UAB
Universitat Autònoma
de Barcelona

Dipòsit digital
de documents
de la UAB

This is the **author's version** of the preprint:

Escrig, Josep [et al.]. *Neural network learning of one-bit protocols for qubit measurement simulation*. Preprint, 2026. 33 pàg. DOI https://doi.org/10.48550/arXiv.2607.23645

This version is available at https://ddd.uab.cat/record/338675

under the terms of the CC BY license.

Neural Network Learning of One-Bit Protocols for Qubit Measurement Simulation

Josep Escrig [1,2] Mani Zartab [3] Giulio Gasbarri [4] Estel Ferrer [1,2] Ramon Muñoz-Tapia [3] and Gael Sentís [3]

[1] *Fundació i2CAT, internet i innovació digital a Catalunya, 08034 Barcelona, Spain*
[2] *Universitat Politècnica de Catalunya, 08005 Barcelona, Spain*
[3] *Física Teòrica: Informació i Fenòmens Quàntics, Departament de Física, Universitat Autònoma de Barcelona, 08193 Bellaterra (Barcelona), Spain*
[4] *Naturwissenschaftlich-Technische Fakultät, Universität Siegen, Siegen 57068, Germany*

(Dated: July 28, 2026)

Communication complexity provides a natural framework for quantifying the classical resources required to reproduce quantum statistics. In the qubit prepare-and-measure scenario, two classical bits have been shown to be necessary and sufficient to simulate arbitrary qubit states and arbitrary quantum measurements exactly. However, this result does not exclude the possibility that restricted families of measurements may admit accurate 1-bit classical approximations. We use a neural network procedure to demonstrate that a single bit can achieve high average accuracy for specific measurement families. A performance analysis of our neural network reveals that symmetric measurements with uniformly weighted elements, such as those forming regular polyhedra, are particularly amenable to this restricted communication. By analyzing the patterns learned by the neural network, we derive an analytical protocol that is extremely accurate for finite informationally complete symmetric configurations and becomes exact in the limit of a continuous isotropic measurement.

arXiv:2607.23645v1 [quant-ph] 26 Jul 2026

I. INTRODUCTION

Understanding how much classical communication is required to reproduce quantum statistics provides an operational way to compare quantum and classical information processing [1, 2]. This perspective is especially relevant because quantum states serve as information carriers in communication protocols, cryptography, and dimension-limited information-processing tasks. In this setting, simulation costs provide a concrete benchmark for the expressive power of quantum communication: they indicate when a low-dimensional quantum system can encode correlations that would require substantially larger classical messages. The prepare-and-measure(PM) framework has been used to propose dimension witnesses [3], the certification of quantum devices [4], the analysis of communication advantages [5], and the design and verification of quantum networks [6], among other applications.

Several advances in this direction have shown that certain quantum statistics admit surprisingly efficient classical simulations. In Ref. [7], the authors introduced a coding scheme for the classical teleportation of a qubit, requiring on average only 2.19 bits of classical information to reproduce the statistics of projective measurements. Toner and Bacon [8] subsequently showed that shared classical randomness together with a single bit of communication suffices to reproduce the correlations of local projective measurements on a maximally entangled pair. In the PM scenario, they further showed that two classical bits are sufficient to simulate the statistics of arbitrary projective measurements on a qubit. Degorre *et al.* [9] later observed that the communication step in the above protocol can be replaced by a source of biased shared randomness depending on the qubit state, thereby identifying a key ingredient underlying classical simulations of quantum statistics. Building on these ideas, Renner *et al.* [10] extended the simulation of qubit statistics from projective to generalized measurements, proving that any qubit PM scenario can be simulated exactly with a worst-case communication cost of two bits. Extensions to higher-dimensional systems have also been investigated [11–14], including approximate simulation protocols and analyses of average communication cost. Ultimately, the existence of an exact classical simulation protocol for PM scenarios in dimension $d \geq 3$ using a finite amount of classical communication remains open.

The 2-bit worst-case simulation of arbitrary qubit PM statistics, however, does not preclude more economical descriptions in less demanding regimes. One natural relaxation is to move from worst-case to average communication cost. Indeed, recent work has shown that, by encoding the messages sent by Alice, qubit PM statistics can be simulated with an average communication cost of 1.89 bits [14]. This suggests that the apparent 2-bit barrier may be a consequence of demanding a uniform worst-case guarantee over all state preparations and measurements, and that more efficient simulations may be possible once the relevant operational figure of merit is adapted.

A second route toward lower communication cost is to restrict the family of quantum states and/or measurements to be simulated. This is particularly natural in PM scenarios, where practical communication tasks often involve structured ensembles of preparations and measurements rather than the full set of qubit statistics. From this perspective, the central question becomes which physically or operationally meaningful subsets of qubit statistics already admit an even more efficient classical simulability. Identifying such subsets would help clarify which features of quantum PM experiments are responsible for their classical communication cost.

Further motivation comes from recent numerical evidence in the Bell scenario. In Ref. [15], neural network methods were used to search for 1-bit classical protocols simulating the statistics of local projective measurements on bipartite entangled two-qubit states. Strikingly, although one bit of communication is known to be insufficient in full generality, the authors found that 1-bit protocols approximate the target quantum correlations extremely well across the tested instances, and no explicit counterexample was identified by their search. This suggests that one bit of communication can already capture a surprisingly large portion of quantum statistics, extending the intuition provided by the Toner–Bacon protocol beyond the maximally entangled singlet case [8].

These observations motivate the search for 1-bit simulation protocols in PM scenarios. In this work, we focus on restricted families of measurements and ask whether their qubit statistics can be reproduced, or accurately approximated, using only a single bit of classical communication. To this end, we adopt a neural network(NN)- procedure, in the spirit of Ref. [15], using the network as a tool to discover candidate protocols rather than imposing a fixed analytical ansatz from the outset. When successful, such searches provide evidence for hidden structure in the corresponding family of measurements and point toward compact classical descriptions of their quantum statistics. Thus, our goal is to map regimes of 1-bit-simulability in qubit PM scenarios and to use protocols, discovered by NN-procedure, as evidence for new analytically tractable classical simulations.

Following this approach, we first train a neural network on randomly generated PM instances and use it to identify structured families of measurements that are strong candidates for 1-bit simulability. The families singled out by the optimization are highly-symmetric POVMs whose elements are distributed close to uniformly on the Bloch sphere. By inspecting the behavior learned by the network, we then extract an explicit analytical protocol, thereby moving from a black-box numerical search to a concrete classical simulation strategy.

Our analysis shows that the extracted 1-bit protocol, while it is not exact for arbitrary finite POVMs, yields highly accurate approximations to the quantum probabilities for informationally-complete symmetric POVMs, and the approximation improves as the number of outcomes increases. In the continuous limit, corresponding to the covariant qubit POVM, the protocol becomes exact, placing this measurement within the class of exactly 1-bit-simulable PM resources. These results provide evidence that symmetry can substantially reduce the communication needed to simulate quantum statistics, and they illustrate how neural network searches can help uncover analytical structure in PM simulation problems.

This paper is structured as follows. In Section II we introduce the theoretical background and definitions used throughout this paper. In Section III we describe the NN-procedure used to learn quantum statistics under the restrictions of classical shared randomness and one bit of classical communication, and in Section IV we present the numerical results. In Section V we infer the classical 1-bit protocol and study its properties. We end with the conclusions and include the technical details in the Appendices.

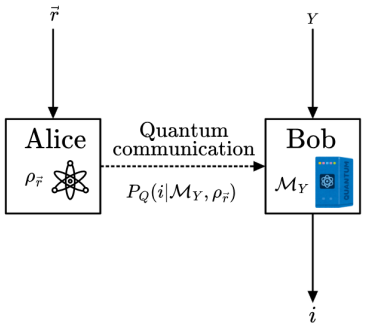

Figure 1. Quantum PM scenario. Alice prepares the qubit state $\rho_{\vec{r}}$ and sends it to Bob, who performs the POVM $\mathcal{M}_Y$ and reports the outcome i.

II. BACKGROUND

A general m-outcome quantum measurement is described by a POVM, *i.e.*, a set of positive semidefinite operators $\mathcal{M} := \{M_i\}_{i=1}^m$ satisfying the completeness condition $\sum_{i=1}^m M_i = \mathbb{1}$. The probability of obtaining outcome i when applying $\mathcal{M}$ on a generic state ρ is given by the Born rule $P_Q(i|\rho,\mathcal{M}) = \mathrm{Tr}(M_i\rho)$. For qubits, each POVM element can be written in the Bloch representation as $M_i = p_i\big(\mathbb{1}+\vec{y}_i\cdot\vec{\sigma}\big)$, where $p_i \geq 0$, $\sum_i p_i = 1$, and $|\vec{y}_i| \leq 1$. Similarly, we can express a qubit state as $\rho = (\mathbb{1}+\vec{r}\cdot\vec{\sigma})/2$.

The probability $P_Q(i|\rho,\mathcal{M})$ then takes the form

$$P_Q(i|\rho,\mathcal{M}) = p_i\big(1+\vec{y}_i\cdot\vec{r}\big). \tag{1}$$

Let us now introduce the quantum PM scenario. In this setup, Alice prepares a qubit state $\rho_{\vec{r}}$, represented by its Bloch vector $\vec{r}$, and sends it to Bob through a quantum communication channel. Bob then performs a POVM $\mathcal{M}_Y$ specified by $Y := (p_i,\vec{y}_i)_{i=1}^m$, and reports the outcome i with probability given by (1). The scenario is illustrated in Fig. 1.

A classical simulation replaces the transmitted qubit by a classical message, assisted by shared randomness. Alice receives the state description $\vec{r}$, Bob receives the measurement description Y, and they share a random variable independent of both. Alice sends a classical messace c to Bob. Bob, uses c, Y, and the share randomness to generate an outcome i. The protocol exactly simulates the quantum PM scehnario if, for all states and measurements, the resulting classical probability $P_C(i|\vec{r},Y)$ coincides with the quantum one *i.e*,

$$P_C(i|\vec{r},Y) = P_Q(i|\rho_{\vec{r}},\mathcal{M}_Y) \quad \forall i,\vec{r},Y. \tag{2}$$

If the equality holds only up to a chosen error metric, evaluated on a spceific class of states and measurements, we will refer the protocol as an approximate simulation.

Renner *et al.* [10] showed that the qubit PM scenario admits an exact classical simulation with two bits of communication, and that two bits are necessary in the worst case. For comparison with the results of the following sections, we briefly recall their protocol. Without loss of generality, it is enough to consider pure states, $|\vec{r}| = 1$, since mixed states can be decomposed as a convex combination of pure states, and the corresponding classical randomness can be incorporated into the shared randomness of the protocol. Similarly, for measurements, it is enough to consider POVM elements proportional to rank-one projectors, (*i.e.*, $|\vec{y}_i| = 1, \forall i$), since general qubit POVMs can be obtained from rank-one refinements by classical post-processing. The protocol is illustrated in Fig. 2 and proceed as follows:

1. Alice is given the Bloch vector $\vec{r}$, and Bob is given the POVM specified by Y. In addition, they share two random unit vectors $\vec{\lambda}_1, \vec{\lambda}_2 \in \mathbb{S}^2$.

