

Article

Not peer-reviewed version

Quantum and Quantum-Inspired Stereographic K Nearest-Neighbour Clustering

[Ark Kumar Modi](#)^{*}, [Alonso Viladomat Jasso](#)^{*}, [Roberto Ferrara](#), Janis Noetzel, [Christian Deppe](#), [Maximilian Schädler](#), [Fred Fung](#)

Posted Date: 19 September 2023

doi: 10.20944/preprints202305.0051.v3

Keywords: Quantum K-Means; Quantum Machine Learning; Quantum Computing; K-Means Clustering; 6G Communication; Quadrature Amplitude Modulation; Quantum-Classical Hybrid Algorithms; Quantum-Inspired Algorithms



Preprints.org is a free multidiscipline platform providing preprint service that is dedicated to making early versions of research outputs permanently available and citable. Preprints posted at Preprints.org appear in Web of Science, Crossref, Google Scholar, Scilit, Europe PMC.

Copyright: This is an open access article distributed under the Creative Commons Attribution License which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Disclaimer/Publisher's Note: The statements, opinions, and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions, or products referred to in the content.

Article

Quantum and Quantum-Inspired Stereographic K Nearest-Neighbour Clustering

Alonso Viladomat Jasso ^{2,*†}, Ark Modi ^{1,*†}, Roberto Ferrara ¹, Christian Deppe ¹, Janis Nötzel ², Fred Fung ³ and Maximilian Schädler ³

¹ Institute for Communications Engineering, TUM School of Computation, Information and Technology, Technical University of Munich; firstname.lastname@tum.de

² Emmy Noether group for Theoretical Quantum System Design, Chair of Theoretical Information Technology, Technical University of Munich; firstname.lastname@tum.de

³ Optical and Quantum Laboratory, Munich Research Center, Huawei Technologies Düsseldorf GmbH, Riesstr. 25-C3, 80992 Munich, Germany

* Correspondence: viladomat.jasso@tum.de, ark.modi@tum.de

† These authors contributed equally to the work

Abstract: Nearest-neighbour clustering is a simple yet powerful machine learning algorithm that finds natural application in the decoding of signals in classical optical-fibre communication systems. Quantum k-means clustering promises a speed-up over the classical k-means algorithm; however, it has been shown to currently not provide this speed-up for decoding optical-fibre signals due to the embedding of classical data, which introduces inaccuracies and slowdowns. Although still not achieving an exponential speed-up for NISQ implementations, this work proposes the generalised inverse stereographic projection as an improved embedding into the Bloch sphere for quantum distance estimation in k-nearest-neighbour clustering, which allows us to get closer to the classical performance. We also use the generalised inverse stereographic projection to develop an analogous classical clustering algorithm and benchmark its accuracy, runtime and convergence for decoding real-world experimental optical-fibre communication data. This proposed 'quantum-inspired' algorithm provides an improvement in both the accuracy and convergence rate with respect to the k-means algorithm. Hence, this work presents two main contributions. Firstly, we propose the general inverse stereographic projection into the Bloch sphere as a better embedding for quantum machine learning algorithms; here, we use the problem of clustering quadrature amplitude modulated optical-fibre signals as an example. Secondly, as a purely classical contribution inspired by the first contribution, we propose and benchmark the use of the general inverse stereographic projection and spherical centroid for clustering optical-fibre signals, showing that optimizing the radius yields a consistent improvement in accuracy and convergence rate.

Keywords: quantum k nearest-neighbour; quantum machine learning; quantum computing; k-means clustering; 6G communication; quadrature amplitude modulation; quantum-classical hybrid algorithms; quantum-inspired algorithms

1. Introduction

Quantum Machine Learning (QML), using quantum algorithms to learn quantum or classical systems, has attracted much research in recent years, with some algorithms possibly gaining an exponential speedup [1–3]. Since machine learning routines often push real-world limits of computing power, an exponential improvement to algorithm speed would allow for such systems with vastly greater capabilities [4]. Google's 'Quantum Supremacy' experiment [5] showed that quantum computers can naturally solve specific problems with complex correlations between inputs that can be incredibly hard for traditional ("classical") computers. Such a result suggests that machine learning models executed on quantum computers could be more effective for specific applications. It seems quite possible that quantum computing could lead to faster computation, better generalisation

on less data, or both, for an appropriately designed learning model. Hence, it is of great interest to discover and model the scenarios in which such a “quantum advantage” could be achieved. A number of such “Quantum Machine Learning” algorithms are detailed in papers such as [2,6–9]. Many of these methods claim to offer exponential speedups over analogous classical algorithms. However, some significant gaps exist between theoretical prediction and implementation on the path from theory to technology. These gaps result in unforeseen technological hurdles and sometimes misconceptions, necessitating more careful case-by-case studies such as [10].

It is known from the literature that the k -nearest-neighbour clustering algorithm (kNN) can be applied to solve the problem of phase estimation in optical fibres [11,12]. A quantum version of this kNN has been developed in [2], promising an exponential speedup. However, the practical usefulness of this algorithm is under debate [13]. The encoding of classical data into quantum states has been proven to be a complex task which significantly reduces the advantage of known quantum machine learning algorithms [13]. There are claims that the speedup is reduced to only polynomial once the quantum version of the algorithm takes into account the time taken to prepare the necessary quantum states. Furthermore, for near-intermediate scale quantum (NISQ) [3] applications, we should not expect the availability of QRAM, as this assumes reliable memories and operations which are still several milestones out of reach [14]. For this reason, it is not currently possible to use the fully quantum clustering algorithm and thus we resort to using *hybrid quantum-classical* kNN algorithms. Any classical implementation of kNN clustering involves, among other steps, repeated evaluations of a dissimilarity and a loss function; changing the dissimilarity leads to a different clustering. A hybrid quantum classical kNN clustering algorithm utilizes quantum methods only to estimate the dissimilarity, eliminating the need for long-lasting quantum memories. However, reproducing the dissimilarity of a classical kNN algorithm using quantum methods can be prohibitively restrictive. The quantum dissimilarity also depends on the embedding (how the classical data is encoded in quantum states) and might only approximate the classical one, introducing fundamental deviations from the classical kNN algorithm. In [10], we applied a hybrid quantum-classical algorithm with modified angle embedding to the problem of k -means clustering for 64-QAM (Quadrature Amplitude Modulation) optical-fibre data (a well-known technical problem in signal processing through optical-fibre communication links) provided by Huawei [15], and show that this currently does not yield an advantage due to both the embedding and the current speed and noise of quantum devices.

In this work, we use the same problem and datasets to bring two main but independent contributions using the generalised inverse stereographic projection. First, we embed classical 2-dimensional data by computing the ISP onto the 3-dimensional sphere, and use the resulting normalised vector as Bloch vector to produce a pure quantum state of one qubit, which we call stereographic embedding. The resulting quantum dissimilarity directly translates into the cosine dissimilarity, thus making the quantum algorithm mathematically closer to the classical k -means algorithm. This means that no inherent limitation is introduced by the embedding and any loss in performance of this hybrid algorithm can be compensated for by improving the noise level and the speed of the quantum device. We thus propose stereographic embedding as an improved quantum embedding that may lead to improvement in several quantum machine learning algorithms (although there might still not be a practical quantum time advantage).

The second contribution comes from the benchmarking of the hybrid stereographic quantum mentioned above. Since, as already mentioned, the resulting hybrid clustering algorithm is mathematically equivalent to a classical ‘quantum-inspired’ kNN algorithm, in order to assess its performance in the absence of noise we simply test the equivalent classical quantum-inspired kNN algorithm. This algorithm is the result of first computing the ISP of the data and then performing clustering using a novel ‘quantum’ centroid update. We observe an increase in accuracy and convergence performance over k -means clustering on the 2-dimensional optical-fibre data. This suggests, as a purely classical second main contribution, that an advantage in decoding 64-QAM

optical-fibre data is achieved by performing clustering in the inverse stereographically projected sphere and by using the spherical centroid.

The paper is structured as follows. In the remainder of this introduction, we discuss related works and our contribution to it. In Section 2, we introduce the experimental setup generating the 64-QAM optical-fibre transmission data and define clustering, the stereographic projection and the necessary quantum concepts for the hybrid protocols. Next, Section 3 introduces the developed Stereographic Quantum kNN (SQ-kNN), while Section 4 defines the developed quantum-inspired 2D Stereographic Classical kNN (2DSC-kNN) algorithm and proves its equivalence to the SQ-kNN quantum algorithm. In Section 5, we describe the various experiments for testing the algorithms, present the obtained results, and discuss their conclusions. We end the main text in Section 6 proposing some directions for future research, some of which are partially discussed in the appendix.

1.1. Related Work

A unifying overview of several quantum algorithms is presented in [16] in a tutorial style. An overview targeting data scientists is given in [17]. The idea of using quantum information processing methods to obtain speedups for the k-means algorithm was proposed in [18]. In general, neither the best nor even the fastest method for a given problem and problem size can be uniquely ascribed to either the class of quantum or classical algorithms, as seen in the detailed discussion presented in [9]. The advantages of using local (classical) processing units alongside quantum processing units in a distributed fashion are discussed in [19]. The accuracy of (quantum) K-means has been demonstrated experimentally in [20] and in [21], while quantum circuits for loading classical data into a quantum computer are described in [22].

An algorithm is proposed in [2] that solves the problem of clustering N -dimensional vectors to M clusters in $\mathcal{O}(\log(MN))$ time on a quantum computer, which is exponentially faster than $\mathcal{O}(\text{poly}(MN))$ time for the (then) best known classical algorithm. The approach detailed in [2] requires querying the QRAM [23] for preparing a ‘mean state’, which is then used to find the inner product between the centroid (by default the mean point) using the SWAP test [24–26]. However, there exist some significant caveats to this approach. Firstly, this algorithm achieves an exponential speedup only when comparing the bit-to-bit processing time with the qubit-to-qubit processing time. If one compares the bit-to-bit execution times of both algorithms, the exponential speedup disappears, as shown in [13,27]. Secondly, since stable enough quantum memories do not exist, a hybrid quantum-classical approach must be used in real-world applications. Namely, all the information must be stored in classical memories, and the states to be used in the algorithm are prepared in real time. The process of preparing quantum states from classical data is known as ‘Data Embedding’ since we are embedding the classical data into quantum states. This, as mentioned before [4,27], slows down the algorithm to only a polynomial advantage over classical k-means. However, we propose an approach whereby this step of embedding can be treated as a data pre-processing step, allowing us to achieve some advantages in accuracy and convergence rate, taking a step towards making the quantum approach more viable. Instead of using a quantum algorithm, classical alternatives mimicking their behaviour, collectively known as quantum-inspired algorithms, have shown much promise in achieving classically some types of advantage that are demonstrated by quantum algorithms [4,27–29], but as [9] remarks, the massive increase in runtime with rank, condition number, Frobenius norm, and error threshold make the algorithms proposed in [13,27] impractical for matrices arising from real-world applications. This observation is supported by [30].

Recent works such as [27] suggest that even the best QML algorithms, without state preparation assumptions, fail to achieve exponential speedups over their classical counterparts. In [4], it is pointed out that most QML algorithms are incomparable to classical algorithms since they take quantum states as input and output quantum states, and that there is no analogous classical model of computation where one could search for similar classical algorithms. In [4], the idea of matching state preparation assumptions with ℓ^2 -norm sampling assumptions (first proposed in [27]) is implemented

by introducing a new input model, *sample and query access* (SQ access). In [4], the Quantum K-Means algorithm described in [2] is 'de-quantised' using the 'toolkit' developed in [27], i.e. a classical quantum-inspired algorithm is given that, with classical SQ access assumptions replacing quantum state preparation assumptions, matches the bounds and runtime of the corresponding quantum algorithm up to polynomial slowdown. From the works [4,27,31], we can conclude that the exponential speedups of many quantum machine learning algorithms that are under consideration arise not from the 'quantumness' of the algorithms but instead from strong input assumptions, since the exponential part of the speedups vanish when classical algorithms are given analogous assumptions. In other words, in a wide array of settings, these algorithms do not give exponential speedups but rather yield polynomial speedups on classical data.

The fundamental aspect that allowed for the exponential speedup in [27] is exemplified by the problem of recommendation systems. The philosophy of classical recommendation algorithms before this breakthrough was to estimate all the possible preferences of a user and then suggest one or more of the most preferred objects. A quantum algorithm in [8] promised an exponential speedup but provided a recommendation without estimating all the preferences; namely, it only provided a *sample* of the most preferred objects. This process of sampling, along with state preparation assumptions, was, in fact, what gave the quantum algorithm an exponential advantage. The new classical algorithm obtains comparable speedups also by only providing samples rather than solving the whole preference problem. In [4], it is argued that the time taken to create the quantum state should be included for comparison since the time taken is not insignificant; it is also claimed that for every such linear algebraic quantum machine learning algorithm, a polynomially slower classical algorithm can be constructed by using the binary tree data structure described in [27]. Since then, more sampling algorithms have shown that multiple quantum exponential speedups are not due to the quantum algorithms themselves but due to the way data is provided to the algorithms and how the quantum algorithm provides the solutions [4,31–33]. Notably, in [33] it is argued that there exist competing classical algorithms for all linear algebraic subroutines and thus for many quantum machine learning algorithms. However, as pointed out in [9] and proven in [30], significant caveats exist to these aforementioned results of quantum-inspired algorithms. The polynomial factor in these algorithms often contains a very high power of the rank and condition number, making them suitable only for sparse low-rank matrices. Matrices of real-world data are often relatively high in rank and hence unfavourable for such sampling-based quantum-inspired approaches. Whether such sampling algorithms can be used also highly depends on the specific application and whether or not samples of the solution instead of the complete data are suitable. It should be pointed out that quantum machine learning algorithms generally do not provide an advantage if such complete data is needed.

The method of encoding classical data into quantum states contributes to the complexity and performance of the algorithm. An extensive analysis and testing of the hybrid quantum-classical implementation of the quantum k-means algorithm using angle embedding can be found in [10]. In this work, the use of the ISP is proposed. Others have explored this procedure [34–36] as well; however, the motivation, implementation, and use vary significantly, as well as the procedure for embedding data points into quantum states. There has also been no extensive testing of the proposed methods, especially not in an industry context. In our method, we exclusively use pure states from the Bloch sphere since this reduces the complexity of the application. Lemma 3 assures that our method with existing quantum techniques is applicable for nearest neighbour clustering. In contrast, the density matrices of mixed states and the normalised trace distance between the density matrices are used for binary classification in [34,35]. A crucial thing to consider here is to distinguish the contribution of the ISP from the quantum effects. We will see in Section 5 that the ISP seems to be the most important contributing factor. In [37], it is also proposed to encode classical information into quantum states using the ISP in the context of quantum generative adversarial networks. Their motivation for using the ISP is due to the fact that it is injective and can hence be used to uniquely represent every point in the 2D plane without any loss of information. On the other hand, angle embedding loses all amplitude

information due to the normalisation of all points. A method to transform an unknown manifold into an n -sphere using ISP is proposed in [38] - here, however, the property of their concern was the conformality of the projection since subsequent learning is performed upon the surface. In [39], a parallelised version of [2] is developed using the FF-QRAM procedure [40] for amplitude encoding and the ISP to ensure an injective embedding.

In the method of Spherical Clustering [41], the nearest neighbour algorithm is explored based on the cosine similarity measure (Eq. (27) and Lemma 2). The cosine similarity is used in cases of information retrieval, text mining, and data mining to find the similarity between document vectors. It is used in those cases because the cosine similarity has low complexity for sparse vectors since only the non-zero co-ordinates need to be considered. For our case as well, it is in our interest to study Eqs. (22) to (24) with the cosine dissimilarity. This approach becomes particularly relevant once we employ stereographic embedding to encode the data points into quantum states.

1.2. Contribution

In this work, we first develop generalised stereographic embedding for hybrid quantum-classical kNN clustering as a better encoding that allows the quantum algorithm (Section 3) to outperform the accuracy and convergence of classical k-means algorithm in the absence of noise; in contrast, angle embedding introduces fundamental limitations to the accuracy not due to quantum noise. To validate this statement, we simulate this algorithm classically, which translates into an equivalent classical quantum-analogous stereographic kNN clustering algorithm (Section 4). One must note that we do not demonstrate that running the stereographic quantum kNN algorithm is more practical than the classical k-means algorithm in the NISQ context. We show that stereographic quantum kNN clustering converges faster and is more accurate than other hybrid quantum-classical kNN algorithms with angle or amplitude embedding. In parallel, the benchmarking of the classical stereographic kNN algorithm lets us claim that for the problem of decoding 64-QAM optical-fibre signals the generalised ISP and spherical centroid can allow for better accuracy and convergence.

The extensive testing upon the *real-world, experimental QAM dataset* (Section 2.1) revealed some significant results regarding the dependence of accuracy, runtime, and convergence performance upon the radius of projection, number of points, noise in the optical-fibre, and stopping criterion - described in Section 5. Noteworthy, we observe the existence of a finite optimal radius for the ISP (not equal to 1). To the best of our knowledge, no other work has considered a generalised projection radius for quantum embedding or studied its effect. Through our experimentation, we have verified that there exists an ideal radius greater than 1 for which accuracy performance is maximised. The advantageous implementation of the algorithm upon experimental data shows that our procedure is quite competitive. The fact that the developed quantum algorithm has an entirely classical analogue (with comparable time complexity to the classical k means algorithm) is a distinct advantage in terms of in-field deployment, especially compared to [2,9,18,34–36,39]. The developed quantum algorithm also has another advantage in the context of Noisy Intermediate-Scale Quantum (NISQ) realisations - it has the least circuit depth and circuit width among all candidates [2,9,36,39] - making it easier to implement with the current quantum technologies. Another significant contribution is our generalisation of the dissimilarity for clustering; instead of Euclidean dissimilarity (distance), we consider other dissimilarities which might be better estimated by quantum circuits (Appendix E). A somewhat similar approach was developed in parallel by [42] in the context of amplitude embedding. All previous approaches [2,9,36,39] only try to estimate the Euclidean distance. We also make the contribution of studying the relative effect of ‘quantumness’ and the ISP, something completely overlooked in previous works. We show that the quantum ‘advantage’ in accuracy performance touted by works such as [34–36,39] is in reality quite suspect and achievable through classical means. In the appendix, we describe a generalisation of the stereographic embedding - the Ellipsoidal embedding, which we expect to give even better results in future works.

Other secondary contributions of our work include:

- The development of a mathematical formalism for the generalisation of kNN to indicate the contribution of various parameters such as dissimilarities and dataspace (Section 2.4);
- Presenting the procedure and circuit for *stereographic embedding* using the Bloch embedding procedure, which consumes only $O(1)$ in time and resources (Section 3.1).

2. Preliminaries

In this section, for completeness, we touch upon some concepts and background information required to understand the paper. These concepts range from small general statements on quantum states (Bloch sphere, fidelity and Bell-state measurement in Sections 2.2 and 2.3) to the mathematical formalism of kNN (Sections 2.4 to 2.6), and stereographic projection (Section 2.7). We begin by first describing the optic-fibre experimental setup used to collect the 64-QAM dataset, upon which the clustering algorithms were tested and benchmarked.

2.1. Optical-Fibre Setup

M-ary Quadrature Amplitude Modulation (M-QAM) is a simple and popular protocol for digital data transmission through analog communication channels. It is widely used in optical-fibre communication networks, and the decoding process of the received data often uses the k -nearest-neighbour algorithm to cluster nearby points. More details, including the description of the model used in the experiments, can be found in Appendix A. We now describe the experimental setup used to collect the dataset that is used for benchmarking the clustering algorithms.

The dataset contains a launch-power (laser-power feed into the fibre) sweep (four datasets collected at four different launch powers) of 80 km fibre transmission of coherent dual polarization (DP)-64QAM with a gross data rate of 960×10^9 bits/s. For the dataset, we assumed 15% overhead for forward error correction (FEC) and used 3.47% overhead for pilots and training sequences, so the net bit rate is 800×10^9 bits/s. Note that the pilots and training sequences are removed after the MIMO equalizer. An overview of the experimental setup [10,15] to capture this real-world database is shown in Fig. 1. Four 120×10^9 Samples/s digital-to-analog converters (DACs) generate an electrical signal amplified by four 60 GHz 3dB-Bandwidth amplifiers. A tunable 100 kHz external cavity laser (ECL) source generates a continuous wave signal that is modulated by a 32 GHz DP-I/Q modulator. The receiver comprises an optical 90° -hybrid and four 100 GHz balanced photodiodes. The electrical signals are digitized using four 10-bit analog-to-digital converters (ADCs) with 256×10^9 Samples/s and 110 GHz. Subsequently, the raw signals are pre-processed by the receiver digital signal processing (DSP) blocks.

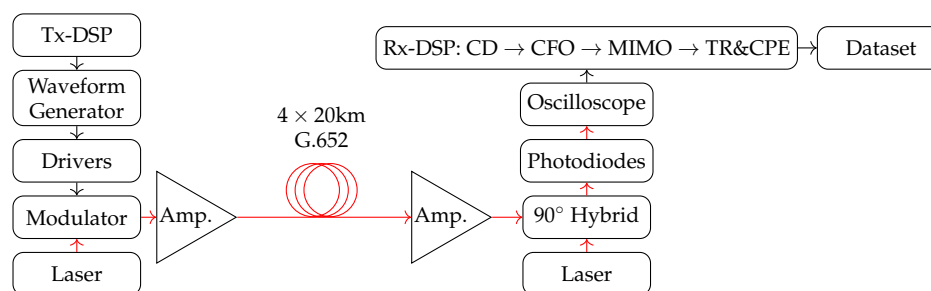


Figure 1. Experimental setup over a 80 km G.652 fiber link at optimal launch power of 6.6 dBm. Chromatic dispersion (CD) and carrier frequency offset (CFO) compensation, multiple-input multiple-output (MIMO) equalizer, timing recovery (TR) and carrier phase estimation (CPE) [10,15].

The datasets were collected in a very short time, corresponding to the memory size of the oscilloscope, which is limited. This is referred to as offline processing. At the receiver, the signals were normalised to fit the alphabet. The average launch power in watts can be calculated as follows:

$$P_{(W)} = 1W \cdot 10^{(P_{(dBm)})/10} / 1000 = 10^{(P_{(dBm)}-30)/10}$$

There are four sets of published data with different launch powers, corresponding to different levels of non-linear distortions during transmission: 2.7dBm, 6.6dBm, 8.6dBm, and 10.7dBm. Each dataset consists of the ‘alphabet’ (initial analog transmission values), the error-corrected received analog values, and the true labels of the transmitted points. The data has been explained and visualised in detail in the Appendix A. To quantify the system performance of an amplified coherent optical communication system, one uses either a launch power sweep or an OSNR (Optical Signal to Noise Ratio) sweep. While the OSNR metric is used when the system is operating in the linear region, the launch power is the preferred metric to show the performance degradation in the nonlinear region since the induced nonlinear effects are directly proportional to the launch power. The signal-to-noise (SNR) ratio for each launch power can be computed using the following expression, where \mathbf{z} are the received noisy signals and \mathbf{x} are the noiseless target symbols (the launched signals):

$$SNR = 10 \log_{10} \left(\frac{\text{mean}(\|\mathbf{z}\|^2)}{\text{mean}(\|\mathbf{z} - \mathbf{x}\|^2)} \right).$$

After obtaining this noisy real-world dataset, our task is to decode the received analog values into bit-strings. The kNN is the candidate of choice since it classifies datasets into clusters by associating an ‘average point’ (centroid) to each cluster. In our method, the objective of the clustering algorithm is first to identify, using the set of received signals, a given number M of centroids (one for each cluster) and then to assign each signal to the ‘nearest’ centroid. The second step is classification. This creates the clusters, which can then be decoded into bit signals through the process of demapping. Demapping consists of mapping the original transmission constellation (alphabet) to the current centroids and then assigning the bit-string label associated with that initial transmission point to all the points in the cluster of that centroid. This process completes the final step of the QAM protocol, translating the analog values to bit-strings read by the receiver. The size M of the constellation is known since we know beforehand which QAM protocol is being used. We also know the ‘alphabet’, i.e. the initial and ideal points at which the signals were transmitted.

2.2. Bloch Sphere

It is well known that all the qubit pure states can be obtained from the zero state using the unitary U [43]

$$|\psi(\theta, \phi)\rangle = U(\theta, \phi) |0\rangle = \cos(\theta/2) |0\rangle + e^{i\phi} \sin(\theta/2) |1\rangle \quad (1)$$

where

$$U(\theta, \phi) := \begin{pmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} & e^{i\phi} \cos \frac{\theta}{2} \end{pmatrix}. \quad (2)$$

These are unit vectors in the unit sphere of \mathbb{C}^2 , but it is also well known that the corresponding density matrices are uniquely represented by the Bloch vectors

$$\mathbf{a}(\theta, \phi) := (\sin \theta \cos \phi, \sin \theta \sin \phi, \cos \theta)$$

as points in the unit sphere $S^2(1) \subset \mathbb{R}^3$ [43] (the Bloch sphere) through the relation

$$\rho(\theta, \phi) = |\psi(\theta, \phi)\rangle\langle\psi(\theta, \phi)| \quad (3)$$

$$= \begin{pmatrix} \cos^2 \frac{\theta}{2} & -e^{-i\phi} \cos \frac{\theta}{2} \sin \frac{\theta}{2} \\ e^{i\phi} \cos \frac{\theta}{2} \sin \frac{\theta}{2} & \sin^2 \frac{\theta}{2} \end{pmatrix} \quad (4)$$

$$= \frac{1}{2} \begin{pmatrix} 1 + \cos \theta & -e^{-i\phi} \sin \theta \\ e^{i\phi} \sin \theta & 1 - \cos \theta \end{pmatrix} \quad (5)$$

$$= \frac{1}{2} (\mathbb{1} + \mathbf{a}(\theta, \phi) \cdot \vec{\sigma}) \quad (6)$$

where $\mathbb{1}$ is the identity matrix and $\vec{\sigma} = (\sigma_x, \sigma_y, \sigma_z)$ is the vector of Pauli matrices

$$\sigma_x = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad \sigma_y = i \begin{pmatrix} 0 & -1 \\ 1 & 0 \end{pmatrix} \quad \sigma_z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix}.$$

Regarding mixed states, notice that Eq. (6) is linear and thus, convex combinations of density matrices translate to convex combinations of Bloch vectors, meaning that the interior of the sphere represents the mixed states. Namely, the most general qubit quantum states can be represented by

$$\rho_{\mathbf{a}} \equiv \rho(\mathbf{a}) = \frac{1}{2} (\mathbb{1} + \mathbf{a} \cdot \vec{\sigma}), \quad \|\mathbf{a}\|_2 \leq 1. \quad (7)$$

Finally, since the Pauli matrices are orthogonal operators under the Hilbert-Schmidt inner product, this inner product is easily computed as

$$\text{Tr}(\rho_{\mathbf{a}_1} \rho_{\mathbf{a}_2}) = \frac{1}{2} (1 + \mathbf{a}_1 \cdot \mathbf{a}_2). \quad (8)$$

which for pure states coincides with the fidelity.

