UNIVERSITY OF COPENHAGEN

FACULTY OF SCIENCE

Bachelor thesis

Malthe Asmus Marciniak Nielsen

Statistical methods for single-shot readout discrimination in superconducting qubits

Date: June 2022

Advisors: Assistant professor Morten Kjaergaard & Post doc. Jacob Hastrup

Abstract

Superconducting quantum bits (qubits) are a leading platform for realizing high-fidelity quantum computing [1]. One aspect of this, is the ability to measure the qubit states to high fidelity. This is typically done by recording 'single shots' of the qubit state, which corresponds to a (complex) voltage. This voltage can be decomposed into an in-phase (I) and quadrature (Q) phase. This thesis is focused on statistical methods for discriminating between '0' and '1' in the IQ plane. Although the classifier effects readout fidelity significantly, there are no standardized method of classifier determination today. In this project we investigate the machine learning algorithm Support Vector Machines (SVM) as an automated method of determining the state classifier yielding the highest readout fidelity for four different qubit spanning three different devices. Furthermore, we analyze the performance of the classifiers using a multiple of statistical methods. This was achieved by the creation of the Python package Readout Discrimination Tools. We conclude that SVM efficiently determined quantum states with the highest fidelity being $85.5 \pm 0.4\%$ across the four used datasets. Single-shot sample sizes were determined to influence the readout fidelity when below 2000 single-shots pr. state.

List of abbreviations

SVM	Support Vector Machines	5
RDT	Readout Discrimination Tools	5
QHO	Quantum Harmonic Oscillator	5
IQ-plane	In-phase and Quadrature plane	5
RBF	Radical Basis Function	10
KNN	K-Nearest Neighbors	11
AdaBoost	Adaptive Boosting	11
LDA	Linear Discriminant Analysis	11
PCA	Principal Component Analysis	13
STD	Standard Deviation	13
Aalto	Aalto 190701 qubit 4	14
MIT T5	MIT 1023 from cryostat T5	14
MIT T3	MIT 1023 from cryostat T3	14
Soprano	Soprano device qubit 2	15

Table of contents

1 Introduction

In the past decades the field of quantum computing has made major progress. Experiments like Quantum Supremacy from Google strongly suggest and has mostly established that [2]: The theory of quantum computing is true. All points to that quantum computers work, and they have the potential to redefine huge sections of our society, ranging from drug discovery, cyber security, banking industry and even weather forecasting.

Quantum computers can compute exponentially faster than conventional computers. This is achieved by exploiting the core principles of quantum mechanics, that a qubit (quantum bit), unlike a conventional bit, can occupy multiple states at simultaneously [3, 4]. This means that even a few thousand of qubits potentially have a greater computational power than all conventional computers, combined. It likewise means that some unsolvable problems today, can be solved by ease in the future.

Computations depend on the output of bits. For conventional computers, bits are determined by a threshold in voltage, where only if the voltage is above a threshold, the bit is one. Analogously, for superconducting qubits these outputs are complex numbers referred to as single-shots. The unit of single-shots are likewise voltage, but in a complex space called the In-phase and Quadrature plane (IQ-plane). Determination of the threshold or classifier in the IQ-plane is, due to data noise non-trivial, but highly important due to the classifiers direct correlation with the outcome of computation [1, 5, 6].

In this project we attempted to optimize the determination of the classifier, by introducing the single-shot data to machine learning algorithms, which are primarily Support Vector Machines (SVM). We subsequently evaluated the performance of the classifier itself with the parameter readout fidelity. The qubits are not necessarily robust systems and therefore needs frequently re-calibration. By development of a Python packages, we created an easy-to-use tool to calibration single-shot readout classifiers.

2 Theory

In the following section we shall describe the theories used in this thesis, spanning from a general overview of superconducting qubits to methods in the field of machine learning. Finally we will introduce a Python-code package named Readout Discrimination Tools (RDT), as this will be the main tools for analyzing the subject of single-shot state distribution.

2.1 Superconducting qubits

Superconducting qubits use the superconducting properties of some metals when cooled to near base temperature ($T = 0$K. One promising type of superconducting qubit is the Transmon qubit [7]. A Transmon uses a Josephson junction to perturb the Quantum Harmonic Oscillator (QHO), a parallel LC-oscillator, achieving energy gaps in states where $\hbar\omega_{01} > \hbar\omega_{12}$. The QHO without a Josephoson junction consist of an inductor L (linear) and capacitor C in parallel corresponding to the Hamiltonian [3, 4]:

$$H = 4E_{CL}n^2 + \frac{1}{2}E_L\phi^2, \tag{1}$$

where $E_{CL} = e^2/(2C)$ is the energy required to add an extra single electron to the island, n is the number of Cooper-pairs and E_L is the energy from inductance L. ϕ is the phase of the waveform. When replacing the linear inductor$L$ with the Josephson junction, a nonlinear inductor, the Hamiltonian is modified to:

$$H = 4E_{CJ}n^2 + E_J cos(\phi), \tag{2}$$

where $E_{CJ} = e^2/(2C_{\sum})$, with $C_{\sum} = C_s + C_J$, where C_s is the capacitance from the circuit and C_J is the self-capacitance of the Josephson junction. $E_J = I_C\Phi_0/2\pi$ is the energy from the Josephson junction as a function of the critical current of the junction. This ensures that the potential is non-parabolic, which changes the energy gap between states to $\hbar\omega_n > \hbar\omega_{n+1}$ [3, 4]. The energy gab between the two lowest states are thereby distinguishable from higher energy gabs, which enable us to make a controllable qubit with these low energy gabs.

By placing a resonator close to our Josephson junction we can couple the two with a coupling constant g, resulting in a signal passing through the resonator, which will be effected by the state of the qubit. This coupling can be estimated from the dispersive shift:

$$\chi = g^2/\Delta, \tag{3}$$

where $\Delta =\mid \omega_q - \omega_r \mid$, with ω_q and ω_r as the qubit and resonator frequencies, respectively. This allows us to distinguish between two states by in using the Hamiltonian for dispersive approximation [3]:

$$H_{disp} = (\omega_r + \chi\sigma_z)(a^\dagger a + \frac{1}{2}) + \frac{\tilde{\omega_q}}{2}\sigma_z, \tag{4}$$

where a is the annihilation of a single excitation of the resonator, and $\tilde{\omega_q} = \omega_q + q^2/\Delta$ is the Lamb shift. This results in the dispersive shift for a Transmon qubit being qubit state, σ_z, dependent.

2.1.1 Single-shot readout data

We showed in the previous section that we are able to distinguish between states due to the state dependent frequency of the resonator. To utilize this in the most beneficial way, we want achieve the greatest separation between the voltages of the two states, meaning using a frequency just in between the two state frequencies. This maximises the separation in the IQ-plane, where I and Q is the in-phase and quadrature components of the voltage, respectively. After inducing a microwave signal to the resonator the reflected signal has the form [8]:

$$s(t) = A_{RO}\cos(\omega_{RO}t + \theta_{RO}), \tag{5}$$

where A_{RO} is the readout amplitude, θ_{RO} is the phase and ω_{RO} is the probe frequency. By transforming the into a real and imaginary part, we get the following equation, showing that the reflected signal contains both real and imaginary parts:

$$\begin{aligned} s(t) = Re\{A_{RO}\cos(\omega_{RO}t + \theta_{RO} \\ +j\sin(\omega_{RO}t + \theta_{RO})\}, \end{aligned} \tag{6}$$

We can extract values of I and Q by using a analog IQ-mixer. By defining a local oscillator (LO) signal with frequency $\omega_{LO} = \omega_{RO}$, we get a signal as a function of time t:

$$y(t) = A_{LO}\cos(\omega_{LO}t), \tag{7}$$

By splitting the LO signal and phase-shifting one part by $\pi/2$, we get I and Q parts of the LO signal:

$$y_I(t) = \frac{A_{LO}}{2}\cos(\omega_{LO}t) \tag{8}$$

$$y_Q(t) = -\frac{A_{LO}}{2}\sin(\omega_{LO}t) \tag{9}$$

Likewise, the same signal splitting occurs for the reflected signal $s(t)$, but with no phase-shift induced:

$$s_I(t) = \frac{A_{LO}}{2}\cos(\omega_{LO}t + \theta_{RO}) \tag{10}$$

$$s_Q(t) = \frac{A_{LO}}{2}\cos(\omega_{LO}t + \theta_{RO}) \tag{11}$$

Finally the I and Q values can be calculated by integration over the time-averaging (demodulation length) T:

$$I = \frac{1}{T}\int_0^T dt s_I(t)y_I(y) \tag{12}$$

$$Q = \frac{1}{T}\int_0^T dt s_Q(t)y_Q(y) \tag{13}$$

Subsequently the I and Q values can be used to calculate the amplitude and phase:

$$A_{RO} \propto \sqrt{I^2 + Q^2} \tag{14}$$

$$\theta_{RO} = \arctan(\mathrm{Q/I}) \tag{15}$$

Assuming a perfect no-noise qubit, all single-shots would fall close to the same point in the IQ-plane, but due to qubit decay (T1 and T2), measuring uncertainties, etc., the single-shot follows a Gaussian distribution. The difficulties in classification of states is discussed in the following section.