2. Alice computes the classical bits

$$c_1 = H(\vec{\lambda}_1 \cdot \vec{r}), \qquad c_2 = H(\vec{\lambda}_2 \cdot \vec{r}), \tag{3}$$

   and sends them to Bob, where $H(x)$ is the Heaviside step function.

3. Bob defines the effective hidden variables

$$\vec{\lambda}'_i = (2c_i - 1)\vec{\lambda}_i, \tag{4}$$

   that is, he flips $\vec{\lambda}_i$ whenever $c_i = 0$.

4. Bob selects $\vec{y}_i$ with probability p_i. He then sets $\vec{\lambda}' = \vec{\lambda}'_1$ if $|\vec{\lambda}'_1 \cdot \vec{y}_i| \geq |\vec{\lambda}'_2 \cdot \vec{y}_i|$, and $\vec{\lambda}' = \vec{\lambda}_2$ otherwise.

5. Bob reports the outcome i with probability

$$P_B(i|\vec{\lambda}', Y) = \frac{p_i(\vec{y}_i \cdot \vec{\lambda}')\, H(\vec{y}_i \cdot \vec{\lambda}')}{\sum_{j=1}^{m} p_j(\vec{y}_j \cdot \vec{\lambda}')\, H(\vec{y}_j \cdot \vec{\lambda}')}. \tag{5}$$

As shown in Ref. [10], the statistics generated by this protocol can be written as

$$P_C(i|\vec{r}, Y) = \int_{\mathbb{S}^2} d\vec{\lambda}\, \rho(\vec{\lambda}|\vec{r}, Y)\, P_B(i|\vec{\lambda}, Y), \tag{6}$$

where $d\vec{\lambda} = d\Omega/(4\pi)$ denotes the normalized uniform measure on the sphere. The effective distribution $\rho(\vec{\lambda}|\vec{r}, Y)$, which encodes the statistical effect of the flipping and comparison steps, is given by:

$$\rho(\vec{\lambda}|\vec{r}, Y) = 8H(\vec{r} \cdot \vec{\lambda}) \sum_{j=1}^{m} p_j(\vec{y}_j \cdot \vec{\lambda})\, H(\vec{y}_j \cdot \vec{\lambda}). \tag{7}$$

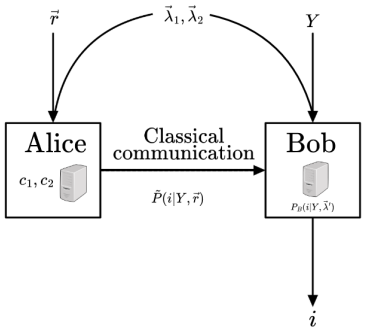

Figure 2. 2-bit classical simulation protocol of Ref. [10] for the PM scenario. Alice and Bob share $\vec{\lambda}_1, \vec{\lambda}_2 \in \mathbb{S}^2$, Alice sends the bits c_1, c_2, and Bob outputs i according to the protocol.

III. NEURAL NETWORK LEARNING PROCEDURE

Another important result of Ref. [10] is that no classical protocol using fewer than two classical bits is able to reproduce the statistics of all possible quantum input setups in this setting.

However, this does not exclude the possibility that restricted families of measurements may admit simpler classical simulations. *A priori*, it remains unclear which structural properties of a POVM may enable such a reduction, and which protocols could possibly achieve it. Our aim is

to use a NN-procedure, as a data-driven probe for identifying such regimes.

For a given POVM $\mathcal{M}_Y$, Alice and Bob share a random unit vector $\vec{\lambda} \in \mathbb{S}^2$, sampled uniformly from the Bloch sphere. Alice receives the input state $\rho_{\vec{r}}$ and sends to Bob the single bit $c = H(\vec{\lambda} \cdot \vec{r})$. The neural network takes as input the pair $(c, \vec{\lambda})$ and outputs a probability distribution over the outcomes of the POVM, $P_{\mathrm{NN}}(i|c, \vec{\lambda}, Y)$. This output plays the role of Bob's response function. The dependence on Y is implicit in the trained network: for each POVM $\mathcal{M}_Y$ considered, a separate network is trained. The statistics generated by the NN-procedure are obtained by averaging the response function over the shared randomness. For a fixed input state $\rho_{\vec{r}}$, the averaged prediction is

$$P_{\mathrm{NN}}(i|\vec{r}, Y) = \int_{\mathbb{S}^2} d\vec{\lambda}\, P_{\mathrm{NN}}\big(i|H(\vec{\lambda} \cdot \vec{r}), \vec{\lambda}, Y\big). \tag{8}$$

In practice, this integral is estimated by Monte Carlo sampling.

The training set consists of M Haar-random pure input states $\{\vec{r}^{(j)}\}_{j=1}^{M}$. For each state $\vec{r}^{(j)}$, we sample N independent shared random vectors $\{\vec{\lambda}^{(j,k)}\}_{k=1}^{N}$ and define the corresponding bits $c^{(j,k)} = H\big(\vec{\lambda}^{(j,k)} \cdot \vec{r}^{(j)}\big)$. For each state j and outcome i, the Monte Carlo average of

the neural network response is

$$\bar{p}_i^{(j)} = \frac{1}{N}\sum_{k=1}^{N} P_{\mathrm{NN}}\big(i|c^{(j,k)}, \vec{\lambda}^{(j,k)}, Y\big). \tag{9}$$

The target probability is the Born probability

$$P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y) = p_i\big(1 + \vec{y}_i \cdot \vec{r}^{(j)}\big). \tag{10}$$

The network is trained by minimizing the mean squared error(MSE) between the averaged neural network probabilities and the corresponding Born probabilities,

$$\mathcal{L}_{\mathrm{MSE}} = \frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{m}\Big[\bar{p}_i^{(j)} - P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y)\Big]^2. \tag{11}$$

We use MSE as the training loss because it is smooth and penalizes large deviations more strongly, which is convenient for numerical optimization.

To evaluate the accuracy of the trained network, we use the mean absolute error(MAE),

$$\mathrm{MAE} = \frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{m}\Big|\bar{p}_i^{(j)} - P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y)\Big|. \tag{12}$$

Unlike the MSE, the MAE directly quantifies the average absolute discrepancy between the predicted and target probabilities, providing a more standard figure of merit for comparing the accuracy of different protocols.

Two important remarks are in order. First, for each POVM whose statistics we wish to simulate, the neural network must be retrained. Nevertheless, as we show below, the analysis of the resulting network responses allows us to infer a POVM-independent 1-bit protocol. Second, because the MAE is evaluated over randomly sampled states and shared random vectors, it quantifies an *average* simulation accuracy. The training methodology and network architecture are described in more detail in Appendix A.

IV. NUMERICAL PERFORMANCE

We now analyze the numerical performance of the neural network, introduced in the previous section, with a focus on how it varies depending on the characteristics of the POVMs.

Throughout the numerical experiments, in this section, unless otherwise stated, we fix the parameters to $N = 4000$ shared randomness samples and $M = 2000$ input states during the training stage and $N = 4000$, $M = 1000$ during the test stage. We found these values to provide a good compromise between numerical accuracy and computational cost. Additional implementation details are given in Appendix A.

We first study the NN-procedure on randomly generated POVMs with different numbers of outcomes. Random POVMs can be sampled in several inequivalent ways, depending on the chosen measure over the space of effects [16]. In the present work, we use a constructive procedure based on random Bloch-sphere directions followed by a numerical search for nonnegative weights satisfying the POVM completeness condition.

Fig. 3 summarizes the MAE obtained in 100 experiments over 100 different randomly generated POVMs with numbers of outcomes ranging from $m = 3$ to $m = 8$.

The figure reports the median, mean, and interquartile range of the MAE values obtained with the NN-procedure. For comparison, we also show the MAE obtained by direct finite-sample estimation of the Born probabilities. This Born rule curve should be interpreted as a sampling baseline: it quantifies the error due to finite sampling when the target probabilities are known.

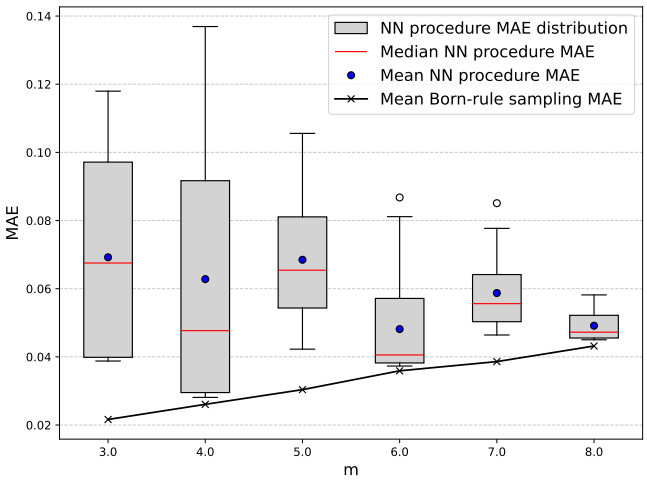

Figure 3. MAE of the probabilities calculated with the NN-procedure for POVMs with different numbers of elements m, compared with the mean MAE sampled directly from the Born rule.

The most immediate observation is that the MAE of the NN-procedure is systematically larger than the MAE obtained by direct Born rule sampling. Nevertheless, the results also show that some POVM instances are reproduced much more accurately than others. In particular, for certain values of m the best cases approach the Born rule baseline rather closely. This suggests that the performance of the 1-bit protocol may depend on structural properties of the measurement.

By direct inspecting the well-performing POVMs, we observe that the uniformity in the weight $\{p_i\}_{i=1}^m$ distribution across the POVM elements seems to play an important role. To investigate this hypothesis, we plot the MAE as a function of the standard deviation of the POVM weights. A smaller standard deviation corresponds to a more uniform distribution of the weights, with Std $= 0$ corresponding to the equal-weight case $p_i = 1/m$ for all i. Fig. 4 shows that the accuracy of the NN-procedure improves significantly as this standard deviation decreases, for any number of outcomes m. The

most accurate cases are concentrated near the standard deviation Std = 0.

