UNIVERSITY OF COPENHAGEN
FACULTY OF SCIENCE

Bachelor Thesis

Quantum State Tomography

A Comparison Between Linear Inversion, Maximum Likelihood Estimation and Bayesian Inference

Casper Wied / nqs117

Supervisors:
Assistent Professor: Morten Kjærgaard
Post doc: Jacob Hastrup

Spring 2022

Abstract

Building a successful universal quantum computer will be a revolutionary breakthrough and have a significant impact on current challenges in the scientific community. There are many problems that need to be solved before this becomes a reality. One of these issues stems from the fact that quantum states collapse when measured, rendering the information limited. Quantum State Tomography aims to solve this problem, which can be implemented using different models. In this thesis three models, Linear Inversion, Maximum Likelihood Estimation and Bayesian Inference, have been numerically implemented in Python along with code to simulate quantum measurements. All 3 models show potential, but each have their advantages and disadvantages. One has to be careful due to the fact that Linear Inversion can return non physical states if the purity of the state is near unity and if the number of measurements are sufficiently low. With Maximum Likelihood Estimation one can risk that the resulting state has eigenvalues equal to zero, in the same parameter extrema as Linear Inversion. Bayesian Inference solves both these issues, but the computational time is significantly higher. All three models have shown their robustness on simulated data and are capable of running at least 4 qubit tomography. In addition, the tomographic models were implemented on superconducting qubits. All 3 Quantum State Tomography models performed well on the data, showing that control of the qubit was achieved. Which demonstrates their functionality on experimental data.

Abbreviations

Contents

1 Introduction

The idea of utilizing the laws of quantum mechanics to create a universal quantum computer has been around for decades. But the field has heavily accelerated the last 5-10 years[1][2][3]. A whole array of unsolved or computationally heavy problems could be solved if a fully functional quantum computer could be designed. Especially in the scientific community, leaps in A.I., machine learning, computational chemistry, drug development and simulation, cybersecurity and cryptography and even weather forecasting, could be made. The world of quantum computing is a fast moving area where different scientific fields interface to solve problems.
Quantum computing aims to improve our ability to solve these problems using superposition and entanglement of quantum objects. The quantum object of choice varies from using light, to ions, to single electrons. The current leading method is by using super conducting qubits, which exploit the superconducting trait of certain metals, when cooled down to near $0K$[4]. What unites all these methods is their need for a robust system to measure, readout and post process the output, to actually extract the information of interest.
Many different sub-fields have spawned to combat different problematic aspects of quantum computing. One such problem comes from the collapse of the quantum state when measured, resulting in only a partial picture of the output being gained. This is where Quantum State Tomography (QST) has been successfully implemented[5][6][7][8][9]. One generates and measures the same quantum state several times in different projected planes, whereafter the true state is reconstructed from said data. It is comparable to reconstructing a 3 dimensional object using only the shadows cast in the 3 dimensions. This picture is for 1 qubit, but if a state is comprised of n qubits, then 4^n dimensions are needed.
This is an exponentially increasing problem, and sophisticated computational tools are being implemented. Within the field of QST several different mathematical methods are employed, three of which will be covered in this thesis. Linear Inversion (LI) Tomography, Maximum Likelihood Estimation (MLE) Tomography and Bayesian Inference (BI) Tomography. All three have been numerically implemented in Python. Their flaws will be characterized and their fidelity compared. Moreover, a script for simulating measurements on quantum states has been developed and is used as input for these models. Lastly the models will be tested on experimental data, measured on the commercial 5 qubit superconducting Soprano device from Quantware[10].

2 Mathematical Concepts

2.1 Density Matrices

A classical bit only assumes one of two distinct states (0 or 1), whereas in quantum computing the quantum bit is a two level quantum system, which allows for a superposition of the two states[1][11]. The qubit is represented by a unit vector:

$$|\psi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle \tag{2.1}$$

Where the states $|0\rangle$ and $|1\rangle$ are defined as

$$|0\rangle = \begin{bmatrix}1\\0\end{bmatrix} \text{ and } |1\rangle = \begin{bmatrix}0\\1\end{bmatrix} \tag{2.2}$$

Here α and β are the probability amplitudes of the two parts of the superposition, where $|\alpha|^2 + |\beta|^2 = 1$[8][9].

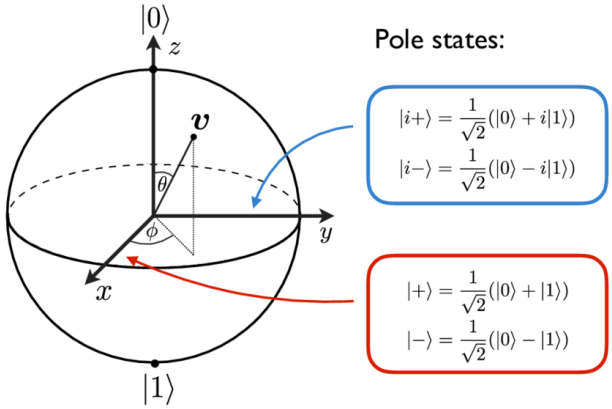


Figure 2.1: The Bloch sphere. State vector represented by v. The computational basis z, and the two additional basis x and y, span the Hilbert space, which houses the quantum state. Figure from [12].

