PAPER • **OPEN ACCESS**

Towards reflectivity profile inversion through artificial neural networks

To cite this article: Juan Manuel Carmona Loaiza and Zamaan Raza 2021 *Mach. Learn.: Sci. Technol.* **2** 025034

View the article online for updates and enhancements.

MACHINE
LEARNING
Science and Technology

OPEN ACCESS

RECEIVED
30 July 2020

REVISED
24 January 2021

ACCEPTED FOR PUBLICATION
11 February 2021

PUBLISHED
21 April 2021

PAPER

Towards reflectivity profile inversion through artificial neural networks

Juan Manuel Carmona Loaiza and Zamaan Raza

Jülich Centre for Neutron Science (JCNS) at Heinz Maier-Leibnitz Zentrum (MLZ), Lichtenbergstraße 1, 85748 Garching, Germany

E-mail: j.carmona.loaiza@fz-juelich.de

Keywords: neutron scattering, reflectometry, inverse problems, machine learning, algorithms, neural networks , data science

Abstract

The goal of specular neutron and x-ray reflectometry is to infer a material's scattering length density (SLD) profile from its experimental reflectivity curves. This paper focuses on the investigation of an original approach to the ill-posed non-invertible problem which involves the use of artificial neural networks (ANNs). In particular, the numerical experiments described here deal with large data sets of simulated reflectivity curves and SLD profiles, and aim to assess the applicability of data science and machine learning technology to the analysis of data generated at large-scale neutron scattering facilities. It is demonstrated that, under certain circumstances, properly trained deep neural networks are capable of correctly recovering plausible SLD profiles when presented with previously unseen simulated reflectivity curves. When the necessary conditions are met, a proper implementation of the described approach would offer two main advantages over traditional fitting methods when dealing with real experiments, namely (1) sample physical models are described under a new paradigm: detailed layer-by-layer descriptions (SLDs, thicknesses, roughnesses) are replaced by parameter-free curves $\rho(z)$, allowing *a priori* assumptions to be used in terms of the sample family to which a given sample belongs (e.g. 'thin film,' 'lamellar structure',etc.); (2) the time required to reach a solution is shrunk by orders of magnitude, enabling faster batch analysis for large datasets.

1. Introduction

Neutron and x-ray specular reflectometry are established experimental techniques whose aim is to investigate interfacial structures at the sub-nanometer scale through the measurement and analysis of reflectivity curves [1–3].

In a typical specular reflectometry experiment, a collimated neutron or x-ray beam of wavelength λ impinges on the surface of a flat sample at an incident angle θ. The incident angle is varied and the specular reflectivity is measured as the ratio of the reflected and incident beam intensities, $R(\theta) = I_R(\theta)/I_0(\theta)$.

Theoretically, in the absence of significant non-specular scattering from in-plane variations of the SLD, neutron specular reflectivity is accurately described by a one-dimensional Schrödinger wave equation[1],

$$-\frac{\partial^2\psi(k_{0z},z)}{\partial z^2} + 4\pi\,\rho(z)\psi(k_{0z},z) = k_{0z}^2\,\psi(k_{0z},z), \tag{1}$$

where ψ is the wave function, ρ is the SLD profile of a given sample, k_{0z} is the wave vector's z component and z is the depth inside the sample, perpendicular to the sample's interfaces (for more details see e.g. [4] and references therein).

[1] Throughout this work, derivations and discussions mainly focus on neutron reflectometry; however, the same derivations and approach apply to the case of x-rays with little or no change.

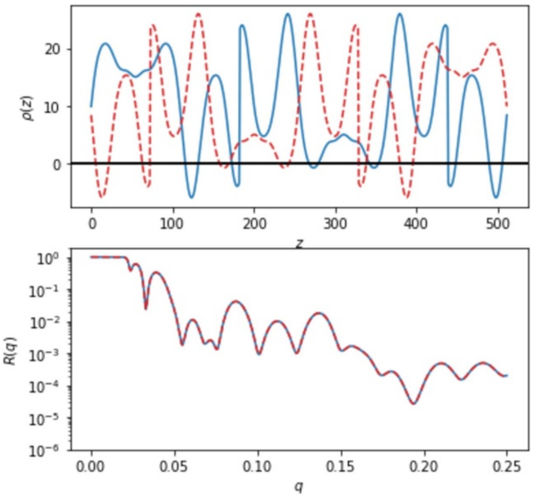


Figure 1. Two different anti-periodic SLD profiles, one the midpoint reflection of the other, produce the exact same reflectivity curve.

In terms of the solution to equation (1), the amplitude of the reflected wave can be represented by the integral

$$r(Q) = \frac{4\pi}{iQ}\int_0^L \psi(k_{0z}, z)\,\rho(z)\,\mathrm{e}^{ik_{0z}z}\mathrm{d}z, \tag{2}$$

where $r(Q)$ is the complex-valued reflection amplitude as a function of the wave vector transfer perpendicular to the surface, $Q = 2\,k_{0z} = 4\pi\sin(\theta)/\lambda$, and L is the thickness of the SLD profile. However, the only quantity accessible to measurements is the reflectivity, which can be expressed in terms of the amplitude as $R(Q)\equiv r^{*}r$.

1.1. The phase problem and fitting

The measured reflectivity, $R(Q)$, does not carry any information regarding the phase, making the inference of an SLD profile from a reflectivity curve a non-invertible inverse problem: at a theoretical level, there are families of SLD profiles which produce exactly the same reflectivity curve. In particular, this applies to any anti-periodic SLD profile that is reflected at the midpoint (see figure 1). For an experimenter measuring reflectivities, $R(Q) = r^{*}r$, both SLD profiles are indistinguishable.

The degeneracy that causes different SLD profiles to produce identical reflectivity curves is known as the phase problem and is accentuated by the truncation of the reflectivity data at a maximum value of Q and the statistical uncertainty associated with the noisy data points (e.g. see [5]).

Typically, the data obtained from reflectometry experiments are analyzed in terms of physical models, trusting their ability to reproduce the measured experimental reflectivity curves. Using specialized software (e.g. [6, 7]), an iterative process of parameter optimization is thus established in which, at each iteration, (a) certain parameters of the physical model are set, (b) theoretical reflectivity curves are calculated and (c) a comparison is made between the theoretical and experimental reflectivity curves. The latter comparison is quantified by a figure of merit (FOM) and the goal of the iterative process is reached when the FOM reaches its minimum value, namely, when the experimental and the theoretical reflectivity curves are as close as allowed by the physical model and the experimental resolution. Such an iterative *fitting* process is far from immediate, requiring experimenters to test various FOMs, several minimization algorithms, multiple sets of model parameters and even different physical models.

In recent years, BornAgain [6], well-established software for simulating and fitting neutron and x-ray grazing-incidence small-angle scattering (GISAS) data has started to offer fitting and simulation capabilities for reflectometry data as well. The present work aims to explore the possibilities that machine learning has to offer the development of reflectivity data-driven software.

2. The expressive power of artificial neural networks

Deep neural networks can be thought of as compositions of multiple simple functions (called layers) that can approximate rather complicated functions. In fact, the celebrated universal approximation theorem states that depth-2 networks with suitable activation functions can approximate any continuous function in a compact domain to any desired accuracy [8–11]. However, the size of such a neural network could be exponential in the input dimension, which means that a depth-2 network may have a huge width [12–14]. Part of the recent renaissance in artificial neural networks (ANNs) is based, not on enabling wider networks to be trained, but on the empirical observation that *deep neural networks* tend to achieve greater expressive power per parameter than their shallow counterparts.

It has been shown that any Lebesgue-integrable function from $\mathbb{R}^N \rightarrow \mathbb{R}$ can be approximated by a fully connected rectified linear unit (`ReLU`) deep neural network of width $N+4$ to an arbitrary accuracy with respect to the L_1 distance and, except for a negligible set, all functions from $\mathbb{R}^N$ to $\mathbb{R}$ cannot be approximated by any `ReLU` network whose width is no more than N [15]. Hanin [16] showed that any continuous function $f\colon [0,1]^{d_{\text{in}}}\mathbb{R} \rightarrow \mathbb{R}^{d_{\text{out}}}$ can be approximated by a net of width $d_{\text{in}} + d_{\text{out}}$, also obtaining quantitative depth estimates for such an approximation in terms of the modulus of continuity of *f*. At the same time, they claimed that there were no conclusive results regarding the depth that such a network should have, and, even in the case that a precise ANN architecture can be defined to achieve a given precision, nothing can be said regarding the success of the training process.

