Determining cross sections from transport coefficients using deep neural networks

P W Stokes¹, D G Cocks², M J Brunger³ and R D White¹

¹College of Science and Engineering, James Cook University, Townsville, QLD 4811, Australia  
²Research School of Physics and Engineering, Australian National University, Canberra, ACT 2601, Australia  
³College of Science and Engineering, Flinders University, Bedford Park, Adelaide, SA 5042, Australia

E-mail: peter.stokes@my.jcu.edu.au

Abstract. We present a neural network for the solution of the inverse swarm problem of deriving cross sections from swarm transport data. To account for the uncertainty inherent to this somewhat ill-posed inverse problem, we train the neural network using cross sections from the LXCat project, paired with associated transport coefficients found by the numerical solution of Boltzmann’s equation. The use of experimentally measured and theoretically calculated cross sections for training encourages the network to avoid unphysical solutions, such as those containing spurious energy-dependent oscillations. We successfully apply this machine learning approach to simulated swarm data for electron transport in helium, separately determining its elastic momentum transfer and ionisation cross sections to within an accuracy of 4% over the range of energies considered. Our attempt to extend our method to argon was less successful, although the reason for that observation is well-understood. Finally, we explore the feasibility of simultaneously determining the cross sections for multiple scattering processes using this approach.

Keywords: swarm analysis, inverse problem, Boltzmann equation, machine learning

Submitted to: Plasma Sources Sci. Technol.
1. Introduction

Control and optimisation of plasma processing is dependent on accurate modelling of the associated charged particle transport. Fundamental to this is the provision of complete and accurate sets of cross sections. These cross sections must have embedded particle, momentum and energy balance built into them. Experiment (e.g. crossed beam [1]) and theory (e.g. convergent close-coupling [2–5]) can provide some of this required information however there is often still uncertainty associated with these cross-sections (e.g. where extrapolation is required, or where theories break down) and here “educated guesses” are often used. Swarm experiments provide accurate yet independent data to assess the self-consistency of these cross sections [6], which in principle makes them very useful to evaluate their utility.

In this study, we assess the inference/extraction of microscopic information from measured swarm data. The first attempts at deriving electron scattering cross sections from swarm transport coefficients were made in the 1920s by Mayer [7], Ramsauer [8] and Townsend et al. [9]. Early approaches such as these assumed a simplified form of the electron energy distribution function (EEDF), such as a Maxwellian or Druyvesteyn distribution, from which theoretical transport coefficients could be calculated and contrasted with those measured experimentally. By iteratively adjusting the cross sections until the calculated transport coefficients coincided with those from experiment, a plausible solution to this inverse swarm problem could be found. The accuracy of this iterative approach was improved in the 1960s when Phelps, alongside numerous collaborators, began determining the EEDF from the numerical solution of Boltzmann’s equation [10–14]:

\[
\left( \frac{\partial}{\partial t} + \mathbf{v} \cdot \frac{\partial}{\partial \mathbf{r}} - \frac{eE}{m} \cdot \frac{\partial}{\partial \mathbf{v}} \right) f(t, \mathbf{r}, \mathbf{v}) = \left( \frac{\partial f}{\partial t} \right)_{\text{coll}},
\]

where \( f(t, \mathbf{r}, \mathbf{v}) \) is the phase-space distribution function of the electron swarm, \( e \) is the elementary charge, \( m \) is the electron mass, \( E \) is the applied electric field, and \( \left( \frac{\partial f}{\partial t} \right)_{\text{coll}} \) contains operators for all the considered electron collision processes.

When the available swarm transport data is limited, the inverse swarm problem becomes ill-posed with solutions that are non-unique and that consist of cross sections that are sensitive to small variations in the transport coefficients, as illustrated in Figure 1. In such situations this lack of information can to some extent be made up for by the experience and intuition of an expert. Due to this required expertise, as well as the tedious nature of this trial and error approach to swarm analysis, a number of automated methods for solving the inverse swarm problem have been proposed [15–21]. Of particular relevance to the present work is an investigation in the early 1990s by Morgan [19], who solved the inverse swarm problem using an artificial neural network that had been trained on exemplar mappings between swarm data and cross sections. He concluded that such a machine learning approach was well-suited for obtaining an approximate cross section, that could subsequently be refined further by nonlinear least-squares optimisation. Since Morgan’s pioneering investigation, there have been major developments in the field of machine learning that have allowed for much larger and more powerful models to be realised [22–27]. Given that there is now also a wealth of cross section data, made available by the LXCat project [28–30], this data-driven approach to the inverse swarm problem warrants renewed investigation.

The remainder of this paper is structured as follows. In the following section, we present a new neural network for the solution of the inverse swarm problem. In Section 3 we apply this neural network to determine cross sections of helium and argon using simulated swarm transport data. Finally, Section 4 presents some conclusions and a discussion of future generalisations and
Determining cross sections from transport coefficients using deep neural networks

Figure 1. Euler diagram illustrating the somewhat ill-posed nature of the inverse swarm problem. Arrows denote the forward problem of mapping from a set of cross sections to a set of corresponding transport coefficients. This mapping is well-posed and can be achieved numerically by solving Boltzmann’s equation or by performing a Monte Carlo simulation, among other techniques. Conversely, the inverse problem is ill-posed due to its potentially ambiguous solutions and its sensitivity to the transport data. By training the neural network, Eq. on cross sections from the LXCat project, we restrict the solution to a subset of physically-plausible cross sections, thereby improving the posedness of the inverse problem.

applications of this work.