2.1.2 Readout fidelity

Quantum computing is limited by data noise, and to quantify this limitation, fidelity is commonly used. Readout fidelity is a measure of the ability to determine single-shots belonging to the ground or excited state. This fidelity can be determined in multiple ways. Here we present three of the methods. Common for all methods is that one compare single-shots with two different state preparations; one with no microwave pulse applied to the qubit, meaning the recorded single-shot data will correspond to the qubit being in the ground state, and the other with a pulse, preferably a π-pulse, where single-shot data will correspond to the qubit being in the excited state, see Figure 1a. Here the ground and excited state are visualised in the IQ-plane and a overlab in states are seen. The illustrated simulated data depicts a qubit at a temperature $T = 0K$. In data there would be single-shots with a π-pulse applied, meaning a pulse for the excited state, where the single-shots would show in the ground state, due to T1 qubit decay. Moreover, there would be single-shots prepared for the ground state shown in the excited state in the IQ-plane, due to the qubit temperature being non-zero. These being thermally excited single-shots. The rate of thermal excitation is a function of temperature derived from Boltzmann's equation of thermal probability [1, 9]:

$$\frac{p_0}{p_1} = \exp\left(\frac{\varepsilon_1 - \varepsilon_0}{k_B T}\right) = \exp\left(\frac{\varepsilon_1}{k_B T}\right), \tag{16}$$

where p_0 and p_1 are the probability of being in the ground and excited state. ε_0 and ε_1 is the relative energy states, where ε_1 can be defined:

$$\varepsilon_1 = \hbar\omega = 2\hbar\pi \cdot f, \tag{17}$$

where, f is the frequency. Finally the qubit temperature can be determined as:

$$T = \frac{2\hbar\pi \cdot f}{ln(p_0/p_1)} k_B, \tag{18}$$

Figure 1b shows readout fidelity of a simulated dataset (no thermal population, no qubit decay during readout) determined with the following calculation::

$$u_0(s) = \begin{cases} 1 & |s - \mu_0| > |s - \mu_1| \\ 0 & |s - \mu_0| \leq |s - \mu_1| \end{cases} \tag{19}$$

$$n_i(s) = \sum u_i(s), \tag{20}$$

where s is a specific single-shot. The mean of the ground state (μ_0) and excited state (μ_1) are estimated from the single-shots corresponding to specific state. Every single-shot is sorted into one of two bins (n_0, n_1) depending on distance from the two state means. Readout fidelity of the $i'th$ state is then defined as:

$$\text{fidelity}_i = \frac{n_i}{n_0 + n_1}, \tag{21}$$

When defining readout fidelity as shown in Equation 21, one assumes that the mean of each dataset corresponds to the state mean. Due to thermal excitation one can justify that the calculated means μ_0 and μ_1 falls in the interval of the true state means $[\mu_{0true} : \mu_{1true}]$, but likewise that $\mu_n \neq \mu_{ntrue}$, where n is a state.

Another method of estimation of readout fidelity is shown in Figure 1c, where a 2-dimensional double Gaussian is fitted to the combined set of single-shots. The double Gaussian is derived from a single Gaussian:

$$\begin{aligned} gauss(x,y) = A \cdot \exp(-a(x-x_0)^2 + \\ 2b(x-x_0)(y-y_0) + c(y-y_0)^2) + k, \end{aligned} \tag{22}$$

where x_0 and y_0 are the x and y coordinates of the mean, A is the amplitude, k is a constant and a, b and c are defined as follows:

$$a = \frac{\cos(\theta)^2}{2\sigma_x^2} + \frac{\sin(\theta)^2}{2\sigma_y^2}, \tag{23}$$

$$b = \frac{-\sin(\theta)^2}{4\sigma_y^2} + \frac{\sin(2\theta)^2}{4\sigma_y^2}, \tag{24}$$

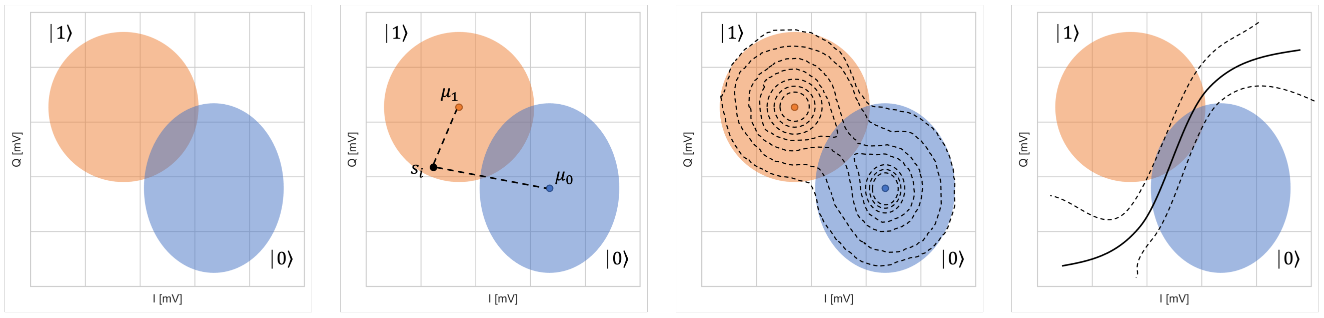
$$c = \frac{\sin(\theta)^2}{2\sigma_x^2} + \frac{\cos(\theta)^2}{2\sigma_y^2}, \tag{25}$$

where σ_x and σ_y are the width in the x and y plane respectively, and θ is the rotation around the (x_0, y_0). Combining two Gaussian's like shown:

$$\begin{aligned} gauss_{double}(x,y) = gauss_0(x,y) \\ + gauss_1(x,y), \end{aligned} \tag{26}$$

where the amplitudes A_0 and A_1 corresponds the to likelihood of generation of single-shots in the ground state and excited state, respectively. One can then used the fitted parameters and determine the amplitudes for only the ground state prepared data. Finally the readout fidelity of the ground stated can be determined as A_0. Comparing to equation 21, this method determines means, which are not limited in the previously mentioned interval, and the means can thereby be $\mu_n = \mu_{ntrue}$. This method assumes that the single-shots follows a Gaussian distribution. With T1 and T2 decay this assumption is not necessarily true but an estimation [5].

The final method, shown in Figure 1c, shows fidelity defined by machine learning algorithms. This is described in details in the following section.



(a) (b) (c) (d)

Figure 1: (a) The two state marked as $|0\rangle$ and $|1\rangle$. This is a simulation, assuming no thermal population (i.e. $T = 0$K). **(b)** Fidelity calculated with Equation 21. **(c)** States fitted with double Gaussian. **(d)** Fitted SVM on single-shots.

2.2 Machine learning

The determination of the classifier yielding the highest readout fidelity for a given set of single-shots in the IQ-plane can be achieved in multiple ways [10, 11]. For a dataset with low noise and great distance between the two Gaussian distributed states, one might be encouraged to assume choosing an arbitrary line dividing the two states, would yield the same readout fidelity, as a non-linear classifier function. Although this may be the case, the assumption can be proven or dis-proven by the use of machine learning.