Qubits can be represented on the Bloch sphere, which is a two dimensional complex Hilbert space, visualized as a 3d sphere. Where the state, $|\psi\rangle$, is described as

$$|\psi\rangle = \cos(\theta/2)\,|0\rangle + e^{i\phi}\sin(\theta/2)\,|1\rangle \tag{2.3}$$

where θ is the polar angle and ϕ is the azimuthal angle. A density matrix is a generalized description of the state vector, which only can describe pure states, whereas a density matrix can describe both pure and mixed states[13] (It is important to note that a mixed state is not the same as a superposition)[14]. A density matrix is a positive semi definite Hermitian $d \times d$ matrix with unit trace. The set of density matrices are defined by four rules

$$S = \{\rho \,\epsilon\, \mathbb{C}^{d\times d},\ \rho = \rho^\dagger,\ tr(\rho) = 1,\ \rho \preceq 0\} \tag{2.4}$$

where $\mathbb{C}^{d\times d}$ is the complex space of size $d \times d$, $\rho^\dagger$ is the conjugate transpose of the density matrix, $tr(\rho)$ is the trace of the density matrix and the positive semi definite property is shown by $\rho \preceq 0$[9]. These rules need to be upheld if the density matrix is physical. Otherwise the state will not be limited to the Hilbert space which describes it. The density matrix, ρ, can be described as an ensemble of pure states $|\psi_i\rangle$ each prepared with a probability of p_i.

$$\rho = \sum p_i\,|\psi_i\rangle\,\langle\psi_i| \tag{2.5}$$

Here $|\psi_i\rangle\,\langle\psi_i|$ is the inner product of the pure state. When the density matrix is described in terms of the X,Y and Z basis, as seen in figure 2.1, a convenient way to reparameterize the density matrix is

$$\rho = \frac{1}{2}(I + a_x\sigma_x + a_y\sigma_y + a_z\sigma_z) \tag{2.6}$$

Where a_x, a_y and a_z are some values and the Pauli matrices are defined as

$$\sigma_x = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix},\ \sigma_y = \begin{bmatrix} 0 & -i \\ i & 0 \end{bmatrix},\ \sigma_z = \begin{bmatrix} 1 & 0 \\ 0 & -1 \end{bmatrix} \tag{2.7}$$

For n qubits this can be generalized to

$$\rho = \frac{1}{2^n}\vec{a}\cdot\vec{P} \tag{2.8}$$

Where $\overrightarrow{a}$ is a vector of values and $\overrightarrow{P}$ is a vector of Pauli operators. Both of these vectors hold 4^n elements, where for $\overrightarrow{a}$ they correspond to permutations between the a-values of each qubits three basis + identity and for $\overrightarrow{P}$ corresponds to permutations between the Pauli matrices + identity. As mentioned there exist both pure and mixed states. The density matrix of a pure state can be described by a single inner product of a state vector and is represented on the Bloch sphere as a vector stretching to the shell of the sphere. Whereas the mixed state is a statistical ensemble of possible outcomes, represented on the Bloch sphere as a vector inside the sphere[15]. Moreover the purity of a state is defined as

$$purity = tr(\rho^2) \tag{2.9}$$

where trace is the sum of the diagonal matrix elements. For a pure state the purity must be equal to one. A purity lower than unity represents a mixed state, and the lower the purity, the more mixed the state is. The purity of a maximally mixed state is

$$tr(\rho_{mix}^2) = \frac{1}{2^n} = \frac{1}{d} \tag{2.10}$$

where n is the number of qubits and d the dimensions of the Hilbert space[9][8].

2.2 Measurements

When one wants to measure a quantum state, one measures the density matrix. To do this one uses an observable, which is a Hermitian $d \times d$ matrix related to a discrete random variable within the system. Lets call the variable O and the possible values are represented by the eigenvalues of the observable O; $\{o_1, o_2, ..., o_d\}$[16]. The expected value of O, for a system ρ is defined by

$$\langle O \rangle = tr(O\rho) \tag{2.11}$$

Since O is diagonalizable it can be spectrally decomposed

$$O = \sum_{i=1}^{d} o_i E_i = \sum_{i=1}^{d} o_i \left|u_i\right\rangle \left\langle u_i\right| \tag{2.12}$$

where $E_i = \left|u_i\right\rangle \left\langle u_i\right|$ is a positive semi-definite Hermitian matrix, which is as a projector into eigenspace corresponding to o_i, where $\left|u_i\right\rangle$ are its eigenvectors. This allows for eq 2.12 to be substituted into eq 2.11, giving rise to

$$\langle O \rangle = \sum_{i=1}^{d} o_i tr(\rho E_i) = \sum_{i=1}^{d} o_i p_i \tag{2.13}$$

