Insights into one-body density matrices using deep learning

Jack Wetherell[a,b‡], Andrea Costamagna[c,d,e,b], Matteo Gatti[a,b,c], and Lucia Reining[a,b]

The one-body reduced density matrix (1-RDM) of a many-body system at zero temperature gives direct access to many observables, such as the charge density, kinetic energy and occupation numbers. It would be desirable to express it as a simple functional of the density or of other local observables, but to date satisfactory approximations have not yet been found. Deep learning is the state-of the art approach to perform high dimensional regressions and classification tasks, and is becoming widely used in the condensed matter community to develop increasingly accurate density functionals. Autoencoders are deep learning models that perform efficient dimensionality reduction, allowing the distillation of data to its fundamental features needed to represent it. By training autoencoders on a large data-set of 1-RDMs from exactly solvable real-space model systems, and performing principal component analysis, the machine learns to what extent the data can be compressed and hence how it is constrained. We gain insight into these machine learned constraints and employ them to inform approximations to the 1-RDM as a functional of the charge density. We exploit known physical properties of the 1-RDM in the simplest possible cases to perform feature engineering, where we inform the structure of the models from known mathematical relations, allowing us to integrate existing understanding into the machine learning methods. By comparing various deep learning approaches we gain insight into what physical features of the density matrix are most amenable to machine learning, utilising both known and learned characteristics.

1 Background and Objectives

The development of modern technology is driven by our understanding of the behavior of systems at the quantum mechanical level. Theory and numerical calculations play an important role in the development of this understanding. However, real materials consist of interacting particles, which gives rise to the vastly unfavourable computational and memory scaling required to solve the underlying equations. If we could solve the many-body Schrödinger equation for the ground-state wavefunction and store such an object, observables could be calculated as expectations values, but this is not possible for systems of interest. The Hohenberg-Kohn theorems within density functional theory (DFT) tell us that we can instead describe any observable in terms of the much more manageable electron density[1], but the form of almost all such functionals is unknown.

The one-body reduced density matrix (1-RDM) can be thought of as an intermediate quantity between these two extremes. As with the density, it avoids the problem of having to store a function of all the spin and spacial coordinates of the system. For a N-electron spin-resolved system at zero temperature the 1-RDM is given by

$$\gamma(r,r') = N \int \Psi(r,r_2,r_3,\dots)\Psi^*(r',r_2,r_3,\dots)dr_2dr_3\dots. \quad (1)$$

Its diagonal is the charge density $n(r) = \gamma(r,r)$. The expectation value of any local or non-local one-body operator in terms of the density matrix is

$$O[\gamma] = \int O(r,r')\gamma(r,r')drdr'. \quad (2)$$

In particular, the kinetic energy K of the many-body system reads

$$K[\gamma] = -\frac{\hbar^2}{2m}\int \nabla^2\, \gamma(r,r')\big|_{r=r'} dr'. \quad (3)$$

While reduced density matrix functional theory (RDMFT)[2–12] performs a constrained minimisation of the total energy E over the 1-RDM, it would be possible to perform the minimisation over the density itself if we could express the 1-RDM as functional of the density. This would allow for direct minimisation of the energy within DFT without the need for a Kohn-Sham (KS) auxiliary system[13], which introduces orbitals[14]. *Therefore it would be highly desirable to find the functional $\gamma[n]$, as this would allow these key quantities to be themselves expressed as functionals of the charge density.* The search for such a functional does not have to be completely blind. In particular, the density matrix is an object that is subject to many constraints[2]. Not all functions $f(r,r')$ are valid density matrices, in the sense that they can be computed from the ground state wavefunction of a Hamiltonian with a local and static potential. The knowledge of constraints is crucial when building functionals, as it considerably reduces the domain of legitimate functionals one must search over[15].

In the data science community, there is an exponential growth of modern machine learning methods, that each day are being applied to successfully solve increasingly difficult problems with astonishing accuracy. Such problems were previously thought to be impossible to solve numerically, in particular in the field of image processing. As the 1-RDM stored on a numerical grid is essentially an image, with a dominant spacial structure, the question naturally arises: *Can these methods be used to gain new insights into the 1-RDM and help us find the functional we desire?*

Machine learning is becoming increasingly utilised in the field

[a] Laboratoire des Solides Irradiés, École Polytechnique, CNRS, CEA/DRF/IRAMIS, Institut Polytechnique de Paris, F-91128 Palaiseau, France.
[b] European Theoretical Spectroscopy Facility (ETSF).
[c] Synchrotron SOLEIL, L'Orme des Merisiers, Saint-Aubin, BP 48, F-91192 Gif-sur-Yvette, France.
[d] Politecnico di Torino, 10129 Torino, Italy.
[e] Université Paris-Saclay, 91405, Orsay, France.
‡ Personal email: jack.wetherell@polytechnique.edu; Personal webpage: https://jw1294.github.io/

of condensed matter physics[16–34]. In particular, machine learning has been shown to yield impressive results for the computation of the exchange-correlation potential within DFT. In a recent work[35], small exactly solvable molecules are used to train a machine learning model for the exchange-correlation potential. The authors demonstrate that this can then be used to predict the properties of more complex molecules. This exploits the holographic density principal of molecules[36], which suggests that the behaviour at a given part of a large molecule (for example a bond) is also present in a small molecule. Machine learning is also widely utilised within condensed matter physics, and has been shown to be able to perform the Hohenberg-Kohn mapping from the external potential to the charge density directly using kernel ridge regression[37].

We wish to augment machine learning models with our current approaches, such that only the smallest possible parts, which are the most difficult to approximate, have to be learned. This raises three fundamental questions: *Can machine learning give insights to the 1-RDM, in particular constraints? Can machine learning algorithms optimised for image processing learn the functional $\gamma[n]$, and can we integrate this with pre-exiting physically-based models so we need only learn the neglected phenomena, and if so which part is the most amenable to machine learning?*

2 Machine Learning Methods

Deep learning is a powerful method within machine learning that is used to perform very high dimensional and extremely non-linear fitting using a large data-set on powerful hardware. We now introduce the deep learning methods that we utilise to answer our proposed questions, and how in particular they relate to physical problems faced in quantum chemistry.

2.1 Deep Neural Networks

Deep neural networks are numerical models that are trained to recognise patterns and relationships between data. For our purposes we will use them to perform generalised regression. If we have a labeled data-set of known inputs $\{x\}$ and known outputs $\{y\}$ a deep neural network can learn any non-linear map $f : x \rightarrow y$, that can make predictions on novel x values. This is learned through the process of gradient descent, where the parameters of the network are adjusted to minimise the error of predictions made on known data. With proper structuring the inputs and outputs can be of any form: images, functions, numerical values etc, and the model with enough complexity can learn any arbitrarily non-linear mapping.

The simplest type of neural network we will utilise is the multilayer perceptron (MLP)[38], illustrated in figure 1. A MLP is composed of layers of perceptions, each holding one value, computed as a weighted linear sum of its inputs $\{x\}$ passed through some non-linear activation function σ: $\sigma\left(\sum_i w_i x_i + b\right)$, where $\{w\}$ are the weights of the layer and b is the perception's bias. It is the many layers of these perceptrons, each a non-linear combination of all the perceptrons in the previous layer, that allows the network to learn highly intricate relationships. During training the weights and biases are adjusted through gradient descent, after being randomly initialised, until the error with respect to the known data is minimised.

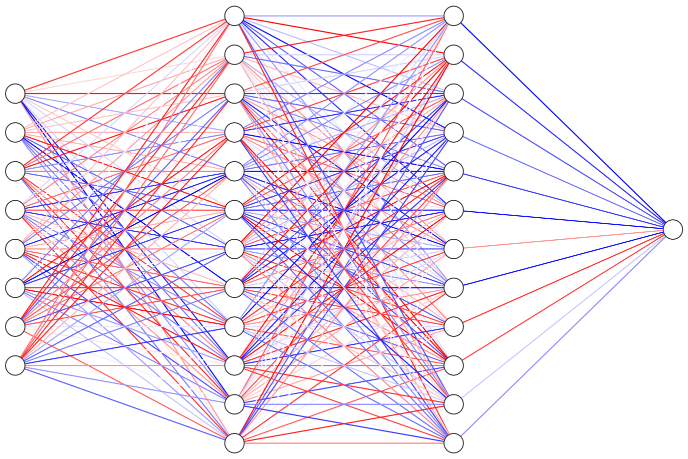

Fig. 1 An illustration of a multilayer perceptron (MLP)[39]. The circles represent the layers of perceptrons, that are fully connected between layers. The red and blue lines represent the values of the weights of each layer (one set of $\{w_i\}$ for each perceptron), where blue indicates a positive weight, and red a negative weight. Each perceptron also has a bias b that is not shown. It is these weights and biases that are adjusted during the training via gradient descent. The middle two layers are termed *hidden layers* as they are not directly connected to the inputs or outputs. With a sufficient number of perceptrons in the hidden layers, this network can in principal learn any arbitrarily complex mapping from the 8 input values, to the 1 output value $y_1 = f(x_1, x_2, \ldots, x_8)$.

2.2 Autoencoders

Autoencoders (AE) are deep neural networks that are trained to perform efficient generalised data compression[40]. They consist of a neural network that is trained to reproduce exactly its own input data as output data $f_{\mathrm{AE}} : x \rightarrow x$. A single hidden layer acts as a bottleneck, containing fewer perceptions than in the input and output layers, compressing the data to a latent space. The two parts of the autoencoder can be separated into the encoder $f_{\mathrm{E}} : x \rightarrow x'$ and decoder $f_{\mathrm{D}} : x' \rightarrow x$, where x' has a smaller dimensionality then x.

The amount to which data can be compressed depends on its features. For example, images of only faces can be compressed significantly more than general images, as they are more heavily constrained. Therefore the use of AEs can be thought of as *domain-specific* data compression, as the network learns the underlying features of a specific data-set (domain), and so learns to exploit these in order to achieve a greater degree of compression. In general, there is a deep connection between compression and constraints: the more data is constrained, the more it can be compressed losslessly. We propose training an autoencoder on the 1-RDM in order to inform: to what extent it can be compressed, the nature of the compression, and how one can extract these constraints. In particular, we would hope the AE would learn the 1-RDM $\gamma(r, r')$ can be compressed to a latent space of the dimensionality of its diagonal $n(r)$.

