A Physics driven approach to solving inverse problem using Neural Networks

Master's Project

Written by *Yifan Liu*

September 3, 2021

Supervised by
Klaus Mosegaard

University of Copenhagen

UNIVERSITY OF COPENHAGEN

Name of Institute:	University of Copenhagen
Name of Department:	Niels Bohr Institute
Author(s):	Yifan Liu
Email:	liu.yi.fan@icloud.com
Title and subtitle:	A Physics driven approach to solving inverse problem using Neural Networks -
Supervisor(s):	Klaus Mosegaard
Handed in:	03.09.2021
Defended:	

Name ______________________________

Signature __________________________

Date ______________________________

Abstract

Recent developments in machine learning made it possible to solve complex problems through a purely statistical approach. One might ask is it possible to combine the power of neural networks and prior information, in this case laws of physics, to make a neural network drastically better? This project explores this idea by implementing physics information in the loss function of a fully connected feed forward neural network, in order to generally solve a scaleable inverse problem with out training for each specific case. Despite limitations of computational resource, the final results from physics informed neural network(PINN) is slightly better in goodness of prediction than the non-physics informed neural network(noPINN) when there are sufficient amount of training samples and neural network layers.

Contents

1 Introduction

Throughout human history, empiricism has played a big role on our understanding of nature. The science revolution in the Renascence period gave empiricism a boost with the scientific way of thinking and studying, which in turn led to huge leaps forward for human civilization.

For the past many years, Physics had been well regulated and constrained by the existing paradigm. Hypothesis were made before discoveries, researches were driven by theoretical predictions. The laws of Physics are well formulated in the language of mathematics.

As technologies advance, new tools were invented and gained huge traction in the scientific community. One of these tools is the use of machine learning. Contrary to mathematical formulas machine learning does not need to follow any mathematical constructs. Instead, this approach only looks at the data that is given to it and the pattern in the data. This statistical approach has served machine learning well but machine learning can be better if it makes use of the existing theories in Physics and decades of knowledge that humans have obtained beforehand. The Neural network that makes use of Physics information is generally called Physics informed neural network(PINN).

The potential benefits of physics informed neural networks could be shorter training time, more accurate results and more importantly physically realistic results. To understand how PINN works and different approaches to PINN, first let us get an understanding of the neural networks and how it works in general.

Overview of this thesis. Section 1 introduces basic concepts of artificial neural networks, physics informed neural networks and inverse problems. Section 2 contains methods used to generate synthetic seismic waves. Section 3 will take a detailed look into the neural networks setup used to generate the results. Section 4 describes all the relevant physics information and considerations when implementing them. Section 5 contains the criteria for evaluating results and different results. Section 6 contains discussions of different considerations and leanings from results. Section 7 highlights points of interest for future research. Lastly, section 8 concludes this thesis.

1.1 A brief introduction to artificial neural networks

There are many different types of neural networks; some of the most popular neural networks to date are Convolutional Neural Network(CNN) that excels in pattern and image recognition[1] and Recurrent Neural Network(RNN) that stands out in time series forecasting.[2] There are also many different ways to train the networks, such as, supervised learning, unsupervised learning and last but not least re-enforcement learning. In this thesis, we mainly focus on training a feed forward neural network with supervised learning.

Feed forward means that input signal is only travelling towards the direction of the output neurons. All neural networks that have the same feed forward characteristic can be called a Feed forward neural network(FFNN). Only fully connected layers are used in this thesis, it means all neurons from the previous layer are connected to all neurons in the next layer. A basic feed forward fully connected neural network consists of a few key components as follow,

- Activation function
- Fully connected neural layers consisting of weights and biases
- Loss function
- Optimizer

The best way to illustrate the FFNN with fully connected layers is through linear algebra with matrices.

1.1.1 Activation function

Activation functions are functions that map the input values of a neural network layer to other values that are often restricted to a certain range or have a certain characteristic. Some examples of activation function include the binary step function that maps x to 1 if x is larger or equal to 0 while mapping x to 0 if x is smaller than 0 as shown in eq.1 or the hyperbolic tangent function that maps $x \in R$ to values between 1 and -1 as shown in eq.2.

$$f(x) = \begin{cases} 0 & x < 0 \\ 1 & x \geq 0 \end{cases} \tag{1}$$

$$f(x) = tanh(x) = \frac{e^x - e^{-x}}{e^x + e^{-x}} \tag{2}$$

Activation functions are generally used to help neural networks to learn non-linear features of a problem.

1.1.2 Fully connected neural network layers

Fully connected layers in a neural network consists of activation function, weights and biases. Weights can be represented as a matrix while biases can be represented with a row vector.

Imagine we have a row vector of size R as our input, after it passes through the activation function of choice, it then multiplies with a matrix of the size R x C, which will give another row vector of size C. The new row vector is then added with another row vector of size C. This new row vector is the output values after one fully connected neural networks layer. Next, this row vector is passed through the rest of the layers of the neural network. Here the matrix of size R x C is the weights matrix and the row vector of size C is the biases matrix. This process is shown in fig.1.

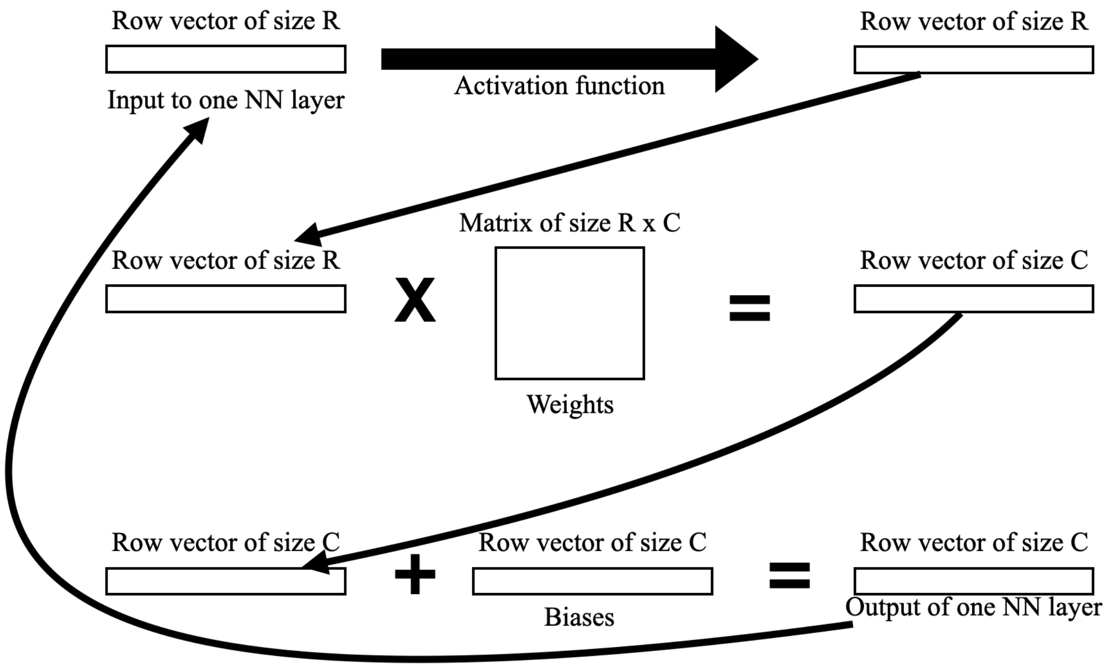


Figure 1: This process is repeated until the last layer of the network.

1.1.3 Loss function

In supervised learning, the output of the last layer of this network is compared to a predetermined row vector of the same size according to the loss function. One of the most common loss functions is Mean Square Error(MSE) as seen in eq.31. Where N is the number of training sample sets.

$$loss = MSE = \frac{1}{N}\sum^{N}(prediction - target)^2 \tag{3}$$

The values of the loss function is the key to another part of the neural network, the optimizer.

1.1.4 Optimizer

Optimizers work to determine how to tweak the weights and biases in order to minimize or maximize the loss function. The most popular optimizers use a form of gradient descent to determine the gradient of the loss function with respect to weights and biases in the layers of the neural network. One of the standard optimizer choice is Adam optimizer which uses Stochastic gradient descent with adaptive moment estimation to minimize the loss function.

1.2 Introduction to physics informed neural networks (PINN)

There are a few different approaches to include physics information into neural networks, in this thesis we look mainly into two different approaches. One of which involves direct manipulation of inverse problem parameters during training and the other involves manipulation of the standard loss function.

1.2.1 PINN with direct manipulation of parameters

To illustrate the basic principles of this method we take one of the examples about solving Korteweg–De Vries equation(KdV) from our main reference paper[3].

A KdV equation is a mathematical model of waves on shallow water surfaces, it has the following form,

$$u_t + \lambda_1 u u_x + \lambda_2 u_{xxx} = 0 \tag{4}$$

Where $u(t, x)$ is the wave value at location x and time t, u_t denotes the first order time derivative of u and u_x denotes the first order derivative of u while u_{xxx} denotes the third order deviated of u both with respect to x. λ_1 and λ_2 in this case are the parameters we need to solve.

This particular choice of inverse problem is interesting because the KdV equation originates from Burgers equation which is a fundamental partial differential equation that deals with the dispersion and reflection of shock waves, or in another term waves with discontinuities, which is notoriously hard to resolve by classical numerical methods."[3].

For this PINN, the optimizer not only needs to tweak weights and biases it also has direct access to the two parameters λ_1 and λ_2. The structure of the neural network is also slightly different as eq.4 is directly incorporated in the network structure. The details to how exactly it is done can be found in Appendix.A.

The advantages of this approach, based on the reference paper, is that the end values of the parameters λ_1 and λ_2 have a objectively small error.

The limitations on the other hand, include requirements of training for every case with different λ values and the flexibility of the inverse problem is also limited, meaning that the number of parameters is pretty limited as there is a need for an explicit expression containing the parameters and the problem always involves some sort of partial differential equation.

1.2.2 PINN with custom loss function

Another approach to introduce physics information to neural networks is to define a custom loss function with regulating terms encoding the prior information, in this case, physics information. [4] As shown below,

$$loss = g(prediction) + ... \tag{5}$$

Here g is the laws of physics containing prior information of which we evaluate the predictions from the neural network with. And there could be more terms in the loss function with different prior information. A regular non-physics informed neural network(noPINN) learn by minimizing the Mean Square Error(MSE) between the prediction and the target output in the case of supervised learning as shown in eq.31. This results in a prediction based purely on statistical information. With physics informed regulating terms in the loss function the neural network is expected to be able to take physics information into consideration while learning to minimize the new loss function.

The expected advantage of this approach is that there is no need for training for each specific case. Once the training is done the now PINN will be able to generate predict for new cases with respect to laws of physics.

Some of the limitations of this approach is that it is difficult to reach a meaningful balance of the normal MSE term in the loss function and the new regulating terms with physics information. If the normal MSE term is weighed too much it can overpower the optimizer and end up with a result that is no different than the ones without physics information. If the new regulating term is overpowered the optimizer would likely ignore the normal MSE term and hence ignore the statistical information. Another limitation is that the extra regulating term could be overly complex that the CPU would have to take the work load instead of the GPU which will end up increase the training time significantly.

1.3 Introduction to inverse problems

"Inverse problems are problems where physical data from indirect measurements are used to infer information about unknown parameters of physical systems. Noise-contaminated data and prior information on model parameters are the basic elements of any inverse problem."[5].

Imagine we have data $d = (d_1, d_2, ..., d_N)$, model parameters $m = (m_1, m_2, ..., m_M)$ and the physical relation that links model parameters to data $d = g(m)$. When m and g is known but d is unknown, we have a forward problem. When d and g is known but m is unknown, we have an inverse problem.

Simply put, it is an inverse problem when we have observational data but we want to calculate the original conditions of which the data was generated in.

An example could be as previously described in section 1.2.1. When we have the wave values $u(t, x)$ of the KdV equation in eq.4, but the λ_1 and λ_2 is unknown. The act of solving λ_1 and λ_2 is solving an inverse problem.

1.3.1 Solving inverse problems

One of the most primitive ways to solve an inverse problem is to solve the forward problem using the tweaked parameters and calculate the misfit between the forward problem solution and the true data. There after go back to tweak the parameters again and calculate the new misfit. This is similar to how the optimizer works in a neural network. the weights and biases would be equivalent to the parameters of the inverse problem and the loss function values would be the misfit of the data. And this is exactly the case for PINN with direct manipulation of parameters.

1.3.2 Difficulties in solving an inverse problem

One of the difficulties in solving inverse problems is the non-uniqueness of solutions. That means there might be multiple solutions to a inverse problem that give the same misfit of the data. This is especially apparent when dealing with under-determined problems which mean there is less data than there are parameters to solve. This under determined problem is explored in this thesis by varying the training sample size and the results can be seen in fig.10.

Another difficulty associated with solving inverse problem is the hidden physics relation, if the physics relation between the data and the parameters are not known then it is necessary to develop a theory to describe the physics relation before solving the inverse problem.

2 Seismic Wave Scattering

Since we want our test problem to be scale-able with many parameters and at the same time to be able to quickly generate a large amount of synthetic training data, we have chosen a classic seismic wave scattering problem. The following section refers to the paper from D. C. Ganley[6].

2.1 Basic concepts of seismic wave propagation

By sending a pressure wave from the earth surface vertically downwards and measure the reflected seismogram we will be able to have an idea about the layer structure of the earth directly below the surface in terms of their acoustic impedance. That is because the pressure wave will be reflected at the boundary of two layers with different impedance values with the reflection coefficient defined as follow,

$$R_1 = \frac{\rho_1 v_{p1} - \rho_2 v_{p2}}{\rho_1 v_{p1} + \rho_2 v_{p2}} \tag{6}$$

v_{p1} and v_{p2} here represents the velocity of the pressure wave, also known as acoustic impedance, at two different layers. ρ is the density of earth at each layer. R_1 is the reflection coefficient at the boundary of layer 1 and layer 2 for waves travelling from layer 1 towards layer 2 (down going). The transmission coefficient of the pressure wave is then,

$$T_1 = 1 + R_1 = \frac{2\rho_1 v_{p1}}{\rho_1 v_{p1} + \rho_2 v_{p2}} \tag{7}$$

The down going wave can hence be expressed as follow,

$$D_{i+1} = T_i D_i' + R_i' U_{i+1} \tag{8}$$

Here D_{i+1} is the down going wave at the top of in layer i+1. U_{i+1} is the up going wave at the top of layer i+1 as seen in fig.2. T_i and R_i' are the transmission and reflection

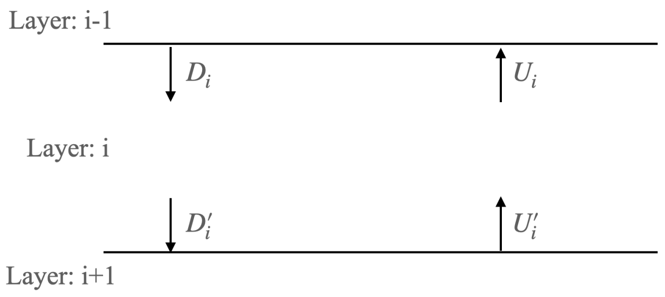


Figure 2: An illustration of down going (D) and up going (U) waves within a layer. Here prime denotes the waves at the bottom of a layer.

coefficient at the layer boundary between layer i and i+1. Prime denotes that it is the corresponding coefficient for up going wave. In this case, R_i' is the same as $-R_i$. Waves D and U are the Fourier Transformation of the corresponding waves in time domain, hence they are frequency dependent complex values.

