# Neural-network quantum state tomography

Dominik Koutný, Libor Motka, Zdeněk Hradil, and Jaroslav Řeháček
*Department of Optics, Palacky University, 17. listopadu 12, 77146 Olomouc, Czech Republic*

L. L. Sánchez-Soto
*Departamento de Óptica, Facultad de Física, Universidad Complutense, 28040 Madrid, Spain and*
*Max-Planck-Institut für die Physik des Lichts, 91058 Erlangen, Germany*

We revisit the application of neural networks techniques to quantum state tomography. We confirm that the positivity constraint can be successfully implemented with trained networks that convert outputs from standard feed-forward neural networks to valid descriptions of quantum states. Any standard neural-network architecture can be adapted with our method. Our results open possibilities to use state-of-the-art deep-learning methods for quantum state reconstruction under various types of noise.

## I. INTRODUCTION

Modern quantum technologies exploit distinctive features of quantum systems to achieve performances unattainable by classical strategies. This potential advantage hinges on the capability to create, manipulate, and measure quantum states. Any experimental procedure in this area requires a reliable certification of these steps: this is precisely the province of quantum state tomography (QST) [1].

The goal of QST is to estimate the unknown quantum state through measurements performed on a finite set of identical copies of the system. If the state is described by the density matrix $\varrho$, living in a $d$-dimensional Hilbert space, $O(d/\varepsilon)$ copies are required to obtain an estimate of $\varrho$ with an error (understood as total variation distance) less than $\varepsilon$ [2]. This clearly illustrates the resource requirements of QST for large-scale systems.

In a broad sense, QST is an inverse problem [3–5]. As such, the linear inversion [6] is probably the most intuitive approach to the topic. Yet it has some cons too: it might report a non-physical state and the mean squared error bound of the estimate cannot be determined analytically. To bypass these drawbacks a variety of useful QST methods, such as Bayesian tomography [7, 8], compressed sensing [9, 10], or matrix-product states [11, 12], are at hand, although the maximum-likelihood estimation (MLE) is still the most common approach [13, 14].

From a modern perspective, QST is fundamentally a data processing problem, trying to extract information from as few noisy measurements as possible. Therefore, the estimation algorithms used in QST can be easily translated into tasks in machine learning (ML) [15–17]. Actually, neural networks (NNs) have been used to address data-driven problems: examples in quantum information include identifying phase transitions [18], detecting nonclassical features [19–21], quantum error correction [22–25], calibrating quantum devices [26, 27], speeding up quantum optimal control [28], and designing quantum experiments [29–31], to cite only but a few.

Recently, ML has been applied to QST with very promising results [32–40]. In particular, generative models [41, 42], usually restricted Boltzmann machines, have been used to treat the measurement outcomes on a quantum state [43]. These are NNs containing two layers, visible and hidden, with all-to-all connections between the neurons in different layers and none

inside each layer. This technique, although powerful, suffers from difficulties with sampling and a lack of straightforward training for larger models.

The use of feed-forward architectures, including recurrent NNs, has been recently advocated [44, 45] because these architectures are easier to train without any need for sampling steps, using gradient-based optimization with backpropagation. However, generative tasks in ML often use variational autoencoders [46] and generative adversarial NNs [47]. These are now being actively explored for learning quantum states [48–51].

Our motivation in this paper is to address the benefits of NN-based reconstruction over standard techniques. To fairly benchmark the performance we pick three representative estimators, namely linear inversion, its positivized version, and MLE, and compare them with a typical NN estimator, obtained with a feed-forward architecture. As measurement, we choose the so-called square root measurement, which was introduced as a "pretty good measurement" [80] for distinguishing possibly nonorthogonal states. Using the Hilbert-Schmidt distance between the true and the reported states as our main indicator, our results suggest that NNs predict unknown quantum states about three orders of magnitude faster compared to linear and MLE estimates. Interestingly, the average errors are similar for all the estimators considered in all dimensions. This confirms the power of deep-learning-based tools for the quantum realm.

This paper is organized as follows. In Sec. II, we briefly discuss the basic tools of QST we need for our purposes. In Sec. III we describe the details of our NN architecture and training methods. Then, we present the performance of the different estimators in Sec. IV, while our conclusions are summarized in Sec. V.

## II. BACKGROUND

We first set the stage for our model. We shall be considering a $d$-dimensional quantum system, described by a $d \times d$ density matrix $\varrho$, which requires $n \equiv d^2 - 1$ independent real numbers for its specification.

The goal of QST is to estimate $\varrho$ from measurements performed on identically prepared copies of the system. These measurements are, in general, represented by positive operator-

valued measures (POVMs) [52]: they are a set of positive Hermitian operators $\{\Pi_\ell\}$, with the properties

$$\Pi_\ell \geq 0\,, \qquad \Pi_\ell^\dagger = \Pi_\ell\,, \qquad \sum_\ell \Pi_\ell = \mathbb{1}\,. \qquad (2.1)$$

Each POVM element represents a single output of the measuring apparatus. We take every measurement as yielding $m$ distinct outcomes (which we assume to be discrete). According to Born's rule, the probability of detecting the $\ell$th output is given by

$$p_\ell = \mathrm{Tr}(\varrho \Pi_\ell)\,. \qquad (2.2)$$

To invert this equation, it is convenient to map both $\varrho$ and $\{\Pi_\ell\}$ into a suitable vector form. To this end, we use a traceless Hermitian operator basis $\{\Gamma_k\}$ ($k = 0, \ldots, n$) and $\Gamma_0 = \mathbb{1}$, satisfying $\mathrm{Tr}(\Gamma_j) = 0$ and $\mathrm{Tr}(\Gamma_j \Gamma_k) = \delta_{jk}$. In this way, we get the parametrization

$$\varrho = r_0\,\Gamma_0 + \sum_{k=1}^{n} r_k\,\Gamma_k \qquad \Pi_\ell = w_\ell + \sum_{k=1}^{n} \mathsf{C}_{\ell k}\,\Gamma_k\,. \quad (2.3)$$

Although the condition $\mathrm{Tr}\,\varrho = 1$ sets $r_0 = 1/d$, we leave $r_0$ as a parameter to keep the same number of unknowns as for the approach using Cholesky decomposition to be described later. The important point is that the state is characterized by the Bloch vector [53–56] $r_k = \mathrm{Tr}(\varrho \Gamma_k)$, whereas $\mathsf{C}_{\ell k} = \mathrm{Tr}(\Pi_\ell \Gamma_k)$ is a $m \times n$ real matrix describing the explicit relation between the theoretical probabilities $\mathbf{p}$ and the state parameters $\mathbf{r}$.