Using the Bloch sphere representation of qubit quantum states also makes it easy to find orthogonal states and compute diagonalizations. Indeed, let \mathbf{a} be a unit vector ($\|\mathbf{a}\| = \mathbf{a} \cdot \mathbf{a} = 1$), thus representing the pure state $\rho_{\mathbf{a}} = \frac{1}{2} (\mathbb{1} + \mathbf{a} \cdot \vec{\sigma})$, then the orthogonal state to $\rho_{\mathbf{a}}$ is simply the antipodal point

$$\rho_{-\mathbf{a}} = \frac{1}{2} (\mathbb{1} - \mathbf{a} \cdot \vec{\sigma}) \quad (9)$$

which can be shown by computing the inner product as in Eq. (8)

$$\text{Tr}(\rho_{+\mathbf{a}} \rho_{-\mathbf{a}}) = \frac{1}{4} (1 + \mathbf{a} \cdot (-\mathbf{a})) = 0. \quad (10)$$

Hence, the Bloch eigenvectors for any Bloch vector \mathbf{a} are $\pm \frac{\mathbf{a}}{\|\mathbf{a}\|}$, the two antipodal points where the line of \mathbf{a} intersect the Bloch sphere. Namely, for any mixed quantum state corresponding to the Bloch vector \mathbf{a} , we can decompose the quantum state as

$$\frac{1}{2} (\mathbb{1} + \mathbf{a} \cdot \vec{\sigma}) = p \frac{1}{2} \left(\mathbb{1} + \frac{\mathbf{a}}{\|\mathbf{a}\|} \cdot \vec{\sigma} \right) + (1-p) \frac{1}{2} \left(\mathbb{1} - \frac{\mathbf{a}}{\|\mathbf{a}\|} \cdot \vec{\sigma} \right) \quad (11)$$

$$= \frac{1}{2} \left(\mathbb{1} + (2p-1) \frac{\mathbf{a}}{\|\mathbf{a}\|} \cdot \vec{\sigma} \right) \quad (12)$$

with

$$2p-1 = \|\mathbf{a}\| \quad \implies \quad p = \frac{1}{2} (1 + \|\mathbf{a}\|). \quad (13)$$

In the next section, we discuss how we use the Bell-state measurement to estimate the fidelity between quantum states and exposit when this should be chosen over the SWAP test.

2.3. Bell-State Measurement and Fidelity

We use the Bell-state measurement to estimate the fidelity between two pure states. The Bell-state measurement is defined as the von-Neumann measurement of the maximally entangled basis

$$|\phi_{ij}\rangle := CNOT(H \otimes \mathbb{1}) |ij\rangle, \quad (14)$$

which by construction is equivalent to a standard basis measurement after $(H \otimes \mathbb{1})CNOT$ as displayed in Fig. 2. This measurement can be used to estimate the fidelity as follows.

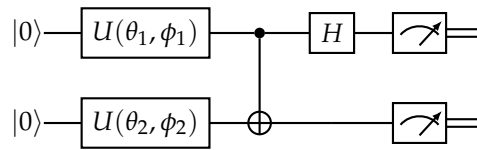


Figure 2. Quantum circuit of the Bell-state measurement. The measurement is obtained by first transforming the Bell basis into the standard basis with $(H \otimes \mathbb{1})CNOT$ and then measuring in the standard basis.

Lemma 1. Let $|\psi\rangle$ and $|\chi\rangle$ be two qubit pure states and let $|\phi_{11}\rangle := CNOT(H \otimes \mathbb{1}) |11\rangle$ (the singlet Bell state). Then

$$|\langle\phi_{11}|(|\psi\rangle \otimes |\chi\rangle)|^2 = \frac{1}{2}(1 - |\langle\psi|\chi\rangle|^2). \quad (15)$$

Proof. Let us write the states as

$$|\psi\rangle = \psi_0 |0\rangle + \psi_1 |1\rangle \quad |\chi\rangle = \chi_0 |0\rangle + \chi_1 |1\rangle, \quad (16)$$

Then the state before the standard-basis measurement is

$$|\psi_{out}\rangle = (H \otimes \mathbb{1}) CNOT(|\psi\rangle \otimes |\chi\rangle) = \frac{1}{\sqrt{2}} \begin{pmatrix} \psi_0\chi_0 + \psi_1\chi_1 \\ \psi_0\chi_1 + \psi_1\chi_0 \\ \psi_0\chi_0 - \psi_1\chi_1 \\ \psi_0\chi_1 - \psi_1\chi_0 \end{pmatrix} \quad (17)$$

and in particular, the probability of outcome $ij = 11$ (i.e. simultaneous measurement of both qubits yields value '1' on each qubit) can be written as

$$|\langle\phi_{11}|(|\psi\rangle \otimes |\chi\rangle)|^2 = |\langle 11|\psi_{out}\rangle|^2 = \frac{1}{2}|\psi_0\chi_1 - \psi_1\chi_0|^2. \quad (18)$$

The fidelity is obtained now by adding and subtracting $\psi_0^*\psi_0\chi_0^*\chi_0 + \psi_1^*\psi_1\chi_1^*\chi_1$ and computing

$$|\langle\phi_{11}|(|\psi\rangle \otimes |\chi\rangle)|^2 = \frac{1}{2}(1 - \psi_1\chi_0\psi_0^*\chi_1^* - \psi_0\chi_1\psi_1^*\chi_0^* - \psi_0^*\psi_0\chi_0^*\chi_0 - \psi_1^*\psi_1\chi_1^*\chi_1) \quad (19)$$

$$= \frac{1}{2}(1 - |\psi_0^*\chi_0 + \psi_1^*\chi_1|^2) \quad (20)$$

$$= \frac{1}{2}(1 - |\langle\psi|\chi\rangle|^2), \quad (21)$$

concluding the proof. \square

Lemma 1 is used to construct the quantum clustering algorithm in Section 3. We will use the quantum circuit of Fig. 2 for fidelity estimation in the developed quantum algorithm.

Remark 1. *Since we are only interested in the $ij = 11$ outcome and we are measuring qubits, the coarse-grained projective measurement defined by $|\phi_{11}\rangle\langle\phi_{11}|$ and $\mathbb{1} - |\phi_{11}\rangle\langle\phi_{11}|$ is sufficient for computing the inner product. The non-destructive version of this measurement is known as the SWAP test [25,26], first described in [24]. This test has been used extensively for overlap estimation in quantum algorithms [2]. The SWAP test requires one to only measure an ancilla qubit instead of the two input qubits, leaving them in the post-measurement state, which can be used later for other purposes. However, given the current limitations of NISQ technologies, storing quantum information for reuse is quite impractical; therefore, we prefer the destructive measurement version for overlap estimation. Namely, we use the Bell-state measurement instead of the SWAP test because the post-measurement state is unnecessary.*

2.4. Nearest-Neighbour Clustering Algorithms

Clustering is a simple, powerful and well-known machine-learning algorithm that has been extensively used throughout the literature. In this section, we summarise some standard and basic notions introduced by clustering and define this class of heuristic algorithms precisely so that we can make clear the difference between regular clustering and the quantum and quantum-inspired clustering algorithms introduced in this paper. We first define the involved variables needed for the kNN.

Definition 1 (Clustering State). *We define a k -Nearest-Neighbour Clustering State, or clustering state for short, as a collection $(D, \bar{\mathbf{c}}, \mathbb{D}, d)$ where*

- \mathbb{D} is a space called *dataspace* with elements called *points*.
- $D \subseteq \mathbb{D}$ is a subset called *dataset* consisting of points called *datapoints*.
- $\bar{\mathbf{c}} = (\mathbf{c}_1 \ \mathbf{c}_2 \ \dots \ \mathbf{c}_k) \subseteq \mathbb{D}^k$ is a list (of size k) of points called *centroids*
- $d : \mathbb{D} \times \mathbb{D} \mapsto \mathbb{R}$ is a lower bounded function called *dissimilarity function*, or *dissimilarity* for short.

Note that d does not have to be a distance metric. We now define the basic steps that are repeated in the clustering algorithm.

Definition 2 (Clusters and Centroid update). *Let $(D, \bar{\mathbf{c}}, \mathbb{D}, d)$ be a clustering state. We define the clusters of the state as, for each $j = 1, \dots, k$, the set*

$$C_j(\bar{\mathbf{c}}) = \left\{ \mathbf{p} \in D \mid d(\mathbf{p}, \mathbf{c}_j) \leq d(\mathbf{p}, \mathbf{c}_\ell) \quad \forall \ell = 1, \dots, k, \mathbf{p} \notin \bigcup_{\ell < j} C_\ell(\bar{\mathbf{c}}) \right\}. \quad (22)$$

We now define the possible new centroids of a subset $C \subseteq D$ as the set

$$\mathcal{P}(C) := \operatorname{argmin}_{\mathbf{x} \in \mathbb{D}} \sum_{\mathbf{p} \in C} d(\mathbf{x}, \mathbf{p}) \quad (23)$$

of all points minimising the total (and thus the average) dissimilarity. Then, we call a centroid update any function $\mathbf{c}^{\text{update}} : \mathcal{P}(\mathbb{D}) \rightarrow \mathbb{D}$ (where \mathcal{P} denotes the power set) of clusters such that $\mathbf{c}^{\text{update}}(C)$ is a possible new centroid, namely such that $\mathbf{c}^{\text{update}}(C) \in \mathcal{P}(C)$, for all $j = 1, \dots, k$. We then define the following short-hand notation for the centroid update of $\bar{\mathbf{c}}$, namely the new list of centroids

$$\bar{\mathbf{c}}^{\text{update}}(\bar{\mathbf{c}}) = \left(\mathbf{c}^{\text{update}}(C_1(\bar{\mathbf{c}})), \dots, \mathbf{c}^{\text{update}}(C_k(\bar{\mathbf{c}})) \right). \quad (24)$$

We now define the general k -nearest-neighbour clustering algorithm.

Definition 3 (*K-Nearest-Neighbour Clustering Algorithm (kNN)*). Finally, we define a K -Nearest-Neighbour clustering algorithm (kNN) as a pair of clustering state and centroid update $(D, \bar{\mathbf{c}}_1, \mathbb{D}, d, \bar{\mathbf{c}}^{\text{update}})$. The kNN algorithm defines a sequence of clustering states $(D, \bar{\mathbf{c}}_i, \mathbb{D}, d)$ via $\bar{\mathbf{c}}_{i+1} = \bar{\mathbf{c}}^{\text{update}}(\bar{\mathbf{c}}_i)$ for all $i \in \mathbb{N}$ which we call the iterations (of the algorithm).

A point of note is that Eq. (23) implies that the new centroid is one of the points \mathbf{x} in the dataspace that minimises the total (and hence the average) dissimilarity with all the points \mathbf{p} in the cluster. Also, notice that this definition requires one to initialise or populate the list $\bar{\mathbf{c}}$ with initial values, i.e. the *initial centroids* $\bar{\mathbf{c}}_1$ must be defined as a starting point for the clustering algorithm. The initial centroids can be assigned randomly or defined as a function of parameters such as the dataset.

Another comment about Eq. (23): in our case, we will see later in Section 3.2 that all choices of points from the set \mathcal{P}_j will be equivalent. As in our algorithm, this freedom of choice can be exploited to reduce the amount of computation or for other optimisations.

Notice that Eqs. (23) and (24) implies that centroids are generally not part of the original data set; however, according to Eqs. (23) and (24) they must be restricted to the space in which the dataset is defined. Definitions involving centroids for which $\bar{\mathbf{c}} \notin \mathbb{D}^k$ are possible but are not used in this work.

One can see that any kNN can be broken down into two steps that keep alternating until a stopping condition (a condition which, when true, forces the algorithm to terminate) is met: a *cluster update* which updates the points associated with the newly calculated centroid, and then a *centroid update* which recalculates the centroid based upon the new points associated to it through its cluster. For the cluster update, the value of the centroid calculated in the previous iteration is taken, and its cluster set is constructed by collecting all the points in the dataset that are ‘closer’ to it than any other centroid. The ‘closeness’ is computed by using a pre-defined dissimilarity. In the next step, the centroids are updated by searching in the dataspace, for each updated cluster, a new point for which the sum of dissimilarities between that point and all points in the cluster is minimised.

This procedure will lead to different results if one changes the dissimilarity or the space of data points or both. In this paper, we explore the effects of changing this dissimilarity as well as the space of data points, and we shall explain it in the context of quantum states.

2.5. Euclidean Dissimilarity and Classical Clustering

It can be seen from the centroid update in Eqs. (22) and (23) that the dissimilarity plays a central role in the clustering algorithm. The nature of this function directly controls the first step of cluster update since the dissimilarity is used to compute the ‘closeness’ between any two points in the dataspace. It is also apparent that if the dissimilarity is changed in the centroid update, the points at which the minimum is achieved could also change.

The Euclidean dissimilarity $d_e : \mathbb{R}^n \times \mathbb{R}^n \rightarrow \mathbb{R}$ is defined simply as the square of the Euclidean distance between the points:

$$d_e(\mathbf{a}, \mathbf{b}) = \|\mathbf{a} - \mathbf{b}\|^2. \quad (25)$$

For a finite subset $C \subset \mathbb{R}^n$, the minimisation of Eqs. (23) and (24) yields a unique point, reducing to the average point of the cluster:

$$\mathbf{c}_e^{\text{update}}(C) := \operatorname{argmin}_{\mathbf{x} \in \mathbb{R}^n} \sum_{\mathbf{p} \in C} d_e(\mathbf{x}, \mathbf{p}) = \frac{1}{|C|} \sum_{\mathbf{p} \in C} \mathbf{p}, \quad (26)$$

which we call the *Euclidean centroid update*. This is the most typical case of centroid update, where the new centroid is updated as the mean point of all points in the cluster. This corresponds to the classic k -means clustering algorithm [44], which now can be defined as follows.

Definition 4 (*n*-Dimensional Euclidean Classical kNN (*n*DEC-kNN)). An *n*-dimensional classical Euclidean kNN algorithm is any clustering algorithm with dataspace \mathbb{R}^n , Euclidean dissimilarity and cluster update as the average point, as in Eq. (26). Namely, any clustering algorithm of the form $(D, \bar{\mathbf{c}}, \mathbb{R}^n, d_e, \mathbf{c}_e^{\text{update}})$.

The computation of the centroid through Eq. (26) instead of Eqs. (23) and (24) reduces the complexity of the centroid update step; such a reduced expression is used to compute the updated centroids rather than searching the entire dataspace for the minimising points during the centroid update.

2.6. Cosine Dissimilarity

In this work, we project the collected two-dimensional dataset (described in Section 2.1) into a sphere via the ISP. After this projection, the calculation of the centroids according to Eq. (26) would generally yield centroids which lie inside the sphere instead of on the $S^2(r)$ surface due to the convex nature of the sphere's surface.

In our work, to use qubit pure states, we restrict the dataspace \mathbb{D} to the sphere surface $S^2(r)$, forcing the centroids to lie on the surface of a sphere. This naturally leads to the question of what the proper reformulation of Eqs. (23) and (24) is, and whether a computationally inexpensive formula similar to Eq. (26) exists for this case as well. This question will be answered in Lemma 3. For this purpose, it is useful to first define the cosine dissimilarity [45] and see how it relates to the Euclidean dissimilarity.

Definition 5 (Cosine Dissimilarity). For two points, \mathbf{a} and \mathbf{b} in an inner-product space \mathbb{D} the cosine dissimilarity is defined as:

$$d_s(\mathbf{a}, \mathbf{b}) = 1 - \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}, \quad (27)$$

where $\mathbf{a} \cdot \mathbf{b}$ is the inner product between the two points expressed as vectors from the origin, $\|\mathbf{a}\|$ is the norm of \mathbf{a} induced by the inner product.

This is called *cosine* dissimilarity because when $\mathbf{a}, \mathbf{b} \in \mathbb{R}^n$ the cosine dissimilarity $d_s(\mathbf{a}, \mathbf{b})$ reduces to $1 - \cos(\alpha)$, where α is the angle between \mathbf{a} and \mathbf{b} . The cosine dissimilarity is also known sometimes as *cosine distance* (although it is not a distance), while $\frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|}$ is well known as *cosine similarity*. This quantity, by construction, only depends on the direction of the vectors and not their magnitude. Said otherwise, we have

$$d_s(\mathbf{a}, \mathbf{b}) = d_s(c\mathbf{a}, \mathbf{b}) = d_s(\mathbf{a}, c\mathbf{b}) \quad (28)$$

for any positive constant $c > 0$. We also note that cosine dissimilarity of Eq. (27) can be related to the Euclidean dissimilarity of Eq. (25) if \mathbf{a} and \mathbf{b} lie on the *n*-sphere $S^n(r) := \{ \mathbf{s} \in \mathbb{R}^{n+1} \mid \|\mathbf{s}\|_2 = r \}$ of radius r , as stated by the following lemma.

Lemma 2. Let d_s and d_e be the cosine and Euclidean dissimilarities, respectively. Let $\mathbf{s}_1, \mathbf{s}_2 \in S^n(r)$ be points on the *n*-sphere of radius r , then

$$d_e(\mathbf{s}_1, \mathbf{s}_2) = 2r^2 d_s(\mathbf{s}_1, \mathbf{s}_2). \quad (29)$$

Proof. Assuming $\mathbf{s}_1, \mathbf{s}_2 \in S^n(r)$, Eq. (27) reduces to:

$$d_s(\mathbf{s}_1, \mathbf{s}_2) = 1 - \frac{1}{r^2} \mathbf{s}_1 \cdot \mathbf{s}_2, \quad (30)$$

then

$$2r^2 d_s(\mathbf{s}_1, \mathbf{s}_2) = 2r^2 - 2\mathbf{s}_1 \cdot \mathbf{s}_2 = \|\mathbf{s}_1\|^2 + \|\mathbf{s}_2\|^2 - 2\mathbf{s}_1 \cdot \mathbf{s}_2 = \|\mathbf{s}_1 - \mathbf{s}_2\|^2 = d_e(\mathbf{s}_1, \mathbf{s}_2), \quad (31)$$

concluding the proof. \square

From this, we can expect that the minimiser of the centroid update equation (Eq. (23)) computed using the cosine dissimilarity will closely relate to the Euclidean centroid update. However, the derivation is not straightforward since the Euclidean centroid update does not lie on the same sphere, but lies inside at a smaller radial distance. This is shown in the following lemma.

Lemma 3. *Let $C \subset S^n(r)$ be a finite set, then*

$$\mathbf{c}_s^{\text{update}}(C) := \operatorname{argmin}_{\mathbf{x} \in S^n(r)} \sum_{\mathbf{p} \in C} d_s(\mathbf{x}, \mathbf{p}) = r \frac{\sum_{\mathbf{p} \in C} \mathbf{p}}{\left\| \sum_{\mathbf{p} \in C} \mathbf{p} \right\|}. \quad (32)$$

We call this the cosine or spherical centroid update. In particular, thus

$$\mathbf{c}_s^{\text{update}}(C) = r \frac{\mathbf{c}_e^{\text{update}}(C)}{\left\| \mathbf{c}_e^{\text{update}}(C) \right\|} \quad (33)$$

where $\mathbf{c}_e^{\text{update}}(C) = \frac{1}{|C|} \sum_{\mathbf{p} \in C} \mathbf{p}$ is the Euclidean centroid update of Eq. (26).

Proof. The second claim is trivial, we thus have to prove only the first claim. Given that $C \subseteq S^n(r)$, then according to Lemma 2, the cosine dissimilarity given in Eq. (27) reduces for all $\mathbf{a}, \mathbf{b} \in C$ to:

$$d_s(\mathbf{a}, \mathbf{b}) = 1 - \frac{\mathbf{a} \cdot \mathbf{b}}{\|\mathbf{a}\| \|\mathbf{b}\|} = 1 - \frac{1}{r^2} \mathbf{a} \cdot \mathbf{b}. \quad (34)$$

The minimisation in Eq. (23) can then be calculated for the cosine dissimilarity with a Lagrangian (see Eq. (37)) that satisfies Eq. (23) and Eq. (24) at the minimising point. Namely, we have to find $\mathbf{x} \in \mathbb{R}^n$ that minimises

$$f(\mathbf{x}) = \sum_{\mathbf{p} \in C} d(\mathbf{x}, \mathbf{p}), \quad (35)$$

subject to the restriction condition that assures that $\mathbf{x} \in S^n(r)$, that is

$$g(\mathbf{x}) = \|\mathbf{x}\|^2 - r^2 = 0. \quad (36)$$

Such a Lagrangian is expressed as

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) - \lambda g(\mathbf{x}) = \sum_{\mathbf{p} \in C} d_s(\mathbf{x}, \mathbf{p}) - \lambda (\|\mathbf{x}\|^2 - r^2), \quad (37)$$

where λ is the Lagrangian multiplier. We then calculate the centroid update by employing the derivative criteria to Eq. (37).

$$0 = \nabla \left(\sum_{\mathbf{p} \in C} \left(1 - \frac{1}{r^2} \mathbf{x} \cdot \mathbf{p} \right) - \lambda (\|\mathbf{x}\|^2 + r^2) \right) = -\frac{1}{r^2} \sum_{\mathbf{p} \in C} \mathbf{p} - 2\lambda \mathbf{x} \quad (38)$$

Therefore the following holds:

$$\mathbf{x} = -\frac{1}{2\lambda r^2} \sum_{\mathbf{p} \in C} \mathbf{p}. \quad (39)$$

Substituting Eq. (39) into the restriction in Eq. (36), we obtain the multiplier λ as:

$$|\lambda| = \frac{1}{2r^3} \cdot \left\| \sum_{\mathbf{p} \in C} \mathbf{p} \right\|. \quad (40)$$

Therefore, the critical point and minimising point \mathbf{c}_s is written as

$$\mathbf{c}_s = \frac{r}{\left\| \sum_{\mathbf{p} \in C} \mathbf{p} \right\|} \sum_{\mathbf{p} \in C} \mathbf{p}, \quad (41)$$

as claimed. \square

We can observe that Lemma 3 implies that the minimiser obtained by restricting the point to lie on the surface of the sphere is the projection (from the origin) of the minimiser of the Euclidean dissimilarity into the sphere's surface.

Corollary 1. *Let $C \subset S^n(r)$ be a finite set, the possible new centroids of C under cosine dissimilarity in \mathbb{R}^3 are*

$$\mathcal{P}_s(C) := \operatorname{argmin}_{\mathbf{x} \in \mathbb{R}^n} \sum_{\mathbf{p} \in C} d_s(\mathbf{x}, \mathbf{p}) = \left\{ r \sum_{\mathbf{p} \in C} \mathbf{p} : r > 0 \right\}. \quad (42)$$

We call these the cosine possible new centroids.

Proof. We have

$$\mathbf{c}_s^{\text{update}} := \operatorname{argmin}_{\mathbf{x} \in \mathbb{R}^n \setminus \{0\}} \sum_{\mathbf{p} \in C} d_s(\mathbf{x}, \mathbf{p}) \quad (43)$$

$$= \operatorname{argmin}_{\mathbf{x} \in S^n(r), r > 0} \sum_{\mathbf{p} \in C} d_s(\mathbf{x}, \mathbf{p}) \quad (44)$$

$$= \left\{ r \frac{\sum_{\mathbf{p} \in C} \mathbf{p}}{\left\| \sum_{\mathbf{p} \in C} \mathbf{p} \right\|} \right\}_{r > 0} \quad (45)$$

$$= \left\{ r \sum_{\mathbf{p} \in C} \mathbf{p} : r > 0 \right\}, \quad (46)$$

where the last equality follows from Eq. (28), namely

$$d_s \left(r \frac{\sum_{\mathbf{p} \in C} \mathbf{p}}{\left\| \sum_{\mathbf{p} \in C} \mathbf{p} \right\|}, \mathbf{p} \right) = d_s \left(\sum_{\mathbf{p} \in C} \mathbf{p}, \mathbf{p} \right) \quad (47)$$

for all r and \mathbf{p} ; thus making all the points $r \sum_{\mathbf{p} \in C} \mathbf{p}$, $r > 0$ equivalent possibilities for the centroid update. \square

2.7. Stereographic Projection

The inverse stereographic projection (ISP), shown in Fig. 3, is a bijective mapping

$$s_r^{-1} : \mathbb{R}^n \mapsto S^n(r) \setminus \{N\} \quad (48)$$

from the Euclidean space \mathbb{R}^n into an n -sphere $S^n(r) \subset \mathbb{R}^{n+1}$ without the north pole N .

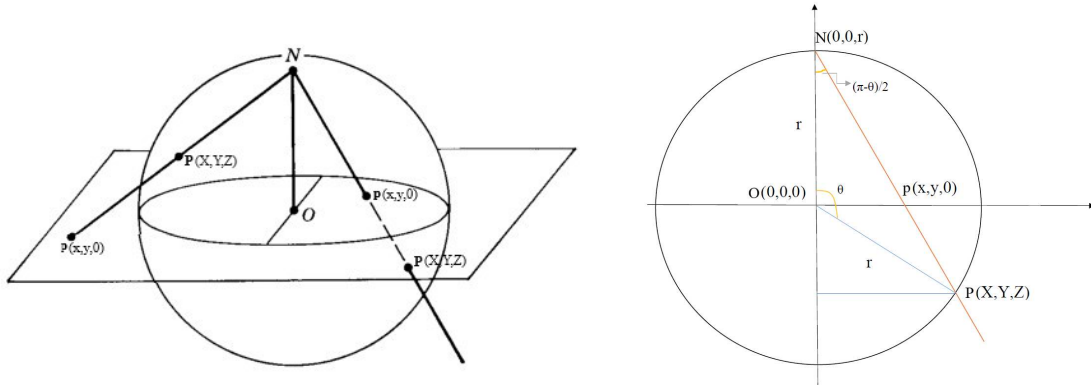


Figure 3. Inverse Stereographic Projection (ISP) from [46] (left) and ISP with different radii (right). The figure on the right is the cut of the figure in the left going through the N , O and P points.

This mapping is interesting because of the natural equivalence between the 3D unit sphere $S^2(1)$ and the Bloch sphere of qubit quantum states. In this case, as displayed in Fig. 3, the ISP maps a 2-dimensional point $\mathbf{p} = (p_x, p_y) \in \mathbb{R}^2$ into a three-dimensional point $s_r^{-1}(\mathbf{p}) = (s_x(\mathbf{p}), s_y(\mathbf{p}), s_z(\mathbf{p})) \in S^2(r) \setminus \{(0,0,r)\}$ through the following set of transformations:

$$s_x(\mathbf{p}) = p_x \cdot \frac{2r^2}{p_x^2 + p_y^2 + r^2} \quad s_y(\mathbf{p}) = p_y \cdot \frac{2r^2}{p_x^2 + p_y^2 + r^2} \quad s_z(\mathbf{p}) = r \cdot \frac{p_x^2 + p_y^2 - r^2}{p_x^2 + p_y^2 + r^2} \quad (49)$$

$$= p_x \cdot \frac{2r^2}{\|\mathbf{p}\|^2 + r^2} \quad = p_y \cdot \frac{2r^2}{\|\mathbf{p}\|^2 + r^2} \quad = r \cdot \frac{\|\mathbf{p}\|^2 - r^2}{\|\mathbf{p}\|^2 + r^2}. \quad (50)$$

The polar and azimuthal angles of the projected point are given by the expressions:

$$\phi(\mathbf{p}) = \tan^{-1}\left(\frac{p_y}{p_x}\right) \quad \theta(\mathbf{p}) = 2 \cdot \tan^{-1}\left(\frac{r}{\|\mathbf{p}\|}\right) \quad (51)$$

This information, particularly Eq. (51), will allow us to associate each point in \mathbb{R}^2 to a unique quantum state through the Bloch sphere. Still, the inverse stereographic projection does not need to be bound to the preparation of quantum states and can be used as a transformation between classical kNN algorithms. Indeed, we can stereographically project and then perform classical clustering on the 3D data, namely perform 3DEC-kNN as defined in Definition 4.