In contrast to these apparent theoretical drawbacks, one of the first architectures that is taught when studying ANNs—a single hidden layer of width 128, can classify 28×28 pixel images (i.e. points in $\mathbb{R}^{784}$) into ten discrete categories through a mapping into the real unit interval (e.g. `TensorFlow` tutorials [17]). These achievements, in apparent contradiction to the theoretical results described above, are possible because the images under classification do not sample the whole $\mathbb{R}^{784}$ but are only drawn from a very limited subspace of it. In fact, when the trained network is presented with images that do not belong to that subspace, the ANN fails—it may even classify apparently random noise as some of the ten digits with almost 100% certainty.

In this work, our focus was to train simple and small ANNs targeted at specialized kinds of SLD profiles, as opposed to larger and more general-purpose neural networks, since any attempt to teach a single ANN a general pseudo-inverse function at the moment is almost certainly doomed to fail.

3. Related work

Non-invertible inverse problems are not unique to x-ray and neutron reflectometry, and several other scientific communities have already started to investigate the usefulness of ANNs for tackling them, showing astonishing performance for applications such as low-dose computed tomography and various sparse data problems. While there are few theoretical results, some well-posedness results and quantitative error estimates have been found for some problems [18]. For instance, in electrical impedance tomography (EIT), which represents a typical non-linear ill-posed problem, the electrical properties of tissues are determined by injecting a small amount of current and measuring the resulting electric potential, which must be transformed into a tomographic image by some reconstructive algorithm. Many artificial intelligence approaches for tackling EIT have been used in the past few years (e.g. [19] and references therein) with outstanding results.

In the realm of x-ray reflectivity, recent work has shown that properly trained ANNs with simple fully connected architectures can be used to characterize thin-film properties (thickness, roughness and density) from XRR data within milliseconds, using minimal *a priori* knowledge. Their results differ from traditional least-mean-squares fitting by less than 20% [20]. Such an approach could benefit the study of the growth behavior of thin films.

Current software packages that are available to infer physical models from reflectivity curves allow users to define SLD multi-layer models and, after some fitting process, choose the parameter combination that best reproduces the experimental data; all of this happens with varying degrees of interactivity, either through an application programming interface (API) or a graphical user interface (GUI). Examples of parameters that can be fitted are layer thicknesses, the roughnesses between layers and layer SLD values. In contrast, the physical models used to represent SLD profiles in the present work do not make use of the multi-layer abstraction. The models used in the present work deal with quasi-continuous SLD profiles that have an SLD value defined at each point and that vary continuously throughout the sample without the need to introduce extra abstractions such as interfacial roughness.

4. Application to reflectivity profile inversion

4.1. Scaling of the problem

The only physical quantities involved in the calculation of a reflectivity curve, assuming a perfect instrument able to measure up to $Q \rightarrow \infty$ and an SLD profile extending up to $z \rightarrow \infty$, are the wave transfer vector Q and the SLD profile $\rho(z)$. These quantities can be further reduced using dimensional analysis. In fact, the number of dimensionless groups that define the problem, which equals the total number of physical quantities (Q and ρ) minus the fundamental dimensions (length), is only one ($= 2 - 1$). By choosing an arbitrary SLD scale ρ_0 and defining the dimensionless parameter $p = \rho/\rho_0$, equation (2) can be re-cast in the following form:

$$r_0(Q) = \sqrt{\rho_0} \times \frac{4\pi}{iQ} \int_0^\infty \psi(\eta, \xi)\, p(\xi)\, \mathrm{e}^{i\eta\xi} \mathrm{d}\xi, \tag{3}$$

where $\xi = z\sqrt{\rho_0}$, $\eta = k_{0z}/\sqrt{\rho_0}$, and $p(\xi) = \rho(z)/\rho_0$ are the dimensionless depth, wave vector and SLD profile, respectively. Thus, to solve for a different SLD scale, ρ^*, it is enough to solve for $r_0(Q)$ and rescale afterwards by $\sqrt{\rho_*/\rho_0}$, i.e.

$$r_*(Q) = \sqrt{\frac{\rho_*}{\rho_0}} \times r_0(Q). \tag{4}$$

In the following, the SLD scale of the problem is chosen to be that of the substrate, $\rho_0 = \rho_{\text{subs}}$, i.e. $p_{\text{subs}} \equiv 1$.

4.2. Data simulation

The phase problem implies that a single input (e.g. a reflectivity curve) may be consistent with two or more different outputs (e.g. SLD profiles). If one were to train an ANN to find a pseudo-inverse transformation, using data containing different output targets corresponding to the same input (different branches), the training process would not be successful, as different branches would cause the weights of the ANN to drift in inconsistent directions. To avoid such a situation, it is necessary to ensure that either (*1) the solution space has no branches or (*2) the training targets all lie in the same branch of the solution space. For the latter scenario, it must also be kept in mind that ANNs trained in such a way will only be useful as long as the expected solutions are consistent with the branch to which the training targets belong. Looking to fulfill *1, a set of artificial SLD profiles is generated which can offer a 1-1 correspondence with their associated reflectivity curves, e.g. the SLD profiles are odd with respect to the middle of the depth[2]. To try to fulfill *2 to some extent, two sets of SLD profiles are generated, each set corresponding to a physically relevant type of sample, namely, single films and lamellar structures.

All the simulated SLD profiles used as training targets in this work have an overall thickness $L = 512$ Å, and are sampled from 512 equally spaced points within the semi-closed interval $z = (0, L]$, between two semi-infinite fronting and backing mediums of constant SLD, $\rho_{-\infty} = 0$ and $\rho_{+\infty} = 10^{-6}$ Å^{-2}, respectively. Each SLD profile was thus modeled as a `multilayer` composed of 512 slices, each 1 Å thick, with no interfacial roughness. In this way, smooth SLD profiles were mimicked by quasi-continuous small variations of the SLD between contiguous layers throughout the whole interfacial structure. Such SLD profiles are then used to simulate the corresponding reflectivity curves for which the wave vector transfer is limited to an interval $0 \leqslant Q \leqslant Q_{\max} = 0.25$, sampled by 129 equally spaced points. Imperfections in the reflectivity curves are only characterized by a background of 10^{-6} and a Q-resolution of 5%.

To bring the so-called *features* (in the case at hand, the reflectivity curves) into a small dynamic range, the average reflectivity curve needs to be subtracted from all reflectivity curves in the set. Thus, the new set of curves is a zero-mean set of curves. Each of the resulting zero-mean curves is then rescaled by dividing it by the standard-deviation curve. Finally, the obtained zero-mean, unit-standard-deviation set of curves is ready to be used for training. In the following, this set of curves is referred to as the *training features*. In contrast to the reflectivity curves, the SLD profiles, i.e. the *training targets*, are left unchanged.

Figure 2 gives an overview of the three different data sets used and their preprocessing; a more detailed discussion of each is provided in section 5.

4.3. Network architecture

The network architecture is dictated, to some extent, by the problem constraints: The number of input neurons must be the same as the input reflectivity curve lengths, which is chosen to be $N_{\text{in}} = 129$. The number of output neurons must be the same as the expected SLD profile resolution points, which are $N_{\text{out}} = 512$.

Hidden layers are chosen to be fully connected layers, a choice also dictated by the problem constraints. In fact, the choice of this kind of layer, as opposed to the so-called *convolutional* layers powering state-of-the-art image recognition and segmentation algorithms, is well grounded in a simple observation: while convolutional layers are powerful at detecting shapes and measuring their relative positions with

[2] Note that it has not been rigorously shown that a 1:1 correspondence exists. Several profiles from the family could still produce the same reflectivity profile.