The up going wave can hence be expressed as follow,

$$U_i' = R_i D_i' + T_i' U_{i+1} \tag{9}$$

By combining eq.8, eq.9 and take into account the damping coefficient α and time delay in travelling within the layer between D_i, D_i' and U_i', U_i, we get the following propagation equation governing up going and down going waves,

$$D_i = \frac{e^{\alpha d_i} e^{iwd_i/c_i}}{T_i}(D_{i+1} + R_i U_{i+1}) \tag{10}$$

$$U_i = \frac{e^{-\alpha d_i} e^{-iwd_i/c_i}}{T_i}(R_i D_{i+1} + U_{i+1}) \tag{11}$$

Here d_i and c_i are the depth and the speed of the wave at layer i, w is the angular frequency of the wave in frequency space.

2.2 Generation of synthetic seismic waves

From eq.10 and eq.11, we have the basic formulations of the propagation method governing the wave propagation from the bottom layer up to the top layer. This requires that the down going and up going wave at the bottom layer is known. While up going wave is 0 at the bottom layer, the corresponding down going wave at the bottom layer, in this case, is an unknown variable. On the other hand, down going wave at the top layer can be represented by 1 as it is assumed to be a spike wavelet in frequency space. And the up going wave at the top is an unknown variable.

A cleaver work around of the unknown down going wave at bottom layer is to calculate the ratio of up going and down going wave $Y_i = U_i/D_i$. By dividing eq.11 with eq.10, the following expression for Y_i can be obtained,

$$Y_i = e^{-2\alpha d_i} e^{-2iwd_i/c_i} \left(\frac{R_i + Y_{i+1}}{1 + R_i Y_{i+1}}\right) \tag{12}$$

At the bottom layer $Y_{bottom} = 0$, while at the top layer $Y_{top} = U_{top}$ as $D_{top} = 1$.

The seismic response at the top layer from a down going spike wavelet can then be expressed as,

$$X(w) = U_1(w) + D_1(w) = U_1(w) + 1 \tag{13}$$

$X(w)$ is the Fourier transform of the synthetic seismogram at the surface. Multiplying this with the Fourier transformation of a different wavelet will give the seismic response from that particular down going wavelet. Inverse Fourier transform the resulting $X(w)$ would give the seismic response in time domain.

From here on, in order to generate synthetic seismograms of primary waves that are to be used in our neural networks training, we first define a structure of under earth layers that is discretized to 1000 layers and each layer has a corresponding impedance value that in term determines the primary wave velocity at the respective layers. Each layer has a thickness of $d_i = 2 * dT * c_i$, dT here is the two way time thickness, to ensure all layers have the same two way time thickness in the time domain. In this way, the fluctuations in the generated seismic response, when plotted along with the layering structure in the time domain, happen at the same time in the x-axis as the reflective layer boundaries that caused the fluctuations.

3 Neural Networks Setup

The neural network is set up and trained on Google Colab platform. It uses Keras with TensorFlow back-end as the base for neural networks training, it also utilizes GPU acceleration. This section includes some of the technical details of the supervised learning setup, in order to give an approximate picture of the training process.

3.1 Data generation & preparation

The data we generate are seismic responses from a structure discretized to 100 layers. The depth of the layers are measured in two way time, all the layers have the same two way time thickness of 0.01 [seconds]. The impedance values for each layers have a unit of $[km \cdot kg/s/m^3]$ and are generated randomly with uniform distribution between the values 1 to 9. Damping coefficient α is set to 0 through out this thesis in order to simplify the problem.

The resulting seismic wave have 4096 time steps where each time step is 0.001 [seconds]. The seismic data is generated using MATLAB codes based on the propagation principle in section 2.2. Instead of a spike wavelet the code utilized a Mexican hat wavelet as the down going wavelet at the top layer. The MATLAB codes were originally provided by Professor Klaus Mosegaard. An example of the generated data can be seen in fig.3.

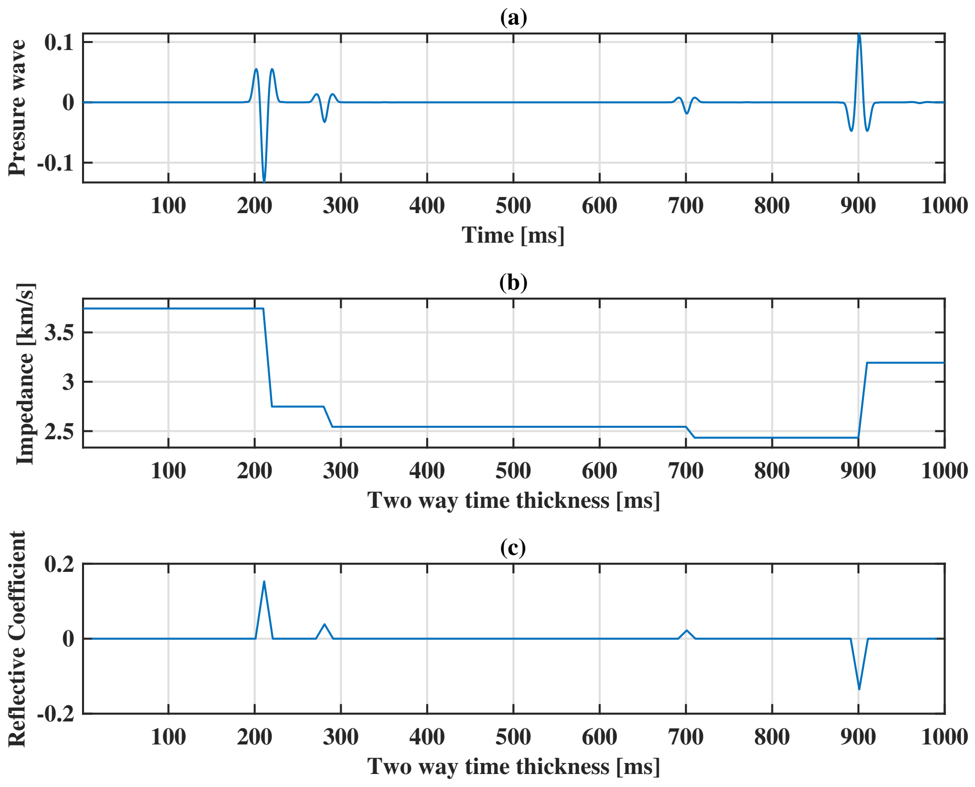


Figure 3: (a) The 0-1000 [ms] section of the seismic wave response generated by the impedance values structure in (b). (b) The impedance values structure in two way time domain in this case with 5 distinct layers and 100 numeric layers, each numeric layers have a two way time thickness of 10 [ms]. (c) The corresponding reflective coefficient, following eq.6, of the layering structure in (b).

3.1.1 Choice of training input

For the training input we only make use of the first 3000 data points, first 3 seconds, of the seismic wave in time domain as our input. This is mainly due to the fact that all the responds after first 1 second of the data is due to reflections between layers. We would like to include the reflections, hopping the network will learn more from the reflected waves. But at the same time the reflections decrease in amplitude as time passes. Hence first 3 seconds are enough

The total dimension of the problem is also dependent on the total number of data points as follow,

$$Dimension = N_{input} + N_{output} \tag{14}$$

N_{input} and N_{output} are the number of data points used as input and output respectively.

This is important as the minimum number of the training sample should exceed the number of dimensions to avoid it being an under determined problem.

3.1.2 Choice of target output

While we have tried many different types of target output, include but not limited to impedance values in two way time domain and logarithmic impedance values in two way time domain. We landed on reflective coefficient in two way time domain. The sizes of the target output is the same as the number of layer boundaries, in this case a 1 by 99 row vector.

The advantage of using reflective coefficient is that it has a range between -1 and 1 which does not require future processing. Unlike using impedance values directly which poses a non-uniqueness challenge, different impedance values depending on the starting impedance value could end up with same reflective coefficient. A comparison between directly predicting impedance values and directly predicting reflective coefficient using non-physics informed neural network can be seen in fig.4 and fig.5. It is also apparent that the direct predictions of reflective coefficient eliminated many small variations in reflective coefficient values compared to directly predicting impedance values.

The advantage of using layer thickness in two way time domain is that there is a clear correspondence of when there is a peak of seismic wave and when the reflective coefficient is not 0.

3.2 Training setup

We tried mainly with 8 neural networks layers and 12 neural networks layers. The structure for 8 neural networks layers is 4 fully connected layers with 3000 neurons, 3 fully connected layers with 1000 neurons and 1 fully connected layer with 99 neurons. The structure for 12 neural networks layers is 6 fully connected layers with 3000 neurons, 5 fully connected layers with 1000 neurons and 1 fully connected layer with 99 neurons.

The network uses early stopping to stop training if the loss function has not improved for more than 0.001 for a certain number of epochs. This number has changed along the way and for different situations but we ended up with 4 epochs. The loss function value it monitors is the validation loss which is the loss function value of the validation set. When the training is stopped by early stop it also restores the weights and biases of the epoch that gave the best validation loss value.

The network also uses a learning rate reduction call back function that also monitors the validation loss value, if that did not decrease for a certain amount for a certain number of epochs the learning rate of the optimizer is reduced by a factor. All these factors have also changed many times along the way. Here I provide an approximate range of these parameters. The validation loss minimum delta is between 0.001 and 1. The number of epochs before changing the learning rate is 3 epochs. The factor that the learning rate reduces by is 0.2. The minimum learning rate can be achieved is $1E-9$. The starting learning rate is $1E-5$.

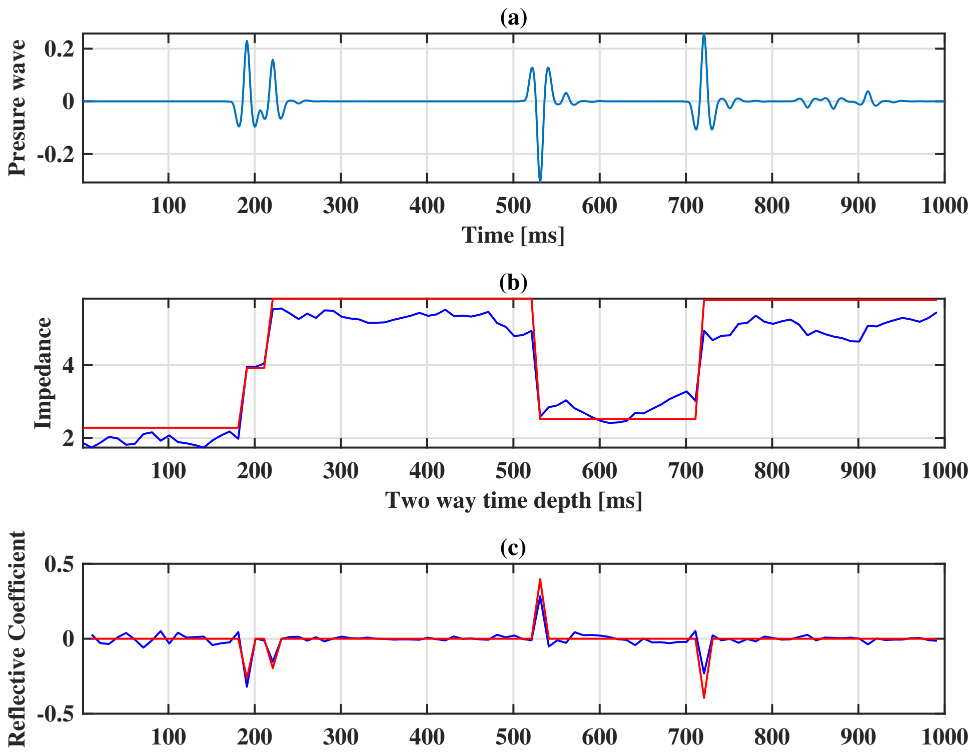


Figure 4: (a) The 0-1000 [ms] section of the input data to the neural network from the validation set that are used to generate predictions in (b). (b) The red line is the true impedance values, the blue line is the direct prediction of impedance values. (c) The red line is the reflective coefficient from true impedance values and the blue line is the reflective coefficient from predicted impedance values.

Batch sizes are set according to training sample size, the optimizer used is Adam, stochastic gradient descent with adaptive moment estimation and The activation function used in every layer of the network is hyperbolic tangent.

A batch size is the number of the training samples passed through the network at the same time. While an epoch is defined when all training samples passes through the network. So for a training sample size of 100 and batch size of 10. There will be 10 batches before 1 training epoch is complete. The weights and biases are tweaked while evaluating each batches. So for our 10 batch epoch the optimizer tweaks the weights and biases 10 times per epoch.

3.3 Training output

There are two different output predictions after training. One is what we call predictions from non-physics information neural network, in this case, the loss function directly computes the mean square error(MSE) between predicted reflection coefficient values and the target reflection coefficient values. A simple example of training input and output as well as the target output for supervised learning can be seen in fig.5. A more complex example can be seen in fig.6.