Definition 6 (3D Stereographic Classical kNN (3DSC-kNN)). Let s_r^{-1} be an ISP, and let $(D, \bar{\mathbf{c}}, \mathbb{R}^2, d_e)$ be a clustering state (recall, D is the dataset and $\bar{\mathbf{c}}$ are the initial centroids). We then define the 3D Stereographic Classical kNN (3DSC-kNN) as $(s_r^{-1}(D), s_r^{-1}(\bar{\mathbf{c}}), \mathbb{R}^3, d_e, \mathbf{c}_e^{\text{update}})$.

Here we apply s_r^{-1} elementwise and thus $s_r^{-1}(\bar{\mathbf{c}}) = (s_r^{-1}(\mathbf{c}_1), \dots, s_r^{-1}(\mathbf{c}_k))$ for any list of centroids $\bar{\mathbf{c}}$ and $s_r^{-1}(D) = \{s_r^{-1}(\mathbf{p}) : \mathbf{p} \in C\}$ for any set of points C , and where C_j are the clusters as defined in Eq. (22).

Remark 2. Derivations and further observations can be found in Appendix C. Of particular note, as explained in more detail in Appendix C.2, is that changing the plane's distance from the centre of the sphere is equivalent to a change of radius. Therefore we can limit our analysis to projections where the centre of the plane is also the centre of the sphere without loss of generality.

3. Stereographic Quantum Nearest-Neighbour Clustering (SQ-kNN)

This section proposes and describes the quantum k (nearest-neighbour) clustering using stereographic embedding. In Section 4, we demonstrate an equivalent quantum-inspired (classical) version. Section 3.1 defines the method to convert the classical data into quantum states. In what follows, we describe how these states are manipulated so that we obtain an output that can be used to perform clustering, using the circuit of Section 2.3 for dissimilarity estimation. Section 3.2 defines the quantum algorithm in terms of Definitions 1 to 3, and Section 3.3 discusses the complexity and scalability of the algorithm. Section 3.4 discusses the SQ-kNN algorithm in the context of mixed states.

3.1. Stereographic Embedding, Bloch Embedding and Quantum Dissimilarity

For quantum algorithms to work on classical data, the classical data must be converted into quantum states. This process of encoding classical data into quantum states is also called *embedding*. The embedding of classical data into quantum states is not unique, and each technique's pros and cons must be weighed in the context of a specific application. The process of data embedding is an active field of research. More details on existing embedding can be found in Appendix B.

Here, we propose the *stereographic embedding* as an improved embedding of classical vector $\mathbf{p} \in \mathbb{R}^2$ into quantum state using its stereographic projection. We can split stereographic embedding into two steps: inverse stereographic projection and Bloch embedding. We define Bloch embedding, a variation of angle embedding, as follows.

Definition 7 (Bloch embedding). *Let $\mathbf{P} \in \mathbb{R}^3$. We define the Bloch embedded quantum state, or Bloch embedding for short, of \mathbf{P} as the quantum state*

$$\psi_{\mathbf{P}} := \frac{1}{2} \left(\mathbb{1} + \frac{\mathbf{P}}{\|\mathbf{P}\|} \cdot \vec{\sigma} \right) \quad (52)$$

which is simply the pure state obtained using $\mathbf{P}/\|\mathbf{P}\|$ as Bloch vector.

At this point, we define this general embedding for general 3-dimensional points since this general form will yield the quantum dissimilarity defined next. We will also define this embedding in the context of the ISP in Definition 9 below.

To obtain $\psi_{\mathbf{P}}$ the state can be encoded as explained in the preliminaries in Section 2.2, through Eqs. (2) and (6). For Bloch embedding, the θ and ϕ of Eqs. (2) and (6) would be the polar and azimuthal angles of \mathbf{P} respectively. We now define the quantum dissimilarity as follows.

Definition 8 (Quantum Dissimilarity). *For any two points $\mathbf{P}_1, \mathbf{P}_2 \in \mathbb{R}^3$, we define the quantum dissimilarity as*

$$d_q(\mathbf{P}_1, \mathbf{P}_2) := \frac{1}{2} (1 - \text{Tr}(\psi_{\mathbf{P}_1} \psi_{\mathbf{P}_2})), \quad (53)$$

where $\psi_{\mathbf{P}}$ is the Bloch embedding of \mathbf{P} .

Notice that, as per this definition, the classical 2-dimensional points are embedded in pure states only. In Section 3.4, we consider Bloch embedding the centroids also into mixed states, showing that it does not give an advantage in our framework. This quantum dissimilarity can be obtained either with the SWAP test or with the Bell state measurement on $\psi_{\mathbf{P}_1} \otimes \psi_{\mathbf{P}_2}$ as described in Section 2.3. In our application, we use the Bell state measurement (depicted in Fig. 2), as we do not need the extra resources of the SWAP test that allow us to keep the post-measurement state. For more details, see Lemma 1.

By Eq. (8), the quantum dissimilarity is proportional to the cosine dissimilarity.¹

$$d_q(\mathbf{P}_1, \mathbf{P}_2) = \frac{1}{2}(1 - \text{Tr}(\psi_{\mathbf{P}_1}\psi_{\mathbf{P}_2})) = \frac{1}{4} \left(1 - \frac{\mathbf{P}_1}{\|\mathbf{P}_1\|} \cdot \frac{\mathbf{P}_2}{\|\mathbf{P}_2\|} \right) = \frac{1}{4}d_s(\mathbf{P}_1, \mathbf{P}_2) \quad (54)$$

It is also proportional to the Euclidean distance for points on the same sphere (points with the same magnitude) as per Lemma 2. Namely, if $\mathbf{s}_1, \mathbf{s}_2 \in S^2(r)$ then:

$$d_q(\mathbf{s}_1, \mathbf{s}_2) = \frac{1}{8r^2}d_e(\mathbf{s}_1, \mathbf{s}_2). \quad (55)$$

We can finally define stereographic embedding as follows.

Definition 9 (Stereographic Embedding). *The stereographic embedding of a classical vector $\mathbf{p} \in \mathbb{R}^2$ consists of*

1. *projecting the 2D point \mathbf{p} into a point on the sphere of radius r in 3D space through the ISP:*

$$\mathbf{s} := s_r^{-1}(\mathbf{p}) \in S^2(r) \subset \mathbb{R}^3; \quad (56)$$

2. *Bloch embedding \mathbf{s} into $\psi_{\mathbf{s}} = \psi_{s_r^{-1}(\mathbf{p})}$.*

Comparing the distance estimate of the Stereographic embedding procedure (Eqs. (55) and (A34)) with that for the hybrid quantum-classical k-means with angle embedding (the ‘distance loss function’ described in [10]), we can see that the theoretical performance has been improved, since the estimate has been much improved with respect to the closeness to Euclidean distance. This leads us to expect a performance improvement of the SQ-kNN algorithm over the hybrid quantum-classical implementation of quantum k-means with angle embedding.

A very time-consuming computational step of kNN involves the repeated calculations of distances between the data set points meant to be classified and each centroid. In the case of the quantum kNN in [2], since angle embedding is not injective, many steps must be spent after estimating the fidelity to calculate the distance between the points using the norms. Even in [34,35,39], the norms of the points have to be stored classically, leading to much computational expense. Our method has the clear benefit of calculating the cosine dissimilarity directly through fidelity estimation. No further calculations are required due to all stereographically projected points having the same norm r in the sphere, and the existence of a bijection between the ISP and the original 2D datapoints, thus saving computational time and resources. In summary, Eqs. (54) and (55) portray a method to measure a dissimilarity that leads to consistent clustering involving pure states.

As one can see, in the case of stereographically projected points, d_q is directly proportional to the Euclidean dissimilarity between them. Since all the points after projection into the sphere have equal modulus r , and each projected point corresponds to a unique 2D data point, we can *directly compare the probability of getting outcome $ij = 11$ on the Bell-state measurement circuit for cluster assignment*. This eliminates extra steps needed during computation to account for the different moduli of points on the 2-dimensional plane.

3.2. The SQ-kNN Algorithm

We now have all the building blocks to define the quantum clustering algorithm. The quantum part will be the dissimilarity estimation d_q , obtained by embedding the data into quantum states as described in Section 3.1 and then feeding it into the quantum circuit described in Section 2.3 and Fig. 2

¹ This might not be true for other definitions of quantum dissimilarity, as in Section 3.4 where we redefine it to include embedding into mixed states.

to estimate an outcome probability. The finer details of distance estimation are further described in Appendix E. We can now formally define the developed algorithm building on the definition of clustering state (Definition 1), of clustering algorithm and cluster update provided by Definitions 2 and 3, of ISP as defined in Section 2.7 and of quantum dissimilarity d_q from Definition 8.

Definition 10 (Stereographic Quantum kNN (SQ-kNN)). Let s_r^{-1} be the ISP, let d_q be the quantum dissimilarity and let $(D, \bar{\mathbf{c}}, \mathbb{R}^2, d_e)$ be a clustering state (where D and $\bar{\mathbf{c}}$ are 2-dimensional dataset and initial centroids). We then define the Stereographic Quantum kNN (SQ-kNN) as the kNN clustering algorithm

$$\left(s_r^{-1}(D), s_r^{-1}(\bar{\mathbf{c}}), \mathbb{R}^3, d_q, \bar{\mathbf{c}}_q^{\text{update}} \right) \quad (57)$$

where $\bar{\mathbf{c}}_q^{\text{update}} := \sum_{\mathbf{p} \in C} \mathbf{p}$.

The complete process of performing SQ-kNN in practice can be described in detail as follows.

1. First, prepare to embed the classical data and initial centroids into quantum states using the ISP: project the 2-dimensional datapoints and initial centroids (in our case, the alphabet) into a sphere of radius r and calculate the polar and azimuthal angles of the points. This first step is executed entirely on a *classical* computer.
2. Cluster Update: The calculated angles are used to create the states using Bloch embedding (Definition 7). The dissimilarity between the centroid and point is then estimated using the Bell-state measurement. Once the dissimilarities between a point and all the centroids have been obtained, the point is assigned to the cluster of the 'closest' centroid. This is repeated for all the points that have to be classified. The quantum circuit and classical controller handle this step entirely. The controller feeds in the classical values at the appropriate times, stores the results of the various shots and classifies the point to the appropriate cluster.
3. Centroid Update: Since any non-zero point on the subspace of \mathbf{c}_s (see Corollary 1, Fig. 4) is an equivalent choice, to minimise computational expense, the centroids are updated as the sum point of all points in the cluster - as opposed to the average, for example, which minimises the Euclidean dissimilarity (Eq. (26)).

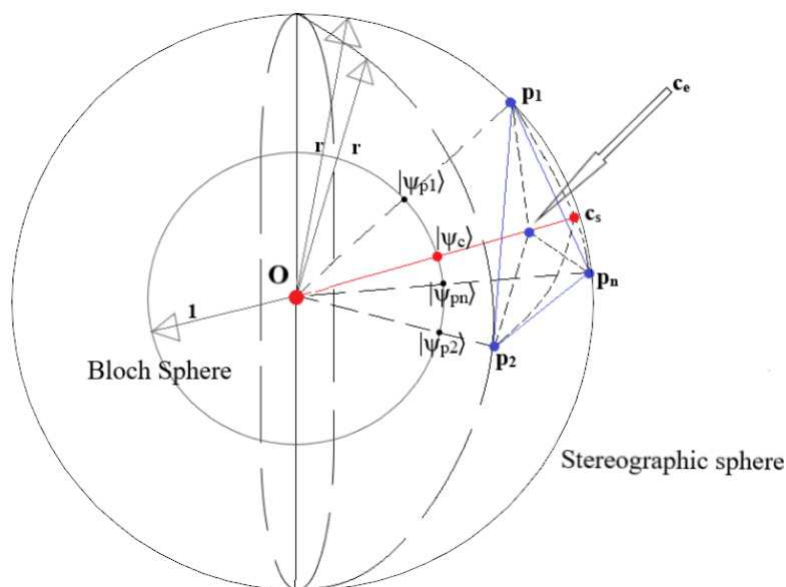


Figure 4. A diagram providing a visual intuition for how the stereographic quantum kNN (SQ-kNN) is equivalent to the 2DSC-kNN.

Once the centroids are updated, Step 2 (Cluster Update) is repeated, followed once again by Step 3 (Centroid Update) until a decided stopping condition is fulfilled.

Compared to 2D quantum kNN clustering with angle or amplitude embedding, the differences with the SQ-kNN algorithm lie in the embedding and the post-processing after the inner-product estimation.

- The stereographic embedding of the 2D datapoints is done by inverse stereographic projecting the point into a sphere of a chosen radius and then producing the quantum state obtained by rescaling the sphere to radius one.

In contrast, in angle embedding, the coefficients of the vectors are used as the angles of the Bloch vector (also known as dense angle embedding [47]), while in amplitude embedding, they are used as the amplitudes in the standard basis. For 2D vectors, amplitude embedding allows one to encode only one coefficient (instead of two) in one qubit, and sometimes angle embedding would also encode only one coefficient by using a single rotation (standard angle embedding [48]). Both angle and amplitude embeddings require the lengths of the vectors to be stored classically beside the quantum state, which is not needed in Bloch embedding.

- No post-processing is needed after the overlap estimation of stereographically embedded data as the obtained estimate is already a linear function of the inner product, as opposed to standard approaches using angle or amplitude encoding. Amplitude embedding also requires non-trivial computational time in the state preparation process. In contrast, in angle embedding, though the state preparation time is constant, recovering a useful dissimilarity (e.g. Euclidean) may involve many post-processing steps.

In short, stereographic embedding has the advantage of angle over amplitude embedding of being able to encode all values of a vector and low state preparation time, and the advantage of amplitude versus angle embedding in the recovery of the dissimilarity.

3.3. Complexity Analysis and Scaling

Let the ISP of a d -dimensional point $\mathbf{p} = [x_1 \ x_2 \ \dots \ x_d]$ into $S^d(r)$, using the point $(r, 0, 0, \dots, 0)$ (the 'North pole') as the projection point, be the point $\mathbf{s} = [s_0 \ s_1 \ \dots \ s_d]$. It is known that the Cartesian coordinates of \mathbf{s} are given by:

$$s_0 = r \frac{\|\mathbf{p}\|^2 - r^2}{\|\mathbf{p}\|^2 + r^2} \quad s_i = \frac{2r^2 x_i}{\|\mathbf{p}\|^2 + r^2} \quad \forall i = 1, \dots, d \quad (58)$$

where $\|\mathbf{p}\|^2 = \sum_{j=1}^d x_j^2$. Hence one can see that the time complexity of projection for a single d -dimensional point is $O(d)$, provided we only need the Cartesian coordinates. However, for the Stereographic Embedding procedure, one would need to calculate the angles made by \mathbf{s} with the axes of the space, making the time complexity of projection $O(\text{poly}(d))$. Therefore, the total time complexity of the Stereographic Embedding for a d -dimensional dataset D of size $|D| = N$ and k centroids is given by $O((k + N)\text{poly}(d))$. We now specify two strategies for scaling our algorithm for higher dimensional datapoints.

3.3.1. Using Qubit-Based System

We consider the case where we have two d -dimensional vectors $\mathbf{p}_1, \mathbf{p}_2 \in \mathbb{R}^d$, and we want to compute the quantum dissimilarity of vectors. If we have a qubit-based system and use dense angle encoding for encoding the stereographically projected point, we would encode the d calculated angles using $\frac{d}{2}$ qubits. Namely, for the $(d + 1)$ -dimensional projection of a d -dimensional point \mathbf{p}_1 , one would

get d angles $[\theta_1 \theta_2 \dots \theta_d]$ that specify the projected point \mathbf{s}_1 on $S^d(r)$. We then encode this vector using the same unitary as follows:

$$|\psi_1\rangle = \bigotimes_{j \in \{1,3,\dots,d-1\}} |\psi_{1_j}\rangle = \bigotimes_{j \in \text{odd}(d)} U(\theta_j, \theta_{j+1}, 0) |0\rangle. \quad (59)$$

If d is odd we can pad $[\theta_1 \theta_2 \dots \theta_d]$ with an extra 0 to make it even. The other point $\mathbf{s}_2 = s_r^{-1}(\mathbf{p}_2)$ will be encoded into the state

$$|\psi_2\rangle = \bigotimes_{j' \in \{1,3,\dots,d-1\}} |\psi_{2_{j'}}\rangle = \bigotimes_{j' \in \text{odd}(d)} U(\theta'_{j'}, \theta'_{j'+1}, 0) |0\rangle. \quad (60)$$

Now, to find the overlap between the states, one would have to perform the Bell-state measurement (Section 2.3) pairwise using $|\psi_{1_j}\rangle$ and $|\psi_{2_{j'}}\rangle$ as inputs, i.e.

$$|\langle \phi_{11} | |\psi_{1_j}\rangle \otimes |\psi_{2_{j'}}\rangle|^2 = \frac{1}{2}(1 - |\langle \psi_{1_j} | \psi_{2_{j'}}\rangle|^2) \quad (61)$$

In the common practical case of the vectors being expressed in an orthogonal basis, one would only have to find the overlap for $j = j'$. We would then have the quantum dissimilarity by adding up the individual probabilities

$$d_q(\mathbf{p}_1, \mathbf{p}_2) = \sum_{j \in \text{odd}(d)} |\langle \phi_{11} | |\psi_{1_j}\rangle \otimes |\psi_{2_j}\rangle|^2 \quad (62)$$

This procedure has a time complexity of $O(d)$. It is important point to note that this quantum dissimilarity will no longer correspond directly to the inner product or Euclidean distance between either \mathbf{p}_1 and \mathbf{p}_2 or \mathbf{s}_1 and \mathbf{s}_2 . With the strategy of pairwise overlap estimation, we see that if the number of shots to estimate $|\langle \phi_{11} | (|\psi_{1_j}\rangle \otimes |\psi_{2_j}\rangle)|^2$ is kept constant, the error in estimation will be $\propto d$. Hence, taking into account the increase in the number of shots to estimate the quantum dissimilarity with a given total error ϵ , the time complexity of this qubit implementation of overlap estimation between two points using SQ-kNN scales as $O(\epsilon^{-1} \text{poly}(d))$. Hence for all points and clusters, the time complexity would be $O(\epsilon^{-1} kN \text{poly}(d))$.

It is shown in [49] that collective measurements are a better strategy than repeated individual measurements for overlap estimation. Although this is shown in [49] for estimating overlap between two states given the availability of multiple copies of the same states, similar collective measurement strategies could be applied in this case for better results. In conclusion, the time complexity of SQ-kNN for qubit-based implementation is

$$O(\epsilon^{-1} kN \text{poly}(d)). \quad (63)$$

3.3.2. Using Qudit-Based System

Consider

$$|\mathbb{1}\rangle := \sum_{i \in \{0, \dots, d-1\}} |ii\rangle.$$

Then, for any two real vectors $|\psi\rangle = \sum \psi_i |i\rangle$ and $|\phi\rangle = \sum \phi_i |i\rangle$, namely when $\psi_i, \phi_i \in \mathbb{R}$ we have

$$\langle \mathbb{1} | (|\psi\rangle \otimes |\phi\rangle) = \sum \psi_i \phi_i = \langle \phi^* | \psi\rangle = \langle \phi | \psi\rangle.$$

Now, if we make a qudit Bell measurement², one of the basis states of this von Neumann measurement will be

$$|\Phi\rangle \equiv |\Phi\rangle_d = \frac{1}{\sqrt{d}} |\mathbb{1}\rangle.$$

Thus, the inner product between two real vectors can still be measured with a Bell measurement, but the resulting probability of measuring outcome $|\Phi\rangle$ scales as

$$|\langle\Phi|(|\psi\rangle \otimes |\phi\rangle)|^2 = \frac{1}{d} |\langle\phi|\psi\rangle|^2;$$

meaning that, as the inner product remains constant going to higher dimensions, the number of shots needed to estimate the inner product with constant precision scales polynomially in the dimension. In contrast, such complexity for the SWAP test remains constant because the contribution of the fidelity to the outcome probability is not divided by the dimension d . This is why the SWAP test is usually considered for inner product estimation, even if in the case of qubits the Bell measurement is a simpler solution [50].

3.4. SQ-kNN and Mixed States

Instead of estimating the quantum dissimilarity, we can use the datapoints produced by the ISP to perform classical kNN clustering on the 3D projected data. We called this the 3DSC-kNN (3D Stereographic Classical kNN) in Definition 6. This algorithm produces centroids that are inside the sphere. As previously pointed out, when computing the Euclidean 3D centroid on the data projected on the sphere, the result is a point inside the sphere rather than on the sphere itself.

In the Bloch sphere, internal points are mixed states, namely states with added noise. In contrast, the quantum algorithm (SQ-kNN) always produces pure centroids, namely points on the surface of the sphere. The only noiseless states are the pure states on the surface of the sphere, and thus the intuition is that arguably mixed states should not help. However, this is not immediately clear from the algorithm. Comparing 3DEC-kNN to SQ-kNN, it is thus natural to ask whether embedding the centroids into mixed states inside the Bloch sphere improves the accuracy.

Here, we show that the intuition is correct, namely that projecting into the pure state centroid is a better option. The reason is that while the quantum dissimilarity is proportional to the Euclidean dissimilarity for states in the same sphere, the same is not true for Bloch vectors with different lengths.

To allow for mixed state embedding, we can modify the definition of quantum dissimilarity (Eq. (55)) to produce mixed states whenever the 3D vector has a length of less than one. This results in the following new quantum dissimilarity.

Definition 11 (Noisy Quantum Dissimilarity). *Let $B^2(1) = \{\mathbf{P} \in \mathbb{R}^3 \mid \|\mathbf{P}\| \leq 1\}$ be the ball of radius 1. We define the noisy quantum dissimilarity as the function $\tilde{d}_q : B^2(1) \times B^2(1) \rightarrow \mathbb{R}$*

$$\tilde{d}_q(\mathbf{P}_1, \mathbf{P}_2) := \frac{1}{2} (1 - \text{Tr}(\rho_{\mathbf{P}_1} \rho_{\mathbf{P}_2})) \quad (64)$$

where $\rho_{\mathbf{P}}$ is the quantum state of the Bloch vector \mathbf{P} as in Eq. (7).

² This is obtained as in Fig. 2 where the Hadamard is replaced with the Fourier transform and the CNOT with $\sum |i\rangle\langle i| \otimes |i+j \bmod d\rangle\langle j|$. If we have multiple qubits instead of qudits, then the solution is even simpler: perform a qubit Bell measurement with each pair of qubits. This is because the tensor product of maximally entangled states is still a maximally entangled state.

Now suppose we have a convex combination of pure states; namely, suppose we have

$$\bar{\rho} = \sum_i p_i \rho_{\mathbf{P}_i} \quad \|\mathbf{P}\|_i = 1 \forall i. \quad (65)$$

where p_i is a probability distribution ($p_i > 0$ and $\sum p_i = 1$). By linearity, we have

$$\bar{\rho} = \rho \left(\sum p_i \mathbf{P}_i \right) =: \rho(\bar{\mathbf{P}}) \quad \bar{\mathbf{P}} = \sum p_i \mathbf{P}_i, \quad (66)$$

By convexity, $\bar{\mathbf{P}}$ will always lie in the sphere. Namely, we have $\|\bar{\mathbf{P}}\| \leq 1$ and thus by linearity

$$\tilde{d}_q(\bar{\mathbf{P}}, \mathbf{P}) = \sum p_i \tilde{d}_q(\mathbf{P}_i, \mathbf{P}) = \sum p_i d_q(\mathbf{P}_i, \mathbf{P}). \quad (67)$$

The result in Eq. (67) can be interpreted as another two-step process: first, repeatedly performing the Bell-state measurement of *each state* $\rho_{\mathbf{P}_i}$ that makes up the cluster and $\rho_{\mathbf{P}}$ corresponding to the datapoint, to estimate each individual dissimilarity; and then, taking the weighted average of the dissimilarities according to the composition of the mixed state centroid. This procedure is clearly impractical experimentally and also no longer correlates to the cosine dissimilarity for mixed states.

Computing the diagonalization of $\bar{\rho}$ as per Eq. (11)

$$\rho = p \rho \left(\frac{\mathbf{P}}{\|\mathbf{P}\|} \right) + (1-p) \rho \left(\frac{-\mathbf{P}}{\|\mathbf{P}\|} \right) \quad p = \frac{1}{2}(1 + \|\bar{\mathbf{P}}\|) \quad (68)$$

$$= p \psi_{\bar{\mathbf{P}}} + (1-p) \psi_{-\bar{\mathbf{P}}} \quad (69)$$

(where ψ is the Bloch embedding) makes the estimation more practical by reducing it to two estimations of d_q , namely

$$\tilde{d}_q(\bar{\mathbf{P}}, \mathbf{P}) = p \tilde{d}_q \left(\frac{\bar{\mathbf{P}}}{\|\bar{\mathbf{P}}\|}, \mathbf{P} \right) + (1-p) \tilde{d}_q \left(-\frac{\bar{\mathbf{P}}}{\|\bar{\mathbf{P}}\|}, \mathbf{P} \right) \quad (70)$$

$$= p d_q(\bar{\mathbf{P}}, \mathbf{P}) + (1-p) d_q(-\bar{\mathbf{P}}, \mathbf{P}). \quad (71)$$

The implementation portrayed at Eq. (71) simplifies the measurement procedure of the mixed state. Furthermore, instead of estimating $d_q(\pm\bar{\mathbf{P}}, \mathbf{P})$ separately, the estimation can be done directly by preparing $\psi(\bar{\mathbf{P}})$ with probability p and $\psi(-\bar{\mathbf{P}})$ with probability $1-p$, and finally collecting all the outcomes in a single estimation, which requires a larger number of shots to achieve the same precision of estimation. Another issue is that the points $\bar{\mathbf{P}}, -\bar{\mathbf{P}}$ have to be computed, which is quite time-consuming. This is true even for Eq. (67); however, a number of shots proportional to the number of Bloch vectors \mathbf{P}_i in the cluster is needed for an accurate estimation. Regardless, linearity and convexity make it clear that using mixed states can only increase the quantum dissimilarity.

Namely, while in Euclidean dissimilarity, points inside the sphere can reduce the dissimilarity, the quantum dissimilarity is proportional to the Euclidean dissimilarity only for unit vectors and actually increases for points inside the Bloch sphere. Hence we conclude that the behaviour of 3DSC-kNN does not carry over to SQ-kNN.