In principle we could use a MLP with a bottleneck layer as our deep autoencoder. However, this would be onerously expensive and inefficient, for the same reasons MLPs are rarely used for image processing: they do not exploit the spacial structure, treating each pixel of data totally independently from the rest. Instead, density matrices do have strong spacial structure: for example, for the most common external potentials they are continuous and smooth. Therefore, we utilise convolutional autoencoders (CAE) to learn the constraints of the 1-RDM. CAEs convolve several kernels over the two dimensional input image using element wise multiplication[41,42]. These values are then passed to some non-linear activation function σ. This can be thought of as 'scanning' over the image with a filter representing a particular feature. The resultant values describe the similarity between a region of the image and the feature of interest. It is this that exploits the spacial structure of the image. This process is repeated until the spacial information of the image has been converted from real space to a 1-dimensional feature space. This is our bottleneck layer. This process is then reversed using transpose convolutions layers (that perform the inverse operation) until the image is recovered. This is illustrated in figure 2. *It is the kernels of this network that are adjusted throughout training*, until the input image can be reconstructed as the output to a required tolerance over the data-set. This then allows us to learn arbitrarily non-linear constraints of the 1-RDM, and find a latent feature space where the 1-RDM as a functional of the density may be simpler. We employ CAEs to learn constraints and develop approximate functionals for the 1-RDM on a large data set.

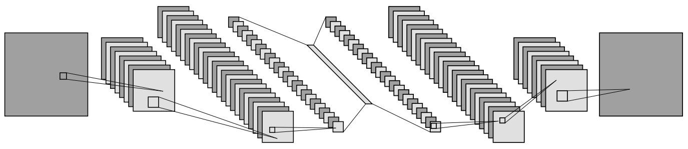
Fig. 2 Illustration of a convolutional autoencoder (CAE)[39]. The 2 dimensional input image is convolved with kernels reducing the spacial dimensions and increasing the number of features until the data is totally reduced to a feature latent space with far fewer degrees of freedom that the original image. This compression can be achieved as the image contains some intrinsic structure, as opposed to totally random pixel values. This process is then inverted with transpose convolutional layers until the original image dimension is recovered. This network is trained by adjusting the kernels through gradient descent until the output image reproduces the input image to a given tolerance over the training data-set.

2.3 Principal Component Analysis

The simplest autoencoder we can imagine is dimensional reduction via principal component analysis (PCA)[43]. PCA consists of computing the linear transform to an orthogonal space that is designed such that each component is ordered by its variance[44]. This is illustrated in figure 3. If the variance of a given component is zero, that component can be neglected such that the original data is recovered exactly upon the inverse transformation. Good approximations are obtained when components are neglected whose variance is small. It is important to note that the PCA is a strictly linear transformation, and so can only determine linear constraints in data (in contrast to CAEs).

We will now introduce how PCA is performed on a data-set

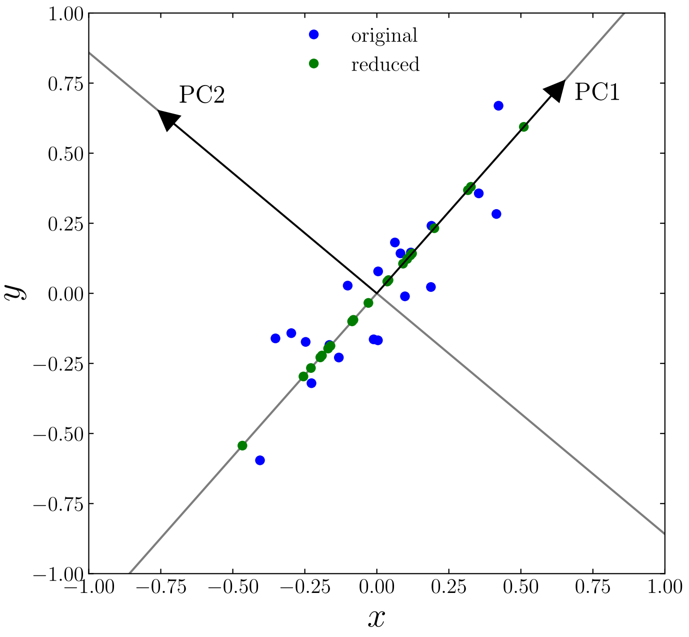

Fig. 3 A simple illustration of principal component analysis (PCA). The blue dots show a set of paired data points $\{(x_i, y_i)\}$. The PCA applied to this data yields an orthonormal basis shown by the two black arrows. They are ordered by their variance, with principal component 1 (PC1) being the component of most variation. If we then use this to perform a lossy compression we simply discard the principal component 2 (PC2) and perform the inverse transform, yielding the reduced data points shown in green.

consisting of T NxN matrices. We begin by considering element t of our data-set:

$$\gamma^{(t)} = \begin{bmatrix} \gamma^{(t)}_{N1} & \cdots & \gamma^{(t)}_{NN} \\ \vdots & \ddots & \vdots \\ \gamma^{(t)}_{11} & \cdots & \gamma^{(t)}_{1N} \end{bmatrix} \in \mathbb{R}^{N\times N}. \tag{4}$$

In order to represent this matrix, it is possible to define a N^2-dimensional basis, each component of which points to a different entry of the matrix

$$\mathscr{B}_e = \{|e_r\rangle\}_{r=1}^{N^2} = \{|e_{r[i,j]}\rangle\}_{i,j=1}^{N}$$

This basis leads to the 'flattened' version of the original matrix, represented as the following vector:

$$|\gamma^{(t)}\rangle = \sum_{i,j=1}^{N} \langle e_{r[ij]}|\gamma^{(t)}\rangle \cdot |e_{r[ij]}\rangle = \sum_{i,j=1}^{N} \gamma^{(t)}_{ij} |e_{r[ij]}\rangle \tag{5}$$

or, also

$$\underline{\gamma}^{(t)} = \begin{bmatrix} \gamma^{(t)}_{11} & \gamma^{(t)}_{12} & \cdots & \gamma^{(t)}_{NN} \end{bmatrix} \in \mathbb{R}^{N^2}. \tag{6}$$

Out data-set of T such vectors is then denoted:

$$\Gamma = \begin{bmatrix} \underline{\gamma}^{(1)} \\ \vdots \\ \underline{\gamma}^{(T)} \end{bmatrix} \tag{7}$$

The disposal of this data-set allows us to define a new basis with which it is possible to describe the γ-vectors. The PCA is then considered a linear numerical method to determine the following two sets of quantities:

- $|\gamma_0\rangle$: The mean matrix. The knowledge of this allows writing each matrix under analysis in terms of its variations from the mean $|\gamma\rangle = |\gamma_0\rangle + |\tilde{\gamma}\rangle$. This is termed the mean-adjusted matrix. The mean matrix components in the previously defined basis read
$$(\gamma_0)_{ij} = \frac{1}{T}\sum_{t=1}^{T}\gamma_{ij}^{(t)}.$$
- $\mathscr{B}_{pca} = \{|p_i\rangle\}_{i=1}^{N^2}$: A new basis, corresponding to the principal components (or principal directions). They are the normalized eigenvectors of the covariance matrix
$$C = \frac{1}{T-1}\Gamma^\dagger\Gamma = \sum_{r,r'=1}^{N^2} c_{r,r'}|e_r\rangle\langle e_{r'}|$$
$$c_{r,r'} = \frac{1}{T-1}\sum_{t=1}^{T}(\gamma_r^{(t)} - (\gamma_0)_r)(\gamma_{r'}^{(t)} - (\gamma_0)_{r'}).$$
The eigenvalues of such a matrix are termed the variances $\{\sigma_i^2\}_{i=1}^{N^2}$. The principal components are the directions along which, in the data-points, there are the most informative variations with respect to the average matrix (see figure 3). They are sorted by importance depending on the value of the associated eigenvalue.

The knowledge of the data-set in this form implies that, by solving the eigenequation, one can determine the coefficients $\langle e_r|p_i\rangle$ in the expansion
$$|p_i\rangle = \sum_{r=1}^{N^2}\langle e_r|p_i\rangle|e_r\rangle. \tag{8}$$
Each 1-RDM can be expressed in this new basis in an expansion called *principal components decomposition*:
$$|\gamma\rangle = |\gamma_0\rangle + \sum_{i=1}^{N^2}\langle p_i|\tilde{\gamma}\rangle\cdot|p_i\rangle. \tag{9}$$
The main property of PCA is that the existence of linear constraints in between the features of the object under analysis (entries of the matrix) leads to vanishing eigenvalues, associated to non-informative principal components. This allows the compression of the information by using a number $\nu < N^2$ of components. For example, if the matrix is symmetric $\gamma_{i,j}^{(t)} = \gamma_{j,i}^{(t)}\forall t$, the data can be compressed to $\nu \leq \frac{N(N+1)}{2}$, and the matrix can be fully represented using a reduced number of principal components
$$|\gamma\rangle = |\gamma_0\rangle + \sum_{i=1}^{\nu}\langle p_i|\tilde{\gamma}\rangle\cdot|p_i\rangle. \tag{10}$$

3 The data-set

In order to investigate to what extent deep learning can answer our questions of interest, we construct a large training and testing data-set of external potentials, charge densities and 1-RDMs. To generate the data-set we use the iDEA code[45,46]. This exactly solves the many-body Schrödinger equation for finite systems of up to four electrons interacting via a softened Coulomb interaction on a one-dimensional real-space grid given any arbitrary local external potential. In addition, it provides implementations of many widely-used approximate methods[47]. After computing the exact many-body wavefunction, any required observables can be obtained via expectation values directly. The model systems solved by the iDEA code have in the past been used to develop improved approximations to DFT[48,49], many-body perturbation theory[50], as well as investigating the nature of exact potentials[51], where the model systems have been shown to well describe crucial features as that of real three-dimensional molecules[52].

The training data is composed of a large family of randomly generated two-electron systems in their spin-resolved ground-state. For each system we: construct a randomly generated smooth potential $V(x)$ for which we determine the exact ground-state many-body wavefunction. From this we compute the charge density $n(x)$ and 1-RDM $\gamma(x,x')$. We also, for the same potential, compute the charge density and 1-RDM using purely non-interacting electrons (NON) and unrestricted Hartree-Fock (UHF). As these are finite systems in the ground state, the 1-RDMs are real-valued functions. We define the external potential as a sum of randomly distributed Fourier components within a large confining potential[53]:
$$V(x) = Dx^{10} + T\sum_{n=1}^{N}\left(a_n\cos\left(\frac{n\pi x}{L}\right) + b_n\sin\left(\frac{n\pi x}{L}\right)\right), \tag{11}$$
where $L = 15$(a.u.)* is the width of system, $N = 3$ is the number of Fourier terms, $D = 10^{-11}$ is the damping factor of confining term and $T = 0.1$ is damping factor of Fourier terms. a_n and b_n are generated randomly with a uniform distribution from $-\frac{2L}{3}$ to $\frac{2L}{3}$. We generated a data-set of 50,000 systems for 2, 3, 4, 6, and 62 grid points.

Figure 4 shows the first five elements of the 50,000 test systems in the data-set with 62 grid points. The systems display a wide range of potentials, densities and 1-RDMs, exhibiting a wide range of localisation and correlation. We use this data-set to train and test our deep learning models.