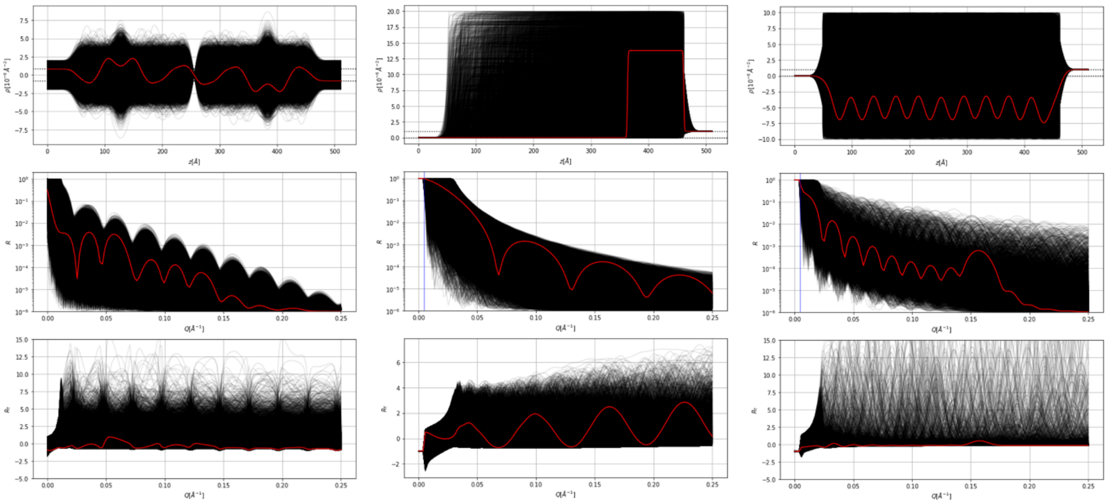


Figure 2. In the top row, three different families of SLD profiles are shown: odd functions, single films and lamellar structures. The middle row shows the simulated reflectivity curves obtained from the SLDs in the top panels. The bottom row show the rescaling of the reflectivity curves which have zero mean and unit standard deviation. A single random curve is highlighted in red to give a qualitative impression of the features defining each family of curves.

respect to one another, independent of their absolute positions within the scene or image, in the case at hand, the opposite is actually needed: the absolute position of all intensity data points along the q line is all that matters to define a reflectivity curve.

To reproduce the non-linearities, `ReLU` functions are used after each hidden layer, except for the one before the output layer and, in order to prevent *overfitting*, a single dropout layer with a rate of 0.5 is added as a regularizer before the output layer.

The architecture so far has been dictated to a great extent by the problem. However, there are no definite rules for selecting either a concrete number of hidden layers or the number of neurons present in each hidden layer. In order to make these concrete choices, preliminary experiments are carried out for different network depths and widths (i.e. the number of hidden layers, N_L, and the number of neurons per hidden layer, N_N, respectively). The architecture is arbitrarily restricted to hidden layers that all possess the same number of neurons, with $N_\mathrm{L} = 0, 1, 2, 4, 8, 10$; $N_\mathrm{N} = \{0.25, 0.5, 1, 2, 4\} \times N_\mathrm{out}$[3]. Each one of these prototype networks is then trained and tested using 100 K artificial and non-physical SLD profiles of odd-function shapes (left panel of figure 2). The performance of each ANN is tested by feeding them 5 K previously unseen reflectivity curves and calculating the mean absolute error (MAE),

$$\mathrm{MAE}_j = \mathrm{MAE}(\rho_j, \tilde{\rho}_j) = \frac{1}{\mathrm{N}_{\mathrm{RES}}} \sum_k \rho_{jk} - \tilde{\rho}_{jk}, \tag{5}$$

between the ANN-predicted SLD profile and the known SLD target profile.

The time to train, the memory space required, and the MAE incurred by each of the architectures tested are shown in figure 3. From this analysis, the best architecture was selected, $N_\mathrm{L} = 2$ and$N_\mathrm{N} = 2$, and used for the numerical experiments described in the remaining part of the present work.

4.4. Training

At training time, the optimizer of choice is the adaptive moment estimation algorithm (ADAM [21]), together with a mean-squared-error (MSE) loss function. All models are set to be trained for 500 epochs and an early stopping callback with a `patience` parameter value of ten epochs is also provided. This callback prevents overfitting by stopping the ANNs from training whenever the error in the validation set does not decrease any more through the epochs. It is also important to define the number of samples needed to train: too few samples do not allow the network to learn; too many samples waste time and resources. To this end, several preliminary experiments were carried out as well, in which the network training time, the MAE and the MSE were evaluated with respect to the number of training samples. Figure 4 shows that the time it takes

[3] Throughout the rest of the manuscript, N_N refers to the multiplicative factor of N_out, and not to the actual number of neurons per hidden layer itself.

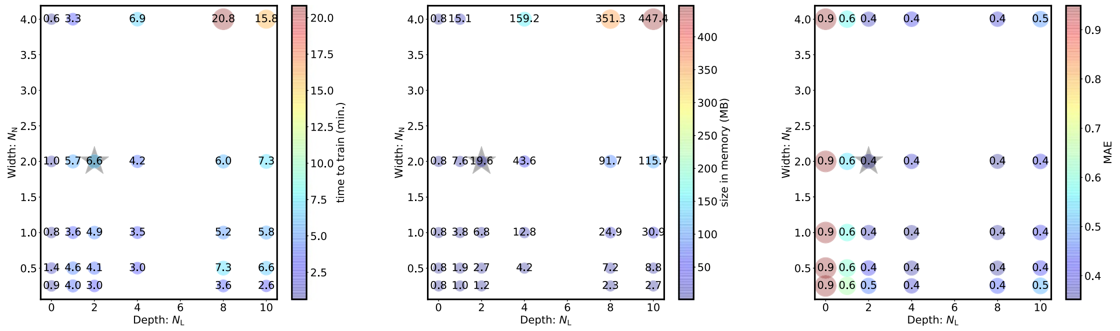


Figure 3. Several network architectures were built and tested using 100 K artificially generated SLD profiles and reflectivity curves. The left panel shows the times each architecture took to complete training; the middle panel shows their disk space requirements; and the right panel shows the MAE incurred between the predicted SLD profiles and the true targets present in the test set. Ideally, a good ANN is accurate, trains fast and requires small amounts of disk space. Each architecture tested is described by its number of hidden layers, N_L and the number of neurons in each hidden layer N_N.

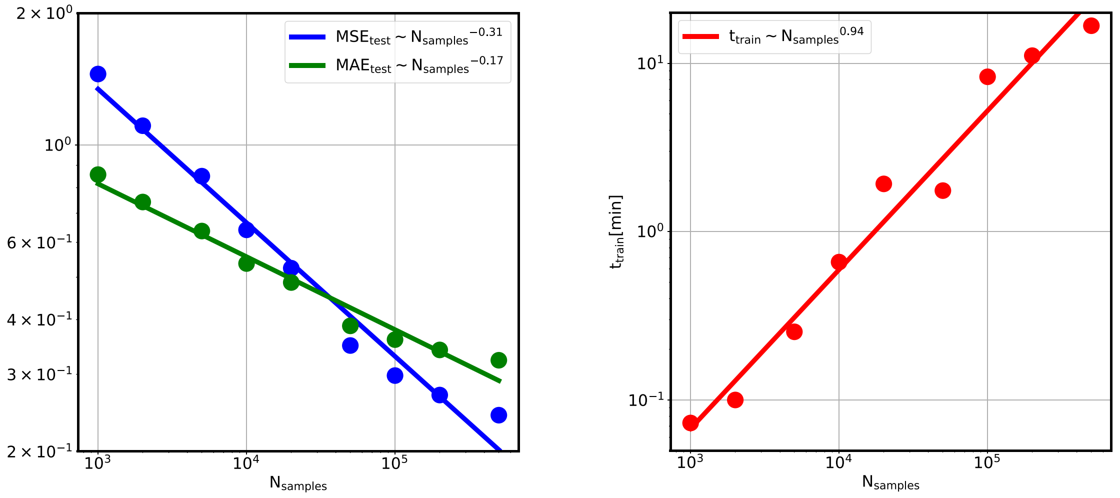


Figure 4. The MAE and the MSE are plotted against the number of training samples in the left panel. In the right panel, the plot shows the time it takes for a neural network to be trained as a function of the number of training samples. While the time it takes to train the ANN is directly proportional to the number of samples, the MAE and the MSE only decrease as weak power laws with exponents of $\simeq -1/6$ and $-1/3$, respectively.

to train the ANN is directly proportional to the number of samples, whereas the MAE and the MSE only decrease as weak power laws with exponents of $-1/6$ and $-1/3$ respectively. This means that to reduce the MSE and the MAE by some 10%–20% at most, it would be necessary to double the sample size, which in turn would double the time to train, and some juggling with the machine's memory would be required. All of this assumes that the power law does not get weaker for a larger number of samples. Weighing all of these factors, a set of $N_s = 100$ K samples was used for training with each of the three artificial datasets used.