4. Quantum-Inspired Stereographic k -Nearest-Neighbour Clustering

We have detailed the developed quantum algorithm in the previous Section 3. This section develops the classical analogue to this quantum algorithm - the 'quantum-inspired' classical algorithm. A table summarising all the algorithms discussed in this paper, including the next one, can be found in Table 1. We begin by defining this analogous classical algorithm in terms of the clustering state (Definition 1), deriving a relationship between the Euclidean and spherical centroids given datapoints that lie on a sphere, and then proving our claim that the defined classical algorithm and previously described stereographic quantum kNN are indeed equivalent.

Recall from Lemma 3 that

$$\mathbf{c}_s^{\text{update}}(C) := \operatorname{argmin}_{\mathbf{x} \in S^n(r)} \sum_{\mathbf{p} \in C} d_s(\mathbf{x}, \mathbf{p}) = r \frac{\sum_{\mathbf{p} \in C} \mathbf{p}}{\left\| \sum_{\mathbf{p} \in C} \mathbf{p} \right\|}. \quad (72)$$

Definition 12 (2D Stereographic Classical kNN (2DSC-kNN)). Let s_r^{-1} be the ISP, and let $(D, \bar{\mathbf{c}}, \mathbb{R}^2, d_e)$ be a 2D euclidean clustering state. We define the 2D Stereographic Classical kNN (2DSC-kNN) as

$$\left(s_r^{-1}(D), s_r^{-1}(\bar{\mathbf{c}}), S^2(r), d_s, \bar{\mathbf{c}}_s^{\text{update}} \right). \quad (73)$$

Remark 3. Notice that due to the cluster update being cosine ($\bar{\mathbf{c}}_s^{\text{update}}$) and Lemma 3, we can equivalently substitute d_s with d_e , namely we can substitute it without changing the outcome of the cluster update. In our implementation, we use the Euclidean dissimilarity for simplicity of coding.

To expand upon Definition 12, for the quantum-inspired/classical analogue stereographic kNN, the steps of execution are as follows:

1. Stereographically project all the 2-dimensional data and initial centroids into the sphere $S^2(r)$ of radius r . Notice that the initial centroids will lie on the sphere by construction.
2. Cluster Update: Form the clusters using the method defined in Eq. (22), i.e. form all $C_j(\bar{\mathbf{c}}_i)$. Here, $\mathbb{D} = S^2(r)$ and dissimilarity $d = d_s(\mathbf{p}, \mathbf{c}) = \frac{1}{2r^2} d_e(\mathbf{p}, \mathbf{c})$ (Definition 5 and Lemma 2).
3. Centroid Update: A closed-form expression for the centroid update was calculated in Eq. (41) $\left(\mathbf{c}_s^{\text{updated}} = \frac{r}{\left\| \sum_{\mathbf{p} \in C} \mathbf{p} \right\|} \sum_{\mathbf{p} \in C} \mathbf{p} \right)$. This expression recalculates the centroid once the new clusters have been formed. Once all the centroids are updated, Step 2 (cluster update) is then repeated, and so on, until a stopping condition is met.

Table 1. Summary of various kNN algorithms, where SC stands for “stereographic classical” (2D or 3D) and SQ for “stereographic quantum”. Here, D is the 2-dimensional dataset, $\bar{\mathbf{c}}$ are the 2-dimensional initial centroids (initial transmission points), s_r^{-1} is the ISP into $S^2(r)$, and the dissimilarities d_e, d_s and d_q are defined in Eq. (25) and Definitions 5 and 8, respectively. The option of using d_e instead of d_s in the 2DSC-kNN is due to Remark 3.

Algorithm	Reference	Dataset D	Initial Centroids $\bar{\mathbf{c}}_1$	Dataspace \mathbb{D}	Dissimilarity d	Centroid Update $\mathbf{c}^{\text{update}}$
2DEC-kNN	Definition 4	D	$\bar{\mathbf{c}}$	\mathbb{R}^2	d_e	$\frac{1}{ \bar{\mathbf{c}} } \sum_{\mathbf{p} \in C} \mathbf{p}$
3DSC-kNN	Definition 6	$s_r^{-1}(D)$	$s_r^{-1}(\bar{\mathbf{c}})$	\mathbb{R}^3	d_e	$\frac{1}{ \bar{\mathbf{c}} } \sum_{\mathbf{p} \in C} \mathbf{p}$
2DSC-kNN	Definition 12	$s_r^{-1}(D)$	$s_r^{-1}(\bar{\mathbf{c}})$	$S^2(r)$	d_s (or d_e)	$r \frac{\sum_{\mathbf{p} \in C} \mathbf{p}}{\left\ \sum_{\mathbf{p} \in C} \mathbf{p} \right\ }$
SQ-kNN	Definition 10	$s_r^{-1}(D)$	$s_r^{-1}(\bar{\mathbf{c}})$	\mathbb{R}^3	d_q	$\sum_{\mathbf{p} \in C} \mathbf{p}$

4.1. Equivalence

We now want to show that the 2DSC-kNN algorithm of Definition 12 is equivalent to the previously defined quantum algorithm using stereographic embedding (Definition 10). For that, we first define the equivalence of two clustering algorithms.

Definition 13 (Equivalence of Clustering Algorithms). Let $\mathcal{K} = (D, \bar{\mathbf{c}}_1, \mathbb{D}, d, \bar{\mathbf{c}}_{\text{update}})$ and $\mathcal{K}' = (D', \bar{\mathbf{c}}'_1, \mathbb{D}', d', \bar{\mathbf{c}}'_{\text{update}})$ be two clustering algorithms. They are said to be equivalent if there exists a transformation $t : \mathbb{D} \rightarrow \mathbb{D}'$ such that it maps the data, initial centroids and centroid update, and clusters of \mathcal{K} to the data, initial centroids and centroid update, and clusters of \mathcal{K}' ; namely if

1. $D' = t(D)$,

2. $\bar{\mathbf{c}}'_1 = t(\bar{\mathbf{c}}_1)$ and $\bar{\mathbf{c}}'_{\text{update}} = t \circ \bar{\mathbf{c}}_{\text{update}}$,
3. $C'_j(t(\bar{\mathbf{c}})) = t(C_j(\bar{\mathbf{c}}))$ for all $j = 1, \dots, k$ and any $\bar{\mathbf{c}} \in \mathbb{D}^k$.

where we apply t elementwise and thus $t(\bar{\mathbf{c}}) = (t(\mathbf{c}_1), \dots, t(\mathbf{c}_k))$ for any list of centroids $\bar{\mathbf{c}}$ and $t(C) = \{t(\mathbf{p}) : \mathbf{p} \in C\}$ for any set of datapoints C , and where C_j are the clusters as defined in Eq. (22).

Theorem 1. SQ-kNN (Definition 10) and 2DSC-kNN (Definition 12) are equivalent.

Proof. By definition, let $(D, \bar{\mathbf{c}}_1, \mathbb{R}^2, d_e)$ be the 2D clustering state and thus giving us the SQ-kNN algorithm as

$$\mathcal{K} = (S, \bar{\mathbf{s}}_1, \mathbb{R}^3, d_q, \bar{\mathbf{c}}_q^{\text{update}}) \quad (74)$$

and the 2DSC-kNN clustering algorithm as

$$\mathcal{K}' = (S, \bar{\mathbf{s}}_1, S^2(r), d_s, \bar{\mathbf{c}}_s^{\text{update}}) \quad (75)$$

where

$$S := s_r^{-1}(D) \quad \bar{\mathbf{s}}_1 := s_r^{-1}(\bar{\mathbf{c}}_1) \quad (76)$$

Let us use the notation $\hat{\mathbf{p}} := \frac{\mathbf{p}}{\|\mathbf{p}\|}$ and define the transform $t : \mathbb{R}^3 \mapsto S^2(r)$ as $t(\mathbf{p}) = r \hat{\mathbf{p}}$, which rescales any vector to have length r . Observe that trivially for all $\mathbf{p} \in S^2(r)$, $t(\mathbf{p}) = \mathbf{p}$ and thus $t \circ s_r^{-1} = s_r^{-1}$. Therefore

$$t(S) = S, \quad t(\bar{\mathbf{s}}_1) = \bar{\mathbf{s}}_1. \quad (77)$$

Also, the equivalence of centroids is obtained since

$$t(\mathbf{c}_q^{\text{update}}(C)) = t\left(\sum_{\mathbf{p} \in C} \mathbf{p}\right) = r \frac{\sum_{\mathbf{p} \in C} \mathbf{p}}{\left\|\sum_{\mathbf{p} \in C} \mathbf{p}\right\|} = \mathbf{c}_s^{\text{update}}(C). \quad (78)$$

For the clusters, we prove the equivalence of the cluster updates as follows. We will use $d_s(\mathbf{a}, \mathbf{b}) = 4 \cdot d_q(\mathbf{a}, \mathbf{b})$ (Eq. (54)) and the fact that d_s and d_q are invariant under t , namely $d_s \circ t = d_s$ and $d_q \circ t = d_q$, or more explicitly

$$d_s(t(\mathbf{a}), \mathbf{b}) = d_s(\mathbf{a}, \mathbf{b}) = d_s(\mathbf{a}, t(\mathbf{b})), \quad d_q(t(\mathbf{a}), \mathbf{b}) = d_q(\mathbf{a}, \mathbf{b}) = d_q(\mathbf{a}, t(\mathbf{b})). \quad (79)$$

Let now $\bar{\mathbf{s}} \in (\mathbb{R}^3)^k$. Then, using the above equations and that $t(\mathbf{s}) = \mathbf{s}$, $t(S) = S$, we have

$$C'_j(t(\bar{\mathbf{s}})) = \left\{ \mathbf{p} \in S \mid d_s(\mathbf{p}, t(\mathbf{s}_j)) \leq d_s(\mathbf{p}, t(\mathbf{s}_\ell)) \quad \forall \ell = 1, \dots, k, \mathbf{p} \notin \bigcup_{\ell < j} C'_\ell(t(\bar{\mathbf{s}})) \right\} \quad (80)$$

$$= \left\{ \mathbf{p} \in S \mid d_q(\mathbf{p}, \mathbf{s}_j) \leq d_q(\mathbf{p}, \mathbf{s}_\ell) \quad \forall \ell = 1, \dots, k, \mathbf{p} \notin \bigcup_{\ell < j} C_\ell(\bar{\mathbf{s}}) \right\} \quad (81)$$

where the change in the dissimilarity inequality has also transformed the calculation of $C'_\ell(\bar{\mathbf{s}})$ into the calculation of $C_\ell(\bar{\mathbf{s}})$. We are now done, since $t(\mathbf{s}) = \mathbf{s}$ for $\mathbf{s} \in S$ and thus

$$C'_j(t(\bar{\mathbf{s}})) = \left\{ t(\mathbf{p}) \in t(S) \left| d_q(\mathbf{p}, \mathbf{s}_j) \leq d_q(\mathbf{p}, \mathbf{s}_\ell) \quad \forall \ell = 1, \dots, k, \mathbf{p} \notin \bigcup_{\ell < j} C_\ell(\bar{\mathbf{s}}) \right. \right\} \quad (82)$$

$$= t(C_j(\bar{\mathbf{s}})) \quad (83)$$

This concludes the proof \square

The following discussion provides a visual intuition of Theorem 1. In Fig. 4, the sphere with centre origin (O) and radius r is the stereographic sphere into which the 2-dimensional points are projected, while the sphere with centre O and radius 1 is the Bloch sphere. The points $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$ are the stereographically projected points defining a cluster, corresponding to the previously used labels $\mathbf{s}_1, \mathbf{s}_2, \dots, \mathbf{s}_n$. The centroid \mathbf{c}_e is obtained with the euclidean average in \mathbb{R}^3 . In contrast, the centroid \mathbf{c}_s is restricted to be in $S^2(r)$ and equal \mathbf{c}_e rescaled to lie on this sphere. The quantum states $|\psi_{\mathbf{p}_1}\rangle, |\psi_{\mathbf{p}_2}\rangle, \dots, |\psi_{\mathbf{p}_n}\rangle$ are obtained after Bloch embedding the stereographically projected points $\mathbf{p}_1, \mathbf{p}_2, \dots, \mathbf{p}_n$, and $|\psi_{\mathbf{c}}\rangle$ is the quantum state obtained after Bloch embedding the centroid. The points marked on the Bloch sphere in Fig. 4 are the Bloch vectors of the quantum states $|\psi_{\mathbf{p}_1}\rangle, |\psi_{\mathbf{p}_2}\rangle, \dots, |\psi_{\mathbf{p}_n}\rangle$ and $|\psi_{\mathbf{c}}\rangle$.

One can see from Definition 7 that the origin, any point \mathbf{p} on the sphere and $|\psi_{\mathbf{p}}\rangle$ are collinear. Hence, it can be seen that in the process of SQ-kNN clustering, the points on the stereographic sphere are projected radially into the sphere of radius 1. Once the labels were assigned in the previous iteration, the new centroid is computed, giving an integer multiple of the average point \mathbf{c}_e , which lies within the stereographic sphere. Crucially, when we embed this new centroid into the quantum state for the quantum dissimilarity calculation of the next step, since we only use the polar and azimuthal angle of the point for embedding (see Definition 7), *the prepared quantum state is also projected into the surface of the Bloch sphere* - or, in other words, a pure state is prepared ($|\psi_{\mathbf{c}}\rangle$). Hence, we can see that all the dissimilarity calculations in the SQ-kNN will take place between points on the *surface* of the Bloch sphere, even though the calculated quantum centroid is *contained outside* the stereographic sphere. This argument also illustrates why *any* point on the ray $\overrightarrow{O\mathbf{c}_e\mathbf{c}_s}$ can be used for the centroid update step of the stereographic quantum kNN; any chosen point on the ray, when embedded into a quantum state for dissimilarity calculations will reduce to $|\psi_{\mathbf{c}}\rangle$.

In short, we know from Lemma 3 that O, \mathbf{c}_e , and \mathbf{c}_s lie on a straight line. Therefore one can see that if the Bloch sphere is scaled by r , the point on the Bloch sphere corresponding to $|\psi_{\mathbf{c}}\rangle$ will transform to \mathbf{c}_s , i.e. $0, |\psi_{\mathbf{c}}\rangle, \mathbf{c}_e$ and \mathbf{c}_s are all collinear. Eq. (55) shows that SQ-kNN clustering clusters points on a sphere as per Euclidean dissimilarity; that implies that simply scaling the sphere makes no difference to the clustering. Therefore, we conclude that clustering on the surface of the stereographic sphere S^2 (2DSC-kNN) is equivalent to the quantum algorithm with stereographic embedding (SQ-kNN).

4.2. Complexity Analysis and Scaling

As we showed in Section 3.2, the time complexity of ISP for calculating Cartesian coordinates of a d -dimensional vector is $O(d)$. Hence the total time complexity of projection for the 2DSC-kNN will be $O((k + N)d)$ where $N = |D|$ is the total number of points, and k is the total number of centroids. Since the cluster update step uses Euclidean dissimilarity, it will take $O(kNd)$ time in total ($O(d)$ for each distance calculation, which is to be done for each pair of N points and k centroids). The centroid update expression (Eq. (41)) can be calculated in $O(Nd)$ making the total time for this step $O(kNd)$ since we have k centroids. Hence we have

$$\text{Time complexity of 2DSC-kNN algorithm} = O(kNd), \quad (84)$$

on par with the classical k-means clustering algorithm, and at least polynomially faster than the stereographic quantum kNN (Eq. (63)) or Lloyd's quantum clustering algorithm (taking into account input assumptions) [2,9,13].

5. Experiments and Results

We defined the procedure for SQ-kNN in Section 3. Section 3.1 introduces our idea for state preparation - projecting the 2-dimensional data points into a *higher* dimension. Section 3.1 details the hybrid quantum-classical method used for our process and then proves that the output of the quantum circuit is not only a valid but also an excellent metric that can be used for distance estimation between two points. Section 4 describes the quantum-inspired classical algorithm analogous to the quantum algorithm (2DSC-kNN). In this section, we test and compare the quantum-inspired rather the quantum algorithm for two main reasons:

- The hardware performance and availability of quantum computers (NISQ devices) is currently so much worse than that of classical computers that no advantage can likely be obtained with the quantum algorithm.
- The goal of this paper is not to show a “quantum advantage” in time complexity over the classical k-means in the NISQ context - it is to show that stereographic projection can lead to better learning for classical clustering and be a better embedding for quantum clustering. In particular, the equivalence between 2DSC-KNN and SQ-KNN proves that noise is the only limitation for the stereographic quantum algorithm to achieve the accuracy of the quantum-inspired algorithm.

All the experiments were carried out on a server with the following specifications: 2 Intel Xeon E5-2687W v4 chips clocked at 3.0 GHz (24 cores / 48 threads), 128GB RAM. All experiments are performed on the real-world 64-QAM data provided by Huawei (see Section 2.1 and Appendix A). Due to the extensive nature of testing and the large volume of analysis generated, we do not present all the figures in the following sections. Figures which sufficiently demonstrate general trends and observations have been included here. An exhaustive collection of all figures and other such analysis results, as well as the source code, real-world data, and raw data collected, can be obtained from the [GitHub repository of the project](#).

The terminology used is as follows:

1. *Radius*: the radius of the stereographic sphere into which the 2-dimensional points are projected.
2. *Number of points*: the number of points upon which the clustering algorithm was performed. For every experiment, the selected points were a random subset of all the 64-QAM data (of a specific noise) with cardinality equal to the required number of points. The random subset is created using the `numpy.random.sample()` from the Python Numpy library.
3. *Number of runs*: Since for each choice of parameters for each experiment we select a subset of points at random, we repeat each of the experiments many times to remove bias from the random choice and obtain stable averages and standard deviations for the collected performance parameters (described in another list below). This number of repetitions is the “number of runs”.
4. *Dataset Noise*: As explained in Section 2.1, data was collected for four different input powers. The data is divided into four datasets labelled with powers 2.7, 6.6, 8.6 and 10.7 dBm.
5. *Natural endpoint*: The natural endpoint of a clustering algorithm occurs when

$$C_j(\bar{\mathbf{c}}_{i+1}) = C_j(\bar{\mathbf{c}}_i) \quad \forall j = 1, \dots, k \quad (85)$$

i.e. when all the clusters remained unchanged (stay the same) even after the centroid update. It is the natural endpoint since if the clusters do not change, the centroids will not change either in the next iteration, leading to the same clusters (Eq. (85)) and centroids for all future iterations.

The algorithms that we test are:

- **2DSC-kNN**: The quantum-analogue algorithm of Definition 12, the classical equivalent of the SQ-KNN and the most important candidate for our testing.
- **2DEC-kNN**: The standard classical kNN of Definition 4 implemented upon the original 2-dimensional dataset ($n = 2$), which serves as a baseline for performance comparison.
- **3DSC-kNN**: The standard classical kNN, but implemented upon the stereographically projected 2-dimensional dataset, as defined in Definition 6. We again emphasise that in contrast to the 2DSC-kNN the centroid lies *within* the sphere, and in contrast to the 2DEC-kNN, the clustering takes place in \mathbb{R}^3 . This algorithm serves as another control, to gauge the relative impacts of stereographically projecting the dataset versus restricting the centroid to the surface of the sphere. It is an intermediate step between the 2DSC-kNN and the 2DEC-kNN algorithms.

From these algorithms, we measure the following performance parameters (or KPIs, Key Performance Indicators):

- *Accuracy*: Since we have the true labels of the datapoints available, we can measure the accuracy of the algorithm as the percentage of points that have been given the correct label, i.e. symbol accuracy rate. All accuracies are recorded as a percentage.
- *Symbol or Bit error rate*: As mentioned in Appendix A, due to Gray encoding, the bit error rate is approximately $\frac{1}{6}$ of the symbol error rate, which in turn is simply one minus the accuracy. Although error rates are the standard performance parameter in channel coding, we decided to measure the accuracy instead, which is the standard performance parameter in machine learning.
- *Accuracy gain*: The gain is calculated as (*accuracy of candidate algorithm - accuracy of 2-dimensional classical k-means clustering algorithm*), i.e. it is the increase in accuracy of the algorithm over the baseline, defined as the accuracy of the classical k-means clustering algorithm acting on the 2D dataset for those number of points.
- *Number of iterations*: One iteration of the clustering algorithm occurs when the algorithm performs the cluster update followed by the centroid update (the algorithm must then perform the cluster update again). The number of times the algorithm repeats these two steps before stopping is the number of iterations. We use the number of iterations the algorithm requires to reach its 'natural endpoint' as a proxy for *convergence performance*. The lesser the number of iterations performed, the faster the algorithm's convergence. The number of iterations does not directly correspond to time performance since the time taken for one iteration differs between all algorithms.
- *Iteration gain*: The gain in iterations is defined as (*the number of iterations of 2D k-means clustering algorithm - the number of iterations of candidate algorithm*), i.e. the gain is how many fewer iterations the candidate algorithm took than the 2DEC-kNN algorithm to reach its natural endpoint.
- *Execution time*: The amount of time taken for a clustering algorithm to give the final output (the final centroids and clusters) given the 2-dimensional data points as input, i.e. the time taken end to end for the clustering process. All times in this work are recorded in *milliseconds* (ms).
- *Execution time gain*: This gain is calculated as (*the execution time of 2DEC-kNN k-means clustering algorithm - the execution time of candidate algorithm*).
- *Overfitting Parameter*: (accuracy in testing – accuracy in training).

With these algorithms and variables, we perform two main experiments:

1. The Overfitting Test: The dataset is divided into a 'training' and a 'testing' set, to characterise the clustering and classification performance of the algorithms.
2. The Stopping Criterion Test: The iterations and other performance parameters are varied, to test whether and what kind of stopping criterion is required.

We see that the tested algorithms display some very promising and interesting results. We manage to get improvements in accuracy and convergence performance almost across the board, and we discover the very important optimisation parameters of the radius of projection and the stopping criterion.

5.1. Experiment 1: Overfitting

Here, the datasets were divided into training and testing data. First, a random subset of cardinality equal to the number of points was chosen from the dataset, and then 80% of the selected points were assigned as 'Training Data', while the other 20% was assigned as 'Testing Data'.

In the training phase, the algorithms were first run on the training data with the maximum possible iterations set to 50 to keep an acceptable running time. The stopping criterion for all algorithms was chosen as the natural endpoint - the algorithm stopped either when the number of iterations hit 50, or when the natural endpoint was reached (whichever happened first). The final centroid co-ordinates ($\bar{c}_{\text{last iteration}}$) were recorded in the training phase, to be used for the testing phase, along with several performance parameters. The recorded performance parameters were the algorithm's accuracy, the number of iterations taken and the execution time.

Once the training was over, the centroids calculated at the end of training were used as the initial centroids for the testing set datapoints, and the algorithm was run with the maximum number of iterations set to 1, i.e. the calculated centroids were then used to classify the remaining points as per the dissimilarity and dataspace of each algorithm. The recorded performance parameters were the algorithm's accuracy and execution time. Once both the testing and training accuracy had been recorded, the overfitting parameter was also recorded.

For each set of input variables (just the number of points for 2DEC-kNN clustering, the radius and number of points for the 2DSC-kNN and 3DSC-kNN clustering), the entire experiment (training and testing) was repeated 10,000 times in batches of 100 to calculate reasonable standard deviations for every performance parameter.

There are several reasons for this choice of experiment:

- It exhaustively covers all the parameters that can be used to quantify the performance of the algorithms. We were able to observe very important trends in the performance parameters with respect to the choice of radius and the effect of the number of points (affecting the choice of when one should trigger the clustering process on the collected received points).
- It avoids the commonly known problem of overfitting. Though this approach is not usually used in testing the kNN due to its iterative nature, we felt that from a machine learning perspective, it is useful to know how well the algorithms perform in a classification setting as well.
- Another reason that justifies the training and testing approach (clustering and classification) is the nature of the real-world application setup. When transmitting QAM data through optical-fibre, the receiver receives only one point at a time and has to classify the received point to a given cluster in real-time using the current centroid values. Once a number of data points have accumulated, the kNN algorithm can be run to update the centroid values; after the update, the receiver will once again perform classification until some number of points has been accumulated. Hence, we can see that in this scenario, the clustering and the classification performance of the chosen method become important.

5.1.1. Results

We begin the presentation of the results of this experiment by first showing the characterisation of the 2DSC-kNN algorithm with respect to the input variables.

Fig. 5 characterises the testing and training accuracy of the 2DSC-kNN algorithm acting upon the 2.7dBm dataset, i.e. classification and clustering performance respectively. Fig. 6 portrays the same results in the form of a heat map, with a focus on the region of interest of the algorithm. These figures are representative of the trends of all four datasets.

Fig. 7 characterises the convergence performance of the quantum algorithm - it shows how the number of iterations required to reach the natural endpoint of the 2DSC-kNN algorithm varies as the number of points and radius of projection changes. Once again, the figures for all the other datasets follow the same pattern as the included figures.

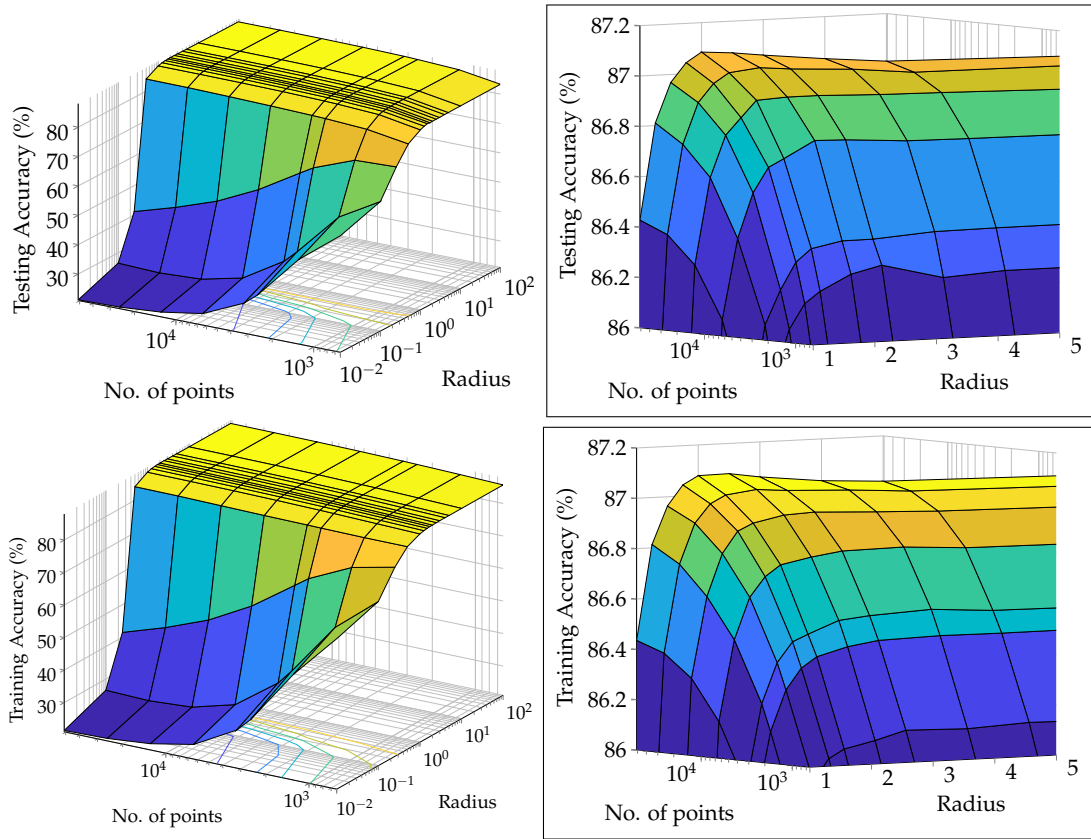


Figure 5. Mean *testing* (top) and *training* (bottom) accuracy vs Number of points vs Projection radius for the 2DSC-kNN algorithm acting upon the 2.7 dBm dataset. Full data on the left and a close-up on the right.