2. Neural network regression of cross sections

Artificial neural networks are highly-parameterised mathematical functions capable of universal function approximation. By carefully adjusting the parameters of a neural network, in a process called training, it is possible to approximate arbitrary multivariate vector-valued functions. In other words, neural networks are capable of performing nonlinear mappings between vector spaces. In this section, we present a neural network for the solution of the inverse swarm problem that learns from the inverse of the forward mapping from cross sections to transport coefficients. This process is schematically depicted in Figure 1. Our neural network is of a similar architecture to that employed by Morgan and, as such, we will begin by outlining the structure of Morgan’s neural network before describing how ours differs.

2.1. Model introduction and overview

To perform the cross section regression given the transport coefficients, we use a fully-connected neural network (FCNN). The simplest FCNN is an affine transformation of an input vector $x$ to an output vector $y$:

$$y = Wx + b,$$

where the matrix $W$ and vector $b$ contain the network parameters. More generally, to describe nonlinear relationships, additional affine transformations are applied, each interleaved with a
Determining cross sections from transport coefficients using deep neural networks

nonlinear activation function. For example, Morgan considered an FCNN of the form [19]:

\[ y = (A_3 \circ \tanh \circ A_2 \circ \tanh \circ A_1) (x), \]  

(3)

where a hyperbolic tangent activation function is applied element-wise throughout, and each affine transformation \( A_n (x) = W_n x + b_n \) has associated parameters \( W_n \) and \( b_n \). This network is said to have 4 layers of neurons, corresponding to the vectors of numbers reached at each stage of the computation. For example, the above network has an input layer \( x \), an output layer \( y \) and 2 intermediate hidden layers \( x_1 \) and \( x_2 \), which arise as follows:

\[ x_1 = \tanh (A_1 (x)) , \]  

(4)

\[ x_2 = \tanh (A_2 (x_1)) , \]  

(5)

\[ y = A_3 (x_2) . \]  

(6)

While the input and output layers, \( x \) and \( y \), are generally of a size fixed by the nature of the problem at hand, the number and size of the hidden layers must be specified. With too few hidden neurons, the predictive power of the model is hindered. Conversely, a model that is too complex can overspecialise to the training data, resulting in poor generalisation to gases outside of the training dataset. Fortunately, this latter problem becomes less likely to occur as the amount of training data increases.

The main architectural difference between our network and Morgan’s is the structure of the input and output vectors, \( x \) and \( y \). Morgan considered an input vector \( x \) containing measured drift velocities \( W \) and characteristic energies \( D_T / \mu \), where \( D_T \) is the bulk transverse diffusion coefficient and \( \mu \) is the bulk electron mobility, and an output vector \( y \) as a discrete representation of the cross section of interest, \( \sigma (\varepsilon) \) [19]:

\[ x = \begin{bmatrix} W (E_1 / n_0) \\ D_T (E_1 / n_0) / \mu (E_1 / n_0) \\ W (E_2 / n_0) \\ D_T (E_2 / n_0) / \mu (E_2 / n_0) \\ \vdots \end{bmatrix}, \quad y = \begin{bmatrix} \sigma (\varepsilon_1) \\ \sigma (\varepsilon_2) \\ \vdots \end{bmatrix}. \]  

(7)

Here, the input transport coefficient measurements are performed at various reduced electric fields \( E_1 / n_0, E_2 / n_0, \ldots \), where \( n_0 \) is the number density of the background neutrals, and the output cross section is sampled at discrete energies \( \varepsilon_1, \varepsilon_2, \ldots \). As input in our case, in addition to the transport coefficients of bulk drift velocity \( W \) and bulk longitudinal diffusion \( n_0 D_L \), we also consider the first Townsend ionisation coefficient \( \alpha / n_0 \). For the output cross section, rather than discretising as above, we use the neural network itself as a discrete approximation of the cross section as a function of energy. Energy, \( \varepsilon \), now becomes an input to the neural network alongside the transport coefficients, and the output is now a single number corresponding to the cross section evaluated at this energy, \( \sigma (\varepsilon) \):

\[ x = \begin{bmatrix} \varepsilon \\ W (E_1 / n_0) \\ n_0 D_L (E_1 / n_0) \\ \alpha (E_1 / n_0) / n_0 \\ W (E_2 / n_0) \\ n_0 D_L (E_2 / n_0) \\ \alpha (E_2 / n_0) / n_0 \\ \vdots \end{bmatrix}, \quad y = \sigma (\varepsilon) . \]  

(8)
Determining cross sections from transport coefficients using deep neural networks

Figure 2. Diagram (not to scale) of the fully-connected neural network used for regression of a cross section (yellow) as a function of energy (red), given an associated set of transport coefficients (blue). The number and size of intermediate hidden layers (grey) affects the capacity of the model to perform this nonlinear mapping. For the cross section fits in Section 3, 3 hidden layers were used, each containing 100 neurons. For the simultaneous fitting of multiple cross sections in Section 3.3 additional output neurons were added accordingly.

Although it is not shown here, before being used for regression all the data is log-transformed with Eq. (12). This is detailed in the following section.

This architectural change of making the energy an explicit input, illustrated in Figure 2, provides a number of benefits. Firstly, rather than having to decide on a one-size-fits-all cross section discretisation, the neural network learns a suitable discretisation that is tailored to the training data. Secondly, now that there is only one output neuron per cross section, far fewer weights are required in the final affine transformation, making the network simpler to train, faster to evaluate, and less likely to overspecialise to the training data. This is most beneficial when multiple output cross sections are considered, such as in Section 3.3. Finally, it is now relatively trivial to constrain the cross section training data when there are known upper and lower bounds obtained from, for example, crossed beam experiments [1]. This is awkward to do when simultaneously predicting a cross section at multiple energies, as in Eq. (7), because then constraints must also be enforced simulatenously for every energy considered.

