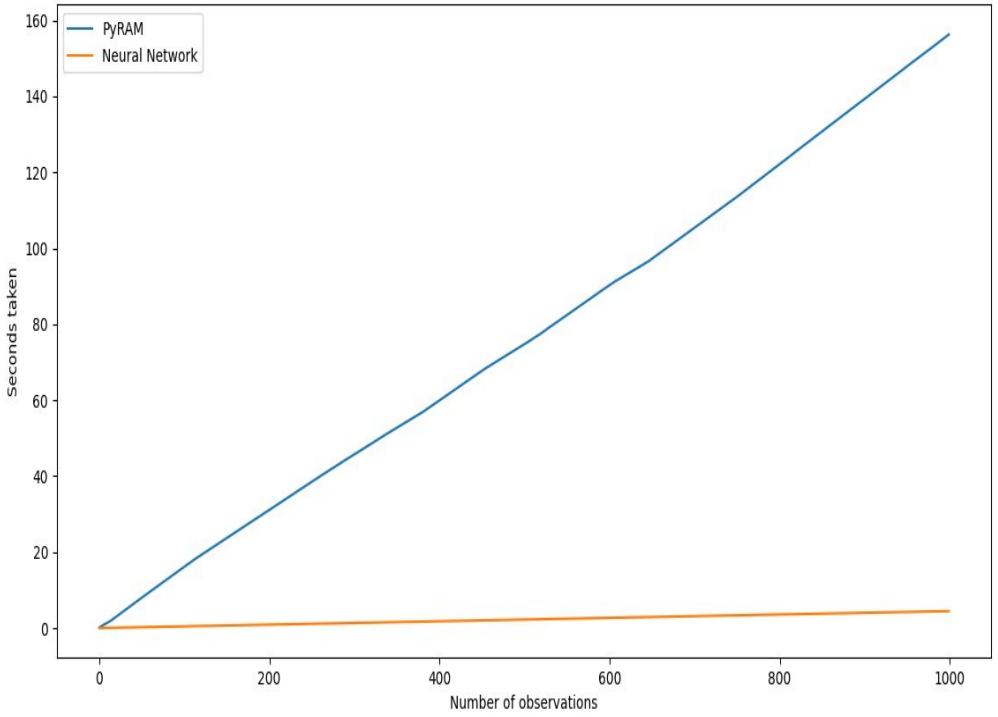
Error histogram-



(Fig-12)

As can be observed from figures 7, 9 and 11, the errors are mostly concentrated about the 0-20 region with a very few of them reaching -60dB or 60dB. This is reinforced by the error histogram, which shows that the majority of errors lie in the spike noticeable in the 0-20dB range (Fig 8, 10, 12).

The main issue we wanted to tackle was of the computational time. For this, we run both the PyRAM and the neural network for 1000 sender receiver pairs, and plot the run times in the same plot. The results are more than promising -



The total run time for 145000 pairs on a single core of an intel i5 processor was more than 12 hours, and this scales linearly with time. and as we can see, the neural network can run very fast, a significant improvement. This can also be sped up with parallel computing and running the neural networks on GPUs. Moreover, 99% of the compute time in the case of the neural network is of the data retrieval step, the automation step previously mentioned. The computation is performed almost instantly.

Advantages/Disadvantages

This section goes over the advantages and disadvantages we may face on using the Neural network over the traditional PyRAM model

Advantages -

- In real time usage, PE RAM model goes through the matrix decomposition step, whereas the neural network will go through the predict step. As shown above, the time taken is significantly reduced with this substitution.
- In terms of inputs, as mentioned before, PE-RAM model requires hyperparameters as well, the number of pade terms to calculate for, the step size of the range and depth, and a tradeoff between computation and accuracy. Such inputs are not required in case of the neural network.
- This system is a first of its kind, where the user needs to input only the sender and the receiver coordinates (though this is a trivial step, maybe why it is absent in literature).

But there are certain disadvantages as well -

- The black box assumption - As neural networks get deeper and more complex, the functions learnt by them become complex, and have to be treated as a black box[18]. This can lead to comprehensibility, and is an active area of research in the AI community.
- The errors of the predictions can sometimes reach 60dB (or -60dB), defeating the purpose of accurately predicting the transmission loss. These large deviations need to be analysed, and corrective measures need to be implemented for this.

Conclusions

As we have seen in the above sections, despite the disadvantages, ML techniques do have a significant edge over the traditional computational techniques. The final deliverable is a system where, upon input of the sender location and the receiver location, the program outputs the transmission loss between them. This will require automated scripts to retrieve data from databases such as ETOPO1, World Ocean Database, sound speed values using a computation readily available and easy to program.

Furthermore, the computational times implicate a significant improvement in real time usage. The accuracy, though dwindling, is for a small number of observations (so small it isn't visible in the histograms). Corrective measures may be put in place for this, else, this technique overcomes many shortcomings of the current SOTA.