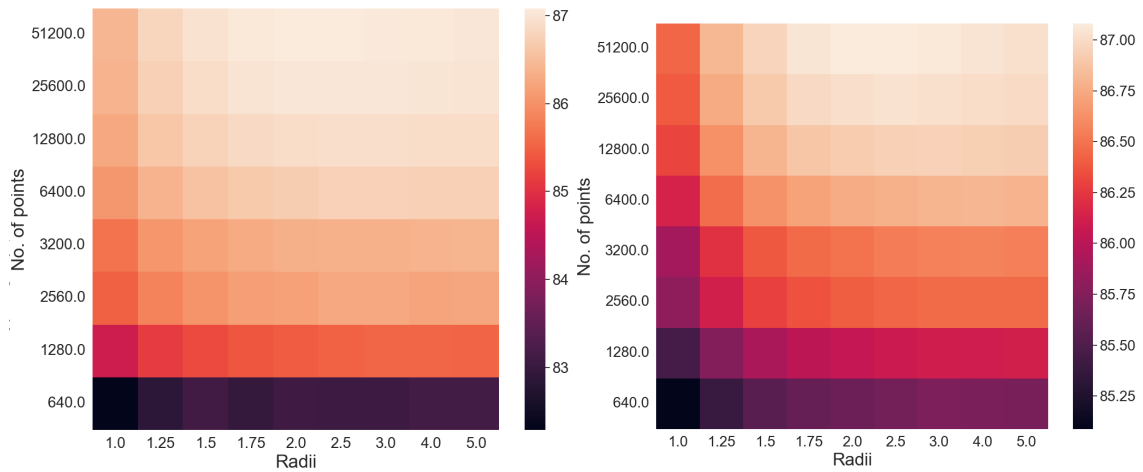


Figure 6. Heat map of Mean accuracy vs $\ln(\text{Number of points})$ vs projection radius for the 2DSC-kNN algorithm acting upon the 2.7 dBm dataset. Testing on the left, training on the right.

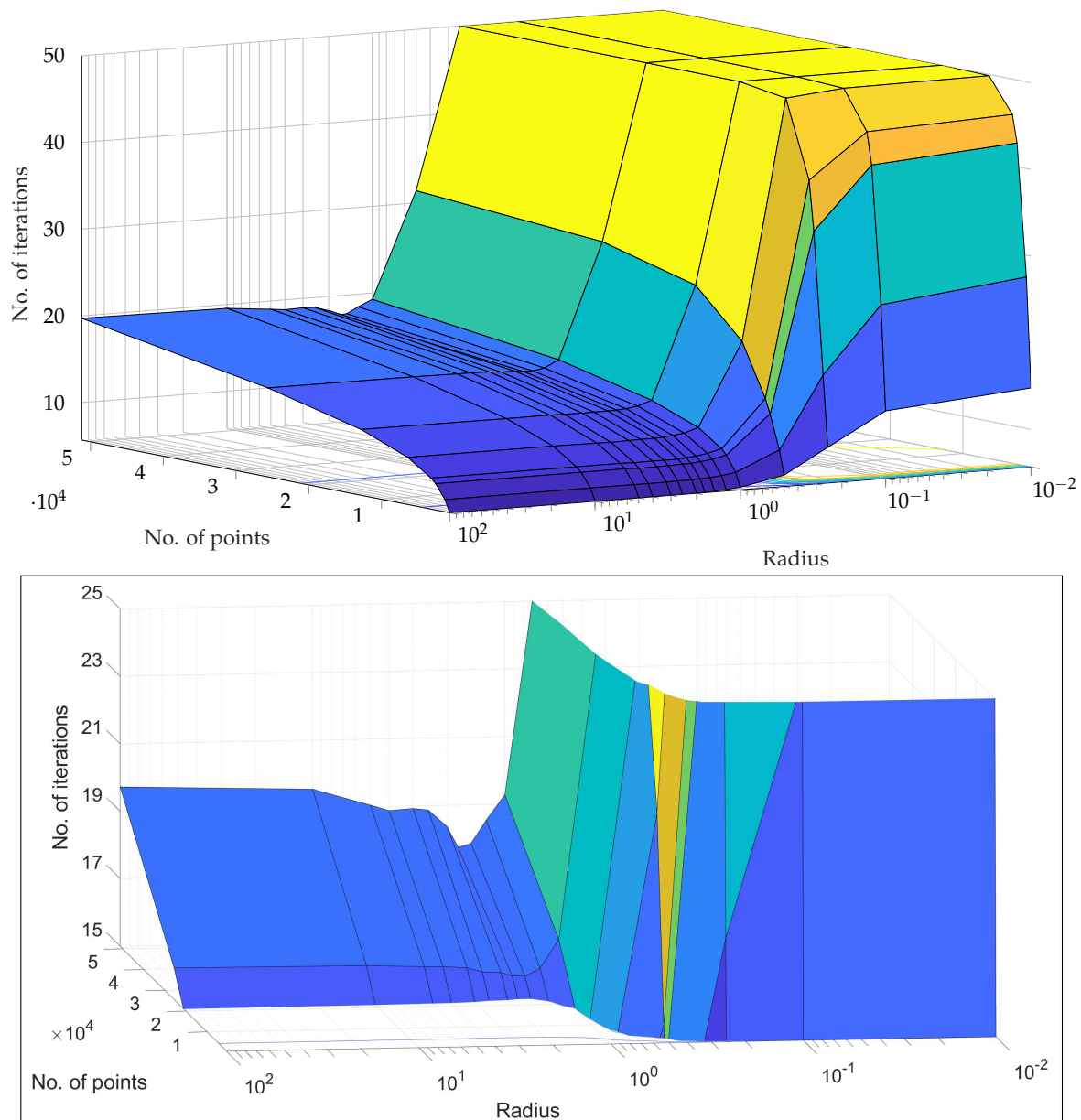


Figure 7. Mean number of iterations in *training* vs Number of points vs $\log(\text{projection radius})$ for the 2DSC-kNN algorithm acting upon the 10.7 dBm dataset. Full data at the top and close up at the bottom.

We then compare the performance of the 2DSC-kNN algorithm with that of the 3DSC-kNN and 2DEC-kNN algorithms.

Accuracy Performance

In all the figures in this section, the winner is chosen as the radius for which the maximum accuracy is achieved for the given number of points.

Fig. 8 depicts the variation in testing accuracy with the number of points for all three algorithms along with error bars. As mentioned before, this characterises the performance of the algorithms in ‘classification’ mode, that is, when the received points must be decoded in real-time.

Fig. 9 portrays the trend in training accuracy with the number of points for all three algorithms along with error bars. This characterises the performance of the algorithms in ‘clustering’ mode, that is, when the received points must be used to update the centroid for future re-classification or if the received datapoints are stored and decoded in batches.

Fig. 8 and Fig. 9 also plot the gain in testing and training accuracies respectively for the 3DSC-kNN and 2DSC-kNN algorithms. The label of the points in these figures is the radius of ISP for which that accuracy gain was achieved.

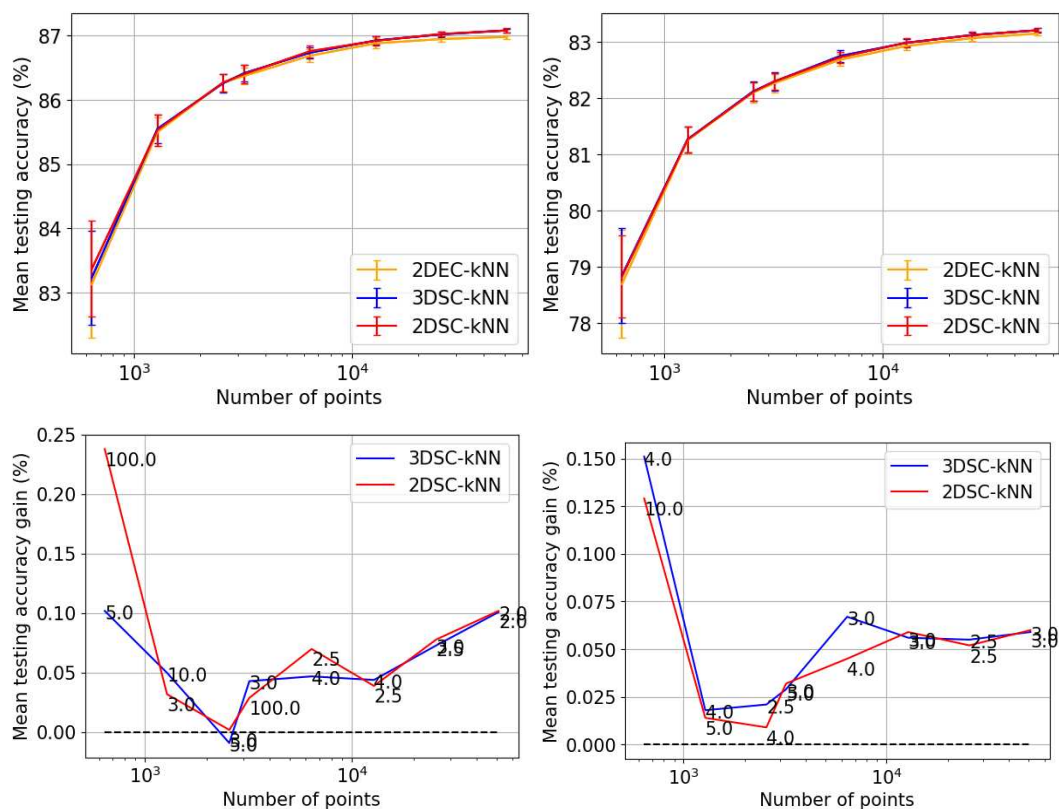


Figure 8. Maximum mean testing accuracy (top) and maximum accuracy gain (bottom) among all tested radii vs number of points (logarithm scale) for the 2.7dBm (left) and 10.7dBm (right) datasets. The labels at the bottom figures correspond to the radius where each value was achieved.

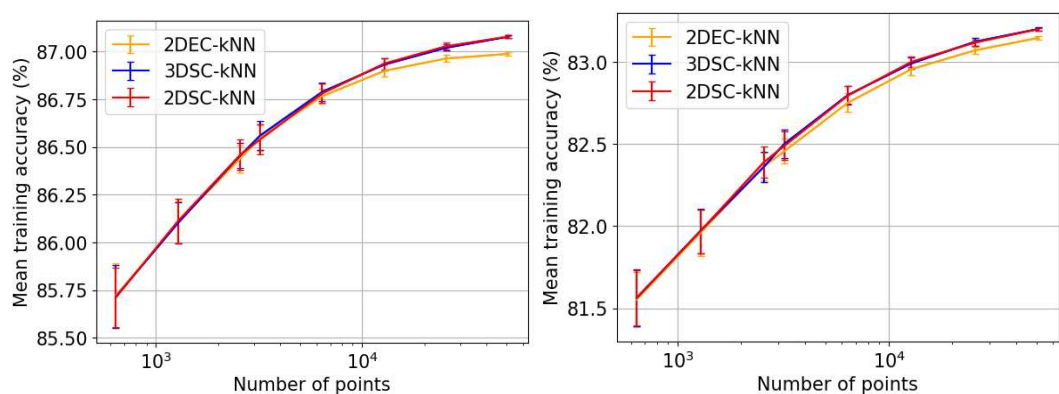


Figure 9. Cont.

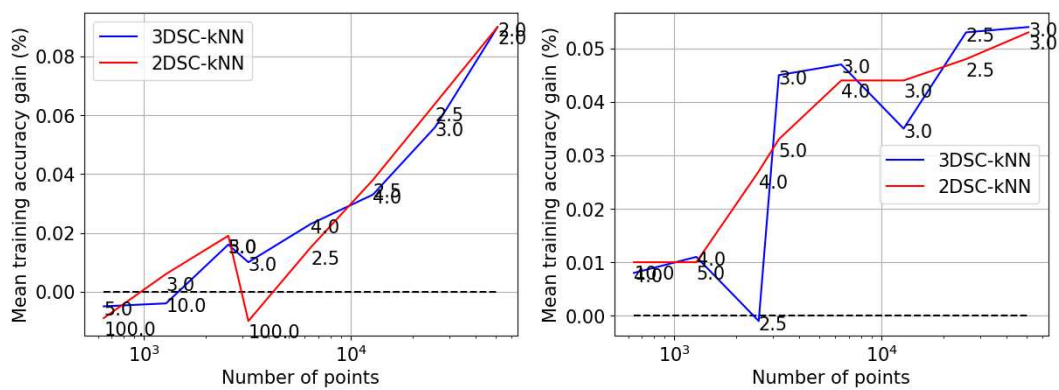


Figure 9. Maximum mean training accuracy (top) and accuracy gain (bottom) among all tested radii vs number of points (logarithm scale) for the 2.7dBm (left) and 10.7dBm (right) datasets. The labels at the bottom figures correspond to the radius where each value was achieved.

Iteration Performance

In all the figures in this section, the winner is chosen as the radius for which the minimum number of iterations is achieved for the given number of points.

Fig. 10 shows how the required number of iterations for all three algorithms varies as the number of points increases. Fig. 10 also displays the gain of the 2DSC-kNN and 3DSC-kNN algorithms in the number of iterations to reach their natural endpoints. The label of the points in these figures is the radius of ISP for achieving that iteration gain.

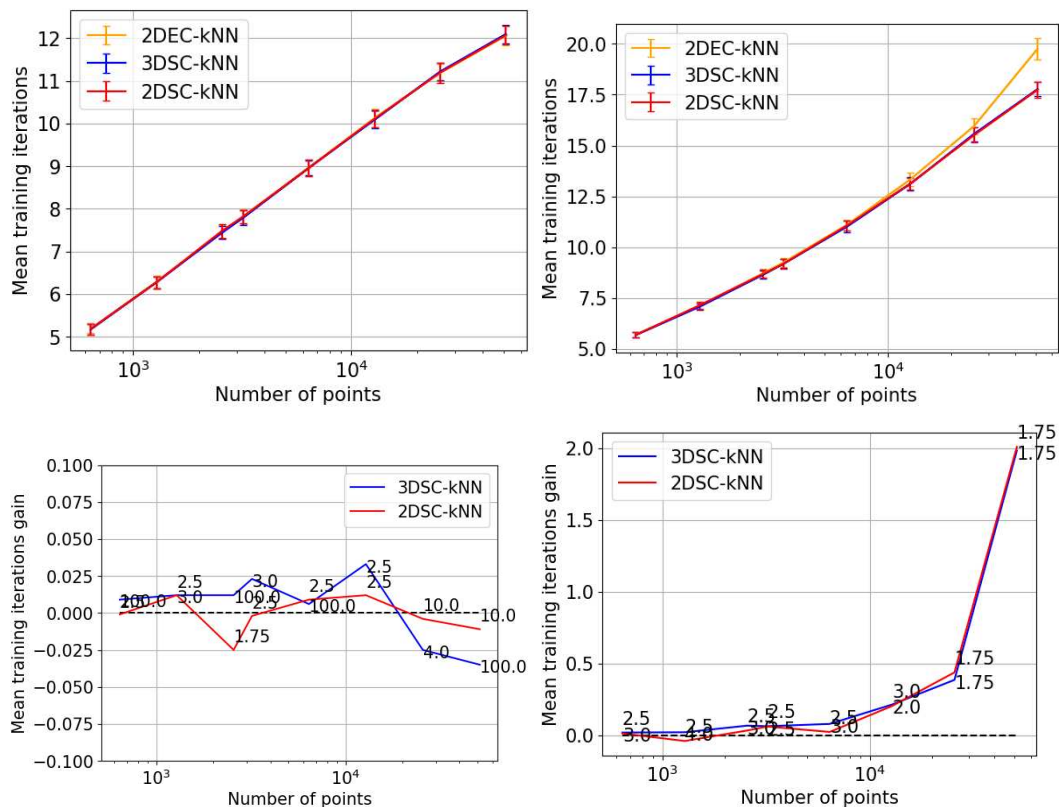


Figure 10. Best mean training iterations (top) and iteration gain (bottom) performance among all tested radii vs number of points (logarithm scale) for the 2.7 and 10.7dBm datasets. The labels at the bottom figures correspond to the radius where each value was achieved.

Time Performance

In all the figures in this section, the winner is chosen as the radius for which the minimum execution time is achieved for the given number of points.

Fig. 11 puts forth the dependence of testing execution time upon the number of points for all three algorithms along with error bars. As mentioned before, these times are effectively the amount of time the algorithm takes for one iteration. This characterises the performance of the algorithms when performing direct classification decoding of the received points in real time.

This figure reveals the trend in training execution time with the number of points for all three algorithms, along with error bars. This characterises the time performance of the algorithms when performing clustering - that is, when the received points must be used to update the centroid for future re-classification or if the received datapoints are stored and decoded in batches.

Fig. 12 graphs the gains in testing and training execution times for the 3DSC-kNN and 2DSC-kNN algorithms.

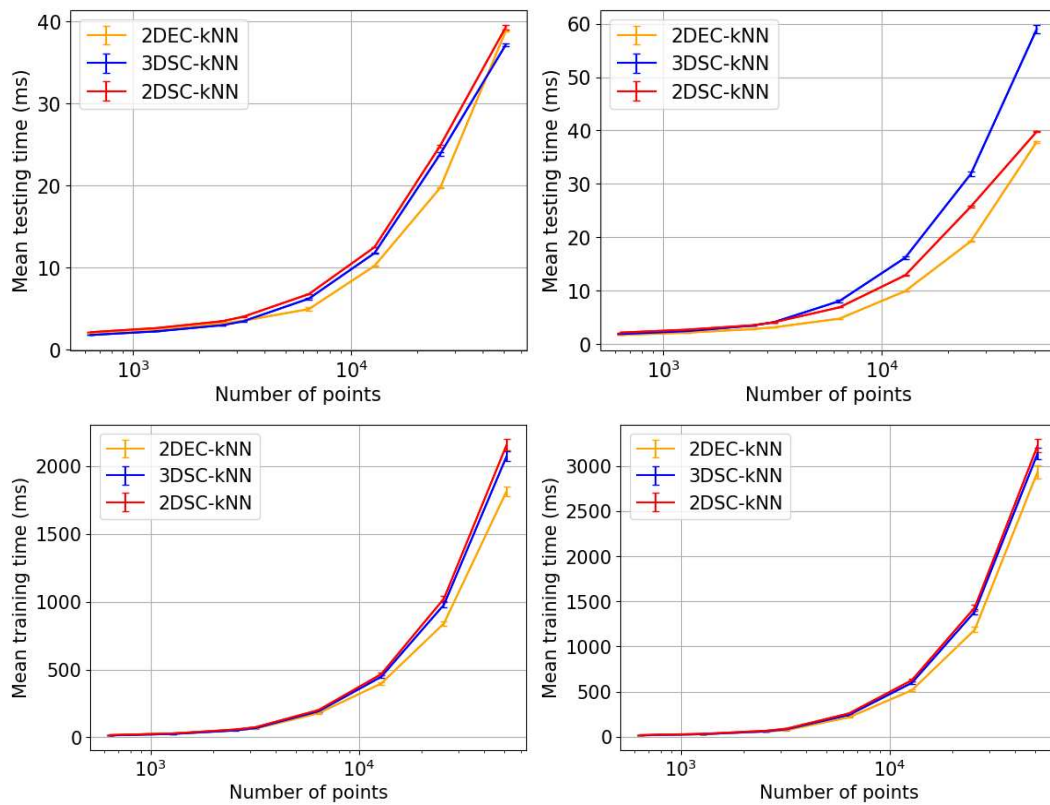


Figure 11. Best mean testing (top) and training (bottom) execution time among all tested radii number of points (logarithm scale) for the 2.7dBm (left) and 10.7dBm (right) datasets.

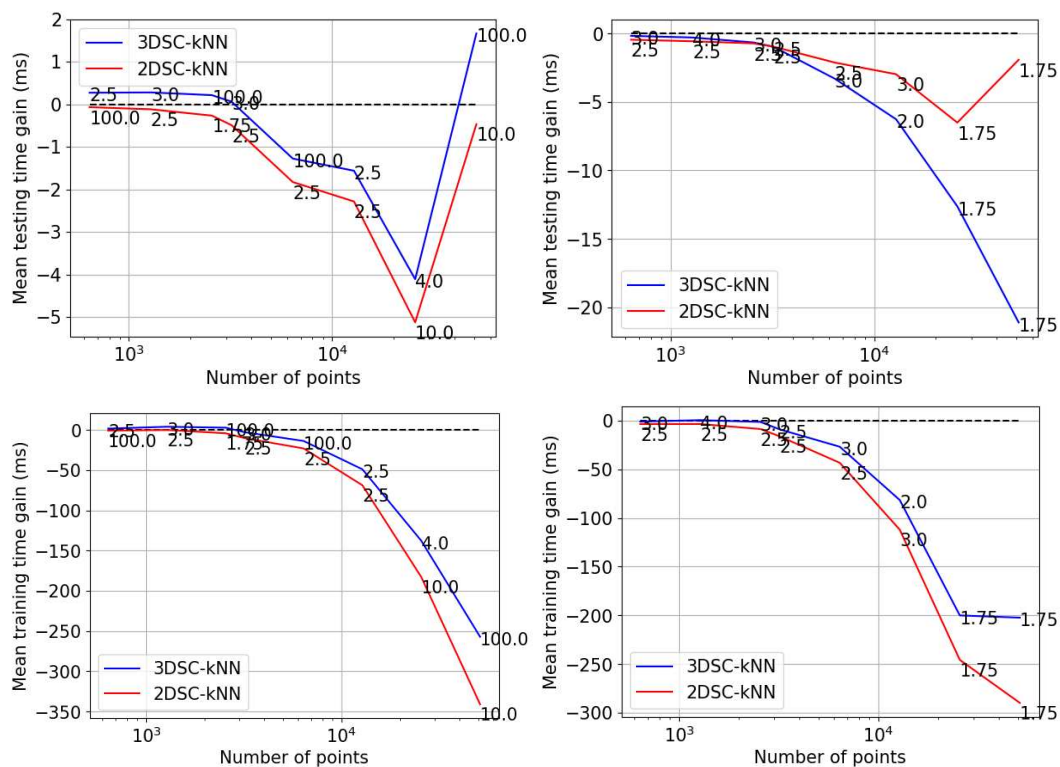


Figure 12. Best mean testing (top) and training (bottom) execution time gain vs number of points (logarithm scale) for the 2.7dBm (left) and 10.7dBm (right) datasets. The labels at the figures correspond to the radius where each value was achieved.

Overfitting Performance

Fig. 13 exhibits how the overfitting parameter for the 2DEC-kNN k-means clustering, 3DSC-kNN and 2DSC-kNN algorithms vary as the number of points changes.

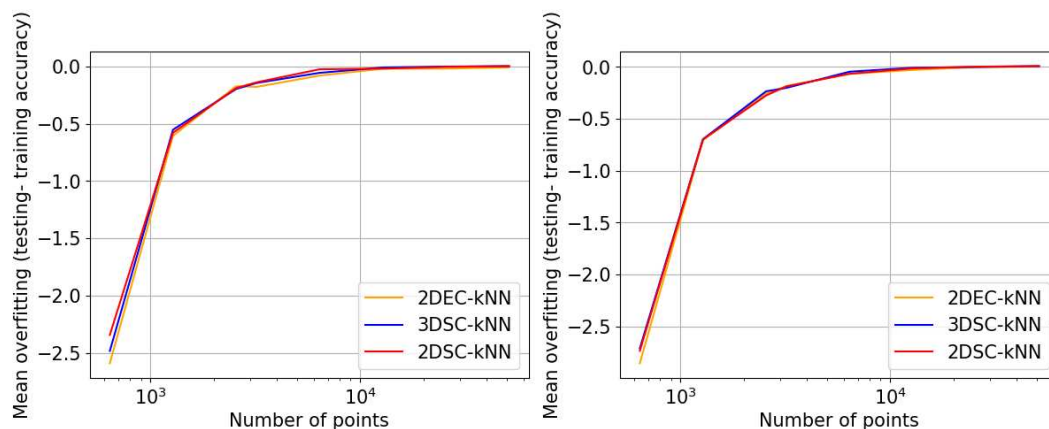


Figure 13. Mean overfitting parameter vs the natural log of the number of points for the 2.7dBm (left) and 10.7dBm (right) datasets.

5.1.2. Discussion and Analysis

From Fig. 5 we can see that there is an *ideal radius* > 1 for which maximum accuracy is achieved. This ideal radius is usually between 2 and 5 for our datasets. For a good choice of radius (> 1), the accuracy increases monotonically (with an upper bound) with the number of points. In contrast, for a poor choice of radius (< 1), the accuracy nosedives as the number of points increases. This is due to the clusters getting squished together near the North pole of the stereographic sphere (the point

$(0, 0, r)$). If one is dealing with a large number of points, the accuracy becomes even more sensitive to the choice of radius as the decline in accuracy for a bad radius is much steeper as the number of points increases. These observations hold for both training and testing accuracy (classification and clustering), regardless of the noise in the dataset. These observations are also well-reflected in the heatmaps, where one can see that the best training and testing performance is for $r = 2$ to 3 and the maximum number of points. It would seem that choosing too large of a radius is not too harmful. This might hold true for the classical algorithms, but when the quantum algorithm is deployed, all the points will be clustered around the South pole of the Bloch sphere and even minimal noise in the quantum circuit will degrade performance. Hence, there is a sweet spot of radius to be chosen.

Fig. 7 also shows that there is an ideal radius > 1 for which one needs the minimum number of iterations to reach the natural endpoint. This ideal radius is once again between 2 and 5 for our datasets. As the number of points increases, the number of iterations always increases. The increase is minimal for a good choice of radius, while for a bad choice, the convergence is very slow. For our experiments, we chose the maximum iterations as 50 hence the observed plateau at 50 iterations. If one is dealing with a large number of points, the convergence becomes more sensitive to the choice of radius. The increase in iterations for a poor choice of radius is much steeper. 2DSC-kNN algorithm and 3DSC-kNN algorithm display near-identical performance.

From Fig. 8 we can see that both 2DSC-kNN algorithm and 3DSC-kNN algorithm perform better in accuracy than the 2DEC-kNN algorithm for all datasets. The advantage becomes more definitive as the number of points increases as the increase in accuracy moves beyond the error bar. We observe the highest increase in accuracy for the 2.7dBm dataset.