4 Learning Constraints

4.1 Constraints of the charge density

We will now investigate to what extent the machine can learn fundamental principles. Not all functions of two variables $f(x,x')$ are 1-RDMs due to its constraints, and so we now investigate to what extent the machine can learn such constraints. To begin we will focus on the simplest example possible. We will use the data-set of only 2-points, as it makes it possible to visualise and quantify all the relationships between the data. We will first see if we can use PCA to learn the known constrains of the *charge density*. As the space only contains 2 points, the density is represented by 2 values $n_1 = n(x_1)$, $n_2 = n(x_2)$. The following are two known constraints:

1. $n_1 + n_2 = \frac{N}{\Delta x}$

* Hartree atomic units: $m_e = \hbar = e = 4\pi\varepsilon_0 = 1$.

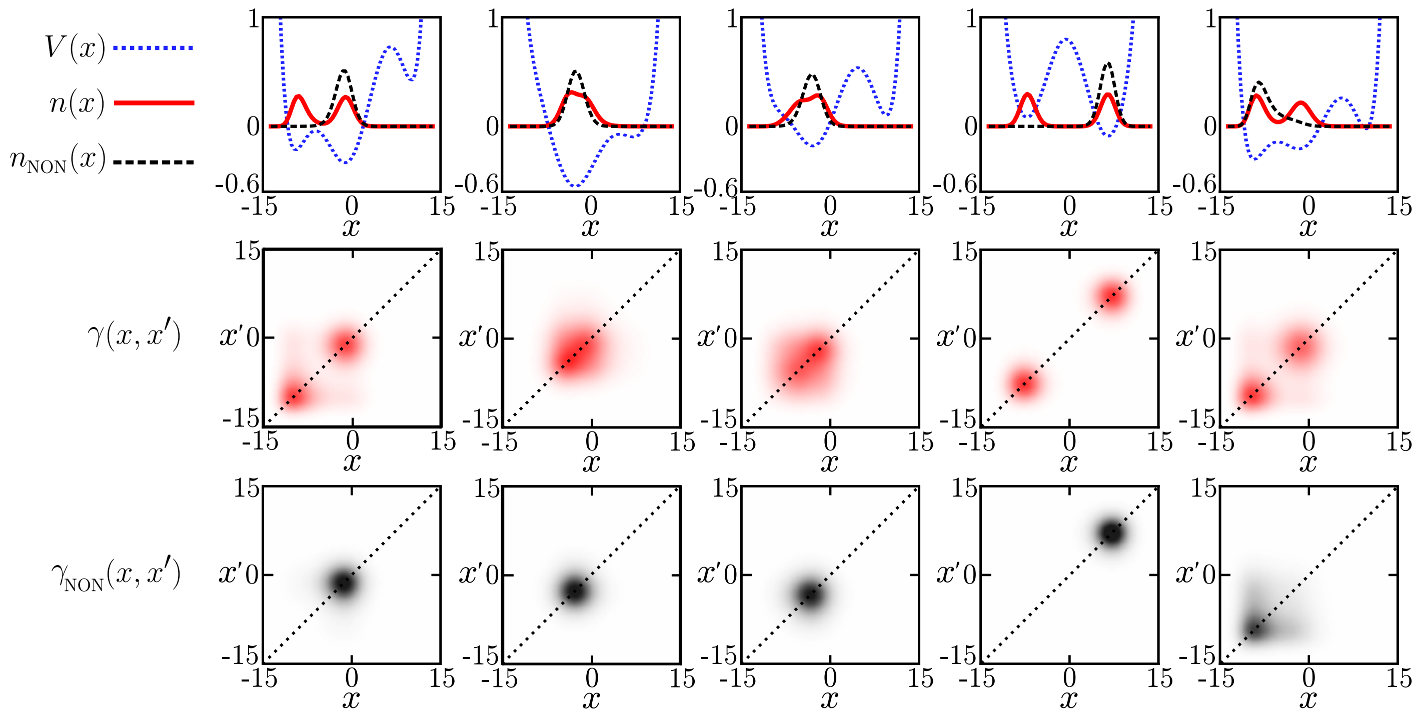

Fig. 4 The first 5 elements of the 50,000 systems in the data-set. Row 1 shows the randomly generated (as defined by equation 11) external potential $V(x)$, the interacting charge density $n(x)$, and purely non-interacting charge density $n_{\mathrm{NON}}(x)$. These potentials give rise to a wide range of density shapes, locations and overlaps. Row 2 shows each of the systems exact 1-RDM $\gamma(x,x')$, where the dotted lines indicate the diagonal $x=x'$. This can be contrasted with row 3, showing the purely non-interacting 1-RDM $\gamma_{\mathrm{NON}}(x,x')$. All quantities are given in a.u.

2. $n_1>0$ and $n_2>0$.

Applying PCA to the data-set we find the components shown in figure 5(a). The principal components have the variances $[2.19\times10^{-3}, 4.19\times10^{-16}]$. The PCA has encoded some physical insight: the first principal component describes that if an amount of charge is removed from one spacial position, it must be added to the other spacial position. This describes the charge is free to move along the x-axis. Component 2 describes adding and removing charge from the system. As the variance of this component is (numerically) zero, it indicates that the amount of charge must be the same as the average system, therefore illustrating the conservation of particle number. The range of these components in the data-set gives us the positivity condition. This shows that in this simple case, PCA can be used to encode both of the constraints on the density, due to their linearity. We see this trend continues to the 3, 4, 6, and 62 grid point data-sets: the PCA finds that a N point system can be reduced to *at least*

$$N\rightarrow N-1 \tag{12}$$

components losslessly, as the final component is entirely determined by the linear normalisation condition. Therefore the points of the density lie in a N dimensional-flat plane.

When considering the solution to a quantum system we assume $V(x)$ can take any form that gives a valid solution of the Schrödinger equation. But in reality, when studying a class of systems, such as molecules, the range of external potentials is much smaller, simply determined from the atomic positions and charges. These potentials are constrained by the atomic nature of matter within the Born-Oppenheimer framework. Constraining $V(x)$ in this way has the effect of also constraining the observables - maybe some of these constraints are linear. If there are additional linear constraints, they will be found by PCA. Our 62-point data-set has a characteristic well defined structure, as it is formed from a Fourier series and confining potential, in addition to the usual smoothness and continuous requirements of the charge density. This is in contrast to the 2 point case where potential is essentially 2 independent random values, where there is no concept of smoothness. The inset of figure 5 (b) shows the logarithm (to the base 10) of the variance of each component in the 62-point case. This shows that only 41 components are necessary to describe the density to numerical accuracy, much smaller than $N-1$ (61). This has captured these additional linear constraints. Figure 5 (b) shows how the structure of the density is assembled from adding successive principal components. Only 15 are needed to reduce the error to 10^{-6} (a.u.). This illustrates that using a constrained class of external potentials, in addition to the usual smoothness constraints, leads to additional constraints in the charge density, which in turn leads to additional linear constraints that can be extracted using PCA.

4.2 Constraints of the density matrix

We now apply the PCA to the 1-RDM. We would expect the PCA to learn the same linear constraints as for the density, as the 1-RDM contains the density along its diagonal. Moreover, the PCA should learn the additional linear constraint of symmetry, so altogether:

1. $\sum_i \gamma_{ii}=\frac{N}{dx}$

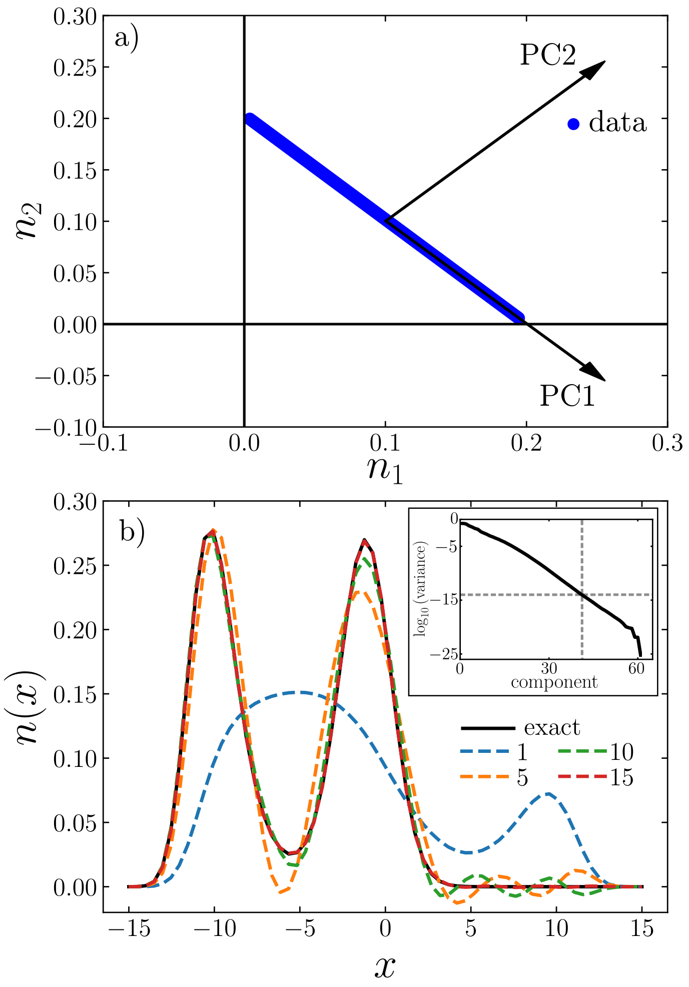

Fig. 5 PCA being applied to the charge density. Panel (a) shows the density data values of the 2-point data-set along with the two orthogonal principal components (PC). PC1 corresponds to the charge moving between the two points, and PC2 corresponds to changing the net value of charge. From the variances $[2.19\times10^{-3}, 4.19\times10^{-16}]$ we see that the PCA has learned that the density is constrained by the total charge. By looking at the data this way we can see clearly that this is a linear constraint that the PCA can capture exactly. The inset in panel (b) shows the logarithm (to the base 10) of variance of the PCA components for the 62-point data-set. The horizontal grey dotted line illustrates floating point numerical precision, and the vertical indicates the 42 components needed to obtain such accuracy. Panel (b) shows an example 62-point density of various numbers of included principal components, along with the exact density for comparison.

2. $\gamma_{ii} > 0\ \forall i$

3. $\gamma_{ij} = \gamma_{ji}\ \forall i, j$.

