Machine learning plasma-surface interface for coupling sputtering and gas-phase transport simulations

Florian Krüger, Tobias Gergs, and Jan Trieschmann

1 Institute of Theoretical Electrical Engineering, Ruhr University Bochum, 44801 Bochum, Germany
2 Electrodynamics and Physical Electronics Group, Brandenburg University of Technology Cottbus–Senftenberg, Siemens-Halske-Ring 14, 03046 Cottbus, Germany

(Dated: January 20, 2022)
Abstract

Thin film processing by means of sputter deposition inherently depends on the interaction of energetic particles with a target surface and the subsequent transport of film forming species through the plasma. The length and time scales of the underlying physical phenomena span orders of magnitudes. A theoretical description which bridges all time and length scales is not practically possible. A unified model approach which describes the dynamics of both the solid and the gas-phase, however, remains desired. In fact, advantage can be taken particularly from the well-separated time scales of the fundamental surface and plasma processes by evaluating both independently. Initially, surface properties may be \textit{a priori} calculated from a surface model and stored for a number of representative incident particle energy distribution functions. Subsequently, the surface data may be provided to gas-phase transport simulations via appropriate model interfaces (e.g., analytic expressions or look-up tables) and utilized to define insertion boundary conditions. During run-time evaluation, however, the maintained surface data may prove to be not sufficient (e.g., too narrow input data range). In this case, missing data may be obtained by interpolation (common), extrapolation (inaccurate), or be supplied on-demand by the surface model (computationally inefficient).

In this work, a potential alternative is established based on machine learning techniques using artificial neural networks. As a proof of concept, a multilayer perceptron network is trained and verified with sputtered particle distributions obtained from transport of ions in matter (TRIM) based simulations for Ar projectiles bombarding a Ti-Al composite. It is demonstrated that the trained network is able to predict the sputtered particle distributions for unknown, arbitrarily shaped incident ion energy distributions. It is consequently argued that the trained network may be readily used as a machine learning based model interface (e.g., by quasi-continuously sampling the desired sputtered particle distributions from the network), which is sufficiently accurate also in scenarios which have not been previously trained.
I. INTRODUCTION

Plasma based sputter processing is commonly used for the deposition of thin films (e.g., functional coatings for corrosion protection and wear-resistance applications or optical coatings) [1–6]. Within these processes, two characteristic features of technological plasmas are utilized: (i) The plasma supplies a continuous flux of ions directed toward the bounding surfaces. (ii) In front of these surfaces, the plasma establishes a thin space charge region (sheath) which is accompanied by a substantial local electric field and a drop in the corresponding potential (in particular at electrically driven electrodes). The amounts and the energy distributions of ion species from the plasma can be adjusted with the operating conditions (e.g., pressure and gas composition, voltage and waveform). [1]

Sputtering itself is initiated prevalently by positively charged ions which are accelerated toward the electrode and act as projectiles, hitting the surface material with high kinetic energy. The momentum and kinetic energy of the projectiles is transferred to the surface (usually referred to as target) through a collision cascade. [2, 7–9] The sputtering regime may be classified depending on projectile energy. With energies of a few hundred eV, technological plasmas typically yield binary collision conditions [10]. Resulting from this process, target material atoms are ejected into the plasma with a certain energetic and angular distribution (EAD) [11, 12].

The sputtered particles are transported through the plasma subject to collisions. These can be elastic collisions with neutral gas atoms/molecules or plasma ions causing background gas heating, [13] or inelastic collisions with energetic electrons causing excitation and ionization [1, 3, 14].

When reaching a surface (in particular the substrate), particles may attach and form a thin solid film. This process is reliant on the EAD of impinging particles. Growth rate, crystalline structure and, consequently, mechanical and electrical properties of the formed films are just some of the affected properties [5]. In particular the sputter yield, the ion-induced secondary electron emission coefficient, and the film’s electrical conductivity have an inherent influence on the discharge operation [15–18]. A feedback loop with a strong bidirectional influence between plasma and surface establishes [2, 5, 10, 19, 20].

Computer-based simulations play an integral role in understanding these interactions. Models may be formulated to describe the individual processes on their respective time and length
scales. However, these scales combined span orders of magnitude from atomic to reactor scale and, hence, do not allow for a dynamical, simultaneous theoretical solution. As a remedy for this problem, advantage may be taken of the well-separated time and length scales by evaluating them independently. The models may be subsequently combined, bridging the different scales (e.g., using serial coupling) as suggested in the field of material science \[21, 22\]. This requires a consistent formulation of model interfaces, which is the focus of this work.

In terms of the individual models, technological plasmas are frequently analyzed using fluid models or particle in cell (PIC) simulations (depending on the operating regime), \[23 - 25\], while the transport of sputtered particles is prevalently described via test particle methods (TPMs) \[26 - 28\]. For the specific case of ionized physical vapor deposition (PVD) such as high power impulse magnetron sputtering (HiPIMS), which is operated at low gas pressures below 1 Pa, self-consistent particle in cell/direct simulation Monte Carlo (PIC-DSMC) simulations may be required \[29\]. The mentioned gas-phase models resolve the plasma and particle dynamics on the reactor scale (i.e., process time and reactor dimensions). In contrast, the surface processes on the atomic scale are often studied by means of molecular dynamics (MD), kinetic Monte Carlo (kMC), or transport of ions in matter (TRIM) simulations \[30 - 35\]. The choice of the respective model depends severely on the physical process of concern and the investigated material system. A review of theoretical methods for modeling the plasma-solid interface with a focus on the solid dynamics is provided in \[36\].

