# Fully convolutional 3D neural network decoders for surface codes with syndrome circuit noise

**Spiro Gicev[1,2*], Lloyd C. L. Hollenberg[1,2] and Muhammad Usman[2,3]**

[1] Center for Quantum Computation and Communication Technology, School of Physics, University of Melbourne, Parkville, 3010, VIC, Australia
[2] School of Physics, University of Melbourne, Parkville, 3010, VIC, Australia
[3] Data61, CSIRO, Clayton, VIC 3168, Australia
[*] Author to whom any correspondence should be addressed.

E-mail: sgicev@student.unimelb.edu.au

**Abstract.** Artificial Neural Networks (ANNs) are a promising approach to the decoding problem of Quantum Error Correction (QEC), but have observed consistent difficulty when generalising performance to larger QEC codes. Recent scalability-focused approaches have split the decoding workload by using local ANNs to perform initial syndrome processing and leaving final processing to a global residual decoder. We investigated ANN surface code decoding under a scheme exploiting the spatiotemporal structure of syndrome data. In particular, we present a vectorised method for surface code data simulation and benchmark decoding performance when such data defines a multi-label classification problem and generative modelling problem for rotated surface codes with circuit noise after each gate and idle timestep. Performance was found to generalise to rotated surface codes of sizes up to $d = 97$, with depolarisation parameter thresholds of up to 0.7% achieved, competitive with Minimum Weight Perfect Matching (MWPM). Improved latencies, compared with MWPM alone, were found starting at code distances of $d = 33$ and $d = 89$ under noise models above and below threshold respectively. These results suggest promising prospects for ANN-based frameworks for surface code decoding with performance sufficient to support the demands expected from fault-tolerant resource estimates.

*Keywords*:quantum computing, quantum error correction, convolutional neural network, decoders

## 1. Introduction

QEC is expected to facilitate quantum speed-ups by implementing fault-tolerant quantum circuits on noisy hardware [1, 2]. Resource estimates for fault-tolerant algorithm implementations make assumptions regarding multiple characteristics of the underlying computational devices [3, 4]. One assumption involves the characteristics of quantum device noise, often described using parametrisation by a uniform error rate, $p$, associated with each quantum gate. The threshold results yielded by this approach suggest the possibility of arbitrarily error suppression at realistic physical error rates [5, 6]. Multiple experimental devices have demonstrated quantum operations with error rates below threshold, with some performing small-scale logical circuits [7,8]. Resource estimates have introduced another assumption described as the decoding accuracy and reaction time, which together specify expected Logical Error Rates (LERs) and decoding algorithm latencies, respectively. An architecture's decoding reaction time can be the limiting factor governing the maximum depth of fault-tolerant algorithms, even on hardware which is otherwise well below threshold [9], highlighting the importance of efficiency when developing classical algorithms to support noisy quantum hardware [10].

Existing approaches to decoding can be broadly categorised to be based on either look-up tables [11], tensor networks [12], matching [13], clustering [14] or Machine Learning (ML) [15]. In terms of accuracy, look-up tables, tensor network and ML techniques have demonstrated performance comparable to optimal maximum-likelihood decoding under known error models. When details of underlying error models are absent or limited, ML approaches have been shown to be the best performing [16], however with optimality more difficult to show. Beyond accuracy, the practicality of a decoding algorithms additionally depends on their latency and scalability. Look-up tables are limited by the memory they require. Tensor network techniques are limited by increasing computational complexity associated with the entanglement associated with their bond-dimension. ML methods are limited primarily by their training time. In each case the most accurate algorithms are limited to low distance QEC codes. This highlights the importance of the accuracy-latency/scalability trade-off [9], and motivates the use of efficient heuristic algorithms based on matching and clustering, such as MWPM and Union-Find (UF) [13,14]. However, care must still be taken to manage the large overhead such methods yet demand when put in practice [17].

ANN-based decoding has had growing interest for various codes, error models, and ANN designs [18–41]. Ni [42] investigated scalable decoding of toric codes using Convolutional Neural Networks (CNNs) structured similarly to Renormalisation Group (RG) decoders. Under the bit-flip noise model, this decoder achieved performance comparable with MWPM. Meinerz *et al.* [43] investigated the decoding performance of toric codes with depolarising noise up to distance 255, and with depolarising noise and syndrome measurement errors up to distance 63. ANN decoders were constructed using dense layers, and translated adjacent to nontrivial syndrome bits to calculate a set of preliminary data qubit corrections. Any remaining nontrivial syndrome bits were

decoded with MWPM or UF. In [44] a similar approach for unrotated surface codes was developed independently using a Fully Convolutional Neural Network (FCNN). Boundary information was included in the input, allowing generalisation to codes of different shapes and sizes, up to $d = 1025$. Chamberland *et al.* [45] developed on these results by investigating FCNNs decoding on rotated surface codes suffering circuit noise using 3D convolutional layers. Ueno *et al.* [41] performed resource estimates for implementations on superconducting hardware. Further work is yet required to achieve satisfactory ANN decoding under constraints imposed by realistic hardware.

In this work, we investigate ANN approaches to decoding of rotated surface codes suffering circuit noise facilitated by a data representation utilising tiled unit cells. We focus on systems of large code distances ($d > 13$) and errors throughout each step of syndrome measurement circuits. We find that the periodic representation we used enabled significant vectorisation in simulated data preparation and augmentation. The periodic data representation also highlighted an interpretation of low-level decoding as a highly periodic form of conditional distribution sampling. Three decoders are described corresponding to a supervised multi-label classifier trained on unmodified data (corresponding to randomly sampled circuit noise), a supervised multi-label decoder trained on augmented data, and a diffusion model trained on unmodified data.

The remainder of this work is as follows. In Section 2.1 we discuss the rotated surface code memory decoding problem under circuit noise, and the demands imposed by experimental hardware. In Section 2.2 we discuss how surface code simulations can be represented in a periodic representation, facilitating parallel simulation and augmentation of decoding data. In Section 2.3 we investigate the LER performance of ANN decoders trained in the framework of a a multi-label classification problem and a conditional diffusion problem. In Section 2.4 we investigate the timing of multi-label classification decoders. In section 2.6 we briefly comment on the demands imposed by superconducting and trapped atom/ion hardware. Finally, in Section 3 we discuss our results, summarising implications and promising future directions for large-scale ANN decoders.

## 2. Results

### *2.1. Problem Formulation*

*2.1.1. Rotated Surface Codes* Surface codes are topological QEC codes compatible with square arrays of qubits with nearest neighbour connectivity. A distance 5 rotated surface code is shown in Figure 1. As stabiliser codes [46], surface codes use stabiliser operator measurements to extract information regarding locations of errors which may have occurred. Each stabiliser operator acts on up to four adjacent data qubits,

$$V_i = \bigotimes_{j \in F_i} X_j, \quad P_i = \bigotimes_{j \in F_i} Z_j, \tag{1}$$

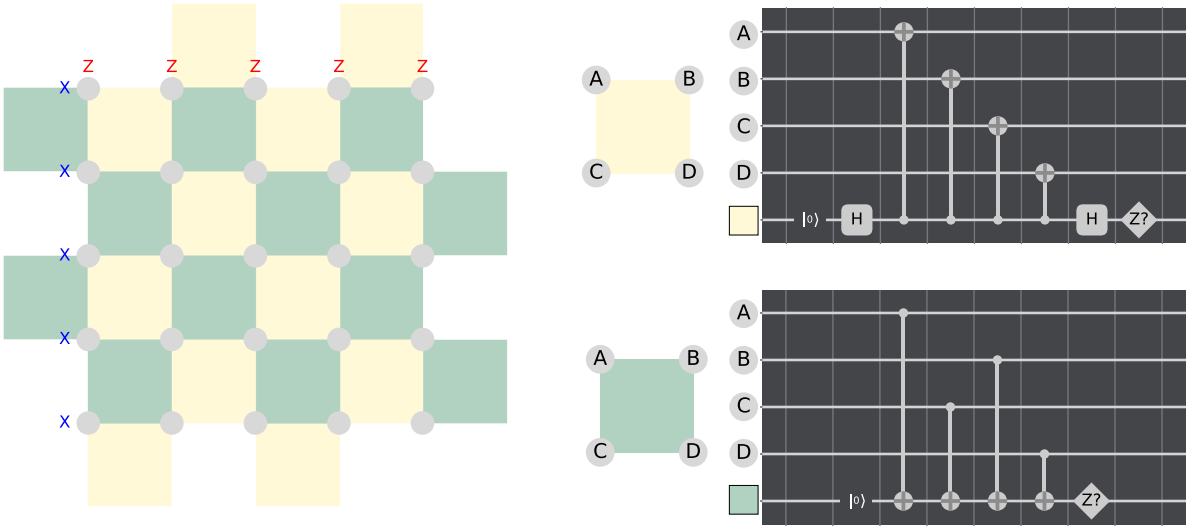


**Figure 1.** A distance 5 rotated surface code logical qubit with examples of an $X$ and $Z$ logical operator shown with chains of blue and red labels, respectively. Measurement circuits of $X$ and $Z$ surface code stabiliser operators are shown to the right. Data qubits are shown in grey. $X$ and $Z$ stabiliser operators are shown with yellow and green tiles, respectively.

where $V_i$ and $P_i$ are vertex and plaquette operators associated with the face $F_i$, and $j \in F_i$ denotes the indices of data qubits on the border of $F_i$. Stabiliser generator operators near code boundaries act on fewer data qubits. Vertex and plaquette operators have $\pm 1$ eigenvalues and must simultaneously commute, restricting valid sets of stabilisers. A set of data qubits are in a surface code code-space state if and only if they are a simultaneous $+1$ eigenstate of all stabiliser generators. Operation which leaves all stabiliser eigenvalues unchanged correspond to logical operators. Nontrivial logical operators correspond to unbroken chains of data qubit operators between two distinct boundaries which commute with all stabilisers, as shown in Figure 1. While this formulation allows multiple distinct families of codes, rotated surface codes have recently dominated research interest and will be the focus of the remainder of this chapter.

Stabiliser operators may be measured simultaneously with the parallel application of two-qubit gates using ancilla qubits similar to those shown in Figure 1. One ancilla qubit is reserved for each stabiliser generator and two-qubit gate connectivity is required only between each ancilla and the data qubits the associated stabiliser generator acts upon. There is some flexibility in the gate sequences facilitating syndrome measurement. The most important consideration is to avoid single errors propagating to two-qubit errors in the same direction as logical operators [47]. The eight time step gate sequence used in this work contains modifications during the first and final measurement cycle involving data qubit reset and measurement, respectively. In this work, the optimisations

associated with delayed reset and accelerated measurement of qubits near edges were neglected in order to keep all resets and measurements simultaneous during each cycle. As measurements proceed, decoding algorithms may be executed to find a set of errors consistent with the observed measurement values. This must be performed so that error corrected logical qubit measurements are known. While some decoders offer corrections to logical qubit measurements directly, the local relationship between errors and syndrome bits suggests the possibility of utilising significantly local approaches to decoding, and realising the associated computational benefits.

*2.1.2. Decoding Demands* Rotated surface codes are expected to underpin quantum memory and logical computation. In these contexts, decoding must be performed at pace with the syndrome measurements of quantum hardware and with accuracy and scale sufficient to achieve the required suppression of LERs. While additional nuances exist in precise resource estimation [9, 48, 49], goals for decoding performance generally correspond to latency of approximately 1 microsecond and LERs of approximately $10^{-9}$ logical errors per $n_c = d$ syndrome measurement cycles. Surface code patches of size corresponding to distance $d = 33$ logical qubits are usually the largest considered. Logical computation involves decoding systems of multiple connected surface code patches when lattice surgery is used, and multiple disconnected patches subject to correlated errors when transversal logical CNOT gates are used. The simulation and experimental progress towards logical operations of systems of increasing scales is an active area of research [50].

Quantum memory is a simpler benchmark which is regularly studied in simulation and experiment corresponding to initialisation, idling, and measurement of individual logical qubits prepared in the $X$ or $Z$ basis. A representation of a quantum memory benchmark is shown in Figure 2 (a). Data qubits are initially prepared in a particular product state associated with the desired logical qubit initialisation, with this information hidden until the evaluation step at the end of the benchmark. Syndrome measurement cycles then proceed, each consisting of time-steps corresponding to ancilla qubit initialisation, change of basis, CNOT gates, changing back to the $Z$ basis, and finally ancilla qubit measurement. Syndrome measurement cycles are repeated $n_c$ times to simulate behaviour when repeated cycles are performed to avoid logical time-like errors during logical qubit interactions. Finally, data qubits are measured, providing a final set of syndrome changes associated with the compatible basis as well as a logical qubit measurement result which awaits a possible correction. As many fault-tolerant logical operations produce additional feed-forward logical operations conditioned of measurement results, further calculation may need to be postponed until error corrected measurement results are known.