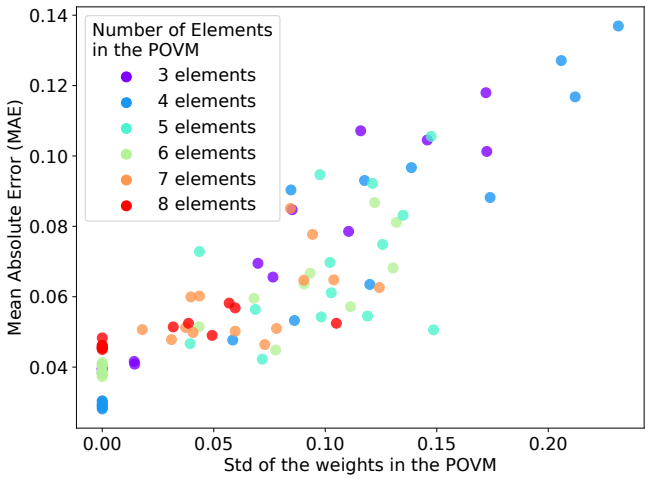

Figure 4. MAE of the probabilities obtained from the NN-procedure for different POVMs as a function of Std of the weights $\{p_i\}_{i=1}^m$ of the elements of each POVM. POVMs with different numbers of elements m are represented with different colors.

Motivated by this numerical trend, we focus from this point onwards on POVMs with equal weights, *i.e.*, $p_i = 1/m\,, \forall i$. In the rank-one case, these measurements can be written as $\left\{\frac{2}{m}\,|\psi_i\rangle\langle\psi_i|\right\}_{i=1}^m$, where each $|\psi_i\rangle$ is a normalized vector. The POVM normalization condition is therefore $\sum_{i=1}^m |\psi_i\rangle\langle\psi_i| = (m/2)\mathbb{1}$, which implies that the vectors $\{|\psi_i\rangle\}_{i=1}^m$ form a unit-norm tight frame. We refer to this class as *equal-trace rank-one POVMs* or e-POVMs. Accordingly, they have also been termed unit-norm tight-frame POVMs in the literature [17, 18]. Geometrically, under the Bloch-sphere representation, the pure states associated with an e-POVM correspond to m points on the sphere whose centroid lies at the origin. Thus, the POVM normalization condition admits a simple interpretation as a balance condition on the corresponding Bloch-sphere configuration $\sum_{i=1}^m \vec{y}_i = \vec{0}$, with $|y_i| = 1$, where the POVM elements take the form $M_i = \frac{1}{m}\,(\mathbb{1} + \vec{y}_i \cdot \vec{\sigma})$.

When needed, we refer to the informationally complete(IC) members of this class as eIC-POVMs.

Within the class of e-POVMs, it is natural to examine highly regular configurations [19] of Bloch vectors. We therefore compare the performance of the NN-procedure on regular polygons and regular polyhedra inscribed in the Bloch sphere. Both classes have equal weights and balanced Bloch vectors, but they differ in an important respect: regular polygons lie in a plane and are therefore not informationally complete, whereas regular polyhedral configurations span $\mathbb{R}^3$ and define eIC-POVMs. The set of regular polygons and polyhedra inscribed in the Bloch sphere[1]

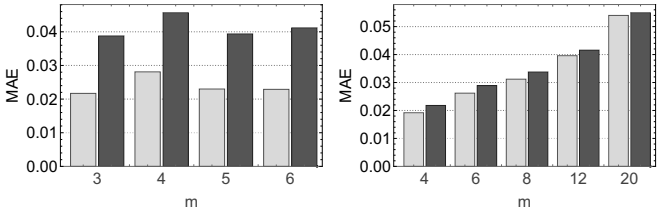

Figure 5. MAE of the NN-procedure (dark gray) and of the Born rule (light gray) for various e-POVMs with m outcomes, taking 4000 samples (value of N) and averaging over 1000 different states (value of M). In the left plot, we have polygon configurations, while on the right plot we have regular polyhedra.

Fig. 5 compares the MAE of the NN-procedure with the MAE obtained by direct sampling from the Born rule for several such configurations. The comparison reveals a clear qualitative distinction. For polygonal configurations such as the triangle, square, pentagon, and hexagon, the MAE of the NN-procedure remains noticeably above the Born rule baseline. By contrast, for regular polyhedra configurations, the gap between the NN-procedure and the Born rule baseline is significantly smaller. This suggests that the symmetric three-dimensional configuration of measurement vectors, representing an informationally complete POVM plays an important role.

Overall, the numerical results support two main observations. First, 1-bit NN-procedure can reproduce the statistics of certain POVMs with high average accuracy, even though one bit cannot simulate the full qubit PM scenario exactly. Second, this performance is not uniform across all measurements. It strongly depends on the distribution of the POVM weights. In particular, the protocol performs best for

eIC-POVMs in highly symmetric configurations.

V. INFERRED CLASSICAL 1-BIT PROTOCOL

The numerical results of the previous section indicate that the NN-procedure performs particularly well for eIC-POVMs whose Bloch vectors form highly isotropic three-dimensional configurations. This motivates the search for a transparent analytical description of the behavior learned by the neural network.

We first inspect the neural-network response and extract a simple analytical rule. We then define the corresponding 1-bit protocol and study its numerical performance for finite eIC-POVMs. Finally, we show that the protocol, although not exact for finite eIC-POVMs

[1] These structures also belong to the class of "highly-symmetric POVMs" defined in Ref. [19].

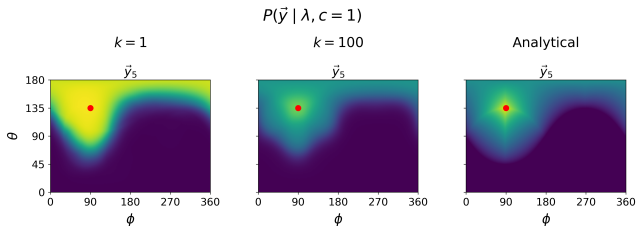

Figure 6. Conditional response probabilities for outcome $i = 5$ of the six-direction Cartesian, or octahedral, eIC-POVM, conditioned on a transmitted bit $c = 1$. The left panel shows the output of a single trained neural network ($k = 1$), the center panel shows the output averaged over an ensemble of $k = 100$ independently trained networks, and the right panel shows the response of the analytical 1-bit protocol in (17). The probabilities are represented as a function of the shared random vector $\vec{\lambda}$, parametrized by the polar angles (ϕ, θ). The red marker indicates the Bloch vector $\vec{y}_5$ associated with the selected outcome.

in general, becomes exact for regular polyhedral configurations when the input state aligns with one of the measurement outcomes, and also, in the continuous isotropic limit.

A. Extraction and definition of the 1-bit protocol

To identify the structure learned by the NN-procedure, we inspect the neural-network output for a representative eIC-POVM.

We use the six-direction Cartesian configuration, namely the POVM whose Bloch vectors are $\pm\hat{x}$, $\pm\hat{y}$, and $\pm\hat{z}$. This is the octahedral configuration on the Bloch sphere and corresponds to the union of the three mutually unbiased qubit bases.

Fig. 6 shows the conditional response of the neural network for the outcome associated with $+\hat{z}$, and for the transmitted bit $c = 1$, $P_{\rm NN}(i = 5|c = 1, \vec{\lambda}, Y)$, as a function of the shared vector $\vec{\lambda}$, parametrized by the polar angles (θ, ϕ). The red dot indicates the polar direction of the POVM element $\vec{y}_5$. The left panel shows the response of a single trained neural network. Although the response is smooth and shows a clear geometric dependence on $\vec{\lambda}$, the fluctuations associated with a particular training run prevent sharp boundaries between the decision regions, making difficult to infer a simple analytical rule directly. To reduce these network-dependent fluctuations, we average the response probabilities over 100 independently trained networks. The resulting ensemble-averaged response is shown in the central panel. The corresponding response maps for all outcomes and for both values of the communicated bit are provided in Appendix E.

The ensample-averaged response reveals a simple pattern. For $c = 1$, the probability associated with the POVM element $\vec{y}_i$ is approximately zero on the hemisphere opposite to $\vec{y}_i$, and increases approximately linearly with the scalar product $\vec{y}_i \cdot \vec{\lambda}$ on the hemisphere centered on $\vec{y}_i$. This suggests the proportionality rule

$$P_{\rm NN}(i|c = 1, \vec{\lambda}, Y) \propto (\vec{y}_i \cdot \vec{\lambda})\, H(\vec{y}_i \cdot \vec{\lambda}), \tag{13}$$

For $c = 0$, the same pattern is observed after the flip of the hidden variable λ, *i.e.*, $\vec{\lambda} \mapsto -\vec{\lambda}$, leading to

$$P_{\rm NN}(i|c = 0, \vec{\lambda}, Y) \propto (-\vec{y}_i \cdot \vec{\lambda})\, H(-\vec{y}_i \cdot \vec{\lambda}). \tag{14}$$

Equations (13) and (14) suggest that the dependence on the communicated bit can be described by a sign flip of the shared vector λ. This motivates the following 1-bit protocol:

1. Alice receives the description of a qubit state $\rho_{\vec{r}}$, represented by its Bloch vector $\vec{r}$, and Bob receives the description of an e-POVM $\mathcal{M}_Y$ specified by Y (Remember, $p_i = 1/m\ \forall i$). In addition, they share a random unit vector $\vec{\lambda} \in \mathbb{S}^2$, uniformly distributed over the Bloch sphere.

2. Alice computes the bit

$$c = H(\vec{\lambda} \cdot \vec{r}), \tag{15}$$

and sends it to Bob.

3. Bob defines the effective hidden variable

$$\vec{\lambda}' = (2c - 1)\vec{\lambda}. \tag{16}$$

Thus, Bob flips the shared vector whenever $c = 0$.

4. Bob reports the outcome i with probability

$$P_{1-\rm bit}(i|\vec{\lambda}', Y) = \frac{(\vec{y}_i \cdot \vec{\lambda}')\, H(\vec{y}_i \cdot \vec{\lambda}')}{\sum_{j=1}^{m} (\vec{y}_j \cdot \vec{\lambda}')\, H(\vec{y}_j \cdot \vec{\lambda}')}. \tag{17}$$

The right panel of Fig. 6 shows the response of (17) for the octahedral outcome associated to $i = 5$. Its geometric structure closely reproduces the ensamble-averaged neural network response: the probability of outcome i is maximal at the Bloch vector $\vec{y}_i$ and concentrated in the hemisphere defined by it and increase with the positive overlap $\vec{y}_i \cdot \vec{\lambda}'$.

B. Numerical performance

We tested the analytical protocol in (17) on six e-POVMs: the tetrahedron ($m = 4$), the octahedron ($m = 6$), the hexahedron ($m = 8$), the icosahedron ($m = 12$), the dodecahedron ($m = 20$), and the icosidodecahedron[2] ($m = 30$). The probabilities generated by

[2] A quasi-regular polyhedron

the protocol are compared with the target Born probabilities and with the probabilities obtained by direct finite sampling from the Born rule.