The data provided by one and required by another model may be exchanged between surface and gas-phase models via appropriate interfaces. For instance, the two models may be consistently coupled using the EADs of impinging and sputtered particles. This information exchange may be realized in terms of analytic expressions (e.g., Sigmund-Thompson distribution \[7, 9\]) or look-up tables (LUTs). Specifying accurate and computationally feasible interfaces in cases which involve only a few non-reactive species (e.g., argon and metal) is straightforward. In the simplest approach, the interfaces are reduced to integrated surface coefficients neglecting the particles’ energy distributions (e.g., sputter yield, electron emission coefficient). Due to being rough estimates, this is not advised in the context of sputtering simulations even for the simplest systems. Taking into account the particle EADs (not integrated) provides a comparably simple yet more accurate description. However, complex surface and gas-phase systems (e.g., Ar, O\(_2\), N\(_2\) and Ti-Al-O-N) imply a higher order po-
rameter space, complicating the interface models substantially [17, 20]. As all combinations of species involved need to be parameterized, LUT approaches are rendered impractical. In order to overcome the problems arising from the system complexity, we propose an alternative based on machine learning as a numerical plasma-surface interface. Artificial neural networks (ANNs) are trained using a priori determined EADs of incident and sputtered particles. Ultimately, the aim is to predict sputtered particle EADs for arbitrary incident particle EADs, given three important features: i) reasonably abundant data is available from simulations and/or experiments, ii) generalization of the respective data is accomplished, iii) complex physics are preserved, which cannot be achieved by more simplified approaches. As a proof of concept, in this work fulfillment of these requirements is demonstrated using a multilayer perceptron (MLP) network. It is trained with and validated by sputtered particle EADs obtained from TRIDYN simulations [32] for the relative simple case of Ar projectiles bombarding a Ti-Al composite.

The manuscript is structured as follows: In Section II A the simulation scenario and its inherent challenges are introduced. Subsequently, in Sections II C to II E our machine learning approach to the model interface is detailed. Thereafter, applicability of the proposed model is demonstrated and corresponding calculation results are presented in Section III. Finally, the work is summarized and conclusions are drawn in Section IV.

II. PLASMA-SURFACE INTERFACE

A plasma-surface interface for Ar projectiles incident on a Ti-Al composite target is developed in the following. The rather simple material system is chosen as an unsophisticated training and validation demonstrator. It allows for a detailed investigation of the prediction from incident to outgoing particle distributions using an MLP network. If desired, it is straightforward to extend the presented concept to different projectile species or surface composites (provided availability of training data).

A. Physical scales

The conceptual need for a plasma-surface model has been briefly introduced previously. To obtain a more extensive perspective, it is instructive to consider the fundamental physical
Figure 1: Schematic of the physical time and length scales.

scenario and its inherent time and length scales. Essentially three well-separated regimes
have to be distinguished in technological sputter discharges: (i) heavy species gas-phase
dynamics, which are on the order of macroscopic quantities; (ii) intermediate non-equilibrium
electron gas-phase dynamics, which are dictated by the electrons’ small mass; (iii) solid
state surface dynamics, which inherently take place on the atomic level. Note that this
classification ignores the electron dynamics in the solid, which are excluded for simplicity
but may be taken into account if desired. The time and length scales are summarized in
Figure 1 while a detailed description is given below.

(i) Heavy species in the gas-phase are characterized by (close to) equilibrium gas dy-
namics. Correspondingly, their length scale is specified by the collision mean free path
\( \lambda_c \approx \frac{k_B T}{p \sigma} \approx 10^{-2} \) m (where \( p \approx 0.7 \) Pa is the gas pressure, \( k_B \) is the Boltzmann con-
stant, \( T \approx 500 \) K is the gas temperature, and \( \sigma \approx 10^{-18} \) m\(^2\) is the collision cross section).
It is approximately on the order of the vacuum chamber dimensions \( L \approx 10^{-2} \ldots 10^{-1} \) m
within PVD processes. In contrast, the time scale is dictated by the mean collision time,
which is on the order of \( \tau_c \approx \frac{\lambda_c}{v_{th}} \approx 10^{-5} \) s for heavy particle species (neutrals and ions)
close to room temperature (i.e., with thermal velocity \( v_{th} \)).

(ii) Electron time and length scales are substantially different due to their small mass
\( m_e \approx 10^{-5} \) m\( \text{Ar} \). The intrinsic length scale is the Debye screening length \( \lambda_D = \sqrt{\frac{\epsilon_0 T_e}{n_e e^2}} \approx 10^{-4} \) m, with elementary charge \( e \), electron temperature \( T_e \approx 3 \) eV, electron density \( n_e \approx 10^{16} \) m\(^{-3}\), and vacuum permittivity \( \epsilon_0 \). In the low temperature plasma regime,
the electron time scale is given by the inverse electron plasma frequency \( \tau_e \approx \frac{1}{\omega_{pe}^{-1}} = 1/\sqrt{e^2 n_e/(\epsilon_0 m_e)} \approx 10^{-10} \) s.
(iii) Solid state surface dynamics are governed by molecular interactions on the atomic level. In this regime, simulations must be able to resolve molecular oscillations in time and molecular bonds in space. Using the classical harmonic oscillator approximation, the vibrational frequency is given by $v_{\text{mo}} = \sqrt{K/\mu}$, with force constant $K$ and reduced mass $\mu$. With suitable magnitudes for $\mu \approx 10^{-27}$ kg and $K \approx 10^{-3}$ Nm$^{-1}$, a characteristic surface time scale $\tau_{\text{mo}} = v_{\text{mo}}^{-1} \approx 10^{-15}$ s is found. Regarding the length scale, the bond length $l$ in crystals in a face centered cubic (FCC) crystal system can be described by $l = a/\sqrt{2}$, with lattice constant $a$. For stoichiometric Ti-Al which crystallizes in FCC-type structure this yields a length scale on the order of $l \approx 10^{-10}$ m. The same order of magnitude is expected for amorphous Ti-Al.

Comparing the characteristic scales of the gas-phase and surface regime, it is found that $\tau_{\text{mo}} \ll \tau_c \ll \tau_c$ and $l \ll \lambda_D \ll \lambda_c$. The intrinsic time and length scales differ significantly, spanning about 10 orders of magnitude in time and 8 orders of magnitude in length. It is evident that defining time and length scales for a common simulation is not possible. The disparate models need to be considered separately and coupled appropriately.