Our neural network, as used in Section 3, contains 3 hidden layers and uses the “swish” activation function throughout, where

\[
swish(x) = x / (1 + e^{-x})
\]

and

\[
\sigma(x) = (A_4 \circ \text{swish} \circ A_3 \circ \text{swish} \circ A_2 \circ \text{swish} \circ A_1)(x).
\] (9)

2.2. Data preparation and model training

To train the neural network, Eq. (9), we require example inputs and outputs, a cost function that indicates the performance of the neural network on this training data, and an optimisation algorithm to minimise this cost function by varying the parameters of the neural network.

We consider first the issue of determining suitable cross sections for the training. For the purpose of fitting elastic momentum transfer cross sections, Morgan generated training examples using a power-law model of the form

\[
\sigma(\varepsilon) \propto \varepsilon^p,
\]

where \(-1 \leq p \leq 1\) [10]. Similar options exist for
Figure 3. Scatter plot illustrating the rough power-law relationship between the magnitude and threshold energy of the excitation and ionisation cross sections on LXCat. Pearson’s correlation coefficient between the log-transformed maximum cross section and log-transformed threshold energy is \( r = -0.738 \) for excitation processes and \( r = -0.755 \) for ionisation processes. By training the neural network, Eq. (9), on cross sections sampled using Eq. (10), we ensure this negative correlation is reflected in the training data.

Other processes, e.g., Machacek et al. described the cross section for total positronium formation by positron scattering using a surge function \( \sigma(\varepsilon) \propto \varepsilon^p e^{-b\varepsilon} \), where \( p \) and \( b \) are shape parameters [33]. Parameterised cross sections such as these are ideal for machine learning, as they can be sampled from indefinitely for training data. That said, we choose instead to train using a finite set of cross sections from the LXCat project [28–30], so as to expose the neural network to as much measured and calculated cross section physics as possible. To make up for this finite amount of cross section data, we generate training data by random interpolation of LXCat cross sections. Specifically, given a unique pair of cross sections, \( \sigma_1(\varepsilon) \) and \( \sigma_2(\varepsilon) \), as well as a uniformly distributed mixing ratio \( r \in [0,1] \), we form each training cross section by evaluating:

\[
\sigma(\varepsilon) = \sigma_1^{1-r}(\varepsilon + \varepsilon_1 - \varepsilon_1^{1-r}\varepsilon_2) \sigma_2^{r}(\varepsilon + \varepsilon_2 - \varepsilon_1^{1-r}\varepsilon_2),
\]

where \( \varepsilon_1 \) and \( \varepsilon_2 \) are the respective threshold energies of \( \sigma_1(\varepsilon) \) and \( \sigma_2(\varepsilon) \). This is a weighted geometric average of the two cross sections that has been shifted to have a new threshold of \( \varepsilon_1^{1-r}\varepsilon_2 \), which is itself a weighted geometric average of the separate threshold energies. We use the same ratio \( r \) when mixing both cross sections and threshold energies, as there is a negative correlation between the magnitude of a cross section and its threshold energy, depicted in Figure 3 that we would like to see reflected in the training data.

Each cross section generated using Eq. (10) must be sampled at discrete energies in order to be used for training. To aid the neural network in learning the global energy-dependence of the cross sections, we sample at a large number of energies over the domain of interest. These energies are also selected geometrically:

\[
\varepsilon = \varepsilon_{\text{min}}^{1-s}\varepsilon_{\text{max}}^s,
\]

where \( s \) is uniformly distributed in \([0,1]\).
where \([\varepsilon_{\text{min}}, \varepsilon_{\text{max}}]\) is the energy domain under consideration and \(s \in [0, 1]\) is a uniformly distributed random number.

Finally, from our chosen cross section or cross sections, we must determine corresponding transport coefficients to complete our input/output training pair. For efficient and robust generation of this swarm data, here we apply the two-term approximation \([35, 36]\) to Boltzmann’s equation (1) and then perform backward prolongation \([37]\) of the EEDF by inward integration from high to low energies. This integration is performed numerically, using an adaptive order adaptive energy Adams-Moulton method \([38]\), as implemented in the *DifferentialEquations.jl* software ecosystem \([39–41]\).

Since cross sections, energies and transport coefficients all span many orders of magnitude, we make sure to log-transform everything before training. That is, given a strictly positive quantity \(z\), we apply a log-transformation that is scaled and shifted to ensure all the training data lies within the domain \([-1, 1]\):

\[
z \mapsto \log \left( \frac{z}{\sqrt{z_{\max} z_{\min}}} \right),
\]

where \(z_{\min}\) and \(z_{\max}\) are the extrema of all instances of \(z\) in the training dataset. If any instance of \(z\) happens to be zero, we replace it with a suitably small positive number before applying the above transformation.

To train the neural network, we minimise its mean absolute error on the training set of cross sections:

\[
\frac{1}{N} \sum_{i=1}^{N} |y_i - \sigma(x_i)|,
\]

where the index \(i\) ranges over the entire set of \(N\) training examples \((x_i, y_i)\), and \(\sigma(x_i)\) is the corresponding neural network prediction for each. We have purposefully avoided using the mean squared error, as that encourages the neural network to preferentially improve upon its worst predictions. This sounds reasonable, and is the motivation behind least-squares regression, but as we are solving an inverse problem there is an inherent uncertainty to the solution that fundamentally limits how well the network can perform. Encouraging the neural network to do the impossible of accurately and consistently fitting the least certain cross section values is expected to hinder its performance overall.