Where $tr(\rho E_i)$ is equal p_i, which is the probability of o_i occurring when ρ is measured. The components E_i are a part of a complete set of Positive Operator-Values Measure (POVM). The sum of the POVM elements must equal unity.

$$\sum_{i=1}^{N} E_i = I \tag{2.14}$$

It is worth noting that measurement of a single observable is not enough to completely determine the density matrix. In fact at least $d^2 - 1 = 4^n - 1$ observables are needed (-1 because the identity is trivial), where d is the dimensions of the state and n is the number of

qubits comprising the state. Also one needs to be able to generate the same quantum state consistently due to Heisenberg's uncertainty principle not allowing for two non commuting observables to be measured at the same time. Another caveat is the fact that we can only measure $\{0, 1\}$, which usually is remapped to $\{1, -1\}$. Meaning that even if the expectation value of an observable is 0, a measurement will return -1 or 1. To solve this, one also needs to prepare the same state n times and measure the same observable every time to statistically determine the expectation value[9][8]. In eq 2.6 & and 2.8 the observables of choice are the Pauli matrices described in eq 2.7. With the 6 POVM elements being the inner product of the respective positive or negative basis vectors of the three cardinal basis $(|0\rangle, |1\rangle, |+\rangle, |-\rangle |i+\rangle, |-i+\rangle)$. These are the observables that will be used on wards.

2.3 Expectation Values

The number of observables that need to be measured can be reduced from $4^n - 1$ to 3^n. The example below is for a 2 qubit state, but can be expanded to n qubits. A frequency of each of the $2^n = 4$ outcomes can be calculated if one was to measure the quantum system n times. This can be done for each observable, which in this case is every permutation between the X,Y and Z basis of the two qubits, resulting in $3^n = 9$ observables (XX,YY,ZZ,XY,YX,XZ,ZX,YZ,ZY), rather than $4^n - 1$.[16]. From these frequencies a frequency vector can be constructed for each observable resulting in $3^n = 9$ frequency vectors each of length $2^n = 4$.

$$\overrightarrow{f} = \begin{bmatrix} f_1 \\ f_2 \\ \vdots \\ f_{2^n} \end{bmatrix} \tag{2.15}$$

Where e.g the XX measurement would be.

$$\overrightarrow{f_{XX}} = \begin{bmatrix} f_{00} \\ f_{01} \\ f_{10} \\ f_{11} \end{bmatrix} \tag{2.16}$$

Where $\{00, 01, 10, 11\}$ are the four binary outcomes of said measurement in the XX basis. To estimate the expectation values of a state in each measurement basis, one can use the following matrix relation.

$$\overrightarrow{f} = \frac{1}{2^n} \cdot H_{2^n} \cdot \overrightarrow{\langle O \rangle} \tag{2.17}$$

Where H_{2^n} is the 2^n-rank Hadamard matrix.

$$H_2 = \begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}, \quad H_{2^n} = H_2^{\bigotimes n} \tag{2.18}$$

Which in the XX measurement case would be

$$\begin{bmatrix} f_{00} \\ f_{01} \\ f_{10} \\ f_{11} \end{bmatrix} = \frac{1}{4} \cdot \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & -1 & 1 & -1 \\ 1 & 1 & -1 & -1 \\ 1 & -1 & -1 & 1 \end{bmatrix} \cdot \begin{bmatrix} \langle II \rangle \\ \langle IX \rangle \\ \langle XI \rangle \\ \langle XX \rangle \end{bmatrix} \tag{2.19}$$

Only only needs to invert the Hadamard matrix (and normalization) to calculate the expectation values. Showing that from 3^n measurements, all 4^n (including identity) expectation

values can be calculated. In fact this method results in $3^n \cdot 2^n = 6^n$ expectation values including duplicates, which can be averaged to reduce the uncertainty on the measurements.

3 Quantum State Tomography

3.1 Data Simulation

A script for simulating quantum measurements was implemented to generate data for the QST models. It is based on section 2.2 and 2.3. The projectors of the three cardinal basis (X, Y and Z) of each qubit is used as the POVM set to simulate all data used. But it is possible to use any arbitrary POVM set in principle. When more than one qubit is simulated, the POVM set is expanded into a tensor product between every permutation between the set of qubits respective projectors. When simulating data it is possible to customize a variety of parameters including, number of qubits and shots, probability amplitudes, purity, measuring basis and whether to entangle the system[17].