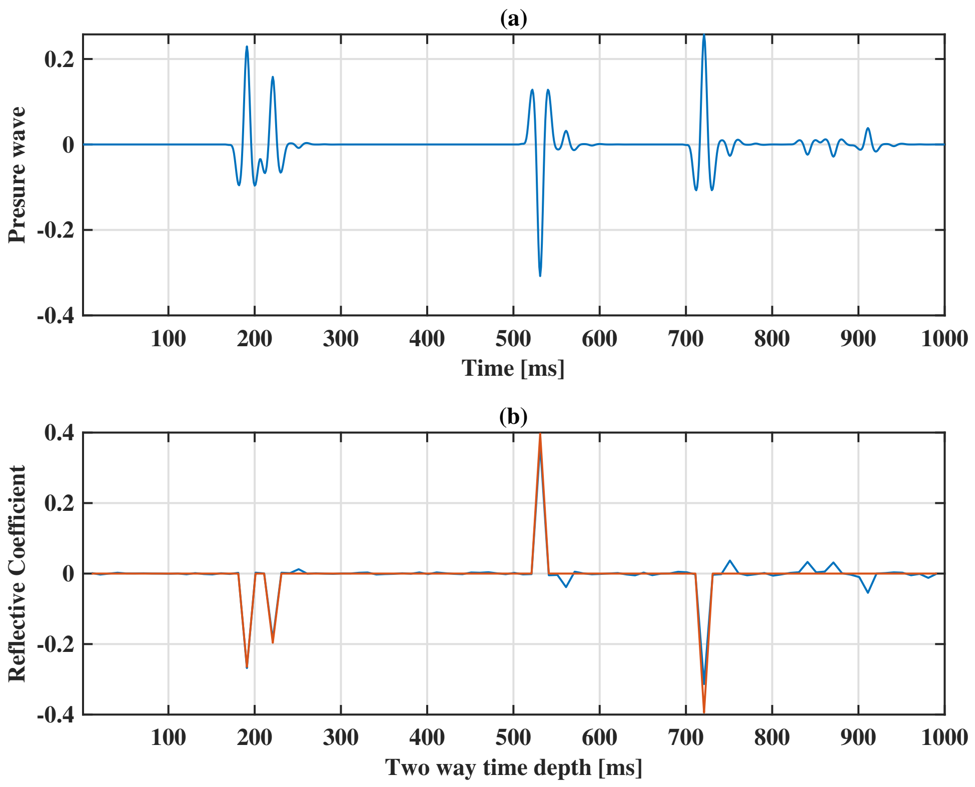


Figure 5: (a) The 0-1000 [ms] section of the input data to the neural network from the validation set that are used to generate predictions in (b). (b) The red line is the true reflective coefficient and the blue line is the predicted reflective coefficient from noPINN.

The second is predictions from pure physics information neural network(PINN), in this case, the loss function computes the chosen physics information as described under Physics information in section 4 without computing the MSE between the predicted reflection values and the target reflection coefficient values. The physics information is calculated for a certain number of frequencies ω in Fourier Space as the computation time is the limiting factor. The reasons why certain ω is selected will be explained under "Selection of ω" in section 4.7.

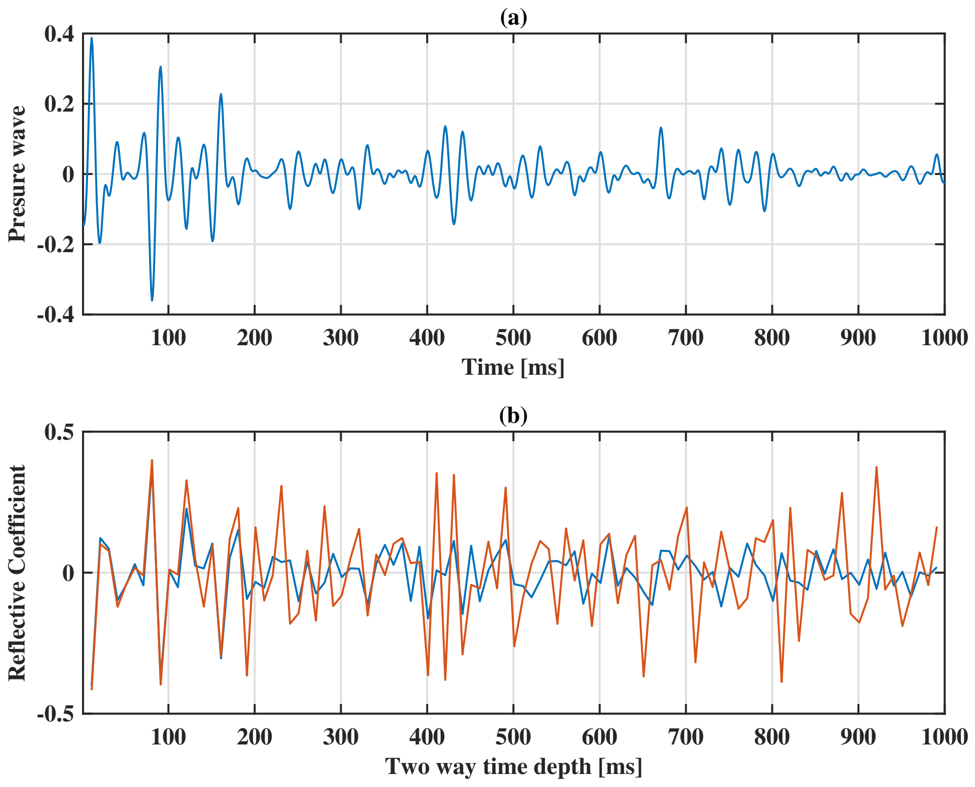


Figure 6: (a) The 0-1000 [ms] section of the input data to the neural network from the validation set that are used to generate predictions in (b). (b) The red line is the true reflective coefficient and the blue line is the predicted reflective coefficient from noPINN.

4 Physics Information

While our preliminary results from the non-physics informed fully connected neural network looks promising for simple 5-distinct-layer cases as seen in fig.5. The non-physics informed neural network failed to deliver satisfactory results when it comes to more complicated cases, for example the 100-distinct-layer cases seen in fig.6. This more complex case is what we are trying to improve by implementing physics information.

This section contains the description, in chronological order, of different kinds of physics information proposals that were explored in this project and the reasons behind the selection the frequencies ω to implement these physics information in. As all the physics information in this section are a function of ω in Fourier space.

One of the critical criteria for choosing what physics information to use is how good the physics information involves all the predicted layers of the problem. A good physics information should involve all the predicted impedance values on all the layers. This is to ensure that the physics information can provide a holistic picture of the problem formulation. The effects of different PINNs that are not our final physics information of choice can be found in Appendix B.

4.1 Wave ratio Y at the bottom layer

This is one of the most intuitive physics information for this problem. As the seismic waves on the top layer were generated with a propagator that goes from bottom layer to the top layer as seen in eq.12. It assumes that the up going, down going wave ratio Y on the bottom layer is 0, that means the up going wave at the bottom layer is 0. This physics information can be used as part of the loss function neural network while training. As it makes use of all the impedance values in the predicted layer structure. Therefore it is reasonable to expect this physics information to regulate all the impedance values at all the layers at the same time.

To implement this physics information, we will need to generate the up going, down going wave ratio Y at the top layer first, using the true impedance values, through the propagator method in eq.12. This ratio Y on the top layer is then back propagated from the top layer to the bottom layer using the predicted impedance values. The backwards propagator method can be derived from eq.12 to be,

$$Y_{i+1} = \frac{e^{2\alpha d_i + 2iwd_i/c_i} Y_i - R_i}{1 - e^{2\alpha d_i + 2iwd_i/c_i} R_i Y_i} \tag{15}$$

Here if the predicted impedance values get an up going, down going wave ratio of 0 at the bottom layer, then we say the prediction satisfies the physics. Hence this is one of the terms the neural networks optimizer can minimize. Because Y is a complex number we only take the absolute value of it as shown below.

$$Loss = \sum^{N_{samples}} |Y_{bottom}| \tag{16}$$

4.2 Residual of wave ratio at the top layer

As it is possible to get the up going, down going wave ratio Y at the top layer from the propagator. We can compare the misfit of the results generated from the true impedance values and the results generated using predicted impedance values as shown below,

$$Loss = \sum^{N_{samples}} |predY_{top} - trueY_{top}| \tag{17}$$

This is relevant as for solving inverse problems the only criteria is how close the result is to the observed data. Here the wave ratio Y on the top layer is the representation of the observed data at a specific frequency in frequency space.

The seismic wave on the top layer in time domain essentially contains the same information as its counterpart in frequency domain. Hence the comparison of misfit in frequency domain is valid. On the other hand, it is not possible to compare the misfit in time domain, even tho we would very much like to, as transforming from frequency domain to time domain requires the knowledge of all the frequencies but we only have computational capacity for a few frequencies.

4.3 Up going wave at the bottom layer

Slightly different from the previous wave ratio Y, this physics information only take into account the up going wave at the bottom layer. The closer to 0 the absolute value of the up going wave the better. This physics information is in principal better than using the wave ratio at the bottom layer, as the assumption of the propagator method is that the up going wave is 0, hence the wave ratio Y is 0. By isolating up going wave from the wave ratio, this eliminates the possibility of a false optimization as wave ratio Y could be minimized by maximizing down going wave while keeping the up going wave non-zero.

To implement this physics information we will need to make use of eq.10 and eq.11 to get two expressions of U_{i+1} and D_{i+1} as a function of U_i and D_i as seen below,

$$D_{i+1} = \frac{T_i(D_i e^{-\alpha d_i} e^{-iwd_i/c_i} - R_i U_i e^{\alpha d_i} e^{iwd_i/c_i})}{R_i^2 - 1} \tag{18}$$

$$U_{i+1} = \frac{T_i(U_i e^{\alpha d_i} e^{iwd_i/c_i} - D_i R_i e^{-\alpha d_i} e^{-iwd_i/c_i})}{R_i^2 - 1} \tag{19}$$

Then just as in the previous section, we first get the wave ratio Y at the top layer with the propagator method using true impedance values at each layer. As wave ratio Y is known, so are starting values of D_1 and U_1. We can then make use of eq.18 and eq.19 with predicted impedance values to propagate down to the bottom layer to get the up going wave at the bottom layer. The loss function term is then defined as follow.

$$Loss = \sum^{N_{samples}} |U_{bottom}| \tag{20}$$

4.4 Sum of residuals of waves at every layer

As the results of previous physics information did not meet our expectation, we decided to include up going and down going waves at every layer and compare the residuals between the two waves generated using predicted impedance values and true impedance values.

The implementation of this physics information is the same as for up going wave at the bottom layer, as in order to get the up going wave at the bottom layer we will need to propagate through all the layers. Hence we already have all the up going, down going wave information at each layer. The additional information is that we will need to do the same propagation with true impedance values to get true up going, down going wave values at each layer. We can then subtract predicted values from true values to get the residuals at each layer as follow,

$$Loss = \sum_{i=1}^{N_{layers}} (|trueU_i - predU_i| + |trueD_i - predD_i|) \tag{21}$$

This physics information is expected to be better than the previous ones due to the extra information it encodes between all the layers. Ideally it would not only help the network to learn about the impedance structure as a whole, but also help to learn the relations between each layers. One draw back of this physics information, however, is that it could overly emphasize on the early layers, in this case the layers towards the surface as the propagation direction is from top to bottom, a small residual during the early stage could end up with a huge residual towards the end, somewhat like a butterfly effect. Hence there is a chance that it would not treat all the layers equally.

4.5 Sum of residuals of wave ratios at every layer

This is an extension of the sum of residuals of waves at every layer it is the same approach but this time take the sum of the residuals of wave ratios Y at each layer.

This can be done by just using the propagator method in eq.12. There are two separate propagation, one with true impedance values and the other with predicted impedance values. The residuals were then taken as follow

$$Loss = \sum_{i=1}^{N_{layers}} |trueY_i - predY_i| \tag{22}$$

This physics information has the same drawbacks as the previous one. But this time the propagation is from the bottom layer to the surface. So it would emphasis on the bottom layers instead of top layers.

4.6 Sum of residuals of wave ratios at every layer including backward propagation

This physics information is an addition to the sum of residuals of wave ratios at every layer. It not only includes the previously mentioned sum, it also includes the sum of residuals of wave ratios when using a propagator going from top layer to the bottom layer. This way it will hopefully eliminate the previously mentioned drawbacks about not treating the layers equally.

The implementation starts the same by getting the true wave ratio Y at top layer, while summing the residuals of wave ratios, along the way, propagating from bottom to top layer. The true wave ratio Y is then used as a starting point for the backwards propagator shown in eq.15, using both true impedance values and predicted impedance values. The Loss value is then as shown below,

$$Loss = \sum_{i=N_{layers}}^{1} |Up\_trueY_i - Up\_predY_i| + \sum_{i=1}^{N_{layers}} |Down\_trueY_i - Down\_predY_i| \tag{23}$$

4.7 Selection of frequencies ω for physics informed training

In ideal scenarios, all frequencies should be used as a constrain for physics informed neural network. However, the reality is physics informed neural networks take a longer time to train than non-physics informed neural networks, depending on the complexity of the physics information and how many frequencies are chosen to implement these physics information. Therefore it is essential to be able to choose the correct frequencies to apply the physics information to.

This section takes a deeper look behind each physics information and proposes a selection of frequencies for the respective physics information to implement and the reasons behind the selection process.

4.7.1 Criteria for choosing a certain range of frequencies

When choosing a frequency to implement a certain physics information, it is intuitive to take a look at the results of the physics information based on the predictions produced by non-physics informed neural network. In order to then maximize the effect of the physics information it could be a good idea to take the frequencies that has the largest resulting values of that particular physics information. As the Loss function that consists of this physics information is to be minimized. This in principle would make it easier to minimize the value of the physics information at that frequency.

Another factor is that we would like to apply the physics information to as many frequencies as possible but there is a limit on how many frequencies we can use. So by limiting the range of the preferred frequency, the gap between the physics information constrained frequencies can be smaller and therefore hopefully function better.

4.7.2 Example frequency selection for sum of residuals of wave ratio Y at the bottom layer including including backward propagation

The physics information calculated according section 4.6 using predictions made using no physics information can be seen in fig.7.