Machine learning comes in a variety of different forms, but the denominator of them all is a cost function. Cost functions are functions which are minimized or maximized during the machine learning progress. In the case of the single-shot readout classification, the machine learning algorithm maximizes the readout fidelity. In other words, the algorithm evaluates the frequency of rightly compared to wrongly assigned single-shots.

2.2.1 Cross validation

A common imperfection in machine learning algorithms is over-fitting [12]. A method to overcome this drawback of over-fitting is named K-fold. When using K-fold, one simply splits the dataset into K-folds, meaning K sub datasets. Artifacts like small clusters of noise can effect the hyper-parameters, the parameters fitted during the algorithm, if cross validation is not used. It is common practice to divide a dataset into 10 randomly chosen folds of equal size. One fold is set aside for final evaluation of the algorithms output. The remaining n-1 folds are then used as training sets, as seen in Figure 2. Here a 5 fold setup is used as illustration. One fold, called the 'test data', is saved and stays hidden from the algorithm. The remaining 4 folds are used as 'training data'. Training data can be fitted in multiple ways. One way is called 'one against the rest'. As depicted in figure 2 'the one against the rest' method uses one of the K-folds as the test data in $i'th$ iteration. After all iterations are performed the results are averaged and the hyper-parameters thereby determined. The final step is comparing the determined hyper-parameters with the initial test data. The performance of the algorithm is defined as the score from the initial test dataset. K-folds are one of multiple outer-parameters, which are not changed during fitting but can impact the outcome significantly.

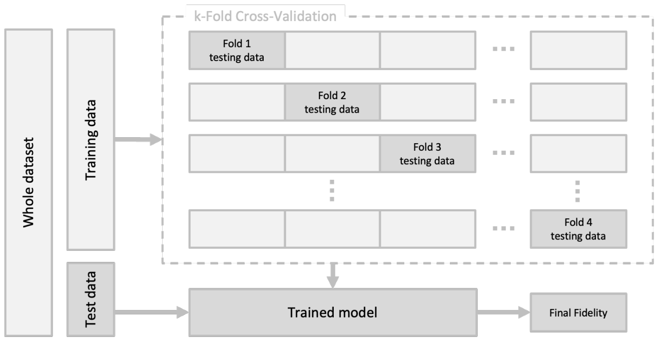


Figure 2: The structure of data handling in cross validation, when using K-fold. The dataset is divided into K-fold, in the case $K = n$. One fold is saved as 'test data' and remains unseen for the algorithm until final performance testing. The reaming $K - 1$ folds are used as training data.

2.2.2 Support vector machine

The objective of the SVM is to the maximize the margin in-between the separate classes of a given dataset. The SVM is a discriminative classifier functioning in a N-dimensional space by separating the hyperplanes of the space [13, 12, 14]. In other words, it separates labeled classes of a dataset. For single-shot data in the IQ-plane (2D space) containing only two states, the SVM would return a classifier in the form of a function, which would determine the state of each single-shot in the dataset. Visually a SVM functions as a classifier by transforming a N-dimensional space into a higher dimension, as seen in Figure 3, where a 2-dimensional space (3a) is transformed into 3 dimensions by a Gaussian kernel (3b). The hyperplane with the maximized margin between classes is thereby determined (3c). By optimizing a margin a SVM algorithm has an advantage compared to other machine learning algorithms, because a SVM classifier is determined with a degree uncertainty. There are multiple different forms of SVM's called kernels. This project focuses on the differences in the linear (Equation 27), polynomial (Equation 28), Radical Basis Function (RBF) (Equation 29) and Sigmoid kernels (Equation 30). Each kernel consists of an individual set of hyper parameters, as shown below, which are varied during the fitting.

$$K_{linear}(x_i, x_j) = (x_i, x_j) + c, \tag{27}$$

where c is an offset constant common to all SVM kernels used. By only containing one hyper-parameter, the linear SVM kernel is the simplest and thereby the most consistent, but likewise the least computationally demanding.

$$K_{polynomial}(x_i, x_j) = (\gamma(x_i \cdot x_j) + r)^d + c, \tag{28}$$

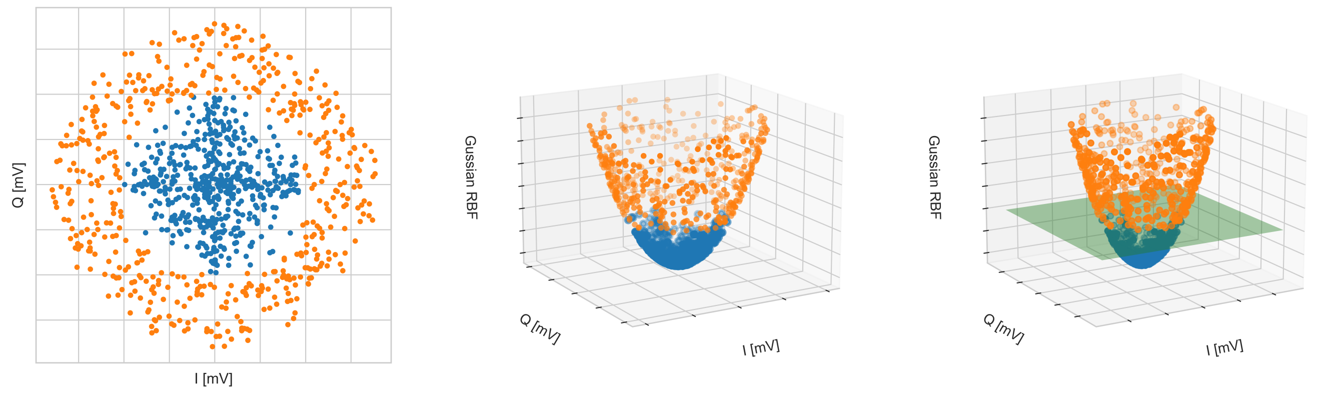
where d is the polynomial degree, r is an offset constant and γ determines the influence of a single training set.

$$K_{RBF}(x_i, x_j) = \exp(-\gamma \left\| x_i - x_j \right\|^2) + c, \tag{29}$$

where γ likewise determines the influence of a training set.

$$K_{sigmoid}(x_i, x_j) = \tanh(\gamma(x_i \cdot x_j) + r) + c, \tag{30}$$

where r is a offset constant. A benefit of SVM's is the ability to classify in N-states for multi qubit operations [15, 16].



(a) **(b)** **(c)**

Figure 3: **(a)** Two class data, not solvable with a linear classifier in 2D-space. **(b)** Same data in 3D-shape, transformed by a Gaussian kernel. **(c)** Classes divided in 3D-shapes by hyper-plane.

2.2.3 Alternative kernels

In addition to SVMs a different set of algorithms were tested as a baselines during this project. Below we briefly discuss these:

K-Nearest Neighbors (KNN): A classifier which uses proximity as its only parameter. It counts the number of same-class neighbors of a specific data-point i. The $i'th$ data-point is thereby weighed and the algorithm maximizes the weight for the data-set.

Decision Tree: A classic decision tree classifier [11].

Adaptive Boosting (AdaBoost) : A meta-estimator that combines multiple 'weak' classifiers into one strong. It uses decision trees with only one spilt, called decision stump.

Linear Discriminant Analysis (LDA): Lowers the dimension by creating a new axis, where the distance between mean of classes is maximized and the variation within each class is minimized.