3.1.1 Numerical implementation

The data simulation code is based on the QuTiP Python package, which defines quantum states as its own object type[18]. The data simulation class is comprised of many functions, mainly split in two categories. Generating the random/specific quantum state, and measuring said quantum state. The most central function in this case is the one that actually measures the generated state.

```
def measure_qubit(self):
    # Calculate the probability distributions
    prob_dict, tens_mat = self.calc_prob_nsite()
    # Create all possible outcomes
    row = list(it.product([-1,1], repeat=self.nqubits))
    idx = np.arange(len(row))
    expect_list = []
    count_list = []
    # Loop through all sets of probabilties and choose between
    # the different outcomes with the given probabilities
    if self.total_shot == False:
        pass
    elif self.total_shot == True:
        self.shots = int(self.shots/(3**self.nqubits))
    for i, prob in enumerate(prob_dict.values()):
        # Measure for n shots
        measure = list(np.random.choice(len(row), self.shots, p=
   prob))
        # Count the frequency of each outcome and find the measured
    probability
        counts = np.array([measure.count(x)/self.shots for x in idx
   ])
        # Use this to calculate all expectation values
        expect1 = (self.hadamard*counts).real
```

Listing 1: Qubit measurement

First the function *calc_prob_nsite*() is called which initializes the chosen basis and, using a QuTiP function generates the probability distribution of outcomes of the generated state in the chosen basis. These probability distributions are in turn used to actually simulate the measurement process, which in the process also generates the random measurement "noise". This is done by using the *np.random.choice*() function, where the options for each measurement basis are the 2^n possible outcomes, chosen with its respective probabilities. The frequency of each outcome is calculated and converted into a probability, which can be used as seen in eq 2.17 to calculate the measured expectation values[17].

3.2 Linear Inversion

LI is the most basic technique used to reconstruct the density matrix, using the concepts described in sections 2.1, 2.2 and 2.3. It can be derived from Borns rule.

$$\langle O_i \rangle = tr(O_i \rho) \tag{3.20}$$

Where $\langle O_i \rangle$ is the expectation value of some observable. The equation can, according to eq 2.8, be rewritten as

$$\langle O_i \rangle = tr(O_i \rho) = tr(\overrightarrow{a_i} O_i \overrightarrow{P_i}) \tag{3.21}$$

Where $\overrightarrow{a_i}$ is a vector with values related to the density matrix, O_i is the observable and $\overrightarrow{P_i}$ is the corresponding Pauli matrix vector. This can be generalized for all 4^n expectation values.

$$\begin{bmatrix} \langle O_0 \rangle \\ \langle O_1 \rangle \\ \vdots \\ \langle O_{4^n} \rangle \end{bmatrix} = \overrightarrow{M} \cdot \overrightarrow{a} \tag{3.22}$$

Where $\overrightarrow{M}$ is a $4^n \times 4^n$ matrix, where every component is defined as

$$M_{ij} = tr(O_i P_j) \tag{3.23}$$

Which shows that LI can be used to calculate the a values and in turn the density matrix using any tomographically complete set of observables. The convenient choice is choosing the basis as the X,Y and Z basis, where the observables correspond to the Pauli matrices described in eq 2.7. This results in the matrix elements being defined by the Kronecker delta

$$M_{ij} = tr(P_i P_j) = 2^n \delta_{ij} \tag{3.24}$$

which results in a $\overrightarrow{M}$ being equal to a $4^n \times 4^n$ identity matrix, allowing for a direct conversion between the a values and the expectation values, making for easy calculation of the density matrix described in eq 2.8.

But LI has a severe disadvantage, due to the fact that it can output nonphysical quantum states with negative eigenvalues, by breaking condition 3 in eq 2.4, stating that the trace of the density matrix is equal to 1. This in turn breaks condition 1, due to the fact that the density matrix no longer will be restrained in the $\mathbb{C}^{d \times d}$ Hilbert space. This happens due to the random nature of measuring[7].

3.2.1 Numerical Implementation

The implementation of LI Tomography is mostly based on one quite simple function.

```
def get_lin_inv_rho(self):
    # np.einsum is used to calculate the dot product as fast as
   possible.
    rho = (1/(2**self.nqubits))*np.einsum('i,ijk->jk',np.array(self
   .expectation),self.paulis)
    rho = Qobj(rho, dims=[[2]*self.nqubits,[2]*self.nqubits])
    return rho
```

Listing 2: Linear Inversion Function

The density matrix is calculated by multiplying the dot product between the expectation values and the corresponding POVM (the basis are just the x,y and z basis in this case) with a normalization factor that is defined by the number of qubits[17].