The demonstration case selected in this work is concerned with the interaction of heavy particles (projectiles) with a solid surface. The link directed from gas-phase to solid surface is forthrightly established in terms of energy distributions of incident species. These are assumed to be available from a model (i.e., they are at this point assumed to be arbitrary, but physical). The goal is to establish a return link from the solid surface to the gas-phase based on an atomic scale surface model. It is infeasible to obtain the EADs of outgoing species for every possible incident species’ energy distribution. Therefore, the surface model is initially solved for a representative set of a priori specified cases. As detailed later, the simulation code TRIDYN is used for this purpose. The obtained training data are then used to train an ANN model interface. The latter is finally intended to predict – at run-time – the EADs of outgoing species for arbitrary incident ones and, therewith, to specify the inflow of particles in the gas-phase model.

As the depicted demonstration case is concerned solely with heavy particle species, electron dynamics are ignored in this consideration. Moreover, also the individual heavy species gas-phase and solid surface models are detailed only as far as it relates to the plasma-surface interface to be established. A comprising and detailed treatment is out of the scope of this work.
B. Data set generation and structure

Sufficiently sound and extensive sets of training data are indispensable for accurate ANN predictions. Fundamental methods like molecular dynamics (MD) may be utilized to provide these data sets at the expense of corresponding computing resources [34, 35, 43, 44]. To demonstrate applicability of the ANN model interface, however, transport of ions in matter (TRIM) based simulations are devised in this work, due to their significantly lower computational cost. General descriptions of the TRIM method and a discussion of detailed aspects can be found elsewhere [10, 30–32, 45]. Briefly, the interaction of projectiles incident on an amorphous solid compound and the dissipation of recoil cascades therein are simulated in binary collision approximation [30]. Although strictly valid only at high projectile energies in the keV range, TRIM based simulations are widely applied also at lower energies [10].

For a sequence of impinging projectiles with a given energy distribution, these simulations provide the EADs of sputtered and reflected particles in terms of a sequence of outgoing particle properties (i.e., species, energy, directional cosines). TRIDYN [32] was used to generate training data of Ar impinging on a Ti-Al composite surface. While it allows for relatively low computation times compared to MD, it is nonetheless not suitable for run-time integration into gas-phase simulations as discussed in Section II E.

Simulation results were obtained using the parameters listed in Table I and II. The statistical quality of the training samples was intentionally limited to $N_{sp} = 10000$ incident projectiles. Using a larger number would improve the statistical quality of the results – and is well within the capabilities of TRIDYN. However, since the presence of statistical noise allows for the investigation of important characteristics of the network, noisy distributions have been purposely chosen. A comparison with higher statistical quality results is presented subsequently.

Each simulation is performed using a unique distribution of incident particles $f_i[l]$ with energy $E_l$ of the $l$-th bin. Because of the strong anisotropic acceleration of plasma ions in the sheath, ions are assumed to impact surface normal (i.e., polar angle $\theta = 0$ and no angular distribution). With regard to the training process, the energy distributions could be of arbitrary shape, but mono-energetic beams, discrete Gaussian and bimodal distributions have been chosen in order to represent physically relevant cases. The respective normalized,
Table I: Global TRIDYN parameters.

| parameter          | symbol | value  |
|--------------------|--------|--------|
| no. of projectiles  | $N_{sp}$ | $10^4$ |
| total fluence      | $\Delta \Phi_{tot}$ | $3 \text{ Å}^{-2}$ |
| pseudoatom weight  | $N_0$   | 3000   |
| angle of radiation | $\theta$ | 0      |
| max. depth         | $x_{max}$ | 1500 Å |
| no. of depth intervals | $\Delta x$ | 500 |

Table II: Element specific TRIDYN parameters. $\varepsilon_{s,t}$ denotes the surface binding energy matrix elements, where index combinations $s, t \in \{\text{Ar, Al, Ti}\}$ iterate over the individual contents of each target component, $Q u_s$ is the respective initial atomic fraction, and $n_s$ the atomic density.

| element | $\varepsilon_{s,\text{Ar}}$ (eV) | $\varepsilon_{s,\text{Al}}$ (eV) | $\varepsilon_{s,\text{Ti}}$ (eV) | $Q u_s$ | $n_s$ (Å$^{-3}$) |
|---------|-------------------------------|-------------------------------|-------------------------------|--------|-----------------|
| Ar      | 0                             | 0                             | 0                             | 0      | 2.49            |
| Al      | 0                             | 3.36                          | 4.12                          | 0.5    | 6.02            |
| Ti      | 0                             | 4.12                          | 4.89                          | 0.5    | 5.58            |

discrete functions are

\[
f_i[l] = \begin{cases} 
1 & \text{if } E = E_l \\
0 & \text{if } E \neq E_l
\end{cases}, \quad (1)
\]

\[
f_i[l] = e^{-\frac{(E-E_l)^2}{\sigma^2}}, \quad (2)
\]

\[
f_i[l] = \begin{cases} 
\sqrt{\frac{-1 + 2(E-E_l)}{\Delta E}} & \text{if } \frac{E_l - E}{\Delta E} \geq 1 \\
0 & \text{else}
\end{cases}. \quad (3)
\]

Therein, $E_l$ denotes the energy of the $l$-th bin, $E$ is the distributions’ median and mean energy, $\sigma^2$ is the variance, and $\Delta E$ is the energy mode separation. TRIDYN evaluates the energy and the directional cosines of individual particles reflected
by or sputtered of the target surface. For a sequence of incident projectiles, the sputtering
events take place on a statistical basis. By creating histograms of the simulation output,
each run gives the EADs \( f_{o,s}[m,n] \) with energy \( E_m \) and surface normal cosine \( c_n = \cos \theta_n \) of
outgoing species \( s \), as a function of the incident particle energy distribution \( f_i[l] \).

To generate an entire data set, a variety of cases have been simulated, varying the mean
energy \( E \) from 0 to 1200 eV in steps of 8 eV for distributions (1)–(3), respectively. For
distribution (2), the variance was set to \( \sigma^2 = 0.3 \) eV, \( \Delta E = 30 \) eV was chosen for distribution
(3). In general, bigger training sets provide trained ANNs with better predictive qualities.
However, in this work the number of sets was intentionally limited to 439 (i.e., 150 per
distribution (1)–(3) with 11 omitted at the outskirts, due to the width of the peaks) to
provide a proof of concept also for computationally limited cases (e.g., MD).