3 Python script

The main part of this project has been the development of the Python-code package Readout Discrimination Tools (RDT). While this thesis only focuses on single-qubits, the purpose of RDT was to create an easy-to-use package with the ability to classify and analyze states in single-shot data from one or multiple qubits. It is necessary to re-calibrate the single-shot classifiers frequently, due to the previously mentioned need of re-calibration of qubits. By applying SVM algorithms to single-shot data we theorized that a low-uncertainty, high fidelity classifier could be determined. A great part of RDT contains statistical tools of performance evaluation, meaning the RDT package is dual purpose by not only determining a single-shot readout classifier but also by providing an analyzing tool of the classifier itself. Figure 4 shows the structure of RDT and the data flow within. Inputting single-shot data belonging to two separate states, one being the ground state and the other the excited state as complex numbers of the IQ-plane, where-after RDT defines the data-set size as specified by the user. After selection

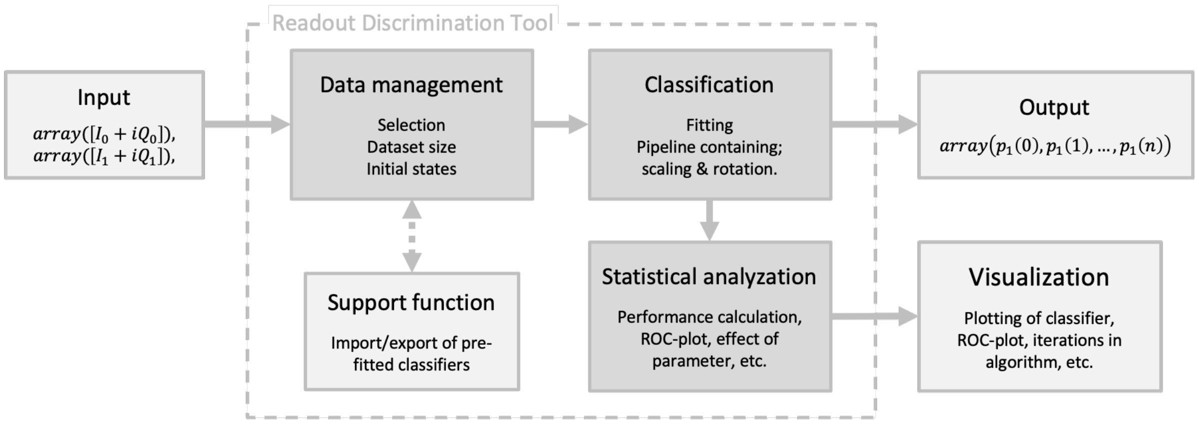


Figure 4: General structure of RDT. Data is imported (left) as arrays of complex numbers. The data is then transformed in Data Management, where the data is down scaled to fit dataset size. Training of classifier is performed with output of expectation values in return. Statistical analysis can return multiple different plots describing the performance and robustness of the classifier.

of data, the fitting of the classifier is performed, where after the user can save the fitted SVM and use it to estimate expectation values of data subsets. Moreover, the user can run a multiple of statistical tests on the classifier to evaluate it's performance and robustness.

3.1 Data import

When determining the classifier using the SVM two separate sets of data are needed. Half of the data points must stem from the experiment without a π-pulse applied, which corresponds to the qubit in the ground state. The other half of the data points stem from an experiment with a π-pulse, which corresponds to the qubit being in its first excited state. The single-shots are labeled responding to their state, meaning that all data points from the first half of the dataset are labeled as the ground state (0). The second half is labeled as the excited state (1). If more than two subsets of data are imported into RDT, the code will automatically determined the best candidates for classifier determination, by calling the _min_max_index() function, see Listing 1. If 'entries' is defined as 'None', the first index is defined as the ground state index. The excited state index is defined as the index for witch the mean of all single-shots in the subset is the furthest away from the mean of the ground state. This function ensures the optimal data is used for classification.

```python
def _min_max_index(self, X=None):
    if X is None:
        X = self.h5data

    means = linalg.norm(X, axis=2)
    data_mean = sum(means, axis=1)

    min_, max_ = argmin(data_mean),
   argmax(data_mean)
    return data_mean, [min_, max_]
```

Listing 1: Shows the _min_max_index() function from RDT code, which determines the minimum and maximum index of complex datasets. This is used for selecting the excited state data.

```python
def set_pipeline(self, scalar=True, pca=True):
    if scalar == True:
        scalar = StandardScaler()
    else:
        scalar = None

    if pca == True or int:
        if pca == int:
            pca = PCA(n_components=pca)
        else:
            pca = PCA(n_components=2)
    else:
        pca = None

    self.pipeline = Pipeline(steps=[
        ('transformer', scalar),
        ('PCA', pca),
        ('classifier', self.classifier)
    ])
```

Listing 2: Shows the set_pipeline() function from RDT code, which constructs a pipeline with normalization, Principal Component Analysis (PCA) and the classifier function. This ensures that the Standard Deviation (STD) and rotation will not effect the outcome of the classifier algorithm.

After the two indices have been selected, a pipeline is constructed, to ensure that the user receives the classifier with the highest readout fidelity. A pipeline of instructions is set ensuring a uniform calculation of the classifier, see Listing 2. The pipeline has three components. Firstly it standardizes the data, meaning scaling the dataset to have a mean at (0, 0) and a variance of 1. This is optimal for machine learning algorithms. The second step of the pipeline is using PCA, which rotates the data in the IQ-plane to ensure the greatest separation between the different states on the I axis (PCA component axis 1) [17]. Finally the SVM machine learning algorithm is applied to the dataset.

3.2 Simple classifier test

Determination of the best SVM classifier for a dataset of single-shots can be done using the script in Listing 3. Here the datafile is selected and run with the RDT package. The return of the script is the classifier plotted with the single-shot data.

```python
import quantum_fitter.readout_tools as rdt

# Set up path
file_path = '.../example_data/ss_q1_rabi_v_ampl_5_fine.hdf5'

# Set the instence
rd = rdt.Readout(file_path, size=4000, verbose=1)

# Plot
rd.plot_classifier()
```

Listing 3: A short script for determination of the best classifier for at given dataset. The rd.plot_classifier() function plots the classifier.

3.3 Testing effect of parameter

An important aspect of the RDT code is the ability to determine the effect a given parameter has on the outcome of the performances of the classifier. To estimate the effect one can sweep the performances as a function of the parameter in a chosen interval. Therefore, to test the effect of a parameter, the parameter needs to be changed where-after the classifier can be determined. This can be done like seen in Listing 4. Here the effect of the dataset size is being test.

```python
import quantum_fitter.readout_tools as
    rdt
from tqdm import tqdm

# Set up path
file_path = '.../
    ss_q1_rabi_v_ampl_5_fine.hdf5'


# Set the instence
rd = rdt.Readout(file_path, verbose=0)

# Set the forloop
set_list = np.geomspace(20, 4000, num
    =50, dtype=int)
for i in tqdm(set_list):
     rd.set_dataset_size(size=i)
     rd.set_plot_dir(i, param_name='
    Dataset size', score_name=None)

# Plots figure
rd.plot_param_effect()
```

Listing 4: To determine the effect of a parameter, the listed script is executed. The analyzed parameter here is dataset size. The classifiers performance will be estimated for all values in 'set_list'.

3.4 Rabi data fitting

The performance of a fitted classifier can be evaluate in multiple ways. One of them is by fitting rabi oscillations. For a qubit with perfect state preparation and high separation of states in the IQ-plane, a fit of rabi oscillations will have an amplitude $A = 0.5$. If the separation or state preparation is non-perfect the absolute value of the amplitude $\mid A \mid < 0.5$. Rabi oscillations can be fitted using the RDT packages. This is done by determining the expectation value of all indexes in the dataset and fitting this with the following equation:

$$f(x) = A \cdot sin(\omega x + \varphi) + c, \tag{31}$$

where A is the amplitude, ω is the rabi frequency, φ is the phase and c is a constant offset.

4 Results and discussion

In the following section we show the results of the RDT package used on four different single-shot dataset and discuss the implications of these.

4.1 Data acquisition

We aimed to investigate the general performance and overall usability of SVM's as classifiers for single-shot readout states, and to accomplish this we picked three various rabi dataset stemming from two different devices from prior studies in the CQED-at-QDev-group. These qubits are; the Aalto 190701 qubit 4 (Aalto), the MIT 1023 from cryostat T5 (MIT T5) and the MIT 1023 from cryostat T3 (MIT T3). The rabi experiments for Aalto and MIT T5 were performed by ph.d. Jose Manuel Chavez-Garcia, and MIT T3 was performed by ph.d. Oscar Erlandsson. Furthermore, we used the five qubit Soprano device from Qunatware shown in Figure 5.