5. Results

Three different datasets have been used to train three different neural networks: (a) SLD profiles possessing odd symmetry, (b) single films and (c) lamellar structures. Thus, while all the networks are architecturally equal, their learned weights are different.

5.1. Odd SLD profiles

If an SLD profile possesses the symmetry $\rho(z + L/2) = -\rho(z) + \mathrm{const}$, $z \in [0, L/2]$, the same reflectivity curve it produces is recovered by reflecting it at its mid-point (c.f. section 1.1). Thus, by defining a dataset composed only of odd SLD profiles, it is ensured that there is a one-to-one correspondence between an SLD profile and its associated reflectivity curve. Figure 5 shows five test SLD profiles recovered after a neural network is trained using such a dataset. The overall MAE for the test set, composed of 5 K samples, lies at around 0.35.

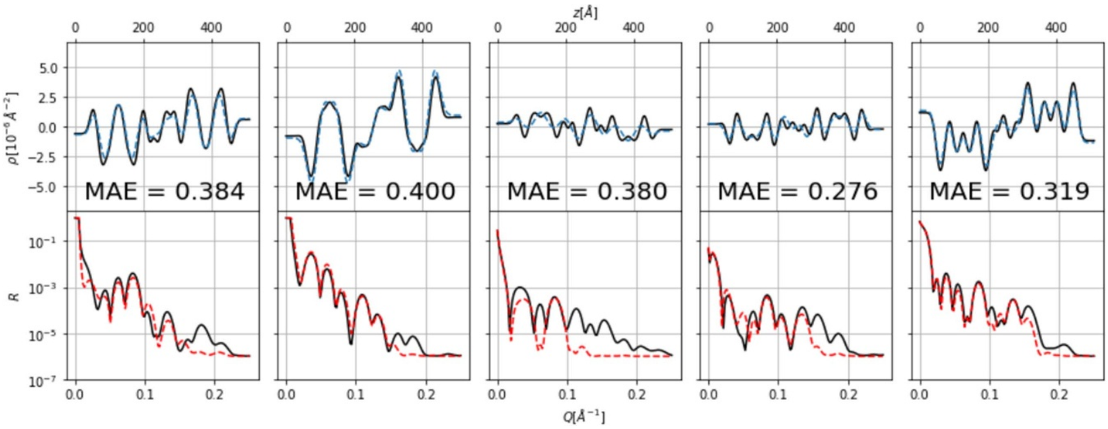


Figure 5. By training an ANN using a set of 100 K random odd SLD profiles, the phase problem can be circumvented and there is a one-to-one correspondence between an SLD profile and its associated reflectivity curve. Five test reflectivity curves (black lines; bottom panels) were shown to the trained network for the first time; from those curves, the network guessed the SLD profiles that produced them (blue dashed lines; top panels). In the figure, the mean absolute errors (MAEs) with respect to the true SLD profiles (black lines; top panels) are also shown. Additional reflectivity curves are shown (red dashed lines, bottom panels), generated from the SLD profiles predicted by the ANN.

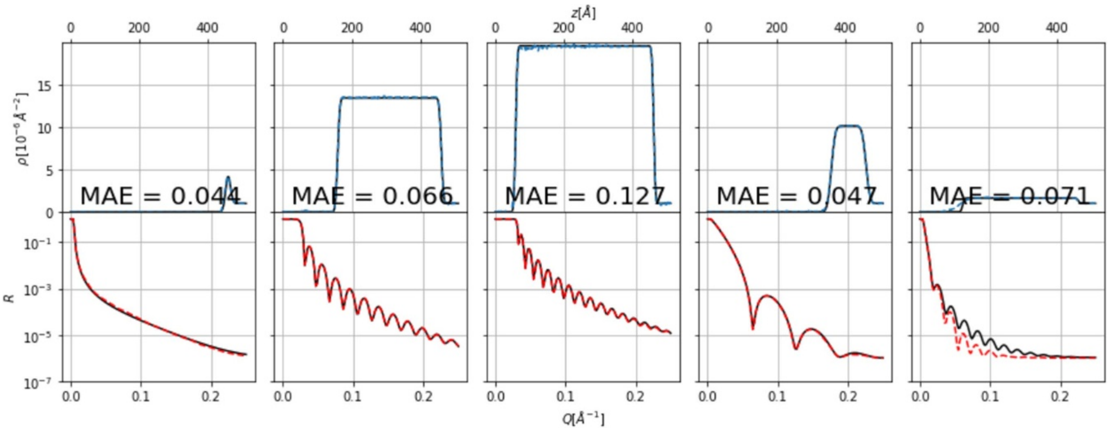


Figure 6. Single films have simple reflectivity curves for which the trained network is able to correctly predict the associated SLD profiles. Five test reflectivity curves (black lines; bottom panels) were shown to the trained network for the first time; from those curves, the network guessed the SLD profiles that producesd them (blue dashed lines; top panels). In the figure, the MAEs with respect to the true SLD profiles (black lines; top panels) are also shown. Additional reflectivity curves are shown (red dashed lines, bottom panels), calculated from the SLD profiles predicted by the ANN.

5.2. Training on films with positive SLDs

The simplest family of SLD profiles is that of single layers on top of a substrate. In this family, each SLD profile is made up of only three regions: the superstrate, $\rho_{\text{sup}} = 0$ extending from $z = -\infty$ to $z = \Delta_-$; the substrate, $\rho_{\text{sub}} = 1$ extending from $z = 512 - \Delta_+$ to $z = \infty$; and a film of constant SLD ρ_f extending from $z = \Delta_-$ to $z = 512 - \Delta_+$. Here, $\Delta_{+/-}$ are buffer regions defined to smoothen the transition between the SLD of the film and that of the super- and substrates. We set $\Delta_+ = 50$ for all SLD profiles of this data set while $\Delta_- \in [50, \Delta_+)$ is randomly chosen for each generated profile and effectively defines the thickness of the film. The SLD of each film is randomly chosen such that $\rho_f \in [0, 20]$.

Figure 6 shows five test SLD profiles recovered after a neural network was trained using such a dataset. The overall MAE for the test set, composed of 5 K samples, lies at around 0.08.

5.3. Training on lamellar structures

The family of SLD profiles for this data set was defined by lamellar structures, each one with n_r equally spaced regions of alternating SLDs between ρ_1 and ρ_2. Both SLD values were chosen randomly from values between -10 and 10, and the number of regions was randomly chosen to be between 1 and 64, extending from $z = \Delta$ to $z = 512 - \Delta$, where $\Delta = 50$ defined a buffer that smoothed the transition between the lamellar sample and its surroundings. The profiles obtainedwere smoothed using a Gaussian filter of a randomly chosen width between 0 and 10.