We implement and train the model using the *Flux.jl* machine learning framework \([42]\). In constructing the neural network, we initialise the parameters in \(b_n\) to zero and those in \(W_n\) to uniform random numbers as described by Glorot and Bengio \([43]\). Then, we train by minimising Eq. (13) using the Adam optimiser \([44]\) with step size \(\alpha = 10^{-3}\) and exponential decay rates \(\beta_1 = 0.9\) and \(\beta_2 = 0.999\). By definition, training improves the performance of the neural network on the training examples, with the hope that good performance on the training set will be correlated with good performance in general. However, with too much training there lies a danger of the neural network learning features that are unique to the training data, at the expense of generalisation \([45]\). A common technique to avoid this is to simply stop the training before the network’s ability to generalise worsens, as quantified by some other measure independent of the data used for training \([46]\). For this measure, we use the necessary condition that the solution to the inverse swarm problem must also satisfy the forward swarm problem. Thus, we continually apply the Boltzmann solver, while training, to determine how well the fitted cross sections reproduce the transport coefficients that were used to perform the fit. We then stop training when the mean squared error in this reproduced swarm data reaches a minimum, as demonstrated in Figure 4.
3. Application to simulated swarm data

In this section, we simulate pulsed Townsend swarm experiments for electron transport in helium and argon at a temperature $T = 300K$, across a range of $50$ reduced electric fields $E/n_0$ spaced exponentially between $10^{-3}Td$ and $10^3Td$ inclusive, where $1Td = 1$ Townsend $= 10^{-21}$Vm$^2$. Using the resulting transport coefficients of bulk drift velocity $W$, bulk longitudinal diffusion coefficient $n_0D_L$ and Townsend coefficient $\alpha/n_0$ as inputs, we then apply the neural network Eq. (9) toward fitting various cross sections of helium and argon over the energy domain $[10^{-1}$eV, $10^2$eV]. For both gases, cross section data are sourced from the Biagi v7.1 database [47].

3.1. Fitting elastic momentum transfer cross sections

We consider first fitting helium’s elastic momentum transfer cross section (MTCS), while assuming its inelastic cross sections are known. For the training data, we use Eq. (10) to randomly sample $10^4$ elastic MTCS from those plotted in Figure [5] from LXCat [2–5, 47–69]. Of course, helium is excluded when performing this sampling. For each of these generated cross sections, corresponding swarm transport coefficients are found. Finally, each cross section is sampled at $100$ random energies between $10^{-1}$eV and $10^2$eV, using Eq. (11), resulting in a total of $10^6$ training examples. We split these training exemplars into $10$ batches of $10^5$ examples each, making sure to shuffle the training data beforehand so that each batch is representative of the training set as a whole. Training then proceeds iteratively by repeatedly cycling through each batch in turn, updating the neural network
Determining cross sections from transport coefficients using deep neural networks

Figure 5. All currently available elastic momentum transfer cross sections from the LXCat project [2–5, 47–69]. These are randomly sampled from using Eq. (10) and the resulting cross sections are employed in the training of the neural network depicted in Figure 2 for the purpose of fitting elastic momentum transfer cross sections. A similar approach is taken when fitting cross sections for other collision processes.

parameters each time using the optimiser. Training is continued until the mean squared error in the resulting transport coefficients reaches a minimum, as demonstrated in Figure 4. The final result of this process, for the fitted cross section and corresponding transport coefficients, is plotted in Figure 6. The neural network is found to be accurate here to within 4%, for both the elastic MTCS and its corresponding transport coefficients.

We now repeat the above process for argon (whose elastic MTCS is now excluded from the training data instead of helium’s) and plot the results in Figure 7. Unlike helium, argon’s elastic MTCS is an outlier among the training data, with its Ramsauer-Townsend minimum dipping below the rest of the LXCat cross sections in Figure 5. Because of this, the neural network struggles to determine the correct magnitude of that minimum, believing it to be more than twice as large as it should. In effect, our choice of training data has incorrectly constrained the fitted cross section. The error at larger energies, above 0.4eV, is however fortunately considerably better, lying to within 20%. This larger error, as compared to helium, is likely due in part to the neural network compensating for the incorrect Ramsauer-Townsend minimum, so as to better match the transport coefficients. Lastly, it should be noted that a larger error is to be expected for argon. This follows as the chosen swarm data provides less information about argon’s elastic MTCS than it does for helium’s, as evidenced by the mean energy of electrons at the highest applied field, $E/n_0 = 1000$Td, which is $\sim 133$eV for helium and only $\sim 15$eV for argon.
Figure 6. Neural network regression of helium’s elastic MTCS, alongside corresponding plots of the transport coefficients. The neural network determines the cross section to within an accuracy of 4% here, while also being consistent with the provided swarm data to within the same margin of error.
Figure 7. Neural network regression of argon’s elastic MTCS, alongside corresponding plots of the transport coefficients. Due to the uniqueness of argon’s Ramsauer-Townsend minimum, the training data, plotted in Figure 5 is insufficient to perform an accurate cross section fit. Away from that minimum, the fit is accurate to within 20%. It is expected that the fit quality would improve if more suitable cross sections were to be available and thus used for training.
3.2. Fitting threshold cross sections

We turn now to fitting the ionisation cross section of helium, as a representative example of data with a clear threshold energy. We generate our training data using LXCat as before, only now employing the ionisation cross sections instead \[2-5, 47-52, 55-71\]. Fortunately, besides from this change in the utilised cross section data, no other alterations are required to the neural network or training procedure in order to fit a threshold cross section. Figure 8 plots the result of the neural network regression, alongside corresponding transport coefficients. Above threshold, the fit of helium’s ionisation cross section is in error by at most 4\%, hence on a par with the previous fit of helium’s elastic MTCS. It is very promising to see that the neural network is able to successfully identify the ionisation threshold from the transport data. Machine learning approaches thus have the potential for determining unknown threshold energies, such as the neutral dissociation threshold of tetrahydrofurfuryl alcohol (THFA), a molecule of biological interest \[72-79\].