In this section, we quantify the discrepancy using the Kullback-Leibler divergence(KLD), averaged over randomly chosen input states. In contrast with the MAE used in Sec. III to define the training loss and measures the average absolute deviation between probabilities, the KLD is more sensitive to the full shape of the probability distribution, and in particular to relative errors in outcomes with small probabilities. For this reason, the numerical values obtained from the KLD can differ significantly from those obtained with the MAE. While both measures lead to neural networks that performed similarly both in training and testing, the extra sensitivity of the KLD is useful to us now to closely compare the complete output distributions generated by different protocols. The results are shown in Fig. 7.

For small sample sizes, the KLD is dominated by statistical fluctuations, and both curves decrease as the number of samples increases. The Born-rule sampling baseline continues to approach zero, whereas the KLD of the 1-bit protocol eventually reaches a nonzero plateau. This plateau represents the intrinsic bias of the protocol with respect to the Born probabilities and shows that the finite-POVM simulation is not exact.

The agreement is nevertheless very accurate for all regular polyhedral configuration studied. Moreover the intrinsic discrepancy becomes small for the POVMs with larger and more isotropically distributed sets of Bloch vectors, such as the dodecahedral and the icosidodecahedral configurations.

We also compare the analytical 1-bit protocol directly with the neural network output. Fig. 8 shows the mean KLD of the analytical protocol of a single trained neural network, an ensemble average over 10 independently trained neural networks, and direct sampling from the Born rule.

The analytical protocol gives a smaller KLD than both the single-network response and the ensemble-average neural network response, showing that the explicit rule in (17) better reproduces the Born statistics.

These results indicate that, while the neural network learns the correct qualitative structure, the explicit analytical rule provides a more accurate realization of this structure at the level of the full output distribution. Note also that the averaged neural network response has a similar mean KLD than that of a single network. Despite not improving statistical accuracy, averaging over independently trained networks was key to distill the geometric response pattern condensed in Eqs. (13) and (14).

Fig. 8 also shows that the KLD of the 1-bit protocol approaches the KLD obtained by direct sampling from the Born rule as the number of POVM elements increases. This trend is consistent with Fig. 7: at 10^5 samples, the 1-bit protocol is already extremely close to the Born rule sampling benchmark, especially for the dodecahedral and icosidodecahedral POVMs. This supports the interpretation that regular polyhedral eIC-POVMs with more elements provide better finite approximations to the isotropic continuum construction discussed below.

In conclusion, the 1-bit protocol improves on the numerical performance of the trained NN-procedures while

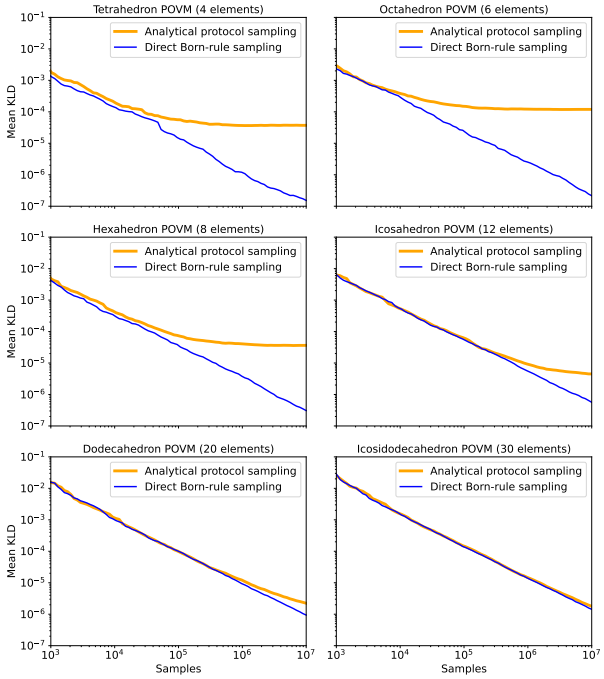

Figure 7. KLD of the analytical 1-bit protocol and of direct Born-rule sampling for six regular eIC-POVMs: tetrahedron, octahedron, hexahedron, icosahedron, dodecahedron, and icosidodecahedron ($m = 4, 6, 8, 12, 20, 30$, respectively). Each curve is averaged over $M = 50$ randomly sampled input states. The blue curves represent the Born-rule sampling baseline, while the yellow curves represent the analytical 1-bit protocol.

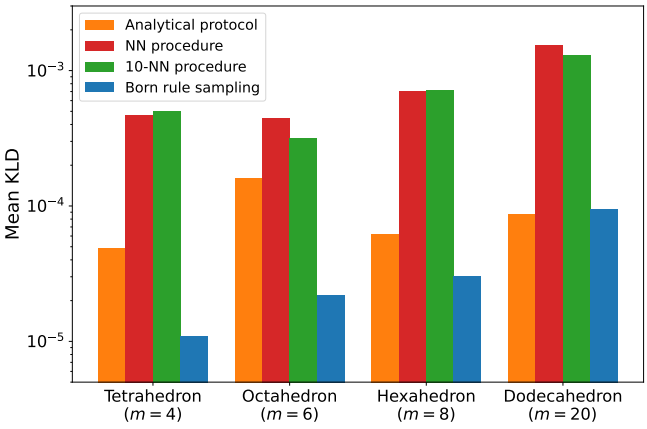

Figure 8. Mean KLD for the analytical protocol, the NN-procedure, an ensemble of ten NN-procedures, and direct sampling from the Born rule. For each configuration, the mean is computed over $M = 20$ random input states $\vec{r}$, using $N = 10^5$ samples per state.

retaining the same qualitative feature: 1-bit communication reproduces the target statistics with high accuracy for eIC-POVMs.

C. The continuous POVM limit

The analytical expression for the probability distribution over outcomes of the 1-bit protocol takes the following form. The probability density of the shared vector $\vec{\lambda}$ after step 3 of the protocol [cf. (16)] is

$$\rho(\vec{\lambda}|\vec{r}) = 2H(\vec{r}\cdot\vec{\lambda})\,, \tag{18}$$

and the probability of obtaining outcome i given a state with Bloch vector $\vec{r}$ reads [cf. (6)]

$$P_{1-\mathrm{bit}}(i|\vec{r},\,Y) = 2\int d\vec{\lambda}\frac{H(\vec{r}\cdot\vec{\lambda})\,(\vec{y}_i\cdot\vec{\lambda})\,H(\vec{y}_i\cdot\vec{\lambda})}{\sum_{j=1}^{m}(\vec{y}_j\cdot\vec{\lambda})\,H(\vec{y}_j\cdot\vec{\lambda})}, \tag{19}$$

where recall that $d\vec{\lambda} = d\Omega/4\pi$ is the Haar measure in $\mathbb{S}^2$. Although (19) provides a remarkably accurate approximation to the eIC-POVM probabilities, it is not exact in general. This can be seen from a simple example. For instance, in the octahedral configuration, consider the probability of obtaining the outcome $+\hat{z}$ when Alice's state points along the diagonal in the x-z plane. In this case, one can show analytically that the quantum probability and the probability generated by the 1-bit protocol have a discrepancy of $\mathcal{O}(10^{-3})$. At the other extreme, if Alice's state happens to align with one of Bob's measurement outcomes, *i.e.*, $\vec{r} = \vec{y}_i$ for some i, one can show that the 1-bit protocol is exact for all outcomes i (See Appendix C.).

This notwithstanding, in the limit of a continuous uniform POVM $\mathcal{M} = \{2\,|\vec{m}\rangle\langle\vec{m}|\,d\vec{m}\}_{\vec{m}\in\mathbb{S}^2}$, $d\vec{m} = d\Omega/4\pi$, (19) yields the exact quantum probability density, which reads

$$P_Q(\vec{m}|\rho_{\vec{r}},\mathcal{M}) = 1 + \vec{r}\cdot\vec{m}\,. \tag{20}$$

Indeed, by rotational symmetry the denominator in (19) becomes the number

$$\int d\vec{m}\,(\vec{m}\cdot\vec{\lambda})\,H(\vec{m}\cdot\vec{\lambda}) = \frac{1}{4}. \tag{21}$$

Using $x\,H(x) = (x+|x|)/2$, we get

$$P_{1-\mathrm{bit}}(\vec{m}|\vec{r},Y) = 4\int d\vec{\lambda}H(\vec{r}\cdot\vec{\lambda})\left[\,(\vec{m}\cdot\vec{\lambda}) + |\vec{m}\cdot\vec{\lambda}|\right]. \tag{22}$$

The first term is proportional to $\vec{r}\cdot\vec{m}$, while the second is again a number, *i.e.*, $P_{1-\mathrm{bit}}(\vec{m}|\vec{r},Y) = \gamma_1 + \gamma_2\vec{r}\cdot\vec{m}$. Taking $\vec{m} = -\vec{r}$, the integral vanishes and one gets $\gamma_1 = \gamma_2$. For $\vec{m} = \vec{r}$ we get $\gamma_1 = 1$, and we recover (20).

We have thus so far established that the 1-bit protocol provides accurate but not exact eIC-POVM statistics in general, and that it simulates exactly the statistics of the continuous POVM.

Worst-case $L_1(P_{1-\mathrm{bit}}, P_Q)$		
Configuration	m	L_1
Projective	2	0.2138
Tetrahedron	4	0.0093
Octahedron	6	0.0159
Hexahedron	8	0.0085
Icosahedron	12	0.0034
Dodecahedron	20	0.0029
Icosidodecahedron	30	0.0028

Table 1. Worst-case L_1 distance for the measurement configurations considered. The order of magnitude of error, decreases by increasing the number of outcomes, starting from $\mathcal{O}(10^{-1})$ for $m = 2$ up to $\mathcal{O}(10^{-3})$ for $m = 30$.

We bridge these facts by giving evidence of the decreasing behavior of the L_1 distance $L_1(P_{1-\mathrm{bit}}, P_Q) \coloneqq \sum_{j=1}^{m}|P_{1-\mathrm{bit}}(j|\vec{r},Y) - P_Q(j|\rho_{\vec{r}},\mathcal{M}_Y)|$ of the 1-bit protocol for eIC-POVMs with increasing number of outcomes. Note that MAE is the average L_1 distance over batch of inputs, hence the choice. We do this in two ways. For spherical t-designs (a subset of the class of eIC-POVMs, where $m \geq t^2/4$), we prove analytically that the L_1 distance scales as $\mathcal{O}(1/\sqrt{t})$ (see Appendix D.). In addition, we perform a numerical worst-case search for the regular polyhedra considered in Fig. 7. That is, for each POVM configuration, we search for the state in the Bloch sphere that maximizes the L_1 distance when the PM scenario is simulated using the 1-bit protocol. Table 1 provides the results.

VI. CONCLUSIONS

We have shown that one bit of classical communication can approximate with high accuracy qubit prepare-and-measure statistics for restricted, highly structured families of measurements. Using a search by NN-procedure, we identified equal-trace rank-one POVMs—and, in particular, informationally complete symmetric configurations—as the measurement families most amenable to 1-bit classical simulation. The numerical results indicate that uniform weights and three-dimensional symmetry are the key features behind this enhanced simulability.