C. Artificial neural network

An ANN is a collection of interconnected processing units called neurons or nodes. ANNs
can be used to approximate the relationship between given input and output vectors \( x_j \) and
\( y_k \), generally without being programmed with any task-specific rules [47–49]. At each node
\( k \), all inputs \( x_j \) are multiplied with a weight \( w_{kj} \). A constant bias \( b_k \) may be introduced
additionally. A sum is taken over all \( d \) input nodes and the result is passed to a nonlinear
activation function \( \phi \), which gives the output \( y_k = \phi \left( b_k + \sum_{j=0}^{d} w_{kj} x_j \right) \) of node \( k \).

In MLP networks nodes are arranged in layers. In a fully connected network every node of
a layer is connected to every node in the neighboring layers. The typical architecture of a
feedforward neural network (like the one presented in this work) consists of a series of layers,
each of which processing the (potentially nonlinear) outputs of the previous layer. The flow
of information is directed from input to output and induces a causal relationship between
them (cf. Figure 2). The layers between input and output layer are commonly referred to
as hidden layers [48, 50].

Linear hidden layers that may be introduced in addition to nonlinear activation layers in-
crease the network complexity. However, linear algebra shows that adjacent linear layers
can be combined to a single linear layer. Hence, nonlinear activation layers allow for the
representation of dependencies, which are not linearly separable [51].

The number of nodes in the input and output layer of the network are defined by the specific
problem to be approached. In the present scenario, they are specified by the bins of the respective histograms. The network input vector $x_j \equiv f_i[l]$ is given by the discrete energy distribution of the incident species with $\dim(x_j) = \dim(E_i) = 151$ input bins (which directly corresponds to the TRIDYN input). The output vector $y_k$ consists of two-dimensional EAD histograms $f_{o,s}[m,n]$ of all considered species $s$ (i.e., reflected Ar and sputtered Al and Ti). Hence, the output vector corresponds to a flattened and segmented one-dimensional data structure. Specifically, the EAD of each species is initially flattened with respect to surface normal cosine $c_n$ and subsequently the collection of EADs is segmented with respect to species $s$. With $\dim(E_m) = 30$, $\dim(c_n) = 20$, and 3 species, this results in $\dim(y_k) = 1800$ output bins. While the input and output dimensions may, in principle, be chosen arbitrarily, the above values have been chosen as a compromise between data reduction and physical accuracy.

The conceptual structure of the resulting ANN with respect to the data structure is shown in Figure 2. The specific choice of network hyperparameters (e.g., number of layers, nodes per layer, activation functions) are discussed in Sections II E and III A. Following the supervised learning concept, the network has to be trained on a previously known data set $N$ to make useful predictions. This set $N$ is composed of an input set $X$ containing reference inputs $x_j' \in X$ and an output set $Y$ containing the corresponding reference outputs $y_k'(x_j') \in Y$. 

![Figure 2: Conceptual schematic of the utilized ANN.](image)
The latter were a priori calculated using TRIDYN.

For training the network, all weights and biases are randomly initialized. These are subsequently optimized using backpropagation by minimizing the loss function \( e^2 \). In this work, the mean square error \( e^2(y_k, y'_k) \) of the predicted output \( y_k(x_j') \) and the ideal output \( y'_k(x_j') \) was used. The average over all samples of a set defines the global error \( \langle e^2(y_k, y'_k) \rangle \). Minimization of this error is called training. One iteration of the training algorithm is called an epoch. Finally, the trained network can be used to make predictions \( y_k(x_j) \) for previously unknown inputs \( x_j \notin X \). [48, 50]

To set up and train the ANN, the Tensorflow framework 1.8.0 and the Keras API 2.2.0 were used [52, 53]. In addition, graphics processing unit acceleration was employed using CUDA 9.1 and two Tesla K40C GPUs [54].

D. Generalization/Overfitting

Generalization is an ANNs ability to handle novel data. It manifests in meaningful predictions (minimal loss), which often expose minimum statistical residual variation (i.e., noise). Minimal losses may be encountered in training of noisy data also due to overfitting of the weights and/or excessive model complexity. That is, a network’s degrees of freedom may allow for an accurate representation of the training data – including its statistical noise. Notably, this adaption of the network to faulty data often implies bad generalization [48]. The small data set used in this work makes the training procedure especially prone to overfitting. As a remedy the data set \( N \) consisting of 439 samples is partitioned into mutually exclusive training, validation, and test sets \( N = \{N_{train}, N_{val}, N_{test}\} \) with \( N_{train} \cap N_{val} \cap N_{test} = \emptyset \). This very common approach allows for the evaluation of the network performance on statistically independent data sets, as follows:

- Training set \( N_{train} \) (80% of samples) is used to fit the model.
- Validation set \( N_{val} \) (10% of samples) is used to provide an unbiased evaluation of the fit on the training set. This is used for tuning the model hyperparameters (cf. Sections II E and III A) or as reference for early stopping (following paragraph).
- Test set \( N_{test} \) (10% of samples) is used to provide an unbiased evaluation of the final model fit on the training set.
Figure 3 shows the evolution of training and validation losses over 50000 epochs for an exemplary training. Evidently, the training loss is continually declining, while the validation error exposes a local minimum after approximately 2600 epochs. This is a direct consequence of overfitting the data. To alleviate this effect, early stopping and regularization have been implemented. Precisely, in early stopping the training is stopped when the validation error stagnates or increases, while the training error keeps declining. In addition, regularization adds a penalty to the loss function, giving preference to well-generalizing network configurations. [48, 50] The specific parameters are detailed in Section III A.

E. Network structure and hyperparameters

The predictive capabilities of an ANN are significantly influenced by its structure and training configuration. The specifying parameters are commonly referred to as hyperparameters. A non-comprising list of the most relevant hyperparameters, their variation in the subsequent study, and the finally used values are listed in Table III. While not all of the ones listed are introduced or detailed in this work, comprehensive information can be found in [48, 50, 55].