3.3 Maximum Likelihood Estimation

The more sophisticated alternative to LI is QST using MLE. This method eliminates the negative eigenvalue problem. The approach uses the likelihood function

$$\mathcal{L}(\rho) = Pr(D|\rho) \tag{3.25}$$

which describes the probability of the measured data, given some density matrix. The density matrix which results in a maximized likelihood function, should be a good estimate of the true density matrix.

$$\rho = \max(\mathcal{L}(\rho)) \tag{3.26}$$

In this case the measured data comes in the form of expectation values of a set of observables as described in section 2.2 and 2.3. In practise it is easier to minimize the negative log likelihood function

$$F(\rho) = -log(\mathcal{L}(\rho)),\ \rho = \min(F(\rho)) \tag{3.27}$$

Where $F(\rho)$ for 1 qubit is

$$F(\rho) = \sum_{i=x,y,z} (\langle\ O_i\rangle - tr(\rho\sigma_i))^2 \tag{3.28}$$

Where $\langle\ O_i\rangle$ is the measured expectation values in the three cardinal basis and $tr(\rho\sigma_i)$ is the trace of the matrix product between the current guess of density matrix and the Pauli matrices[5][6][19]. This function is for one qubit, but for n qubits i would be every permutation of the three cardinal basis between every qubit. The parameters that the MLE tunes stepwise are not directly the values of the density matrix itself. This is due to the fact that the density matrix would quickly break the rules set in eq 2.4. Instead Cholesky decomposition[20] is used to reparameterize the density matrix

$$\rho = \frac{T(t)T(t)^\dagger}{Tr(T(t)T(t)^\dagger)} \tag{3.29}$$

Where the T matrix is a lower triangular matrix with d^2 t components

$$T = \begin{bmatrix} t_1 & t_{d+1} + it_{d+2} & \dots & \dots & t_{d^2-1} + it_{d^2} \\ 0 & t_2 & t_{d+3} + it_{d+4} & \ddots & \vdots \\ 0 & 0 & \ddots & \ddots & \vdots \\ \vdots & \ddots & \ddots & \ddots & t_{3d-3} + it_{3d-2} \\ 0 & \dots & \dots & \dots & t_d \end{bmatrix} \tag{3.30}$$

which is what the MLE algorithm will tune. This parameterization will ensure positivity and maintain unit trace, which mitigate the problem LI poses[9].

But this approach introduces a new problem in the form of eigenvalues equal to zero which reflect outcomes with probability zero. This is not non physical, but experimentally speaking this is problematic. When measuring, a finite number of measurements are done for every observable and the expectation value will trend towards the "true" expectation values. Meaning that it is wrong to exclude certain outcomes by reducing eigenvalues to zero. [5][6].

3.3.1 Numerical Implementation

The implementation of QST using MLE is centered around the likelihood function as described above. The function is minimized using SciPy.

```
def MLE_Function_QST(t_tunable, measurements, Paulis):
    rho = op_cholesky(t_tunable)
    expect = np.einsum('ij,ljk',rho,Paulis)  # perform matrix
   multiplication over array of paulis
    tr_expect = np.einsum('iij', expect).real # take the real
   component of the trace
    L = np.sum((measurements-tr_expect)**2)
    return L
```

Listing 3: Likelihood Function - MLE

For every step of the minimization the current t matrix guess, is converted into a density matrix using Cholesky Decomposition[20] and matrix multiplied with the set of Pauli matrices. Afterwards, the trace is calculated, simulating a measurement. The trace is always real, but Python will keep +0i, which is why the real component is used. The difference between this value and the "true" measured value is then calculated, and summed for all measured basis. The L value is calculated, the t matrix is tuned and the process is repeated. When the L value decreases, the corresponding density matrix estimate approaches the true density matrix.[17].

3.4 Bayesian Inference

BI is a method which can be implemented with a wide variety of techniques. This method eliminates the respective problems of the two prior methods, meaning no negative eigenvalues and no zero eigenvalues[5][6]. In this particular implementation of BI the same reparameterization as in eq 3.29 & 3.30 is used, except another layer of reparameterization is done,

reducing the number of parameters from d^2 to $d^2 - 1$

$$\begin{aligned} t_1 &= \sin\theta_{d^2-1} \sin\theta_{d^2-2} ... \sin\theta_2 \sin\theta_1 \\ t_2 &= \sin\theta_{d^2-1} \sin\theta_{d^2-2} ... \sin\theta_2 \cos\theta_1 \\ &\vdots \\ t_{d^2-1} &= \sin\theta_{d^2-1} \cos\theta_{d^2-2} \\ t_{d^2} &= \cos\theta_{d^2-1} \end{aligned} \tag{3.31}$$

This can be condensed into

$$t_i = \cos\theta_{i-1} \prod_{j=i}^{d^2} \sin\theta_j \quad \text{for } i = 1, 2, \ldots, d^2 \tag{3.32}$$

where the set of theta values are constricted in the following parameter space

$$\Theta = \{0 < \theta_i < \pi/2, \text{ for } i = 1, 2, \ldots, d-1, \text{ and } 0 < \theta_i < \pi, \text{ for } d \leq i \leq d^2 - 1\} \tag{3.33}$$