Guided by the structure learned by the network, we derived an explicit analytical 1-bit protocol and analyzed its performance on finite symmetric POVMs. Although the protocol is not exact for finite configurations, its error decreases as the measurement becomes more isotropic. In the continuous limit, corresponding to the covariant qubit POVM, the protocol becomes exact. This proves that the continuous isotropic qubit measurement is exactly 1-bit simulable in the prepare-and-measure scenario.

Our results demonstrate that the 2-bit communication cost required for arbitrary qubit POVMs is not representative of all measurement families. Symmetry and

isotropy can substantially reduce the classical resources needed to reproduce quantum statistics. This result loosely aligns with other recent observations pointing out that, perhaps contrary to prior common understanding, SIC-POVMs do *not* comprise the "most quantum" representatives among all measurements: they are not the hardest to simulate by convex combinations of projective measurements [20], and they possess the least intrinsic randomness [21].

More broadly, this work contributes to the growing effort to identify low-communication classical descriptions of nonclassical correlations. Related questions have been studied in Bell-simulation settings, including 1-bit protocols, communication-assisted models, and neural network based approaches to nonclassicality problems [22–24]. Our results show that neural networks can also help uncover analytically tractable 1-bit protocols in prepare-and-measure scenarios, and point toward a systematic classification of measurement families with reduced classical communication cost.

VII. ACKNOWLEDGEMENTS

RMT, GS, and MZ acknowledge financial support from MCIN grant PID2022-141283NB-I00 funded by MCIN/AEI/10.13039/501100011033. MZ acknowledges Some Sankar Bhattacharya and support from the Ministerio de Ciencia e Innovación of the Spanish Government, contract PRE2020-093634. GG acknowledges DAAD, the Deutsche Forschungsgemeinschaft (DFG, German Research Foundation, project numbers 447948357 and 440958198), the Sino-German Center for Research Promotion (Project M-0294), and the German Ministry of Education and Research (Project QuKuK, BMBF Grant No. 16KIS1618K). GS acknowledges funding from the Programa Talent UAB - Banco de Santander. EF acknowledges support by the Generalitat de Catalunya and the European Social Fund grant Joan Oró 2025 FI-1 00848. JE acknowledges the funding received by i2CAT from Department de Recerca i Universitats of the Generalitat de Catalunya.

[1] G. Brassard, Foundations of Physics **33**, 1593 (2003).
[2] H. Buhrman, R. Cleve, S. Massar, and R. De Wolf, Reviews of Modern Physics **82**, 665 (2010).
[3] R. Gallego, N. Brunner, C. Hadley, and A. Acín, Physical Review Letters **105**, 230501 (2010).
[4] C. de Gois, G. Moreno, R. Nery, S. Brito, R. Chaves, and R. Rabelo, PRX Quantum **2**, 030311 (2021).
[5] M. Pawłowski and N. Brunner, Phys. Rev. A **84**, 010302(R) (2011).
[6] J. Bowles, N. Brunner, and M. Pawłowski, Phys. Rev. A **92**, 022351 (2015).
[7] N. J. Cerf, N. Gisin, and S. Massar, Physical Review Letters **84**, 2521 (2000).
[8] B. F. Toner and D. Bacon, Physical Review Letters **91**, 187904 (2003).
[9] J. Degorre, S. Laplante, and J. Roland, Physical Review A **72**, 062314 (2005).
[10] M. J. Renner, A. Tavakoli, and M. T. Quintino, Physical Review Letters **130**, 120801 (2023).
[11] A. Montina, Physical Review A—Atomic, Molecular, and Optical Physics **84**, 042307 (2011).
[12] T. Rudolph, arXiv preprint quant-ph/0608120 (2006).
[13] M. Zartab, G. Gasbarri, G. Sentís, and R. Muñoz-Tapia, Scientific Reports **16**, 21297 (2026).
[14] S. Schlösser and M. Kleinmann, arXiv preprint arXiv:2603.01255 (2026).
[15] P. Sidajaya, A. D. Lim, B. Yu, and V. Scarani, Quantum **7**, 1150 (2023).
[16] T. Heinosaari, M. A. Jivulescu, and I. Nechita, Journal of Mathematical Physics **61**, 042202 (2020).
[17] Y. Eldar and G. Forney, IEEE Transactions on Information Theory **48**, 599 (2002).
[18] P. G. Casazza and G. Kutyniok, *Finite frames: Theory and applications* (Springer Science & Business Media, 2013).
[19] W. Slomczynski and A. Szymusiak, arXiv preprint quant-ph/1402.0375 (2014).
[20] G. Cobucci, R. Brinster, S. Khandelwal, H. Kampermann, D. Bruß, N. Wyderka, and A. Tavakoli, Phys. Rev. Lett. **136**, 060201 (2026).
[21] F. Curran, arXiv preprint quant-ph/2605.18291 (2026).
[22] A. Montina, Physical Review Letters **109**, 110501 (2012).
[23] M. J. Renner and M. T. Quintino, Quantum **7**, 1149 (2023).
[24] T. Kriváchy, Y. Cai, D. Cavalcanti, A. Tavakoli, N. Gisin, and N. Brunner, npj Quantum Information **6**, 70 (2020).
[25] C. Zhu, R. H. Byrd, P. Lu, and J. Nocedal, ACM Transactions on mathematical software (TOMS) **23**, 550 (1997).

Appendix A: Neural network training methodology and hyperparameter selection

1. Training data and batch-averaged loss

For a given POVM $\mathcal{M}_Y$, the training dataset is constructed as follows. First, we generate M random pure qubit states sampled uniformly on the Bloch sphere. We denote their Bloch vectors by $\vec{r}^{(j)}$, with $j = 1, \ldots, M$.

For each state $\vec{r}^{(j)}$, the target quantum probability distribution is computed using the Born rule,

$$P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y) = p_i\big(1 + \vec{y}_i \cdot \vec{r}^{(j)}\big), \qquad i = 1, \ldots, m. \tag{A1}$$

These probabilities constitute the target distributions of the training procedure.

Next, for each state $\vec{r}^{(j)}$, we generate N realizations of the shared randomness. We denote the k-th random vector associated with the state $\vec{r}^{(j)}$ by $\vec{\lambda}^{(j,k)}$, with $k = 1, \ldots, N$. Each vector is sampled uniformly on the unit sphere. The corresponding communicated bit is then computed as

$$c^{(j,k)} = H\big(\vec{\lambda}^{(j,k)} \cdot \vec{r}^{(j)}\big). \tag{A2}$$

Therefore, each state $\vec{r}^{(j)}$ gives rise to N neural network inputs, each of them formed by the three components of $\vec{\lambda}^{(j,k)}$ and the bit $c^{(j,k)}$. The full training dataset contains $M \times N$ input instances. However, these instances are naturally grouped into M batches, each corresponding to one quantum state.

This grouping is essential because the neural network is not required to reproduce the quantum probability distribution for each individual realization of the shared randomness. Instead, the relevant quantity is the output distribution obtained after averaging over the N realizations associated with the same state $\vec{r}^{(j)}$.

Let $\hat{y}_i^{(j,k)}$ denote the neural network prediction for outcome i for the k-th shared randomness realization associated with the state $\vec{r}^{(j)}$. The batch-averaged prediction for that state is

$$\bar{y}_i^{(j)} = \frac{1}{N} \sum_{k=1}^{N} \hat{y}_i^{(j,k)} = \bar{p}_i^{(j)}, \tag{A3}$$

where $\bar{p}_i^{(j)}$ is the same as (9). This quantity is compared with the target quantum probability

$$y_i^{(j)} := P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y). \tag{A4}$$

The loss associated with the state $\vec{r}^{(j)}$ is defined by

$$L_j = \sum_{i=1}^{m} \Big[\bar{p}_i^{(j)} - P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y)\Big]^2. \tag{A5}$$

The full loss function, according to (11) is then obtained by averaging over the M sampled states:

$$\mathcal{L}_{\mathrm{MSE}} = \frac{1}{M} \sum_{j=1}^{M} L_j = \frac{1}{M} \sum_{j=1}^{M} \sum_{i=1}^{m} \Big[\bar{p}_i^{(j)} - P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y)\Big]^2. \tag{A6}$$

In this way, M is the number of quantum states used during training, whereas N is the number of shared randomness samples used for each state. Increasing M improves the coverage of the Bloch sphere during training, while increasing N improves the numerical estimate of the averaged output distribution.

2. Network architecture and optimization

For each POVM, we train an independent feed-forward neural network. The input layer has four neurons, corresponding to the three Cartesian components of $\vec{\lambda}$ and the communicated bit c. The network has two hidden layers with 10 and 16 neurons, respectively. The output layer has m neurons, where m is the number of outcomes of the POVM.

ReLU activation functions are used in the hidden layers. The output layer uses a softmax activation function, ensuring that the neural network output is a normalized probability distribution over the m possible outcomes.

The training-set size is determined by M and N, corresponding to the number of randomly produced states, and number of independent Monte Carlo samples of shared randomness used per each state, respectively. For the number of input states, we set $M_{\text{val}} = \frac{M}{4}$ for validation, and $M_{\text{test}} = 1000$ for testing. For N, in both validation and testing phases, we have $N_{\text{test}} = N_{\text{val}} = N$. The network is trained using the Adam optimizer with a learning rate of 0.001. In all simulations, the number of training epochs is fixed to 100. An example of the training convergence is shown in Fig. 9. The loss defined in (A6) converges to approximately 1.5×10^{-3} after 100 epochs for both the training and validation datasets.

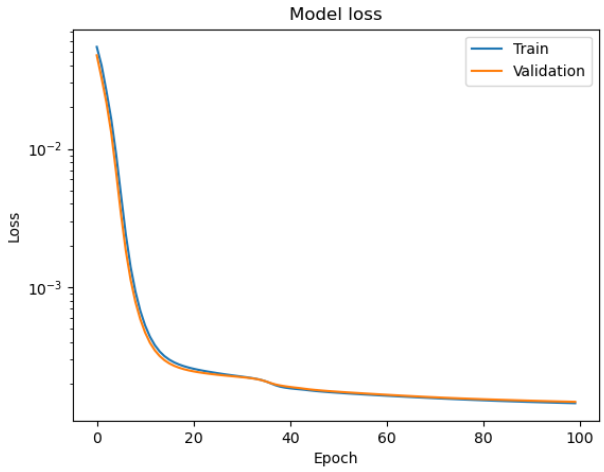

Figure 9. Loss as a function of the number of epochs for the training and validation datasets.