Where again we consider the 2-point case, and so the 1-RDM is represented by 4 values $\gamma_{11} = \gamma(x_1, x_1)$, $\gamma_{12} = \gamma(x_1, x_2)$, $\gamma_{21} = \gamma_{21}(x_2, x_1)$, $\gamma_{22} = \gamma(x_2, x_2)$. We will write these in 'flattened' form.

$$\gamma_{ij} \rightarrow \gamma_{r[ij]}. \tag{13}$$

In this way, the 1-RDM becomes a 4-dimensional vector, and so, in our case, the four elements of the 1-RDM are denoted $\gamma_{11} \rightarrow \gamma_{r=1}$, $\gamma_{12} \rightarrow \gamma_{r=2}$, $\gamma_{21} \rightarrow \gamma_{r=3}$, $\gamma_{22} \rightarrow \gamma_{r=4}$. Where $\gamma_{r=1}$ and $\gamma_{r=4}$ are the diagonal elements. We apply PCA to this 2-point data-set of 1-RDMs. We observe that this yields 2 components with non-zero variance. These are shown in figure 6 (a), and compared to the components of the density obtained in section 4.1. We see that the component 1, corresponding the direction of maximum variance in the data-set, is exactly the same as the corresponding density component, with no non-zero value in the off-diagonal terms. This can be thought as moving along the flat density plane. It is this term (along with the fact that the component changing the net charge has a variance of zero) that captures the first 2 constraints, as in section 4.1. Component 2 has no non-zero values in the diagonal, but only values in the off-diagonal terms. As indicated by arrow pair 1, the two off-diagonal terms have the same value, this has captured the symmetric constraint. The PCA describes: if you set γ_{12} by a given value, you must set γ_{21} to exactly the same value.

We now apply the PCA to the 4-point data-set of 1-RDM. This will inform what compression the PCA can perform losslessly: *can it encode the N^2 elements of the 1-RDM by only N, as in DFT?* We find that this is not the case, as in the 4-point case the PCA can perform the lossless compression from 4^2 elements to 9. In general we find that for an N-point system the PCA can perform lossless compression to *at least*

$$N^2 \rightarrow \frac{N(N+1)}{2} - 1. \tag{14}$$

This is simply the number of diagonal elements subtract 1 plus the half the number of off-diagonal elements. This is exactly the amount that is derived from the three constraints of the 1-RDM, and hence the PCA finds there are no additional linear constraints we were missing. Figure 6 (b) shows the first six non-zero principal components of the 1-RDM. The diagonal values of the first three components correspond exactly to that of the density principal components, and the off-diagonal values are almost zero, except for small features appearing in the elements adjacent to diagonal ones, for example as indicated by arrow pair 2. The remaining components describe only the off-diagonal elements, and once again, due to values coming in pairs (see for example, arrow pair 3), reflect the symmetry constraint. The fact that some off diagonal values are non-zero in the components that correspond to the density is significant as it allows the separation of the linear and non-linear terms of $\gamma[n]$ in a domain specific way. This idea will be developed further in section 5.3.

As we found in section 4.1 that additional linear constraints on the density emerge when the structural constraints are applied to

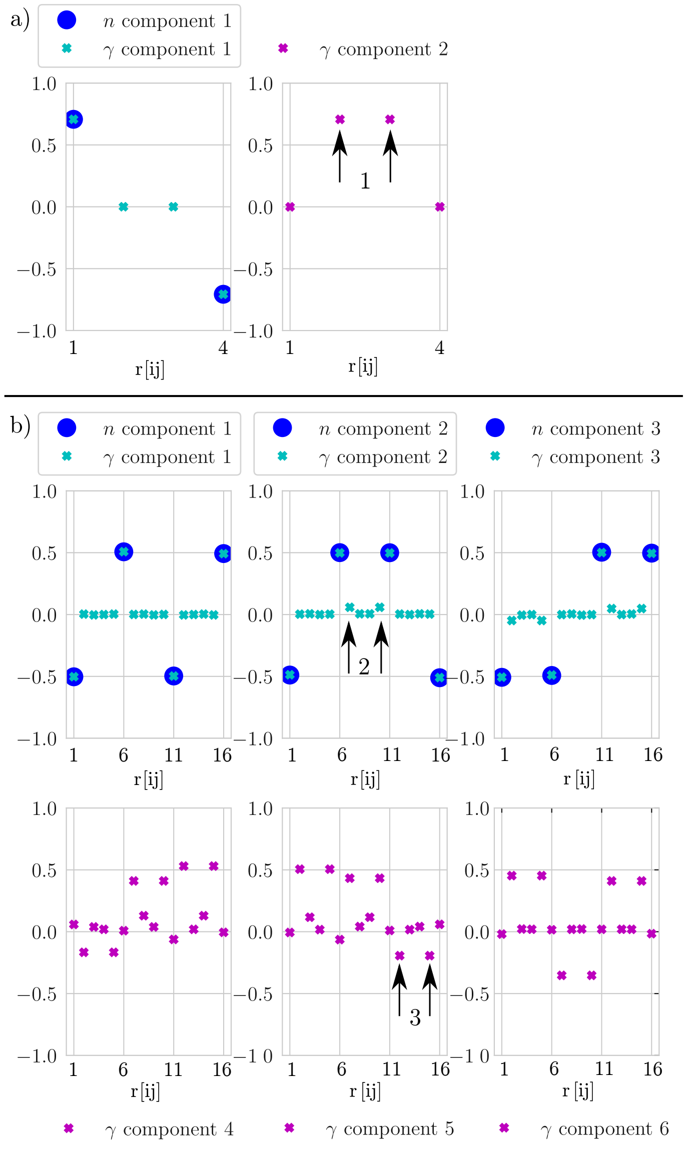

Fig. 6 PCA being applied to the 'flattened' (see equation 13) 1-RDM. Panel (a) shows a comparison of the non-zero variance 1-RDM components with that of the density for the 2-point data-set. *The vertical grey lines indicate the diagonal elements.* For the first principal component the 1-RDM is identical to that of the density along the diagonal and zero value off-diagonal. The second principal component shows that the upper and lower off-diagonal elements must always equal (symmetry constraint indicated with arrow pair 1). Panel (b) shows a comparison of the first 6 non-zero variance 1-RDM principal components in comparison to the 3 non-zero variance density principal components for the 4-point data-set. *Again, the vertical grey lines indicate the four diagonal elements.* The principal components of the density again match that of the diagonal of the first 3 principal components of the 1-RDM. While components 1 is zero for the off diagonal elements, components 2 and 3 have some small contribution to the off diagonal elements, for example the values indicated by arrow pair 2 (see discussion in main text). The next three components have non-zero value in the off diagonal directions, and the figure illustrates that $\gamma_{ij} = \gamma_{ji}$ (see for example arrow pair 3).

the external potential, and due to the smoothness of the density, we would like to see to what extent this extends to the 1-RDM, and to what extent this can be utilised. The top row of Figure 7 (c) shows a 2D-view of the first 5 components of the 1-RDM for the 62-point data-set. We would expect that, if there were no additional linear constraints, PCA would find $62^2 \to \frac{62(62+1)}{2} - 1 = 1952$ lossless compression to be obtained. We find only 327 are required within our numerical precision. This implies that, as we approach the continuum by increasing the number of grid points, additional linear constraints manifest in the 1-RDM. This is because each of the elements γ_{ij} cannot be treated independently, there is an emerging additional structure due to the smoothness and continuous properties of the wavefunction, and from the constraints we impose on the external potential being formed from Fourier components. These properties have no meaning in the 2 point system, and hence do not appear. In the bottom row of 7 (c) we compare the diagonals of these first 5 components (scaled due to the PCA normalisation convention), with the first 5 density components: we see they correspond exactly, but have significant weights off the diagonal elements γ_{ii}. This allows us to describe the linear part of the functional $\gamma[n]$ using our data-set. We will explore constructing functionals from this premise in section 5.3.

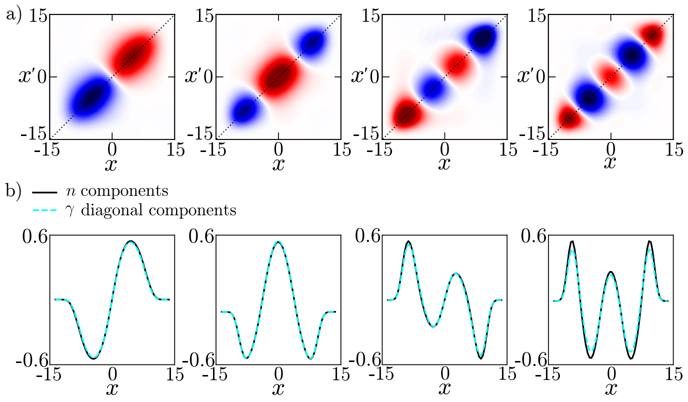

Fig. 7 PCA being applied to the 1-RDM in the 62-point case. Panel (a) is the first 5 principal components of the 1-RDM for the 62-point data-set. Panel (b) compares the (scaled due to the PCA normalisation convention) diagonals of these components to the first 5 principal components of the density.

The PCA is unable to perform the reduction of elements $N^2 \to N$ because it imposes linearity. Without this constraint, we know this mapping is in principle possible as the N^2 elements of the 1-RDM is defined by only N, for example from the external potential or charge density. We now transcend this request for linearity by applying a CAE to the 1-RDM for the 62-point data-set, where we set the number of values in the bottleneck layer to be 512. Applying PCA to the bottleneck data we further reduce to $N = 62$. This yields the final mapping of the model to be $N^2 \to N \to N^2$ as desired. We find that the model can reconstruct the input to a mean average error of 1.7×10^{-3} a.u. (average error of each γ_{ij}) Figure 8 illustrates the CAE being applied to eight example systems. Now we have various machine learning models encoding both linear and non-linear constraints for the 1-RDM, and we can

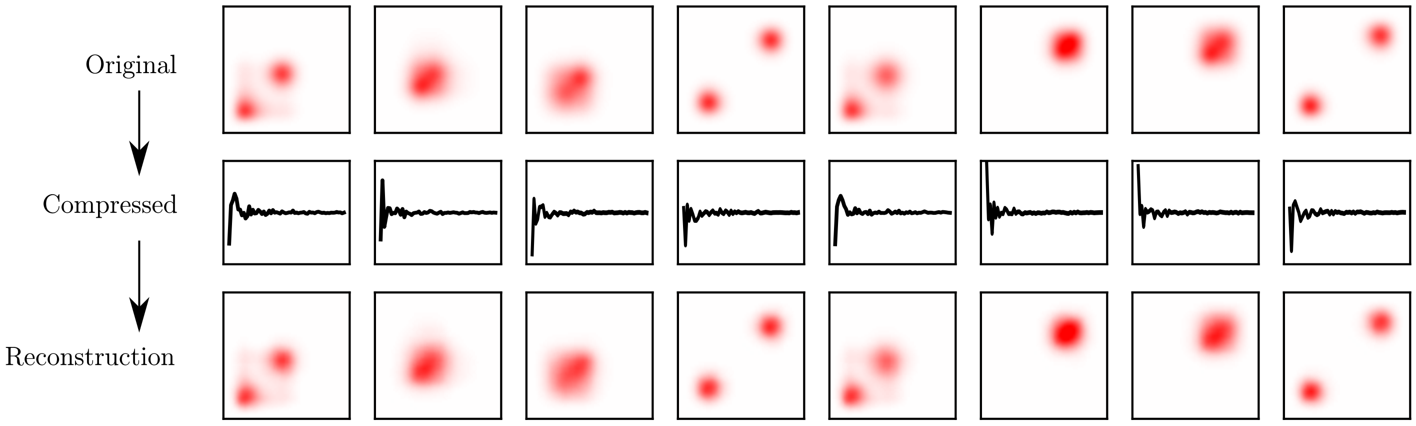

Fig. 8 The CAE ($N^2 \to N \to N^2$) being applied to eight example 1-RDMs (illustrated in figure 2). The top row shows the original exact 1-RDM in each case (with the same axis as figure 4). The second row shows the encoded representation of N values in each case. The final row shows the decoder being applied to the compressed data, reconstructing the original N^2 1-RDM to a mean average error of 1.7×10^{-3} (a.u.) over the data-set.

utilise what has been learned to construct approximations to the functional $\gamma[n]$.