Figure 2 (b) shows a 3D representation of the classical data associated with a rotated surface code logical qubit during a quantum memory benchmark. Rounds where stabiliser generator measurement eigenvalues do not agree with the round prior are shown with coloured markers. The first round compares measured stabiliser values to the values associated with data qubit preparation. The final round corresponds to data

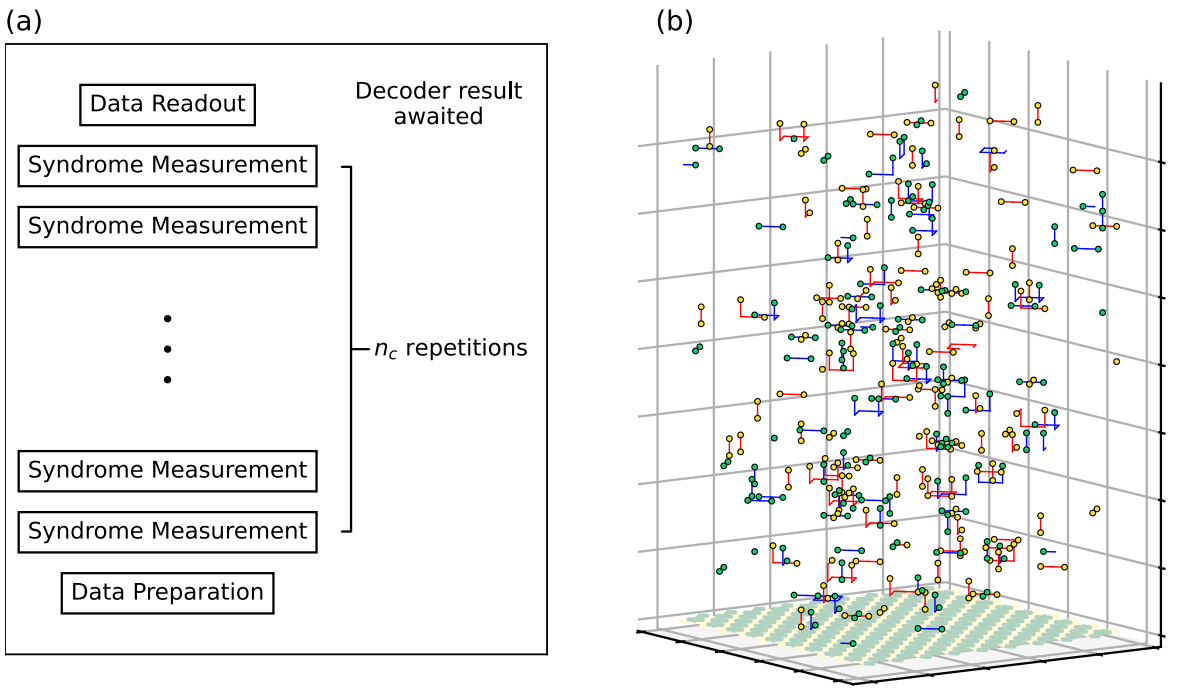


**Figure 2.** Visualising the decoding demands of surface codes. (a) shows a flowchart describing the classical processing demands of rotated surface codes during a quantum memory benchmark. (b) shows a space-time volume showing errors and syndrome changes associated with one quantum memory benchmark instance (time moves vertically upward). Blue and red lines are usually associated usually with $X$ and $Z$ data qubit errors, respectively. Vertical lines are associated with ancilla State Preparation and Measurement (SPAM) errors. $X$ and $Z$ ancilla syndrome bit changes are yellow and green, respectively.

qubit measurement, which is compared to the previous round of stabiliser measurement of the compatible basis. Calculating an appropriate logical correction can be performed by first prescribing a set of corrections compatible with the observed syndrome, and then calculating how these corrections would change the data qubit measurement values. In simulation, the rate at which errors and corrections result in unintended logical operations is used to quantitatively describe decoder performance. Alternatively, the rate at which final logical measurements disagree with data qubit preparation may also be used, and has the benefit of being able to be evaluated experimentally. The LER requirements demanded by fault tolerant algorithm applications are usually set so the expected number of logical errors for the entire algorithm is less than one. From the higher-level perspective of a logical circuit with $n_{\text{gates}}$ logical gates, this would demand the LER $P_L$ per $n_c = d$ cycles defined by

$$p_{\text{L}} \propto \frac{1}{n_{\text{gates}}}. \tag{2}$$

From a lower level perspective, the LER per $n_c = d$ cycles (where $d$ is the code distance)

would need to be inversely proportional to the number of logical blocks, $n_{\text{blocks}}$, defined by

$$p_{\text{L}} \propto \frac{1}{n_{\text{blocks}}}. \tag{3}$$

Codes are generally desired offering the property,

$$p_L \propto p^{d+1/2}, \tag{4}$$

which when combined with proportionality in Equation 3, code distance requirements grow slowly as

$$\frac{d+1}{2} \propto \frac{\log(n_{\text{blocks}})}{\log(1/p)}, \tag{5}$$

so that for a constant error rate, polynomially increasing the number of logical blocks can have errors sufficiently suppressed by linearly increasing code distances. It should be noted that higher order terms do exist in the true form of Equation 4, and that the coefficients of the first few terms can be such that multiple terms are significant at error rates of interest.

Small variations in the above result will also exist due to edge effects and the variance of LER between different blocks. Monte-Carlo simulations can be used to find accurate estimates for more accurate resource requirement estimates.

### *2.2. Data Preparation*

*2.2.1. Unit Cell Representation* A convenient property of rotated surface codes, and more generally surface codes defined on regular lattices [51], is that data qubits and ancilla qubits can be arranged on a regular 2D lattice. Figure 3 (a) shows one particular choice of a unit cell for rotated surface codes. Each unit cell contains space for up to four data qubits and up to four stabiliser generators. We note that a smaller primitive cell is possible, containing two data qubits and two stabiliser generators, but requires a large number of empty cells when describing a rotated surface code in a rectangular array. The presence of data qubits and stabiliser generators can be specified by assigning eight bits (presence bits) to each cell of the system, shown for the distance 5 case in Figure 3 (b). To represent a distance $d$ rotated surface code requires a $n_r \times n_c$ array of cells, where $n_r = n_c = \lfloor (d+3)/2 \rfloor$.

It would be beneficial to also efficiently represent the state of surface codes which have experienced Pauli noise. Two sets of eight additional bits can be used to keep track of whether each qubits has $X$, $Y$, or $Z$ errors present. This is done by decomposing errors into $X$ components and $Z$ components, ignoring global phase, so that $I$, $X$, $Y$, $Z$ correspond to the bits 00, 01, 11, and 10, respectively. Continuing with a unit-cell approach, different points in time can be represented by adding an additional dimension to the lattice. In order to maintain a simple meaning for presence bits, $X$ error bits, and $Z$ error bits, the lattice will only represent points in time before the first CNOT of each syndrome measurement cycle. This is possible since Pauli errors can be propagated

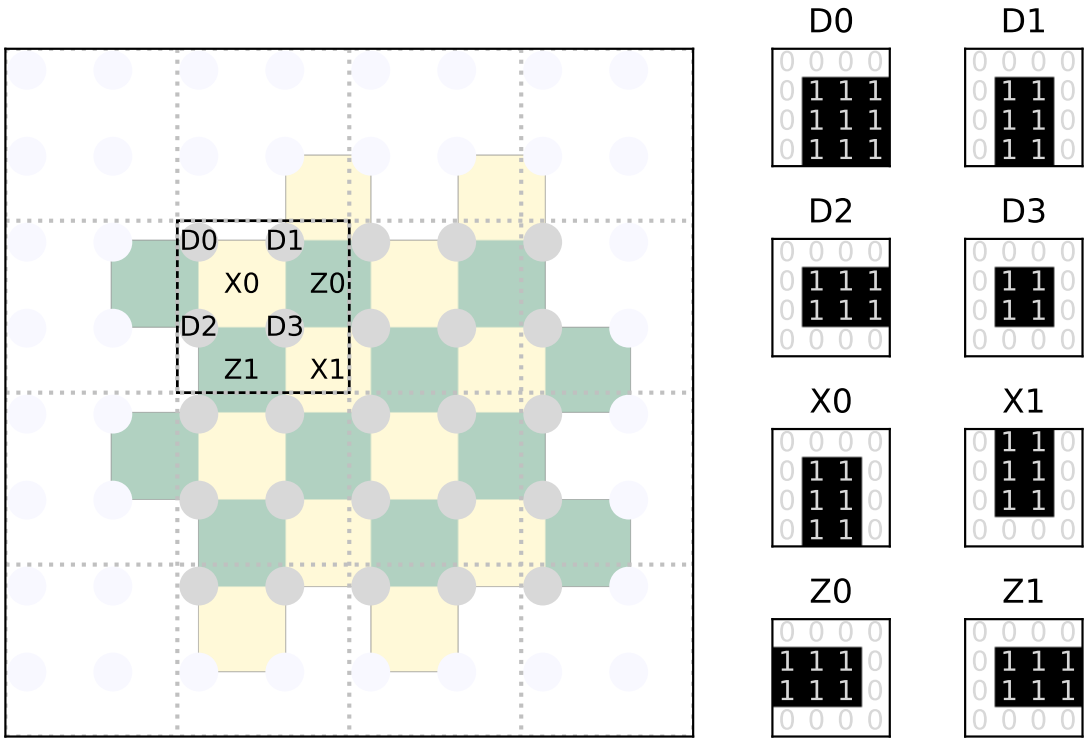


**Figure 3.** A distance-5 rotated surface code represented using an array of unit cells (offset for clarity). Each unit cell may contain up to four data qubits (grey), two $X$ stabiliser generators (yellow), and two $Z$ stabiliser generators (green). Code boundaries are well defined by specifying the presence of qubits by eight bits per unit cell. Unused data qubits are shown in light grey.

through CNOT gates using well known circuit identities [52]. As errors are now possible on ancilla bits, 4 additional bits per unit-cell are required to specify whether errors equivalent to measurement errors are found on ancilla qubits. With this description, the decoder-relevant details of the history of a surface code logical qubit suffering circuit noise can be represented in a form where only equivalent errors just before syndrome extraction begins are specified. This motivates an efficient method of parallel simulation which we present in the next section.

*2.2.2. Parallel Simulation* Surface code simulation is usually performed by simulating syndrome extraction either with stabiliser simulation, or by propagating errors as they occur to find flipped measurement results. The suggestion for representation of logical qubit history described in the previous section motivates a parallel method of simulation. The technique amounts to propagating errors to reference time steps for circuits such as those shown in Figure 4. However, as each individual measurement result still depends on the cumulative effect of the light cone of prior events, general measurement outcomes may be unable to be sampled independently. This is rarely a concern for decoding in QEC,

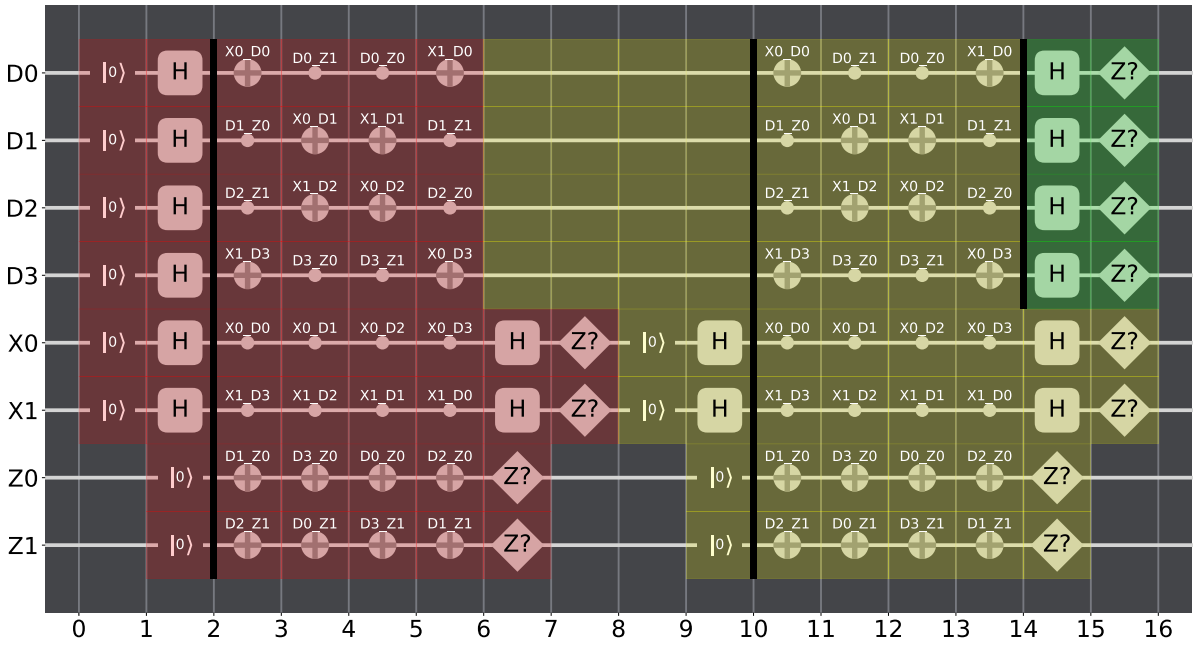


**Figure 4.** Surface code unit cell syndrome measurement with highlighted error detecting regions and reference time steps as black bars. Gates act between qubits and their nearest neighbours, which sometimes are in adjacent cells. At boundaries, gates scheduled on absent qubits are not applied. The red region highlights a data qubit initialisation round. The yellow region highlights a data qubit idling round. The green region highlights data qubit measurement.

however, where syndrome change events are usually sufficient and are sensitive to Pauli noise of only a finite set of time-steps between two consecutive syndrome measurements. Here we will explain how the data of circuit noise errors and syndrome changes can be found by considering the contribution from different classes of errors separately. These are first described with a separation into "class 0", "class 1", and "class 2" errors [53], which correspond to different sets of gate errors, before a decomposition in terms of spacial and temporal correlated syndrome bit flips is instead used.