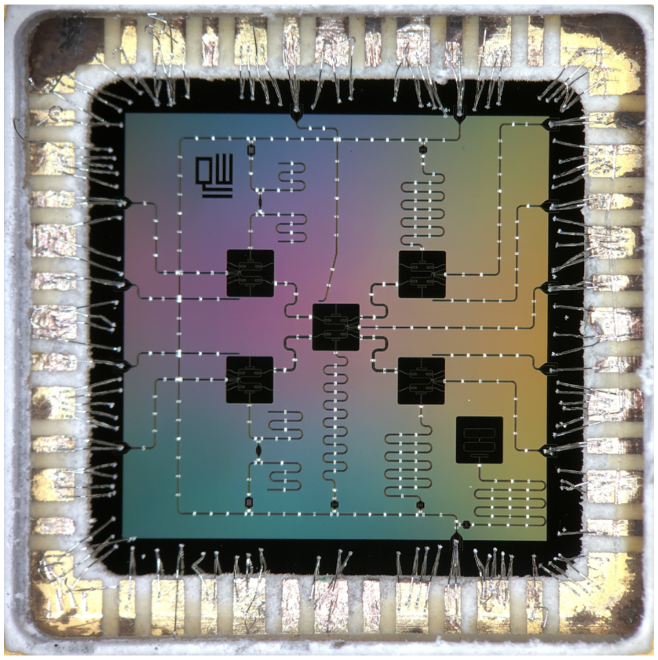


Figure 5: Picture of the Soprano device, showing six qubits (black squares), whereof five are controllable, and the last is a control (bottom right). From Quantware.

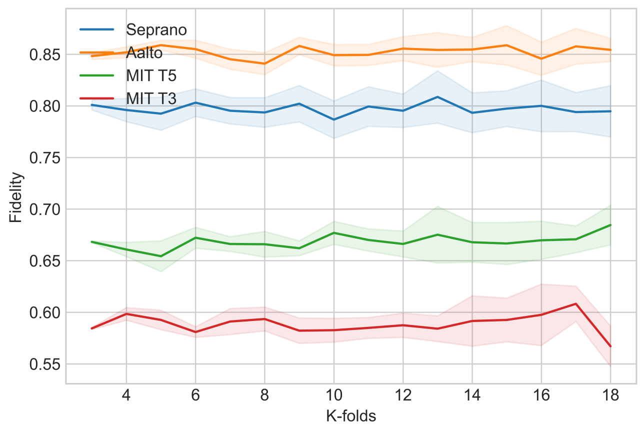


Figure 6: Reaodut fidelity as a function of K-folds for all four qubits. Solid line is the result. Transparent area is the STD.

We calibrated Soprano device qubit 2 (Soprano) with a flux current of 7mA, a drive frequency of 5.02891GHz, a drive power of −35dBM, a readout frequency of 7.63164GHz, a readout power of −37.6915dBm, a readout amplitude of 434.218mV and readout duration of 5.99436us. Finally we swept the pulse amplitude in the interval $[0, 1]$V in 201 steps and took 13.000 single-shots per step. These values were optimized using our qubit calibration script shown in Appendix.

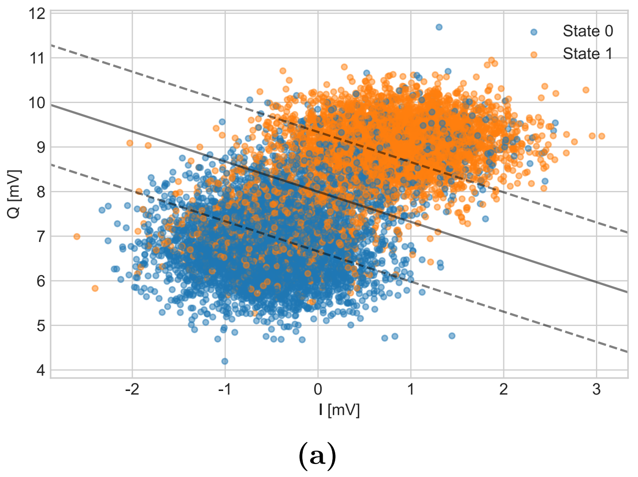
All experiments were performed on all four datasets. However, we focused on the Soprano data. It is noted, when simulated data is used.

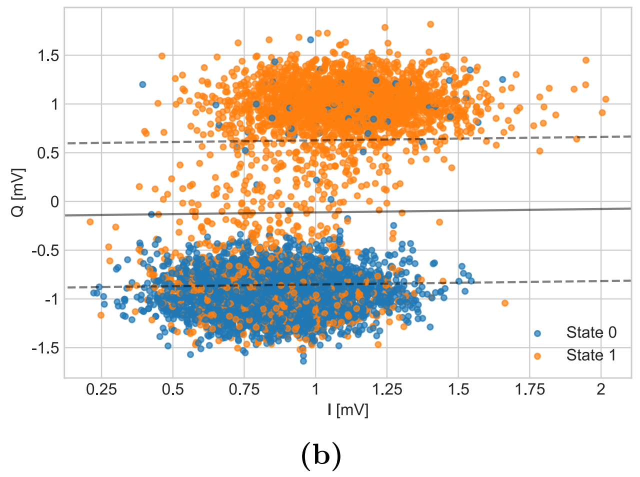
4.2 Outer-parameter determination

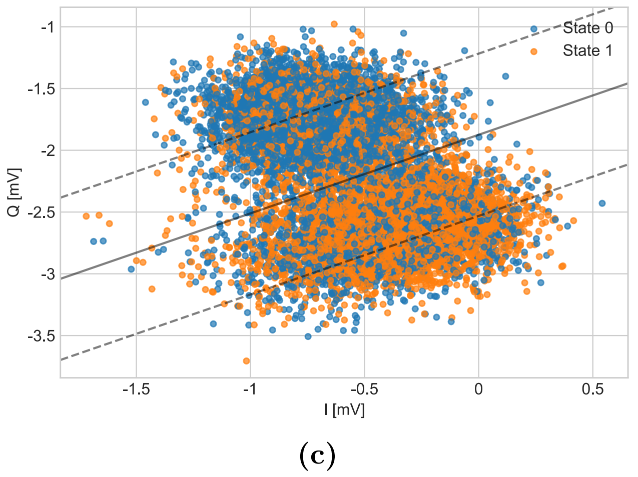
In the following section we investigate the outer-parameters of the machine learning algorithm and their effects on the readout fidelity. Outer-parameters, like mentioned in section 2.2, are parameters other then the SVM's hyper-parameters, which are not alternated doing fitting but may impact the result.

4.2.1 Cross validation optimized at K-fold = 10

We studied the effect of cross validation to determine its impact on the fitted classifier and to evaluate the change in readout fidelity. Initially we ran the algorithm with K-fold= 2, meaning the complete dataset was split into two folds; the test and training subset, effectively making cross validation invalid, and then compared the results to K-fold = 3 to 15, using all kernels letting the algorithm pick the candidate with the highest fidelity. The result seen in Figure 6 shows that the mean readout fidelity of all K-folds for Aalto and Soprano is higher than determined for MIT T5 and MIT T3, with a readout fidelity of $\sim 85\%$ and $\sim 80\%$ for Aalto







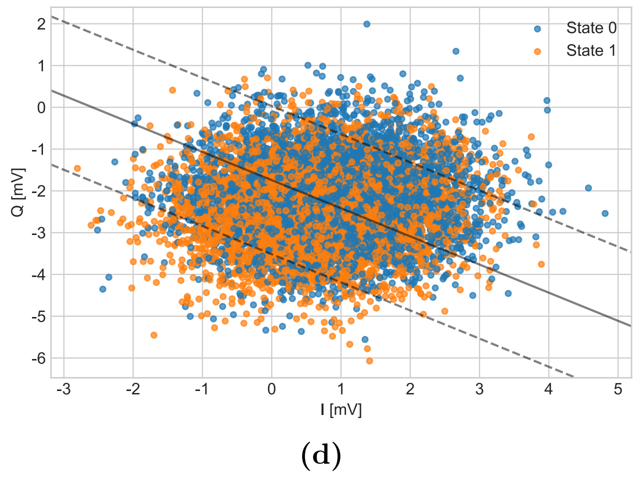


Figure 7: Single-shot data from ground (blue) and excited (orange) state entries, with fitted the best performing classifier. **(a)** the Soprano, **(b)** the Aalto, **(c)** the MIT T5 and **(d)** the MIT T3. All classifier are linear.

and Soprano, respectively. The mean readout fidelity of MIT T5 and MIT T3 are $\sim 58\%$ and $\sim 67\%$, respectively. The lower readout fidelity for the MIT qubits can be justify by the relative distance between state seen in Figure 7. Furthermore, we observed that K-fold did not statistically effect the performance of the classifier for either of the devices. In all further experiments K-fold = 10 used. Because the number of K-fold did not effect the readout fidelity significantly, it is theorized that small sample sizes likewise dose not effect the algorithms ability to determine the optimal classifier for a qubit.