In Fig. 9 one can see the noticeably better performance of the 2DSC-kNN algorithm and 3DSC-kNN algorithm over the 2DEC-kNN algorithm for all datasets than in the testing case (classification mode). Once again the 2.7dBm dataset shows the maximum increase. The advantage again becomes more definitive as the number of points increases as the increase in accuracy moves beyond the error bar. The 2DSC-kNN algorithm and 3DSC-kNN algorithm show an almost identical performance.

From Fig. 8 and Fig. 9 we can also see that almost universally for both algorithms, the gain is greater than 0, i.e. we beat the 2DEC-kNN algorithm in nearly every case! We can also see that the best radius is almost always between 2 and 5. Another observation is that the gain in training accuracy increases with the number of points. The figures further display how similarly the 3DSC-kNN algorithm and 2DSC-kNN algorithm perform in terms of accuracy, regardless of noise.

From Fig. 10 it can be concluded that for low noise datasets, since the number of iterations is already quite low, there is not much gain or loss; all three algorithms perform almost identically. For high-noise datasets, however, both the 3DSC-kNN algorithm and 2DSC-kNN algorithm show significant performance improvement, especially for a higher number of points. For a high number of points, the improvement is beyond the error bars and hence very significant. It can be noticed that the ideal radius for minimum iterations is once again between 2 and 5. Here also the 3DSC-kNN algorithm and 2DSC-kNN algorithm perform similarly, with the 2DSC-kNN algorithm performing better in certain cases.

One learns from Fig. 11 that most importantly, the 2DSC-kNN algorithm and 2DEC-kNN algorithm take nearly the same amount of time for execution in classification mode, and the 2DSC-kNN algorithm in most cases beats the 3DSC-kNN algorithm. Here too, the gain is significant since it is much beyond the error bar. The execution time increases linearly with the number of points, as expected. These conclusions are supported by Fig. 12.

Since the 2DSC-kNN algorithm takes almost the same time, and provides greater accuracy, *it is an ideal candidate to replace the 2DEC-kNN algorithm for classification applications.*

Fig. 11 also shows that all three algorithms take almost the same amount of time for training, i.e. in clustering mode. The 3DSC-kNN and 2DSC-kNN algorithms once again perform almost identically, almost always slightly worse than 2DEC-kNN clustering. Fig. 12 supports these observations. Here execution time increases linearly with the number of points as well, as expected.

In Fig. 13, all three algorithms have nearly identical performance. As expected, the overfitting decreases with an increase in the number of points.

5.2. Experiment 2: Stopping criterion

Based on the results obtained from the first experiment, we performed another experiment to see how the accuracy of the algorithms varies iteration by iteration. It was observed that the natural endpoint of the algorithm was rarely the ideal endpoint in terms of performance. Hence we wished to observe the performance of each algorithm as the number of iterations progressed.

In this experiment, the entire random subset of datapoints was used for the clustering algorithm. The algorithms were run on the dataset, and the accuracy of the algorithms at each iteration as well as the iteration number of the natural endpoint was recorded. The maximum number of iterations was once again 50. By repeating this 100 times for each number of points (and radius, if applicable), we obtained the general performance variation of each algorithm with the iteration number. The input variables were the number of points, the radius of the stereographic sphere and the iteration number; the recorded performance parameters were the accuracy and probability of stopping.

This experiment revealed that the natural endpoint was indeed a poor choice of stopping criterion and that the endpoint should be chosen per some "loss function". It also revealed some important trends in the performance parameters which not only emphasised the importance of the choice of radius and number of points but also gave greater insight into the disadvantages and advantages of each algorithm.

5.2.1. Results

Characterisation of the 2DSC-kNN algorithm

Fig. 14 depicts the dependence of the accuracy of the 2DSC-kNN algorithm upon the iteration number and projection radius for the 2.7dBm dataset. The figures for the rest of the datasets follow the same trends and are nearly identical in shape.

Fig. 15 shows the dependence of the probability of the 2DSC-kNN algorithm reaching its natural endpoint versus the radius of projection and iteration number for the 10.7dBm dataset with 51200 points and for the 2.7dBm dataset with 640 points. Once again, the figures for the rest of the datasets follow the same trends and their shape can be extrapolated from the presented Fig. 15.

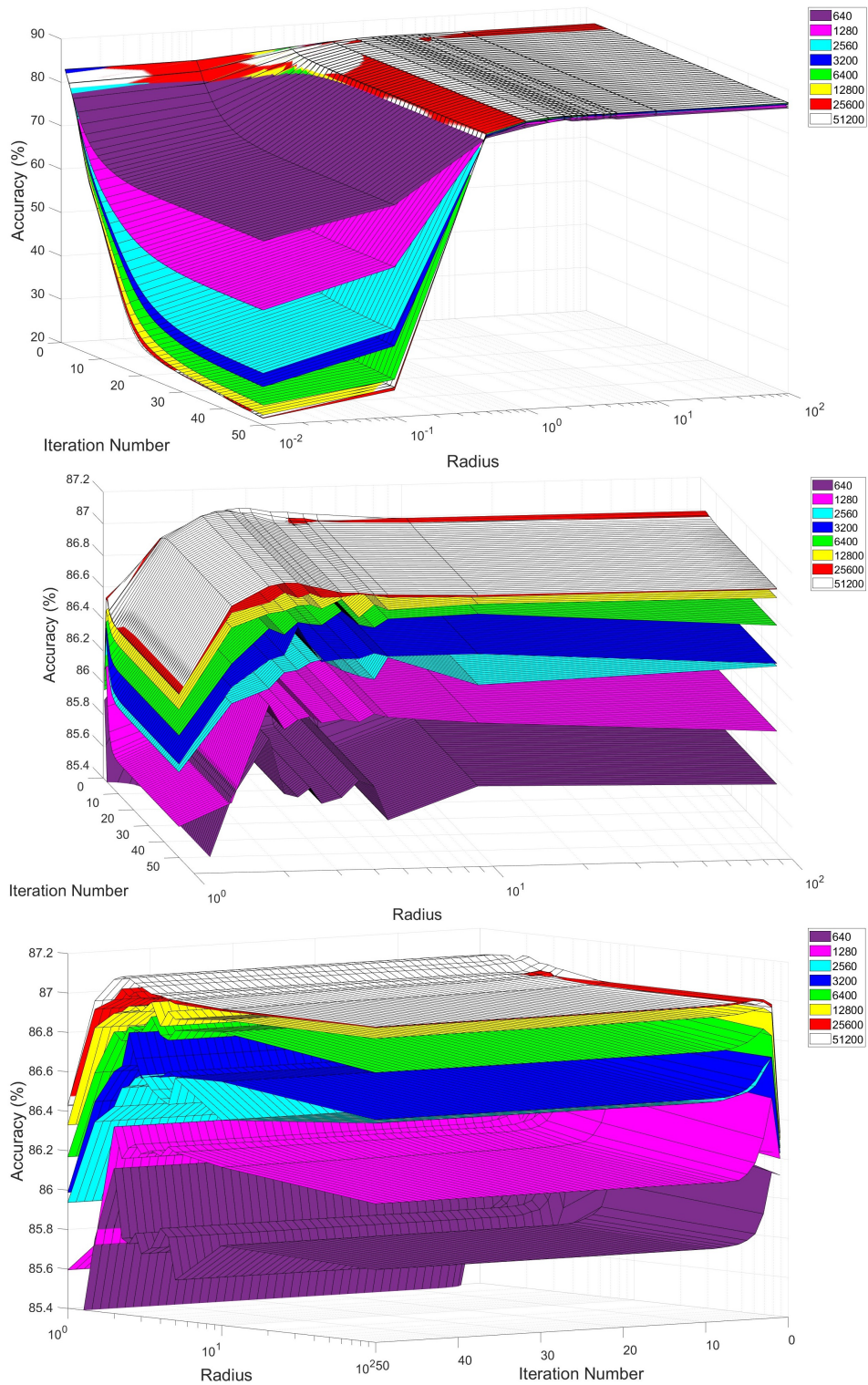


Figure 14. Maximum Accuracy vs iteration number vs projection radius for the 2DSC-kNN algorithm acting upon the 2.7 dBm dataset. Close-up of the maximas at the bottom.

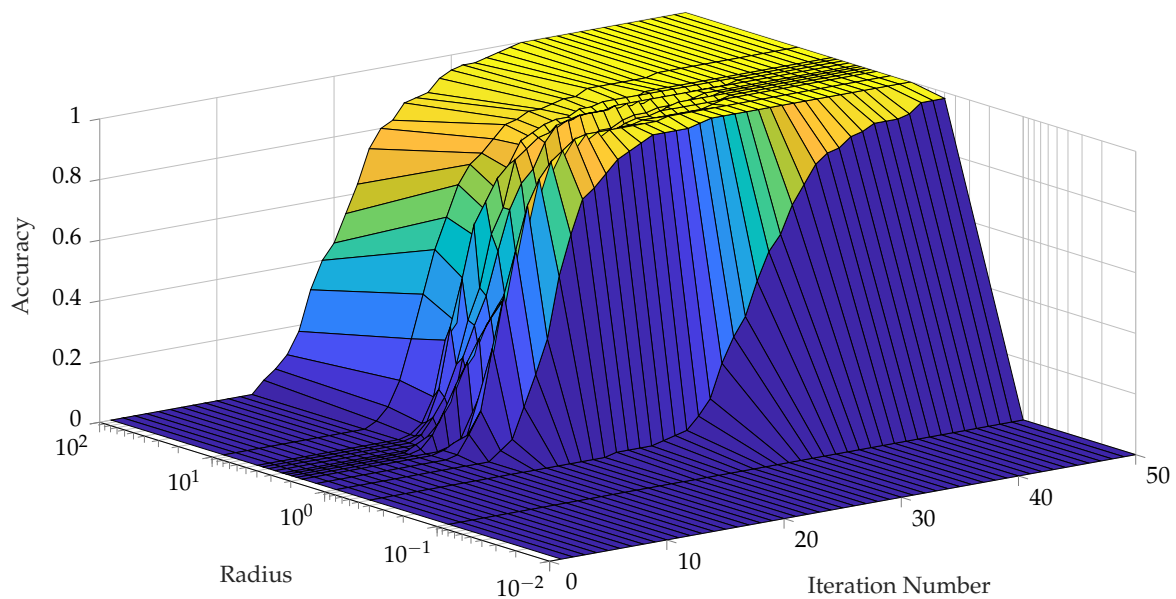
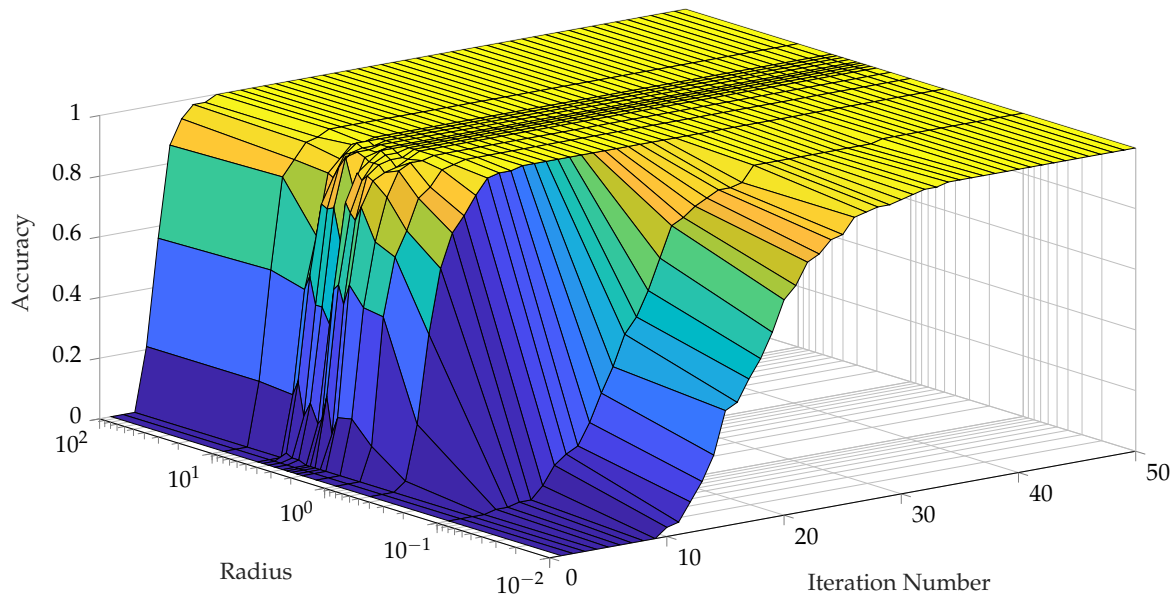


Figure 15. Probability of stopping vs projection radius vs iteration number for 2DSC-kNN algorithm. Top: 2.7dBm dataset with the number of points = 640. Bottom: 10.7dBm dataset with the number of points = 51200.

Comparison with 2DEC-kNN and 3DS-kNN K-Means Clustering

Fig. 16 portrays the gain of the 2DSC-kNN and 3DSC-kNN algorithms in the number of iterations to reach maximum accuracy for the 2.7 and 10.7dBm datasets. In these figures, a gain of 'g' means that the algorithm took 'g' fewer iterations than the classical k means acting upon the 2D dataset did to reach maximum accuracy.

Fig. 17 plots the gain of the 2DSC-kNN and 3DSC-kNN algorithms in the maximum achieved accuracy for the 2.7 and 10.7dBm datasets. Here, a gain of 'g' means that the algorithm was g% more accurate than the maximum accuracy of the classical k means acting upon the 2D dataset.

Lastly, Fig. 18 illustrates the maximum accuracies achieved by the 2DSC-kNN, 3DSC-kNN, and 2DEC-kNN algorithms for the 2.7 and 10.7dBm datasets.

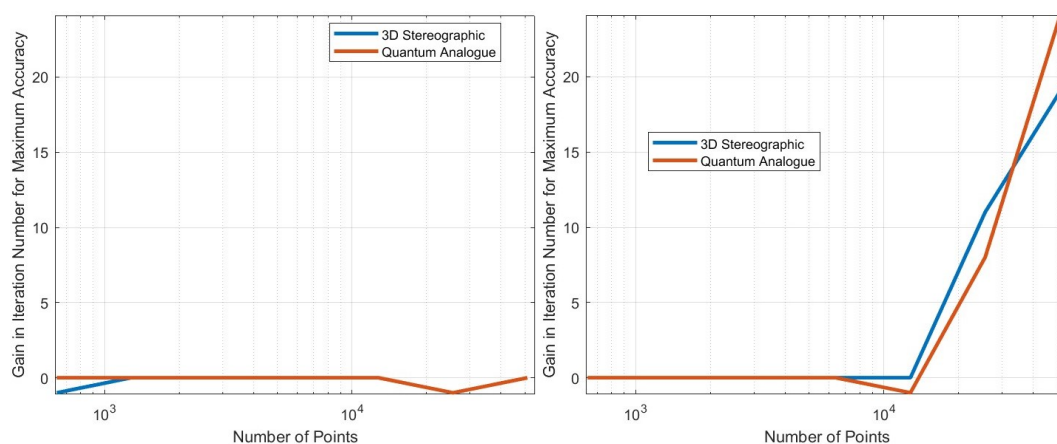


Figure 16. Gain in iteration number at maximum accuracy (number of iterations at maximum accuracy of 2DEC-kNN minus the number of iterations at maximum accuracy of 3DSC-kNN (blue) and 2DSC-kNN (red)) vs number of points for the 2.7dBm (left) and 10.7dBm (right) datasets.

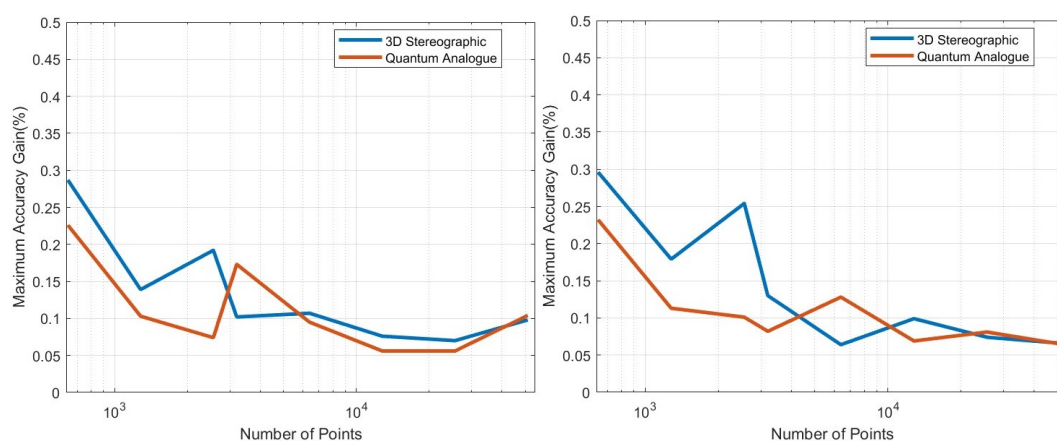


Figure 17. Gain in maximum accuracy of 2DSC-kNN (red) and 3DSC-kNN (blue) algorithms vs number of points for the 2.7dBm (left) and 10.7dBm (right) datasets

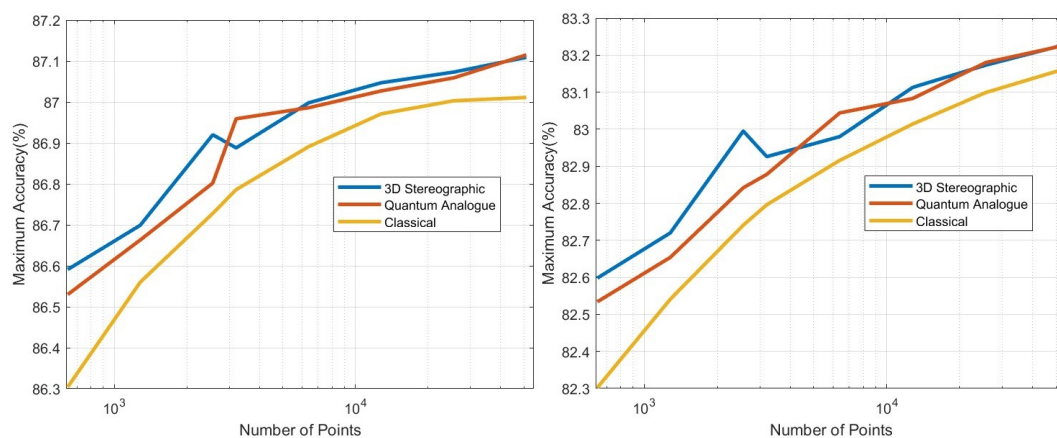


Figure 18. Maximum accuracy of 2dSC-kNN (red), 3DSC-kNN (blue) and 2DEC-KNN (yellow) algorithms vs number of points for the 2.7dBm (left) and 10.7dBm (right) datasets

5.2.2. Discussion and Analysis

Fig. 14 shows that once again, there is an ideal radius for which maximum accuracy is achieved. The ideal projection radius is larger than one, in particular, it seems to be between two and five. Most importantly, there is *an ideal number of iterations for maximum accuracy, beyond which the accuracy reduces*. As the number of points increases, the sensitivity of the accuracy to radius increases significantly. For a bad choice of radius, accuracy only falls with an increase in the number of iterations and stabilises at a very low value. For a good radius, accuracy increases to a point as iterations proceed, and then stabilises at a slightly lower value. If the allowed number of iterations is restricted, the choice of radius to achieve the best results becomes extremely important. With a good radius one can achieve nearly the maximum possible accuracy with very few iterations. As mentioned before, this holds for all dataset noises. As the dataset noise increases, the iteration number at which maximum accuracy is achieved also expectedly increases. Since accuracy always falls after a point, choosing a stopping criterion is essential rather than waiting for the algorithm to reach its natural endpoint. An idea for the stopping criterion is to record the sum of the average dissimilarity for each centroid at each iteration and stop the algorithm if that quantity increases.

Fig. 15 portrays that for a good choice of radius, the 2DSC-kNN algorithm approaches convergence much faster. For $r < 1$ the algorithm converges much slower or never converges. As the number of points increases, the convergence rate for poor radius falls dramatically. For a radius greater than the ideal radius as well, the convergence rate is lower. As one would expect, the algorithm takes longer to converge as dataset noise increases. As mentioned before, if the number of iterations is severely limited, the choice of radius becomes very important. The algorithm can reach its ideal endpoint in very few iterations if the radius is chosen well.

Through Fig. 16 we see that for lower values of noise, both algorithms do not produce much advantage in terms of iteration gain, regardless of the number of points in the data set. However, both algorithms significantly outperform the classical one at higher noise in the dataset and a high number of points. This effect is especially significant for the 2DSC-kNN algorithm. For the highest noise and all the points, it saves over 20 iterations compared to the 2DEC-kNN algorithm - *an advantage of over 50%*. One of the reasons for this is that at low noises, the algorithms already perform quite well, and it is at high noise with a high number of points that the algorithm is stressed enough to reveal the difference in performance. It should be noted that these gains are much higher than when the algorithms are allowed to reach their natural endpoint, suggesting another reason for choosing an ideal stopping criterion.

Fig. 17 shows that for all datasets and numbers of points, the two algorithms perform better than 2DEC-kNN clustering. The 3DSC-kNN algorithm and 2DSC-kNN algorithms perform nearly the same, and the accuracy gain seems to stabilise with an increase in the number of points. Fig. 18 supports these conclusions.

5.3. Overall Observations

Overall Observations from Experiment 1

1. The ideal projection radius is greater than 1 and between 2 and 5. At this ideal radius, one achieves maximum testing and training accuracy, and minimum iterations.
2. In general, the accuracy performance is the same for 3DSC-kNN and 2DSC-kNN algorithms - this shows a significant contribution of the ISP to the advantage as opposed to 'quantumness'. This is a significant distinction, not made by any previous work.
3. 2DSC-kNN and 3DSC-kNN algorithms lead to an increase in the accuracy performance in general, with the increase most pronounced for the 2.7dBm dataset.
4. The 2DSC-kNN algorithm and 3DSC-kNN algorithm give more iteration performance gain (fewer iterations required than 2DEC-kNN) for high noise datasets and for a large number of points.

5. Generally, increasing the number of points favours the 2DSC-kNN and 3DSC-kNN algorithms, with the caveat that a good radius must be carefully chosen.

Overall Observations from Experiment 2

1. These results further stress the importance of choosing a good radius (2 to 5 in this application) and a better stopping criterion. The natural endpoint is not suitable.
2. The results justify the fact that the developed 2DSC-kNN algorithm has significant advantages over 2DEC-kNN k-means clustering and 3DSC-kNN clustering.
3. The 2DSC-kNN algorithm performs nearly the same as the 3DSC-kNN algorithm in terms of accuracy, but for iterations to achieve this max accuracy, the 2DSC-kNN algorithm is better (especially for high noise and a high number of points).
4. The developed 2DSC-kNN algorithm and 3DSC-kNN algorithm are better than the 2DEC-kNN algorithm in general - in terms of accuracy and iterations to reach that maximum accuracy.
5. The supremacy of the 2DSC-kNN algorithm over the 2DEC-kNN algorithm implies that a fully-quantum SQ-kNN algorithm would have an advantage over the fully-quantum k-means algorithm of [2].

6. Conclusion and Further work

This work considers the practical case of performing kNN on experimentally acquired 64-QAM data. This work has described the problem in detail and explained how the SQ-kNN and its classical analogue, the 2DSC-kNN clustering algorithm, can be used. The proposed processes and circuits, as well as the theoretical justification for the SQ-kNN quantum algorithm and the 2DSC-kNN classical algorithm, have been described in detail. Finally, the simulation results on the real-world datasets have been presented, along with relevant analysis. From the analysis, one can see that the classical analogue of the stereographic quantum kNN, the 2DSC-kNN algorithm, is something that should be considered for industrial implementation - the experiments provide proof of concept. It also shows the importance of choosing the projection radius and provides a very useful embedding for quantum machine learning algorithms - the generalised stereographic embedding. The theoretical advantage offered by the SQ-kNN algorithm over the hybrid quantum-classical quantum k-means algorithm has also been demonstrated. Another important inference from the obtained results is that the SQ-kNN algorithm offers a way to achieve the same advantage compared to the fully quantum k-means that 2DSC-kNN has over 2DEC-kNN - by using the stereographically projected quantum states. These results warrant the practical implementation and testing of both quantum and classical algorithms.

Quantum and quantum-inspired computing has the potential to change the way certain algorithms are performed, with potentially significant advantages. However, as the field is still in relative infancy, finding where quantum and quantum-inspired computing fits in practice is a challenging problem. Here, we have seen that quantum and quantum-inspired computing can indeed be applied to signal-processing scenarios and could potentially work well in the noisy quantum era as clustering algorithms that are relatively robust to noise and inaccuracy.

6.1. Future Work

One of the most important directions of future work is to experiment with more diverse datasets. More experimentation may also lead to more sophisticated methods of selecting the radius for ISP. A more detailed analysis of how to choose a radius of projection through analytical methods is another important direction for future work. A differential geometric analysis of the effects of the ISP on a square grid gives a rough intuition of why one needs an appropriate radius. A comparison with amplitude embedding is also warranted. The *ellipsoidal projection* (Appendix D) is another promising and novel idea that is to be explored further. In this project, two different stopping criteria for the algorithm were proposed and revealed a change in its performance; yet there is plenty of room to explore more possible stopping criteria.

Further directions of study include improved overlap estimation methods [49] and communication scenarios where the dimensionality of the data points is greatly increased. For example, when multiple carriers experience identical or at least systematically correlated phase rotations.

Another future work is to benchmark against sampling-based quantum-inspired algorithms. As part of a research analysis to evaluate the best possibilities for achieving a practical speed-up, we investigated the landscape of classical algorithms inspired by the sampling in quantum algorithms. Initially, we found that such algorithms have theoretical complexity competing with quantum algorithms, however only under arguably unrealistic assumptions on the structure of the classical data. As the performance of the quantum algorithms turns out to be extremely poor, this reopens the possibility that quantum-inspired algorithms can yield performance improvements while we wait for quantum computers with sufficiently low noise. Thus future work will also be a practical implementation of the quantum-inspired kNN [13], with the goal of testing the computational advantage over 2DEC-kNN, 3DSC-kNN, and 2DSC-kNN algorithms.

Funding: This work was funded by the TUM-Huawei Joint Lab on Algorithms for Short Transmission Reach Optics (ASTRO). This project has received funding from the DFG Emmy-Noether program under grant number NO 1129/2-1 (JN) and by the Federal Ministry of Education and Research of Germany in the programme of "Souveran. Digital. Vernetzt.". Joint project 6G-life, project identification number: 16KISK002, and of the Munich Center for Quantum Science and Technology (MCQST).

Data Availability Statement: All source codes and the complete set of generated graphs are available through the publicly accessible GitHub repository at the following link: <https://github.com/AlonsoViladomat/Stereographic-quantum-embedding-clustering>. The datasets are not published and are a property of Huawei Technologies.

Acknowledgments: We would like to acknowledge fruitful discussions with Stephen DiAdamo and Fahreddin Akalin during the initial stages of the project.