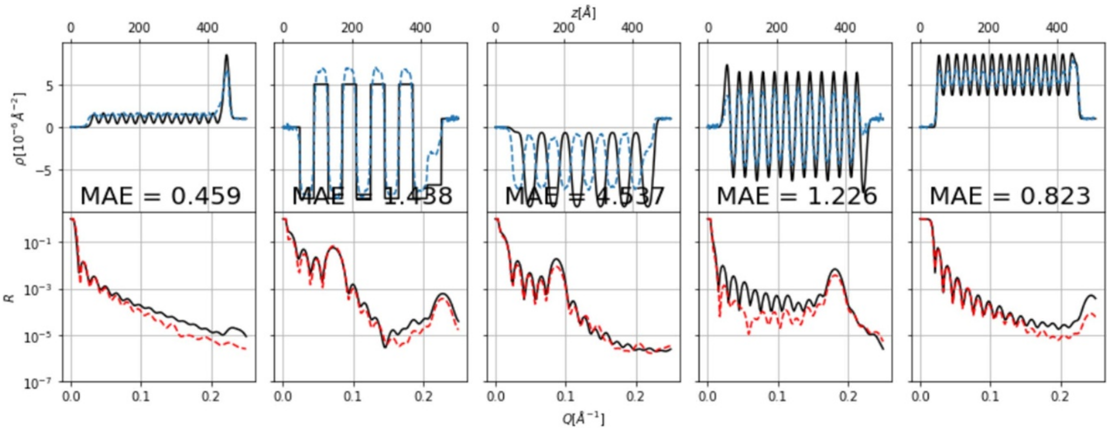


Figure 7. In the case of periodic lamellar SLD profiles, the phase problem is very apparent; even if most of the predictions recover a close approximation to the original reflectivity curves, the predicted SLD profiles differ from the actual targets. Five test reflectivity curves (black lines; bottom panels) were shown to the trained network for the first time; from those curves, the network guessed the SLD profiles that produced them (blue dashed lines; top panels). In the figure, the MAEs with respect to the true SLD profiles (black lines; top panels) are also shown. Additional reflectivity curves are shown (red dashed lines, bottom panels), calculated from the SLD profiles predicted by the ANN.

Figure 7 shows five test SLD profiles recovered after a neural network was trained using such a dataset. The overall MAE for the test set, composed of 5 K samples, lies at around 0.98.

The SLD profiles predicted by the ANN are not always consistent with the target SLD profiles. However, when calculating the reflectivity curves produced by such predicted profiles, the curve obtained is quite similar to the original curve presented to the ANN. For this family of SLD profiles, the trained network is a good example of a pseudo-inverse that, due to the degeneracy of the problem, recovers a plausible solution but not necessarily the correct solution.

5.4. Robustness against noise

In order to train models that are robust against noise, the training set was *augmented* by adding different levels of constant background, δR, modeled as Gaussian random white noise, to each of the ideally simulated reflectivity curves, R_{ideal}. To avoid negative intensities, the noise was added to the square root of the reflectivity signal and the result was squared to recover a noisy version of the original signal:

$$R_{\text{noisy}} = \left(\sqrt{R_{\text{ideal}}} + X_{\mu,\sigma}\right)^2, \tag{6}$$

where $X_{\mu,\sigma}$ is a random variable extracted from a Gaussian distribution with $\mu = 0$ and $\sigma = \sqrt{\delta R}$.

The training was carried out over the augmented data set of artificial SLD profiles of section 5.1. The dataset consisted of 10 K clean reflectivity curves, replicated nine times for each level of background $\delta R \in \{1, 2, 3, 4, 5, 10, 20, 50, 100\} \times 10^{-6}$, for a total of 100 K reflectivity curves. After training, the model was tested over previously unseen reflectivity curves with arbitrary levels of noise, $10^{-6} < \delta R < 10^{-3}$, delivering satisfactory results for background noise levels of up to $\delta R \simeq 10^{-5}$ and performing poorly for reflectivity signals with stronger background components, as can be observed in figures 8 and 9.

5.5. A parameter estimation for a two-layer system

In a real parameter estimation workflow, it is necessary to provide a quantitative analysis including the accuracy of the fit for several parameters (e.g. the thickness of each layer, the SLD value of each layer and the interfacial roughness between each pair of layers). The available software that provides such functionality requires that, for each sample, a model is built by specifying all the parameters to fit. In contrast, the proposed approach is, in this sense, parameter free. A drawback of the proposed approach is its lack of a quantitative parameter estimation mechanism. However, it offers the advantage of immediate response: thousands of reflectivity curves can be inverted into SLD profiles in a couple of seconds. As such, the proposed approach should not be taken as a replacement for available fitting methods, but rather as a preliminary tool to accelerate the process, providing preliminary glimpses of what the ultimate parameters of a traditional fitting process may look like. Figure 10 shows a simulated reflectivity curve (a fake experimental curve) which, after being fed to the neural network, produces an SLD profile from which the experimenter can infer the following parameters: two layers, with thicknesses of around 200 Å and 150 Å each, with

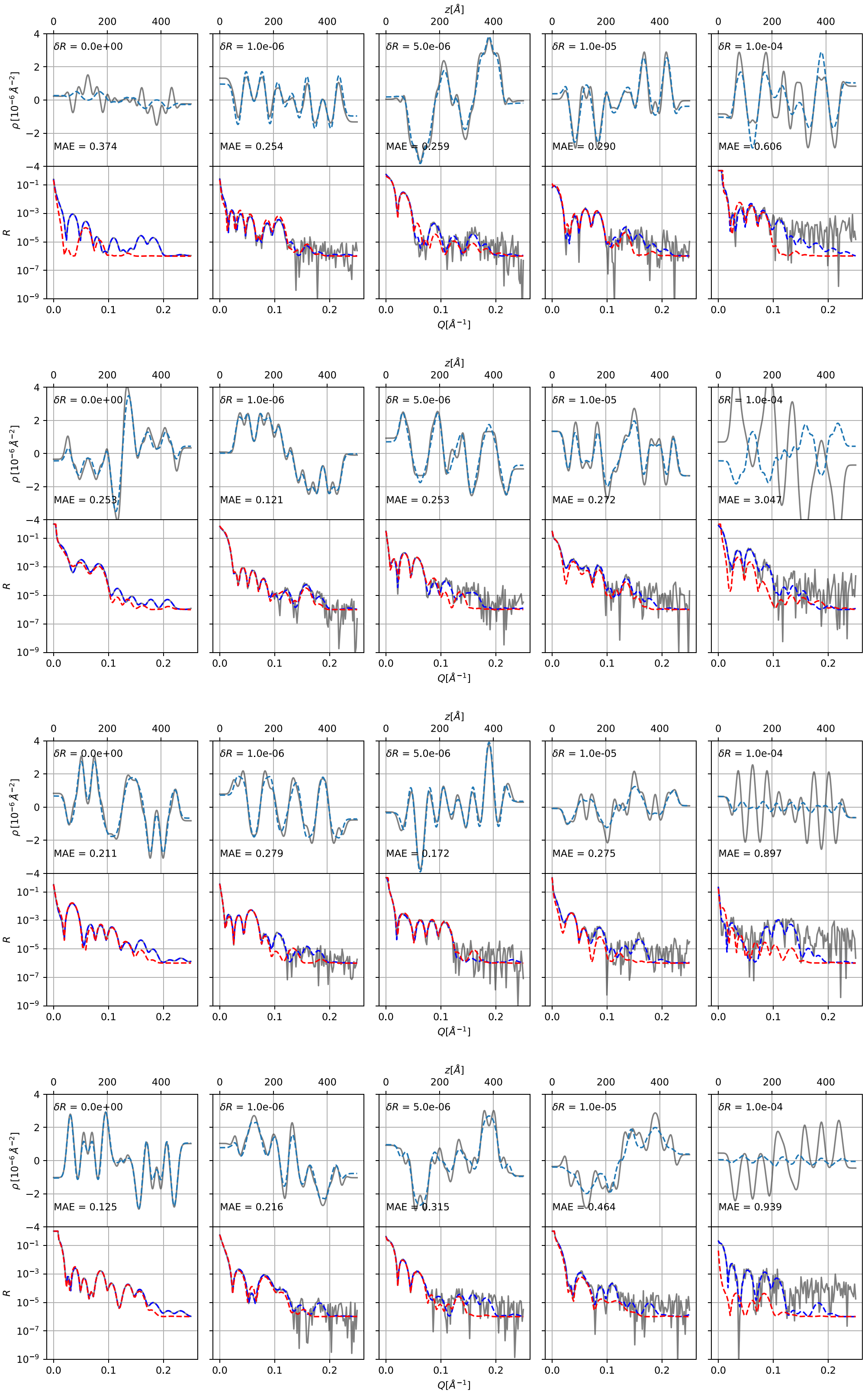


Figure 8. From top to bottom, four rows of panels are shown. In each panel, the top plot shows two SLD profiles and the bottom plot shows three reflectivity curves. The black reflectivity curves in the bottom plots mimic experimental data fed to the neural network, which predict the blue SLD profiles in the top plots. The blue SLD profiles are compared to the target SLD profiles in black and the MAE is reported in each SLD plot. By simulating ideal and noiseless reflectivity curves, the target SLD profiles in black produce the blue reflectivity curves in the bottom plots, while the predicted SLD profiles in blue produce the red reflectivity curves. From left to right, the background noise intensities are $\delta R = \{0, 1, 5, 10, 100\} \times 10^{-6}$ respectively.