3. Performance metric and selection of M and N

The performance of the NN-procedure is evaluated using MAE between the batch-averaged neural network prediction and the target quantum probability distribution:

$$\text{MAE} = \frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{m}\left|\frac{1}{N}\sum_{k=1}^{N}\hat{y}_i^{(j,k)} - y_i^{(j)}\right| = \frac{1}{M}\sum_{j=1}^{M}\sum_{i=1}^{m}\left|\bar{p}_i^{(j)} - P_Q(i|\rho_{\vec{r}^{(j)}}, \mathcal{M}_Y)\right|. \tag{A7}$$

As in the loss function, the average over the N shared randomness realizations is necessary because the relevant quantity is the probability distribution predicted for each state, rather than the output of the network for a single realization of $\vec{\lambda}$.

To select suitable values of M and N, we analyze the performance of the NN-procedure for a POVM with $m = 6$ elements. Fig. 10 shows the MAE for different values of M and N. The error decreases on average as both parameters increase, although the improvement is more pronounced when increasing N. This is expected because a larger value of N gives a more accurate estimate of the batch-averaged distribution that the network is trained to reproduce, whereas a larger value of M improves the coverage of the state space during training.

For comparison, Fig. 10 also includes the MAE obtained from the exact 2-bit protocol of Ref. [10]. Although that protocol reproduces the quantum probabilities exactly in the infinite-sampling limit, its numerical implementation also depends on the number of samples used to estimate the corresponding probabilities. Therefore, finite-N effects are also present in that case.

The comparison shows that the 1-bit NN-procedure remains less accurate than the exact 2-bit simulation, but nevertheless provides a nontrivial approximation to the target quantum statistics.

For all remaining experiments, we fix $N = 4000$, and $M = 2000$, which provides a good compromise between numerical accuracy and computational cost.

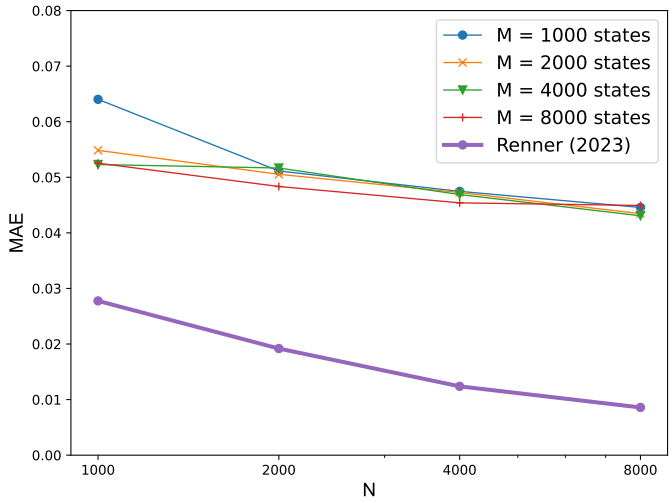

Figure 10. MAE as a function of the batch size N for different sample sizes M, compared with the MAE of the protocol of Renner *et al.* [10].

Appendix B: Generation of random qubit POVMs

Random rank-one qubit POVMs are generated by sets of random points, $\vec{y}$ in the unit sphere $\mathbb{S}^2$. These can be simply obtained from sampling two independent random variables $u, v \sim \mathcal{U}(0,1)$ and defining $\phi = 2\pi u$ and $\theta = \arccos(1-2v)$. The random unit vector in spherical coordinates $\vec{y} = (\sin\theta\cos\phi, \sin\theta\sin\phi, \cos\theta)$ is then assured to be uniformly sampled.

In the implementation, we generated a pool of 10^6 points. For a POVM with m outcomes, we then selected m directions $\{\vec{y}_j\}_{j=1}^m$ at random from this pool. Once the directions $\{\vec{y}_j\}_{j=1}^m$ were fixed, the weights $\{p_j\}_{j=1}^m$ were obtained numerically by enforcing the POVM conditions:

$$\sum_{j=1}^m p_j = 1, \qquad \sum_{j=1}^m p_j \vec{y}_j = \vec{0}. \tag{B1}$$

We used the numerical routine L-BFGS-B [25] that allowed us to fix the initial point to $p_j = 1/m$ and to enforce that no weight became too small. If a sampled set of directions did not admit an admissible solution, a new set of directions was generated and the procedure was repeated.

Appendix C: Exactness of the protocol for the regular polyhedra, where the state aligns with one the measurement outcomes.

In this appendix, we prove that for the regular polyhedra, the protocol gives exact solution whenever $\vec{r} = \vec{y}_i$ for some i.

We start from the analytical description of the protocol from (19) of the main text:

$$P_{1-\text{bit}}(k|\vec{r} = \vec{y}_i, Y) = 2\int_{\mathbb{S}^2} d\vec{\lambda} \frac{H(\vec{y}_i\cdot\vec{\lambda})\vec{y}_k\cdot\vec{\lambda}\, H(\vec{y}_k\cdot\vec{\lambda})}{\sum_{j=1}^m \vec{y}_j\cdot\vec{\lambda}\, H(\vec{y}_j\cdot\vec{\lambda})}. \tag{C1}$$

Now, by $\sum_{j=1}^m \vec{y}_j = 0$, using the identity $xH(x) = \frac{|x|+x}{2}$, and applying $H(\vec{y}_i\cdot\vec{\lambda})$, over the integral region, we get:

$$P_{1-\text{bit}}(k|\vec{r} = \vec{y}_i, Y) = 2\int_{\vec{y}_i\cdot\vec{\lambda}\geq 0} d\vec{\lambda}\, \frac{|\vec{y}_k\cdot\vec{\lambda}|}{\sum_{j=1}^m |\vec{y}_j\cdot\vec{\lambda}|} + 2\int_{\vec{y}_i\cdot\vec{\lambda}\geq 0} d\vec{\lambda}\, \frac{\vec{y}_k\cdot\vec{\lambda}}{\sum_{j=1}^m |\vec{y}_j\cdot\vec{\lambda}|} := I_1 + I_2. \tag{C2}$$

Because the Integrand in I_1 is even under $\vec{\lambda} \mapsto -\vec{\lambda}$, we have:

$$I_1 = \int_{\mathbb{S}^2} d\vec{\lambda} \frac{|\vec{y}_k\cdot\vec{\lambda}|}{\sum_{j=1}^m |\vec{y}_j\cdot\vec{\lambda}|} = \frac{1}{m} \tag{C3}$$

where we have used the symmetry of the regular polyhedra. For I_2 we have:

$$I_2 = 2\vec{y}_k \cdot \int_{\vec{y}_i\cdot\vec{\lambda}\geq 0} d\vec{\lambda} \frac{\vec{\lambda}}{\sum_{j=1}^m |\vec{y}_j \cdot \vec{\lambda}|} \coloneqq 2\vec{y}_k \cdot \vec{w}, \tag{C4}$$

where $\vec{w}$ is an effective vector. Now, let's see how it behaves under the symmetry rotation $R_i\vec{w}$, where R_i is the nontrivial rotation around the axis $\vec{y}_i = R_i\vec{y}_i$, that maps the regular polyhedral shape to itself, up to relabeling the indices:

$$R_i\vec{w} = \int_{\vec{y}_i\cdot\vec{\lambda}\geq 0} d\vec{\lambda} \frac{R_i\vec{\lambda}}{\sum_{j=1}^m |\vec{y}_j \cdot \vec{\lambda}|} = \int_{(\vec{y}_i^T R_i^T R_i\vec{\lambda})\geq 0} |det(R_i)| d\vec{\lambda} \frac{R_i\vec{\lambda}}{\sum_{j=1}^m |\vec{y}_j^T R_i^T R_i\vec{\lambda}|} = \vec{w}. \tag{C5}$$

The last equality results from changing the variable of integration by $\vec{\lambda}' \coloneqq R_i\vec{\lambda}$ and using the fact $|det(R_i)| = 1$. Therefore, we see that $\vec{w} = \alpha\vec{y}_i$. To derive α, we apply the dot product with $\vec{y}_i$:

$$\vec{y}_i \cdot \vec{w} = \alpha = \int_{\vec{y}_i\cdot\vec{\lambda}\geq 0} d\vec{\lambda} \frac{\vec{y}_i \cdot \vec{\lambda}}{\sum_{j=1}^m |\vec{y}_j \cdot \vec{\lambda}|} = \frac{1}{2m}, \tag{C6}$$

where we have used the fact that in the region of integration, $\vec{y}_i \cdot \vec{\lambda} = |\vec{y}_i \cdot \vec{\lambda}|$, and (C3). Now, by putting it into (C4), we have:

$$I_2 = \frac{1}{m}\vec{y}_k \cdot \vec{y}_i, \tag{C7}$$

which alongside with I_1 can be combined and substituted into (C2), which leads to the expression:

$$P_{1-\text{bit}}(k|\vec{r} = \vec{y}_i, Y) = \frac{1}{m}(1 + \vec{y}_i \cdot \vec{y}_k) = P_Q(k|\rho_{\vec{r}=\vec{y}_i}, \mathcal{M}_Y). \qquad \square \tag{C8}$$

Appendix D: Scaling of L_1 distance for spherical t-designs

In this appendix, we study the asymptotic behavior of L_1 distance of the analytical 1-bit protocol from quantum predictions, and provide an upper bound for spherical $t-$design measurements that scales as $\mathcal{O}(1/\sqrt{t})$.

As any spherical $t-$design is a particular example of an e-POVM, we start by writing:

$$M_i = \frac{1}{m}\left(I + \vec{y}_i \cdot \vec{\sigma}\right), \qquad \vec{y}_i \in \mathbb{S}^2, \tag{D1}$$

with $\sum_{i=1}^m \vec{y}_i = \vec{0}$. Without loss of generality, we also restrict to pure input states, so that $\vec{r} \in \mathbb{S}^2$. The Born rule probability is then

$$P_Q(i|\rho_{\vec{r}}, \mathcal{M}_Y) = \frac{1}{m}\left(1 + \vec{y}_i \cdot \vec{r}\right). \tag{D2}$$

It is useful to write the quantum probabilities in integral form as

$$\frac{1}{m}\left(1 + \vec{y}_i \cdot \vec{r}\right) = \frac{8}{m}\int d\vec{\lambda}'\, H(\vec{r}\cdot\vec{\lambda}')\vec{y}_i \cdot \vec{\lambda}'\, H(\vec{y}_i \cdot \vec{\lambda}') = \frac{8}{m}\int_{\vec{r}\cdot\vec{\lambda}'\geq 0} d\vec{\lambda}'\, \vec{y}_i \cdot \vec{\lambda}'\, H(\vec{y}_i \cdot \vec{\lambda}'). \tag{D3}$$

Recall that the 1-bit protocol probability (19) reads

$$P_{1-\text{bit}}(i|\vec{r},\ Y) = 2\int d\vec{\lambda}' \frac{H(\vec{r}\cdot\vec{\lambda}')\,(\vec{y}_i \cdot \vec{\lambda}')\,H(\vec{y}_i \cdot \vec{\lambda}')}{\sum_{j=1}^m (\vec{y}_j \cdot \vec{\lambda}')\,H(\vec{y}_j \cdot \vec{\lambda}')}, \tag{D4}$$

where the denominator can be written as

$$\sum_{j=1}^m (\vec{y}_i \cdot \vec{\lambda}')H(\vec{y}_i \cdot \vec{\lambda}') = \frac{1}{2}\sum_{j=1}^m |\vec{y}_j \cdot \vec{\lambda}'| \coloneqq S_Y(\vec{\lambda}') \tag{D5}$$

In the above we have used the identity $xH(x) = (x + |x|)/2$ and the t-design property $\sum_i \vec{y}_i = 0$.