Conflicts of Interest: The authors declare no conflict of interest.

Abbreviations

A list of abbreviations used in this manuscript can be found in Table 2.

Table 2. The list of abbreviations used in this manuscript.

ADC	Analog-to-digital converter
CD	Chromatic dispersion
CFO	Carrier frequency offset
CPE	Carrier phase estimation
DAC	Digital-to-analog converter
DP	Dual Polarisation
ECL	External cavity laser
FEC	Forward error correction
GHz	Gigahertz
GBd	Gigabauds
GSa/s	$\times 10^9$ samples per second
DSP	Digital signal processing
ISP	Inverse stereographic projection
MIMO	Multiple input multiple output
M-QAM	M-ary Quadrature Amplitude Modulation
QML	Quantum Machine Learning
QRAM	Quantum Random Access Memory
TR	Timing recovery
SQ access	Sample and query access
kNN	k -Nearest-Neighbour clustering algorithm(Definition 3)
FF-QRAM	Flip flop QRAM
NISQ	Near-intermediate scale quantum
\mathbb{D}	Dataspace

Table 2. Cont.

D	Dataset
D	2-dimensional dataset
\bar{c}	set of all M centroids
$C(c)$	Cluster associated to a centroid c
$d(\cdot, \cdot)$	Dissimilarity (measure function)
$d_e(\cdot, \cdot)$	Euclidean dissimilarity
$d_s(\cdot, \cdot)$	Cosine dissimilarity
s_r^{-1}	ISP into a sphere of radius r
$S^n(r)$	n -sphere of radius r
\mathcal{H}_2	Hilbert space of one qubit
nDEC-kNN	n -dimensional Euclidean Classical kNN (Definition 4)
SQ-kNN	Stereographic Quantum kNN (Definition 10)
2DSC-kNN	2D Stereographic Classical kNN (Definition 12)
3DSC-kNN	3D Stereographic Classical kNN (Definition 6)

Appendix A. QAM and Data Visualisation

Quadrature amplitude modulation (QAM) conveys multiple digital bits with each transmission by mixing amplitude and phase variations in a carrier frequency. This is done by changing (modulating) the amplitudes of two carrier waves. The two carrier waves (of the same frequency) are out of phase with each other by 90° ; namely, they are the sine and cosine waves of a given frequency. This condition is known as orthogonality, and the two carrier waves as quadrature. The transmitted signal is created by adding together the two carrier waves (the sine and cosine components). The two waves can be coherently separated (demodulated) at the receiver because of their orthogonality. QAM is used extensively as a modulation scheme for digital telecommunication systems, such as in 802.11 Wi-Fi standards. Arbitrarily high spectral efficiencies can be achieved with QAM by setting a suitable constellation size, limited only by the noise level and linearity of the communications channel [51]. QAM allows us to transmit multiple bits for each time interval of the carrier symbol. The term ‘‘symbol’’ means some unique combination of phase and amplitude [52].

In this work, each transmitted signal corresponds to a complex number $s \in \mathbb{C}$:

$$s = |s|e^{i\phi}, \quad (\text{A1})$$

where $|s|^2$ is the initial transmission power and ϕ is the phase of s . The case shown in Eq. (A1) is ideal; however, in real-world systems, noise affects the transmitted signal, distorting and scattering it in the amplitude and phase space. For our case, the received and partially processed noisy signal can be modelled as follows:

$$s = |s|e^{i\phi} + N, \quad (\text{A2})$$

where $N \in \mathbb{C}$ is a random noise affecting the overall value of ideal amplitude and phase. This model motivates the use of nearest neighbour clustering for cases when the noise N causes the received signal to be scattered in the vicinity of the ideal signal s .

Appendix A.1. Description of 64-QAM Data

The various datasets we collected through the setup described in Section 2.1 are visualised in this section. As mentioned, there are four datasets with launch powers of 2.7, 6.6, 8.6, 10.7 dBm, corresponding to various noise levels. Each data set consists of three variables:

- ‘alphabet’: The initial analog values at which the data was transmitted, in the form of complex numbers, i.e. for an entry $(a + ib)$, the transmitted signal was of the form $a \sin(\theta) + b \cos(\theta)$. Since the transmission protocol is 64-QAM, there are 64 values in this variable. The transmission alphabet is the same irrespective of the non-linear distortions.

- 'rxsignal': The received analog values of the signal by the receiver. This data is in the form of a 52124×5 matrix. Each datapoint was transmitted five times to the receiver, and so each row contains the values detected by the receiver during the different instances of the transmission of the same datapoint. The values in different rows represent unique datapoint values detected by the receiver.
- 'bits': This is the true label for the transmitted points. This data is in the form of a 52124×6 matrix. Since the protocol is 64-QAM, each analog point represents six bits. These six bits are the entries in each column, and each value in a different row represents the correct label for a unique transmitted datapoint value. The first three bits encode the column, and the last three bits encode the row - see Fig. A3.

In Appendix A.1, we can see the analog transmission values (alphabet) for all channels. In the subsequent four figures (Fig. A2 to Fig. A2), we can observe the transmission data for all the iterations for each channel. The first transmission data is represented as blue crosses, the second transmission as orange circles, the third transmission as yellow dots, the fourth transmission as purple stars, and the fifth transmission as green pluses. The de-mapping alphabet is depicted in Fig. A3.

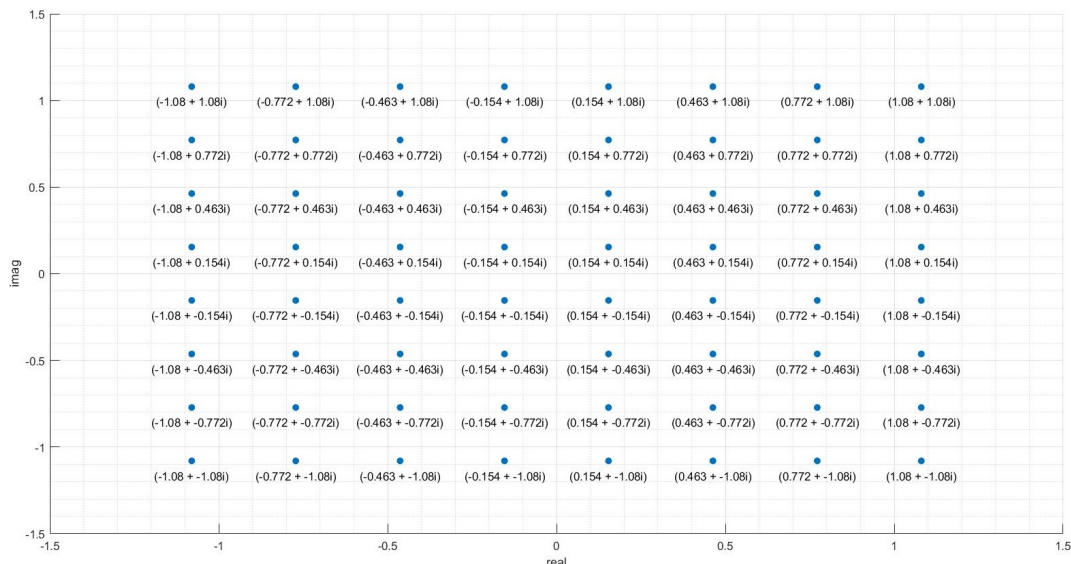


Figure A1. The analog alphabet (initial analog transmission values) for the data transmitted in all the channels. The real part represents the amplitude of the transmitted sine wave, and the imaginary part represents the amplitude of the cosine wave.

One can see from these figures that as the noise in the channel increases, the points are further scattered away from the initial alphabet. In addition, the non-linear noise effects also increase, causing distortion of the 'shape' of the data, most clearly visible in Fig. A2 - especially near the 'corners'. The birefringence phase noise also increases with an increase in the channel noise, causing all the points to be 'rotated' about the origin.

Once the centroids have been found and the data has been clustered, as mentioned before, we need to 'de-map' the analog centroid values and clusters to bit-strings. For this, we need a de-mapping alphabet which maps the analog values of the alphabet to the corresponding bit strings. The de-mapping alphabet is depicted in Fig. A3. It can be seen from the figure that, as in most cases, the points are Gray coded, i.e. adjacent points differ in binary translation by only 1 bit. This helps minimise the number of bit errors per symbol error in case of misclassification or exceptionally high noise. In case a point is misclassified, with most probability, it will be assigned to a neighbouring cluster. Since the surrounding clusters differ by only 1 bit, it minimises the bit error rate. Due to Gray coding, the bit error rate is approximately $\frac{1}{6}$ of the symbol error rate.

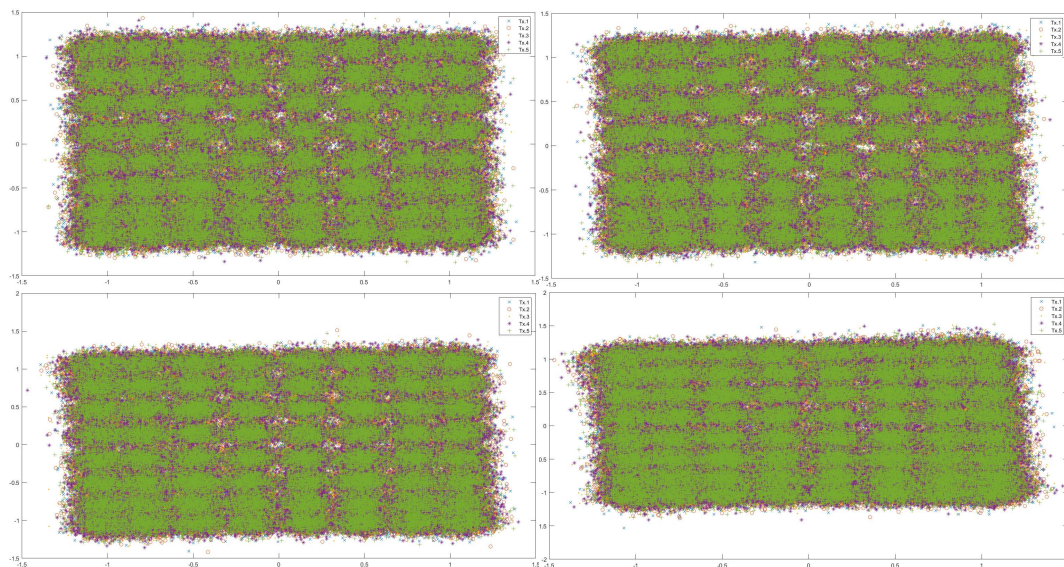


Figure A2. The four datasets detected by the receiver with various launch powers corresponding to different noise levels: 2.7dBm (top left), 6.6dBm (top right), 8.6dBm (bottom left), 10.7 dBm (bottom right). All five iterations of transmission are depicted together.

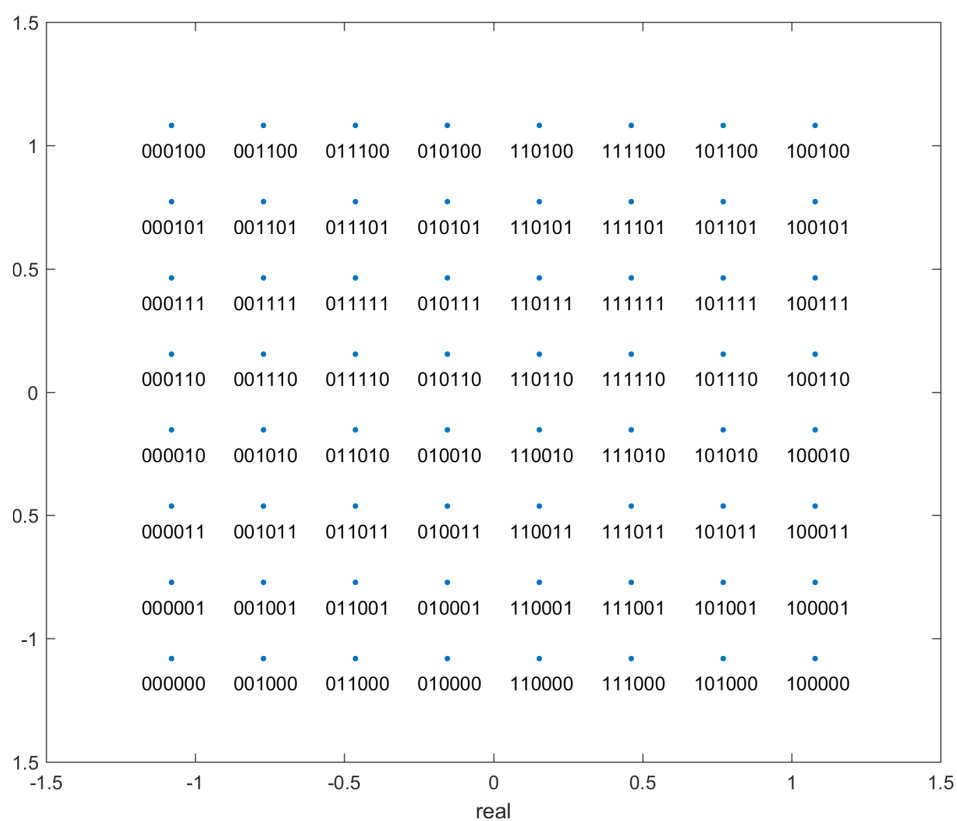


Figure A3. The bit-string mapping and demapping alphabet.

Appendix B. Data Embedding

One needs data in the form of quantum states for processing in a Quantum Computer. However, due to the instability of current qubits, data can only be stored for an extended period in classical form.

Hence, the need arises to convert classical data into a quantum form. NISQ devices have a minimal number of logical qubits, which are, in addition, stable for only a limited time before they lose their quantum information to decoherence. This makes the first step in Quantum Machine Learning to load classical data by encoding it into qubits. This process is called data encoding or embedding. Classical data encoding for quantum computation plays a critical role in the overall design and performance of quantum machine learning algorithms. Table A1 summarises the various forms of data embedding.

Table A1. Summary of embeddings [2,22,53].

Embedding	Encoding	Num. qubits required	Gate Depth
Basis	$x_i \approx \sum_{i=-k}^m b_i 2^i \mapsto b_m \dots b_{-k}\rangle$	$l = k + m$ per data point	$O(\log_2 n)$
Angle	$x_i \mapsto \cos(x_i) 0\rangle + \sin(x_i) 1\rangle$	$O(n)$	$O(1)$
Amplitude	$X \mapsto \sum_{i=0}^{n-1} x_i i\rangle$	$\lceil \log_2 n \rceil$	$O(2^n)$ gates
QRAM	$X \mapsto \sum_{i=0}^{n-1} \frac{1}{\sqrt{n}} i\rangle x_i\rangle$	$\lceil \log_2 n \rceil + l$	$O(\log_2 n)$ queries

Appendix B.1. Angle Embedding

Angle encoding [48,54,55] is one of the most fundamental forms of encoding classical data into a quantum state. Each data point is represented as a separate qubit. The n^{th} classical real number is encoded into the rotation angle of the n^{th} qubit. In its most basic form, this encoding requires N qubits to represent N dimensional data. It is quite cheap to prepare in terms of complexity – all that is needed is one rotation quantum gate for each qubit. This is one of the forms of encoding we have used to implement quantum kNN clustering. It is generally useful for quantum neural networks and other such QML algorithms. Angle encoding encodes N features into the rotation angles of n qubits where $N \leq n$.

The rotations can be chosen as either $R_X(\theta)$, $R_Y(\theta)$ or $R_Z(\theta)$ gates. As a first step, each input data point is normalised to the interval $[0, \pi]$. To encode the data points, a rotation around the y-axis is used. The angle of rotation depends on the value of the normalised data point. This creates the following separable state:

$$|\psi\rangle = R_Y(x_0) |0\rangle \otimes R_Y(x_1) |0\rangle \otimes \dots \otimes R_Y(x_n) |0\rangle \quad (\text{A3})$$

$$= \begin{pmatrix} \cos x_0 \\ \sin x_0 \end{pmatrix} \otimes \begin{pmatrix} \cos x_1 \\ \sin x_1 \end{pmatrix} \otimes \dots \otimes \begin{pmatrix} \cos x_n \\ \sin x_n \end{pmatrix} \quad (\text{A4})$$

It can easily be seen that one qubit is needed per data point, which is not optimal. To load the data, the rotations on the qubits can be performed in parallel; thus, *the depth of the circuit is optimal* [48].

The main advantage of this encoding is that it is very efficient in terms of operations – only a constant number of parallel operations are needed regardless of how many data values need to be encoded. This is not optimal from a qubit point of view (the circuit is very wide), as every input vector component requires one qubit. Another related encoding, dense angle encoding, exploits an additional property of qubits (relative phase) to use only $n/2$ qubits to encode n data points. QRAM can be used to generate the more compact quantum state $\sum |i\rangle R_Y(\theta_i) |0\rangle$

Appendix C. Stereographic Projection

Appendix C.1. ISP for General Radius

In this appendix, the transformations for obtaining the Cartesian coordinates of the projected point on a sphere of general radius are derived, followed by the derivation of polar and azimuthal angles of the point on the sphere. First mentioned are three conditions that the point on the sphere must satisfy, and then follows the rest of the derivation. Refer to Fig. A4 for a better understanding of the conditions and calculations.

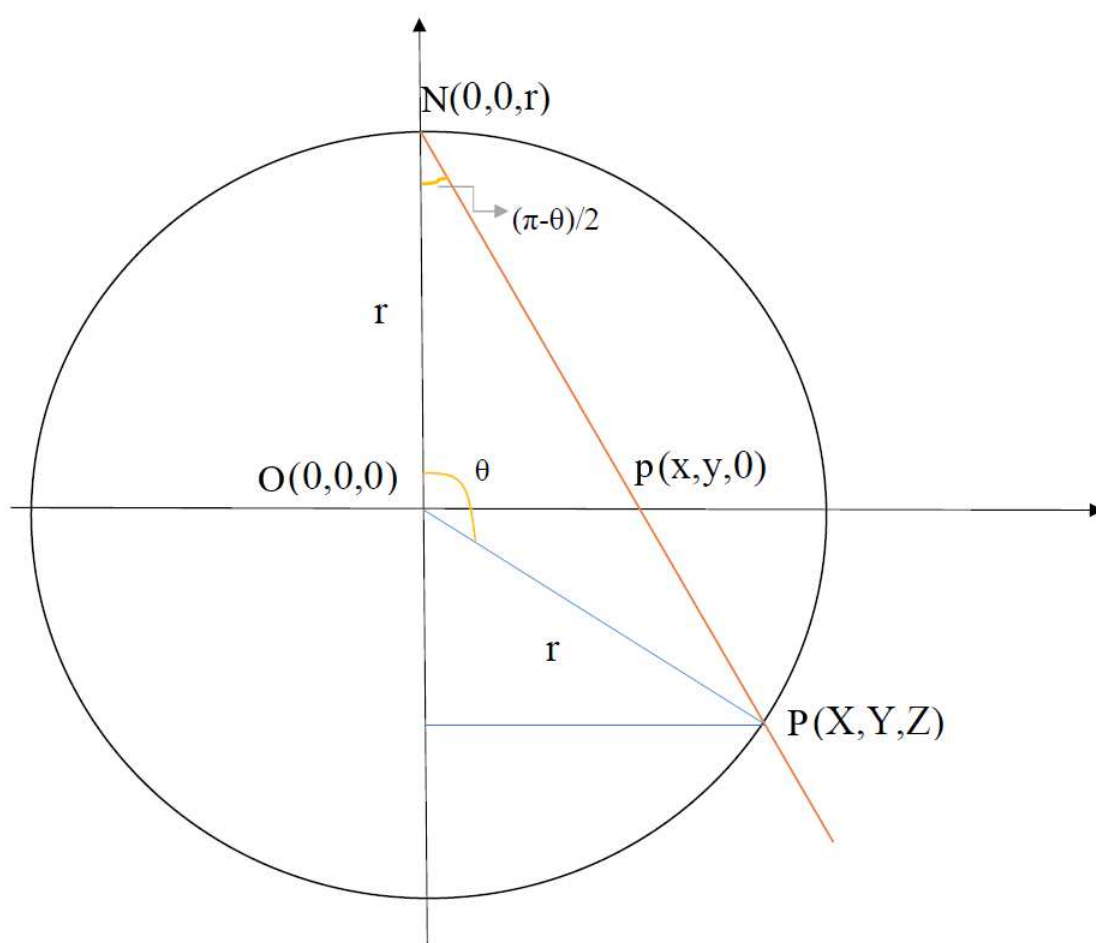


Figure A4. ISP for a sphere of radius 'r'. In this figure, the plane is the plane of angle ϕ with respect to the X-axis perpendicular to the XY plane, i.e. the plane containing the 2D point and its projection.

1. Azimuthal angle of the original point and the projected point must be the same, i.e. the original point, projected point, and the top of the sphere (the point from which all projections are drawn) lie on the same plane, which is perpendicular to the 2D plane.

$$\implies \frac{s_y}{s_x} = \frac{p_y}{p_x} \quad (\text{A5})$$

2. The projected point lies on the sphere.

$$\implies s_x^2 + s_y^2 + s_z^2 = r^2 \quad (\text{A6})$$

3. The triangle with vertices $(0,0,r)$, $(0,0,0)$ and $(p_x, p_y, 0)$ is similar to the triangle with vertices $(0,0,r)$, $(0,0,s_z)$ and (s_x, s_y, s_z) :

$$\implies \frac{\sqrt{p_x^2 + p_y^2}}{r} = \frac{\sqrt{s_x^2 + s_y^2}}{r - s_z}. \quad (\text{A7})$$

Using Eq. (A7) and Eq. (A5), we get

$$s_x = p_x \cdot \left(1 - \frac{s_z}{r}\right) \quad \text{and} \quad s_y = p_y \cdot \left(1 - \frac{s_z}{r}\right)$$

Substituting in Eq. (A6) we get,

$$s_z = r \left(\frac{p_x^2 + p_y^2 - r^2}{p_x^2 + p_y^2 + r^2} \right)$$

Hence one gets the set of transformations:

$$s_x = p_x \left(\frac{2r^2}{p_x^2 + p_y^2 + r^2} \right) \quad (\text{A8})$$

$$s_y = p_y \left(\frac{2r^2}{p_x^2 + p_y^2 + r^2} \right) \quad (\text{A9})$$

$$s_z = r \left(\frac{p_x^2 + p_y^2 - r^2}{p_x^2 + p_y^2 + r^2} \right) \quad (\text{A10})$$

Calculating the polar angle (θ) and azimuthal angle (ϕ):

$$\phi = \tan^{-1} \left(\frac{s_y}{s_x} \right) \quad (\text{A11})$$

$$\implies \phi = \tan^{-1} \left(\frac{p_y}{p_x} \right) \quad (\text{A12})$$

$$\tan \left(\frac{\pi - \theta}{2} \right) = \frac{\sqrt{p_x^2 + p_y^2}}{r} \quad (\text{A13})$$

$$\implies \theta = 2 \cdot \tan^{-1} \left(\frac{r}{\sqrt{p_x^2 + p_y^2}} \right) \quad (\text{A14})$$

Appendix C.2. Equivalence of Displacement and Scaling

Refer to Fig. A5. Here, \mathbf{T} is the point from which all the projections originate; \mathbf{O} and \mathbf{O}' are the centres of the projection spheres of radius r and $(1+\delta)r$ respectively; \mathbf{P} is the point on the 2D plane to be projected; and \mathbf{S} and \mathbf{S}' are the ISP of \mathbf{P} on the spheres of radius r and centre \mathbf{O} and radius r' and centre \mathbf{O}' respectively.

$$|\overline{\mathbf{OT}}| = |\overline{\mathbf{OS}}| = r \quad (\text{A15})$$

$$\implies \angle \mathbf{OTS} = \angle \mathbf{OST} = \theta \quad (\text{A16})$$

$$\implies \angle \mathbf{TOS} = \pi - 2\theta \quad (\text{A17})$$

Also,

$$|\overline{\mathbf{O'T}}| = |\overline{\mathbf{O'S'}}| = (1+\delta)r \quad (\text{A18})$$

$$\implies \angle \mathbf{O'TS'} = \angle \mathbf{O'S'T} = \theta \quad (\text{A19})$$

$$\implies \angle \mathbf{TO'S'} = \pi - 2\theta \quad (\text{A20})$$

Hence

$$\angle \mathbf{TOS} = \angle \mathbf{TO'S'} = \pi - 2\theta$$

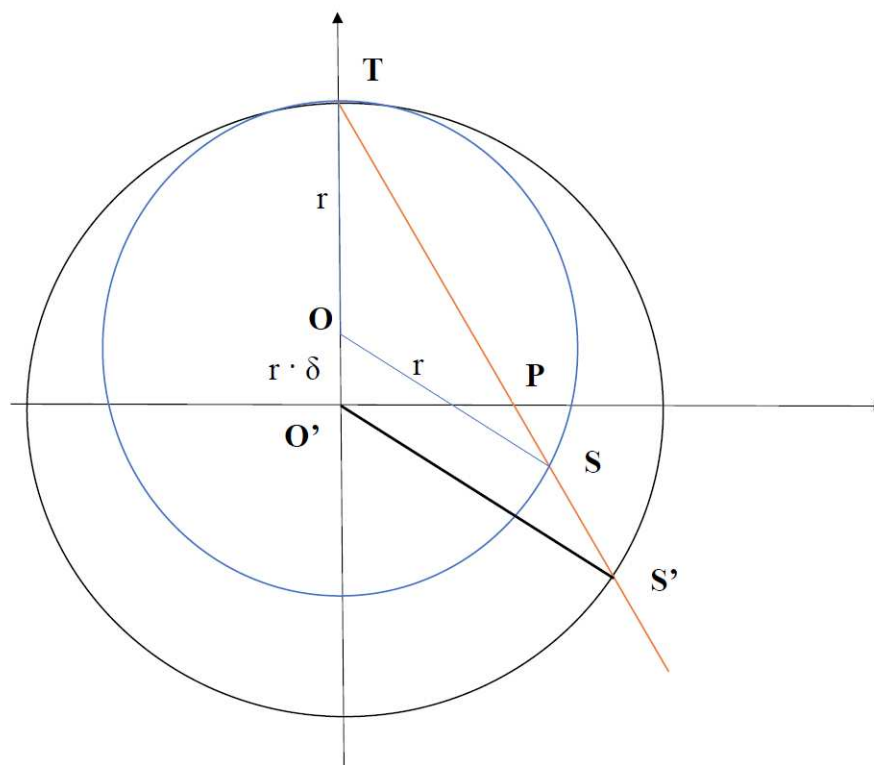


Figure A5. ISP on a sphere displaced above the plane and a sphere centred at origin.

Since both S and S' lie on the same plane, which is a vertical cross-section of the sphere (plane perpendicular to the data plane and passing through the centre of both stereographic spheres), the azimuthal angle of both points is equal ($\phi = \tan^{-1} \left(\frac{p_y}{p_x} \right)$).

Hence one can see that the azimuthal and the polar angle generated by ISP on a sphere of radius r displaced above the 2D plane containing the points by $(1+\delta)r$ is the same as the azimuthal and the polar angle generated by ISP on a sphere of radius $(1+\delta)r$ centred at origin. This reduces the effective number of parameters that can be chosen for the embedding.