4.2.2 Sample size controls readout fidelity

It was shown that splitting datasets into many folds (more than 10) in cross validation did not effect the performance of the classifier negatively. To study the effect of the size of the dataset further we looked at the sample size with a fixed K-fold. We used the same data as previously, with K-fold = 10, and determined the optimal classifier for the dataset with sample size ranging from 20 to 4000 single-shots per state, meaning 40 - 8000 single-shots in total. We observed from the results shown i Figure 8, that the number of single-shots effects the readout fidelity significantly. Classifiers fitted with a number of single-shots ranging from 20 to 38 have a mean readout fidelity of $\sim$ 50%. A great increase in performance was observed from 38 - 42 single-shots. It is hypothesized that this is due to the cross validations threshold in number of iterations, meaning with only 38 data points the algorithm only performed iteration of fitting, yielding significantly lower readout fidelity. In the region 42-1500 single-shots, the STD in performance decreases. From 1500 to 4000 single-shots the performance and STD remains statistically the approximately equal, suggesting that around 2000 single-shots per state is a minimum of sample size for the four specific datasets. To ensure the minimum effect of the data size, 6000 single-shots per state are used in all following experiments.

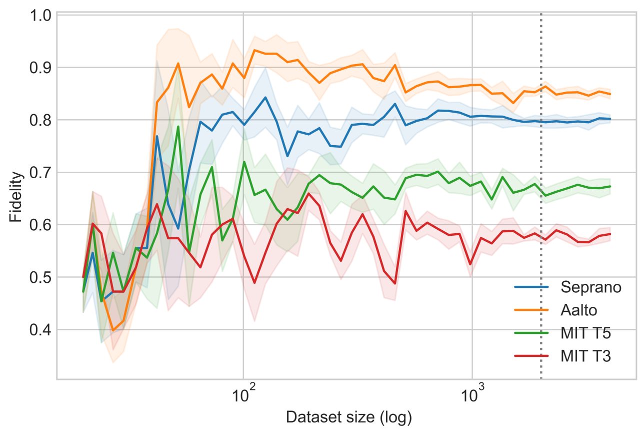


Figure 8: Readout fidelity as a function of dataset size for all four qubits. Solid line is the result. Transparent area is the STD. The vertical dotted line represent the 2000 single-shots line on the log scale

4.3 Kernel comparisons

In the previous section we discussed the outer-parameters effect on the readout fidelity. In the following section we look at the inner-parameters, being the hyper-parameters, and compare the performance of kernels.

4.3.1 Readout fidelity dependents on kernels

We wanted to compare the performance of the different kernels, being linear, polynomial, RBF and Sigmoid. To compare the performance of the kernels all four datasets were used. The hyper-parameters of the kernels were set in following intervals: The degree $d = [2, 4]$, $r = [1, 15]$, $c = [1 \times 10^{-3}, 1 \times 10^{4}]$ and gamma γ as 'scale' or 'auto', being two internal options. The machine learning algorithm was run with 10 K-folds, 12000 data points and tested on the combined dataset of the ground and first excited state. The results listed in Table 1 show the performance of the different kernels for Soprano (tables for other qubits are seen in Appendix). Here we see kernel types listed with number of resources (number of single-shots) used to determined the classifier and the mean training and testing scores. Furthermore, we see the performance of KNN, AdaBoost, decision tree and the LDA. We observed that the linear, polynomial and RBF kernels performed statistically the same, with RBF as marginally better. The Sigmoid kernels performed poorly compared to the previously mentioned kernels. Furthermore, we observe that the SVM kernels perform similar to baseline-algorithms. The performance of the four SVM kernels across all four qubits is shown in Figure 9 and 10, where we observe the

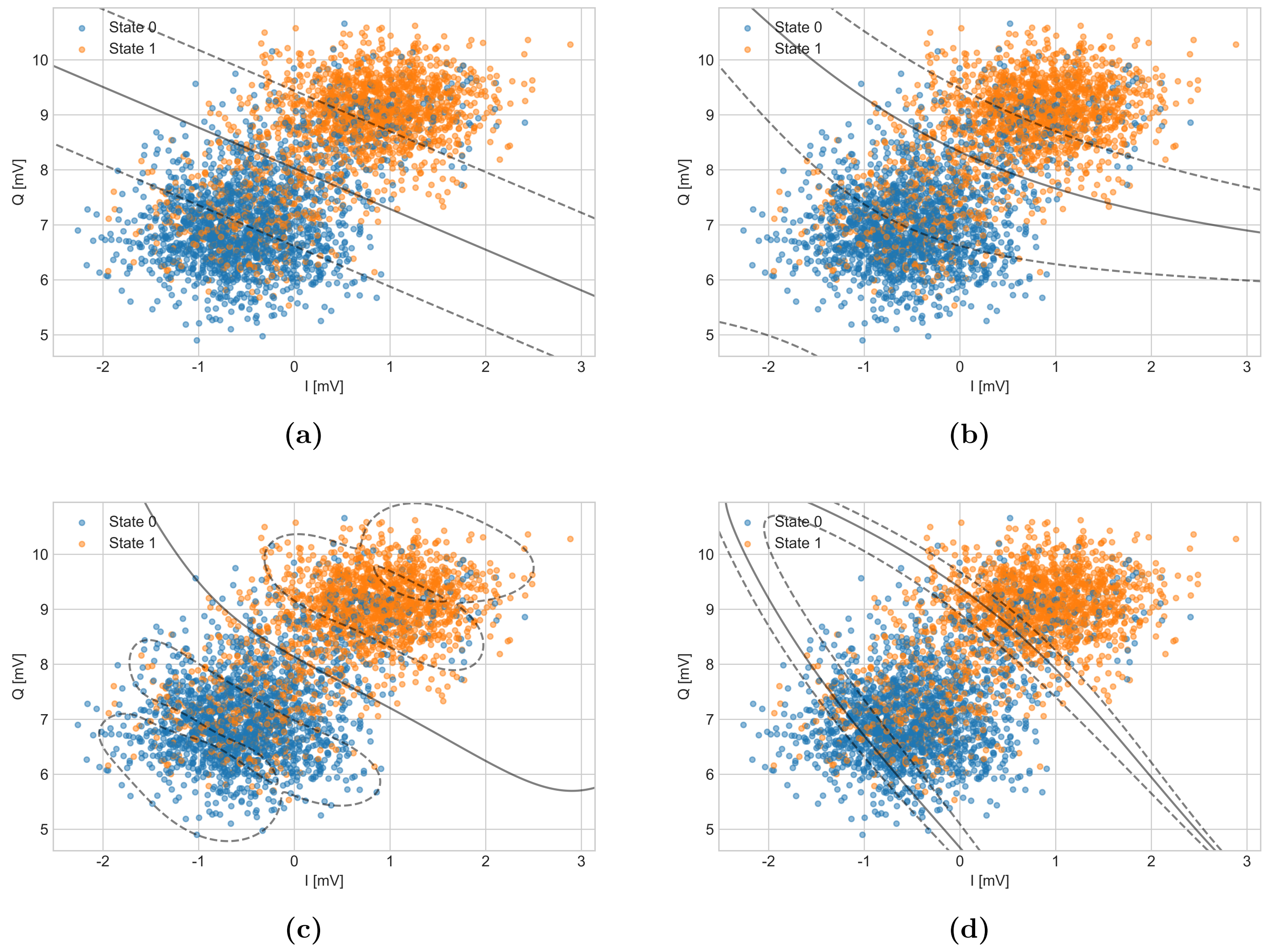


Figure 9: Single-shot data from Soprano, with different kernels. **(a)** Linear kernel, **(b)** Polynomial kernel, **(c)** RBF kernel and **(d)** Sigmoid kernel.

relatively pour readout fidelity when using the Sigmoid. Although the LDA performed similar to the SVM kernels, its limitations of only classifying two classes makes it unsuited for multi-qubit single-shot data. Moreover, the classifiers fitted with KNN, AdaBoost and decision tree are 'jagged', meaning not physical. This is non-problematic for qubits with low fidelity measurements, but it but it could be theorized that for high fidelity readout the jagged classifier would perform less efficiently then the SVM kernels. Because the kernels performance was so similar, we hypothesized no difference overall, except if artifacts are present.