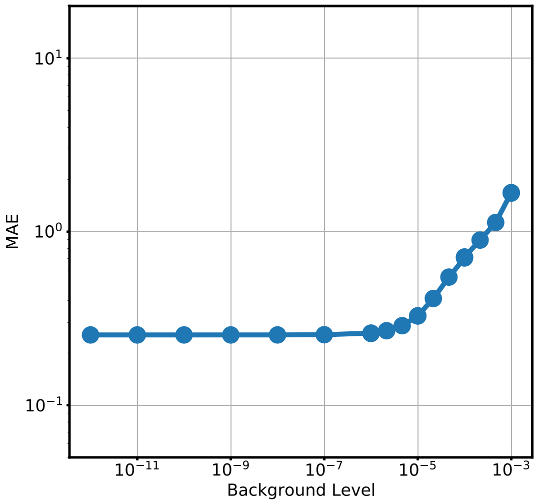


Figure 9. After being trained using a noise-augmented dataset, the neural network was able to recover SLD profiles from noisy reflectivity curves with an MAE of around 0.3, provided that the noise present in the reflectivity curves fed to the network satisfied $\delta R \lesssim 10^{-5}$. For increasing noise levels, the error incurred by the network rose steeply, so that its predictions were no longer reliable.

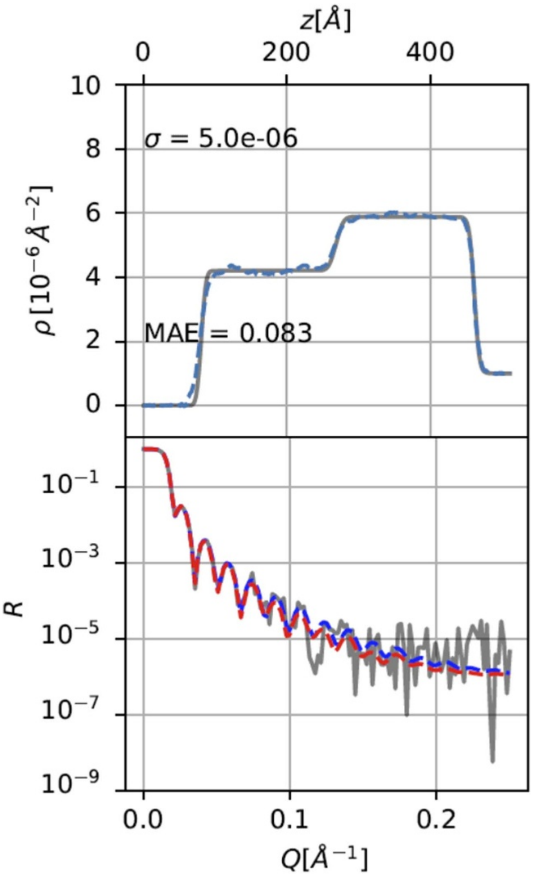
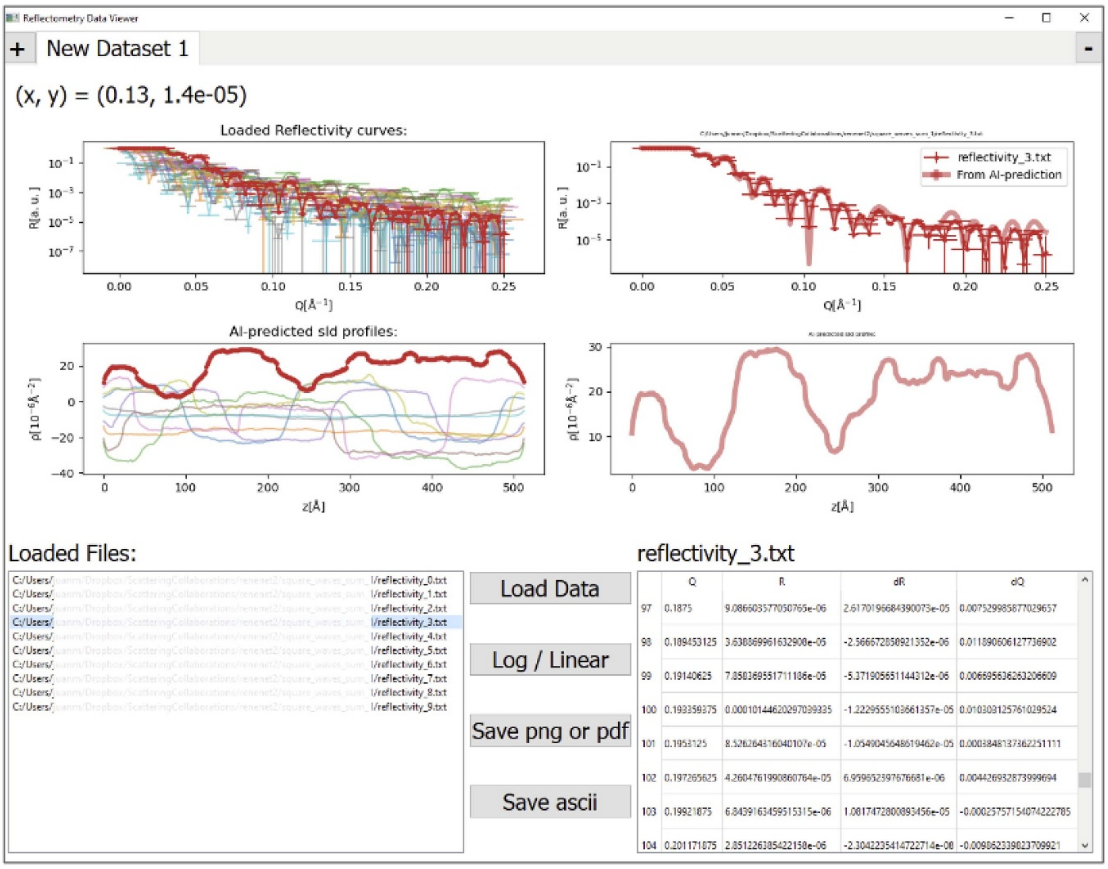


Figure 10. In a real experiment, the neural network would be fed with the experimental curve (in black, bottom plot) and produce a guessed SLD profile (in blue, top plot). Using a simulation package, a reflectivity curve simulated from the guess SLD profile (in red, bottom plot) could then be compared to the experimental data. The ideal SLD profile (in black, top plot) and the noiseless reflectivity curve it produces (in blue, bottom plot) are also shown for completeness, together with the noise level of the fake reflectivity curve and the MAE the neural network produced while guessing the correct SLD profile.

corresponding SLD values of around $\rho = \{4, 6\} \times 10^{-6}$ Å^{-2}; a substrate SLD of $\rho_{\text{sub}} = 1 \times 10^{-6}$ Å^{-2} and a superstrate of SLD $\rho_{\text{sup}} = 0$. These rough estimates can then be fed into a traditional model minimizer as first guesses for the fitting process.



Loaded Files:

C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_0.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_1.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_2.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_3.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_4.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_5.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_6.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_7.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_8.txt
C:/Users/juanm/Dropbox/ScatteringCollaborations/renenet2/square_waves_sum_l/reflectivity_9.txt

reflectivity_3.txt

	Q	R	dR	dQ
97	0.1875	9.086603577050765e-06	2.6170196684390073e-05	0.007529985877029657
98	0.189453125	3.638869961632908e-05	-2.566672858921352e-06	0.011890606127736902
99	0.19140625	7.858369551711186e-05	-5.371905651144312e-06	0.006695636263206609
100	0.193359375	0.00010144620297039335	-1.2229555103661357e-05	0.010303125761029524
101	0.1953125	8.526264316040107e-05	-1.0549045648619462e-05	0.0003848137362251111
102	0.197265625	4.2604761990860764e-05	6.959652397676681e-06	0.004426932873999694
103	0.19921875	6.8439163459515315e-06	1.0817472800893456e-05	-0.00025757154074222785
104	0.201171875	2.851226385422158e-06	-2.3042235414722714e-08	-0.009862339023709921

Figure 11. In a real data-analysis workflow, several reflectivity curves (top-left plot) are loaded from several files (bottom-left list), and the neural network is automatically engaged to produce possible SLD profiles (bottom-left plot), from which the experimenter can *eyeball-estimate* the parameters required to feed traditional minimizers. The contents of the selected file are also shown (bottom-right table), together with the isolated inferred SLD profile (bottom-right plot) and two isolated reflectivity curves (top-right plot) corresponding to the data and the simulated ANN-inferred SLD profile. See also https://youtu.be/NKismnONlmA.