Appendix D. Ellipsoidal Embedding

Here, we first derive the transformations for obtaining the Cartesian coordinates of the projected point on a general ellipsoid, followed by the derivation of polar and azimuthal angles for the point on the ellipsoid. First mentioned are three conditions that the point on the sphere must satisfy, and then follows the rest of the derivation. Refer to Fig. A6 for a better understanding of the conditions and calculations.

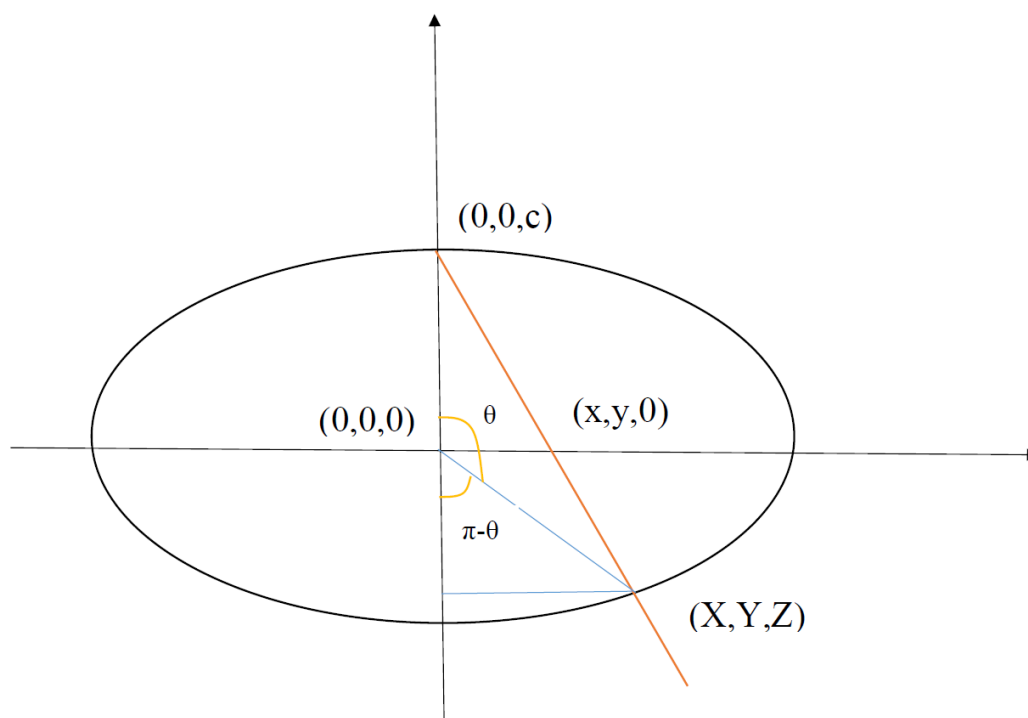


Figure A6. Ellipsoidal Projection: a generalisation of the ISP.

1. Azimuthal angle -

$$\implies \frac{s_y}{s_x} = \frac{p_y}{p_x} \quad (\text{A21})$$

2. The projected point lies on the ellipsoid -

$$\implies \frac{s_x^2}{a^2} + \frac{s_y^2}{b^2} + \frac{s_z^2}{c^2} = 1 \quad (\text{A22})$$

3. The triangle with vertices $(0,0,c)$, $(0,0,0)$ and $(p_x, p_y, 0)$ is similar to the triangle with vertices $(0,0,c)$, $(0,0,s_z)$ and (s_x, s_y, s_z) -

$$\implies \frac{\sqrt{p_x^2 + p_y^2}}{c} = \frac{\sqrt{s_x^2 + s_y^2}}{c - s_z}. \quad (\text{A23})$$

From the above conditions, we have

$$s_x = p_x \cdot \left(1 - \frac{s_z}{c}\right) \quad \text{and} \quad s_y = p_y \cdot \left(1 - \frac{s_z}{c}\right) \quad (\text{A24})$$

Substituting as before, we get

$$s_z = c \cdot \left(\frac{\frac{p_x^2}{a^2} + \frac{p_y^2}{b^2} - 1}{\frac{p_x^2}{a^2} + \frac{p_y^2}{b^2} + 1} \right) \quad (\text{A25})$$

Hence one gets the set of transformations:

$$s_x = p_x \left(\frac{2}{\frac{p_x^2}{a^2} + \frac{p_y^2}{b^2} + 1} \right) \quad (\text{A26})$$

$$s_y = p_y \left(\frac{2}{\frac{p_x^2}{a^2} + \frac{p_y^2}{b^2} + 1} \right) \quad (\text{A27})$$

$$s_z = c \left(\frac{\frac{p_x^2}{a^2} + \frac{p_y^2}{b^2} - 1}{\frac{p_x^2}{a^2} + \frac{p_y^2}{b^2} + 1} \right) \quad (\text{A28})$$

From Fig. A6 one can see that

$$\begin{aligned} \tan(\pi - \theta) &= \frac{\sqrt{s_x^2 + s_y^2}}{-s_z} \\ \therefore \theta &= \tan^{-1} \left(\frac{2\sqrt{p_x^2 + p_y^2}}{c \left(\frac{p_x^2}{a^2} + \frac{p_y^2}{b^2} - 1 \right)} \right) \end{aligned}$$

Also, as before, by the same reasoning

$$\phi = \tan^{-1} \left(\frac{p_y}{p_x} \right)$$

Now that we have these expressions, we have two methods of encoding the datapoint. We can either encode it as before, using the unitary $U(\theta, \phi)$, which would correspond to projecting all the points on the ellipsoid to the surface of the sphere radially; or we could use mixed states to represent the points on the surface of the ellipsoid after rescaling it to lie within the Bloch sphere.

Appendix E. Distance Estimation Using Stereographic Embedding

In our proposed quantum algorithm (the SQ-kNN algorithm), we project the original 2-dimensional points into the stereographic sphere before converting them into quantum states using angle embedding and then estimating the overlap with the Bell state measurement circuit. Due to the many steps of this procedure, it is insightful to calculate the final output in terms of the original input, the 2-dimensional datapoints. It also serves as a helpful point of comparison with the 2DEC-kNN algorithm, where the Euclidean dissimilarity between the 2-dimensional points is used for classification.

Recall Eq. (50) from Section 2.7. Then concatenating the ISP with the cosine dissimilarity, we get a new dissimilarity as follows.

As mentioned before, we begin with two 2D points $\mathbf{p}_1 = \begin{pmatrix} x_1 \\ y_1 \end{pmatrix}$, $\mathbf{p}_2 = \begin{pmatrix} x_2 \\ y_2 \end{pmatrix}$ and compute

$$d_s \circ s_r^{-1}(\mathbf{p}_1, \mathbf{p}_2) := d_s \left(s_r^{-1}(\mathbf{p}_1), s_r^{-1}(\mathbf{p}_2) \right). \quad (\text{A29})$$

To calculate this, we compute $s_r^{-1}(\mathbf{p}_1) \cdot s_r^{-1}(\mathbf{p}_2)$ using Eq. (50):

$$s_r^{-1}(\mathbf{p}_1) \cdot s_r^{-1}(\mathbf{p}_2) = \frac{4r^4 \mathbf{p}_1 \cdot \mathbf{p}_2}{(\|\mathbf{p}_1\|^2 + r^2)(\|\mathbf{p}_2\|^2 + r^2)} + r^2 \frac{(\|\mathbf{p}_1\|^2 - r^2)(\|\mathbf{p}_2\|^2 - r^2)}{(\|\mathbf{p}_1\|^2 + r^2)(\|\mathbf{p}_2\|^2 + r^2)}$$

Combining, we have:

$$d_s \circ s_r^{-1}(\mathbf{p}_1, \mathbf{p}_2) = 1 - \frac{1}{r^2} s_r^{-1}(\mathbf{p}_1) \cdot s^{-1}(\mathbf{p}_2) \quad (\text{A30})$$

$$= \frac{(\|\mathbf{p}_1\|^2 + r^2)(\|\mathbf{p}_2\|^2 + r^2) - 4r^2 \mathbf{p}_1 \cdot \mathbf{p}_2}{(\|\mathbf{p}_1\|^2 + r^2)(\|\mathbf{p}_2\|^2 + r^2)} - \frac{(\|\mathbf{p}_1\|^2 - r^2)(\|\mathbf{p}_2\|^2 - r^2)}{(\|\mathbf{p}_1\|^2 + r^2)(\|\mathbf{p}_2\|^2 + r^2)} \quad (\text{A31})$$

$$= \frac{2r^2 \|\mathbf{p}_1\|^2 + 2r^2 \|\mathbf{p}_2\|^2 - 4r^2 \mathbf{p}_1 \cdot \mathbf{p}_2}{(\|\mathbf{p}_1\|^2 + r^2)(\|\mathbf{p}_2\|^2 + r^2)} \quad (\text{A32})$$

$$= \frac{2r^2 \|\mathbf{p}_1 - \mathbf{p}_2\|^2}{(\|\mathbf{p}_1\|^2 + r^2)(\|\mathbf{p}_2\|^2 + r^2)} \quad (\text{A33})$$

$$= d_e(\mathbf{p}_1, \mathbf{p}_2) \cdot \left(\frac{2r^2}{(r^2 + \|\mathbf{p}_1\|^2)(r^2 + \|\mathbf{p}_2\|^2)} \right) \quad (\text{A34})$$

where d_e is the Euclidean dissimilarity from Eq. (25).

It is illustrative to pick the point (0,0) (origin) and see how this function varies as the other point \mathbf{p} varies. In this case, we have:

$$\frac{1}{2} d_s \circ s_r^{-1}(0, \mathbf{p}) = \frac{r^2 \|\mathbf{p}\|^2}{(r^2 + \|\mathbf{p}\|^2)(r^2)} = \frac{\|\mathbf{p}\|^2}{r^2 + \|\mathbf{p}\|^2} = 1 - \frac{r^2}{r^2 + \|\mathbf{p}\|^2} \quad (\text{A35})$$

For Fig. A7, the radius of the stereographic sphere is assumed to be one. Hence the quantum dissimilarity reduces to:

$$\frac{1}{2} d_s \circ s_r^{-1}(0, \mathbf{p}) = 1 - \frac{1}{1 + \|\mathbf{p}\|^2} \quad (\text{A36})$$

For Fig. A7, the radius of the stereographic sphere is assumed to be 2 and 0.5, respectively.

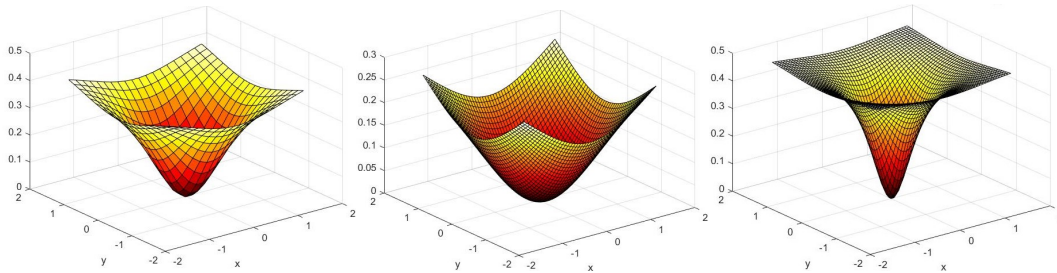


Figure A7. $r = 1$ (left) $r = 2$ (middle) $r = 0.5$ (right).

Appendix F. Rotation Gates and the UGate

The complete expression for the unitary UGate in Qiskit is as follows:

$$U(\theta, \phi, \lambda) := \begin{pmatrix} \cos \frac{\theta}{2} & -e^{i\lambda} \sin \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} & e^{i(\phi+\lambda)} \cos \frac{\theta}{2} \end{pmatrix}$$

Using the $|0\rangle$ state for encoding, we have

$$\therefore U(\theta, \phi, \lambda) |0\rangle = \begin{pmatrix} \cos \frac{\theta}{2} & -e^{i\lambda} \sin \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} & e^{i(\phi+\lambda)} \cos \frac{\theta}{2} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} = \begin{pmatrix} \cos \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} \end{pmatrix}$$

One can see that this state is not dependent on λ . On the other hand, if we use the state $|1\rangle$ for encoding, we have

$$U(\theta, \phi, \lambda) |1\rangle = \begin{pmatrix} \cos \frac{\theta}{2} & -e^{i\lambda} \sin \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} & e^{i(\phi+\lambda)} \cos \frac{\theta}{2} \end{pmatrix} \begin{pmatrix} 0 \\ 1 \end{pmatrix} = \begin{pmatrix} -e^{i\lambda} \sin \frac{\theta}{2} \\ e^{i(\phi+\lambda)} \cos \frac{\theta}{2} \end{pmatrix} = e^{i\lambda} \begin{pmatrix} -\sin \frac{\theta}{2} \\ e^{i\phi} \cos \frac{\theta}{2} \end{pmatrix}$$

One can see that the λ term leads only to a global phase. A global phase will not affect the observable outcome of the SWAP test or Bell-state measurement (due to the modulus operator) - hence once again, no information can be encoded into the quantum state using λ .

For constructing the point (θ, ϕ) on the Bloch sphere, we can use rotation gates as well:

$$(\theta, \phi) := \mathbb{R}_Z(\phi)\mathbb{R}_Y(\theta) |0\rangle \quad (\text{A37})$$

$$= \begin{pmatrix} e^{-i\frac{\phi}{2}} & 0 \\ 0 & e^{i\frac{\phi}{2}} \end{pmatrix} \begin{pmatrix} \cos \frac{\theta}{2} & -\sin \frac{\theta}{2} \\ \sin \frac{\theta}{2} & \cos \frac{\theta}{2} \end{pmatrix} \begin{pmatrix} 1 \\ 0 \end{pmatrix} \quad (\text{A38})$$

$$= \begin{pmatrix} e^{-i\frac{\phi}{2}} & 0 \\ 0 & e^{i\frac{\phi}{2}} \end{pmatrix} \begin{pmatrix} \cos \frac{\theta}{2} \\ \sin \frac{\theta}{2} \end{pmatrix} \quad (\text{A39})$$

$$= \begin{pmatrix} e^{-i\frac{\phi}{2}} \cos \frac{\theta}{2} \\ e^{i\frac{\phi}{2}} \sin \frac{\theta}{2} \end{pmatrix} = e^{-i\frac{\phi}{2}} \begin{pmatrix} \cos \frac{\theta}{2} \\ e^{i\phi} \sin \frac{\theta}{2} \end{pmatrix} \quad (\text{A40})$$

$$= e^{-i\frac{\phi}{2}} U(\theta, \phi) |0\rangle \quad (\text{A41})$$

From Eq. (A41) one can see that $\mathbb{R}_Z(\phi)\mathbb{R}_Y(\theta) |0\rangle$ and $U(\theta, \phi) |0\rangle$ only differ by a global phase ($e^{-i\frac{\phi}{2}}$). Hence, the $\mathbb{R}_Z(\phi)\mathbb{R}_Y(\theta)$ and $U(\theta, \phi)$ operations can be used interchangeably for state preparation since a global phase will not affect the observable result of the SWAP test.

References

1. Harrow, A.W.; Hassidim, A.; Lloyd, S. Quantum Algorithm for Linear Systems of Equations. *Physical Review Letters* **2009**, *103*. <https://doi.org/10.1103/physrevlett.103.150502>.
2. Lloyd, S.; Mohseni, M.; Rebentrost, P. Quantum algorithms for supervised and unsupervised machine learning. *arXiv:1307.0411* **2013**.
3. Preskill, J. Quantum Computing in the NISQ era and beyond. *Quantum* **2018**, *2*, 79. <https://doi.org/10.22331/q-2018-08-06-79>.
4. Tang, E. Quantum Principal Component Analysis Only Achieves an Exponential Speedup Because of Its State Preparation Assumptions. *Physical Review Letters* **2021**, *127*. <https://doi.org/10.1103/physrevlett.127.060503>.
5. Arute, F.; Arya, K.; Babbush, R.; Bacon, D.; Bardin, J.; Barends, R.; Biswas, R.; Boixo, S.; Brandao, F.; Buell, D.; et al. Quantum Supremacy using a Programmable Superconducting Processor. *Nature* **2019**, *574*, 505–510.
6. Schuld, M.; Petruccione, F. *Supervised Learning with Quantum Computers*; Quantum Science and Technology, Springer International Publishing, 2018.
7. M. Schuld, I. Sinayskiy, F.P. An introduction to quantum machine learning. *arXiv:1409.3097 [quant-ph]* **2014**.
8. Kerenidis, I.; Prakash, A. Quantum Recommendation Systems. In Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS 2017); Papadimitriou, C.H., Ed.; Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik: Dagstuhl, Germany, 2017; Vol. 67, *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 49:1–49:21. <https://doi.org/10.4230/LIPIcs.ITCS.2017.49>.
9. Kerenidis, I.; Landman, J.; Luongo, A.; Prakash, A. q-means: A quantum algorithm for unsupervised machine learning. *arXiv:1812.03584* **2018**.
10. Modi, A.; Jasso, A.V.; Ferrara, R.; Deppe, C.; Noetzel, J.; Fung, F.; Schaedler, M. Testing of Hybrid Quantum-Classical K-Means for Nonlinear Noise Mitigation. *arXiv preprint arXiv:2308.03540* **2023**, [[arXiv:quant-ph/2308.03540](https://arxiv.org/abs/2308.03540)].
11. Pakala, L.; Schmauss, B. Non-linear mitigation using carrier phase estimation and k-means clustering. In Proceedings of the Photonic Networks; 16. ITG Symposium. VDE, 2015, pp. 1–5.

12. Zhang, J.; Chen, W.; Gao, M.; Shen, G. K-means-clustering-based fiber nonlinearity equalization techniques for 64-QAM coherent optical communication system. *Optics express* **2017**, *25*, 27570–27580.
13. Tang, E. Quantum Principal Component Analysis Only Achieves an Exponential Speedup Because of Its State Preparation Assumptions. *Phys. Rev. Lett.* **2021**, *127*, 060503. <https://doi.org/10.1103/PhysRevLett.127.060503>.
14. Gambetta, J. IBM's roadmap for scaling quantum technology.
15. Diedolo, F.; Böcherer, G.; Schädler, M.; Calabró, S. Nonlinear Equalization for Optical Communications Based on Entropy-Regularized Mean Square Error. In Proceedings of the European Conference on Optical Communication (ECOC) 2022. Optica Publishing Group, 2022, p. We2C.2.
16. Martyn, J.M.; Rossi, Z.M.; Tan, A.K.; Chuang, I.L. A grand unification of quantum algorithms. *arXiv:2105.02859* **2021**.
17. Kopczyk, D. Quantum machine learning for data scientists. *arXiv:1804.10068* **2018**.
18. Esma Aimeur, G.B.; Gambs, S. Quantum clustering algorithms. *ICML '07: Proceedings of the 24th international conference on Machine learning June 2007* **2007**, p. 1–8.
19. Cruise, J.R.; Gillespie, N.I.; Reid, B. Practical Quantum Computing: The value of local computation. *arXiv:2009.08513* **2020**.
20. Johri, S.; Debnath, S.; Mocherla, A.; Singh, A.; Prakash, A.; Kim, J.; Kerenidis, I. Nearest centroid classification on a trapped ion quantum computer. *npj Quantum Information* **2021**, *7*, 122. <https://doi.org/10.1038/s41534-021-00456-5>.
21. Khan, S.U.; Awan, A.J.; Vall-Llosera, G. K-Means Clustering on Noisy Intermediate Scale Quantum Computers. *arXiv:1909.12183* **2019**.
22. Cortese, J.A.; Braje, T.M. Loading classical data into a quantum computer. *arXiv:1803.01958* **2018**.
23. Giovannetti, V.; Lloyd, S.; Maccone, L. Quantum Random Access Memory. *Physical Review Letters* **2008**, *100*. <https://doi.org/10.1103/physrevlett.100.160501>.
24. Harry Buhrman, Richard Cleve, J.W.; de Wolf, R. Quantum fingerprinting. *arXiv:quant-ph/0102001* **2001**.
25. Ripper, P.; Amaral, G.; Temporão, G. Swap Test-based characterization of decoherence in universal quantum computers. *Quantum Information Processing* **2023**, *22*, 1–14.
26. Foulds, S.; Kendon, V.; Spiller, T. The controlled SWAP test for determining quantum entanglement. *Quantum Science and Technology* **2021**, *6*, 035002.
27. Tang, E. A Quantum-Inspired Classical Algorithm for Recommendation Systems. In Proceedings of the Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing; Association for Computing Machinery: New York, NY, USA, 2019; STOC 2019, p. 217–228, [[arXiv:cs.IR/1807.04271](https://arxiv.org/abs/cs/1807.04271)]. <https://doi.org/10.1145/3313276.3316310>.
28. Chia, N.H.; Lin, H.H.; Wang, C. Quantum-inspired sublinear classical algorithms for solving low-rank linear systems. *arXiv:1811.04852* **2018**.
29. Gilyén, A.; Lloyd, S.; Tang, E. Quantum-inspired low-rank stochastic regression with logarithmic dependence on the dimension. *arXiv:1811.04909* **2018**.
30. Arrazola, J.M.; Delgado, A.; Bardhan, B.R.; Lloyd, S. Quantum-inspired algorithms in practice. *Quantum* **2020**, *4*, 307. <https://doi.org/10.22331/q-2020-08-13-307>.
31. Chia, N.H.; Gilyén, A.; Li, T.; Lin, H.H.; Tang, E.; Wang, C. Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing Quantum machine learning. *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing* **2020**. <https://doi.org/10.1145/3357713.3384314>.
32. Arrazola, J.M.; Delgado, A.; Bardhan, B.R.; Lloyd, S. Quantum-inspired algorithms in practice. *Quantum* **2020**, *4*, 307. <https://doi.org/10.22331/q-2020-08-13-307>.
33. Chia, N.H.; Gilyén, A.; Lin, H.H.; Lloyd, S.; Tang, E.; Wang, C. Quantum-Inspired Algorithms for Solving Low-Rank Linear Equation Systems with Logarithmic Dependence on the Dimension. In Proceedings of the 31st International Symposium on Algorithms and Computation (ISAAC 2020); Cao, Y.; Cheng, S.W.; Li, M., Eds.; Schloss Dagstuhl–Leibniz-Zentrum für Informatik: Dagstuhl, Germany, 2020; Vol. 181, *Leibniz International Proceedings in Informatics (LIPIcs)*, pp. 47:1–47:17. <https://doi.org/10.4230/LIPIcs.ISAAC.2020.47>.
34. Sergioli, G.; Santucci, E.; Didaci, L.; Miszczak, J.A.; Giuntini, R. A quantum-inspired version of the nearest mean classifier. *Soft Computing* **2018**, *22*, 691–705.

35. Sergioli, G.; Bosyk, G.M.; Santucci, E.; Giuntini, R. A quantum-inspired version of the classification problem. *International Journal of Theoretical Physics* **2017**, *56*, 3880–3888.
36. Subhi, G.M.; Messikh, A. Simple quantum circuit for pattern recognition based on nearest mean classifier. *International Journal on Perceptive and Cognitive Computing* **2016**, *2*.
37. Nguemto, S.; Leyton-Ortega, V. Re-QGAN: an optimized adversarial quantum circuit learning framework, 2022. <https://doi.org/10.48550/ARXIV.2208.02165>.
38. Eybpoosh, K.; Rezaghi, M.; Heydari, A. Applying inverse stereographic projection to manifold learning and clustering. *Applied Intelligence* **2022**, *52*, 4443–4457. <https://doi.org/10.1007/s10489-021-02513-0>.
39. Poggiali, A.; Berti, A.; Bernasconi, A.; Del Corso, G.; Guidotti, R. Quantum Clustering with k-Means: a Hybrid Approach. *arXiv preprint arXiv:2212.06691* **2022**.
40. de Veras, T.M.L.; de Araujo, I.C.S.; Park, D.K.; da Silva, A.J. Circuit-Based Quantum Random Access Memory for Classical Data With Continuous Amplitudes. *IEEE Transactions on Computers* **2021**, *70*, 2125–2135. <https://doi.org/10.1109/tc.2020.3037932>.
41. Hornik, K.; Feinerer, I.; Kober, M.; Buchta, C. Spherical k-Means Clustering. *Journal of Statistical Software* **2012**, *50*, 1–22. <https://doi.org/10.18637/jss.v050.i10>.
42. Feng, C.; Zhao, B.; Zhou, X.; Ding, X.; Shan, Z. An Enhanced Quantum K-Nearest Neighbor Classification Algorithm Based on Polar Distance. *Entropy* **2023**, *25*. <https://doi.org/10.3390/e25010127>.
43. Nielsen, M.A.; Chuang, I.L. *Quantum Computation and Quantum Information*; Cambridge University Press, 2000.
44. Lloyd, S. Least squares quantization in PCM. *IEEE Transactions on Information Theory* **1982**, *28*, 129–137. <https://doi.org/10.1109/TIT.1982.1056489>.
45. Schubert, E.; Lang, A.; Feher, G. Accelerating Spherical k-Means. In Proceedings of the Similarity Search and Applications; Reyes, N.; Connor, R.; Kriege, N.; Kazempour, D.; Bartolini, I.; Schubert, E.; Chen, J.J., Eds.; Springer International Publishing: Cham, 2021; pp. 217–231.
46. Ahlfors, L.V. *Complex Analysis*, 2 ed.; McGraw-Hill Book Company, 1966.
47. LaRose, R.; Coyle, B. Robust data encodings for quantum classifiers. *Physical Review A* **2020**, *102*. <https://doi.org/10.1103/physreva.102.032420>.
48. Weigold, M.; Barzen, J.; Leymann, F.; Salm, M. Expanding Data Encoding Patterns For Quantum Algorithms. In Proceedings of the 2021 IEEE 18th International Conference on Software Architecture Companion (ICSA-C), 2021, pp. 95–101. <https://doi.org/10.1109/ICSA-C52384.2021.00025>.
49. Fanizza, M.; Rosati, M.; Skotiniotis, M.; Calsamiglia, J.; Giovannetti, V. Beyond the Swap Test: Optimal Estimation of Quantum State Overlap. *Physical Review Letters* **2020**, *124*. <https://doi.org/10.1103/physrevlett.124.060503>.
50. Foulds, S.; Kendon, V.; Spiller, T. The controlled SWAP test for determining quantum entanglement. *Quantum Science and Technology* **2021**, *6*, 035002. <https://doi.org/10.1088/2058-9565/abe458>.
51. Microsystems, B. Digital modulation efficiencies.
52. Jr., L.E.F. *Electronics Explained: Fundamentals for Engineers, Technicians, and Makers*; Newnes, 2018.
53. Plesch, M.; Brukner, Č. Quantum-state preparation with universal gate decompositions. *Physical Review A* **2011**, *83*, 032302.
54. Quantum Computing Patterns. <https://quantumcomputingpatterns.org/>. Accessed: 2021-10-30.
55. Roy, B. All about Data Encoding for Quantum Machine Learning. <https://medium.com/data-driven-investor.com/all-about-data-encoding-for-quantum-machine-learning-2a7344b1dfef>, 2021.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.