Class 0 errors describe errors that occur during data qubit idling. The four time-steps of idling errors each data qubit experiences can be combined into a single instance of depolarising noise with parameter $p'_4$ using the equation

$$p'_n = 1 - (1 - p')^n, \tag{6}$$

where $p'$ parametrises the intensity of the depolarising noise experienced each time-step individually, setting $n$ to 4. Parametrisation in terms of depolarisation parameters, $p'$, rather than error rates $p = 3p'/4$, allows this expression to be easily interpreted as the probability that at least one depolarisation event occurs. Thus, the contribution to error bits and syndrome bits from class 0 errors can be sampled by independently sampling one instance of depolarising noise per qubit, per time-step. One may also consider the additional idling errors of data qubits near boundaries. Under circuit noise models

parameterised by depolarisation parameters, these additional errors can be handled by a masked application of class 2 errors, discussed later.

Class 1 errors describe errors associated with ancilla qubit initialisation and measurement. These errors can be considered by simply applying bit-flips to syndrome bits directly. $Z$ stabilisers experience only one instance of reset noise and measurement noise, so experience associated syndrome bit flips with probability $p_2'/2$. As $X$ stabilisers also have two additional Hadamard gates, they instead experience syndrome bit flips with probability $p_4'/2$.

Class 2 errors describe those associated with CNOT gates. In circuit noise error models, these are associated with two-qubit depolarising noise after each two-qubit gate. However, as the set of two qubit Pauli gates is invariant under propagation through CNOT gates, class 2 errors can be equally well described by two-qubit depolarising noise just before each CNOT. Pauli errors propagating through more than one CNOT can become higher weight Pauli errors, which demands careful tracking. While some qubits experience CNOT gates, other qubits may have idle time steps when at code boundaries. In place of single qubit depolarising noise, two qubit noise can still be applied on these qubits, where one qubit of the noise channel is absent. Identical error statistics are achieved under uniform noise models if error channels are parametrised by depolarisation parameters $p'$, but not when error channels are parametrised by error rates $p$.

Our particular implementation of parallel simulation decomposes errors into primitive space-like and time-like errors. Each unit cell has up to eight possible space-like errors corresponding to the two $X$ and $Z$ components of errors applied to each of the four data qubit during idle time-steps. Space-like errors flip adjacent anti-commuting stabiliser operator measurements within the same measurement cycle. Space-like errors are also associated with data qubit preparation and measurement errors, as they also flip adjacent stabiliser operators measured on the first syndrome measurement cycle and final set of inferred values from data qubit measurement, respectively. Each unit cell has up to four possible time-like errors associated with erroneous ancilla qubit preparation or measurement. To incorporate the class 2 CNOT errors in this description, we take advantage of the ability to propagate them backwards to find equivalent errors just before the first CNOT of the syndrome measurement cycle. At that point, equivalent errors present on data qubits can be associated with space-like errors and equivalent errors on ancilla qubits can be associated with time-like errors. Care must be taken that global phases are not included as time-like errors, such as when a $Z$ occurs on a freshly initialised $Z$ operator ancilla. Altogether, 12 bits per unit cell are used to specify the equivalent set of new errors that can be thought to have occurred during a particular syndrome measurement cycle.

The task remains to sample and accumulate all errors to form a complete simulation. Firstly, four random floats are sampled per unit cell to find the contribution from simultaneous data qubit idling time-steps. Error rates are modified to include contributions from data qubit initialisation and measurement errors during the initial and final time-steps, respectively. These space-like errors are stored in two $X$ and $Z$

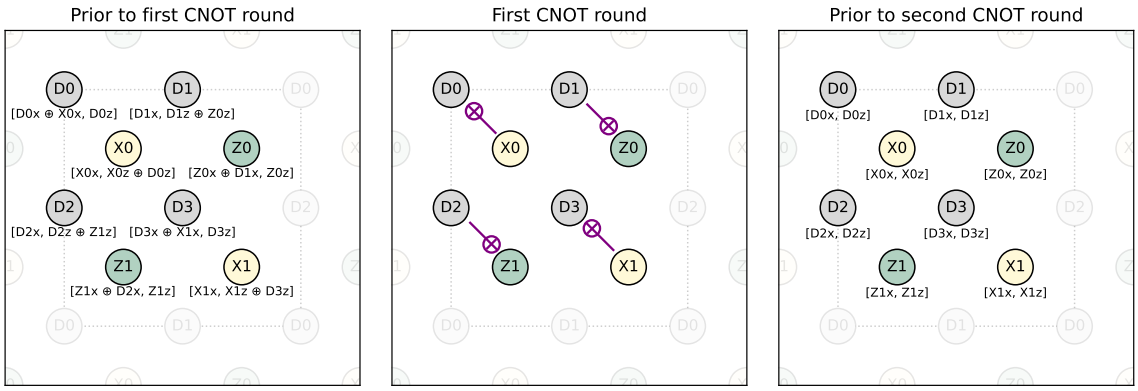


**Figure 5.** Surface code error bits propagating backwards from before the second CNOT round to before the first CNOT round. Error bits are written in row vectors of form $[E_X, E_Z]$, where $E_X$ and $E_Z$ are $X$ and $Z$ error bits, respectively.

space-time error bits associated with the respective qubits. Next, four random floats are sampled per unit cell to find the contribution from ancilla qubit preparation and measurement errors. Differences in $X$ and $Z$ ancilla effective error rates are taken into account. These time-like errors are stored in the four $X$ and $Z$ time-like error bits per cell. Lastly, 16 random floats are sampled per unit cell which corresponds to whether errors occurred during each of the four time-steps of four CNOT gates per unit cell. After sampling according to the error rate, the 16 resultant bits are repeated 4 times to form a mask of 64 bits per unit cell. When this mask is applied to a new sampling of 64 uniformly random bits, the resultant set of 64 bits correspond to the correlated $X$ and $Z$ errors associated with parallel sampling of circuit noise interpreted to occur just before each CNOT gate. The bits associated with missing qubits can be set to zero with the use of presence bit information. The bits which correspond to qubits near edges, which do not have a partner to interact with during some CNOT time-steps, are used as idling error bits, which is a convenient property of noise models parametrised by uniform depolarisation parameters $p'$. The final step involves propagating all error bits to their equivalent values just before the first CNOT. This was done by performing vectorised boolean logic operations on error bits of future time-steps. Alternatively, parallel error propagation may be performed with the use of 2D convolutional kernels. An example of a bit propagation is shown in Figure 5. Making use of the periodicity of surface codes, multiple qubits can have their error states upgraded simultaneously, based on simple boolean functions of their prior (or in this case future) states.

Finally, the associated syndrome change events must be calculated. This is calculated in parallel by summing up the error bits associated with each syndrome change site (modulo 2). The space-time errors and syndrome changes can be plotted together in 3D as shown in Figure 2 (b).

Before proceeding to the application of this simulation technique to decoder training, the connection to decoder graphs and Detector Error Models (DEMs) should be noted.

The parallel simulation described here can be essentially thought of as sampling edges from a decoder hyper-graph, similar to what was recently described by Chamberland *et al.* [45], or sampling flip events from a DEM [54]. The propagation of errors to just before the first CNOT of syndrome extraction is the key additional result, which allows decompositions of equivalent space-like and time-like errors to be ascribed to outcomes of sampling circuit noise. This defines an explicit mapping from circuit noise to correlated phenomenological noise.

*2.2.3. Data Augmentation* Surface code space-time edges can be modified while preserving the effective errors experienced. For example, when stabiliser operators are applied to surface code code-space states during a particular instance of time, the resultant state is identical to the initial state. If instead stabiliser operators were applied to a surface code code-space state with Pauli noise, a global phase of $-1$ may be applied, which can usually be ignored. From the perspective of space-time edges, this corresponds to applying a bit-wise XOR operation between the original space-time edges and a set of space-time edges containing a loop parallel to the $X-Y$ plane (or starting and ending at the same boundary). These space-time edges correspond to a trivial syndrome and do not apply a non-trivial logical operation. We can lift the restriction to the $X-Y$ plane to instead consider all loops of space-time errors, which again result in a trivial syndrome and no net nontrivial logical operators. We will refer to these space-time edges as "space-time simplifiers" or just "simplifiers" for short. Space-time simplifiers can be interpreted as modifications to errors which may have occurred but never have any measurable effects. A set of space-time edges forming a basis for space-time simplifiers can be constructed by considering the application of each different stabiliser operator at all points in time (a total of 4 per unit cell), and also applying each different pair of repeated errors together with adjacent time-like errors on non-commuting stabiliser qubits (a total of 8 per unit cell). The set of simplifiers form a group, where each element is its own inverse, and composition can be defined by addition mod 2 when space-time errors are represented with bits. Associativity and commutativity follow from the definition of the composition operation.

Simplifiers can be applied to reduce the number of nontrivial edges in a space-time edge representation. This can be done by checking whether primitive simplifiers have more than half of their edges activated, and applying rule-based symmetry breaking similar to what was described in previous work [44]. Finding optimal solutions can be expected to be prohibited, in most cases, by in general needing to check an exponential number of distinct primitive simplifier combinations. An example of the effect of simplifier operations on sampled circuit noise close to threshold is shown in Figure 6. From this example, we can see that surface code error data close to threshold frequently contains simplifiers in the form of full space-like stabiliser loops, and features significant variation in how error chains connecting two changed syndrome bits are sampled. Both of these effects are expected to reduce ANN decoder performance.

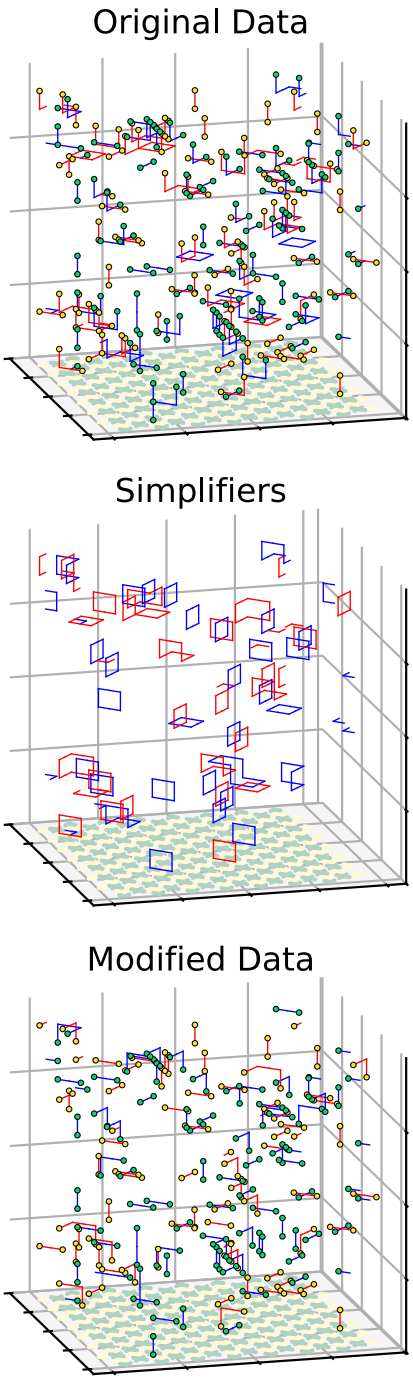


**Figure 6.** The application of simplifier operators on rotated surface code data. The subfigure titled "Original Data" shows errors and syndrome bit changes of a distance 15 code with $p' = 0.005$ noise. The subfigure titled "Simplifiers" shows error chain loops identified to reduce total error count or break symmetry. The subfigure titled "Modified Data" shows the resultant set of edges after applying simplifiers.

### *2.3. Decoder Performance*

*2.3.1. PyMatching Decoders* PyMatching is a Python implementation of MWPM commonly used in QEC research due its relative ease of use and compatibility with the optimised stabiliser simulator Stim [54,55]. Here we will discuss two decoders constructed using PyMatching which will later be used as mop-up decoders for ANN models. The first, referred to as Parity Check Matrix (PCM) PyMatching, is constructed using a matrix defined by the code stabilisers. This then can be used to define a rectilinear matching problem, where diagonal hook errors are not present, effectively ignoring many effects associated with circuit noise such as hook errors and relative error likelihoods. The second, referred to as DEM PyMatching, is constructed utilising a DEM derived from the noisy Stim circuits each of memory experiment. DEMs refer to a set of probabilities assigned to combinations of detection events and logical operator changes caused by independent error events. This allows the construction of decoding graphs which include diagonal hook errors, although still omit some correlations caused when errors cause more than 2 detection events. Results for $n_c = d$ rounds of decoding for distance $d = 5$ to $d = 33$ codes using PCM PyMatching are shown in Figure 7. A threshold at approximately $p' = 5 \times 10^{-3}$ is attained, with logical noise biased towards $Z$ errors. The bias towards $Z$ errors can be understood to arise due to $X$ stabiliser measurement circuits containing two more instances of depolarising noise than $Z$ stabiliser measurement circuits.