4.3.2 Simulated artifacts effect kernels performance

The readout fidelity yielded from the different kernels was shown to be statistically equal across all four dataset, when no artifacts were present. Therefore, we wanted to determine the effect of artifacts. To investigate artifacts effect on the readout fidelity across the kernels, we used simulated Gaussian distributed single-shots with mean $\mu = (0, 10)$ mV and width $\sigma = (0.30, 0.75)$ mV. By appending the simulated single-shots to the ground state of the Soprano data, the readout fidelity was estimated as a function of percentage of added simulated single-shots. The intervals for the hyper-parameters were kept the same. 4000 data points were used.

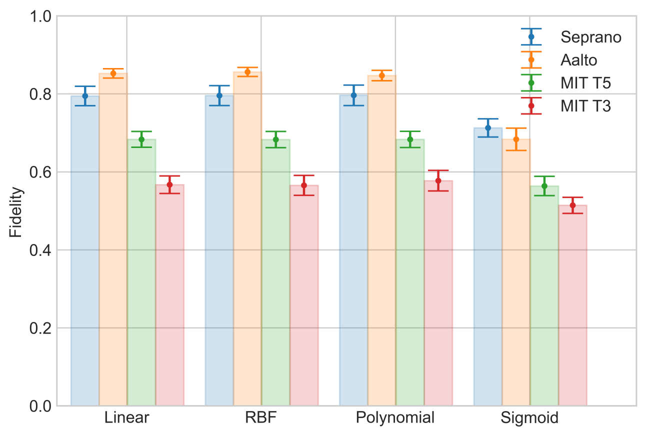


Figure 10: Reaodut fidelity as a function of kernel for all four qubits. Errorbars represent is the STD.

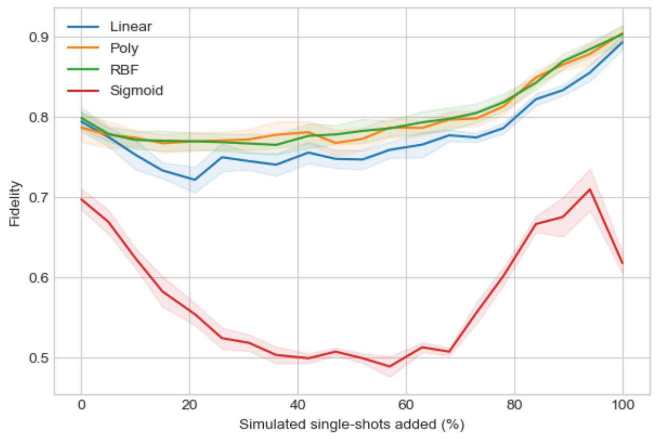


Figure 11: Readout fidelity as a function of % added simulated single-shots for all four kernels. Solid line is the result. Transparent area is the STD.

The results show, that Sigmoid kernel generally performed less efficient, than the linear, polynomial and RBF. The three remaining kernels performed similar at 0% and 100% simulated single-shots added at $\sim$ 80% and $\sim$ 90%, respectively. The polynomial and RBF performed similar in the entire range. Compared to the linear kernel, the readout fidelity has approximately $\sim$ 2% lower, suggesting that the linear kernels might be more affected by artifacts in the IQ-plane. To study this further one should acquire non-simulated real data with different forms of artifacts.

4.4 Performance depending values

Finally, after determination the classifier yielding the highest readout fidelity, we wanted to use this classifier to extract information about our qubits. In the following section we investigate the performance of all four qubits by using the determined classifiers from two entries of the rabi oscillation datasets.

4.4.1 Expectation values from rabi fitting

To estimate the performance of the qubits, we wanted to fit the rabi oscillations. For this purposes we used the four sets of single-shot data with the previously determined classifiers and determined the expectation value at a given pulse amplitude as the ratio of ground compared to excited state single-shots. The results depicted in Figure 12b show Soprano qubit oscillations between the ground and excited state. At a pulse amplitude $A_{pulse} = 0$V the qubit is closest to ground state, with a probability of being in state 1 below 20%. Furthermore, we see the qubit closest to the excited state at pulse amplitude $A_{pulse} = 0.65$V. The oscillation is fitted with Equation 31, where the oscillation amplitude $|A| = 0.235 \pm 0.004$ is a measure of performance. Mentioned in page 14 the oscillation

Classifier	Resources	Mean train score	Mean test score
SVM linear	8748	80.77±0.27 %	81.0±1.5 %
SVM polynomial	8748	80.82±0.27 %	81.0±1.4 %
SVM RBF	8748	81.08±0.27 %	81.3±1.5 %
SVM sigmoid	8748	74.0±0.4 %	74.3±0.9 %
KNN	8748	81.21±0.27 %	81.2±1.4 %
AdaBoost	8748	81.07±0.26 %	81.2±1.3 %
Decision tree	8748	80.95±0.22 %	81.1±1.4 %
LDA	108	84.4±2.1 %	81±10 %

Table 1: Soprano performers for all four kernels and the four other machine learning algorithms. Tables for the other devices can be seen in 9 Appendix.

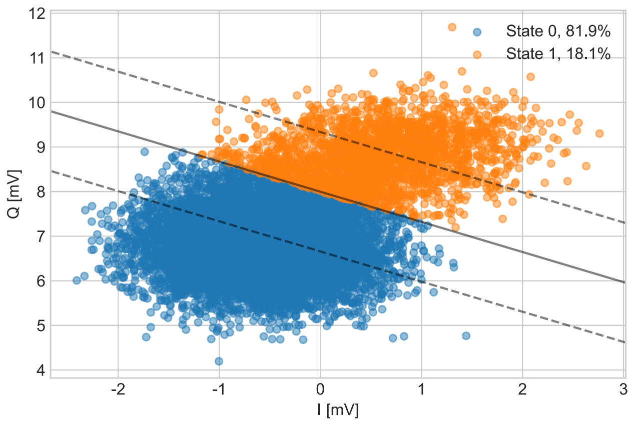
Qubit	Soprano	Aalto	MIT T5	MIT T3
Kernel	Linear	Linear	Linear	Linear
Amplitude (A)	0.235 ±0.004	0.30 ±0.02	0.168 ±0.002	0.080 ±0.003
Frequency (ω)	3.92 ±0.03	9.3 ±0.2	8.11 ±0.04	5.66 ±0.07
Phase (ϕ)	1.92 ±0.03	39.1 ±0.1	1.59 ±0.02	2.05 ±0.07
Offset (c)	0.483 ±0.003	0.39 ±0.01	0.532 ±0.001	0.503 ±0.002
Temperature (T) [mK]	159.5	58.17	248.9	430.7

Table 2: Rabi oscillations from all four qubits fitted with Equation 31. A amplitude $A = 0.5$ corresponds to a fidelity of 1. The offset $c = 0.5$ shows a qubit with equally good ground and excited state. $c > 0.5$ indicates a warm qubit. $c < 0.5$ indicates a non-perfect π-pulse.