6. Discussion

The time required to generate each of the training data sets and to train each of the ANNs is rather short: around a couple of hours for data generation and a similar time for ANN training (provided a GPU is available[4]). For unfortunate cases, in which the data set is not suited for ANN training (because of the phase problem, for example), the overfitting regime may be reached rather soon and the training stops in even less than 10 min.

Typical trained ANNs only require only around 100 MB of disk space, whereas the training data are around at least 500 MB in size (just for reflectivity curves). Once trained, the ANNs can recover plausible SLD profiles from 5 K reflectivity curves in around 0.5 s. What these metrics imply is that the ANNs efficiently abstract the information contained in the training data: as if they were compressing the whole dataset (and more) with a compression ratio of at least 5:1.

A good question to ask now is whether the ANNs actually learn the transformations or are only memorizing the associations between particular SLD profiles and reflectivity curves. While the evolution of the metrics during training and the performance on the test set suggest that actual learning is taking place (see appendix A.2), it must be stressed that the ANNs are certainly not learning a fully general transformation but are rather optimized to interpolate within the family of SLD profiles for which they are trained. Thus, to take advantage of the proposed approach in real experiments, the simulated training data should incorporate the instruments' specifications with as much detail as possible. Additionally, to train an ANN to be robust against noise, for each target SLD profile, several noisy reflectivity curves with varying levels of noise were simulated, effectively allowing the network to give higher priority to data points lying at

[4] In the present work, an Nvidia GeForce GTX 1080 was used for the production runs, while several prototypes were developed on Google Cloud infrastructure.

lower Q values. In section 5.4, it was shown that such an approach to neural network training succeeds in allowing the network to recover SLD profiles provided that the background noise satisfies $\delta R \lesssim 10^{-5}$, a constraint that is usually fulfilled. The trends reported, in fact, demonstrate the possibility of training networks that are robust to noise.

Due to the quick response of the trained ANNs, they could be incorporated into a data pipeline able to potentially provide preliminary analysis of real-time phenomena such as the swelling or drying of thin films (see also [20]), or used in large batch analysis of huge data sets from large facilities. In fact, the time required for an ANN to invert 10 K reflectivity curves is only around 1 s. However, as seen in section 5.3, interpreting reflectivity curves in such a way may offer plausible SLD profiles that nevertheless do not correspond to the actual sample under the beam; thus, a careful examination at a later stage will always be required and, more likely than not, the ANN-inferred SLD profiles should be used in the meantime as starting models to feed traditional fitting methods. In the future, with a higher degree of customization, ANNs could be incorporated into an automated analysis pipeline.

The proposed algorithm has the advantage of being quick for batch analysis at the cost of losing a high degree of per-profile customization. The quick response of neural networks makes it possible for the inferred SLD profiles to be readily shown to experimenters at data-loading time, together with the loaded reflectivity curves. When offered plausible SLD profiles, experimenters can easily use them to infer the relevant parameters with which to feed traditional fitting models. Figure 11 offers a sketch of what such a data loader could look like: a small piece of software that could be included as an optional plug-in for some of the available software packages for reflectometry data analysis (e.g. Bornagain and Refnx [6, 7], among others).

7. Conclusions

In this work, sample physical models were described using a new paradigm: detailed layer-by-layer quantitative descriptions (SLDs, thicknesses, roughnesses) were replaced by parameter-free curves $\rho(z)$, allowing *a priori* assumptions to be used in terms of the sample family to which a given sample belongs (e.g. ‘thin film’, ‘lamellar structure,’ etc). Such an approach is not comparable to traditional fitting methods and acts as a preliminary analysis tool for experimenters to make quick first estimates of the relevant parameters to feed into traditional fitting methods.

Though the ANNs described in this work are not yet capable of interpreting real reflectivity curves in terms of SLD profiles, the numerical experiments carried out show that the short training times, the small space required and the ANN robustness against noise could make it practical to train several ANNs with richer architectures, specialized in different instruments and selected families of samples. To bring the proposed approach to the arena of real experiments, additional information about the instrument, such as its resolution and Q range would also be needed.

Data availability statement

Data sharing is not applicable to this article as no new data were created or analysed in this study.

Acknowledgments

JMCL would like to thank the BornAgain team at JCNS-MLZ, especially Joachim Wuttke for his healthy skepticism and criticism; and Alexander Schober for his optimistic encouragement. I would also like to thank the following for the many fruitful discussions: Wojciech Potrzebowski from the ESS; Miguel González from the ILL; Alexandros Koutsioumpas, Jean F Moulin, Gaetano Mangiapia, and Martin Haese from the MLZ. Their feedback and comments have certainly had a positive impact on the development and presentation of this work.

Appendix

A.1. Training on random SLD profiles

Throughout this work, we have argued that the ANN training process is sure to fail in the majority of cases, as the solution space usually has branches and many of the training targets may lie in inconsistent branches of the solution space, causing the weights of the ANN to drift in inconsistent directions during the training. To prove this point, an additional ANN was trained using a set of SLD profiles composed of a random number of layers of random thicknesses and random heights. Figure A1 shows five test SLD profiles recovered after a neural network was trained using such a dataset. None of the predicted SLD profiles were

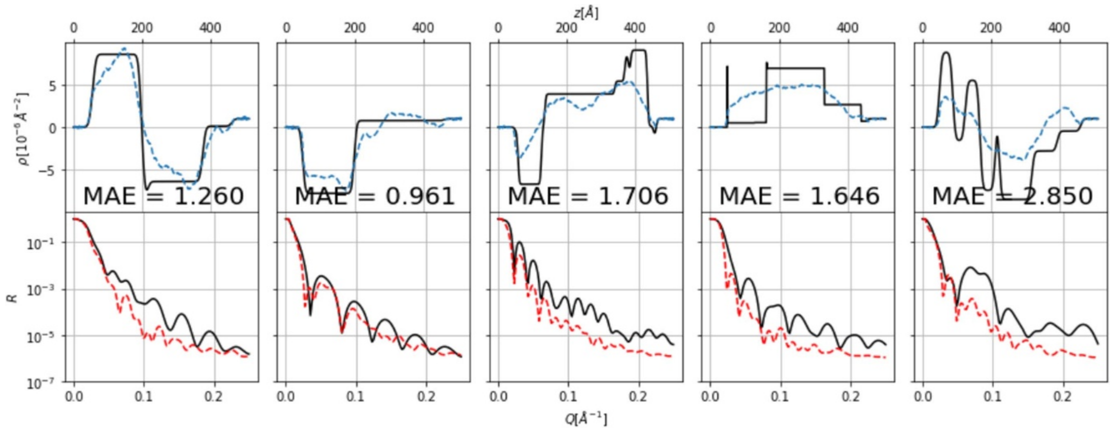


Figure A1. The training of the ANN failed with a set of 100 K random SLD profiles of between 1 and 16 slabs of random thicknesses and heights between −10 and 10. Five test reflectivity curves (black lines; bottom panels) were shown to the trained network for the first time; from those curves, the network guessed the SLD profiles that produced them (blue dashed lines; top panels). In the figure, the MAEs with respect to the true SLD profiles (black lines; top panels) are also shown. Additional reflectivity curves are shown (red dashed lines, bottom panels), calculated from the SLD profiles predicted by the ANN.