This allows for easier implementation of BI, with the use of Markov Chain Monte Carlo (MCMC) and the Metropolis-Hastings Sampling (MHS) Algorithm[9]. The MHS algorithm is what samples the new set of theta values at every step of the MCMC.
This is done by sampling every theta value simultaneously and independently from truncated normal distributions, centered around the previous theta values.

$$\theta_i^* | \theta_i^t \sim TN_{[a_i, b_i](\theta_i^t, \Delta_i^2)} \tag{3.34}$$

The truncation happens at the boundaries described by eq 3.32.

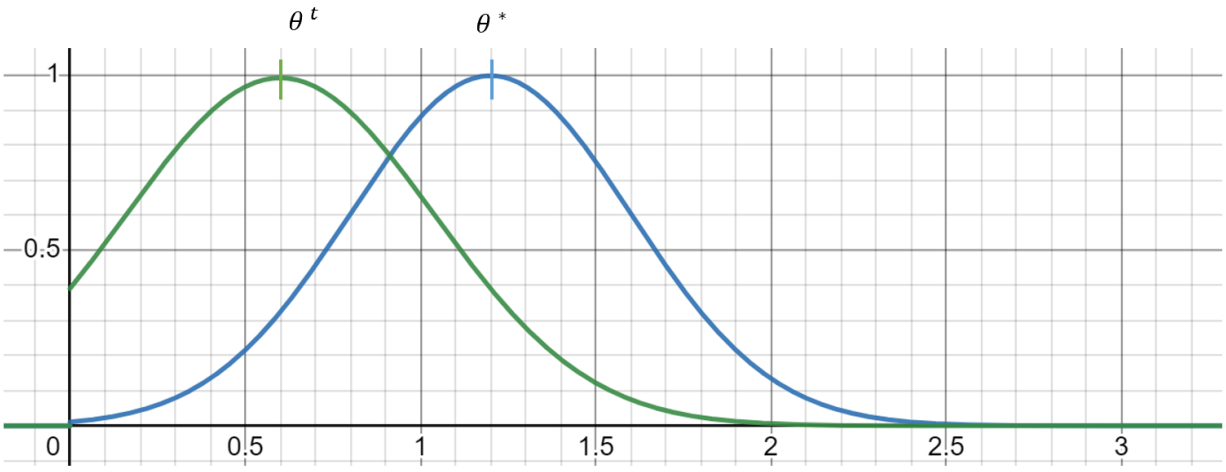


Figure 3.1: Current theta distribution (green), centered around θ^t. Proposal theta distribution (blue), centered around θ^* which was sampled from the current distribution.

The sampling can be seen (in one dimension) in fig 3.1, where θ^t is the current theta value and θ^* is the sampled candidate. The candidate is sampled from the truncated normal distribution centered around the previous theta value. Whether the new set of theta values are accepted or rejected as a set is determined by the MCMC algorithm. To implement this algorithm a prior distribution needs to be selected, which should be uniform, or as close as

possible (a Jeffreys prior can also be used). This prior, $\pi(\theta)$, is multiplied with the likelihood function

$$\pi(\theta|D) \propto \frac{N!}{\prod_{i=1}^{M} n_i!} \prod_{j=1}^{M} tr(\rho(\theta)E_j)^{n_j}\pi(\theta) \tag{3.35}$$

which gives the joint posterior function. Or rather, since the joint posterior is not analytically tractable, the result is proportional to the joint posterior. Here N is the number of shots, M is all possible outcomes, n is the number of counts in each outcome and $tr(\rho(\theta)E_j)$ is the probability of the given outcome, where E_j is the j'th POVM element. The fact that the joint posterior is not analytically tractable is why numerical algorithms are needed. In this implementation the ratio between the joint posterior for the newly sampled set of theta values and the previous set of theta values is used,

$$\frac{\pi(\theta^*|D)}{\pi(\theta^t|D)} = \prod_{i=1}^{M} \frac{tr(\rho(\theta^*)E_i)^{n_i}}{tr(\rho(\theta^t)E_i)^{n_i}} \tag{3.36}$$

which eliminates the prefactors and the prior distribution. Next a proposal distribution is needed, which due to sampling from truncated normal distributions is a product of truncated normal distributions,

$$q(\theta^*|\theta^t) = \prod_{i=1}^{d^2-1} \frac{\frac{1}{\Delta_i}\phi(\frac{\theta^*-\theta_i^t}{\Delta_i})}{\Phi(\frac{b_i-\theta_i^t}{\Delta_i}) - \Phi(\frac{a_i-\theta_i^t}{\Delta_i})} \tag{3.37}$$

where the ratio between sampling the new theta set from the current distributions and vice versa is

$$\frac{q(\theta^t|\theta^*)}{q(\theta^*|\theta^t)} = \prod_{i=1}^{d^2-1} \frac{\Phi(\frac{b_i-\theta_i^t}{\Delta_i}) - \Phi(\frac{a_i-\theta_i^t}{\Delta_i})}{\Phi(\frac{b_i-\theta_i^*}{\Delta_i}) - \Phi(\frac{a_i-\theta_i^*}{\Delta_i})} \tag{3.38}$$