5 Learning Functionals

5.1 Feature Engineering

Before moving on to deep learning the 1-RDM functional, we first investigate to what extent we can assist machine learning models with pre-existing knowledge of the density matrix in the simplest possible case. The cornerstone of this process is *feature engineering*. Any appropriately complex neural network can brute-force correct predictions, but in order to obtain an efficient model it is necessary to determine the best way in which the data should be presented to the machine.

Let us start by considering the functional $\gamma[n]$ for a two points system containing two electrons of opposite spin. As the 1-RDM has the charge density along its diagonal, and is symmetric, this problem reduces to finding the function

$$\gamma_{21}(\gamma_{11}, \gamma_{22}). \tag{15}$$

The universal approximation theorem[54] guarantees that a MLP with a sufficiently large hidden layer can fit any function. In order to ensure only a small hidden layer is needed, we perform feature engineering. Let us start from the two-points Hamiltonian diagonalized by the iDEA code (see section 3):

$$\hat{H} = -t \sum_{\sigma\in\{\uparrow,\downarrow\}} (\hat{c}^\dagger_{1,\sigma}\hat{c}_{2,\sigma} + \hat{c}^\dagger_{2,\sigma}\hat{c}_{1,\sigma}) + U\sum_{i=1}^{2} \hat{n}_{i,\uparrow}\hat{n}_{i,\downarrow} + $$
$$+ U'(\hat{n}_{1,\uparrow}\hat{n}_{2,\downarrow} + \hat{n}_{1,\downarrow}\hat{n}_{2,\uparrow}) + \sum_{i=1}^{2} v_i\hat{n}_i$$

The term U' corresponds to the repulsion of the electrons when populating different sites. The distance in between the points has been appropriately tuned in order to make this term negligible with respect to the on-site repulsion, so that the system can be modelled as an inhomogeneous Hubbard-dimer model:

$$\hat{H} = -t \sum_{\sigma\in\{\uparrow,\downarrow\}} (\hat{c}^\dagger_{1,\sigma}\hat{c}_{2,\sigma} + \hat{c}^\dagger_{2,\sigma}\hat{c}_{1,\sigma}) + U\sum_{i=1}^{2} \hat{n}_{i,\uparrow}\hat{n}_{i,\downarrow} + \sum_{i=1}^{2} v_i\hat{n}_i \tag{16}$$

Starting from this Hamiltonian, let us define an adimensional quantity named interaction strength

$$u = \frac{U}{4t}.$$

This number represents the relative importance of the on-site repulsion with respect to the kinetic term. By performing a variational constrained minimization of the Hamiltonian[55,56] it is possible to extract the desired functional in the two limiting cases of non-interacting electrons :

$$\gamma^0_{21} = \gamma_{21}(u=0) = \sqrt{\gamma_{11}(2-\gamma_{11})}, \tag{17}$$

and of strongly-interacting electrons:

$$\gamma^\infty_{21} = \gamma_{21}(u\to\infty) = \begin{cases} \sqrt{2(\gamma_{11}-1)(2-\gamma_{11})} & \text{if } \gamma_{11} \geq 1 \\ \sqrt{2\gamma_{11}(1-\gamma_{11})} & \text{if } \gamma_{11} < 1. \end{cases} \tag{18}$$

In terms of the variable $\gamma_m = \min\{\gamma_{11}, 2-\gamma_{11}\}$ this becomes

$$\gamma^\infty_{2,1} = \sqrt{2\gamma_m|1-\gamma_m|}. \tag{19}$$

This allows us to drastically reduce the complexity of the network needed for fitting the data. The relations can be written as

$$\gamma_{21}(x_1,x_2) = x_1^{\omega_1}x_2^{\omega_2} = e^{\omega_1\log x_1 + \omega_2\log x_2 + b}, \tag{20}$$

where $\omega_i = 1/2$ and $b = 0$

$$(x_1,x_2) = \begin{cases} (2\gamma_m, 1-\gamma_m) & \text{if strongly-interacting} \\ (\gamma_{11},\gamma_{22}) & \text{if non-interacting.} \end{cases}$$

We can then define

$$O_P = f^\sigma(\sum_k \omega_k\hat{x}_k + b), \tag{21}$$

	ω_1	ω_2	b
γ^0	0.5000 ± 0.0002	0.5001 ± 0.0001	$(2.0 \pm 0.1)10^{-6}$
γ^∞	0.480 ± 0.003	0.480 ± 0.002	$(-4 \pm 8)10^{-5}$

Table 1 Result of the fitting procedure using the logarithmic perceptron as an average over 20 example training sessions, along with the corresponding uncertainty.

where $f^\sigma(x) = e^x$, $\sum_k \omega_k \hat{x}_k$ is the weighted sum of the inputs, that are defined to be $\hat{x} = \log x$ and the bias is given by b. O_P, in the presented form, is the output of a perceptron, which is the simplest neural network, as being composed by one single neuron. This is termed a logarithmic perceptron[57]. It is important to note that a brute force MLP could always yield an equally accurate result, but it would require a large hidden layer of many perceptrons. In contrast this model needs only one. The computational burden has been reduced to a three parameter model to be fitted by the logarithm of the original input data.

We train the logarithmic perceptron using the mean square error loss function and the Adam optimizer with a learning rate of 10^{-3}. The bias has been initialized to 0 and a norm-2 bias regularizer with a coefficient of 10 has been introduced in order to highly penalize any value of the bias different from zero. The average parameters of 20 training sessions, computed both for the interacting and for the non interacting case are reported in table 1. As expected, the machine has learned that the bias is negligible[†] with respect to the ω-parameters, that have been estimated to be $\omega \simeq 0.5$. While for the non-interacting case the result is exact, being the non-interacting condition exactly reproducible, the strongly interacting case only approximately matches the infinitely interacting case, being this condition a limit.

We have determined that two analytical limits can be encoded in an engineered minimally complex architecture. In this small system, this yields a network that is vastly simpler than a brute force MLP. The logarithmic perceptron is engineered to optimally describe the relationship in between the variables and so is a candidate building block to construct neural network models for finding the desired functionals when more than two grid-points are concerned. This is because it could be possible to take advantage of the capability of this perceptron to introduce the correct non-linearity, while possibly allowing to physically interpret the final architecture as a nested combination of Hubbard dimers for modelling systems with a higher number of grid-points[57].

5.2 A perturbative approach

In section 5.1 we have shown that the two point system can be modeled as a Hubbard dimer, and we have given the explicit functional form of the off-diagonal term in the two limiting cases. This then defines a domain between these two cases.

The starting point is to express the equations 17 and 18 in terms of the variable γ_m:

$$\gamma^0_{21}(\gamma_m) = \sqrt{2\gamma_m - 1\gamma_m^2} \tag{22}$$

$$\gamma^\infty_{21}(\gamma_m) = \sqrt{2\gamma_m - 2\gamma_m^2} \tag{23}$$

By observing the structure of these laws, we postulate that the functional form at intermediate values of the interaction strength can be written as

$$\gamma_{2,1}(\gamma_m, u) = \sqrt{2\gamma_m - \chi(\gamma_m, u)\gamma_m^2}. \tag{24}$$

The point $\gamma_m = 1$ is a special value for the Hubbard dimer model since it corresponds to the point in which the value of the density at the two sites is the same. This can only occur when the dimer is symmetric ($v_1 = v_2$). Performing the diagonalization of the Hamiltonian of the symmetric dimer, the value of the off-diagonal term of the 1-RDM as a function of the interaction strength is found to be

$$\gamma_{21}(\gamma_m = 1, u) = -\frac{u - \sqrt{u^2+1}}{1 + u(u - \sqrt{u^2+1})}. \tag{25}$$

This relation fixes the value of the χ function at the symmetric point:

$$\chi(\gamma_m = 1, u) = 2 - [\gamma_{2,1}(\gamma_m = 1, u)]^2. \tag{26}$$

Apart from this, nothing obvious can be said about the γ_m dependence of the χ function. However, we choose to write it as

$$\begin{aligned} \chi(\gamma_m, u) &= \chi(\gamma_m = 1, u) + \Delta\chi(\gamma_m, u) \\ &= \chi^{(0)}(u) + \Delta\chi(\gamma_m, u) \end{aligned} \tag{27}$$

and the following functional constraints must necessarily be true

$$\begin{cases} \Delta\chi(\gamma_m = 1, u) = 0 \\ \Delta\chi(\gamma_m, u = 0) = 0 \\ \Delta\chi(\gamma_m, u \to \infty) = 0. \end{cases} \tag{28}$$

The first constraint, by definition, is valid whatever u in the symmetry point ($\gamma_m = 1$). The second and the third constraints are due to the fact that $\chi^{(0)}(u = 0) = 1$ and $\lim_{u\to\infty} \chi^{(0)}(u) = 2$. Since this must be true for all the values of γ_m, the correction must be zero.

Considering that the correction vanishes at both the borders of our domain, and also at the symmetry point, and that a crossings of two any curves of the off-diagonal term for different values of the interaction strength should not occur, one would expect the correction to be a perturbation of the $\chi^{(0)}$-model.

In figure 9(a) we directly compare the $\chi^{(0)}$-model with the exact .