Results for $n_c = d$ rounds of decoding for distance $d = 5$ to $d = 33$ codes using DEM PyMatching are shown in Figure 8. The results show an increased threshold exceeding $p' = 6.5 \times 10^{-3}$, and a corresponding substantial decrease in LERs below threshold compared to PCM PyMatching. Noise remains biased towards $Z$ errors. Although the optimisation of edge weights and inclusion of hook error edges is generally considered to not demand substantial additional computational overhead, the use of these optimisations does rely on assumptions about the noise present, which may not

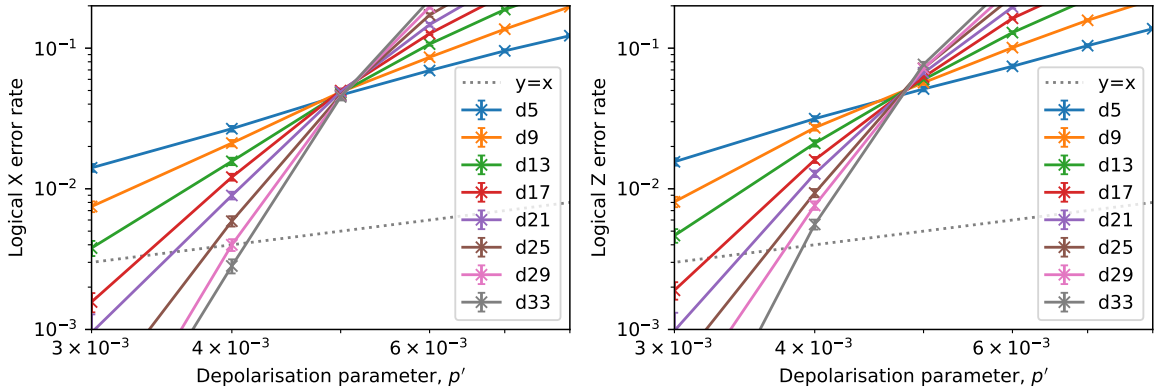


**Figure 7.** Performance of PCM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

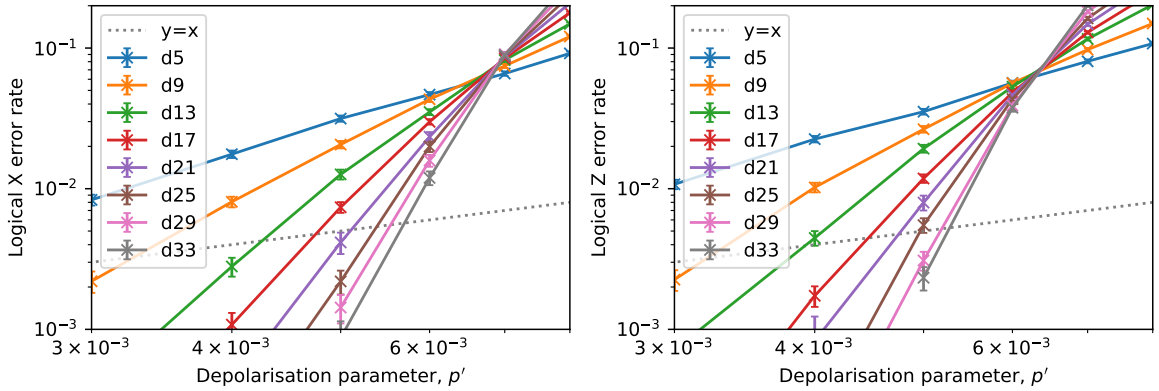


**Figure 8.** Performance of DEM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

be justified on a real quantum device, although ANN-based training can also suffer the same pitfall. Additionally, the performance of PCM PyMatching is still of relevance as a point of comparison when it is used as a global decoder for ANN-based decoding.

An interesting property of circuit noise threshold curves for PyMatching decoders is that intersections between different code distances appear close to the same depolarisation parameter value. A variation in terms of code distance is to be expected, as lower codes are dominated by boundary effects which are gradually overtaken by bulk effects for larger distances. To investigate this phenomenon further, we calculated exact intersection points assuming power-law extrapolation is valid near the intersection points. This corresponds to linear extrapolation on the log-log plot. Results are show in Figures 9 and 10 for PCM and DEM PyMatching, respectively. From the two figures, we note that intersection points indeed tend to increase as code distances increase, suggesting that threshold error rates are likely to be underestimated. Additionally, it can be observed that depolarisation parameters of intersections of $Z$ LER curves are consistently lower than those of the corresponding $X$ LER curves. This can be understood to follow from the asymmetry inherent in the syndrome measurement circuits when Hadamard change of basis operations are used [56].

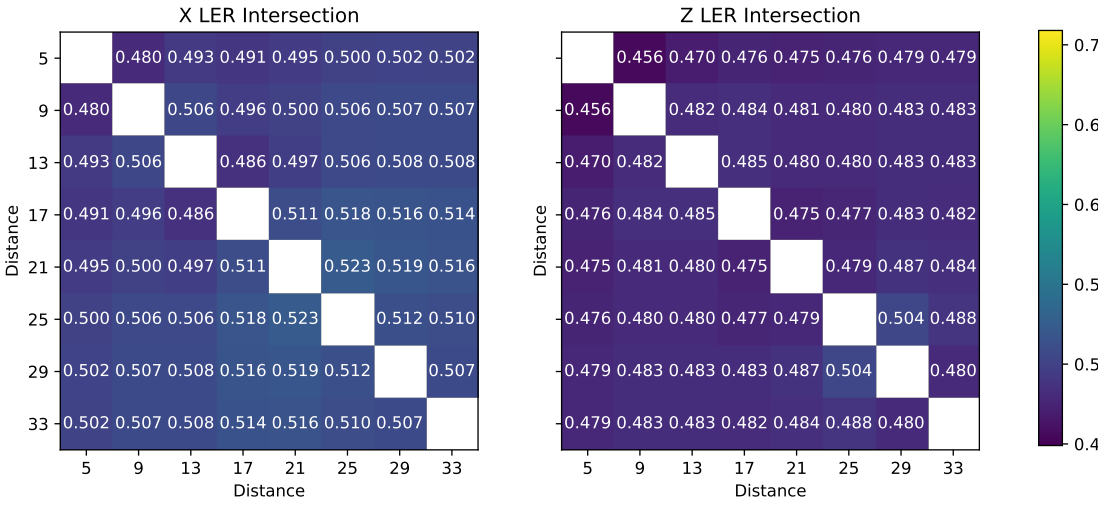


**Figure 9.** Matrices showing intersections between LER curves for rotated surface codes of distances between 5 and 33 when decoded with PCM PyMatching. Text in each cell shows the depolarisation parameter value rounded to three decimal places based on extrapolation of local error rates calculated at depolarisation parameters at the nearest two tenth of a percent.

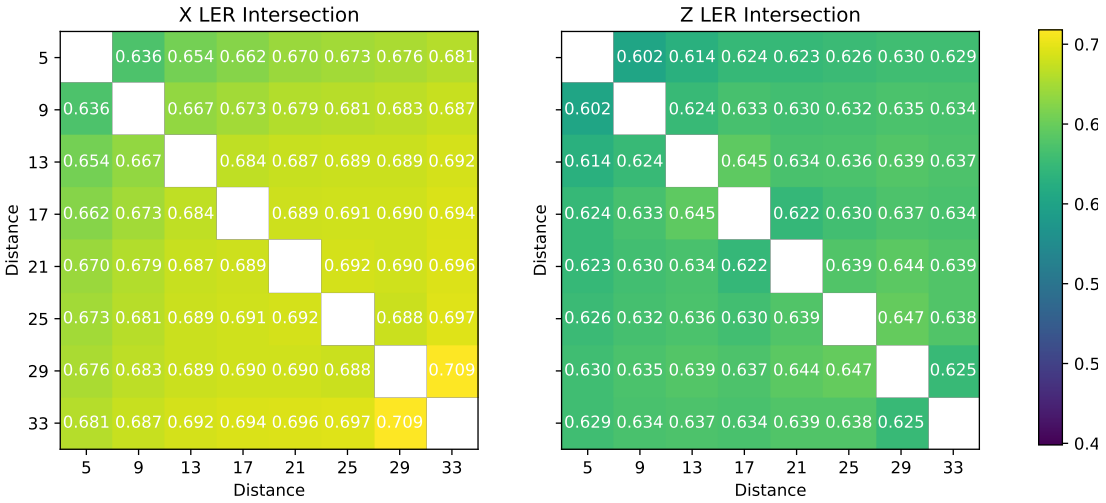


**Figure 10.** Matrices showing intersections between LER curves for rotated surface codes of distances between 5 and 33 when decoded with DEM PyMatching. Text in each cell shows the depolarisation parameter value rounded to three decimal places based on extrapolation of local error rates calculated at depolarisation parameters at the nearest two tenth of a percent.

*2.3.2. Multi-label Classifier Decoder* In order to take advantage of translational symmetry inherent in surface code decoding, the neural network architecture we considered corresponds to repeated three-dimensional convolution layers. Activation functions were set to ReLu activations, with sigmoid activation functions on the last layer. This ensures that the final layer can be interpreted as a probability distribution of error correction bits, conditioned on the syndrome bits and boundary information given as input. Three hidden layers were used, each featuring 128 kernels and kernel sizes of $3 \times 3 \times 3$. Training data was generated by simulating surface code memory experiments via vectorised unit-cell based simulation with uniform depolarisation parameter noise channels. Approximately $500,000$ training instances were generated, corresponding to distance 33 rotated surface codes suffering circuit noise with depolarisation parameter values from 0.001 to 0.007.

Testing was performed by sampling syndrome strings and logical operator changes in Stim [54]. ANN corrections were obtained by providing boundary information and syndrome information as input and receiving corrections in the form of space-like and time-like errors as output. The syndrome bits associated with the output corrections were then used to calculate changes to the original syndrome using local parity sums. The resultant residual syndrome differs from the original syndrome by some syndrome bits being resolved (matched with other bits or to a boundary) and some instead displaced. Residual syndrome bits are decoded with an auxiliary global decoder, in this case PyMatching, which outputs whether a logical operator is expected to be crossed after matching is performed. Results are shown in Figures 11 and 12 for PCM and DEM PyMatching mop-up, respectively. We find that the use of ANN decoding offers LER and threshold improvements for PCM PyMatching. In contrast, performance for DEM PyMatching appears to generally not show improvement compared to using the global decoder alone.

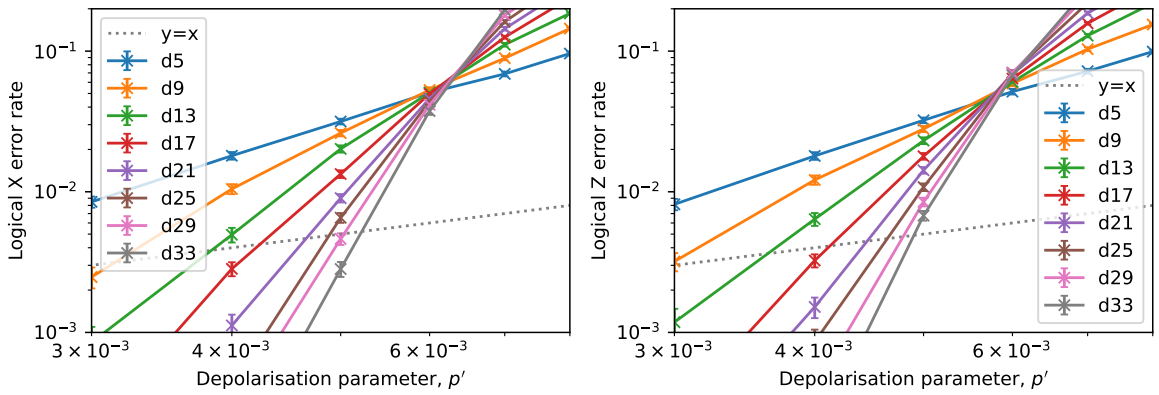


**Figure 11.** Performance multi-label classifier ANN decoding, followed by PCM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

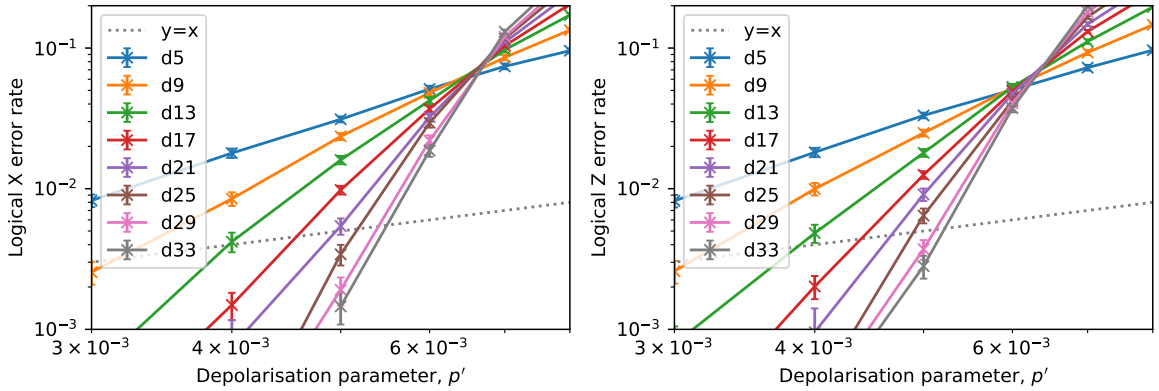


**Figure 12.** Performance multi-label classifier ANN decoding, followed by DEM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

*2.3.3. Simplified Multi-label Classifier Decoder* An identical neural network structure was trained on data having gone through the error simplification process described in Section 2.2.3. The use of error simplification reduces the negative impacts associated with full error loops and variety in error chains. The result is a network which predicts error chains with increased certainty, providing output probabilities away from 0.5. An example showing the difference between outputs for networks trained with unsimplified data compared to simplified data is shown in Figure 13. The key property is the reduced tendency to split error probabilities across multiple equivalent error chains when symmetry is broken by the error simplification process.