In consequence, the inverse problem we have to solve turns out to be the linear system

$$\mathbf{p} = \mathsf{C}\,\mathbf{r}\,, \qquad (2.4)$$

where we have omitted an unessential constant term that can be incorporated into the following discussion in a straightforward way.

In presence of noise and with a finite number of copies the collected data, we will denote by $\mathbf{f}$, deviates from the expected values $\mathbf{p}$. The ultimate goal of QST is to infer the signal parameters $\mathbf{r}$ from the measured noisy data $\mathbf{f}$. A naive solution is to use the estimator

$$\widehat{\mathbf{r}}_{\mathrm{LI}} = \mathsf{C}^-\,\mathbf{f}\,, \qquad (2.5)$$

where the $\mathsf{C}^-$ stands for pseudoinverse [57–59] and the subscript LI reminds us that this is a linear inversion approach.

This $\widehat{\mathbf{r}}_{\mathrm{LI}}$ is also known as the ordinary least squares estimator [60]. As heralded before, the resulting $\widehat{\mathbf{r}}_{\mathrm{LI}}$ is no longer guaranteed to represent a positive semidefinite operator. One might ensure positivity by setting the negative eigenvalues to zero, which has been called the "quick and dirty" approach [61], although this performs poorly.

Another alternative is to use instead the generalized least-square estimator [62], defined as $\widehat{r}_{\mathrm{GLS}} = (\mathsf{X}^{-1}\mathsf{C})^+\mathsf{X}^{-1}\mathbf{f}$, where $\mathsf{X}$ is such that $\mathsf{CC}^\dagger$ is the data covariance matrix. Under the Gauss-Markov assumptions [63] it is the best linear unbiased estimator (usually known as BLUE) [64]. However, for small and medium sized data sets, a reliable estimation of the data

covariances is not possible, and then $\widehat{\mathbf{r}}_{\mathrm{LI}}$ turns out to be a handy estimator.

To circumvent these obstructions we might follow yet another route, introducing instead a semidefinite program that solves (2.4), together with the positivity constraint. The resulting estimator, denoted by $\widehat{\mathbf{r}}_{\mathrm{SDP}}$, is thus a solution of

$$\begin{aligned} &\text{minimize} \quad ||\mathbf{f} - \mathsf{C}\mathbf{r}|| \\ &\text{subject to} \quad \varrho \geq 0 \quad \text{and} \quad \mathrm{Tr}\,\varrho = 1. \end{aligned} \qquad (2.6)$$

Finally, to make our analysis complete and consider a nonlinear estimator, we also incorporate the MLE, which guarantees positivity of the resulting quantum state. Although there is a vast literature on the subject, the MLE estimate $\widehat{\varrho}_{\mathrm{MLE}}$ can be seen as the fixed point of the iterative map [65]

$$\varrho_{k+1} \leftarrow \lambda_k R \varrho_k R, \qquad (2.7)$$

with

$$R = \sum_j \frac{f_j}{p_j} \Pi_j \qquad (2.8)$$

and $\lambda_k$ is a normalization constant. The resulting Bloch-vector $\widehat{\mathbf{r}}_{\mathrm{MLE}}$ estimate is asymptotically unbiased as $f_j \to p_j$. Usually, a few thousands of iterations are needed to observe the stationary point of the map (2.7).

From a numerical point of view, an efficient way to deal with the positivity constraint is to directly decompose the density operator using the famous Cholesky factorization [66]

$$\varrho = \frac{\mathsf{A}\mathsf{A}^\dagger}{\mathrm{Tr}(\mathsf{A}\mathsf{A}^\dagger)}\,, \qquad (2.9)$$

where $\mathsf{A}$ is a complex lower triangular matrix and $\mathsf{A}^\dagger$ its Hermitian conjugate. Born's rule then turns to a set of nonlinear equations, which are rather complicated to solve. For this purpose, we adopt ML techniques.

## III. NEURAL-NETWORK ESTIMATORS

Our goal is to built a NN that links the input observed frequencies $\mathbf{f}$ to either an output true Bloch vector $\widehat{\mathbf{r}}_{\mathrm{NN}}$ or a Cholesky matrix $\mathsf{A}_{\mathrm{NN}}$. The sampled frequencies $\mathbf{f}$ serve as input to the NN, which transforms them into an output Bloch vector or a Cholesky matrix via a series of linear transformations, each followed by evaluation of some nonlinear function. The structure of such transformations is represented by neurons ordered into deep layers. More precisely, the values $\mathbf{z}^{(k)}$ of the neurons in $k$th layer are

$$\begin{aligned} \mathbf{z}^{(k)} &= f\left(\mathbf{y}^{(k)}\right), \\ \mathbf{y}^{(k)} &= \mathsf{W}^{(k-1 \to k)}\mathbf{z}^{(k-1)} + \mathbf{b}^{(k-1)}, \end{aligned} \qquad (3.1)$$

where $\mathsf{W}$ is a matrix of weights connecting neighboring layers that together with the vector of biases $\mathbf{b}$ forms a set of trainable parameters. The nonlinear activation scalar function $f$ can be
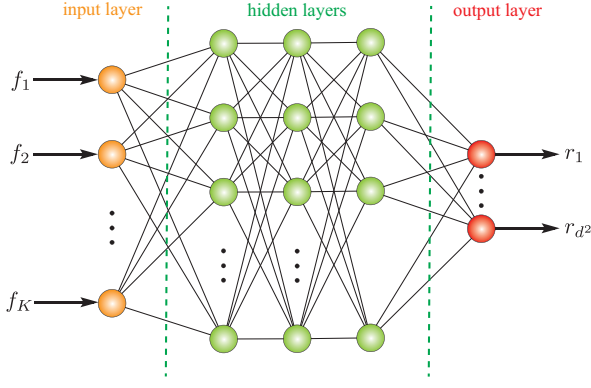
FIG. 1. Sketch of a NN for estimating the Bloch parameters or the elements of the Cholesky matrix from the sampled statistics $\mathbf{f}$. The filled blue ovals represent neurons in deep layers, representing about $2 \times 10^5$ trainable parameters. The output layer has a hyperbolic tangent as an activation function since it has the same bounds as the Bloch parameters and theelements of the Cholesky matrix. In the hidden layers, the rectified linear unit is used as an activation function. The structure of the NN is the same, independently of the dimension or specific parametric representation of quantum states.