As this has verified that the correction is indeed a perturbation of the proposed model, we now employ a neural network architecture to determine this correction. We generate an additional data-set containing 600,000 couples $(\hat{\gamma}_m, \hat{u})$. For a range of values of u different potential landscapes have been defined and the density matrix has been computed. The corresponding values of $\Delta\hat{\chi}(\gamma_m, u)\gamma_m^2$ have been used as labels to be learned in the regres-

† $b = o(\omega)$ since it is smaller than the precision with which the value of ω is known.

sion procedure. The quantities have been redefined as $\hat{x} = 10x$ and the factor 10 has been introduced to ensure the data is of favorable scale for working in non-linearity with the selected activation functions. This activation function is chosen to be the hyperbolic tangent as it is capable of reaching negative values. A reasonable choice for the number of neurons in the three layers has been found to be $(12, 12, 16)$. Due to the simplicity of the model some details of the correction are missed in the fitting, in particular the vanishing of the correction at the symmetry point and the vanishing of the correction in the non-interacting limit. Rather than increasing the complexity of the network we have preferred to impose this functional requirement by multiplying the prediction by two exponential corrections. The equation of the correction reads

$$\Delta\chi(\gamma_m, u)\gamma_m^2 \simeq \frac{O(\hat{\gamma}_m, \hat{u})}{10}(1 - e^{-\frac{1-\gamma_m}{\lambda_g}})(1 - e^{-\frac{u}{\lambda_u}}) \tag{29}$$

where $O(\hat{\gamma}_m, \hat{u})$ is the prediction of the network while $\lambda_g = 0.001$ and $\lambda_u = 0.004$ are two numerical coefficients appropriately chosen. Figure 9(b) compares the inclusion of this correction to the exact result, showing a significant increase in accuracy.

Up to this moment we have used machine learning tools in order to enhance our theoretical models and to learn them. We will now move to larger grid-points systems, using the capability of the machine to learn from the data for the construction of approximate functionals.

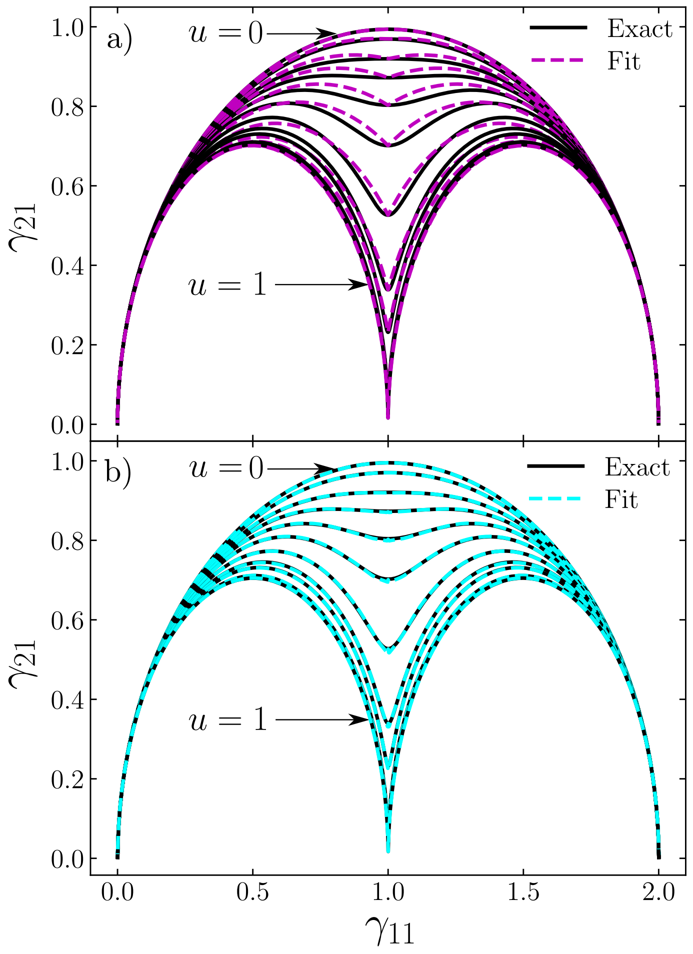

Fig. 9 Panel (a) shows the performance of the zeroth order χ-model for a range of interaction strengths u from 0 to 1, in comparison to the exact case. Panel (b) shows the χ-model with the machine learned correction. This demonstrates that the machine has learned a significant improvement.

5.3 Learning functionals from constraints

We will now move to applying the insights into constraints we gained in sections 4.1 and 4.2 to develop the functional $\gamma[n]$ for the 62 point data-set, and we will then benchmark the resulting estimations of the 1-RDMs against the exact ones. The functional we desire can be split into a linear (L), and non-linear (NL) term:

$$\gamma[n] = \gamma_{\mathrm{L}}[n] + \gamma_{\mathrm{NL}}[n]. \tag{30}$$

In the following we will explicitly perform this linear decomposition. While the linear term will be presented as an explicit functional of the density, the non-linear one will be treated as a perturbation, and will be deep learned in section 5.4.

In section 4.2 we found that with PCA, due to additional structural constraints, the principal components contained non-zero values in both the diagonal and non-diagonal elements. In particular, the diagonal of the 1-RDM components had a significant correspondence to the density components (see figure 7). For a given number of grid points, this correspondence holds for the first ν components, where this value is determined by analysing the PCA components. The purpose of this section is to exploit this correspondence in order to find a linear approximation of $\gamma[n]$. We will begin by formalizing the connection in between the two data-sets introduced in sections 4.1 and 4.2. We will first introduce the principal component decomposition of the 1-RDM in the $\mathscr{B}_e$-basis introduced in section 2.3 and then, starting from the matrix representation of the density data-set, we will express in formulas the content of figure 7. The starting point is the expression of the 1-RDM components in terms of the known projections of the principal components onto the basis defining the entries of the matrix (see section 2.3).

$$\gamma_{i,j} = \langle e_{r[i,j]}|\gamma\rangle =$$
$$= \langle e_{r[i,j]}|\gamma_0\rangle + \langle e_{r[i,j]}|\tilde{\gamma}\rangle =$$
$$= (\gamma_0)_{i,j} + \sum_{k,r'=1}^{N^2} \langle p_k|e_{r'}\rangle\langle e_{r'}|\tilde{\gamma}\rangle\langle e_{r[i,j]}|p_k\rangle$$

Where the $\langle e_r|p_i\rangle$ coefficients are known after the evaluation of the eigenvectors of the covariance matrix.
Before proceeding, we introduce the density vectors:

$$\underline{n} = [n(x_1), 0, \cdots, 0, n(x_2), 0, \cdots, n(x_N)] \in \mathbb{R}^{N^2} \tag{31}$$

and the corresponding data-set

$$\underline{\underline{P}} = \begin{bmatrix} \underline{n}^{(1)} \\ \vdots \\ \underline{n}^{(T)} \end{bmatrix}. \tag{32}$$

Performing a PCA on this data-set allows one to determine the average density $|n_0\rangle$, the decomposition $|n\rangle = |\tilde{n}\rangle + |n_0\rangle$ and the corresponding principal components

$$\{|p_i^n\rangle\}_{i=1}^{N^2}.$$

There are N^2 components since they must form an orthonormal basis of the vector-space. However, only the first $N-1$ principal components will be informative due to the sparseness of the object defined and due to the normalization of the density (see section 4.2). By direct comparison of the principal directions in the two data-sets (see figure 6), it is possible to observe that the first ν principal directions derived from the Γ data-set can be put approximately in a scaled one-to-one correspondence with the first ν principal directions of the sub-data-set P. In particular, let us define the modified principal directions and let us normalize them

$$|q_i\rangle = \sum_{j=1}^{N} \langle e_{r[j,j]}|p_i\rangle|e_{r[j,j]}\rangle \quad i = 1, \cdots, N-1 \rightarrow |q_i^n\rangle = \frac{1}{\sqrt{\langle q_i|q_i\rangle}}|q_i\rangle.$$

This defines an orthonormal basis for describing the diagonals of the matrices in the ensemble. When the matrices under analysis are such that the non-vanishing off-diagonal terms are mainly the ones closer to the corresponding non-vanishing diagonal terms, it must be true that $|q_i^n\rangle \simeq \pm|p_i^n\rangle$.[‡] Where the equality has been observed to be exact for the first ν principal components since the presence in the density matrix of the off-diagonal terms leads to a reduction in priority of the variation along the density. In fact, the main variations in the density are also those more strikingly characterizing the 1-RDM. However, from the $(\nu+1)-th$ component on, while the PCA on the density can provide more details on the remaining changes in the density, orthogonal to the previous ones, the PCA on the density matrix, starts describing the variations along the off-diagonal terms and the mapping in between the two breaks down since the details on the density, being less evident than the ones of the off-diagonal terms, are contained in a diluted way in the remaining components.

We will now move to determine the actual expression of the linear functional. First, we separate the mean-adjusted 1-RDM in four terms, distinguishing the diagonal from the off-diagonal contributions and taking into account the different information content of the first ν principal component with respect to the remaining ones. We will then define a basis orthonormal in the subspace of the densities while carrying off-diagonal information. This definition will allow to approximate the dominant contribution to the exact functional $\gamma[n]$. Let us start by writing $\tilde{\gamma}$ as the sum of four contributions:

$$|\tilde{\gamma}\rangle = |\tilde{n}_{\leq\nu}\rangle + |\tilde{n}_{>\nu}\rangle + |\delta\gamma_{\leq\nu}\rangle + |\delta\gamma_{>\nu}\rangle$$

These four terms correspond to the shifted density reconstructed with the first ν principal components, to the shifted density reconstructed with the remaining principal components, to the off-diagonal terms obtainable with the first ν principal components, from now on termed *free off-diagonal terms*, and to the remaining off-diagonal contributions. Considering or not these terms corresponds to different levels of approximation. For our specific data-set it has been shown that the moment in which the one-to-one mapping stops to hold corresponds to the number of principal components ν at which the cumulative sum of the explained variance ratio reaches a value of 0.91. For this reason, neglecting the term $|\tilde{\gamma}_{>\nu}\rangle = |\tilde{n}_{>\nu}\rangle + |\delta\gamma_{>\nu}\rangle$ will be considered as a reasonable first order approximation, and we will be able to focus on the remaining two terms. This having been said, let us define a new set of vectors:

$$|q_i^\gamma\rangle \doteq \frac{1}{\sqrt{\langle q_i|q_i\rangle}}|p_i\rangle \quad i = 1, \cdots, N-1 \qquad \langle q_i^n|q_j^\gamma\rangle = \delta_{i,j}$$