Utilising error simplification was observed to improve LERs and increase thresholds in previous works [45]. Figures 14 and 15 show LER curves when using PCM PyMatching and DEM PyMatching as a mop-up decoders, respectively. From the figures, it can be seen that there are improvements to LERs and thresholds when compared with the related results for unsimplified data. The improvements are, however, less pronounced for the comparison with the DEM PyMatching decoder. It should be noted that this improved performance comes at the cost of no additional demands for ANN computation at run time, as the network structures are identical. The simplification process does, however, come at the one time cost of increased computational demands during training, as searching for and applying simplification operations requires comparable operations to vectorised simulation.

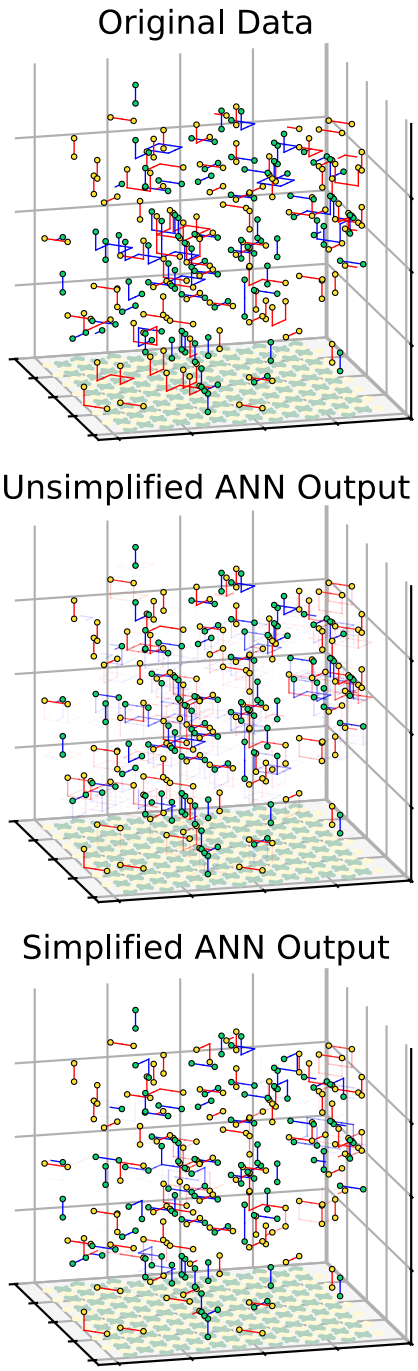


**Figure 13.** Output predictions of ANNs when trained on unsimplified data compared to simplified data. The original data corrections to circuit noise with depolarisation parameter $p = 0.005$.

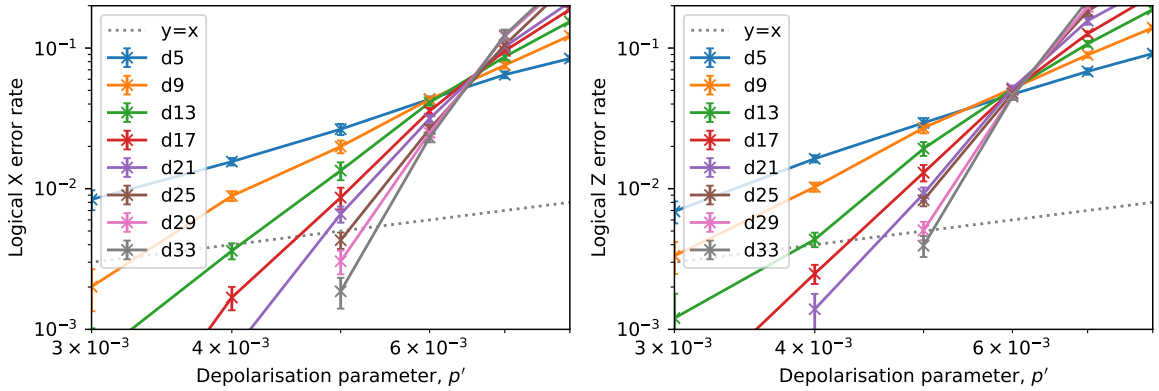


**Figure 14.** Performance multi-label classifier ANN (simplified data during training) decoding, followed by PCM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

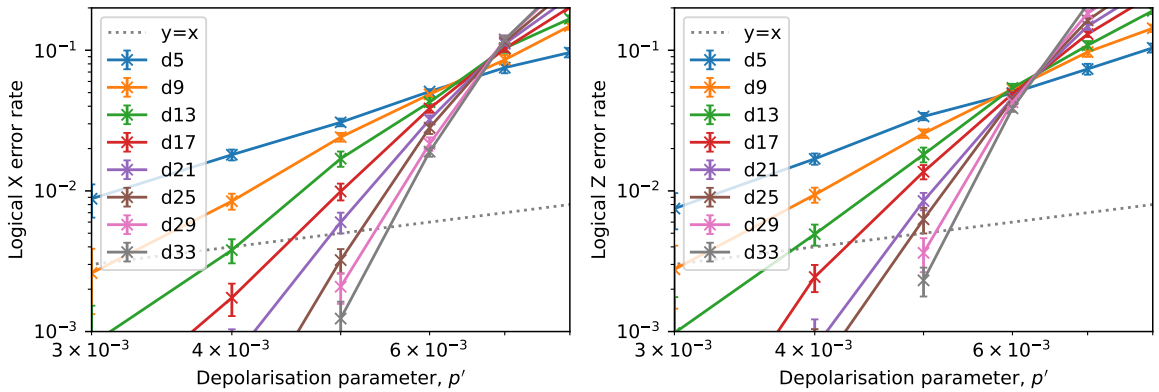


**Figure 15.** Performance multi-label classifier ANN (simplified data during training) decoding, followed by DEM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

*2.3.4. Conditional Diffusion Decoder* Beyond a multi-label classification approach, additional ML paradigms have been developed which may have relevance to the decoding problem of QEC codes. One example corresponds to mapping the problem of generating adequate QEC corrections to generative modelling. Diffusion models have seen recent popularity in generative modelling tasks [57], but have not yet been investigated in the context of QEC decoding. We used our training data to train a similarly structured neural network as our multi-label classification models to instead perform the task of diffusive generative modelling. The input layer of the diffusion model was changed to include a source of tentative corrections, initialised as uniform random values between zero and one. Additionally, the syndrome associated with the tentative corrections is calculated by generalising parity calculations using fuzzy logic. This corresponded to repeated application of fuzzy XOR operations, defined as

$$x \oplus_{\text{fuzzy}} y = x(1-y) + (1-x)y, \tag{7}$$

where the inputs $x, y \in [0, 1]$ correspond to tentative correction values. A fuzzy residual syndrome is finally calculated by calculating the bit-wise fuzzy XOR between the original syndrome and the fuzzy syndrome of the tentative corrections. This is given as an additional input to the network to give information on where further modifications to corrections may be needed. Once this training data is prepared, ANN training can proceed as a supervised learning problem. We note that in contrast to most diffusion models, the model presented here was trained to predict samples of original errors directly, rather than modifications to the tentative corrections. Additionally, no time schedule was used and noise was added to the original errors of the training data by mixing the data with noise scale parameter $p_{mix} \in [0, 1]$. The model was trained on approximately $500,000$ training instances of distance 33 rotated surface codes suffering circuit noise depolarisation parameter values from 0.001 to 0.007. We note that no simplifications to remove loops or break symmetry was applied to the training data.

An interesting characteristic of generative models is that they should be able to produce non-deterministic outputs. This means that, unlike the multi-label classifier decoders which always produce the same set of corrections, repeated sampling of a diffusion model should result is different sets of corrections. Figure 16 shows two shots of ANN corrections performed for the same syndrome, but different random tentative corrections as input. We note that we do observe slight differences in corrections based on input noise given to the network. However, the network still regularly provides the same output, in spite of different noise inputs, in particular near isolated individual errors. We observed that a network trained to perform the diffusion task yields similar uncertainty in output predictions (multiple additional faint edges) as multi-label classification models, but with reduced uncertainty after multiple additional passes. Interestingly, this includes prediction of superfluous corrections in the form of loops. The results show that residual corrections are still sometimes needed from a global decoder.

The LER performance of our diffusion model decoder is shown in Figures 17 and 18 when performing mop-up decoding with PCM PyMatching and DEM PyMatching,

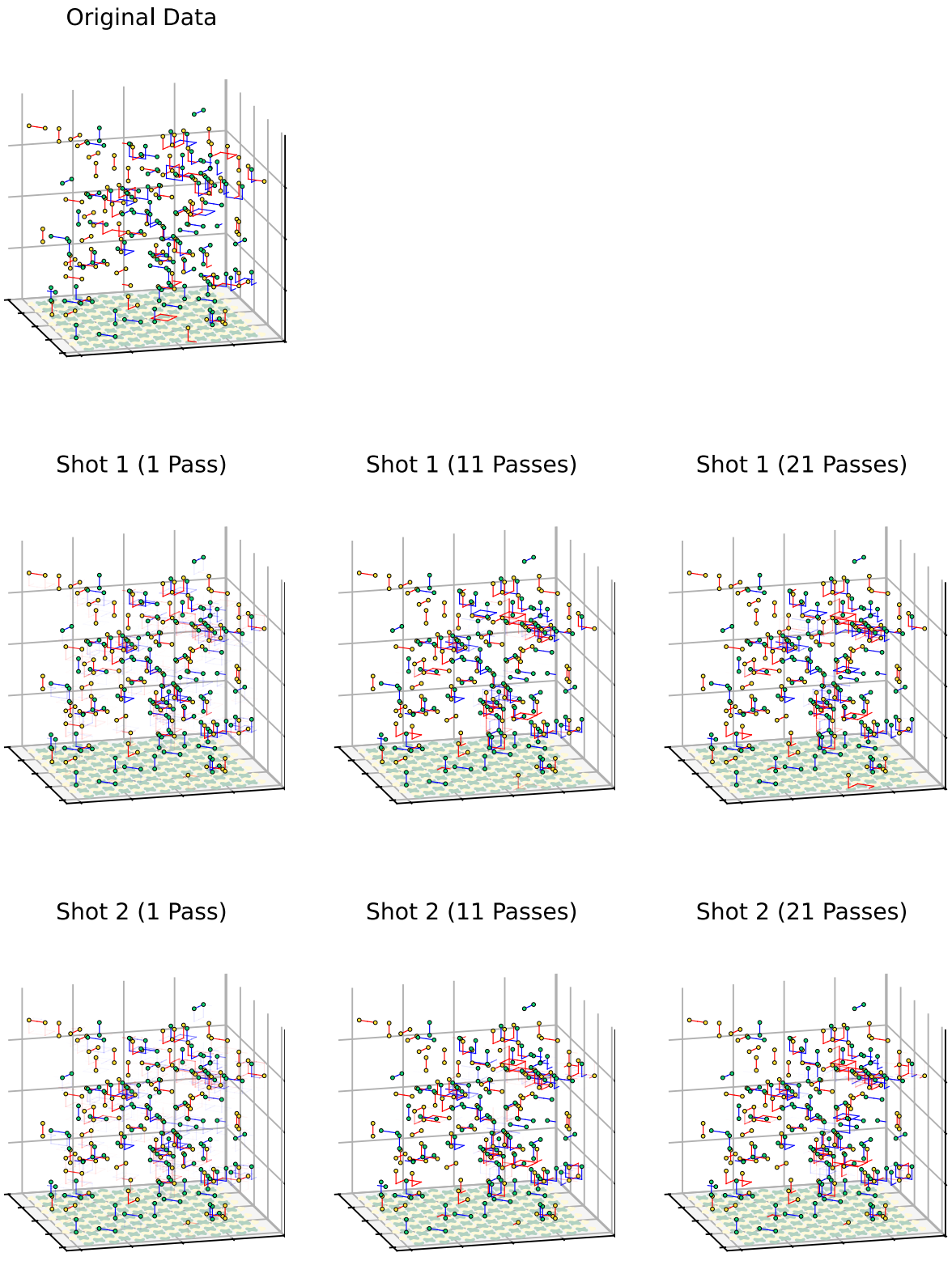


**Figure 16.** Output predictions a diffusion ANN after 1, 11 and 21 passes through the model. Two shots are shown, corresponding to different random noise given as input. The original data corrections to circuit noise with depolarisation parameter $p' = 0.005$.