Since the input and output vectors only define the outer layers of the network, the remaining network structure and most hyperparameters are not predefined by the data structure. Consequently, an optimal set of hyperparameter has to be found by conducting a parameter study. K-fold cross validation has been used to determine the validation error of the different combinations of the hyperparameters [46, 48, 50] and to determine the optimal set. The design requirements are that the precision of predictions should be significantly im-
Table III: ANN hyperparameters (subdivided into topology and learning).

| hyperparameter               | variation          | suboptimal | final |
|------------------------------|--------------------|------------|-------|
| no. of input nodes           | -                  | 151        | 151   |
| no. of output nodes          | -                  | 1800       | 1800  |
| no. of hidden layers         | [1, 3, 5, 7]       | 5          | 3     |
| no. of nodes per layer       | [500, 1000, 2000]  | 1000       | 1000  |
| activation function          | [sigmoid, tanh]    | tanh       | sigmoid |
| no. of activation layers     | [1, 3, 5]*         | 5          | 3     |
| learning rate                | -                  | 0.1        | 0.1   |
| decay                        | -                  | 0.1        | 0.1   |
| momentum                     | -                  | 0.1        | 0.1   |
| patience                     | [0, 500, 1000]     | 1000       | 500   |
| regularization type          | [L1, L2]           | L1         | L1    |
| regularization factor        | [0, 0.1, 0.5, 1]   | 0          | 0.1   |

* The number of activation layers is always chosen to be maximal, as far as the total number of hidden layers allows.

proved compared to simplified methods (e.g., fitted analytical expressions) mentioned above. Since the trained model is intended for run-time evaluation, the prediction time must be considered as well. The computation time of kinetic simulations varies significantly, ranging from hours to weeks. Therefore, the additional prediction time should be evaluated relative to the total evaluation time per time step. For three-dimensional Monte Carlo simulations of a research scale PVD process evaluated on a parallel cluster with 80 CPU cores [28], the total evaluation time per time step is approximately $\Delta t_{\text{eval}} \approx 5$ s. In this case, approximately $N_{\text{pred}} \approx 500$ particles impinging the target surface have to be evaluated per time step. Hence, the total prediction time per time step should be $\Delta t_{\text{pred}} \ll \Delta t_{\text{eval}}/N_{\text{pred}} \approx 10$ ms. The prediction performance is characterized and shown to fulfill this requirement in a detailed hyperparameter analysis presented in following Section III A.
III. RESULTS

As apparent from the previous discussion, a large variety of network configurations is possible even for the conceptually simple class of MLP networks. Not only the network structure, but also the specific implementation of the nodes (e.g., activation function) and choice of global learning parameters (e.g., decay) offer flexibility in network design. A careful selection of corresponding hyperparameters is required for optimal prediction.

A. Hyperparameter space and validation

It is instructive to rigorously map out the hyperparameter space before considering the optimal case used for final evaluation. This was done with a variation of parameters as listed in Table III. It was limited to the presented value ranges due to the amount of combinations to be trained (576 in total) and the corresponding computational effort. The average validation error $\langle e^2_{val} \rangle$ versus the prediction time $\Delta t_{pred}$ of all converged hyperparameter sets is depicted in Figure 4a. Because predictions with low validation error are generally preferable, the correspondingly marked section (red box) is further investigated. The distinction which defines a “low error” is arbitrary and only for illustration purposes. Yet, the substantial disparity from $\langle e^2_{val} \rangle \approx 5$ to 180 underlines the importance of a proper hyperparameter choice.

Figure 4b shows the performance of trained ANNs segregated by the number of hidden layers. The networks containing only 1 hidden layer (green) clearly tend to perform faster than those containing more layers. This is expected, as the number of floating-point operations increases with the number of hidden layers (for a constant number of nodes per layer). Additionally, the causal relationship between the layers requires sequential computation, thereby limiting acceleration through parallelized computing. However, it is also observed that the minimum validation error of these networks is significantly larger than for networks with more hidden layers. This is due to the circumstance that the model complexity is insufficient to adequately model the higher-order input-output relation [49]. In contrast, those networks with 3 or more hidden layers (and appropriate regularization) show comparably small validation errors, while their prediction time analogously increases with the number of hidden layers.

A notable aspect relates to the number of nonlinear activation and hidden layers. Networks with 5 and 7 hidden layers both have a maximum number of 5 nonlinear activation layers,
Figure 4: (a) Performance of converged hyperparameter sets, (b) influence of number of hidden layers on performance for subset, (c) influence of number of nodes per layer on performance for subset, (d) influence of activation function on performance for subset.

since a further increase compromises convergence during training. These networks differ in the number of linear layers, which for the present study manifests in two aspects: (i) an unchanged validation error and (ii) an extended prediction time. This observation is reasoned by the increased but linear network complexity as discussed in Section II C.

When comparing the number of nodes per layer as shown in Figure 4c, no significant trend in the minimal reachable validation error is apparent (i.e., the model complexity is appropriate
for all considered cases). Only a vague trend is noticeable concerning the prediction time: It may be inferred to increase from 500 (blue) over 1000 (red) to 2000 (black) nodes per layer, but a significant overlap renders a general statement difficult. The increased prediction time is most evident for 2000 nodes per layer, reaching prediction times of up to \( \Delta t_{\text{pred}} \approx 1.8 \text{ ms} \). This is again reasoned by the increase in floating-point operations required with increasing number of nodes per layer (for a constant number of hidden layers).

As evident from Figure 4d, the activation function has an important influence on the network performance as well. In fact, the variability of the sigmoid function, with its property to be non-negative, proves most computationally efficient for a comparable structural network complexity (i.e., same number of hidden layers and number of nodes per layer). This trend is clearly observed for example in the case of 2000 nodes per layer and 5 hidden layers. The column at about 1.4 to 1.5 ms shows a clear segregation between tanh and sigmoid based networks, with the latter resulting in faster prediction times. However, while this can be concluded for the specific case investigated in this work, general statements for other problem types are difficult [49].

An additional aspect related to a network’s ability to generalize after training is the strong link of the achievable validation error to the applied regularization. Specifically, the quality of generalization, and therewith the ability to make predictions on unknown input data, was found to be influenced substantially. Without regularization the trained ANNs were prone to overfitting, while regularization during training effectively suppressed false adaption to noise for the same network. Regularization was, therefore, determined to be crucial, due to the comparably small data set \( N \) and the intentionally entailed statistical noise. However, from the obtained data (not shown) no general trend for the applied regularization type and penalty factor could be inferred. This suggests that the existence of regularization, rather than its type or penalty factor, determine a successful adaptation.