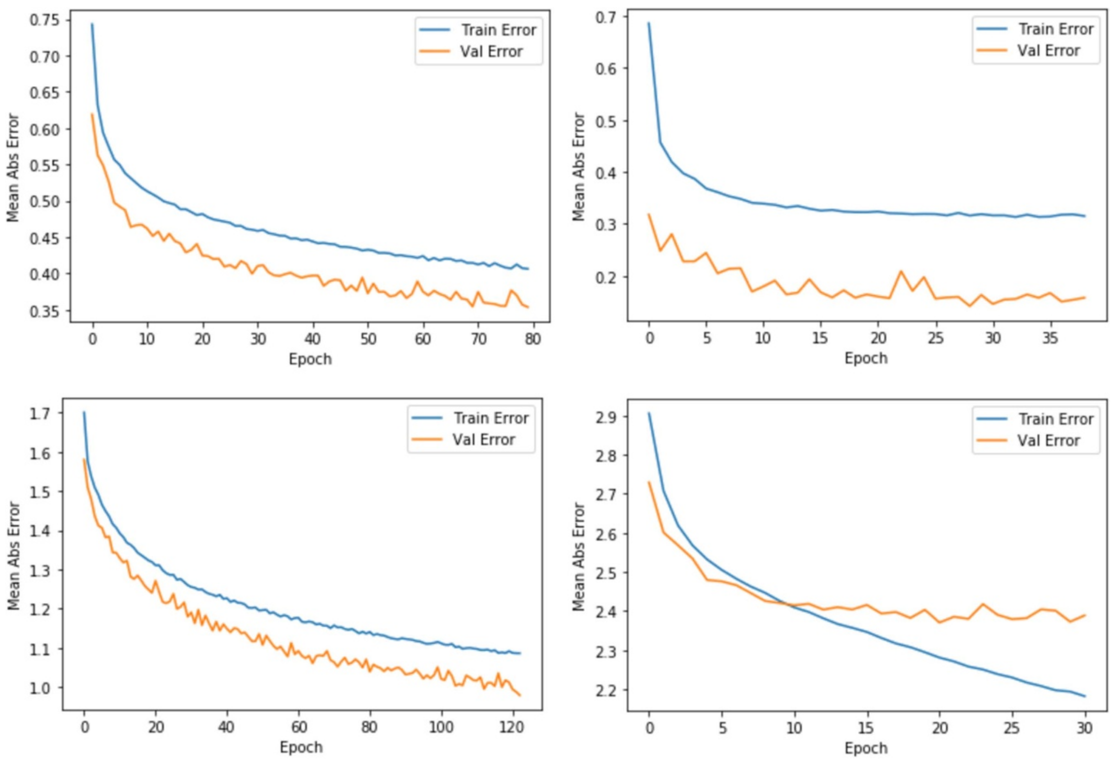


Figure A2. The MAE evolution throughout the epochs as the training proceeds is shown for the different families of SLD profiles studied in this work: odd functions (top-left panel), single films (top-right panel), lamellar structures (bottom-left panel) and random SLD profiles (bottom-right panel). See the text for discussion.

consistent with the targets and none of the recovered reflectivity curves were consistent with the input data. The MAE for the test set, composed of 5 K samples, was around 2.4.

A.2. Metrics evolution

After the ANN architecture is defined, the training proceeds, using the training set to modify the weights of the network and the validation set to asses its performance on unseen data to avoid over-fitting. In this work, the metric used to keep track of the performance of the ANN is the MAE, which is computed after each epoch.

The smoking gun of an over-fitting network is a higher performance (lower error) reported for the training set than that reported for the validation set. By comparison, a healthy network performs as well for

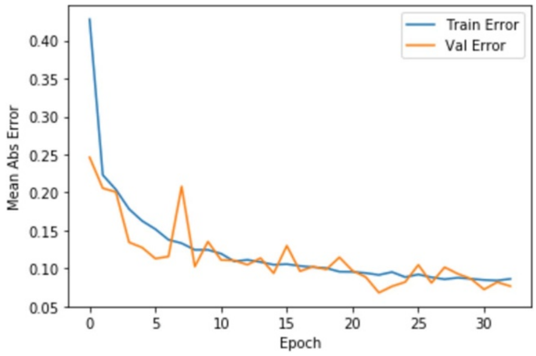


Figure A3. For the case of single-film SLD profiles, the MAE evolution throughout the epochs as the training proceeds is shown for a special ANN in which the dropout layer is not present. See the text for discussion.

the validation set as it does for the training set (in fact, the performance during training is usually a bit better than during validation), and the curves that follow both errors evolve in a similar fashion as epochs progress. To assess the health of the training processes carried out in the present work, the evolution of the MAE for each of the datasets is analyzed in this section (see figure A2).

In this work, there is one case of an over-fitting network, namely, the network trained on pure random data: its performance continues to improve at every epoch for the training set, while stalling at a constant error for the validation set. This behavior demonstrates that the network is memorizing the training set, rather than learning a pseudo-inverse transformation. As discussed in section 4.2, this is to be expected since, due to the ill-posedness of the problem, a pseudo-inverse simply does not exist in this case.

The errors reported for the rest of the networks show lower performance (higher error) for the training set than for the validation set. Such a behavior implies that the network is performing better on unknown data (the validation set) than on known data (the training set), which is counterintuitive. The hypothesis is that such counterintuitive behavior is caused by the dropout layer which, being activated only at training time, lowers the performance on the training set—a condition that is perceived as better validation performance. To test the hypothesis, an extra experiment was carried out in which the dropout layer was removed from an ANN which was later trained using the single-film data set. The evolution of the error during the training process proceeded to be the same as that of the healthy network described above and the expected relation between training and validation errors was recovered: $\mathrm{MAE}_{\mathrm{valid}} \leqslant \mathrm{MAE}_{\mathrm{train}}$ (see figure A3).

ORCID iD

Juan Manuel Carmona Loaiza https://orcid.org/0000-0002-1202-3192

References

[1] Parratt L G 1954 *Phys. Rev.* **95** 359
[2] Penfold J and Thomas R K 1990 *J. Phys.: Condens. Matter.* **2** 1369–412
[3] Tanner B 2018 *Handbook of Advanced Nondestructive Evaluation* ed N Ida and N Meyendorf (Switzerland: Springer) pp 1–34
[4] Majkrzak C, Berk N and Perez-Salas U 2003 *Langmuir* **19** 7796–810
[5] Majkrzak C and Berk N 2003 *Physica B: Condens. Matter* **336** 27–38
[6] Pospelov G, Van Herck W, Burle J, Carmona Loaiza J M, Durniak C, Fisher J M, Ganeva M, Yurov D and Wuttke J 2020 *J. Appl. Crystallogr.* **53** 262–76
[7] Nelson A R and Prescott S W 2019 *J. Appl. Crystallogr.* **52** 193–200
[8] Cybenko G 1989 *Math. Control Signals Syst. (MCSS)* **2** 303–14
[9] Funahashi K I 1989 *Neural Netw.* **2** 183–92
[10] Hornik K, Stinchcombe M and White H *et al* 1989 *Neural Netw.* **2** 359–66
[11] Barron A R 1994 *Mach. Learn.* **14** 115–33
[12] Cohen N, Sharir O and Shashua A 2016 On the expressive power of deep learning: a tensor analysis *Proc. of Machine Learning Research Columbia University, New York* pp 698–728
[13] Eldan R and Shamir O 2016 The power of depth for feedforward neural networks *Proc. of Machine Learning Research Columbia University, New York* pp 907–40
[14] Telgarsky M 2016 Benefits of depth in neural networks *29th Conf. on Learning Theory (Proc. of Machine Learning Research (PMLR) vol 49)* pp 1517–39
[15] Seongmin O 2020 A graph similarity for deep learning *Advances in Neural Information Processing Systems* vol 33 (Red Hook, NY: Curran Associates, Inc.) pp 1–12

[16] Hanin B 2019 *Mathematics* **7** 992
[17] Tensorflow classification tutorial (available at: https://www.tensorflow.org/tutorials/keras/classification)
[18] Li H, Schwab J, Antholzer S and Haltmeier M 2020 *Inverse Problems* **36** 065005
[19] Khan T A and Ling S H 2019 *Algorithms* **12** 88
[20] Greco A, Starostin V, Karapanagiotis C, Hinderhofer A, Gerlach A, Pithan L, Liehr S, Schreiber F and Kowarik S 2019 *J. Appl. Crystallogr.* **52** 1342–47
[21] Kingma D P and Ba J 2014 (arXiv:1412.6980)