3.3. Fitting multiple cross sections simultaneously

Finally, we consider simultaneously fitting helium’s elastic momentum transfer, \(n = 2\) singlet excitation \((2^1S \text{ and } 2^1P)\), \(n = 2\) triplet excitation \((2^3S \text{ and } 2^3P)\) and ionisation cross sections. Accordingly, we now also incorporate excitation cross section data from LXCat into the training procedure \[2-5, 47-52, 53-59, 61-66, 68, 70, 71, 80\]. As in the previous fits, we assume nothing here but the number and type of each cross section. Correspondingly, the output layer of our neural network is now increased to contain 4 neurons, one for each cross section being fitted. How these cross sections are ordered in the output layer does not matter, so long as the ordering is kept consistent when training and applying the network. With this in mind, note that it is not clear how to consistently order each training pair of \(n = 2\) excitation cross sections. This is because excitation processes can be interchanged with one another without any affect on the transport coefficients. If we are not careful, the neural network could proceed to correctly predict the excitation cross sections of helium, but be penalised for doing so because the cost function, Eq. (13), happens to assume the opposite ordering. As such, for this fit, we modify the cost function so as to make it symmetric with respect to the excitation cross sections. In effect, for each training exemplar, we try both permutations of predicted excitation cross sections and select the one that minimises the cost function the most.

Figure 9 plots the result of the neural network regression for this case. Here we observe the fitted elastic MTCS to be accurate to within roughly 10\% over the entire range of energies. This is an expected increase over the maximum 4\% error observed in Section 3.1 where the elastic MTCS was fitted exclusively. For inelastic processes above threshold, we see larger cross section errors of within 20\% for total \(n = 2\) excitation and 25\% for ionisation. Of the pair of \(n = 2\) excitation cross sections determined by the network, one was comparable to the total \(n = 2\) excitation cross section, while the other was made very small. This inability of the network to unfold the separate \(n = 2\) singlet and \(n = 2\) triplet excitation cross sections is likely due in part to their very similar threshold energies. Overall, it is expected that the ability of the neural network to accurately unfold multiple cross sections would improve with the consideration of additional swarm data, such as transport coefficients of helium mixed with other gases.
4. Conclusion

We have presented a machine learning approach to the inverse swarm problem where a neural network is used to approximate unknown cross sections as a functions of energy, given corresponding swarm transport data. By training this network on physical cross sections from the LXCat project [28–30], it was found to yield physically-plausible solutions to the inverse swarm problem that are consistent with the specified transport coefficients. As a demonstration, we applied this network to determine cross sections of helium and argon from simulated pulsed Townsend electron swarm data.
Determining cross sections from transport coefficients using deep neural networks

Figure 9. Neural network regression of helium’s elastic momentum transfer, $n = 2$ singlet excitation, $n = 2$ triplet excitation and ionisation cross sections, alongside corresponding plots of the transport coefficients. As expected, the overall error has increased here compared to fitting helium’s elastic momentum transfer and ionisation cross sections individually. The network was unable to unfold the separate cross sections for $n = 2$ singlet and triplet excitation, although the values it predicts are consistent with the threshold and magnitude of the total $n = 2$ excitation cross section.
over a range of reduced electric fields $E/n_0$. We found that a suitably-trained neural network could determine individual elastic momentum transfer and ionisation cross sections to within 4% accuracy for the swarm data considered. From the same swarm data, we were also able to simultaneously fit the elastic MTCS, total excitation, and ionisation cross sections of helium to within 10%, 20% and 25% accuracy respectively. Promisingly, this data-driven approach to swarm analysis not only avoids the tedium of conventional iteration, but also appears to provide an adequate substitute for the intuition of an expert.

A fundamental limitation of our neural network is that it provides only a single solution to a problem for which many plausible solutions may exist. Ideally, we would like to be able to quantify this uncertainty in the solution and place corresponding error bars on the predicted cross sections. One such approach involves determining a probability distribution of cross sections, from which plausible solutions can be sampled. This process is known as conditional density estimation and a number of machine learning models are capable of performing it, including mixture density networks [81], conditional variational autoencoders [82], conditional generative adversarial networks [83] and conditional flow-based generative models [84–86]. Work on addressing this present limitation in our approach is currently underway.

Finally, before our neural network can be applied to actual swarm measurements, the presence of experimental error in these measurements must be accounted for. Fortunately, by simply introducing simulated experimental error estimates in the transport coefficients used for the training, we should expect the neural network to adapt accordingly. Indeed, this is the principle behind neural network approaches to noise reduction in images [87]. This issue will be addressed in our future studies.

Acknowledgments

The authors gratefully acknowledge the financial support of the Australian Research Council through the Discovery Projects Scheme (Grant #DP180101655).