The goal to utilize the trained ANN in the frame of a Monte Carlo gas-phase simulation imposes the previously mentioned requirement on prediction time. It is evident from Figure 4a that essentially all considered networks fulfill this requirement with \( \Delta t_{\text{pred}} \approx 0.8 \text{ to } 1.8 \text{ ms} \lesssim 10 \text{ ms} \). Nevertheless, a minimal prediction time is preferable given comparable validation errors and is correspondingly chosen. Combining the presented insights suggests the set of optimal hyperparameters: 3 hidden layers for adequate model complexity and 1000 nodes per layer as well as sigmoid activation function for optimal prediction time. Final parameters
are listed in Table III. The optimized network yields an average validation error $\langle e_{val}^2 \rangle = 5.83$ and a prediction time of $\Delta t_{\text{pred}} = 0.87$ ms.

B. Demonstration and test

The previously optimized ANN was used to generate predictions for Ar projectiles hitting a Ti-Al target surface using a randomly selected, previously unknown input energy distribution of the test set $N_{\text{test}}$. In the following example, a Gaussian energy distribution peaked with $\sigma^2 = 0.3$ eV at $E = 590$ eV is further investigated.

Figure 5 shows the correspondingly predicted EADs of Ar, Al, and Ti together with reference
Figure 6: EADs of the high statistical quality TRIDYN reference evaluated for $N_{sp} = 10^6$ projectiles. Al (left), Ar (center), and Ti (right).

TRIDYN results as used for training (low statistical quality). The predictions (top, black) show qualitative and quantitative agreement with the reference solution (bottom, red). This is confirmed by the respective mean value $\langle y_k \rangle$ of the output vectors and their variance $\sigma^2$, given by

$$\sigma^2(y_k) = \frac{1}{\dim(y_k)} \sum_{k=0}^{\dim(y_k)-1} (y_k - \langle y_k \rangle)^2.$$ 

In the presented case these values are: $\langle y_k \rangle = 5.71 \approx \langle y'_k \rangle = 5.67$ and $\sigma^2(y_k) = 150.51 \approx \sigma^2(y'_k) = 158.90$. However, while the distributions’ shapes and magnitudes are accurately reproduced (EADs are not normalized), a higher statistical quality of the predicted data as compared to the reference solution is observed. This is an inherent consequence of the ANN’s ability to generalize, whereby random and false noise contributions are effectively ignored.

To quantify the prediction quality more accurately, a reference TRIDYN simulation with high statistical quality ($N_{sp} = 10^6$ instead of $10^4$ incident projectiles) was performed. This high quality data set $y''_k$ provides a low-noise reference for both the prediction $y_k$ and the training reference $y'_k$. It is depicted in Figure 6 which shows the corresponding EADs for Al (left), Ar (center), and Ti (right). The coefficient of determination $R^2$ gives a statistical measure of the quality of a regression model. Using $y''_k$ as reference, it is defined as

$$R^2 = 1 - \frac{SS_{\text{res}}}{SS_{\text{tot}}}.$$ 

19
with the residual sum of squares

\[
SS_{\text{res}} = \sum_{k=0}^{\dim(y_k)-1} (y_k'' - y_k)^2 = \dim(y_k) e^2(y_k, y_k'')
\]  

(6)

and the total sum of squares of \(y_k''\)

\[
SS_{\text{tot}} = \sum_{k=0}^{\dim(y_k'')-1} (y_k'' - \langle y_k'' \rangle)^2.
\]  

(7)

Notably, only for \(y_k'\) as reference \(SS_{\text{res}} \hat{=} \dim(y_k') e^2(y_k, y_k')\) during training/validation and \(SS_{\text{tot}} \hat{=} \dim(y_k') \sigma^2(y_k')\) is proportional to the previously evaluated variance. Also note that the calculated values comprise the contributions of all EADs (flattened output vector \(y_k\)).

In the investigated case, equations (5)–(7) yield \(R^2_{\text{pred}} = 99.85\%\). It stands out that although the network was exclusively trained on high-noise data it shows excellent agreement with the high quality reference data. Interestingly, it even surpasses the limited accuracy case with \(R^2_{\text{ref}} = 96.12\%\), which proves that the effects of statistical noise have been mitigated significantly through generalization.

The importance of well-chosen hyperparameters that yield an ANN with good generalization capabilities becomes evident when comparing the prediction results of the optimal and a suboptimal set as presented in Figure 7. This configuration was chosen arbitrarily from the remaining hyperparameter sets depicted in Figure 4a. Its parameters are also included in Table III. The suboptimal configuration yields a average validation error of 15.5 (approximately threefold increase) and produces predictions with significantly higher statistical

Figure 7: Predicted EADs for suboptimal hyperparameters for Al (left), Ar (center), and Ti (right).
fluctuations. This is verified by the coefficient of determination with respect to $y_k''$, which yields $R^2_{\text{sub}} = 77.43\%$, significantly worse than the optimal set. From the network structure (cf. Table III) it can be inferred that this is clearly due to failed generalization, rather than too simple network complexity.

C. Integrated distributions and sputter yield

The validity of the predicted results can be further checked by considering the dependencies of integrated quantities (i.e., moments of the distribution) and comparing them against reference results. Integrating the EADs over polar angle or energy separates the energy and angular dependence of the distributions, respectively. The resulting energy distributions are shown in Figures 8a to 8c, while the resulting angular distributions are depicted in Figures 8d to 8f.