It is also convenient to define the quantities

$$Q(i|\vec{\lambda}', Y) = \frac{4}{m}(\vec{y}_i \cdot \vec{\lambda}')H(\vec{y}_i \cdot \vec{\lambda}'), \tag{D6}$$

and

$$Z_Y(\vec{\lambda}') := \sum_{i=1}^{m} Q(i|\vec{\lambda}', Y) = \frac{2}{m}\sum_{j=1}^{m} |\vec{y}_j \cdot \vec{\lambda}'| = \frac{4}{m} S_Y(\vec{\lambda}'), \tag{D7}$$

to have

$$|P_{1-\mathrm{bit}}(i|\vec{r}, Y) - P_Q(i|\rho_{\vec{r}}, \mathcal{M}_Y)| = 2\left|\int_{\vec{r}\cdot\vec{\lambda}'\geq 0} Q(i|\vec{\lambda}', Y)\left[\frac{1}{Z_Y(\vec{\lambda}')} - 1\right] d\vec{\lambda}'.\right| \leq 2\int_{\vec{r}\cdot\vec{\lambda}'\geq 0} Q(i|\vec{\lambda}', Y)\left|\frac{1}{Z_Y(\vec{\lambda}')} - 1\right| d\vec{\lambda}'. \tag{D8}$$

We now use the expansion of the absolute function $|x|$ in terms of a series of Legendre polynomials

$$|x| = \sum_{k=0}^{\infty} c_{2k} P_{2k}(x), \tag{D9}$$

where the coefficients read

$$c_{2k} = (-1)^{k-1}(4k+1)\frac{(2k-3)!!}{(2k+2)!!}, \tag{D10}$$

(note that by analytical extension $c_0 = 1/2$) to write the terms $|\vec{y}_i \cdot \vec{\lambda}'|$ in (D5) in the Legendre polynomials basis:

$$S_Y(\vec{\lambda}') = \frac{1}{2}\sum_{j=1}^{m}[\sum_{k=0}^{\infty} c_{2k} P_{2k}(\vec{y}_j \cdot \vec{\lambda}')]. \tag{D11}$$

Notice that for spherical $t-$designs one has:

$$\begin{aligned} S_Y(\vec{\lambda}') &= \frac{1}{2}\sum_{j=1}^{m}\Big(\sum_{k=0}^{\lfloor t/2\rfloor} c_{2k} P_{2k}(\vec{y}_j \cdot \vec{\lambda}') + \sum_{k=\lfloor t/2\rfloor+1}^{\infty} c_{2k} P_{2k}(\vec{y}_j \cdot \vec{\lambda}')\Big) \\ &= \frac{m}{2}\int_{\mathbb{S}^2} d\vec{y}\Big(\sum_{k=0}^{\lfloor t/2\rfloor} c_{2k} P_{2k}(\vec{y} \cdot \vec{\lambda}')\Big) + \frac{1}{2}\sum_{j=1}^{m}\sum_{k=\lfloor t/2\rfloor+1}^{\infty} c_{2k} P_{2k}(\vec{y}_j \cdot \vec{\lambda}') \\ &= \frac{m}{2}\,\mathbb{E}_{\vec{y}}\Big[\Big(\sum_{k=0}^{\lfloor t/2\rfloor} c_{2k} P_{2k}(\vec{y} \cdot \vec{\lambda}')\Big)\Big] + \frac{1}{2}\sum_{j=1}^{m}\sum_{k=\lfloor t/2\rfloor+1}^{\infty} c_{2k} P_{2k}(\vec{y}_j \cdot \vec{\lambda}'). \end{aligned} \tag{D12}$$

We note that all Legendre polynomials up to $\lfloor t/2\rfloor$ satisfy $\mathbb{E}[P_{2k}(\vec{y}\cdot\vec{\lambda}')] = 0$, except for $k = 0$, which gives $\mathbb{E}[P_0(\vec{y}\cdot\vec{\lambda}')] = 1$, resulting in the first term reproducing a constant value of $m/4$, and therefore, one has:

$$S_Y(\vec{\lambda}') = \frac{m}{4} + R_t(\vec{\lambda}'), \tag{D13}$$

where

$$R_t(\vec{\lambda}') := \frac{1}{2}\sum_{j=1}^{m}\sum_{k=\lfloor t/2\rfloor+1}^{\infty} c_{2k} P_{2k}(\vec{y}_j \cdot \vec{\lambda}'). \tag{D14}$$

Additionally, as:

$$\frac{4}{m}|R_t(\vec{\lambda}')| = \frac{2}{m}\left|\sum_{j=1}^{m}|\vec{y}_j \cdot \vec{\lambda}'| - \frac{m}{2}\right| < 1, \tag{D15}$$

substituting it into (D8) leads to:

$$|P_{1-\text{bit}}(i|\vec{r},Y) - P_Q(i|\rho_{\vec{r}},\mathcal{M}_Y)| \leq 2\int_{\vec{r}\cdot\vec{\lambda}'\geq 0} Q(i|\vec{\lambda}',Y)\left|\frac{1}{(1+\frac{4}{m}R_t(\vec{\lambda}'))}-1\right| d\vec{\lambda}'$$
$$\leq \sup_{\vec{\lambda}'}\left|\frac{4R_t(\vec{\lambda}')}{m+4R_t(\vec{\lambda}')}\right| P_Q(i|\rho_{\vec{r}},\mathcal{M}_Y). \tag{D16}$$

Now, let's analyze $F(R_t(\vec{\lambda}')) := |\frac{4R_t(\vec{\lambda}')}{m+4R_t(\vec{\lambda}')}|$, as a function of R_t. To find an upper bound, from (D15) it follows that:

$$\sup_{\vec{\lambda}'\in\mathbb{S}^2}[F(R_t(\vec{\lambda}')] \leq \frac{4\sup_{\vec{\lambda}'\in\mathbb{S}^2}[|R_t(\vec{\lambda}')|]}{m-4\sup_{\vec{\lambda}'\in\mathbb{S}^2}[|R_t(\vec{\lambda}')|]} \tag{D17}$$

On the other hand:

$$|R_t(\vec{\lambda}')| \leq \frac{1}{2}\sum_{j=1}^{m}\sum_{k=\lfloor t/2\rfloor+1}^{\infty}|c_{2k}|\cdot|P_{2k}(\vec{y}_j\cdot\vec{\lambda}')| \leq \frac{m}{2}\sum_{k=\lfloor t/2\rfloor+1}^{\infty}|c_{2k}| \tag{D18}$$

as $|P_{2k}(\cdot)|\leq 1$ for all k's. Substituting this inequality into (D17), for large enough t, where $\sum_{k=\lfloor t/2\rfloor+1}^{\infty}|c_{2k}| < 1/2$, results in:

$$\sup_{\vec{\lambda}'\in\mathbb{S}^2}[F(R_t(\vec{\lambda}')] \leq \frac{2\sup_{\vec{\lambda}'\in\mathbb{S}^2}|R_t(\vec{\lambda}')|}{1-2\sup_{\vec{\lambda}'\in\mathbb{S}^2}|R_t(\vec{\lambda}')|} \leq \frac{2\sum_{k=\lfloor t/2\rfloor+1}^{\infty}|c_{2k}|}{1-2\sum_{k=\lfloor t/2\rfloor+1}^{\infty}|c_{2k}|}. \tag{D19}$$

We finally upper bound the coefficients $|c_{2k}|$. For this, we rewrite (D10) as:

$$|c_{2k}| = \frac{4k+1}{4k^2+2k-2}\cdot\frac{(2k-1)!!}{(2k)!!} \tag{D20}$$

and treat the first and second term separately. For the first term, we have:

$$\frac{4k+1}{4k^2+2k-2} = \frac{5}{4} \quad \text{for } k=1, \text{ and} \quad \frac{4k+1}{4k^2+2k-2} \leq \frac{1}{k} \quad \text{for } k>1. \tag{D21}$$

For the second term, we use the Wallis identity

$$\frac{2}{\pi}\int_0^{\pi/2} dx\cos^{2k}x = \frac{(2k-1)!!}{(2k)!!}, \tag{D22}$$

together with the inequality $\log(\cos x)\leq -x^2/2$, to obtain

$$\frac{(2k-1)!!}{(2k)!!} \leq \frac{2}{\pi}\int_0^{\pi/2} dx\, e^{-kx^2} < \int_0^{\infty} dx\, e^{-kx^2} = \frac{1}{\sqrt{\pi k}} \tag{D23}$$

Consequently, each term of (D21) is upper bounded by:

$$|c_2| \leq \frac{5}{4\sqrt{\pi}}; \quad |c_{2k>2}| \leq \frac{1}{\sqrt{k^3\pi}}. \tag{D24}$$

Further, the summation $\sum_{k=\lfloor t/2\rfloor+1}^{\infty}|c_{2k}|$ is further upper bounded by using the fact that $\frac{1}{\sqrt{k^3\pi}}$ is a monotonic decreasing function of k and the relationship between the sum and the corresponding integral

$$\sum_{k=\lfloor t/2\rfloor+1}^{\infty}|c_{2k}| \leq \frac{1}{\sqrt{\pi}}\int_{\lfloor t/2\rfloor}^{\infty} k^{-3/2}dk = \frac{2}{\sqrt{\pi}\sqrt{\lfloor t/2\rfloor}}. \tag{D25}$$

Therefore, by putting it into (D19), one gets:

$$\sup_{\vec{\lambda}'\in\mathbb{S}^2}[F(R_t(\vec{\lambda}')] \leq \frac{4}{\sqrt{\pi}\sqrt{\lfloor t/2\rfloor}}\cdot\frac{\sqrt{\pi}\sqrt{\lfloor t/2\rfloor}}{\sqrt{\pi}\sqrt{\lfloor t/2\rfloor}-4} = \frac{4}{\sqrt{\pi}\sqrt{\lfloor t/2\rfloor}-4}. \tag{D26}$$

and further, by substituting it into (D16), one has:

$$|P_{1-\mathrm{bit}}(i|\vec{r},Y) - P_Q(i|\rho_{\vec{r}},\mathcal{M}_Y)| \leq \frac{4}{\sqrt{\pi}\sqrt{\lfloor t/2 \rfloor - 4}} P_Q(i|\rho_{\vec{r}},\mathcal{M}_Y). \tag{D27}$$

Finally, by summing over all outcomes, the upper bound for $L_1(P_{1-\mathrm{bit}}, P_Q)$ reads:

$$L_1(P_{1-\mathrm{bit}}, P_Q) = \sum_{i=1}^{m} |P_{1-\mathrm{bit}}(i|\vec{r},Y) - P_Q(i|\rho_{\vec{r}},\mathcal{M}_Y)| \leq \frac{4}{\sqrt{\pi}\sqrt{\lfloor t/2 \rfloor - 4}} = \mathcal{O}(1/\sqrt{t}) \quad \text{as } t \to \infty. \quad \square \tag{D28}$$

Appendix E: Additional Plots

The main text focuses on a representative conditional response map, namely the outcome $\vec{y}_5$ for the transmitted bit $c = 1$, in order to illustrate how the ensemble-averaged neural network output motivates the analytical 1-bit protocol. Here, we provide the corresponding complete set of response maps for the six-direction Cartesian (octahedral) eIC-POVM. In particular, we display all six outcomes and both possible values of the transmitted bit.