Future Research Directions

- A major choke point is that we base this analysis on the results obtained from PyRAM. This, for one, limits the accuracy of the neural net as it is not possible to get a greater accuracy than PyRAM. A remedy to this would be to gather empirical data[20], and train a neural network directly on this data. This would also mean doing away with the implicit assumptions made by PyRAM
- Neural ODEs - A recent paper[19] comes up with a deep learning method to solve differential equations, that does away with the shortcomings of numerical methods. An attempt can be made to solve the PE using this method, thus not discarding this equation completely
- An attempt can be made at feature engineering, and then applying less sophisticated regression methods to predict the same transmission loss. This increased focus on feature engineering can lead us to a better understanding of the underlying mathematical phenomena, as well as the less sophisticated system will imply a lower cost solution (in terms of compute)

Bibliography

1) Underwater acoustics
2) Machine learning
3) "Underwater Acoustic Modelling and Simulation", Paul C. Etter
4) PyRAM on github - https://github.com/marcuskd/pyram/tree/master/pyram
5) The original RAM models in FORTRAN, C with 3D extensions, and parallelizability - PE Models

References

1. Collins, M. D. (1989). A higher-order parabolic equation for wave propagation in an ocean overlying an elastic bottom. *The Journal of the Acoustical Society of America*, *86*(4), 1459-1464.
2. Collins, M. D. (1993). A split-step Padé solution for the parabolic equation method. *The Journal of the Acoustical Society of America*, *93*(4), 1736-1742.
3. DeSanto, J. A. (1979). Derivation of the acoustic wave equation in the presence of gravitational and rotational effects. *The Journal of the Acoustical Society of America*, *66*(3), 827-830.
4. DiNapoli F.R., Deavenport R.L. (1979) Numerical Models of Underwater Acoustic Propagation. In: DeSanto J.A. (eds) Ocean Acoustics. Topics in Current Physics, vol 8. Springer, Berlin, Heidelberg
5. Weston, D. E., & Rowlands, P. B. (1979). Guided acoustic waves in the ocean. *Reports on Progress in Physics*, *42*(2), 347.
6. McCoy, J. J. (1980). Parabolic wave theories and some recent applications. *Physics of the Earth and Planetary Interiors*, *21*(2-3), 126-133.
7. Etter, P. C. (2018). *Underwater acoustic modeling and simulation*. CRC press.
8. Shalev-Shwartz, S., & Ben-David, S. (2014). *Understanding machine learning: From theory to algorithms*. Cambridge university press.
9. Ruder, S. (2016). An overview of gradient descent optimization algorithms. *arXiv preprint arXiv:1609.04747*.
10. Leroy, C. C., Robinson, S. P., & Goldsmith, M. J. (2008). A new equation for the accurate calculation of sound speed in all oceans. *The Journal of the Acoustical Society of America*, *124*(5), 2774-2782.
11. Amante, Christopher. (2009). ETOPO1 1 arc-minute global relief model : procedures, data sources and analysis. Boulder, Colo. :U.S. Dept. of Commerce, National Oceanic and Atmospheric Administration, National Environmental Satellite, Data, and Information Service, National Geophysical Data Center, Marine Geology and Geophysics Division,
12. Hamilton, E. L. (1976). Sound attenuation as a function of depth in the sea floor. *The Journal of the Acoustical Society of America*, *59*(3), 528-535.
13. Boyer, T. P., Antonov, J. I., Baranova, O. K., Garcia, H. E., Johnson, D. R., Mishonov, A. V., ... & Paver, C. R. (2013). World ocean database 2013.
14. Goodfellow, I., Bengio, Y., & Courville, A. (2016). *Deep learning*. MIT press.
15. Maas, A. L., Hannun, A. Y., & Ng, A. Y. (2013, June). Rectifier nonlinearities improve neural network acoustic models. In *Proc. icml* (Vol. 30, No. 1, p. 3).
16. Srivastava, N., Hinton, G., Krizhevsky, A., Sutskever, I., & Salakhutdinov, R. (2014). Dropout: a simple way to prevent neural networks from overfitting. *The journal of machine learning research*, *15*(1), 1929-1958.

17. Wang, H., & Raj, B. (2017). On the origin of deep learning. *arXiv preprint arXiv:1702.07800*.
18. Bullinaria, J. (1997). Analyzing the internal representations of trained neural networks. *Neural network analysis, architectures and algorithms*, 3-26.
19. Chen, R. T., Rubanova, Y., Bettencourt, J., & Duvenaud, D. K. (2018). Neural ordinary differential equations. In *Advances in neural information processing systems* (pp. 6571-6583).
20. Sessions, V., & Valtorta, M. (2006). The Effects of Data Quality on Machine Learning Algorithms. *ICIQ*, *6*, 485-498.