![Figure 8: Predicted and reference (a) Al energy distribution, (b) Ar energy distribution, (c) Ti energy distribution, (d) Al angular distribution, (e) Ar angular distribution and (f) Ti angular distribution.](image)
to [8]. The integration reduces the statistical scatter with increasing number of particles per bin. The smoothness of the prediction and the reference improve accordingly. For the energy distributions, the relative deviation between prediction and reference is 1.1% on average and 15% at maximum. For the angular distributions it is 3% and 18.4%, respectively. It should be noted that large relative deviations occur exclusively at the low magnitude outskirts of the distributions, which leads to the respective absolute deviation being negligibly small. Furthermore, analogous to the methodology in Section III B, the respective coefficients of determination are $R^2_{\text{pred, E}} = 99.95\%$ and $R^2_{\text{ref, E}} = 99.47\%$ as well as $R^2_{\text{pred, A}} = 99.96\%$ and $R^2_{\text{ref, A}} = 99.28\%$ for energy and angular distributions. This further confirms that the previously observed limited agreement between $y_k'$ and $y_k''$ is due to statistical perturbations, once more indicating the appropriateness of the optimal ANN’s prediction.

Since the respective EADs are not normalized, integrating the EADs over both energy and polar angle and dividing by the total number of incident projectiles gives the sputter yields $Y_{\text{Al}}$ and $Y_{\text{Ti}}$ as well as the reflection coefficient $r_{\text{Ar}}$ for Ar, respectively. Figure 9 shows the corresponding coefficients as a function of incident energy. Evidently, the predicted dependencies again appropriately capture the test data. Small deviations between the predicted and the reference results are observed for some energies. These, however, may be attributed to statistical fluctuations in the training data. A physical interpretation or a specific attribu-

Figure 9: Predicted and reference sputter yield (Al/Ti) and reflection coefficient (Ar) as a function of incident projectile energy using the mono-energetic distribution [1].
tion to features of the trained ANN or to aspects of the machine learning procedure cannot be inferred. A peculiar aspect of physical reliability relates to similar sputter yields for Al and Ti: The fluxes of both metal species emitted from the target surface under continuous ion bombardment are necessarily identical on average. This is a direct consequence of mass conservation in steady state. It is instructive to observe from the Figure 9 that this criterion is similarly fulfilled for the predicted and the reference yields. The mean and maximum deviation for the predicted results are 3.9% and 13%, respectively. These results underline the physical relevance and reliability of the proposed procedure not only for a single interaction case, but over the complete range of incident energies previously trained.

IV. CONCLUSION

This work demonstrates the successful supervised learning of statistically disturbed data sets from the solid surface model TRIDYN [32] using a multilayer perceptron ANN. For a set of input energy distributions of impinging projectile particles, the energy and angular distributions of reflected and sputtered particles are shown to be accurately predicted. The trained ANN is verified to reliably generalize, minimizing the predictions’ dependence on the noise contained within the training data. Specifically, with a coefficient of determination of $R^2_{\text{pred}} = 99.85\%$ the ANN corresponds more accurately to high quality reference data than the noisy training set with $R^2_{\text{ref}} = 96.12\%$. This is in particular valuable in view of the limited size of the training set. This result was enabled by investigating the influence of the governing hyperparameters, which encompasses a variation of the chosen data structure as well as an investigation of the corresponding prediction times. The latter are on the order of $\Delta t_{\text{pred}} \approx 1\, \text{ms}$ and, therewith, much smaller than the approximate gas-phase simulation evaluation time per time step of 10 ms.

From the presented results it can be inferred that our approach is a promising alternative to current methods for interfacing multi-scale models. However, apart from the determined conceptual validity, two farther-reaching aspects seem important to be addressed in the future: Firstly, the predictions’ physical validity needs to be rigorously verified (e.g., distributions non-negative, flux balance fulfilled). Correspondingly, the question should be addressed whether specified aspects can be mathematically proven or inferred from the design of the network. Secondly, the ANN needs to be implemented within a comprising simulation
model. In this respect, initially the ANN should be embedded as a simple plasma-surface interface model. The complexity should subsequently be increased by considering more complex chemical and molecular interactions (i.e., including reactive species and corresponding processes). This logically implies the application of more sophisticated molecular dynamics output data for training.

ACKNOWLEDGEMENT

The authors sincerely thank Professor Dr.-Ing. Thomas Mussenbrock from Brandenburg University of Technology Cottbus–Senftenberg for his advice and support. The authors thank Professor Dr. Wolfhard Möller from Institute of Ion Beam Physics and Materials Research, Helmholtz-Zentrum Dresden-Rossendorf (HZDR) for permission to use the TRI-DYN simulation software. Financial support provided by the German Research Foundation (DFG) in the frame of the collaborative research centre TRR 87 (SFB-TR 87) is gratefully acknowledged.