chosen arbitrarily, depending on the problem. In our case, the rectified linear unit function $f_{\text{ReLU}}(x) := \max(0, x)$ is used in every deep layer except at the output layer, where we take $f(x) = \tanh(x)$. The hyperbolic tangent function maps real numbers into the $(-1, 1)$ interval which coincides with restriction on elements of both Bloch vector and Cholesky matrix. Symbolically, we can express the Bloch vector (and similarly the elements of the Cholesky matrix) as

$$\mathbf{r}_{\text{NN}} = f_{\text{tanh}} \circ \widetilde{\mathbf{W}}^{(n-1 \to \text{out})} \circ \cdots \circ f_{\text{ReLU}} \circ \widetilde{\mathbf{W}}^{(\text{in} \to 1)} \circ \mathbf{f}, \quad (3.2)$$

where $\widetilde{\mathbf{W}}$ is a shortcut for $\{\mathbf{W}, \mathbf{b}\}$.

The NN learns by minimizing the loss function. We chose to work with mean squared error, which takes the form

$$\mathcal{L} = \left\langle \sum_{k=0}^{n} \left| r_k - z_k^{\text{out}}(\widetilde{\mathbf{W}}, f) \right|^2 \right\rangle, \quad (3.3)$$

where $\langle \cdot \rangle$ denotes the average value in the state $\varrho$. Optimization in the NN is done by backpropagating the error. This is arguably the workhorse of most ML algorithms and definitely the standard approach in most situations, which is working with batches of data. The term and its general use in NNs was coined in [67] and a modern overview is given in textbook [68].

We minimize the value of the loss function $\mathcal{L}$ over all components of a given dataset to update weights and biases $\widetilde{\mathbf{W}}$, using a stochastic gradient-based optimization, which is of core practical importance in many fields [69]. A widely accepted algorithm is Adam [70], which is straightforward to implement, computationally efficient, and has little memory requirements. We use an improved version that incorporates Nesterov-accelerated adaptive momentum estimation (Nadam) [71], since recent results indicate that it has better performance [72]. At the step $t$, the Nadam procedure updates

parameters in the form

$$\widetilde{\mathbf{W}}_t \leftarrow \widetilde{\mathbf{W}}_{t-1} - \eta \frac{\bar{\mathbf{m}}_t}{\sqrt{\hat{\mathbf{n}}_t} + \epsilon}, \quad (3.4)$$

with

$$
\begin{aligned}
\mathbf{g}_t &\leftarrow \nabla_{\widetilde{\mathbf{W}}_{t-1}} \mathcal{L}\left(\widetilde{\mathbf{W}}_{t-1}\right), \\
\hat{\mathbf{g}} &\leftarrow \frac{\mathbf{g}_t}{1 - \prod_{j=1}^{t} \mu_j}, \\
\mathbf{m}_t &\leftarrow \mu \mathbf{m}_{t-1} + (1 - \mu) \mathbf{g}_t, \\
\hat{\mathbf{m}}_t &\leftarrow \frac{\mathbf{m}_t}{1 - \prod_{j=1}^{t+1} \mu_j}, \\
\mathbf{n}_t &\leftarrow \nu \mathbf{n}_{t-1} + (1 - \nu) \mathbf{g}_t^2, \\
\hat{\mathbf{n}}_t &\leftarrow \frac{\mathbf{n}_t}{1 - \nu^t}, \\
\bar{\mathbf{m}}_t &\leftarrow (1 - \mu_t) \hat{\mathbf{g}}_t + \mu_{t+1} \hat{\mathbf{m}}_t.
\end{aligned}
\quad (3.5)
$$

Here, $\eta$ represents the learning rate, $\mu$ the exponential decay rate for the first moment estimates $\hat{\mathbf{m}}$, $\nu$ the exponential decay rate for the weighted norm $\mathbf{g}_t^2$, and $\epsilon$ is a parameter that ensures the numerical stability of the Nadam optimization procedure. We set the numerical values $\{\eta, \mu, \nu, \epsilon\}$ to $\{0.001, 0.9, 0.999, 10^{-7}\}$.

All the above is implemented in Keras [73] and Tensorflow [74] libraries for Python. The corresponding code can be found in Ref. [75]. In every epoch, the training dataset is divided into 100 batches. The number of epochs needed to find a global minimum of the loss function varies across different deep NNs. In general, the training stopped after $400 - 2000$ epochs, depending on the dimension in which we estimate quantum states. We defined an early stopping after not finding the better minimum of the loss function in the 200 consecutive epochs. We stress that both $\widehat{\mathbf{r}}_{\text{NN}}$ and $\widehat{\mathbf{r}}_{\text{LI}}$ estimates and the ensuing quantum states are Hermitian matrices but do not incorporate the positivity constraint, in contradiction with $\widehat{\mathbf{r}}_{\text{SDP}}$ and $\widehat{\mathbf{r}}_{\text{MLE}}$.

## IV. RESULTS

Our deep NN is built as follows: The input layer is fed by observed frequencies $\mathbf{f}$ for different quantum states, followed by eight layers consisting of $(200, 180^{\otimes 2}, 160^{\otimes 4}, 100)$ neurons with the ReLU activation function. The output layer, with a hyperbolic tangent activation function, serves as an estimate of Bloch vector or the elements of Cholesky matrix.