Upon closer inspection the physics information presents local minimums at frequencies closest to multiples of π. This is interesting as if ω is divisible by π, that would eliminate the complex part of the exponential term in our up-ward and down-ward propagator seen in eq.12 and eq.15 and result in the following respectively,

$$Y_i = \pm(\frac{R_i + Y_{i+1}}{1 + R_i Y_{i+1}}) \tag{24}$$

$$Y_{i+1} = \frac{\pm Y_i - R_i}{1 \pm R_i Y_i} \tag{25}$$

The complex part of $e^{2\alpha d_i + 2iwd_i/c_i}$ is eliminated because α is set to 0 as described previously and $2d_i/c_i$ is the two way time thickness which is fixed to 0.001. Hence when ω is around 314 in fig.16, the exponential term approximates to $e^{i\pi} = -1$ and we have our

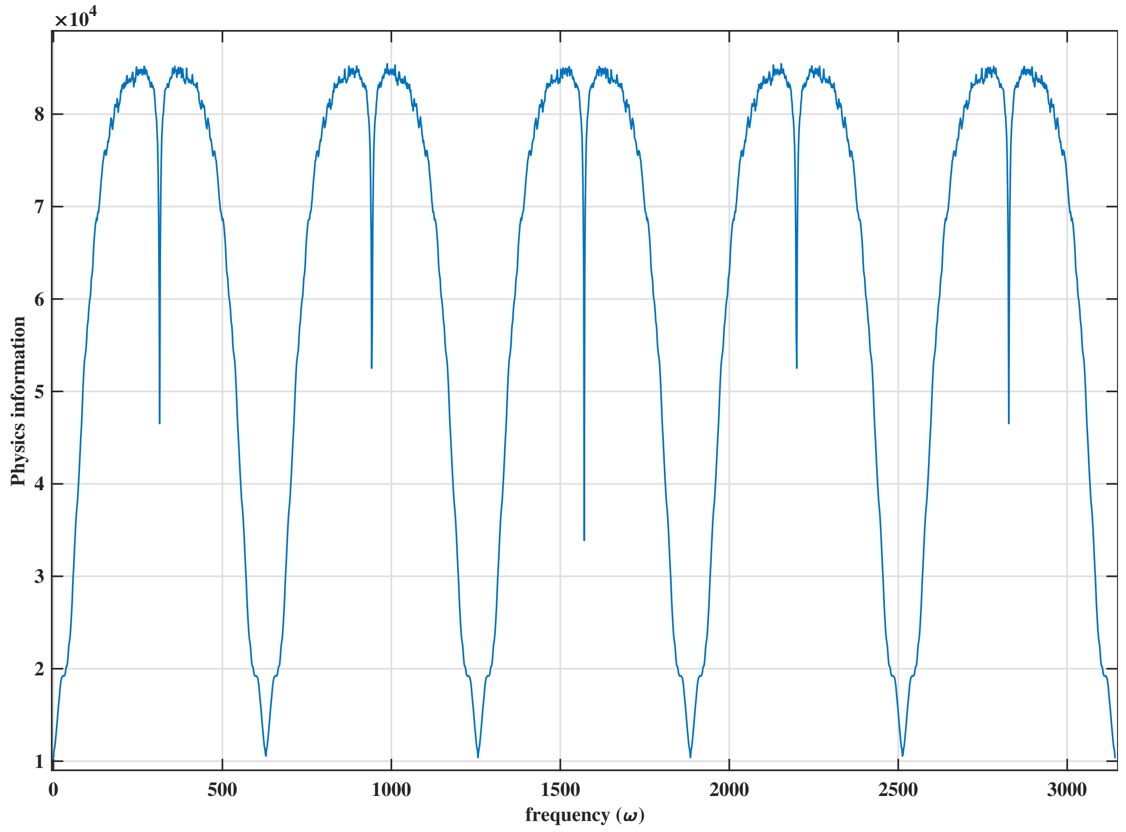


Figure 7: The curve is the result of the physics information in section 4.6 from non-physics informed network as a function of frequency.

local minimum. When ω is around 628 the exponential term approximates to $e^{i2\pi} = 1$ and we have another local minimum.

Another observation from fig.7 is that the curve is repeating itself as ω gets over 2π. This is understandable as the exponential term, which is the only varying factor in eq.12 and eq.15, come back to the staring point as omega completes a full rotation from 0 to 2π.

Last but not least, the curve in the range between the minimum at $\omega = 0$ and the minimum around $\omega = 614$ is approximately axis-symmetric around the y-axis that crosses the minimum point around $\omega = 314$. This is not as apparent as the previous observations, but it is observable when examining the same figure generated with results from physics informed neural network. Then it is possible to see that the same effect of the constrain appears at the corresponding location on the other side of the axis-symmetric axis.

Hence the ideal domain of frequencies to apply this particular physics information constrain lies between $\omega = 0$ and the local minimum around $\omega = 314$. This domain of frequency can actually be applied for all the physics information. As further inspection reveals that they all have the same properties as shown in this example. See Appendix B.

5 Results & Analysis

In this section, we will take a look at the neural networks prediction results from physics informed neural networks and non-physics informed neural networks and compare the predictions. As seen in the simple and complex cases in fig.5 and fig.6. We will mainly be looking at the more complex cases for physics informed neural network as it poses a more non-linear challenge.

5.1 Criteria for goodness of prediction

As there are prediction results generated with physics information and without physics information. We will need to compare the goodness of the predictions in the good old inverse problem way which is the sum of squared residual of the seismic data from the predictions and from the true impedance values. This is shown in eq.26

$$Goodness = \frac{1}{N}\sum^{N}\sum residual^2 \tag{26}$$

Here N is the number of validation sets for the respective simple or complex scenarios. The respective error on the goodness is then the standard deviation of the N values as follow,

$$Error = \sigma(\sum residual^2) \tag{27}$$

The smaller the goodness of prediction the better the prediction.

5.2 Results without physics information

While our predictions looks promising when dealing with simple cases. It is the exact opposite for complex cases with more non-linearity. This non-linearity is directly related to how many distinct layers of different impedance values as shown in fig.8.

5.2.1 Mostly linear scenarios

As seen in fig.5 and fig.8. When it comes to mostly linear scenarios with 5 distinct impedance values the neural network without physics information does a exceptionally good job at predicting the reflective coefficients for both the amplitude and the location. However we can already see that the reflections in seismic wave between layers are causing a problem for the neural network. The network clearly mistakes the reflections between layer boundaries for reflections from layer boundaries. Which is why the predicted reflective coefficients show responses at depths corresponding to these reflections.

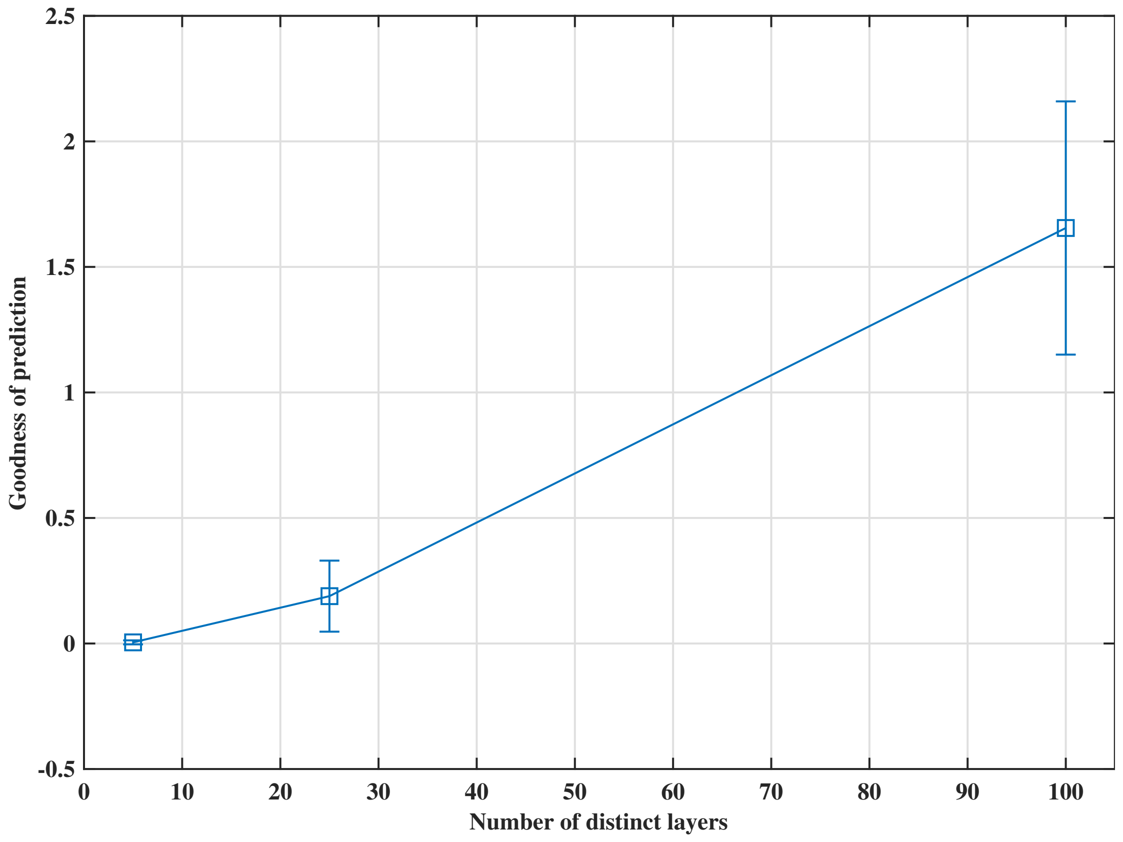


Figure 8: This figure shows the goodness of prediction as a function of number of layers with distinct impedance values. The respective error bars are calculated as shown in eq.27.

5.2.2 Highly non-linear scenarios

As seen in fig.6. The highly non-linear scenario with 100 distinct impedance values presents a challenge for the neural network. The predicted reflection coefficient was able to stay closely with the true reflective coefficient at shallower depths but was ultimately unable to predict the true reflective coefficient at deeper depths and the goodness of prediction confirms our observation as well.

5.3 Results with physics information

Results from the predictions made from neural networks trained with physics information described in section 4.6 can be seen in fig.9.

The neural networks optimizer worked well to minimize the values of the physics informed curve at the selected frequencies. The curve is below the comparison at most places. The goodness of prediction from the physics informed network is however still not better than the ones from non-physics informed neural network. This leads us to

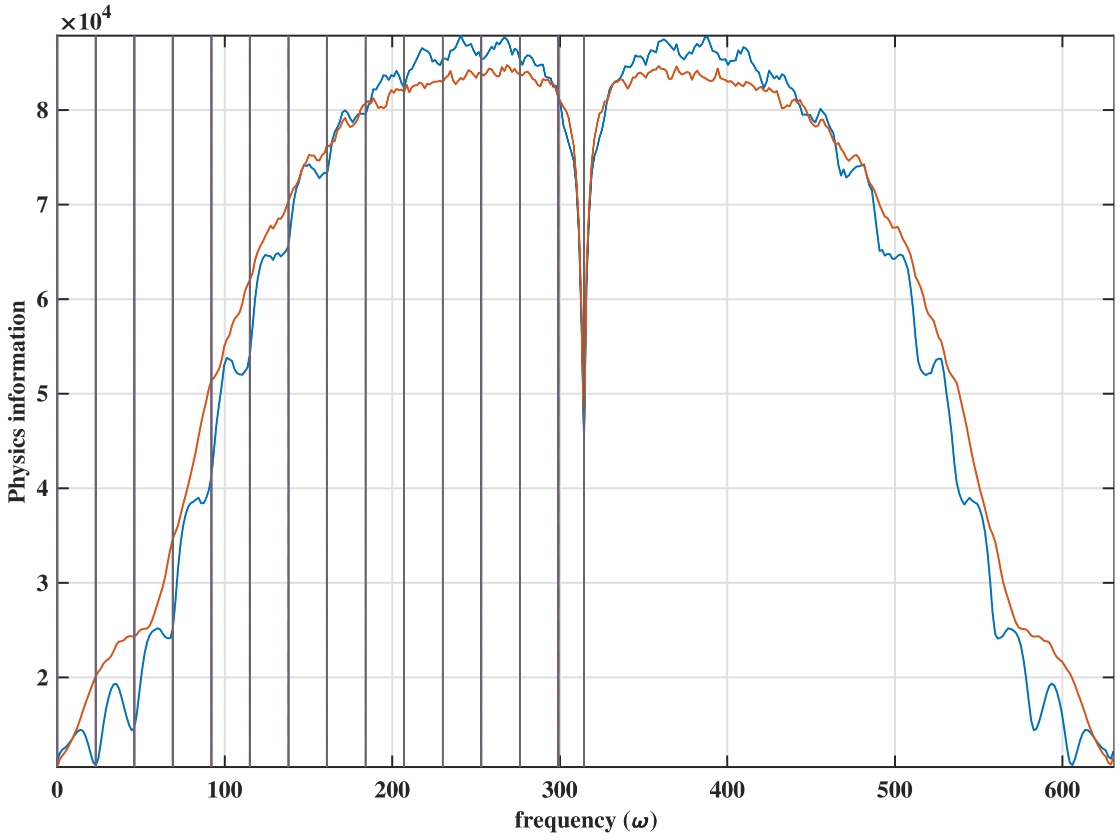


Figure 9: The [0 631] section of fig.7. The red line is the result of the physics information in section 4.6 from non-physics informed network. The blue line is the result of the same physics information from physics informed network. The vertical black lines denote the frequencies where the information was applied.

suspect might there be some other factors that may affect the goodness of prediction.

5.4 Effect of different training sample size

One of the factors that clearly had an effect on the goodness of fit is the number of samples used for training. The effect of different sample size is pretty substantial for our physics information of choice as shown in fig.10. Prediction results with loss function consisting of pure physics information did not outperform the loss function with no physics information for smaller training sample sizes. However, the decrease in goodness of prediction is sharper for physics informed neural network than that for the non-physics informed neural network between 6 and 8 on X-axis. We then decided to expand the graph from 8 to 11 and we see that once the training sample size increases to a certain level the goodness of prediction from physics informed neutral network did catch up with non-physics informed prediction and eventually overtake the prediction made using no physics information.

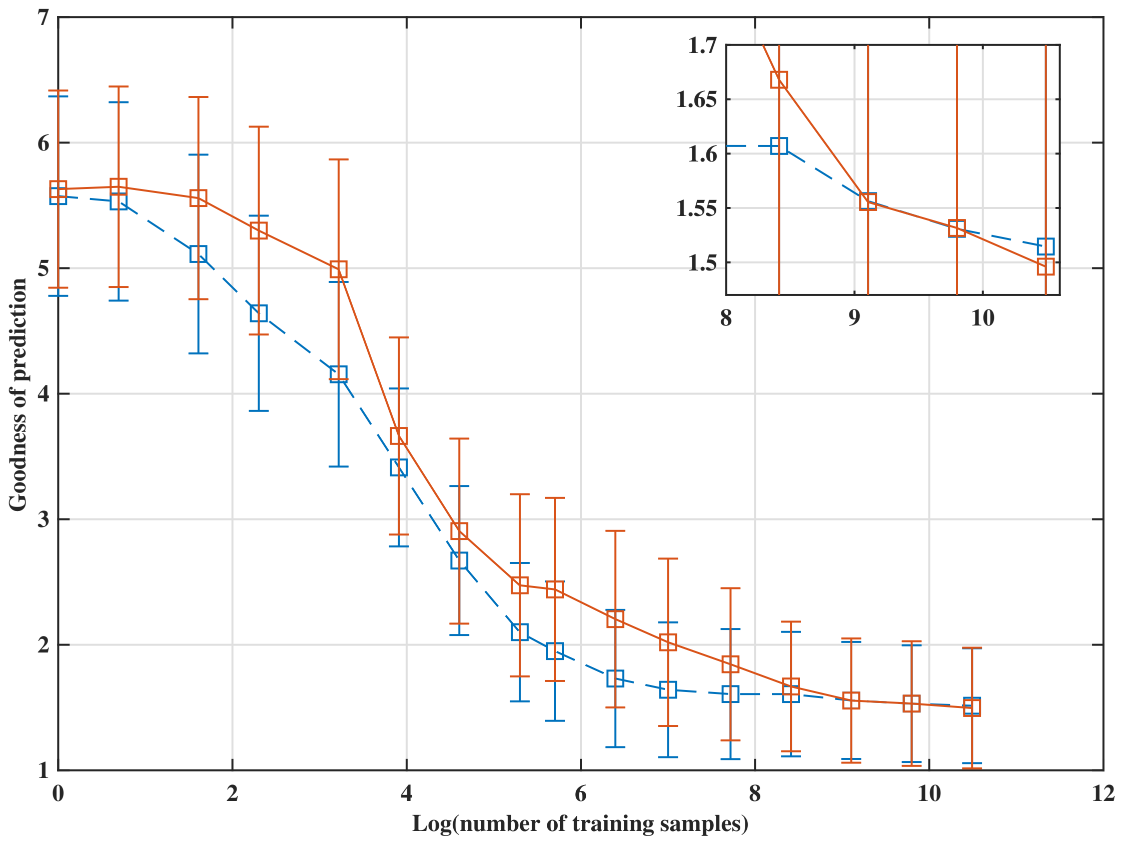


Figure 10: On Y-axis we have the mean goodness of prediction. The dashed curve is the goodness of prediction from non-physics informed neural network, the solid curve is from physics informed neural network.