After the MHS algorithm has sampled new theta values, the MCMC algorithm accepts the new set, $\theta^{t+1} = \theta^*$, with probability

$$\alpha(\theta^t|\theta^*) = min\left(1, \frac{\pi(\theta^*|D)}{\pi(\theta^t|D)} \frac{q(\theta^t|\theta^*)}{q(\theta^*|\theta^t)}\right) \tag{3.39}$$

otherwise $\theta^{t+1} = \theta^t$[9].

3.4.1 Numerical Implementation

To implement BI a combination of MCMC and the MHS algorithm is used. Only the main function is shown below.

```
def getRho(self):
    # Burn in phase
    thetha, theta_distrib_burn = self.burn_in_phase()
    # Sampling phase
    theta_distrib_samp = self.sampling_phase(thetha)
    rho_distrib = [Qobj(btool.op_cholesky(btool.construct_t(list(i)
   , self.dim))) for i in theta_distrib_samp]
    rho_mean = np.mean(np.array(rho_distrib), axis=0)
    rho = Qobj(mean, dims=[[2]*self.nqubits,[2]*self.nqubits]
    return rho, theta_distrib_burn, theta_distrib_samp
```

Listing 4: Bayesian Inference

First the burn in phase is run. Which samples the initial theta values from a uniform distribution and then runs through a set number of steps in the MCMC, without considering any of the previous steps. After this step the chains should have converged to a stable parameter space, whereafter the sampling phase is initiated. Now the sampling phase is identical to the burn in phase, except all the accepted sets of theta values are saved, and a hyper dimensional theta distribution is created. Now from this distribution the mean is found, which corresponds to the best estimate for the true density matrix[17].

4 Results & Discussion

4.1 Linear Inversion - Negative eigenvalues

As mentioned earlier, the main problem with LI is the fact that its output state can be nonphysical, due to eigenvalues of zero. The likelihood of the output being nonphysical when increasing purity, for 1-4 qubits, was studied with numerically generated data to further examine this phenomenon.