The structure of the NN was adjusted heuristically, after having tried multiple settings with differing number of free parameters and deep layers. In terms of the distance between estimated and true quantum states, we got on average the same accuracy for a NN with two layers. However, it was observed that NNs are more likely to return parameters corresponding to positive semidefinite matrices, compared to, e.g., the LI method.

The NN sketch is presented in Fig. 1. We trained in total eight NNs, each with the same structure, for the inference of quantum states in dimensions $d = 3, 5, 7$, and 9. As our target
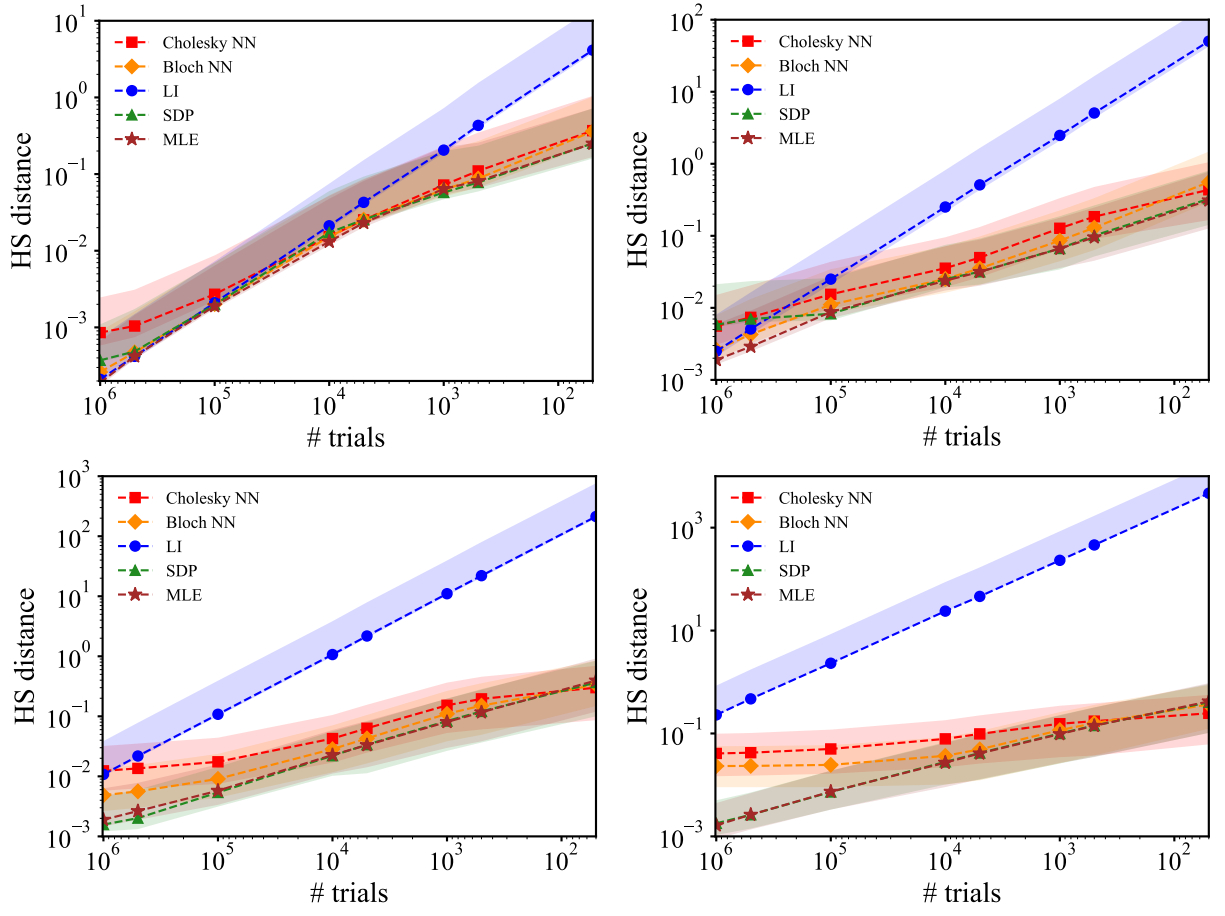
FIG. 2. Average Hilbert-Schmidt distance for different estimating strategies on the number of trials with which we sampled the true probability distribution **p** in dimension 3 (upper left panel), 5 (upper rigt panel), 7 (lower left panel), and 9 (upper right panel). The insets indicate the corresponding estimators. The average errors for both NN in the undersampled regime are of the same order for SDP and MLE in all dimensions. When using highly sampled statistics, number of trials > $10^5$, MLE starts to outperfonm both NN approaches. Confidence intervals of 80% are depicted in respective colors.

states, we use random density matrices $\varrho$ distributed according to [76]

$$\varrho = \frac{XX^\dagger}{\mathrm{Tr}(XX^\dagger)}, \quad (4.1)$$

with X pertaining to the Ginibre ensemble [77], that is, with real and imaginary parts of each matrix entry being independent normal random variables. These are implemented in Python using QuTiP [78].

As heralded in the Introduction, as our measurement scheme, we choose the square root measurement, defined by the rank-one POVM

$$\Pi_\ell = G^{-1/2} \, |\phi_\ell\rangle\langle\phi_\ell| \, G^{-1/2}, \quad G = \sum_\ell |\phi_\ell\rangle\langle\phi_\ell| \,, \quad (4.2)$$

where $|\phi_\ell\rangle$ are randomly generated Haar-distributed pure states [79] ($\ell = 0, \ldots, n$). This POVM is known to be optimal, in the sense that the measurement vectors are the closest in the squared norm to the given states [81, 82].

For each dimension, the training (validation) dataset contains $8 \times 10^5$ ($2 \times 10^5$) points. One quarter of the data in

the training dataset are probabilities $p_\ell = \mathrm{Tr}(\varrho\Pi_\ell)$ sampled with a random number of trials, ranging from $d^2$ up to $10^5$, the rest are theoretical probabilities. Each input vector, containing either theoretical or sampled statistics, corresponds to different randomly generated quantum states (4.1), whereas the measurement (4.2) is fixed for all states in each dimension.