5.5 Effect of different number of neural network layers

Another factor that affected the goodness of prediction is the number of layers of neurons used in the neural network. We started with 8 neural layers but when the number of layers increased from 8 to 12 the PINN outperformed the noPINN on goodness of prediction and outperformed goodness of prediction of the same PINN but with only 8 neural layers, as seen in fig.11.

6 Discussion

In this section we discuss the implication of observations, limitations and improvements that could be made to make physics informed neural network work better.

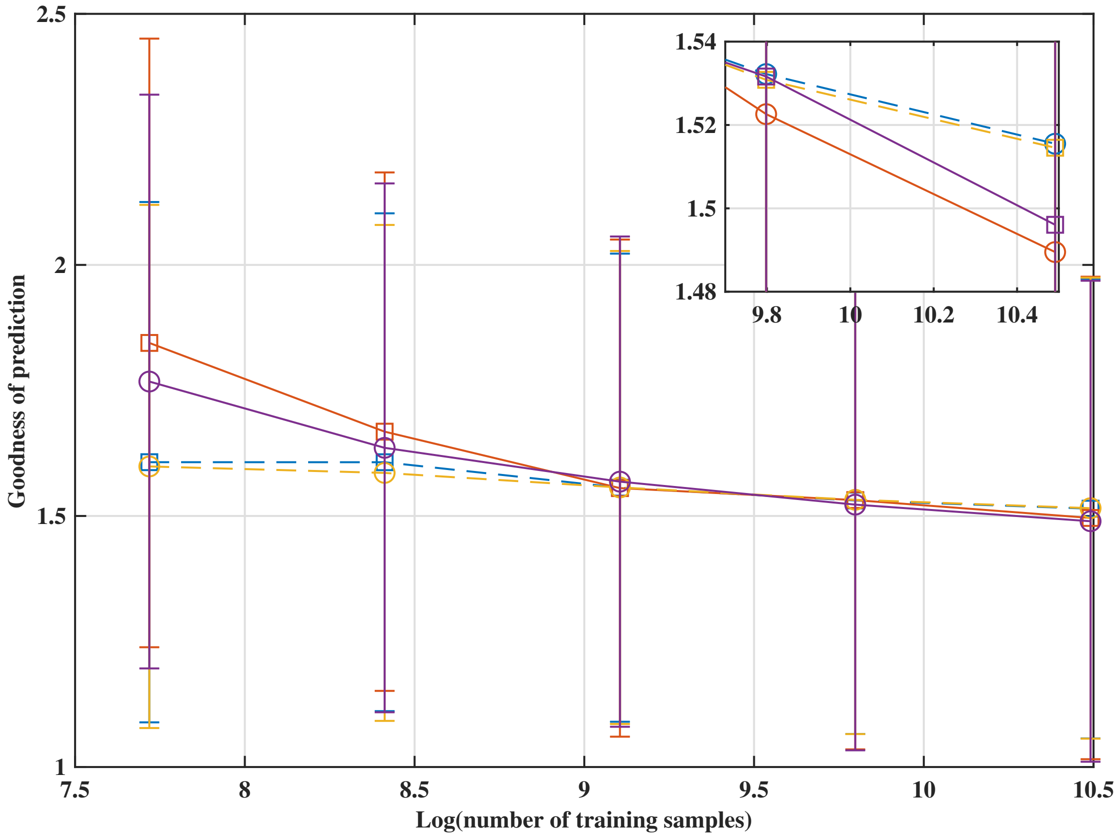


Figure 11: On Y-axis we have the goodness of prediction. The dashed curve is from non-physics informed neural network, the solid curve is from physics informed neural network. The round data points are from 12 neural network layers and the square data points are from 8 neural network layers.

6.1 PINN V.S. noPINN

6.1.1 Effect of more training information and more neural layers

As we can see in fig.10 and fig.11. non-physics informed neural network results in a better goodness of prediction than physics informed network when there are less information to work with. When there are more information, either with a larger training sample size or a deeper neural network structure, the physics informed neural network out performs the non-physics informed neural network.

This means that the extra neurons and extra training samples helped in learning the features of the physics information. This also means that there are more useful information to learn in the physics informed neural network than the non-physics informed neural network.

When only comparing goodness of prediction from physics informed neural networks the extra neural network layers have also helped in getting a better goodness of prediction. This came in contrast with the effect of the extra neural network layers when looking only at non-physics informed neural networks. Results from the non-physics informed neural networks did not improve with extra layers of neurons. Which means that the non-physics informed network have reached the limit of how much it can learn about the problem from a pure statistical point of view.

6.1.2 Number of frequencies

One argument could be made for the sub-optimal performance of the physics informed neural network, that is we are only enforcing the physics information on selected frequencies, in the case of fig.9, 15 out of 2049 frequencies in the frequency space. While the non-physics informed neural network, which does not work in frequency space, in principal works with 100% of the frequencies. So in a way the the physics informed neural network was able to achieve the same result with 0.732% of the constrain is impressive.

6.1.3 Statistical limitation revealed by out of boundary test

We have also challenged our best trained neural network, both physics informed and non-physics informed, for predictions using samples that are out of boundaries of training samples. This means that the neural networks were trained using samples with impedance values within the range of 2.2197 and 5.8785. We now input samples with impedance values that are generated within the range of 1 and 9. This means some impedance values will be out side the range the network was trained to recognize. We then compare the goodness of prediction for both physics informed neural network and non-physics informed neural network. The expectation is that the physics informed neural network will be able to predict better than non-physics informed neural network due to the fact that the laws of physics that the neural network was suppose to learn works across different ranges of impedance values. Hence if the network was able to encode the laws of physics in its weights and biases then it should out perform the non-physics informed neural network. The result however was that both network performed worse than they did originally with a both goodness of prediction around 200%.

This reveals that the network even though learnt about the laws of physics behind the problem formulation, a statistical approach still dominated the results of this test. The limitation of a statistical approach is of course that there will be problems when dealing with previously unseen data like we have seen here.

However, one of the limitation of this out of boundary test is that for our new input data not to be a subset of training data we had to expand the range of impedance values from 2.2- 5.8 to 1-9 which made the seismic wave much more non-linear. And as we can see in fig.8 The more non-linear the worse the goodness of prediction. So the inferior performance can also be attributed to this enlarged non-linearity.

6.1.4 Training time

The training time for non-physics informed network ranges from seconds to a few minutes while training time for physics informed neural network ranges from half an hour to a few hours.

From a practical view point the difference in training time is directly related to the extra calculations involved in the custom defined loss function in the physics informed network. The more frequencies we evaluate the physics informed network in, the longer the training time. We also suspect that the calculations within our custom loss is too complex that the program used the CPU to process it instead of the GPU, hence contributing to the longer training time.

From a philosophical point of view, the longer training time might be tied to the discussion of number of frequencies used in training in section 6.1.2. Solving an inverse problem with sufficient information, in this case sufficient number of frequencies, usually takes less time than solving an inverse problem with not enough information. The non-physics informed network can be considered using all the frequencies in the frequency space to solve the inverse problem hence it takes less time while the physics informed network only uses a few frequencies hence it takes a longer time to solve.

6.1.5 Similarities between two methods

The fact that the two vastly different approaches in training results in strikingly similar results is very encouraging for our approach. A comparison of a non-physics informed prediction and a physics informed prediction with 12 neural network layers and the most amount of training samples can be seen in fig.12.

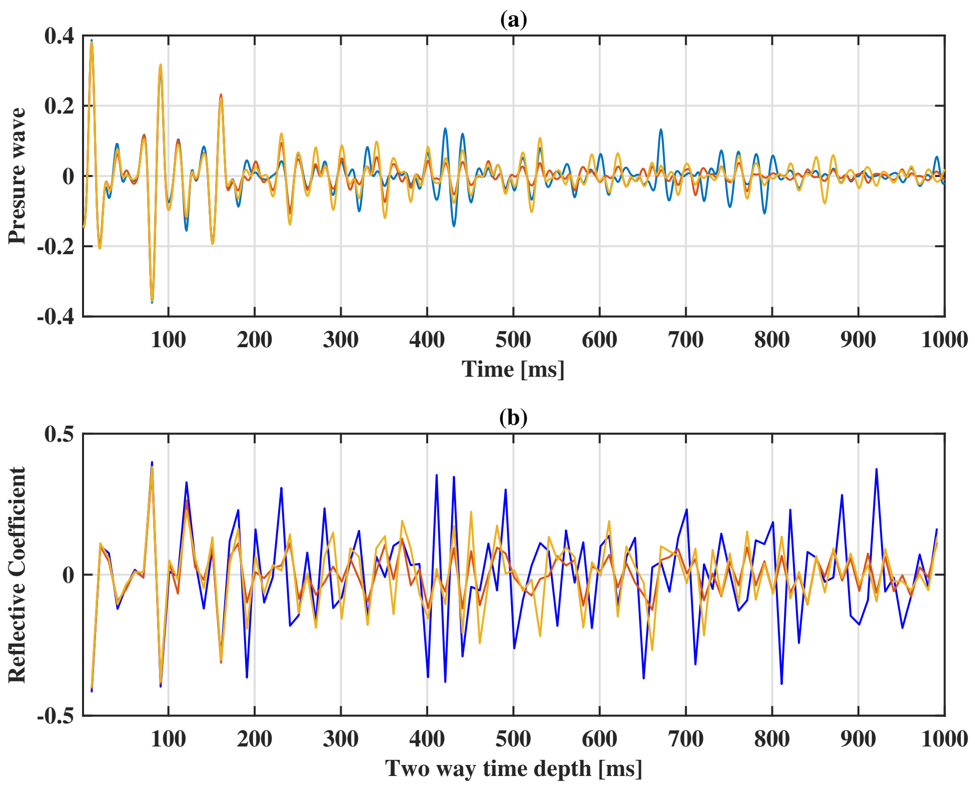


Figure 12: (a) The blue line is the true data from true reflective coefficient structure, red line is the data from non-physics informed prediction while yellow line is the data from physics informed prediction. (b) blue line is the true reflective coefficient, red is the non-physics informed prediction and yellow is the physics informed prediction.

The fact that the predicted reflective coefficient with or without physics information is in phase with each other towards the right of graph fig.12(b) but out of phase with the true reflective coefficient that they are suppose to predict, may suggest that there is a deeper connection between the two methods than there appears to be.

One explanation of this similarity could be that it is the result of the input data we used to generate the output prediction which in this case are both the seismic wave in time domain. Upon closer inspection the phase of the predicted reflection coefficients is somewhat similar to the phase of the seismic wave. The seismic wave however is not in phase with the true reflection coefficients due to non-linearity caused by reflections between layer boundaries. The correlation between the input data and the reflection coefficients can be seen in fig.13.

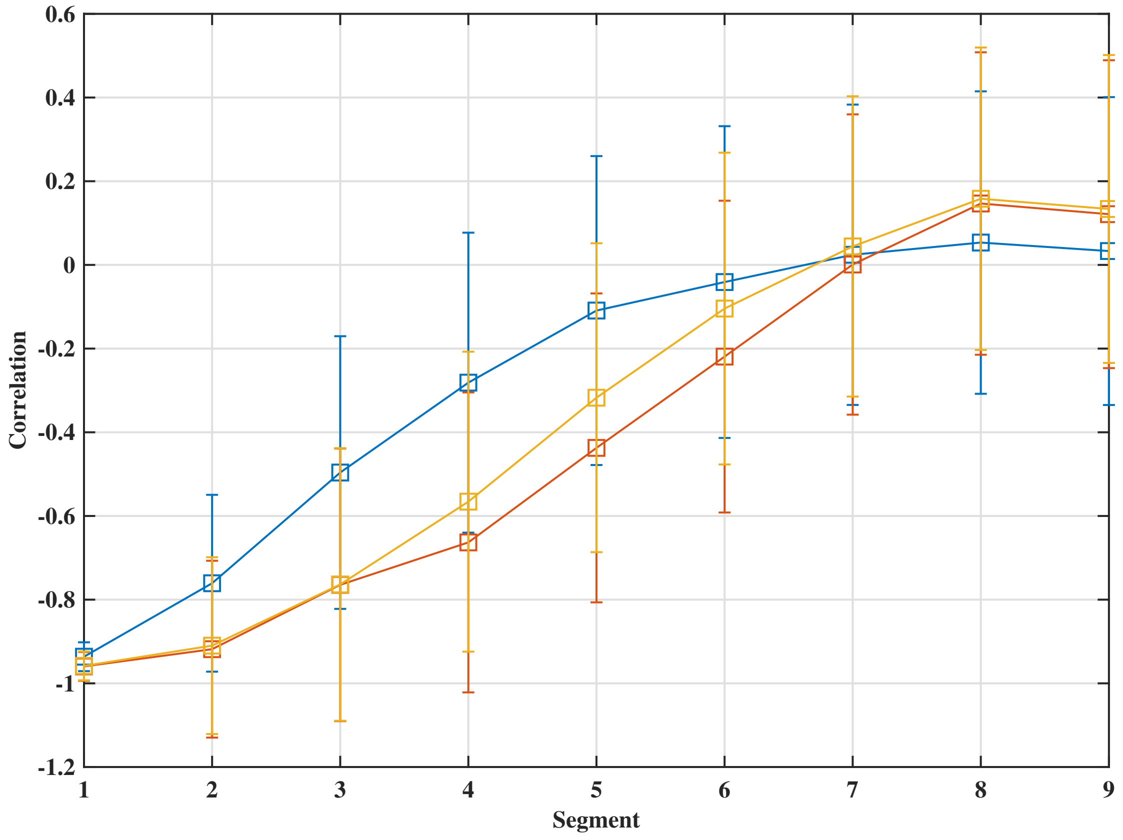


Figure 13: On y-axis we have the correlations between the input data and the reflection coefficients within each segment, on x-axis we have 9 segments of the 99 reflection coefficients each segment contains 11 reflection coefficients. Each reflection coefficients has a corresponding value from the input data at the same two way time depth as the reflection coefficients. Blue line is the correlation between input data and true reflection coefficients, yellow line is the correlation between input data and predicted reflection coefficients from noPINN and red line is the correlation between input data and predicted reflection coefficients from PINN

As shown in fig.13. All the reflection coefficients have a good absolute correlation with the input data in the beginning towards the left, segment 1 contains 11 reflection coefficients between time depth 10 [ms] and 110 [ms]. This supports the fact that non-linearity comes from wave reflections between layer boundaries. As these reflections are not the dominating signal in the beginning but towards the end to the right.