Each figure is organized in six rows and four columns. Row i corresponds to the POVM element $\vec{y}_i$, while the first three columns show the neural network conditional response probabilities obtained from ensembles of size $k = 1$, $k = 5$, and $k = 100$, respectively. The last column displays the corresponding response of the analytical 1-bit protocol defined in (17). In each panel, the probability is shown as a function of the shared random vector $\vec{\lambda}$, parametrized by the polar angles (ϕ, θ). The red marker indicates the direction of the Bloch vector associated with the outcome shown in that row. The common color scale, ranging from 0 to 1, allows a direct comparison of the different ensemble sizes, outcomes, and analytical predictions.

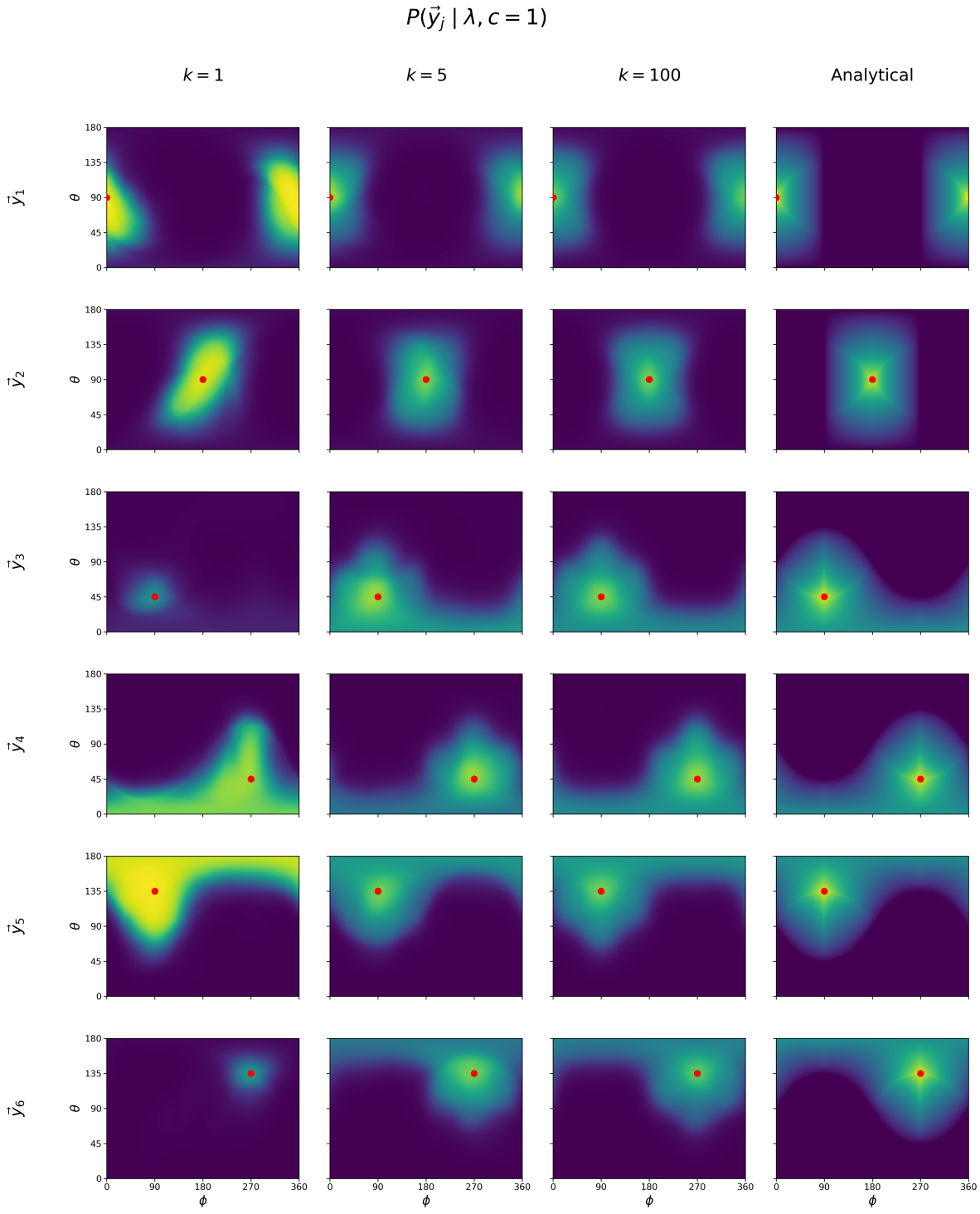

Figure 11. Conditional response maps for the six outcomes of the six-direction Cartesian (octahedral) eIC-POVM, conditioned on the transmitted bit $c = 1$. Each row corresponds to one POVM element $\vec{y}_i$, with $i = 1, \ldots, 6$. The first three columns show the neural network response probabilities $P_{\mathrm{NN}}(i|c = 1, \vec{\lambda}, Y)$ for ensemble sizes $k = 1$, $k = 5$, and $k = 100$, respectively. The fourth column shows the analytical response $P_{1-\mathrm{bit}}(i|\vec{\lambda}', Y)$ from (17), where $c = 1$. The red marker in each panel identifies the Bloch vector $\vec{y}_i$ associated with the displayed outcome.

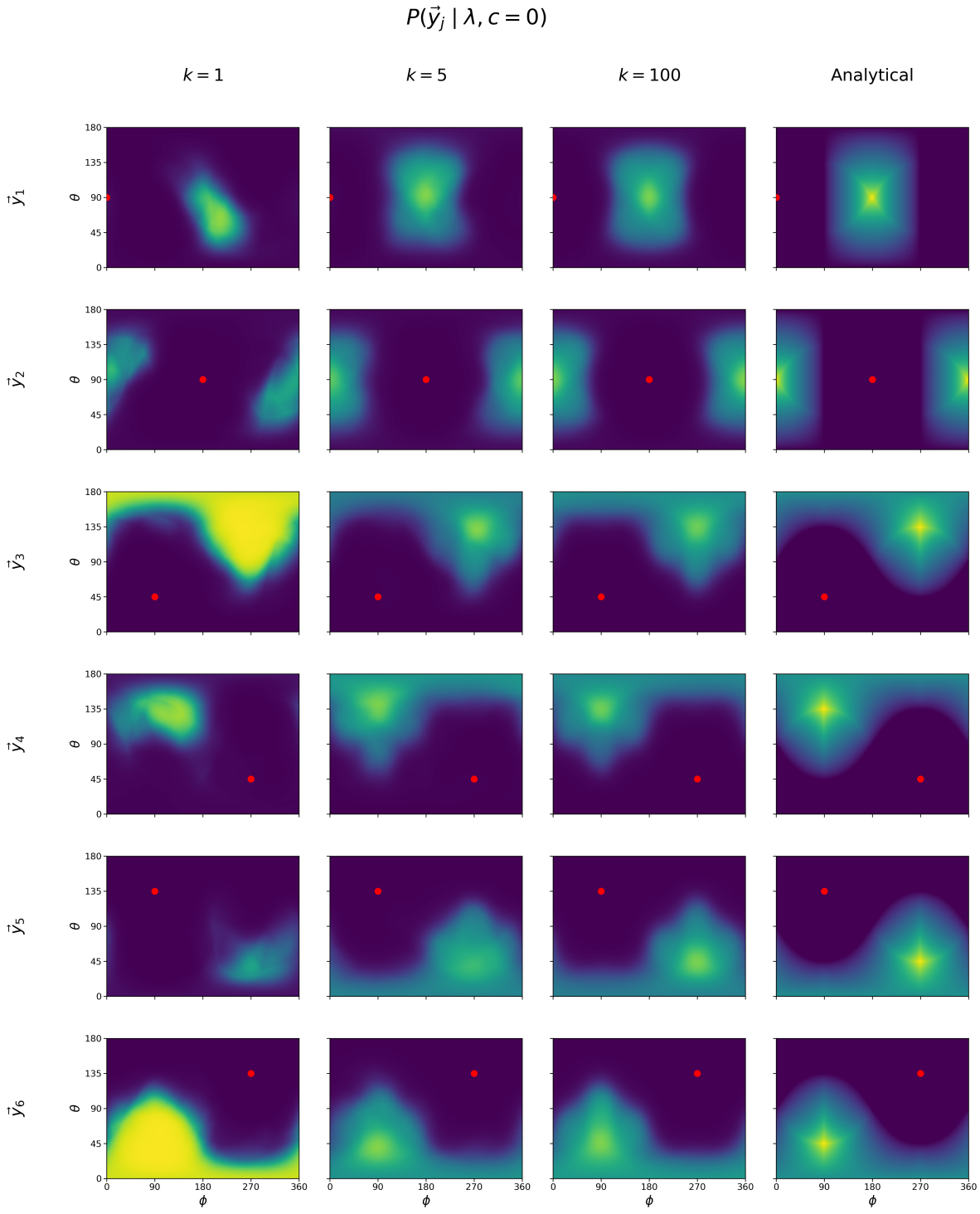

Figure 12. Conditional response maps for the six outcomes of the six-direction Cartesian (octahedral) eIC-POVM, conditioned on the transmitted bit $c = 0$. The arrangement is identical to that in Fig. 11: rows correspond to the outcomes $\vec{y}_1, \ldots, \vec{y}_6$, and columns show the neural network responses for ensemble sizes $k = 1$, $k = 5$, and $k = 100$, followed by the analytical 1-bit response. In this case, the analytical protocol uses the flipped hidden variable $\vec{\lambda}' = -\vec{\lambda}$, as prescribed by (16). The maps show that the same response structure is recovered for $c = 0$, with the support of each outcome shifted according to this sign flip. Red markers indicate the directions of the corresponding POVM Bloch vectors.