Ideally, all data points in the training dataset should consist of only theoretical probabilities so that the NN can extract the appropriate transformation. However, in the undersampled regime, it turns out to be beneficial for the NN to see examples of the sampled statistics. In this case, with a training dataset containing only theoretical probabilities, the NN would lose the ability to predict positive matrices, while with only sampled probabilities in the training dataset, the NN would have a hard time in correctly learning the mapping from probabilities to the Bloch vector or the Cholesky matrix.

The output layer consists of true values of the Bloch vector or elements of the Cholesky matrix. All NNs are trained for at most 2000 epochs, which takes about 12 hours for every NN when estimating density matrices in dimension 9. After the training procedure, we compared estimates of quantum states
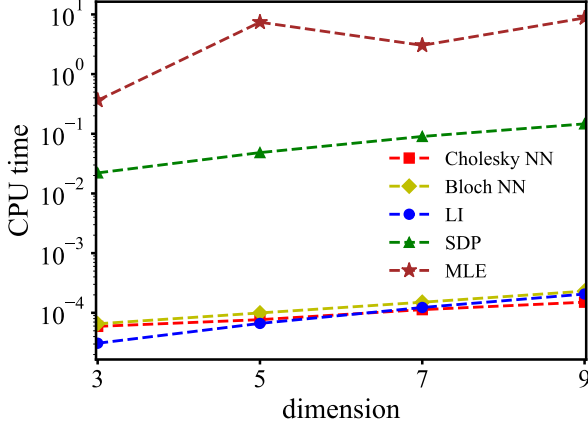
FIG. 3. Average time per state estimation for different tomographical methods, as indicated in the inset. The NN approach is about three orders faster than SDP and about four order faster than MLE.

based on NNs with standard methods, namely LI, SPD, and MLE.

Results are shown in Fig. 2. For each dimension $d$, we generated $10^3$ random density matrices and, using the same measurement scheme as for the training, we obtained a set of $10^3$ probability distributions, often called a test set. After getting the test set, we sampled each probability distribution with a number of trials depicted on the horizontal axis. Then, we used trained NNs, LI, MLE and the SDP to reconstruct density matrices from the sampled statistics. We plot the average Hilbert-Schmidt distance between true and inferred quantum states as a function of the number of trials. As we can see, NNs outperform the $\widehat{\mathbf{r}}_{LI}$ estimator and are better or give errors in the same order of magnitude as $\widehat{\mathbf{r}}_{SPD}$.

For a fixed number of trials, the NN-based Bloch vector has a better average error compared to the Cholesky one, but tends to report Hermitian matrices with nonpositive least eigenvalues. Moreover, NNs work relatively well in the undersampled regime. Interestingly, NNs show the ability to extrapolate beyond the number of samples on which they were trained. This can be appreciated in the lower Hilbert-Schmidt distance for $10^6$ trials, when only up to $10^5$ trials were used to sample the true statistics.

The combination of theoretical and sampled probabilities in the training set was balanced in such a way that the NNs work well in the undersampled regime, but also provide accurate estimates when the number of trials is high. Of course, when the number of trials goes to infinity; i.e, when working with theoretical probabilities, the $\widehat{\mathbf{r}}_{LI}$ and $\widehat{\mathbf{r}}_{MLE}$ estimators converge to the true state of the system.

Figure 3 shows an analysis of the performance of different estimators. We depict the average time per single evaluation of $\widehat{\mathbf{r}}_{LI}, \widehat{\mathbf{r}}_{SDP}, \widehat{\mathbf{r}}_{MLE}$, and both NNs. For the NNs, the times shown are only those associated with the prediction phase, not the training (which takes a lot longer). The semidefinite program infers the quantum state from sampling statistics at around $10^{-1}$

seconds. MLE turns out to be the most time-consuming procedure. Compared to linear inversion and both NN approaches, MLE predicts quantum states about $10^4$ slower. Akin to linear inversion, NN predicts unknown quantum state from the data about three orders faster compared to SPD estimates and about four orders faster compared to the MLE estimates.

Figure 4 summarizes our performance analysis. We compare the quality of $\widehat{\mathbf{r}}_{LI}$ and $\widehat{\mathbf{r}}_{NN}$ in terms of the *quantumness* of the inferred states. We show the mean of the largest negative eigenvalues and ratio of positive semidefinite quantum states among the set of estimated Hermitian matrices on the measured statistics sampled with given number of trials. Note that we have excluded results from MLE and SDP, for those estimators always reconstruct a positive matrix.

As one can see in Fig. 4, NNs can learn the positivity constraint. For example, in dimension 9, considering $\widehat{\mathbf{r}}_{LI}$, only 1% of all Hermitian matrices are positive semidefinite, compared to 17% using NN, estimated from statistics sampled with $10^6$ trials. In the undersampled regime, where the number of trials is in the order of the number of projectors, NNs in each dimension predict the higher number of positive quantum states compared to the $\widehat{\mathbf{r}}_{LI}$ estimator.

The issue with predicting quantum states with negative eigenvalues is that it is more prevalent for states that are singular (i.e., with vanishing determinant), as these are the states that sit on the boundary in the generalized Bloch representation. As such, the results in Fig. 4 depend on the purity of the states to be reported.

## V. CONCLUDING REMARKS

In summary, we have shown how NN can assist in the reconstruction of quantum states. The NN maps the input experimental data to a valid density matrix up to three orders of magnitude faster than the standard QST. This presents a significant advantage for data postprocessing during tomography. The NN learns to represent the state in a way that is well suited for the problem.