These vectors contain the $|q_i^n\rangle$ ones in them and follow their normalization. Thanks to their orthogonality, if the one-to-one mapping were valid for all the first $N-1$ components, they would be a complete basis in the densities-subspace while varying the information on the free off-diagonal terms in the components not shared with the basis $\{|q_i^n\rangle\}_{i=1}^{N-1}$. Even if the mapping is valid only for the first ν principal components, this basis allows nonetheless for the construction of the approximate 1-RDM carrying some information on the off-diagonal behavior while its diagonal corresponds to the density reconstruction obtained by looking at its ν most remarkable features. Let us add and subtract this term in $\gamma_{i,j}$:

$$\gamma_{i,j} = (\gamma_0)_{i,j} + \langle e_{r[i,j]}|\tilde{n}_{\leq\nu}\rangle + \langle e_{r[i,j]}|\delta\gamma_{\leq\nu}\rangle + \langle e_{r[i,j]}|\tilde{\gamma}_{>\nu}\rangle +$$
$$+ \sum_{k=1}^{\nu} \langle e_{r[i,j]}|q_k^\gamma\rangle\langle q_k^\gamma|\tilde{n}\rangle - \sum_{k=1}^{\nu} \langle e_{r[i,j]}|q_k^\gamma\rangle\langle q_k^\gamma|\tilde{n}\rangle =$$
$$= (\gamma_0)_{i,j} + \sum_{k=1}^{\nu} \langle e_{r[i,j]}|q_k^\gamma\rangle\langle q_k^\gamma|\tilde{n}\rangle + (\delta\gamma)_{i,j}$$

‡ The $\pm$ is due to the possible differing convention of the arbitrary directions in the PCA.

where

$$(\delta\gamma)_{i,j} = \langle e_{r[i,j]}|\tilde{n}_{\leq\nu}\rangle + (\delta\gamma_{\leq\nu})_{i,j} - \sum_{k=1}^{\nu}\langle e_{r[i,j]}|q_k^{\gamma}\rangle\langle q_k^{\gamma}|\tilde{n}\rangle + (\tilde{\gamma}_{>\nu})_{i,j}$$

This last quantity must be itself a functional of the density, where the functional relation is non-linear and unknown. This having been done, the functional is now expressed in the form presented in equation 30. Considering the non-linear part, the analysis on the principal values legitimates us to neglect the term $(\tilde{\gamma}_{>\nu})_{i,j}$. For what concerns the remaining contribution it is expected to be small since the basis $\{|q_k^{\gamma}\rangle\}_{k=1}^{N-1}$ has been defined with the exact intent of privileging the exact restoration of the density, being the biggest contribution, while estimating the free off-diagonal terms. By neglecting the $\gamma_{\mathrm{NL}}[n] = \delta\gamma$ term and by writing the resulting expression in terms of the know projections of the principal components onto the $\mathscr{B}_e$-basis, the linear functional is obtained

$$\gamma_{i,j} = (\gamma_0)_{i,j} + \sum_{r':\langle e_{r'}|\tilde{n}\rangle\neq 0}\sum_{k=1}^{\nu}\langle e_{r[i,j]}|q_k^{\gamma}\rangle\langle q_k^{\gamma}|e_{r'}\rangle\langle e_{r'}|\tilde{n}\rangle$$

$$= (\gamma_0)_{i,j} + \sum_{s=1}^{N}(n(x_s) - n_0(x_s))\sum_{k=1}^{\nu}\langle e_{r[i,j]}|q_k^{\gamma}\rangle\langle q_k^{\gamma}|e_{r'[s,s]}\rangle$$

$$\gamma_{i,j}[n] = (\gamma_0)_{i,j} + \sum_{s=1}^{N}(n(x_s) - n_0(x_s))\sum_{k=1}^{\nu}\langle q_k^{1}|e_{r'[s,s]}\rangle\langle e_{r[i,j]}|q_k^{1}\rangle \quad (33)$$

Writing this in the position basis for our 1-dimensional data-set, we arrive at our approximate functional, we term this *the PCA functional*:

$$\gamma_{\mathrm{L}}^{\mathrm{PCA}}[n(\tilde{x})](x,x') = \gamma_0(x,x') + \hat{P}\hat{P}_{\nu}^{-1}(n(\tilde{x}) - n_0(\tilde{x}))(x,x'), \quad (34)$$

where γ_0 is the average density matrix from the data-set (due to the PCA convention to transform between mean-adjusted data), $\hat{P}$ is the PCA 1-RDM transformation, and $(n - n_0)$ is the mean adjusted charge-density. $\hat{P}_{\nu}^{-1}$ is the diagonal-only inverse PCA transform. This takes a mean-adjusted density, and returns the ν PCA components of γ that when transformed to real space will contain that density along its diagonal. This chooses our PCA components of the density matrix so they must have our given density along its diagonal. When the PCA transform is applied to give the density matrix in real space we also obtain off diagonal elements linearly approximated by this inverse transformation. This inverse has some very small eigenvalues due to some of the off-diagonal principal components of γ containing small diagonal values. We use singular value decomposition to remove these eigenvalue in order to perform the inverse.

This approximate functional can be thought of as a *domain specific* linear expansion, akin to a Taylor expansion. It is domain specific in two ways; first that the PCA orders components by variance, where the neglected terms are the smallest possible by definition, and so is engineered for an optimal linear approximation. Secondly, that the region of which the expansion is accurate has been specified by a data-set of systems of interest. This can be made analogous to domain specificity in image compression: An autoencoder can be trained to yield optimal compression on a specific data-set (domain) of images (for example faces). If this were instead trained on all possible images, one would recover something akin to JPEG compression, and hence autoencoders are thought as domain specific image compression. In this way of thinking, this approximate functional is a *domain specific* linear expansion, as it has been trained on a representative data-set of systems, which is a small subset of all possible systems.

Figure 10 shows the application the the PCA functional to nine example systems from the 62-point data-set, with $\nu = 8$. We find that a significant contribution (on average 64.41%) of the off-diagonal elements can be described by this linear functional. This leaves only the non-linear term to be learned. In section 5.4 we will go beyond this linear term using a deep learning model.

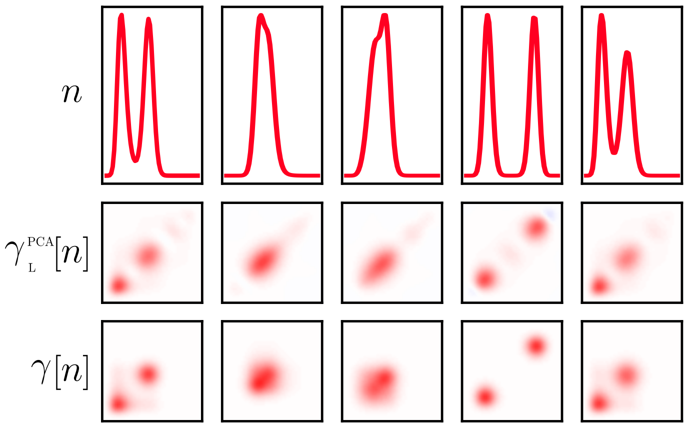

Fig. 10 Evaluating our PCA linear functional for nine sample systems from the 62-point data-set. The axis are the same as in figure 4. The first row shows each system's charge densities. The second row shows the PCA functional being applied to each of the charge densities. The third row shows the exact density matrix corresponding to each of the densities. By taking the average mean percentage difference taken over the entire data-set we find that the linear functional takes account of 64.41% of the whole density matrix, leaving only the remaining to be deep learned.

5.4 Denoising Autoencoders

We will now move to approximating $\gamma[n]$ for the 62-point systems using denoising autoencoders (DAEs). DAEs are convolutional autoencoders used to perform noise reduction in image processing[58]. Usually CAEs are trained to reconstruct their input exactly, but if noise is applied to the data-set, it can instead be trained to construct the clean data from the data with noise added. When given a novel noisy image it can reconstruct the image with the noise removed. We propose that DAEs can be used to develop functionals if we treat the difference between an approximate 1-RDM and the exact to be noise.

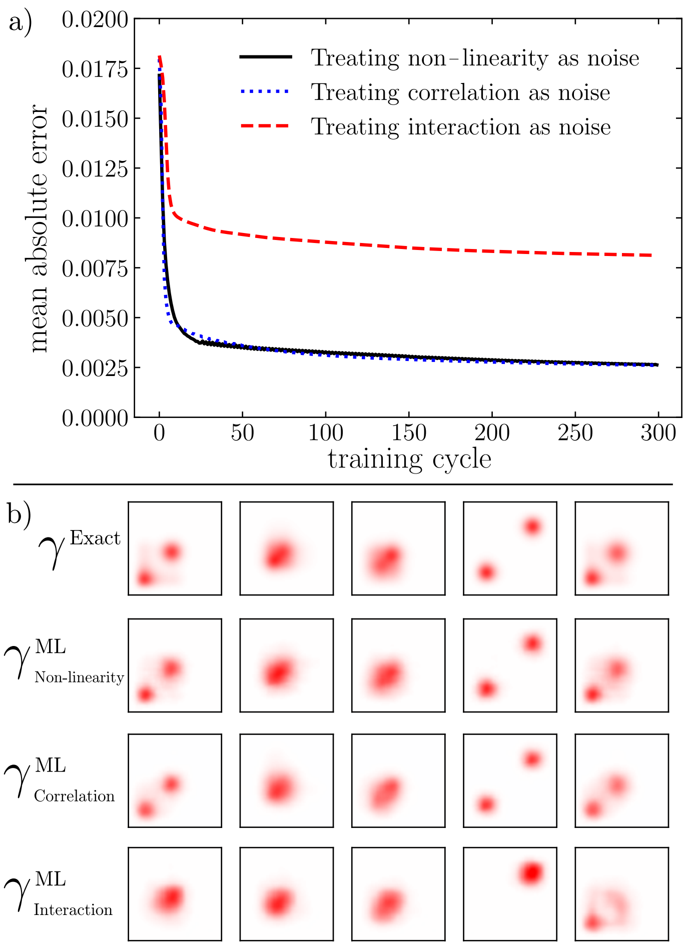

Fig. 11 Contrasting the different quantities we can treat as noise when training a DAE. Panel (a) shows the mean absolute error as a function of training cycle (where the DAE has seen every training sample once). Treating the neglect of non-linearity and correlation as noise converges to a mean absolute error 2.6×10^{-3} (a.u), whereas treating the neglect of the entirety of interaction as noise converges to 8.2×10^{-3} (a.u). Panel (b) shows the application of these three deep learning methods to five example test systems in comparison to the exact.

We have several candidates of what we can consider noise:

- By training our DAE to reconstruct the exact density matrix from the PCA functional introduced in section 5.3, we are considering the neglect of non-linearity as noise.
- By training our DAE to reconstruct the exact density matrix from the purely non-interacting, we are considering the neglect of the whole Coulomb interaction as noise.
- By training our DAE to reconstruct the exact density matrix from the UHF, we are considering as noise the neglect of correlation beyond that simulated by the symmetry breaking.