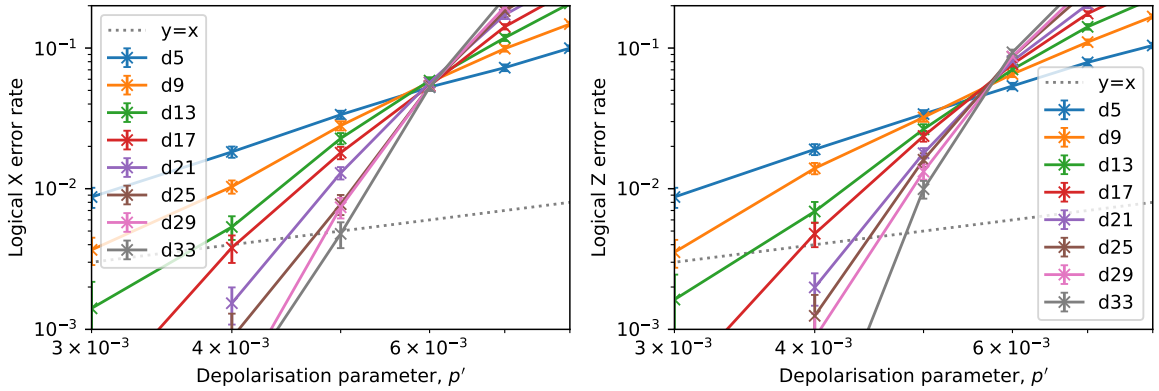


**Figure 17.** Performance 11-pass diffusion ANN decoding, followed by PCM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

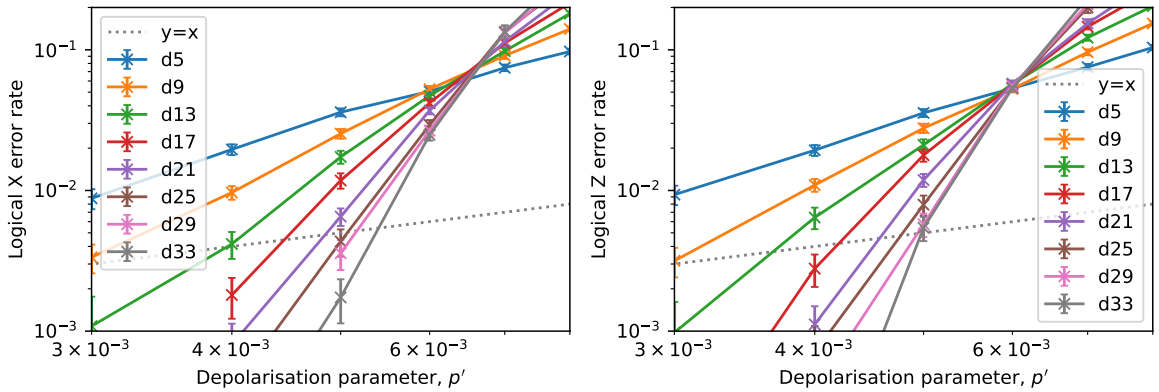


**Figure 18.** Performance 11-pass diffusion ANN decoding, followed by DEM PyMatching for rotated surface codes suffering uniform depolarisation parameter, $p'$, noise.

respectively. We find that comparable performance is achieved compared to the models based on multi-label classification.

As a final comparison between all three ANN decoders presented here, the intersection depolarisation parameters between difference distance codes is shown in Figure 19. The performance of the MWPM decoders each ANN model uses for the mop-up step is shown again in the top row for convenience. The results show that PCM PyMatching benefits from the use of each of the three ANN decoders, with slightly better performance on average achieved when using simplified training data. For DEM PyMatching, the results consistently show modest performance improvements for $X$ LERs, with competitive, but mixed, results for $Z$ LERs.

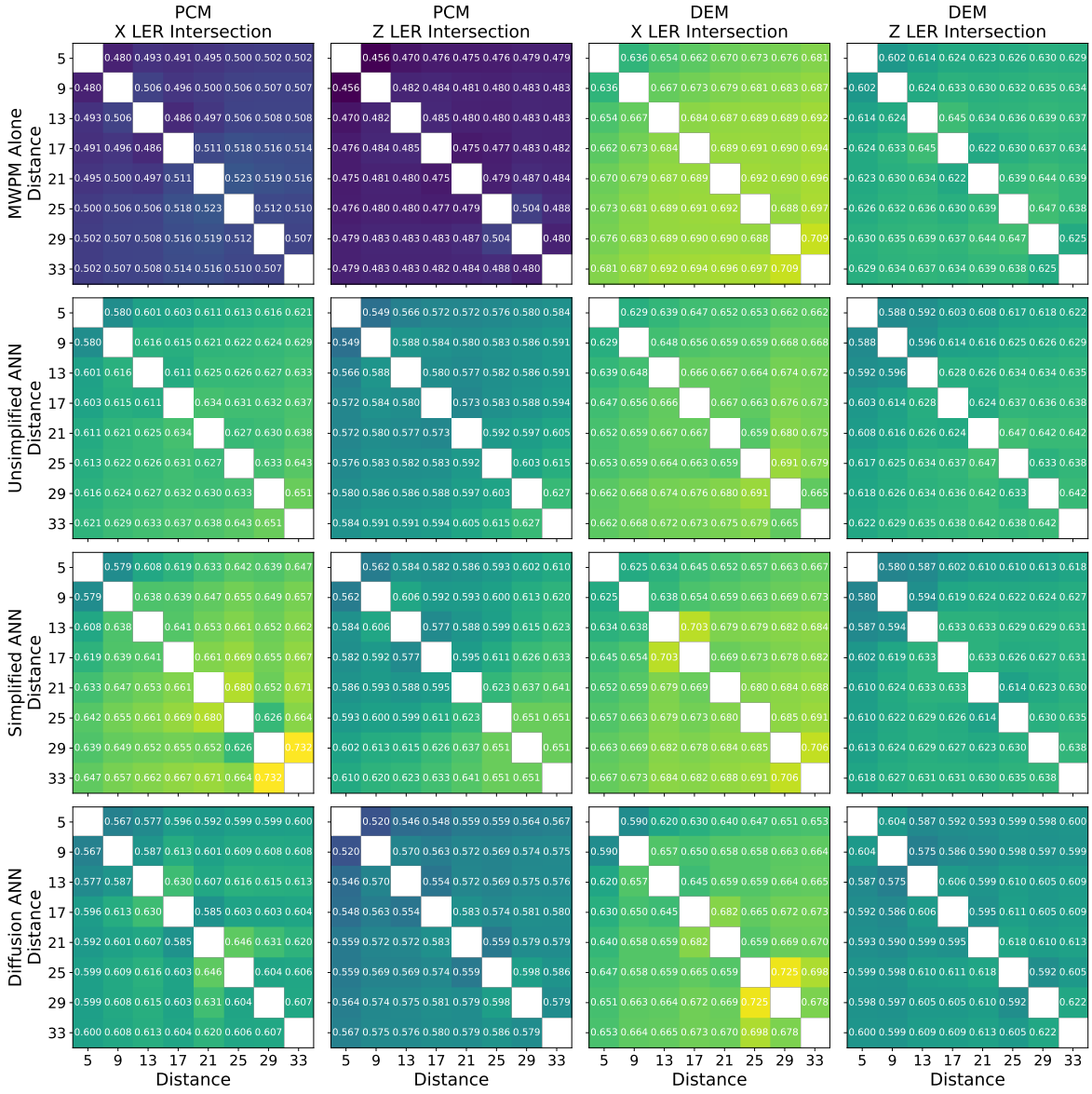


**Figure 19.** Matrices showing intersections between LER curves for rotated surface codes of distances between 5 and 33 when decoded with PCM or DEM PyMatching (MWPM) alone and the three ANN decoders presented in this work. Text in each cell shows the depolarisation parameter value rounded to three decimal places based on extrapolation of local error rates calculated at depolarisation parameters at the nearest two tenth of a percent.

*2.4. Decoding Timing*

Decoder latency was tested by recording the time taken to calculate output corrections or logical corrections from the time the input data is prepared in an appropriate format. When performing these measurements, it was found that a significant dependence on input batch size was present, suggesting a significant overhead present in the benchmarking pipeline which does not grow with increasing batch size. To remove this initial overhead, which is not expected to be present in systems running syndrome decoding continuously during a large QEC benchmark, the time per shot was set as the linear dependence associated with a fit when batch sizes of 32, 64, 128, 192,and 256 were used. Results are shown in Figure 20 comparing PCM PyMatching with unsimplified ANN decoding with PCM PyMatching performing mop-up. We find that, for the test setup, composed of an Intel Core i9-13900K with an Nvidia GeForce RTX 4070 Ti, improvements to decoding times begin to appear only when above threshold and at code distances beyond $d = 30$. This suggests that with moderate further optimisations, decoding speedups may be possible when utilising modest hardware. It should be noted that the time per shot, corresponding to the time needed to decode $n_c = d$ cycles, is still in the millisecond regime. Significant optimisations, perhaps with the use of dedicated hardware of Field Programmable Gate Arrays (FPGAs) and Application-Specific Integrated Circuits (ASICs) may be needed to bring the total latency down to the microsecond regime. Results for larger distances are shown in Figure 21. The memory demands of larger distance simulations prohibit the higher batch-sizes necessary to find latencies by linear fits, and so these results show full times which include non-batch-dependent overheads. We find that crossover points do exist when ANN-based decoding is expected to be faster that PCM and DEM PyMatching alone. Crossover points appear to occur at lower distances for higher error rates. We note an unexpected sensitivity of the ANN part of computations, especially at lower error rates, appears in the data. This may be an artefact of the testing process, as the ANN should be performing the same calculations irrespective of the mop-up component which would follow.

Finally, we investigated the speedup associated with the global decoding task with and without the assistance of the ANN. Figure 22 shows the relative time taken during matching when using PCM PyMatching as a global decoder compared with ANN assistance. We find that relative time is consistently less than one, showing that a significant portion of the decoding problem is handled by the ANN. We find that relative time tends to increase for increasing noise intensities. We can also see that for each different noise model, relative decoding times tend to be lower for larger distances and greater for smaller distances. This suggests that significant advantages may be possible as long as sufficient optimisations are applied to make the highly parallel ANN operations contribute negligibly to the total decoding time.

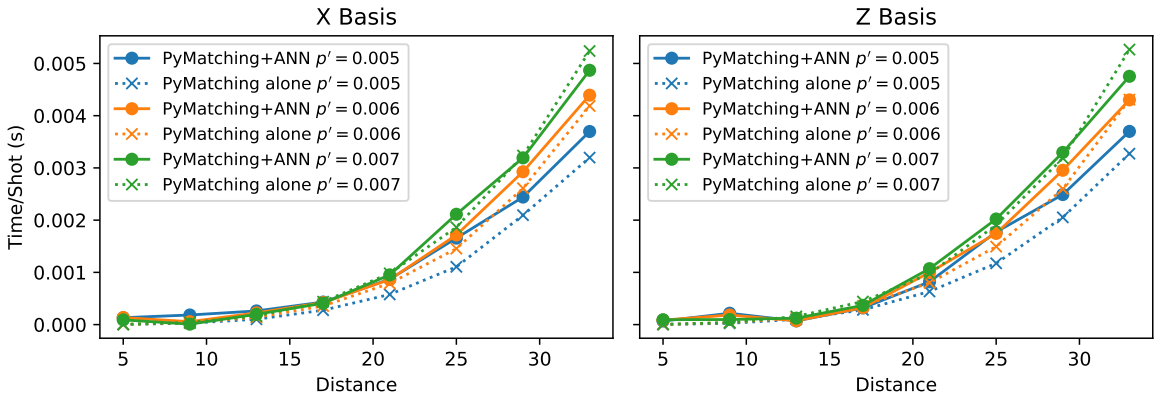


**Figure 20.** Decoding times for PCM PyMatching decoding compared with multi-label ANN decoding with PCM PyMatching for $X$ and $Z$ basis memory experiments with circuit noise depolarisation parameters near threshold, $p'_{\text{th}} \approx 0.006$.

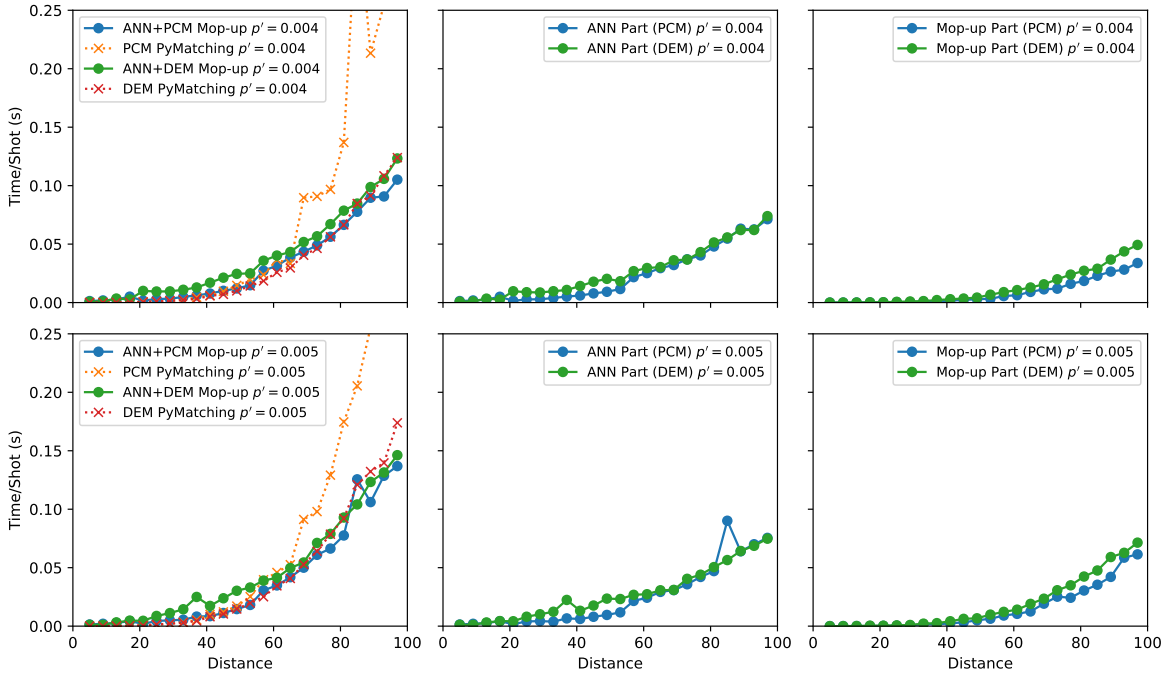


**Figure 21.** Decoding times for PCM PyMatching and DEM PyMatching decoding compared with multi-label ANN decoding with the same decoders used for mop-up for $Z$ basis memory experiments with uniform depolarisation parameter circuit noise. Results are shown for depolarisation parameters $p' = 0.004$ and $p' = 0.005$, corresponding to below-threshold performance.