Our results confirm how some of the latest ideas from deep learning can be quite easily adapted and applied to quantum information tasks with just a few tweaks to incorporate the rules of quantum physics. This opens up a wealth of possible applications, which are the object of intense investigation.
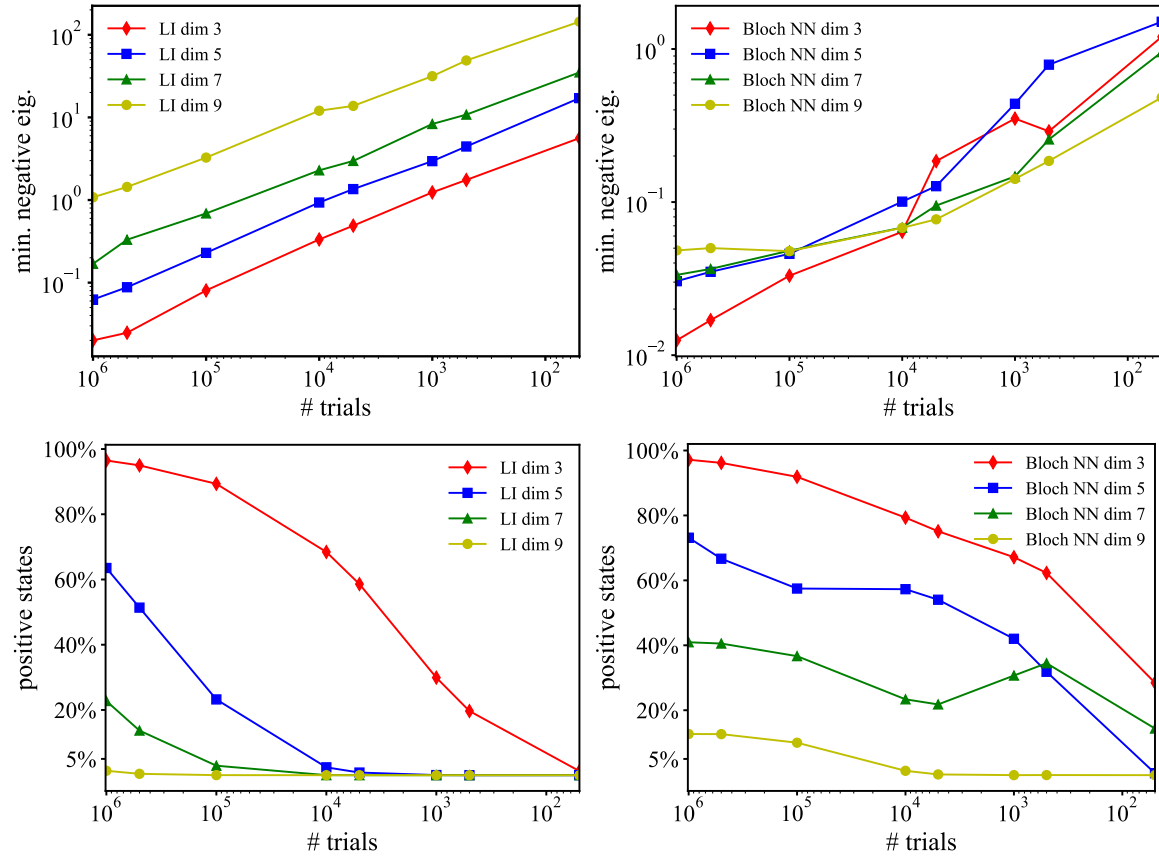
FIG. 4. Quality of estimators based on linear inversion and NN. In all the panels, red diamonds, blue squares, green triangles, and yellow circles represent the results for estimators in dimensions 3, 5, 7, and 9, respectively. The upper plots show the dependence of the most negative eigenvalue on the number of trials for those dimensions and for the LI estimator (left) and the Bloch NN estimator (right). The bottom panels depict the percentage of positive semidefinite states amongst all the reconstructed states with LI (left) and Bloch NN (right). The results are obtained from datasets containing $10^4$ statistics in each dimension.

[1] M. G. A. Paris and J. Řeháček, eds., *Quantum State Estimation*, Lect. Not. Phys., Vol. 649 (Springer, Berlin, 2004).

[2] R. O'Donnell and J. Wright, Efficient quantum tomography, in *Proc. forty-eighth annual ACM symposium on Theory of Computing* (Association for Computing Machinery, Cambridge, MA, USA, 2016) pp. 899–912.

[3] L. M. Artiles, R. D. Gill, and M. I. Guṭǎ, An invitation to quantum tomography, J. Roy. Stat. Soc. B **67**, 109 (2005).

[4] A. Tarantola, *Inverse Problem Theory and Methods for Model Parameter Estimation* (SIAM, Philadelphia, 2005) p. 348.

[5] R. Aster, B. Borchers, and C. Thurber, *Parameter Estimation and Inverse Problems*, 2nd ed. (Academic, Boston, 2013).

[6] B. Qi, Z. Hou, L. Li, D. Dong, G. Xiang, and G. Guo, Quantum state tomography via linear regression estimation, Sci. Rep. **3**, 3496 (2013).

[7] R. Blume-Kohout, Optimal, reliable estimation of quantum states, New J. Phys. **12**, 043034 (2010).

[8] C. Granade, J. Combes, and D. G. Cory, Practical bayesian tomography, New J. Phys. **18**, 033024 (2016).

[9] D. Gross, Y.-K. Liu, S. T. Flammia, S. Becker, and J. Eisert, Quantum state tomography via compressed sensing, Phys. Rev. Lett. **105**, 150401 (2010).

[10] D. Ahn, Y. S. Teo, H. Jeong, F. Bouchard, F. Hufnagel, E. Karimi, D. Koutný, J. Řeháček, Z. Hradil, G. Leuchs, and L. L. Sánchez-Soto, Adaptive compressive tomography with no a priori information, Phys. Rev. Lett. **122**, 100404 (2019).

[11] M. Cramer, M. B. Plenio, S. T. Flammia, R. Somma, D. Gross, S. D. Bartlett, O. Landon-Cardinal, D. Poulin, and Y. K. Liu, Efficient quantum state tomography, Nat. Commun. **1**, 149 EP (2010).

[12] B. P. Lanyon, C. Maier, M. Holzäpfel, T. Baumgratz, C. Hempel, P. Jurcevic, I. Dhand, A. S. Buyskikh, A. J. Daley, M. Cramer, M. B. Plenio, R. Blatt, and C. F. Roos, Efficient tomography of a quantum many-body system, Nat. Phys. **13**, 1158 (2017).

[13] Z. Hradil, Quantum-state estimation, Phys. Rev. A **55**, R1561 (1997).