As time proceeds we can see that the absolute correlation between true reflection coefficients and input data drops faster than all the predicted reflection coefficients. This supports our theory that the predictions are constrained by the limited ability of the network to map input to output in a non-linear fashion. Hence the predicted reflection coefficients are able to maintain a better absolute correlation to the input data while the absolute correlation between true reflection coefficients and input data drops as non-linearity increases due to reflections between layer boundaries.

Towards the right of fig.13. We see that the true reflection coefficients are no longer correlated to the input data while the correlation between predicted reflection coefficients and input data went from negative to positive. Which supports our previous observation that the predicted reflection coefficients are in phase with each other and out of phase with the true reflection coefficients towards the right.

Another observation could be made that towards the right predictions from the PINN has a lower correlation than predictions from noPINN which means that PINN was able to learn the non-linearity slightly better than noPINN however in the middle part of fig.13, the noPINN ends up with lower absolute correlation than PINN which suggests the opposite. This observation again supports our theory that the predictions are influenced by the input data.

This mapping limitation from input to output might suggest that for a feed forward fully connected neural network to solve non-linear inverse problems there will be a need for training for each specific case and a deeper integration of physics information just as described in section 1.2.1 and appendix A.

6.2 Combination of PINN and noPINN

The idea is simple, to harness the speed of the non-physics informed network and the knowledge of the physics informed network. There are mainly two ways to achieve it.

The first way is somewhat related to a concept named transfer learning in the machine learning community which is to train the network first with non-physics information, save the weights and biases and then train the network with physics information with the saved weights and biases as a starting point. There is one major concern about this method that prevented our further exploration. The trained network with on physics information could result in a starting point for the physics informed network at a local minimum. Hence the physics informed network would not be able to learn much about the physics information with the gradient decent methods used by the optimizer.

Another way of training is to train the network using the loss function presented in eq.28.

$$Loss = PI + MSE \tag{28}$$

Here PI denotes Physics information.

The idea of a loss function consisting of both physics information and non-physics information is interesting and is slightly touched during in this thesis. A concern of this method is the weight set up for physics information term and non-physics information term. A biased weight could mean the network would end up towards a physics informed or a non-physics informed scenario. Another concern for this method is tied to the previous discussions about how the non-physics informed network makes use of all the frequencies in the frequency space and the physics informed network works with only a few frequencies. The combination of the two would potentially mean double training on selected frequencies.

Both combination methods are explored during the making of this thesis and the preliminary results did not set these combined methods apart from the main method hence these possibilities are not explored further.

This leaves us with the physics informed network using only physics information and the non-physics informed network using no physics information. This is a more fundamental comparison that clearly examines the performance and accuracy of two different approaches and that is what we chose to do.

6.3 Goodness of prediction as a percentage error

The goodness of prediction defined in section 5.1 is the mean of the sum of squared residual. This goodness value is absolute and not relative. A more relative approach to calculate the goodness of prediction takes inspiration from percentage error and modifies eq.26 and eq.27 to the following

$$Goodness = \frac{1}{N}\sum^{N}\frac{\sum residual^2}{\sum amplitude^2} \tag{29}$$

$$Error = \sigma(\frac{\sum residual^2}{\sum amplitude^2}) \tag{30}$$

The resulting corresponding figures of fig.8, fig.10 and fig.11 can be found in appendix.C.

6.4 Direct prediction of impedance values

As seen in fig.4, the predicted impedance values are around the mean of all the impedance values used for training. The fact that it is not around zero, but with a starting value close to the true impedance values is an encouraging sign as neural networks tends to like normalized values.

6.5 Minimization of physics information

6.5.1 Networks predicting impedance values

Early on during this thesis our networks were trying to predict impedance values directly, for a setup with 1000 numerically discretized layers with 5 distinct layers with different impedance values, and that posed a few challenges as discussed in section 3.1.2. The physics information described in section 4.1 was imposed on the network and the optimizer tried to minimize the physics information. The result is shown in fig.14.

There are a few differences here compared to networks predicting reflection coefficients. First, the resulting values of physics information is not periodic as for network predicting reflection coefficients as described in section 4.7. Second, and here is the interesting part, the physics informed network tried to minimize the frequency at the local minimum at $\omega = 2845$. It succeeded in the way that there is a local minimum at

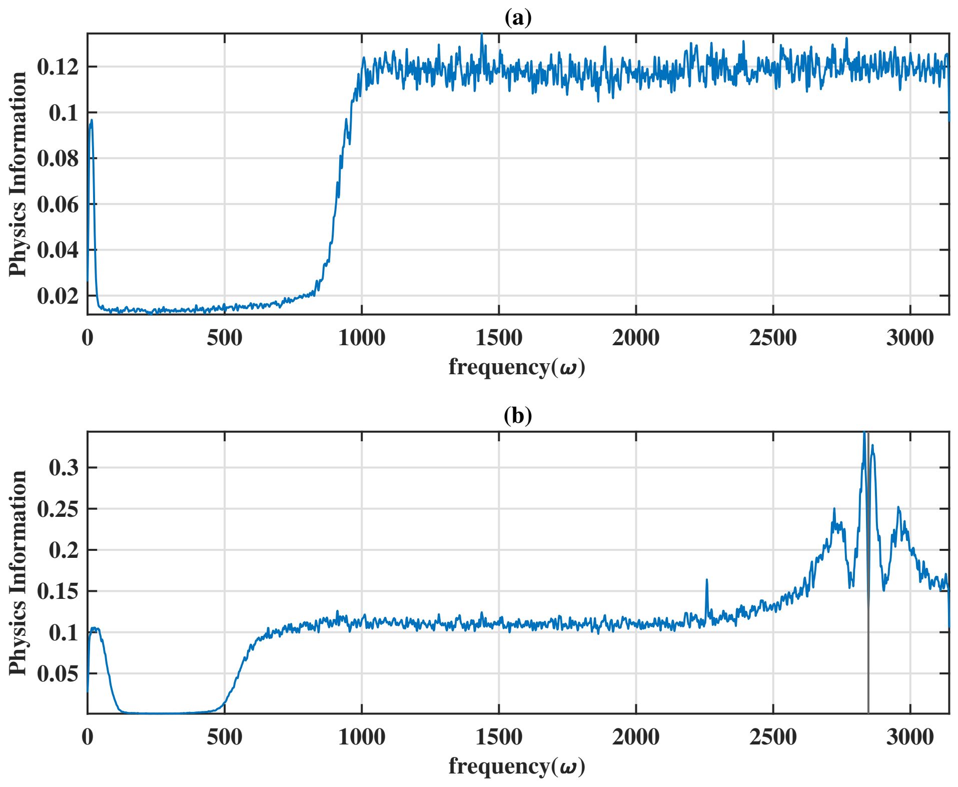


Figure 14: Both curves in (a) and (b) are the results of the physics information in section 4.1 from networks predicting impedance values. (a) From non-physics informed network. (b) From physics informed network. The black vertical line denotes the frequency the physics information was applied.

that specific frequency. But it failed in the sense that it formed the local minimum by elevating the values of the physics information for the surrounding frequencies.

6.5.2 Networks predicting reflection coefficients

Although the results of physics informed neural network were not substantially different than non-physics informed neural network, but contrary to the networks predicting impedance values, the fact that the optimizer was able to minimize, to a certain point, the physics information at selected frequencies and their surroundings, see fig.9, may suggest that the network was able to approach the problem from the perspective of the physics information. Which is encouraging.

And comparing the results from networks predicting impedance values and reflective coefficients, the choice of predicting reflective coefficient clearly made the problem easier to tackle for the optimizer.

6.6 Limitations and improvements

A major limitation that could be improved right out of the box is the limitation of computation power. More computation power would make applying the physics information for every frequency possible and would hopefully improve the results of physics informed neural network.

Another limitation is that we have only looked at one neural network structure which is the full connected feed forward neural network. Other network types and structures, such as Bayesian convolutional neural networks that were used to classify seismic facies [7], may help improve the results and error estimations from both physics informed and non-physics informed neural network.

Yet another limitation involving the neural network setup is the use of early stop method and subsequently revert weights and biases to the epoch with the best loss value. This is somewhat problematic when the loss function does not provide a full picture of the misfit of the data from the predictions. This means even though the loss value could be reverted to when it is minimized that does not necessarily mean the total misfit is at its minimum. This limitation can also be solved by the use of more computation power to directly calculate the misfit in the loss function.

As discussed in section 6.1.5. Both predictions from physics informed neural network and non-physics informed neural network are quit limited by the input data and the limited ability of the feed forward fully connected neural network to learn the non-linearity of the problem. Hence the predictions are somewhat a linear projection of the input data. This limitation might be solved by exploring other neural network structures.

7 Further Study

It could be a good idea to take a deeper look behind the following phenomena shown in this thesis in order to gain a better understanding of how the physics information and neural network work together.

- Formation of local minimums at specific frequencies of physics information as described in section 4.7.2.
- Offset between some local minimums and frequencies where physics information were applied.
- The minimums between $\omega = 0$ and $\omega = 500$ in fig.14.
- The elevated physics information surrounding the physics constrained frequency in fig.14.

8 Conclusion

We explored the possibility of implementing different physics information through the loss function of a fully connected feed forward neural network in order to solve for the impedance structure beneath the surface using seismic waves in time domain observed on the surface. We then compared the performance of the physics informed network with non-physics informed network. It turns out that the physics informed neural network slightly out performs the non-physics informed neural network when there is a large number of training samples and a sufficient amount of neural network layers despite the limitation of the physics information only being applied to less than 1% of all the available frequencies in frequency space.

9 Acknowledgement

I would like to thank my supervisor Professor Klaus Mosegaard for guidance throughout this project and for providing the MATLAB code for generating synthetic seismic waves and Thomas Martini Jørgensen for his engagement and consultation on the neural networks part of this project. I would also like to thank Runhai Feng as well as Pernille Lundsgaard Rasmussen for sharing some of their previous works in this field with me. Lastly, I would like to thank Michael Nathaniel Dyonisius for proofreading.

References

[1] M. Valueva, N. Nagornov, P. Lyakhov, G. Valuev, and N. Chervyakov, “Application of the residue number system to reduce hardware costs of the convolutional neural network implementation,” *Mathematics and Computers in Simulation*, vol. 177, pp. 232–243, 2020.

[2] H. Hewamalage, C. Bergmeir, and K. Bandara, “Recurrent neural networks for time series forecasting: Current status and future directions,” *International Journal of Forecasting*, vol. 37, no. 1, pp. 388–427, 2021.

[3] M. Raissi, P. Perdikaris, and G. Karniadakis, “Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations,” *Journal of Computational Physics*, vol. 378, pp. 686–707, 2019.

[4] T. Kadeethum, T. M. Jørgensen, and H. M. Nick, “Physics-informed neural networks for solving nonlinear diffusivity and biot’s equations,” *PLOS ONE*, vol. 15, pp. 1–28, 05 2020.

[5] K. Mosegaard and T. M. Hansen, “Inverse methods: Problem formulation and probabilistic solutions,” *Geophysical Monograph Series, John Wiley and Sons*, vol. 218, pp. 9–27, 2016.

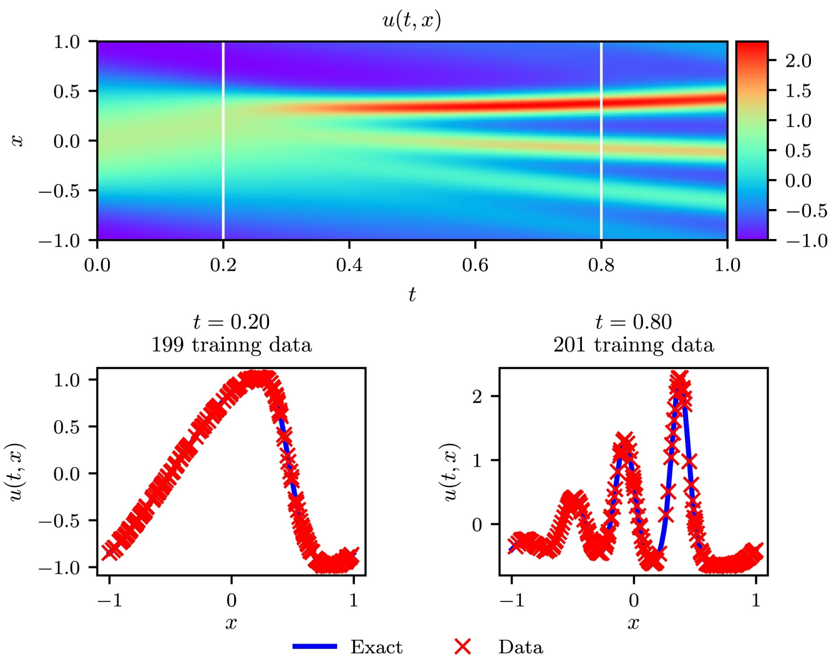
[6] D. C. Ganley, “A method for calculating synthetic seismograms which include the effects of absorption and dispersion,” *Geophysics*, vol. 46, pp. 1100–1107, 08 1981.

[7] R. Feng, N. Balling, D. Grana, J. S. Dramsch, and T. M. Hansen, “Bayesian convolutional neural networks for seismic facies classification,” *IEEE Transactions on Geoscience and Remote Sensing*, pp. 1–8, 2021.

A Technical Details Concerning Implementation of PINN With Direct Manipulation of Parameters

For this PINN, we have one shared set of weights W and biases B filled with truncated normally distributed random numbers and trainable parameter λ_1 and λ_2 with some starting value.