References

[1] Filippelli A R, Lin C C, Anderson L W and McConkey J W 1994 Principles and Methods for Measurement of Electron Impact Excitation Cross Sections for Atoms and Molecules by Optical Techniques Advances in Atomic, Molecular and Optical Physics vol 33 (Elsevier) pp 1–62 URL https://doi.org/10.1016/S1049-250X(08)60032-4
[2] Bray I and Stelbovics A T 1992 Physical Review A 46 6995–7011 ISSN 1050-2947 URL https://doi.org/10.1103/PhysRevA.46.6995
[3] Fursa D V and Bray I 1995 Physical Review A 52 1279–1297 ISSN 1050-2947 URL https://doi.org/10.1103/PhysRevA.52.1279
[4] Zammit M C, Fursa D V and Bray I 2014 Physical Review A 90 022711 ISSN 1050-2947 URL https://doi.org/10.1103/PhysRevA.90.022711
[5] Zammit M C, Savage J S, Fursa D V and Bray I 2016 Physical Review Letters 116 233201 ISSN 0031-9007 URL https://doi.org/10.1103/PhysRevLett.116.233201
[6] White R D, Cocks D, Boyle G, Casey M, Garland N, Konovalov D, Philippa B, Stokes P, de Urquijo J, González-Magaña O, McEachran R P, Buckman S J, Brunger M J, García G,
REFERENCES

Dukjo S and Petrovic Z L 2018 Plasma Sources Science and Technology 27 053001 ISSN 1361-6595 URL https://doi.org/10.1088/1361-6595/aabdd7

[7] Mayer H F 1921 Annalen der Physik 369 451–480 ISSN 00033804 URL https://doi.org/10.1002/andp.19213690503

[8] Ramsauer C 1921 Annalen der Physik 369 513–540 ISSN 00033804 URL https://doi.org/10.1002/andp.19213690603

[9] Townsend J and Bailey V 1922 The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science 43 593–600 ISSN 1941-5982 URL https://doi.org/10.1080/14786442208633916

[10] Frost L S and Phelps A V 1962 Physical Review 127 1621–1633 ISSN 0031899X URL https://doi.org/10.1103/PhysRev.127.1621

[11] Engelhardt A G and Phelps A V 1963 Physical Review 131 2115–2128 ISSN 0031-899X URL https://doi.org/10.1103/PhysRev.131.2115

[12] Engelhardt A G, Phelps A V and Risk C G 1964 Physical Review 135 A1566–A1574 ISSN 0031-899X URL https://doi.org/10.1103/PhysRev.135.A1566

[13] Hake R D and Phelps A V 1967 Physical Review 158 70–84 ISSN 0031-899X URL https://doi.org/10.1103/PhysRev.158.70

[14] Phelps A V 1968 Reviews of Modern Physics 40 399–410 ISSN 00346861 URL https://doi.org/10.1103/RevModPhys.40.399

[15] Duncan C W and Walker I C 1972 Journal of the Chemical Society, Faraday Transactions 2 68 1514 ISSN 0300-9238 URL https://doi.org/10.1039/f29726801514

[16] O’Malley T F and Crompton R W 1980 Journal of Physics B: Atomic and Molecular Physics 13 3451–3464 ISSN 0022-3700 URL https://doi.org/10.1088/0022-3700/13/17/023

[17] Taniguchi T, Suzuki M, Kawamura K, Noto F and Tagashira H 1987 Journal of Physics D: Applied Physics 20 1085–1087 ISSN 0022-3727 URL https://doi.org/10.1088/0022-3727/20/8/020

[18] Morgan W L 1991 Physical Review A 44 1677–1681 ISSN 10502947 URL https://doi.org/10.1103/PhysRevA.44.1677

[19] Morgan W L 1991 IEEE Transactions on Plasma Science 19 250–255 ISSN 19399375 URL https://doi.org/10.1109/27.106821

[20] Morgan W L 1993 Journal of Physics D: Applied Physics 26 209–214 ISSN 00223727 URL https://doi.org/10.1088/0022-3727/26/2/007

[21] Brennan M and Ness K 1993 Australian Journal of Physics 46 249 ISSN 0004-9506 URL https://doi.org/10.1071/PH930249

[22] Hinton G E, Osindero S and Teh Y W 2006 Neural Computation 18 1527–1554 ISSN 08997667 URL https://doi.org/10.1162/neco.2006.18.7.1527

[23] Hinton G E and Salakhutdinov R R 2006 Science 313 504–507 ISSN 00368075 URL https://doi.org/10.1126/science.1127647

[24] Bengio Y, Lamblin P, Popovici D and Larochelle H 2007 Greedy layer-wise training of deep networks Advances in Neural Information Processing Systems 19 (The MIT Press) pp 153–160 ISBN 97802622195683 URL https://doi.org/10.7551/mitpress/7503.003.0024
REFERENCES

[25] Ranzato M A, Poultney C, Chopra S and LeCun Y 2007 Efficient learning of sparse representations with an energy-based model Advances in Neural Information Processing Systems 19 (The MIT Press) pp 1137–1144 ISBN 9780262195683 URL https://doi.org/10.7551/mitpress/7503.003.0147

[26] Lee H, Ekanadham C and Ng A Y 2009 Sparse deep belief net model for visual area V2 Advances in Neural Information Processing Systems 20 ed Platt J C, Koller D, Singer Y and Roweis S T (Curran Associates, Inc.) pp 873–880 ISBN 160560352X URL http://papers.nips.cc/paper/3313-sparse-deep-belief-net-model-for-visual-area-v2.pdf

[27] Glorot X, Bordes A and Bengio Y 2011 Deep Sparse Rectifier Neural Networks Proceedings of the Fourteenth International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research) vol 15 ed Gordon G, Dunson D and Dudík M (Fort Lauderdale, FL, USA: PMLR) pp 315–323 URL http://proceedings.mlr.press/v15/glorot11a.html