[14] D. F. V. James, P. G. Kwiat, W. J. Munro, and A. G. White, Measurement of qubits, Phys. Rev. A **64**, 052312 (2001).

[15] K. Murphy, *Machine Learning: A Probabilistic Perspective* (The MIT Press, Cambridge, 2012).

[16] M. A. Nielsen, Neural networks and deep learning (2018).

[17] P. Mehta, M. Bukov, C.-H. Wang, A. G. R. Day, C. Richardson, C. K. Fisher, and D. J. Schwab, A high-bias, low-variance introduction to machine learning for physicists, Phys. Rep. **810**, 1 (2019).

[18] B. S. Rem, N. Käming, M. Tarnowski, L. Asteria, N. Fläschner, C. Becker, K. Sengstock, and C. Weitenberg, Identifying quantum phase transitions using artificial neural networks on experimental data, Nat. Phys. **15**, 917 (2019).

[19] V. Gebhart and M. Bohmann, Neural-network approach for identifying nonclassicality from click-counting data, Phys. Rev. Research **2**, 023150 (2020).

[20] C. You, M. A. Quiroz-Juárez, A. Lambert, N. Bhusal, C. Dong, A. Perez-Leija, A. Javaid, R. J. León-Montiel, and O. S. Magaña-Loaiza, Identification of light sources using machine learning, Appl. Phys. Rev. **7**, 021404 (2020).

[21] C. Harney, S. Pirandola, A. Ferraro, and M. Paternostro, Entanglement classification via neural network quantum states, New J. Phys. **22**, 045001 (2020).

[22] G. Torlai and R. G. Melko, Neural decoder for topological codes, Phys. Rev. Lett. **119**, 030501 (2017).

[23] P. Baireuther, T. E. O'Brien, B. Tarasinski, and C. W. J. Beenakker, Machine-learning-assisted correction of correlated qubit errors in a topological code, Quantum **2**, 48 (2018).

[24] S. Krastanov and L. Jiang, Deep neural network probabilistic decoder for stabilizer codes, Sci. Rep. **7**, 11003 (2017).

[25] D. Fitzek, M. Eliasson, A. F. Kockum, and M. Granath, Deep q-learning decoder for depolarizing noise on the toric code, Phys. Rev. Research **2**, 023230 (2020).

[26] E. Flurin, L. S. Martin, S. Hacohen-Gourgy, and I. Siddiqi, Using a recurrent neural network to reconstruct quantum dynamics of a superconducting qubit from physical observations, Phys. Rev. X **10**, 011006 (2020).

[27] N. Wittler, F. Roy, K. Pack, M. Werninghaus, A. S. Roy, D. J. Egger, S. Filipp, F. K. Wilhelm, and S. Machnes, Integrated tool set for control, calibration, and characterization of quantum devices applied to superconducting qubits, Phys. Rev. Applied **15**, 034080 (2021).

[28] N. Leung, M. Abdelhafez, J. Koch, and D. Schuster, Speedup for quantum optimal control from automatic differentiation based on graphics processing units, Phys. Rev. A **95**, 042318 (2017).

[29] M. Krenn, M. Malik, R. Fickler, R. Lapkiewicz, and A. Zeilinger, Automated search for new quantum experiments, Phys. Rev. Lett. **116**, 090405 (2016).

[30] A. A. Melnikov, H. Poulsen Nautrup, M. Krenn, V. Dunjko, M. Tiersch, A. Zeilinger, and H. J. Briegel, Active learning machine learns to create new quantum experiments, Proc. Natl. Acad. Sci. USA **115**, 1221 (2018).

[31] L. O'Driscoll, R. Nichols, and P. A. Knott, A hybrid machine learning algorithm for designing quantum experiments, Quantum Machine Intelligence **1**, 5 (2019).

[32] G. Torlai, G. Mazzola, J. Carrasquilla, M. Troyer, R. Melko, and G. Carleo, Neural-network quantum state tomography, Nat. Phys. **14**, 447 (2018).

[33] G. Carleo, Y. Nomura, and M. Imada, Constructing exact representations of quantum many-body systems with deep neural networks, Nat. Commun. **9**, 5322 (2018).

[34] T. Xin, S. Lu, N. Cao, G. Anikeeva, D. Lu, J. Li, G. Long, and B. Zeng, Local-measurement-based quantum state tomography via neural networks, npj Quantum Inf. **5**, 109 (2019).

[35] A. M. Palmieri, E. Kovlakov, F. Bianchi, D. Yudin, S. Straupe, J. D. Biamonte, and S. Kulik, Experimental neural network enhanced quantum tomography, npj Quantum Inf. **6**, 20 (2020).

[36] A. Melkani, C. Gneiting, and F. Nori, Eigenstate extraction with neural-network tomography, Phys. Rev. A **102**, 022412 (2020).

[37] S. Lohani, B. T. Kirby, M. Brodsky, O. Danaci, and R. T. Glasser, Machine learning assisted quantum state estimation, Mach. Learn.: Sci. Technol. **1**, 035007 (2020).

[38] Y. Liu, D. Wang, S. Xue, A. Huang, X. Fu, X. Qiang, P. Xu, H.-L. Huang, M. Deng, C. Guo, X. Yang, and J. Wu, Variational quantum circuits for quantum state tomography, Phys. Rev. A **101**, 052316 (2020).

[39] Y. Quek, S. Fort, and H. K. Ng, Adaptive quantum state tomography with neural networks, npj Quantum Inf. **7**, 105 (2021).

[40] J. Carrasquilla and G. Torlai, How to use neural networks to investigate quantum many-body physics, PRX Quantum **2**, 040201 (2021).

[41] J. Carrasquilla, G. Torlai, R. G. Melko, and L. Aolita, Reconstructing quantum states with generative models, Nat. Mach. Intell. **1**, 155 (2019).

[42] S. Lloyd and C. Weedbrook, Quantum generative adversarial learning, Phys. Rev. Lett. **121**, 040502 (2018).