Consider we are given the wave values u at two time instances $t = 0.2$ and $t = 0.8$ for all x, 199 and 201 data points are extracted across x at these two time instances to be feed into the share weights and biases in the PINN, as shown in fig.15[3]



Correct PDE	$u_t + uu_x + 0.0025u_{xxx} = 0$
Identified PDE (clean data)	$u_t + 1.000uu_x + 0.0025002u_{xxx} = 0$
Identified PDE (1% noise)	$u_t + 0.999uu_x + 0.0024996u_{xxx} = 0$

Figure 15: KdV equation: Top: Solution u(t,x) along with the temporal locations of the two training snapshots. Middle: Training data and exact solution corresponding to the two temporal snapshots depicted by the dashed vertical lines in the top panel. Bottom: Correct partial differential equation along with the identified one obtained by learning λ_1,λ_2

For simplicity purposes, let us first focus on the 199 wave values $u_{1-199}(t = 0.2, x_{1-199})$ and the corresponding values of x_{1-199}.

The shared weights has 5 layers the form of a list of 5 weight matrices [(1, 50),(50, 50),(50, 50),(50, 50),(50, 50)] and the shared biases has 5 layers the form of a list of 5 bias matrices [(1, 50),(1, 50) ,(1, 50) ,(1, 50) ,(1, 50)]. The 5 matrices represent the 5 layers of this fully connected feed forward neural networks.

For $u_{1-199}(t = 0.2, x_{1-199})$, the input to the network are x_{1-199} as a column vector "$CV1$" of size [199, 1]. The operations for the first layer of the neural networks are as follow with some help of pseudo code,

1. Values in $CV1$ is normalized between -1 and 1 to get $CV2$ of size [199, 1].
   - $CV2 = 2 * (CV1 - min(CV1))/(max(CV1) - min(CV1)) - 1$
2. $CV2$ multiplies with first layer of weights $W1$ of size [1, 50] to get a matrix $M1$ of size [199, 50].
   - $M1 = CV2 * W1$
3. Each column of $M1$ is added with each column of the first layer of biases $B1$ of size [1, 50] to get $M2$ of size [199, 50]
   - $M2 = M1 + B1$
4. Values in $M2$ is then fed into a hyperbolic tangent function to get $M3$ of size [199, 50]
   - $M3 = tanh(M2)$

Operations from the second layer of the neural networks are as follow,

5. $M3$ multiplies with second layer of weights $W2$ of size [50, 50] to get $M4$ of size [199, 50].
   - $M4 = M3 * W2$
6. Each column of $M4$ is added with each column of the second layer of biases $B2$ of size [1, 50] to get $M5$ of size [199, 50]
   - $M5 = M4 + B2$
7. Values in $M5$ is then fed into a hyperbolic tangent function to get $M6$ of size [199, 50].
   - $M6 = tanh(M5)$

The operations for the rest of layers of the neural network is the repeat of the second layer. After all 5 layers the final out come is a matrix U of size [199, 50]. At this stage matrix U is assumed to be a representation of $u(t, x)$ in the KdV equation.

This matrix is then operated on with the following steps,

8. U is passed through a tensorflow function using automatic differentiation to get U_x and U_{xxx} of the sizes [199,50].

- $U_x = tf.gredient(U)$
- $U_{xx} = tf.gredient(U_x)$
- $U_{xxx} = tf.gredient(U_{xx})$

9. U_x and U_{xxx} are then used to construct U_t following eq.4. Multiplication between U, U_x and U_{xxx} is element-wise at this step. U_t has the size [199,50].

   - $U_t = -\lambda_1 U U_x - \lambda_2 U_{xxx}$

10. U_t along with U are then used to calculate the final matrix U_1 of size [199,50], which will be used to calculate the value of the loss function in eq.31. Each column in U_1 represents a neural networks prediction of $u(t = 0.2, x_{1-199})$. Δt is the time difference between the two sampled times which is 0.6. $IRKweights_1$ is a custom layer of weights of size [50,50] which remains unchanged during training.

    - $U_1 = U - \Delta t U_t * IRKweights_1$

For $u_{1-201}(t = 0.8, x_{1-201})$, the input to the network is x_{1-201}. The operations for this case is exactly the same from step 1 to 9. Step 10 is replaced by step 11 where U_2 is the outcome.

11. In this case, U_t along with U of size [201,50] are used to calculate U_2 of size [201,50], which will be used to calculate the value of the loss function in eq.31. Each column in U_2 represents a neural networks prediction of $u(t = 0.8, x_{1-201})$. Δt is the time difference between the two sampled times which is 0.6. $IRKweights_2$ is a custom layer of weights of size [50,50] which remains unchanged during training.

    - $U_2 = U - \Delta t U_t * IRKweights_2$

At this stage the PINN has spit out U_1 and U_2 as predictions of $u(t = 0.2, x_{1-199})$ and $u(t = 0.8, x_{1-201})$. The outcomes are used to construct the loss function which is defined as the sum of the squared residuals of each column of U_1 subtracted by the column vector containing $u(t = 0.2, x_{1-199})$, plus the sum of the squared residuals of each column of U_2 subtracted by the column vector containing $u(t = 0.8, x_{1-201})$

$$loss = sum((U_1 - u(t = 0.2, x_{1-199}))^2) + sum((U_2 - u(t = 0.8, x_{1-201}))^2) \tag{31}$$

During training loss value is minimized by the optimizer manipulating weights W and biases B plus the two inverse problem parameters λ_1 and λ_2.

B Supplementary Figures of Results With Different PINNs

B.1 Wave ratio Y at the bottom layer

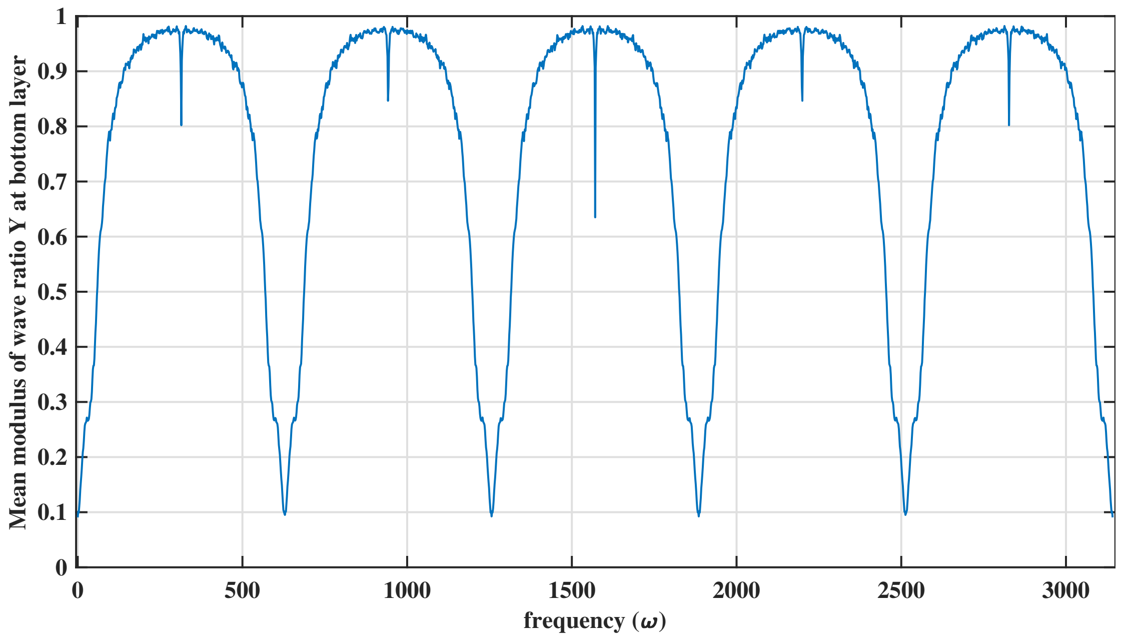


Figure 16: The mean modulus of wave ratio Y at bottom layer of the impedance structure, calculated using non-physics informed predictions from neural network trained using impedance layer structure with 100 distinct impedance values.

B.2 Residual of wave ratio at the top layer

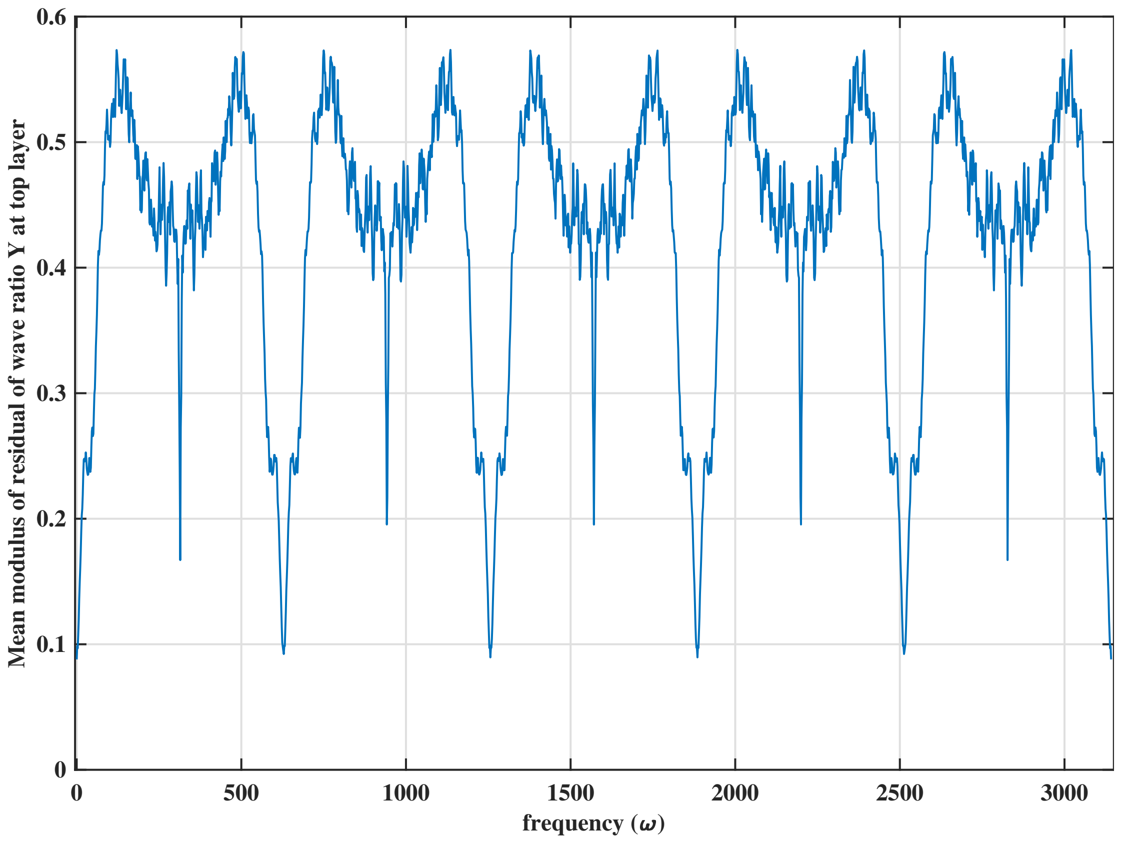


Figure 17: The mean modulus of wave ratio Y at top layer of the impedance structure, calculated using non-physics informed predictions from neural network trained using impedance layer structure with 100 distinct impedance values.

B.3 Up going wave at the bottom layer

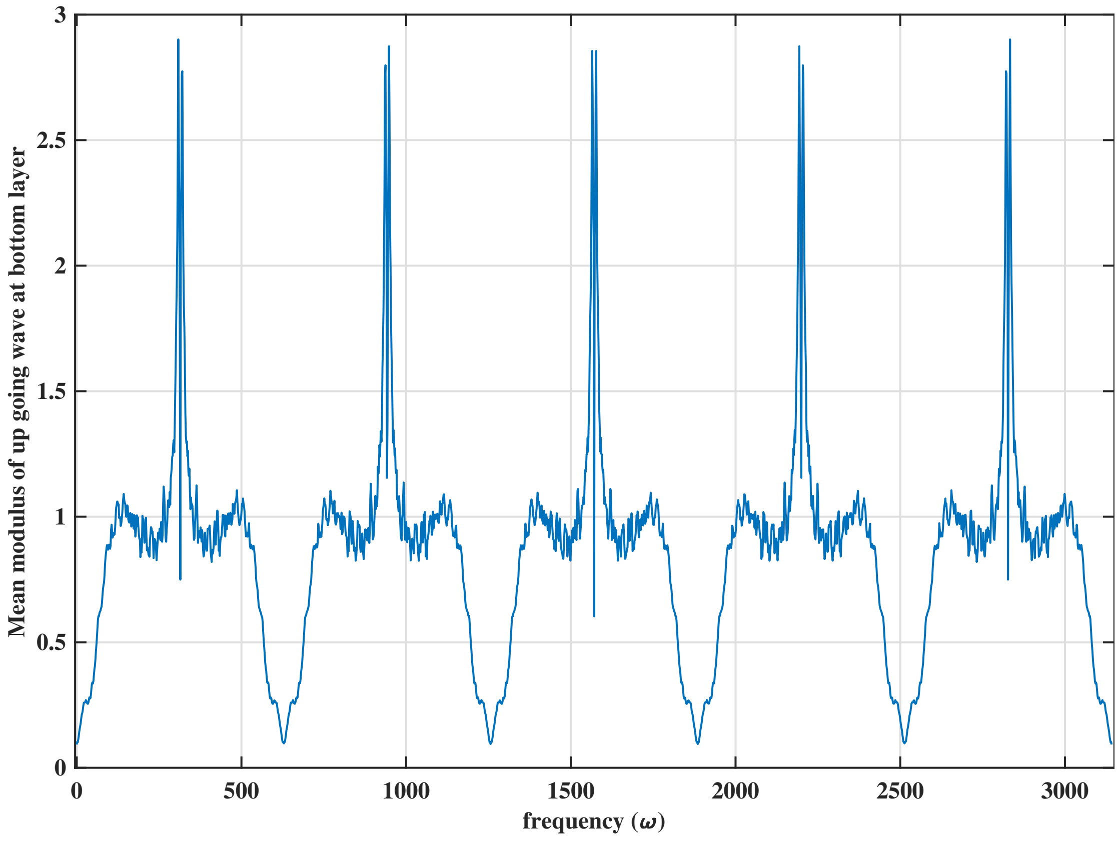


Figure 18: The mean modulus of up going wave at bottom layer of the impedance structure, calculated using non-physics informed predictions from neural network trained using impedance layer structure with 100 distinct impedance values.