[28] Pancheshnyi S, Biagi S, Bordage M, Hagelaar G, Morgan W, Phelps A and Pitchford L 2012 Chemical Physics 398 148–153 ISSN 03010104 URL https://doi.org/10.1016/j.chemphys.2011.04.020

[29] Pitchford L C, Alves L L, Bartschat K, Biagi S F, Bordage M C, Bray I, Brion C E, Brunger M J, Campbell L, Chachereau A, Chaudhury B, Christophorou L G, Carbone E, Dyatko N A, Franck C M, Fursa D V, Gangwar R K, Guerra V, Haefliger P, Hagelaar G J M, Hoels A, Itikawa Y, Kochetov I V, McEachran R P, Morgan W L, Napartovich A P, Puech V, Rabie M, Sharma L, Srivastava R, Stauffer A D, Tennyson J, de Urquijo J, van Dijk J, Viehland L A, Zammit M C, Zatsarinny O and Pancheshnyi S 2017 Plasma Processes and Polymers 14 1600098 ISSN 16128850 URL https://doi.org/10.1002/ppap.201600098

[30] LXCat URL http://www.lxcat.net

[31] Cybenko G 1989 Mathematics of Control, Signals, and Systems 2 303–314 ISSN 0932-4194 URL https://doi.org/10.1007/BF02551274

[32] Hornik K 1991 Neural Networks 4 251–257 ISSN 08936080 URL https://doi.org/10.1016/0893-6080(91)90009-T

[33] Ramachandran P, Zoph B and Le Q V 2017 (Preprint 1710.05941) URL https://arxiv.org/abs/1710.05941

[34] Machacek J R, Blanco F, Garcia G, Buckman S J and Sullivan J P 2016 Journal of Physics B: Atomic, Molecular and Optical Physics 49 064003 ISSN 0953-4075 URL https://doi.org/10.1088/0953-4075/49/6/064003

[35] Hagelaar G J and Pitchford L C 2005 Plasma Sources Science and Technology 14 722–733 ISSN 09630252 URL https://doi.org/10.1088/0963-0252/14/4/011

[36] Robson R E and Ness K F 1986 Physical Review A 33 2068–2077 ISSN 0556-2791 URL https://doi.org/10.1103/PhysRevA.33.2068

[37] Sherman B 1960 Journal of Mathematical Analysis and Applications 1 342–354 ISSN 0022-247X(60)90008-1 URL https://doi.org/10.1016/0022-247X(60)90008-1

[38] Hairer E, Nørsett S P and Wanner G 1993 Solving Ordinary Differential Equations I (Springer Series in Computational Mathematics) vol 8 (Berlin, Heidelberg: Springer Berlin Heidelberg) ISBN 978-3-540-56670-0 URL https://doi.org/10.1007/978-3-540-78862-1

[39] Rackauckas C and Nie Q 2017 Journal of Open Research Software 5 ISSN 2049-9647 URL https://doi.org/10.5334/jors.151
REFERENCES

[40] DelayDiffEq.jl URL https://github.com/JuliaDiffEq/DelayDiffEq.jl
[41] OrdinaryDiffEq.jl URL https://github.com/JuliaDiffEq/OrdinaryDiffEq.jl
[42] Innes M 2018 Journal of Open Source Software 3 602 ISSN 2475-9066 URL https://doi.org/10.21105/joss.00602
[43] Glorot X and Bengio Y 2010 Understanding the difficulty of training deep feedforward neural networks Proceedings of the Thirteenth International Conference on Artificial Intelligence and Statistics (Proceedings of Machine Learning Research vol 9) ed Teh Y W and Titterington M (Chia Laguna Resort, Sardinia, Italy: PMLR) pp 249–256 URL http://proceedings.mlr.press/v9/glorot10a.html
[44] Kingma D P and Ba J L 2015 Adam: A method for stochastic optimization 3rd International Conference on Learning Representations (Preprint 1412.6980) URL https://arxiv.org/abs/1412.6980
[45] Geman S, Bienenstock E and Doursat R 1992 Neural Computation 4 1–58 ISSN 0899-7667 URL https://doi.org/10.1162/neco.1992.4.1.1
[46] Morgan N and Bourlard H 1990 Generalization and Parameter Estimation in Feedforward Nets: Some Experiments Advances in Neural Information Processing Systems 2 ed Touretzky D S (Morgan-Kaufmann) pp 630–637 URL http://papers.nips.cc/paper/275-generalization-and-parameter-estimation-in-feedforward-nets-some-experiments.pdf
[47] Biagi-v7.1 database URL http://www.lxcat.net/Biagi-v7.1
[48] Biagi database URL http://www.lxcat.net/Biagi
[49] Bordage database URL http://www.lxcat.net/Bordage
[50] BSR database URL http://www.lxcat.net/BSR
[51] CCC database URL http://www.lxcat.net/CCC
[52] Christophorou database URL http://www.lxcat.net/Christophorou
[53] COP database URL http://www.lxcat.net/COP
[54] FLINDERS database URL http://www.lxcat.net/FLINDERS
[55] Hayashi database URL http://www.lxcat.net/Hayashi
[56] IST-Lisbon database URL http://www.lxcat.net/IST-Lisbon
[57] Itikawa database URL http://www.lxcat.net/Itikawa
[58] Morgan database URL http://www.lxcat.net/Morgan
[59] Puech database URL http://www.lxcat.net/Puech
[60] QUANTEMOL database URL http://www.lxcat.net/QUANTEMOL
[61] SIGLO database URL http://www.lxcat.net/SIGLO
[62] TRINITI database URL http://www.lxcat.net/TRINITI
[63] Biagi S MAGBOLTZ (versions 7.1, 8.9, 8.97 and 10.6) URL http://magboltz.web.cern.ch
[64] Zatsarinny O and Bartschat K 2004 Journal of Physics B: Atomic, Molecular and Optical Physics 37 4693–4706 ISSN 0953-4075 URL https://doi.org/10.1088/0953-4075/37/23/010
[65] Allan M, Zatsarinny O and Bartschat K 2006 Physical Review A 74 030701 ISSN 1050-2947 URL https://doi.org/10.1103/PhysRevA.74.030701
REFERENCES