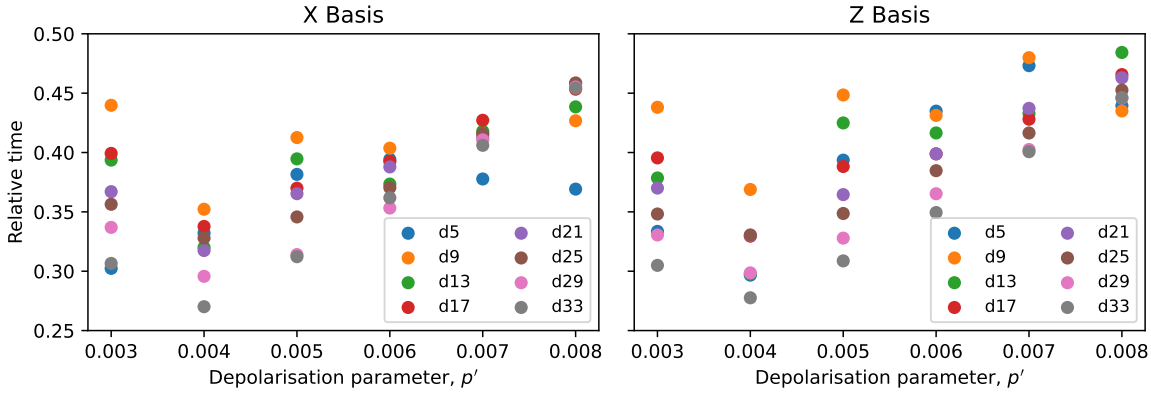


**Figure 22.** Relative matching decoding times for PCM PyMatching decoding compared with multi-label ANN decoding with PCM PyMatching for $X$ and $Z$ basis memory experiments with depolarisation parameter, $p'$, circuit noise.

### *2.5. Comparison With Previous Works*

The most direct comparison of our work to previous research on ANN decoding at large distances is with Meinerz *et al.* [58] and Chamberland *et al.* [45]. A summary is provided in Table 1. Meinerz *et al.*used fully connected ANNs to form predictions for corrections near detection events on surface codes on a torus (toric codes). The application of dense networks near detection events has the advantage of requiring ANN computations only at a small subset of all correction locations, rather than for the whole syndrome history presented here. Considering surface codes defined on a torus has the advantage of perfect periodic boundary conditions, avoiding complications associated with boundaries. The error model considered was depolarising noise on data qubits with syndrome measurement errors on ancilla qubits. Meinerz *et al.*considered PyMatching MWPM and UF as global decoders, applied after the dense ANN. Timing measurements for ANN-assisted PyMatching decoding yielded times of 14.6 ms and 229 ms for error rates of $p = 0.01$ and $p = 0.0378$, respectively. Their implementation of PyMatching alone gave decoding latencies of 211 ms and 294 ms, consistent with the sparse syndrome targeted ANN decoding strategy. It should be noted that a slightly lower threshold was achieved when utilising the speedup of the ANN decoder before PyMatching. When applied with UF, results showed the use of the ANN offered improvements to decoding times, lowering latencies from 11.5 ms to 11.1 ms at $p = 0.01$ and 18.9 ms to 17.8 ms at $p = 0.0378$. Although these offer significantly less improvements than when the ANN was paired with PyMatching, utilising the ANN with UF yields an improved threshold from $p = 0.0379$ to $p = 0.0434$.

Chamberland *et al.* [45] instead considered FCNN decoding of rotated surface codes. The noise model considered was a variant of circuit noise, featuring only one instance of idle noise on data qubits each cycle. The ANN decoder under consideration was

**Table 1.** A comparison between this work and prior work on scalable ANN-based syndrome decoders for QEC codes with syndrome errors.

| Paper | QEC Code | $d_{max}$ | $p_{th}$ | ML Technique | Noise model |
|---|---|---|---|---|---|
| Meinerz *et al.* (2018) [58] | Toric code | 63 | 0.044 | Fully connected ANN | Phenomenological noise |
| Chamberland *et al.* (2018) [45] | Rotated surface code | 17 | 0.005 | FCNN | Circuit noise (uniform error rate) |
| This work. | Rotated surface code | 97 | 0.006 | FCNN, Diffusion model | Circuit noise (uniform depolarisation parameter) |

composed on multiple 3D convolution layers and was used to predict sites of data qubit errors based on syndrome measurement input data. As ancilla qubit errors were unable to be specified, heuristic methods were used to account for them before residual syndromes were processed by a global decoder. At low error rates of $p = 0.001$, up to 98% of nontrivial syndrome bits were able to be decoded, which was argued to suggest global decoder speedups by up to a factor of $10^6$. However, similar to the work of Meinerz *et al.* [58], when global decoding was performed with MWPM, as implemented with PyMatching, threshold error rates decreased slightly from $p = 0.007$ to $p = 0.005$ when comparing PyMatching alone to using an ANN. Chamberland *et al.*also evaluated the latency of their ANN when implemented with FPGAs and found a total latency of 0.673 ms for 17 syndrome measurement rounds of a $d = 17$ code.

### *2.6. Comparison With Currently Available Devices*

Varying characteristics of hardware platforms supporting QEC impose different requirements for the surface code decoding problem. The major differences derive from qubit and gate error characteristics, two-qubit connectivity, gate times and classical resource availability. Here, we will give a brief discussion of the characteristics of currently available superconducting and trapped atom devices to put in context the results presented in this work.

Superconducting quantum devices, such as those developed by IBM and Google [59, 60], possess amongst the fastest gate times, with two qubit gates possible at tens of nanoseconds and measurements at hundreds of nanoseconds. These devices are expected to output syndrome data at $1\mu$s per cycle. The error rates of the best available superconducting quantum devices are currently overall slightly below threshold, but feature significant inhomogeneity between contributions from two-qubit gates, single qubit gates, and SPAM errors. Moreover, these devices are known to experience significant contributions from other error sources, such as cross-talk, leakage and cosmic ray events [61]. Such devices have grown to sizes featuring beyond a hundred qubits, and have demonstrated surface code logical qubits of distances up to $d = 7$ [62]. Connectivity to control electronics is expected to be a significant challenge as such devices continue to scale to sizes featuring multiple logical qubits of distances up to approximately $d = 33$. Such overheads are expected to be required for the largest algorithms under pessimistic estimates for achievable physical error rates of approximately $p = 0.1\%$. In the near

future, when devices grow to sizes corresponding to distances supporting beyond $d = 13$, utilising 3D convolutional ANNs may offer some speed-ups. However, this would require latencies of hardware implementations of ANNs to be negligible compared to the global decoder latency. Additionally, as the noise of such devices can be expected to remain significantly structured, additional research investigating adaptations to structured noise models may be needed to maintain competitive performance compared to weighted matching methods alone.

Trapped atom/ion devices, such as those developed by Quantinuum and Quera, possess much longer qubit coherence times, but also suffer from much longer gate times [63, 64]. A major advantage of such devices compared to other platforms comes from the all-to-all connectivity possible from long range interactions and movement of atomic qubits in optical lattices. The increased connectivity compared to local architectures can be used to implement transversal two-qubit logical operations. When paired with simultaneous decoding of correlated errors, and used of optimised compiling, this has the advantage of overcoming the negative impact of slower gate times [65, 66]. The convolutional decoder presented here has not been optimised for the simultaneous decoding of multiple logical qubits with correlated errors. However, single qubit decoding during idle times may benefit from convolutional ANN decoding speedups, especially if large delays are caused by optimal correlated decoding after transversal logical operations.

## 3. Discussion and Outlook

In this work we investigated a 3D convolutional approach to ANN-based, circuit noise surface code decoding. We described a vectorised method of data preparation, corresponding to parallel propagation of errors to reference time steps and how decoding can be interpreted as a multi-label classification and generative modelling problem. We showed how this allows an ANN based on three-dimensional convolutional layers to be trained on such a task, and evaluated performance with respect to accuracy and latency. We find that such a decoder exhibits competitive performance which may be valuable addition to experimental QEC architectures performing the decoding required for Fault-Tolerant Quantum Computation (FTQC).

There are many ways our work can be extended. On the decoding side, further software optimisations, such as different unit cells, quantization, pruning and parallelisation may further improve desktop performance. Additionally, optimisation can be applied to generative approaches to decoding, such as utilizing noise schedules for diffusion models or avoiding the need to repeated passes by using Generative Adversarial Networks (GANs) instead. Further improvements to performance may also be sought when implemented on dedicated hardware. A complete evaluation when implemented with dedicated hardware would also provide strong information regarding the presence of any bottlenecks present during computation, such as data transfer. Our network was shown to operate only on uniform surface codes, where stabilisers form a checker-board pattern. Some lattice surgery configurations, which are needed to apply a universal set

of logical operators, can also make use of domain walls and twists. An implementation with modified unit cell may be able to accommodate such codes, with generalisation expected to follow if layers remain exclusively convolutional. Additionally, the network can be optimised to work with data when received as a data stream, a possibility readily enabled by the local computational structure in space and time. Once this is achieved, decoding general lattice surgery operations, changing configuration with time, may be possible. On the simulation side, further work can be done to extend the results to other periodic QEC codes such as colour codes.

In summary, the techniques developed in this work show that neural-network based decoding can remain competitive and scalable up to code distances relevant to those expected to yield quantum advantage. With further development, such methods form candidates to assist in the classical data processing demanded by QEC in practical settings.

## 4. Data availability statement

The data that support the findings of this study are available on request from the corresponding author upon reasonable request.

## 5. Acknowledgements

This work was supported by the Australian Research Council funded Center for Quantum Computation and Communication Technology (CE170100012).

## 6. References

[1] Peter W. Shor. "Scheme for reducing decoherence in quantum computer memory". Phys. Rev. A **52**, R2493–R2496 (1995).

[2] P.W. Shor. "Fault-tolerant quantum computation". In Proceedings of 37th Conference on Foundations of Computer Science. Pages 56–65. (1996).

[3] Craig Gidney and Martin Ekerå. "How to factor 2048 bit rsa integers in 8 hours using 20 million noisy qubits". Quantum **5**, 433 (2021).

[4] Markus Reiher, Nathan Wiebe, Krysta M. Svore, Dave Wecker, and Matthias Troyer. "Elucidating reaction mechanisms on quantum computers". Proceedings of the National Academy of Sciences **114**, 7555–7560 (2017).

[5] D. S. Wang, A. G. Fowler, A. M. Stephens, and L. C. L. Hollenberg. "Threshold error rates for the toric and planar codes". Quantum Info. Comput. **10**, 456–469 (2010).

[6] Austin G. Fowler. "Proof of finite surface code threshold for matching". Phys. Rev. Lett. **109**, 180502 (2012).

[7] Alexander Erhard, Hendrik Poulsen Nautrup, Michael Meth, Lukas Postler, Roman Stricker, Martin Stadler, Vlad Negnevitsky, Martin Ringbauer, Philipp Schindler, Hans J. Briegel, Rainer Blatt, Nicolai Friis, and Thomas Monz. "Entangling logical qubits with lattice surgery". Nature **589**, 220–224 (2021).

[8] Dolev Bluvstein, Simon J. Evered, Alexandra A. Geim, Sophie H. Li, Hengyun Zhou, Tom Manovitz, Sepehr Ebadi, Madelyn Cain, Marcin Kalinowski, Dominik Hangleiter, J. Pablo Bonilla Ataides, Nishad Maskara, Iris Cong, Xun Gao, Pedro Sales Rodriguez, Thomas Karolyshyn, Giulia

Semeghini, Michael J. Gullans, Markus Greiner, Vladan Vuletić, and Mikhail D. Lukin. "Logical quantum processor based on reconfigurable atom arrays". Nature **626**, 58–65 (2024).

[9] Nicolas Delfosse, Andres Paz, Alexander Vaschillo, and Krysta M. Svore. "How to choose a decoder for a fault-tolerant quantum computer? the speed vs accuracy trade-off" (2023). arXiv:2310.15313.

[10] F Battistel, C Chamberland, K Johar, R W J Overwater, F Sebastiano, L Skoric, Y Ueno, and M Usman. "Real-time decoding for fault-tolerant quantum computing: progress, challenges and outlook". Nano Futures **7**, 032003 (2023).

[11] Poulami Das, Aditya Locharla, and Cody Jones. "Lilliput: a lightweight low-latency lookup-table decoder for near-term quantum error correction". In Proceedings of the 27th ACM International Conference on Architectural Support for Programming Languages and Operating Systems. Page 541–553. ASPLOS '22New York, NY, USA (2022). Association for Computing Machinery.

[12] Sergey Bravyi, Martin Suchara, and Alexander Vargo. "Efficient algorithms for maximum likelihood decoding in the surface code". Phys. Rev. A **90**, 032326 (2014).