B.4 Sum of residuals of waves at every layer

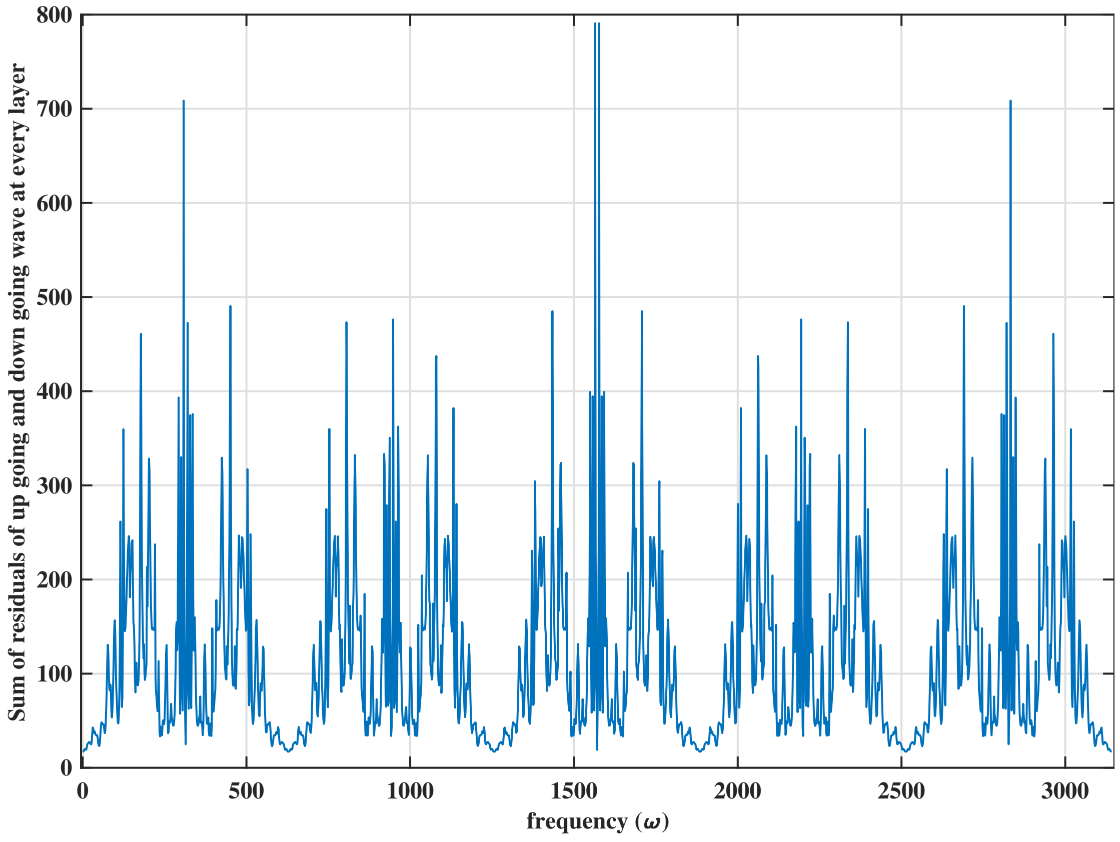


Figure 19: The mean modulus of the sum of residuals of waves at every layer of the impedance structure, calculated using non-physics informed predictions from neural network trained using impedance layer structure with 100 distinct impedance values.

B.5 Sum of residuals of wave ratios at every layer

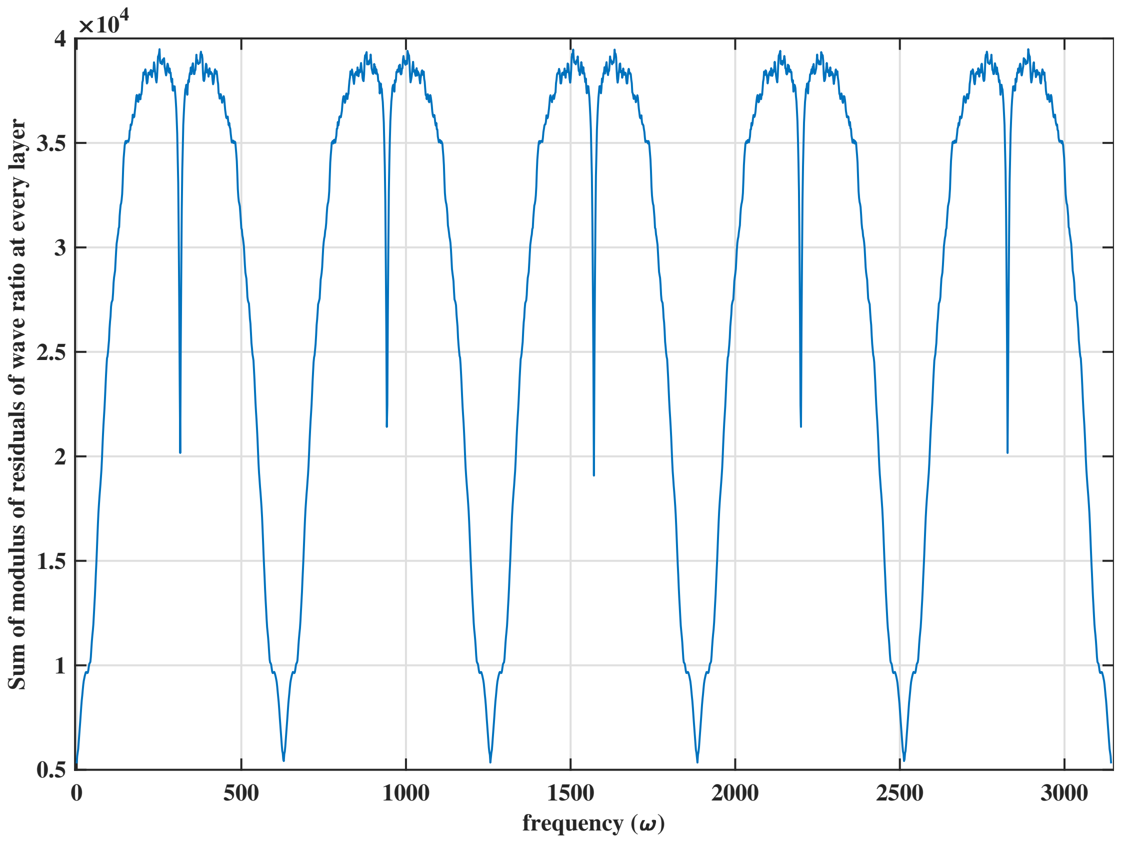


Figure 20: The mean modulus of the sum of residuals of wave ratios at every layer of the impedance structure, calculated using non-physics informed predictions from neural network trained using impedance layer structure with 100 distinct impedance values.

C Percentage Goodness

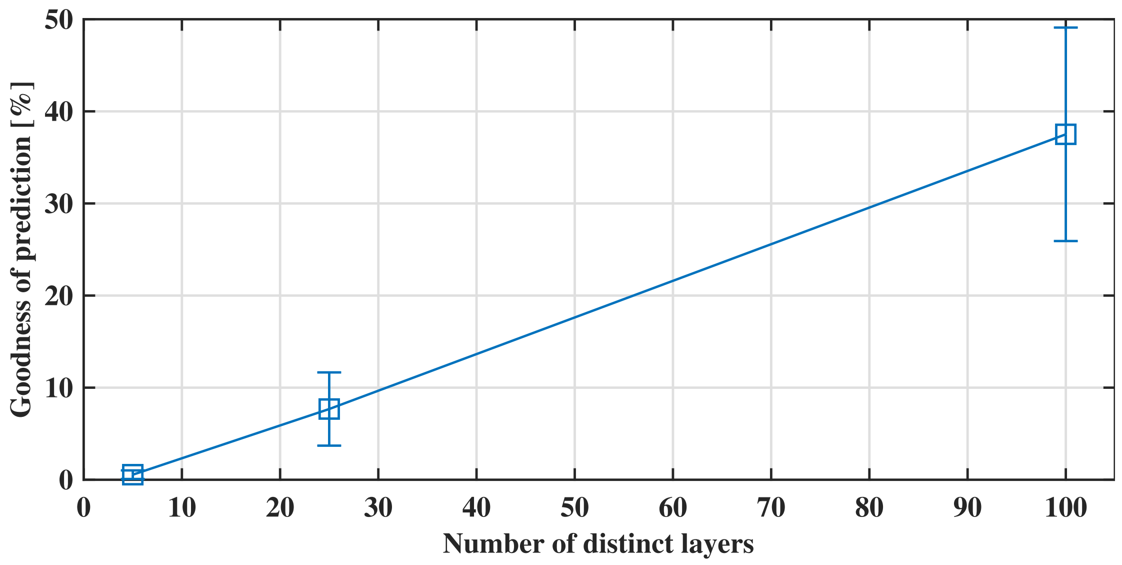


Figure 21: This figure shows the percentage goodness of prediction as a function of number of layers with distinct impedance values. The respective error bars are calculated as shown in eq.30.

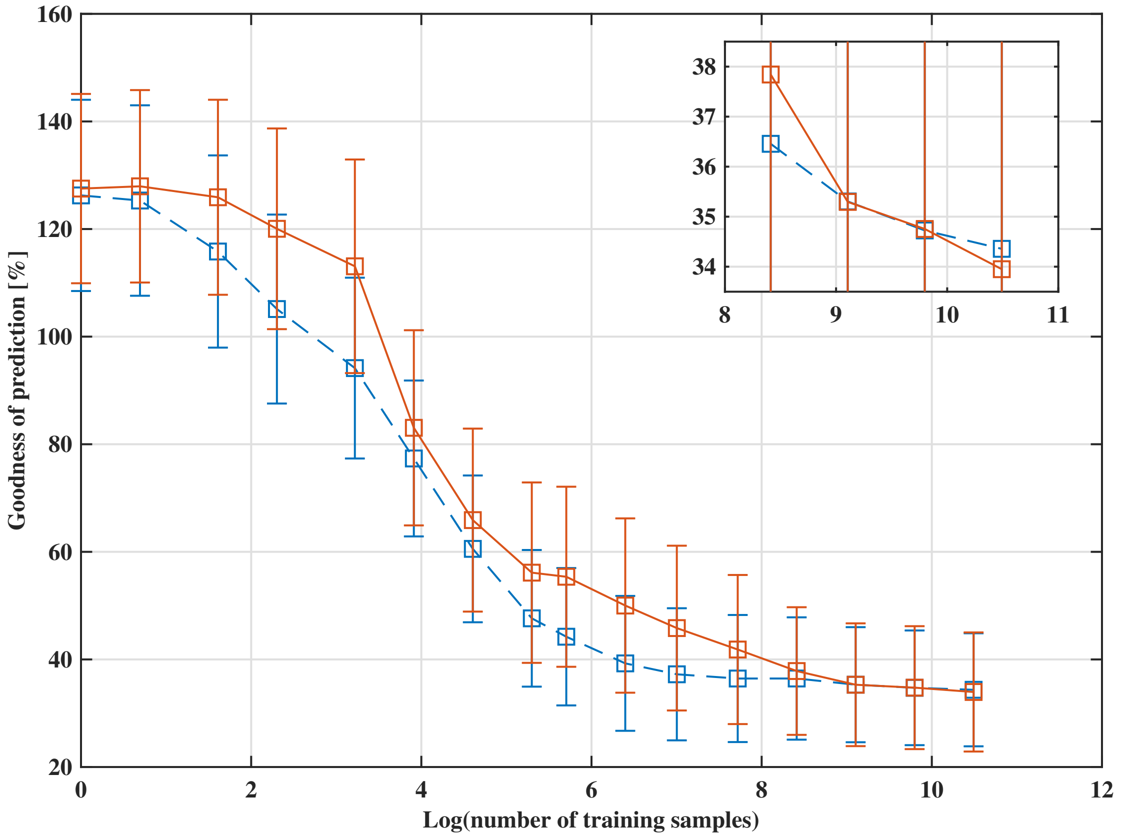


Figure 22: On Y-axis we have the percentage goodness of prediction. The dashed curve is the goodness of prediction from non-physics informed neural network, the solid curve is from physics informed neural network.

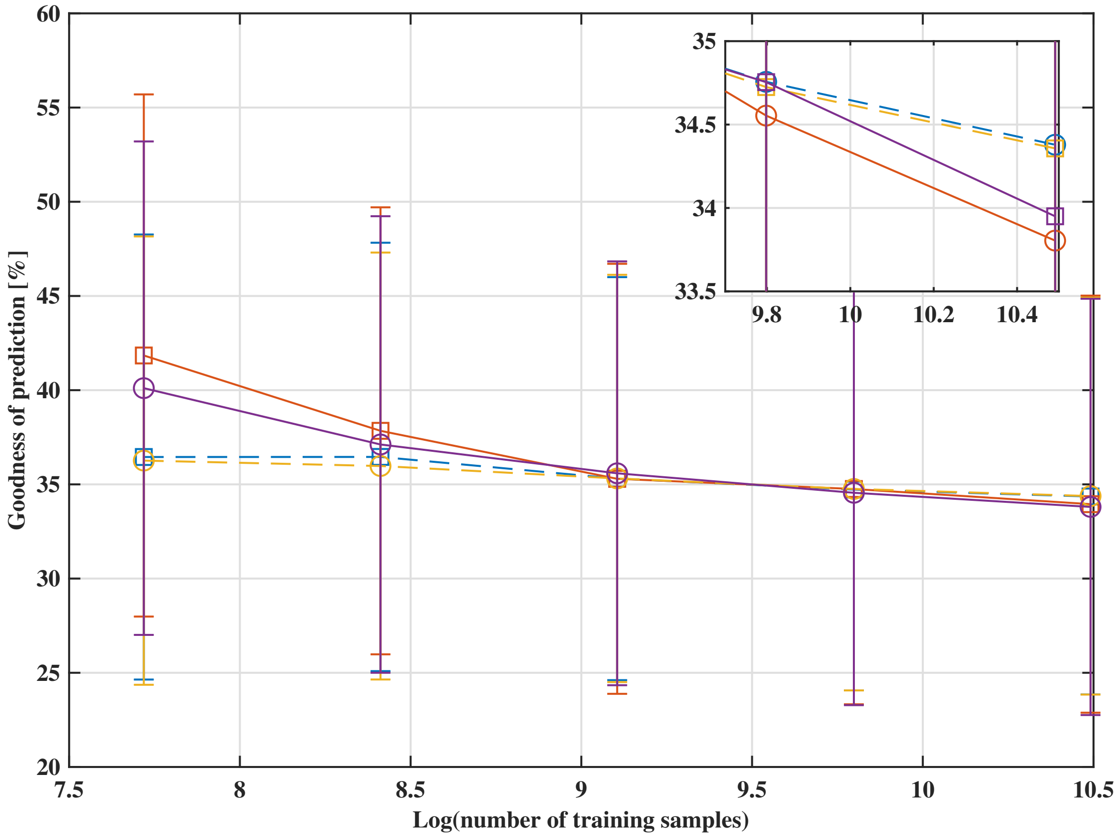


Figure 23: On Y-axis we have the percentage goodness of prediction. The dashed curve is from non-physics informed neural network, the solid curve is from physics informed neural network. The round data points are from 12 neural network layers and the square data points are from 8 neural network layers.