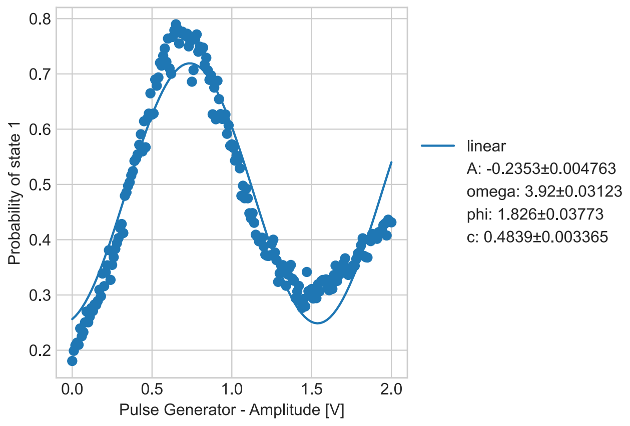
amplitude for a perfect qubit is $|A| = 0.5$. All fitted parameters are listed in Table 2, where we observe that the Aalto amplitude obtains the highest value $A = 0.30 \pm 0.02$, but with an offset $c = 0.39 \pm 0.01$ suggesting that the device is at a low temperature but the π-pulse is not probably calibrated. We can conclude that the SVM classifiers are able to efficiently fit and classify rabi oscillations, and that none of the qubits yield a oscillation amplitude $A = 0.5$. To study this further one could determine the β-matrix of systems, fit with a double Gaussian and compare the results with the results from Table 2 [18].

4.4.2 Temperature calculations

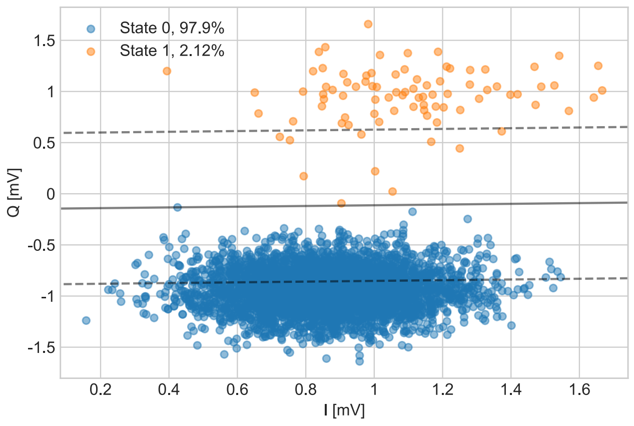
Ground state prepared single-shots from a qubit at base temperature ($T = 0$K would be expected to be in the ground state in the IQ-plane. When the qubit temperature raises above base temperature $T > 0$K, thermal excitation becomes present, meaning ground state prepared single-shots would populate the excited state. We can therefore estimate the qubit temperature from the expectation value of the ground state data. We did this by using best classifier and Equation 18. Figure 12a shows the single-shot data of Soprano with no pulse. A qubit temperature of 159.5 mK was determined. The temperatures for all four qubits are shown in Table 2, where it can be observed, that the qubit temperature 58.17 mK of Aalto is considerable lower compared to the other three qubits. This can likewise be seen in Figure 12c and 12d. Due to the overestimation of expectation values dis-



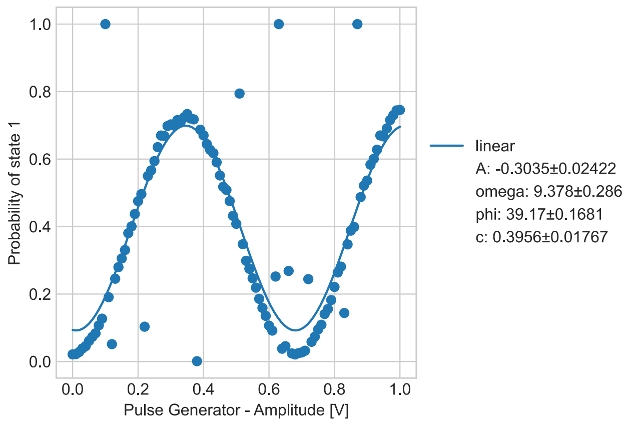
(a)



(b)



(c)



(d)

Figure 12: **(a)** Soprano single-shot data from ground fitted the best performing classifier. **(b)** Soprano rabi oscillation fitted with 31. **(c)** Aalto single-shot data from ground fitted the best performing classifier. **(d)** Aalto rabi oscillation fitted with 31.

cussed in subsection Readout Fidelity, the temperatures are suspected to be overestimated as well. To study this further one should compare the temperature calculation results when using different determination methods of expectation values.

5 Conclusion

We can conclude that Support Vector Machines can be used as an effective method for classification of states in superconducting single-shot data in the IQ-plane. Furthermore, it was shown that the sample size and cross validation are key components when determining the classifier with the highest readout fidelity. Hyperparameters was determined to effect readout fidelity greatly. Simulating single-shots artifacts in ground state showed that a linear kernels is more sensitive than both a RBF and a polynomial kernel. For Gaussian distributed single-shots the linear, polynomial and RBF kernels performed similar, but deviating from Gaussian distributions showed that polynomial and RBF overall actives a higher readout fidelity. Lastly we showed that classifier can be used when fitting rabi oscillations and that temperatures can be calculated from this data.

6 Future perspectives

To study this field further one might investigate the effect of different methods in determination of readout fidelity, this being with the methods described in section 2.1.2 Readout fidelity. We theorized that the qubit temperatures were overestimated due to T1 decay and overlab in states. One might compare temperature calculations from different fidelity estimation methods. Lastly, this entire subject of readout fidelity depends highly on qubit state preparation. To achieve high fidelity readout in single-shot data, one should study qubit calibration and the automation of this.

7 Acknowledgments

I would like to thank my supervisors, assistant professor Morten Kjaergaard and Post doc. Jacob Hastrup (NBI University of Copenhagen) for guiding and advising me in this project. Furthermore, I would like to give a thanks to the entire CQED-at-QDev, whom always were ready with a helping hand, especially ph.d. Jose Manuel Chavez-Garcia and ph.d. Oscar Erlandsson for providing single-shot data. Finally I would like to thank Casper Wied for sparring session.

8 References

[1] Johannes et al. Heinsoo. Rapid high-fidelity multiplexed readout of superconducting qubits. 1 2018.

[2] Frank et al. Arute. Quantum supremacy using a programmable superconducting processor. Nature, 574(7779):505–510, 10 2019.

[3] Philip Krantz and Morten et al. Kjaergaard. A Quantum Engineer's Guide to Superconducting Qubits. 4 2019.

[4] Yvonne Y. et al. Gao. A practical guide for building superconducting quantum devices. PRX Quantum, 2(4):040202, 6 2021.

[5] Jonathan et al. Burnett. Decoherence benchmarking of superconducting qubits. 1 2019.

[6] Steffen et al. Schlör. Correlating decoherence in transmon qubits: Low frequency noise by single fluctuators. 1 2019.

[7] Jens et al. Koch. Charge-insensitive qubit design derived from the Cooper pair box. Physical Review A - Atomic, Molecular, and Optical Physics, 76(4):042319, 10 2007.

[8] Nathan K. Langford. Circuit QED - Lecture Notes. 10 2013.

[9] Malcolm et al. Carroll. Dynamics of superconducting qubit relaxation times. 5 2021.

[10] Benjamin et al. Lienhard. Deep Neural Network Discrimination of Multiplexed Superconducting Qubit States. 2 2021.

[11] Sebastian et al. Krinner. Realizing Repeated Quantum Error Correction in a Distance-Three Surface Code. 12 2021.

[12] Chih-Chung et al. Chang. LIBSVM: A Library for Support Vector Machines. Technical report.

[13] Corinna et al. Cortes. Support-Vector Networks Editor. Technical report, 1995.

[14] Bernhard Schölkopf, Sch¨ Schölkopf, Alex J Smola, Robert C Williamson, and Peter L Bartlett Rsise. New Support Vector Algorithms. Technical report.

[15] Koby et al. Crammer. On the Algorithmic Implementation of Multiclass Kernel-based Vector Machines. Technical report, 2001.

[16] Ting-Fan et al. Wu. Probability Estimates for Multi-class Classification by Pairwise Coupling. Technical report, 2004.

[17] Madeleine et al. Udell. Generalized Low Rank Models. 10 2014.

[18] J. M. et al. Chow. Detecting highly entangled states with a joint qubit readout. Physical Review A - Atomic, Molecular, and Optical Physics, 81(6):062325, 6 2010.