[1] B. N. Chapman, Glow discharge processes (John Wiley & Sons, Hoboken, USA, 1980).
[2] S. M. Rossnagel, J. J. Cuomo, and W. D. Westwood, Handbook of plasma processing technology: fundamentals, etching, deposition, and surface interactions (Noyes Publications, Park Ridge, USA, 1990).
[3] M. A. Lieberman and A. J. Lichtenberg, Principles of Plasma Discharges and Materials Processing, 2nd ed. (Wiley, Hoboken, USA, 2005).
[4] T. Makabe and Z. L. Petrovic, Plasma Electronics : Applications in Microelectronic Device Fabrication, 1st ed. (CRC Press, 2006).
[5] P. M. Martin, ed., Handbook of Deposition Technologies for Films and Coatings, 3rd ed. (William Andrew Publishing, Boston, USA, 2010).
[6] K. Sarakinos, J. Alami, and S. Konstantinidis, Surface and Coatings Technology 204, 1661 (2010).
[7] M. W. Thompson, Philosophical Magazine 18, 377 (1968).
[8] P. Sigmund, Physical Review 184, 383 (1969).
[9] P. Sigmund, Physical Review 187, 768 (1969).
[10] R. Behrisch and W. Eckstein, Sputtering by Particle Bombardment, Topics in Applied Physics, Vol. 110 (Springer, Berlin, Germany, 2007).
[11] G. Betz and K. Wien, International Journal of Mass Spectrometry and Ion Processes 140, 1 (1994).
[12] M. Stepanova and S. K. Dew, Journal of Vacuum Science and Technology A 19, 2805 (2001).
[13] D. W. Hoffman, Journal of Vacuum Science and Technology A 3, 561 (1985).
[14] Y. P. Raizer, Gas Discharge Physics (Springer, Berlin, Germany, 1991).
[15] D. Depla, G. Buyle, J. Haemers, and R. De Gryse, Surface and Coatings Technology 200, 4329 (2006).
[16] D. Depla, J. Haemers, and R. De Gryse, Thin Solid Films 515, 468 (2006).
[17] D. Depla, S. Mahieu, R. Hull, R. M. Osgood, J. Parisi, and H. Warlimont, eds., Reactive Sputter Deposition, Springer Series in Materials Science, Vol. 109 (Springer, Berlin, Germany, 2008).
[18] D. Depla, X. Y. Li, S. Mahieu, and R. D. Gryse, Journal of Physics D: Applied Physics 41, 202003 (2008).
[19] S. Berg, H.-O. Blom, T. Larsson, and C. Nender, Journal of Vacuum Science & Technology A 5, 202 (1987).
[20] S. Berg and T. Nyberg, Thin Solid Films 476, 215 (2005).
[21] J. Q. Broughton, F. F. Abraham, N. Bernstein, and E. Kaxiras, Physical Review B 60, 2391 (1999).
[22] E. Weinan, B. Engquist, X. Li, W. Ren, and E. Vanden-Eijnden, Communications in Computational Physics 2, 367 (2007).
[23] C. K. Birdsall and A. B. Langdon, Plasma Physics via Computer Simulations (IOP Publishing, Bristol, UK, 1991).
[24] G. Colonna and A. D’Angola, eds., Plasma Modeling, 1st ed. (IOP Publishing, Bristol, UK, 2016).
[25] J. van Dijk, G. M. W. Kroesen, and A. Bogaerts, Journal of Physics D: Applied Physics 42, 190301 (2009).
[26] R. E. Somekh, Journal of Vacuum Science and Technology A 2, 1285 (1984).
[27] G. M. Turner, I. S. Falconer, B. W. James, and D. R. McKenzie, Journal of Applied Physics 65, 3671 (1989).
[28] J. Trieschmann and T. Mussenbrock, Journal of Applied Physics 118, 033302 (2015).
[29] V. V. Serikov, S. Kawamoto, and K. Nanbu, Plasma Science, IEEE Transactions on 27, 1389 (1999).
[30] J. P. Biersack and L. G. Haggmark, Nuclear Instruments and Methods 174, 257 (1980).
[31] W. Eckstein and J. Biersack, Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms 2, 550 (1984).
[32] W. Möller and W. Eckstein, Nuclear Instruments and Methods in Physics Research Section B 2, 814 (1984).
[33] A. F. Voter, in Radiation effects in solids, NATO science series. II, Mathematics, physics and chemistry No. v. 235 (Springer, Dordrecht, The Netherlands, 2007).
[34] D. B. Graves and P. Brault, Journal of Physics D: Applied Physics 42, 194011 (2009).
[35] E. C. Neyts and P. Brault, Plasma Processes and Polymers 14, 1600145 (2017).
[36] M. Bonitz, A. Filinov, J. W. Abraham, K. Balzer, H. Kählert, E. Pehlke, F. X. Bronold, M. Pamperin, M. M. Becker, D. Loffhagen, and H. Fehske, arXiv:1809.02473 [physics] (2018), arXiv: 1809.02473.
[37] G. A. Bird, Molecular Gas Dynamics and the Direct Simulation of Gas Flows (Oxford University Press, New York, USA, 1994).
[38] E. H. Holt and R. E. Haskell, Foundations of plasma dynamics (The Macmillan Company, New York, USA, 1965).
[39] S. Wilczek, J. Trieschmann, D. Eremin, R. P. Brinkmann, J. Schulze, E. Schuengel, A. Derzsi, I. Korolov, P. Hartmann, Z. Donkó, and T. Mussenbrock, Physics of Plasmas 23, 063514 (2016).
[40] P. Larkin, Infrared and Raman Spectroscopy, 1st ed. (Elsevier, Oxford, UK, 2011).
[41] W. D. J. Callister and D. G. Rethwisch, Materials Science and Engineering: An Introduction, 9th ed. (Wiley, Hoboken, USA, 2013).
[42] J. L. Murray, Metallurgical Transactions A 19, 243 (1988).
[43] H. M. Urbashek, Nuclear Instruments and Methods in Physics Research Section B: Beam Interactions with Materials and Atoms Nanometric Phenomena Induced by Laser, Ion and Cluster Beams, 122, 427 (1997).
[44] P. Brault and E. C. Neyts, *Catalysis Today Plasmas for enhanced catalytic processes (ISPCEM 2014)*, 256, 3 (2015).

[45] H. Hofsäss, K. Zhang, and A. Mutzke, *Applied Surface Science* 310, 134 (2014).

[46] D. Zwillinger, *Standard Mathematical Tables and Formulae*, 31st ed. (CRC Press, 2002).

[47] H. K. D. H. Bhadeshia, *ISIJ International* 39, 966 (1999).

[48] C. M. Bishop, *Neural Networks for Pattern Recognition* (Oxford University Press, Oxford, UK, 1996).

[49] S. Haykin, *Neural Networks and Learning Machines: A Comprehensive Foundation*, 3rd ed. (Prentice Hall International, New York, USA, 2008).

[50] Y.-H. Pao, *Adaptive Pattern Recognition and Neural Networks*, 1st ed. (Addison-Wesley, Reading, USA, 1989).

[51] G. Cybenko, *Mathematics of Control, Signals and Systems* 2, 303 (1989).

[52] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, J. Levenberg, R. Monga, S. Moore, D. G. Murray, B. Steiner, P. Tucker, V. Vasudevan, P. Warden, M. Wicke, Y. Yu, and X. Zheng, “TensorFlow: An Open Source Machine Learning Framework for Everyone,” (2016), https://tensorflow.org/.

[53] F. Chollet and others, “Keras: The Python Deep Learning library,” (2015), https://keras.io/.

[54] J. Nickolls, I. Buck, M. Garland, and K. Skadron, *ACM Queue* 6, 40 (2008).

[55] R. Reed and R. J. Marks, II, *Neural Smithing: Supervised Learning in Feedforward Artificial Neural Networks* (MIT Press, Cambridge, USA, 1999).

[56] N. R. Draper and H. Smith, *Applied Regression Analysis*, 3rd ed. (John Wiley & Sons, New York, USA, 1998).