[66] Christophorou L G and Olthoff J K 2000 *Journal of Physical and Chemical Reference Data* 29 267–330 ISSN 0047-2689 URL https://doi.org/10.1063/1.1288407

[67] Christophorou L G and Olthoff J K 2000 *Journal of Physical and Chemical Reference Data* 29 553–569 ISSN 0047-2689 URL https://doi.org/10.1063/1.1318910

[68] Alves L L 2014 *Journal of Physics: Conference Series* 565 012007 ISSN 1742-6596 URL https://doi.org/10.1088/1742-6596/565/1/012007

[69] Quantemol URL http://quantemol.com

[70] Phelps database URL http://www.lxcat.net/Phelps

[71] Arthur V. Phelps URL http://web.archive.org/web/20180820113924/http://jilawww.colorado.edu/~avp/

[72] Jones D, Builth-Williams J, Bellm S, Chiari L, Chaluvadi H, Madison D, Ning C, Lohmann B, Ingólfsson O and Brunger M 2013 *Chemical Physics Letters* 572 32–37 ISSN 00092614 URL https://doi.org/10.1016/j.cplett.2013.04.028

[73] Limão-Vieira P, Duflot D, Hubin-Franskin M J, Delwiche J, Hoffmann S V, Chiari L, Jones D B, Brunger M J and Lopes M C A 2014 *The Journal of Physical Chemistry A* 118 6425–6434 ISSN 1089-5639 URL https://doi.org/10.1021/jp501634w

[74] Duque H V, Chiari L, Jones D B, Thorn P A, Pettifer Z, da Silva G B, Limão-Vieira P, Duflot D, Hubin-Franskin M J, Delwiche J, Blanco F, García G, Lopes M C, Ratnavelu K, White R D and Brunger M J 2014 *Chemical Physics Letters* 608 161–166 ISSN 00092614 URL https://doi.org/10.1016/j.cplett.2014.05.087

[75] Mozejko P, Domaracka A, Ptasińska-Denga E and Szmytkowski C 2006 *Chemical Physics Letters* 429 378–381 ISSN 00092614 URL https://doi.org/10.1016/j.cplett.2006.08.077

[76] Bellm S M, Builth-Williams J D, Jones D B, Chaluvadi H, Madison D H, Ning C G, Wang F, Ma X G, Lohmann B and Brunger M J 2012 *The Journal of Chemical Physics* 136 244301 ISSN 0021-9606 URL https://doi.org/10.1063/1.4729466

[77] Duque H V, Chiari L, Jones D B, Pettifer Z, da Silva G B, Limão-Vieira P, Blanco F, García G, White R D, Lopes M C A and Brunger M J 2014 *The Journal of Chemical Physics* 140 214306 ISSN 0021-9606 URL https://doi.org/10.1063/1.4879779

[78] Chiari L, Duque H V, Jones D B, Thorn P A, Pettifer Z, da Silva G B, Limão-Vieira P, Duflot D, Hubin-Franskin M J, Delwiche J, Blanco F, García G, Lopes M C A, Ratnavelu K, White R D and Brunger M J 2014 *The Journal of Chemical Physics* 141 024301 ISSN 0021-9606 URL https://doi.org/10.1063/1.4885856

[79] Brunger M J 2017 *International Reviews in Physical Chemistry* 36 333–376 ISSN 0144-235X URL https://doi.org/10.1080/0144235X.2017.1301030

[80] NGFSRDW database URL http://www.lxcat.net/NGFSRDW

[81] Bishop C M 1994 Mixture density networks URL http://publications.aston.ac.uk/id/eprint/373/

[82] Sohn K, Lee H and Yan X 2015 Learning Structured Output Representation using Deep Conditional Generative Models *Advances in Neural Information Processing Systems* 28 ed Cortes C, Lawrence N D, Lee D D, Sugiyama M and Garnett R (Curran Associates, Inc.) pp 3483–3491 URL http://papers.nips.cc/paper/5775-learning-structured-output-representation-using-deep-conditional-generative-models.pdf
[83] Mirza M and Osindero S 2014 Conditional Generative Adversarial Nets (Preprint 1411.1784) URL https://arxiv.org/abs/1411.1784

[84] Dinh L, Krueger D and Bengio Y 2015 3rd International Conference on Learning Representations, ICLR 2015 - Workshop Track Proceedings (Preprint 1410.8516) URL https://arxiv.org/abs/1410.8516

[85] Dinh L, Sohl-Dickstein J and Bengio S 2016 (Preprint 1605.08803) URL https://arxiv.org/abs/1605.08803

[86] Kingma D P and Dhariwal P 2018 Glow: Generative Flow with Invertible 1x1 Convolutions Advances in Neural Information Processing Systems 31 ed Bengio S, Wallach H, Larochelle H, Grauman K, Cesa-Bianchi N and Garnett R (Curran Associates, Inc.) pp 10215–10224 URL http://papers.nips.cc/paper/8224-glow-generative-flow-with-invertible-1x1-convolutions.pdf

[87] Goodfellow I, Bengio Y and Courville A 2016 Deep Learning (MIT Press) URL http://www.deeplearningbook.org