By training a DAE in each of these three cases we can see which phenomena are most amenable to be deep learned in this fashion. In figure 11 (a) we show, for equivalent DAE architectures, the mean absolute error in the predictions as a function of the training cycle. This shows that the neglect of interaction is the least applicable to be treated as noise. Treating the neglect of non-linearity and neglect of correlation effects as noise are largely equally as applicable, yielding a final mean absolute error of 2.6×10^{-3} (a.u.) (in comparison to the UHF approximation to the 1-RDM itself has a mean absolute error of 2.0×10^{-2} (a.u.) over the data-set). In figure 11 (b) we illustrate the prediction of each of the approximate deep learning methods for five example systems in comparison to the exact. It is clear to see that machine learning the interaction itself is considerable less amenable to machine learning than either the correlation effects beyond UHF and non-linearity. And therefore, combining the linear functional obtained in section 5.3 with a denoising encoder, yields a accurate approximation to the functional $\gamma[n]$.

6 Conclusions

In conclusion, we have shown that insights into the one-body reduced density matrix (1-RDM) can be gained using a variety of machine learning methods. We show that by employing a large data-set of 1-RDMs, the machine can learn the constraints underlying the data. Linear constraints are determined by principal component analysis (PCA). The PCA illustrates that using a constrained class of external potentials, in addition to the usual smoothness constraints, leads to additional linear constraints in the charge density. Subsequently, non-linear constraints can be learned from convolutional autoencoders (CAEs). We show that these constraints can be utilised to build approximations to the 1-RDM as a functional of the charge density. The PCA can be used to construct the linear part of the functional utilizing linear constraints. Subsequently, the neglect of the non-linear contribution can be considered as noise, which in turn can be rectified using a denoising autoencoder (DAE). This approach yields accurate density matrices as functions of the charge density when applied to exactly solvable model systems. We compare what quantity can best be treated as noise in this way, when building functionals using DAEs, and hence which unknown term is most amenable to machine learning. We find that the treatment of interaction is considerably more difficult than non-linearity or correlation effects beyond unrestricted Hartree-Fock (UHF). We also show how existing knowledge of the density matrix can be used to guide machine learning techniques, in particular the construction of networks using logarithmic neurons, which is a candidate to assemble more complex machine learning strategies. This two-way transfer of knowledge between existing approaches and machine learning strategies is expected to help both the analytic design of new functionals, and numerical approaches to materials computation based on machine learning.

Conflicts of interest

There are no conflicts to declare.

Acknowledgements

We thank the University of York for computational resources.

Notes and references

1 P. Hohenberg and W. Kohn, Phys. Rev. **136**, 864B (1964).

2 A. J. Coleman, Rev. Mod. Phys. **35**, 668 (1963).

3 T. L. Gilbert, Phys. Rev. B **12**, 2111 (1975).

4 M. Levy, Proceedings of the National Academy of Sciences **76**, 6062 (1979), https://www.pnas.org/content/76/12/6062.full.pdf .

5 S. M. Valone, The Journal of Chemical Physics **73**, 1344 (1980), https://doi.org/10.1063/1.440249 .

6 K. Pernal and K. J. H. Giesbertz, "Reduced density matrix functional theory (rdmft) and linear response time-dependent rdmft (td-rdmft)," in *Density-Functional Methods for Excited States*, edited by N. Ferré, M. Filatov, and M. Huix-Rotllant (Springer International Publishing, Cham, 2016) pp. 125–183.

7 N. N. Lathiotakis, N. Helbig, and E. K. U. Gross, Phys. Rev. B **75**, 195120 (2007).

8 M. Piris, Phys. Rev. Lett. **119**, 063002 (2017).

9 C. Schilling, The Journal of Chemical Physics **149**, 231102 (2018), https://doi.org/10.1063/1.5080088 .

10 K. J. H. Giesbertz, A.-M. Uimonen, and R. van Leeuwen, The European Physical Journal B **91**, 282 (2018).

11 O. Gritsenko, K. Pernal, and E. J. Baerends, The Journal of Chemical Physics **122**, 204102 (2005), https://doi.org/10.1063/1.1906203 .

12 S. Sharma, J. K. Dewhurst, N. N. Lathiotakis, and E. K. U. Gross, Phys. Rev. B **78**, 201103 (2008).

13 W. Kohn and L. Sham, Phys. Rev. **140**, 1133A (1965).

14 S. Goedecker and C. J. Umrigar, Phys. Rev. Lett. **81**, 866 (1998).

15 J. Hollingsworth, L. Li, T. E. Baker, and K. Burke, The Journal of Chemical Physics **148**, 241743 (2018), https://doi.org/10.1063/1.5025668 .

16 G. Carleo, I. Cirac, K. Cranmer, L. Daudet, M. Schuld, N. Tishby, L. Vogt-Maranto, and L. Zdeborová, Rev. Mod. Phys. **91**, 045002 (2019).

17 J. C. Snyder, M. Rupp, K. Hansen, K.-R. Müller, and K. Burke, Phys. Rev. Lett. **108**, 253002 (2012).

18 L. Li, T. E. Baker, S. R. White, and K. Burke, Phys. Rev. B **94**, 245129 (2016).

19 L. Li, J. C. Snyder, I. M. Pelaschier, J. Huang, U.-N. Niranjan, P. Duncan, M. Rupp, K.-R. MÃ$\frac{1}{4}$ller, and K. Burke, International Journal of Quantum Chemistry **116**, 819 (2016), https://onlinelibrary.wiley.com/doi/pdf/10.1002/qua.25040 .

20 J. Behler and M. Parrinello, Phys. Rev. Lett. **98**, 146401 (2007).

21 A. P. Bartók, M. C. Payne, R. Kondor, and G. Csányi, Phys. Rev. Lett. **104**, 136403 (2010).

22 J. R. Moreno, G. Carleo, and A. Georges, "Deep learning the hohenberg-kohn maps of density functional theory," (2019), arXiv:1911.03580 [cond-mat.dis-nn] .

23 Z. D. Pozun, K. Hansen, D. Sheppard, M. Rupp, K.-R. Müller, and G. Henkelman, The Journal of Chemical Physics **136**, 174101 (2012), https://doi.org/10.1063/1.4707167 .

24 R. T. McGibbon and V. S. Pande, Journal of Chemical Theory and Computation **9**, 2900 (2013).

25 J. L. McDonagh, A. F. Silva, M. A. Vincent, and P. L. A. Popelier, Journal of Chemical Theory and Computation **14**, 216 (2018).

26 M. Rupp, A. Tkatchenko, K.-R. Müller, and O. A. von Lilienfeld, Phys. Rev. Lett. **108**, 058301 (2012).

27 G. Hautier, C. C. Fischer, A. Jain, T. Mueller, and G. Ceder, Chemistry of Materials **22**, 3762 (2010).

28 B. Kolb, L. C. Lentz, and A. M. Kolpak, Scientific Reports **7**, 1192 (2017).

29 K. Ryczko, D. A. Strubbe, and I. Tamblyn, Phys. Rev. A **100**, 022512 (2019).

30 K. T. Schütt, M. Gastegger, A. Tkatchenko, K.-R. Müller, and R. J. Maurer, Nature Communications **10**, 5024 (2019).

31 J. Schmidt, M. R. G. Marques, S. Botti, and M. A. L. Marques, npj Computational Materials **5**, 83 (2019).

32 K. Mills, M. Spanner, and I. Tamblyn, Phys. Rev. A **96**, 042113 (2017).

33 Y. Suzuki, R. Nagai, and J. Haruyama, (2020), arXiv:2002.06542 [physics.comp-ph] .

34 R. Nagai, R. Akashi, and O. Sugino, npj Computational Materials **6**, 43 (2020).

35 Y. Zhou, J. Wu, S. Chen, and G. Chen, The Journal of Physical Chemistry Letters **10**, 7264 (2019).

36 P. G. Mezey, AIP Conference Proceedings **1906**, 020001 (2017), https://aip.scitation.org/doi/pdf/10.1063/1.5012279 .

37 F. Brockherde, L. Vogt, L. Li, M. E. Tuckerman, K. Burke, and K.-R. Müller, Nature Communications **8**, 872 (2017).

38 W. S. McCulloch and W. Pitts, The bulletin of mathematical biophysics **5**, 115 (1943).

39 A. Lenail, http://alexlenail.me/NN-SVG/ .

40 C.-Y. Liou, W.-C. Cheng, J.-W. Liou, and D.-R. Liou, Neurocomputing **139**, 84 (2014).

41 A. Trivedi, S. Srivastava, A. Mishra, A. Shukla, and R. Tiwari, Procedia Computer Science **125**, 525 (2018), the 6th International Conference on Smart Computing and Communications.

42 W. Zhang, K. Itoh, J. Tanida, and Y. Ichioka, Appl. Opt. **29**, 4790 (1990).

43 I. Jolliffe, "Principal component analysis," in *International Encyclopedia of Statistical Science*, edited by M. Lovric (Springer Berlin Heidelberg, Berlin, Heidelberg, 2011) pp. 1094–1096.

44 K. Pearson, The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science **2**, 559 (1901).

45 M. J. P. Hodgson, J. D. Ramsden, J. B. J. Chapman, P. Lillystone, and R. W. Godby, Phys. Rev. B **88**, 241102 (2013).

46 J. Wetherell, M. J. P. Hodgson, L. Talirz, and R. W. Godby, Phys. Rev. B **99**, 045129 (2019).

47 A. R. Elmaslmane, J. Wetherell, M. J. P. Hodgson, K. P. McKenna, and R. W. Godby, Phys. Rev. Materials **2**, 040801 (2018).

48 M. J. P. Hodgson, J. D. Ramsden, T. R. Durrant, and R. W. Godby, Phys. Rev. B **90**, 241107 (2014).

49 M. J. P. Hodgson and J. Wetherell, Phys. Rev. A **101**, 032502 (2020).

50 J. Wetherell, M. J. P. Hodgson, and R. W. Godby, Phys. Rev. B **97**, 121102 (2018).

51 M. J. P. Hodgson, J. D. Ramsden, and R. W. Godby, Phys. Rev. B **93**, 155146 (2016).

52 M. J. Hodgson, E. Kraisler, A. Schild, and E. K. Gross, The journal of physical chemistry letters **8**, 5974 (2017).

53 A. H. Skelt, R. W. Godby, and I. D'Amico, Brazilian Journal of Physics **48**, 467 (2018).

54 K. Hornik, Neural networks **4**, 251 (1991).

55 M. Saubanère, M. B. Lepetit, and G. Pastor, Physical Review B **94**, 045102 (2016).

56 W. Töws and G. Pastor, Physical Review B **83**, 235101 (2011).

57 J. W. Hines, in *Proceedings of the 1996 American Nuclear Society, International Topical Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies*, Vol. 1 (Citeseer, 1996) pp. 235–241.

58 I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, 2016) `http://www.deeplearningbook.org`.