[13] Austin G. Fowler, Adam C. Whiteside, and Lloyd C. L. Hollenberg. "Towards practical classical processing for the surface code". Phys. Rev. Lett. **108**, 180501 (2012).

[14] Nicolas Delfosse and Naomi H. Nickerson. "Almost-linear time decoding algorithm for topological codes". Quantum **5**, 595 (2021).

[15] Giacomo Torlai and Roger G. Melko. "Neural decoder for topological codes". Phys. Rev. Lett. **119**, 030501 (2017).

[16] Johannes Bausch, Andrew W. Senior, Francisco J. H. Heras, Thomas Edlich, Alex Davies, Michael Newman, Cody Jones, Kevin Satzinger, Murphy Yuezhen Niu, Sam Blackwell, George Holland, Dvir Kafri, Juan Atalaya, Craig Gidney, Demis Hassabis, Sergio Boixo, Hartmut Neven, and Pushmeet Kohli. "Learning high-accuracy error decoding for quantum processors". Nature **635**, 834–840 (2024).

[17] Luka Skoric, Dan E. Browne, Kenton M. Barnes, Neil I. Gillespie, and Earl T. Campbell. "Parallel window decoding enables scalable fault tolerant quantum computation". Nature Communications **14**, 7040 (2023).

[18] John Blue, Harshil Avlani, Zhiyang He, Liu Ziyin, and Isaac L. Chuang. "Machine learning decoding of circuit-level noise for bivariate bicycle codes" (2025). arXiv:2504.13043.

[19] Hanyan Cao, Feng Pan, Dongyang Feng, Yijia Wang, and Pan Zhang. "Generative decoding for quantum error-correcting codes" (2025). arXiv:2503.21374.

[20] Lukas Bödeker, Luc J. B. Kusters, and Markus Müller. "On the interpretability of neural network decoders" (2025). arXiv:2502.20269.

[21] Gengyuan Hu, Wanli Ouyang, Chao-Yang Lu, Chen Lin, and Han-Sen Zhong. "Efficient and universal neural-network decoder for stabilizer-based quantum error correction" (2025). arXiv:2502.19971.

[22] Hao Wang, Erjia Xiao, Songhuan He, Zhongyi Ni, Lingfeng Zhang, Xiaokun Zhan, Yifei Cui, Jinguo Liu, Cheng Wang, Zhongrui Wang, and Renjing Xu. "Cim-based parallel fully ffnn surface code high-level decoder for quantum error correction". In 2025 Design, Automation & Test in Europe Conference (DATE). Pages 1–2. (2025).

[23] Oliver Weißl and Evgenii Egorov. "An equivariant machine learning decoder for 3d toric codes". In 2025 International Conference on Quantum Communications, Networking, and Computing (QCNC). Pages 672–676. (2025).

[24] Arshpreet Singh Maan and Alexandru Paler. "Machine learning message-passing for the scalable decoding of qldpc codes". npj Quantum Information **11**, 78 (2025).

[25] Vukan Ninkovic, Ognjen Kundacina, Dejan Vukobratovic, Christian Häger, and Alexandre Graell i Amat. "Decoding quantum ldpc codes using graph neural networks". In GLOBECOM 2024 - 2024 IEEE Global Communications Conference. Pages 3479–3484. (2024).

[26] Hany Ali, Jorge Marques, Ophelia Crawford, Joonas Majaniemi, Marc Serra-Peralta, David Byfield, Boris Varbanov, Barbara M. Terhal, Leonardo DiCarlo, and Earl T. Campbell. "Reducing the error rate of a superconducting logical qubit using analog readout information". Phys. Rev. Appl.

**22**, 044031 (2024).

[27] Weishun Zhong, Oles Shtanko, and Ramis Movassagh. "Advantage of quantum neural networks as quantum information decoders" (2024). arXiv:2401.06300.

[28] Simone Bordoni and Stefano Giagu. "Convolutional neural network based decoders for surface codes". Quantum Information Processing **22**, 151 (2023).

[29] Hanrui Wang, Pengyu Liu, Kevin Shao, Dantong Li, Jiaqi Gu, David Z. Pan, Yongshan Ding, and Song Han. "Transformer-qec: Quantum error correction code decoding with transferable transformers" (2023). arXiv:2311.16082.

[30] Anqi Gong, Sebastian Cammerer, and Joseph M. Renes. "Graph neural networks for enhanced decoding of quantum ldpc codes". In 2024 IEEE International Symposium on Information Theory (ISIT). Pages 2700–2705. (2024).

[31] Brhyeton Hall, Spiro Gicev, and Muhammad Usman. "Artificial neural network syndrome decoding on ibm quantum processors". Phys. Rev. Res. **6**, L032004 (2024).

[32] Hanyan Cao, Feng Pan, Yijia Wang, and Pan Zhang. "qecgpt: decoding quantum error-correcting codes with generative pre-trained transformers" (2023). arXiv:2307.09025.

[33] Boris M. Varbanov, Marc Serra-Peralta, David Byfield, and Barbara M. Terhal. "Neural network decoder for near-term surface-code experiments". Phys. Rev. Res. **7**, 013029 (2025).

[34] Moritz Lange, Pontus Havström, Basudha Srivastava, Isak Bengtsson, Valdemar Bergentall, Karl Hammar, Olivia Heuts, Evert van Nieuwenburg, and Mats Granath. "Data-driven decoding of quantum error correcting codes using graph neural networks". Phys. Rev. Res. **7**, 023181 (2025).

[35] Evgenii Egorov, Roberto Bondesan, and Max Welling. "The end: An equivariant neural decoder for quantum error correction" (2023). arXiv:2304.07362.

[36] Yoni Choukroun and Lior Wolf. "Deep quantum error correction". Proceedings of the AAAI Conference on Artificial Intelligence **38**, 64–72 (2024).

[37] Ramon W. J. Overwater, Masoud Babaie, and Fabio Sebastiano. "Neural-network decoders for quantum error correction using surface codes: A space exploration of the hardware cost-performance tradeoffs". IEEE Transactions on Quantum Engineering **3**, 1–19 (2022).

[38] Pierre-Antoine Mouny, Maher Benhouria, Victor Yon, Patrick Dufour, Linxiang Huang, Yann Beilliard, Sophie Rochette, Dominique Drouin, and Pooya Ronagh. "Towards a cryogenic cmos-memristor neural decoder for quantum error correction". In 2024 IEEE International Conference on Quantum Computing and Engineering (QCE). Volume 01, pages 1258–1263. (2024).

[39] Victor Yon, Frédéric Marcotte, Pierre-Antoine Mouny, Gebremedhin A Dagnew, Bohdan Kulchytskyy, Sophie Rochette, Yann Beilliard, Dominique Drouin, and Pooya Ronagh. "A memristive neural decoder for cryogenic fault-tolerant quantum error correction". Quantum Science and Technology **10**, 025049 (2025).

[40] Mengyu Zhang, Xiangyu Ren, Guanglei Xi, Zhenxing Zhang, Qiaonian Yu, Fuming Liu, Hualiang Zhang, Shengyu Zhang, and Yi-Cong Zheng. "A scalable, fast and programmable neural decoder for fault-tolerant quantum computation using surface codes" (2023). arXiv:2305.15767.

[41] Yosuke Ueno, Masaaki Kondo, Masamitsu Tanaka, Yasunari Suzuki, and Yutaka Tabuchi. "Neo-qec: Neural network enhanced online superconducting decoder for surface codes" (2022). arXiv:2208.05758.

[42] Xiaotong Ni. "Neural network decoders for large-distance 2d toric codes". Quantum **4**, 310 (2020).

[43] Kai Meinerz, Chae-Yeun Park, and Simon Trebst. "Scalable neural decoder for topological surface codes". Phys. Rev. Lett. **128**, 080505 (2022).

[44] Spiro Gicev, Lloyd C. L. Hollenberg, and Muhammad Usman. "A scalable and fast artificial neural network syndrome decoder for surface codes". Quantum **7**, 1058 (2023).

[45] Christopher Chamberland, Luis Goncalves, Prasahnt Sivarajah, Eric Peterson, and Sebastian Grimberg. "Techniques for combining fast local decoders with global decoders under circuit-level noise". Quantum Science and Technology **8**, 045011 (2023).

[46] Eric Dennis, Alexei Kitaev, Andrew Landahl, and John Preskill. "Topological quantum memory". Journal of Mathematical Physics **43**, 4452–4505 (2002).

[47] Yu Tomita and Krysta M. Svore. "Low-distance surface codes under realistic quantum noise". Phys. Rev. A **90**, 062320 (2014).

[48] Wim van Dam, Mariia Mykhailova, and Mathias Soeken. "Using azure quantum resource estimator for assessing performance of fault tolerant quantum computation". In Proceedings of the SC '23 Workshops of the International Conference on High Performance Computing, Network, Storage, and Analysis. Page 1414–1419. SC-W '23New York, NY, USA (2023). Association for Computing Machinery.

[49] Craig Gidney and Martin Ekerå. "How to factor 2048 bit RSA integers in 8 hours using 20 million noisy qubits". Quantum **5**, 433 (2021).

[50] György P. Gehér, Campbell McLauchlan, Earl T. Campbell, Alexandra E. Moylett, and Ophelia Crawford. "Error-corrected Hadamard gate simulated at the circuit level". Quantum **8**, 1394 (2024).

[51] Keisuke Fujii and Yuuki Tokunaga. "Error and loss tolerances of surface codes with general lattice structures". Phys. Rev. A **86**, 020303 (2012).

[52] Emanuel Knill, Raymond Laflamme, and Wojciech H. Zurek. "Resilient quantum computation: error models and thresholds". Proceedings of the Royal Society of London. Series A: Mathematical, Physical and Engineering Sciences **454**, 365–384 (1998).

[53] Austin G. Fowler, Matteo Mariantoni, John M. Martinis, and Andrew N. Cleland. "Surface codes: Towards practical large-scale quantum computation". Phys. Rev. A **86**, 032324 (2012).

[54] Craig Gidney. "Stim: a fast stabilizer circuit simulator". Quantum **5**, 497 (2021).

[55] Oscar Higgott. "Pymatching: A python package for decoding quantum codes with minimum-weight perfect matching". ACM Transactions on Quantum Computing**3** (2022).

[56] Ashley M. Stephens. "Fault-tolerant thresholds for quantum error correction with the surface code". Phys. Rev. A **89**, 022321 (2014).

[57] Jonathan Ho, Ajay Jain, and Pieter Abbeel. "Denoising diffusion probabilistic models". In Proceedings of the 34th International Conference on Neural Information Processing Systems. NIPS '20Red Hook, NY, USA (2020). Curran Associates Inc.

[58] Kai Meinerz, Chae-Yeun Park, and Simon Trebst. "Scalable neural decoder for topological surface codes". Phys. Rev. Lett. **128**, 080505 (2022).

[59] Riddhi S. Gupta, Neereja Sundaresan, Thomas Alexander, Christopher J. Wood, Seth T. Merkel, Michael B. Healy, Marius Hillenbrand, Tomas Jochym-O'Connor, James R. Wootton, Theodore J. Yoder, Andrew W. Cross, Maika Takita, and Benjamin J. Brown. "Encoding a magic state with beyond break-even fidelity". Nature **625**, 259–263 (2024).

[60] Rajeev Acharya et al. "Suppressing quantum errors by scaling a surface code logical qubit". Nature **614**, 676–681 (2023).

[61] Zijun Chen, Kevin J Satzinger, Juan Atalaya, Alexander N Korotkov, Andrew Dunsworth, Daniel Sank, Chris Quintana, Matt McEwen, Rami Barends, Paul V Klimov, et al. "Exponential suppression of bit or phase errors with cyclic error correction". Nature **595**, 383–387 (2021).

[62] Rajeev Acharya et al. "Quantum error correction below the surface code threshold". Nature **638**, 920–926 (2025).

[63] Simon J. Evered, Dolev Bluvstein, Marcin Kalinowski, Sepehr Ebadi, Tom Manovitz, Hengyun Zhou, Sophie H. Li, Alexandra A. Geim, Tout T. Wang, Nishad Maskara, Harry Levine, Giulia Semeghini, Markus Greiner, Vladan Vuletić, and Mikhail D. Lukin. "High-fidelity parallel entangling gates on a neutral-atom quantum computer". Nature **622**, 268–272 (2023).

[64] Karl Mayer, Ciarán Ryan-Anderson, Natalie Brown, Elijah Durso-Sabina, Charles H. Baldwin, David Hayes, Joan M. Dreiling, Cameron Foltz, John P. Gaebler, Thomas M. Gatterman, Justin A. Gerber, Kevin Gilmore, Dan Gresh, Nathan Hewitt, Chandler V. Horst, Jacob Johansen, Tanner Mengle, Michael Mills, Steven A. Moses, Peter E. Siegfried, Brian Neyenhuis, Juan Pino, and Russell Stutz. "Benchmarking logical three-qubit quantum fourier transform encoded in the steane code on a trapped-ion quantum computer" (2024). arXiv:2404.08616.

[65] Daniel Litinski and Naomi Nickerson. "Active volume: An architecture for efficient fault-tolerant

quantum computers with limited non-local connections" (2022). arXiv:2211.15465.

[66] Madelyn Cain, Chen Zhao, Hengyun Zhou, Nadine Meister, J. Pablo Bonilla Ataides, Arthur Jaffe, Dolev Bluvstein, and Mikhail D. Lukin. "Correlated decoding of logical algorithms with transversal gates". Phys. Rev. Lett. **133**, 240602 (2024).