[43] E. S. Tiunov, V. V. Tiunova (Vyborova), A. E. Ulanov, A. I. Lvovsky, and A. K. Fedorov, Experimental quantum homodyne tomography via machine learning, Optica **7**, 448 (2020).

[44] Z. Cai and J. Liu, Approximating quantum many-body wave functions using artificial neural networks, Phys. Rev. B **97**, 035116 (2018).

[45] P. Cha, P. Ginsparg, F. Wu, J. Carrasquilla, P. L. McMahon, and E.-A. Kim, Attention-based quantum tomography, Mach. Learn.: Sci. Technol. **3**, 01LT01 (2021).

[46] D. Kingma and M. Welling, An introduction to variational autoencoders, Found. Trends Mach. Learn. **12**, 307 (2019).

[47] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio, Generative adversarial nets, in *Advances in Neural Information Processing Systems*, Vol. 3 (Curran Associates, Montreal, 2014) pp. 2672–2680.

[48] A. Rocchetto, E. Grant, S. Strelchuk, G. Carleo, and S. Severini, Learning hard quantum distributions with variational autoencoders, npj Quantum Inf. **4**, 28 (2018).

[49] C. Zoufal, A. Lucchi, and S. Woerner, Quantum generative adversarial networks for learning and loading random distributions, npj Quantum Inf. **5**, 103 (2019).

[50] S. Ahmed, C. Sánchez Muñoz, F. Nori, and A. F. Kockum, Quantum state tomography with conditional generative adversarial networks, Phys. Rev. Lett. **127**, 140502 (2021).

[51] S. Ahmed, C. Sánchez Muñoz, F. Nori, and A. F. Kockum, Classification and reconstruction of optical quantum states with deep neural networks, Phys. Rev. Research **3**, 033278 (2021).

[52] C. W. Helstrom, *Quantum Detection and Estimation Theory* (Academic, New York, 1976).

[53] G. Kimura, The Bloch vector for $n$-level systems, Phys. Lett. A **314**, 339 (2003).

[54] R. A. Bertlmann and P. Krammer, Bloch vectors for qudits, J. Phys. A: Math. Theo. **41**, 235303 (2008).

[55] I. P. Mendaš, Classification and time evolution of density matrices for a $n$-state system, J. Math. Phys. **49**, 092102 (2008).

[56] E. Brüning, H. Mäkelä, A. Messina, and F. Petruccione, Parametrizations of density matrices, J. Mod. Opt. **59**, 1 (2012).

[57] R. Penrose, A generalized inverse for matrices, Proc. Cambridge Phil. Soc. **51**, 406 (1955).

[58] A. Ben-Israel and T. N. E. Greville, *Generalized Inverses: Theory and Applications* (Wiley, New York, 1977).

[59] S. L. Campbell and C. D. J. Meyer, *Generalized Inverses of Linear Transformations* (Dover, New York, 1991).

[60] C. Lawson and R. Hanson, *Solving Least Squares Problems* (Prentice-Hall, Englewood Cliffs, 1974).

[61] M. S. Kaznady and D. F. V. James, Numerical strategies for quantum tomography: Alternatives to full optimization, Physical Review A **79**, 022109 (2009).

[62] T. Opatrný, D. G. Welsch, and W. Vogel, Least-squares inversion for density-matrix reconstruction, Phys. Rev. A **56**, 1788 (1997).

[63] M. Hallin, Gauss–Markov theorem in statistics, in *Encyclopedia of Environmetrics* (John Wiley, 2006).

[64] S. M. Kay, *Fundamentals of Statistical Signal Processing*, Vol. 1 (Prentice Hall, Upper Saddle River, 1993).

[65] J. Řeháček, Z. Hradil, E. Knill, and A. I. Lvovsky, Diluted maximum-likelihood algorithm for quantum tomography, Phys. Rev. A **75**, 042108 (2007).

[66] D. Watkins, *Fundamentals of Matrix Computations* (Wiley, New York, 1991).

[67] D. E. Rumelhart, G. E. Hinton, and R. J. Williams, Learning representations by back-propagating errors, Nature **323**, 533 (1986).

[68] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning* (MIT Press, Cambridge, MA, 2016).

[69] S. Ruder, An overview of gradient descent optimization algorithms, arXiv:1609.04747 (2016).

[70] D. P. Kingma and J. Ba, Adam: A method for stochastic optimization (2014).

[71] T. Dozat, Incorporating Nesterov momentum into Adam, (2015).

[72] E. M. Dogo, O. J. Afolabi, N. I. Nwulu, B. Twala, and C. O. Aigbavboa, A comparative analysis of gradient descent-based optimization algorithms on convolutional neural networks, in *2018 International Conference on Computational Techniques, Electronics and Mechanical Systems (CTEMS)* (2018) pp. 92–99.

[73] F. Chollet, Keras (2015).

[74] M. A. *et al*, TensorFlow: Large-scale machine learning on heterogeneous systems (2015), software available from tensorflow.org.

[75] https://github.com/dkoutny/ ReconstructionOfQuantumStatesViaNNs.

[76] V. . A. Osipov, H. Sommers, and K. Życzkowski, Random bures mixed states and the distribution of their purity, J. Phys. A: Math. Theo. **43**, 055302 (2010).

[77] J. Ginibre, Statistical ensembles of complex, quaternion, and real matrices, J. Math. Phys. **6**, 440 (1965).

[78] J. R. Johansson, P. D. Nation, and F. Nori, QuTiP: An open-source Python framework for the dynamics of open quantum systems, Comp. Phys. Commun. **183**, 1760 (2012).

[79] K. Życzkowski, K. A. Penson, I. Nechita, and B. Collins, Generating random density matrices, J. Math. Phys. **52**, 062201 (2011).

[80] P. Hausladen and W. K. Wootters, A 'pretty good' measurement for distinguishing quantum states, J. Mod. Opt. **41**, 2385 (1994).

[81] Y. C. Eldar and G. D. Forney, On quantum detection and the square-root measurement, IEEE Trans. Inform. Theory **47**, 858 (2001).

[82] N. Dalla Pozza and G. Pierobon, Optimality of square-root measurements in quantum state discrimination, Phys. Rev. A **91**, 042334 (2015).