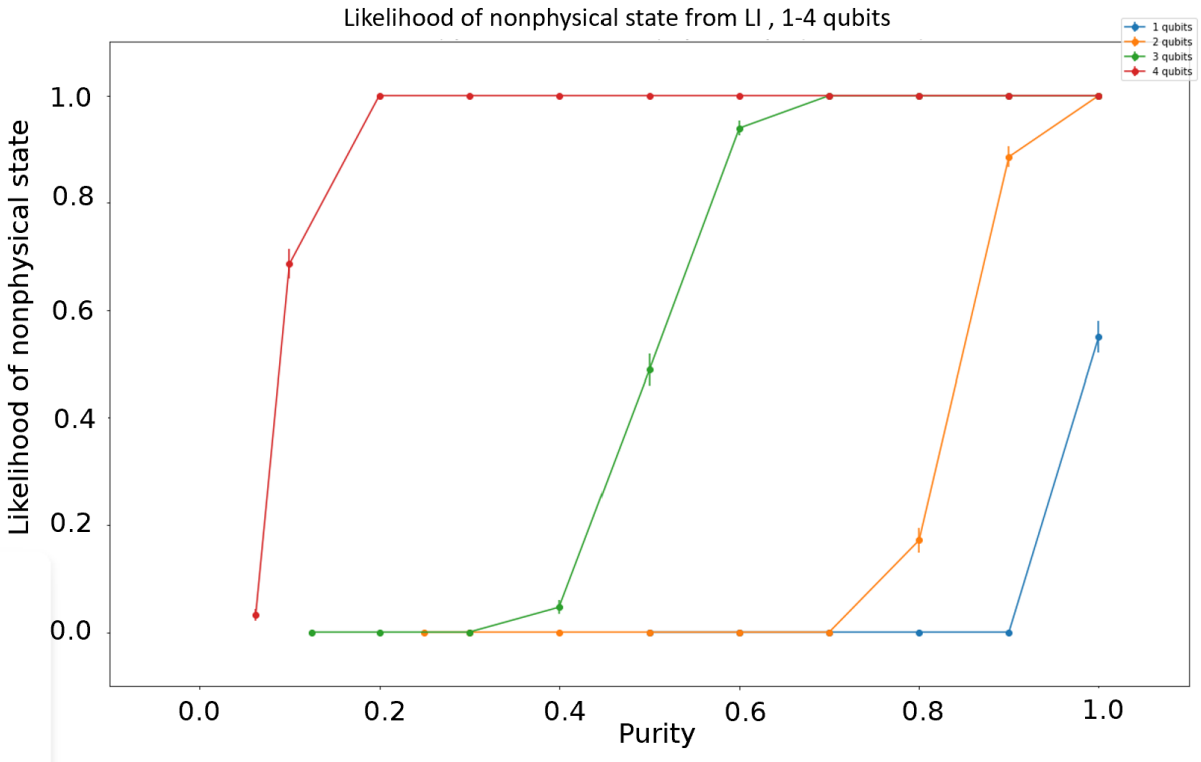
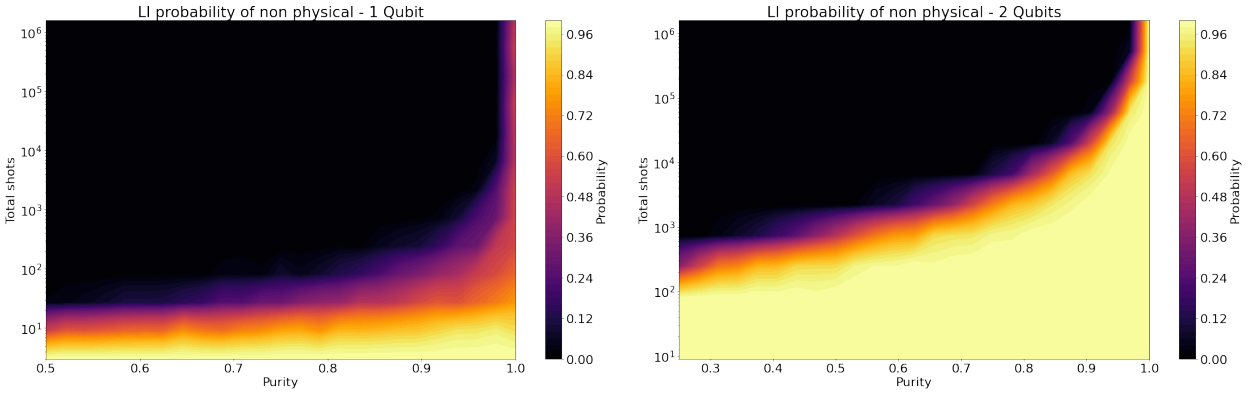
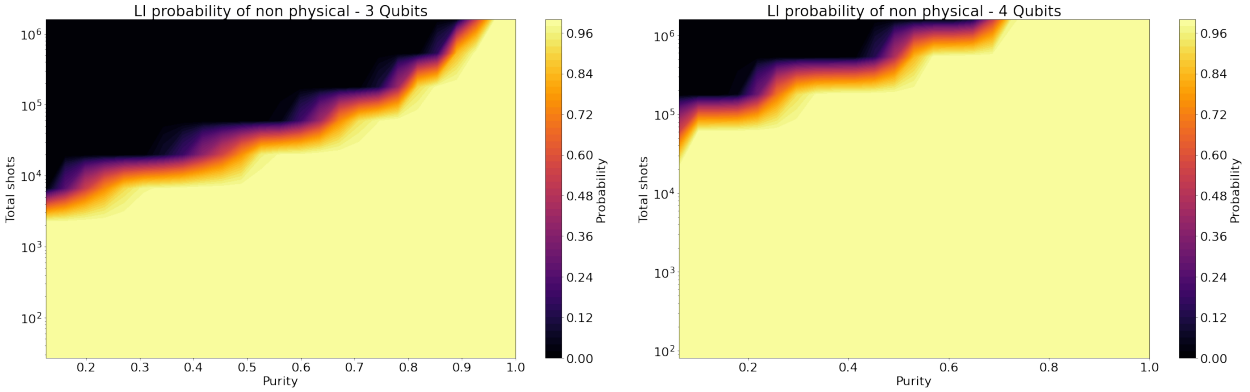


Figure 4.1: Likelihood of unphysical result as function of purity for 1-4 qubits.

For each point 280 random states where generated to ensure that the uncertainty is sufficiently low. Each state was measured 1000 times in each basis. LI Tomography was used and the eigenvalues where calculated. The trend in figure 4.1 seems to be that there is a correlation between high purity of the generated state and a higher likelihood of the LI output being non physical. Moreover when increasing the number of qubits, the increase happens at a lower purity. See appendix figure A.1 for higher resolution of 1 and 2 qubits. To further investigate the parameter space, 2D sweeps of the purity vs the total amount of measurements (shots) was done for 1-4 qubits respectively.



(a) 1 qubit **(b)** 2 qubits



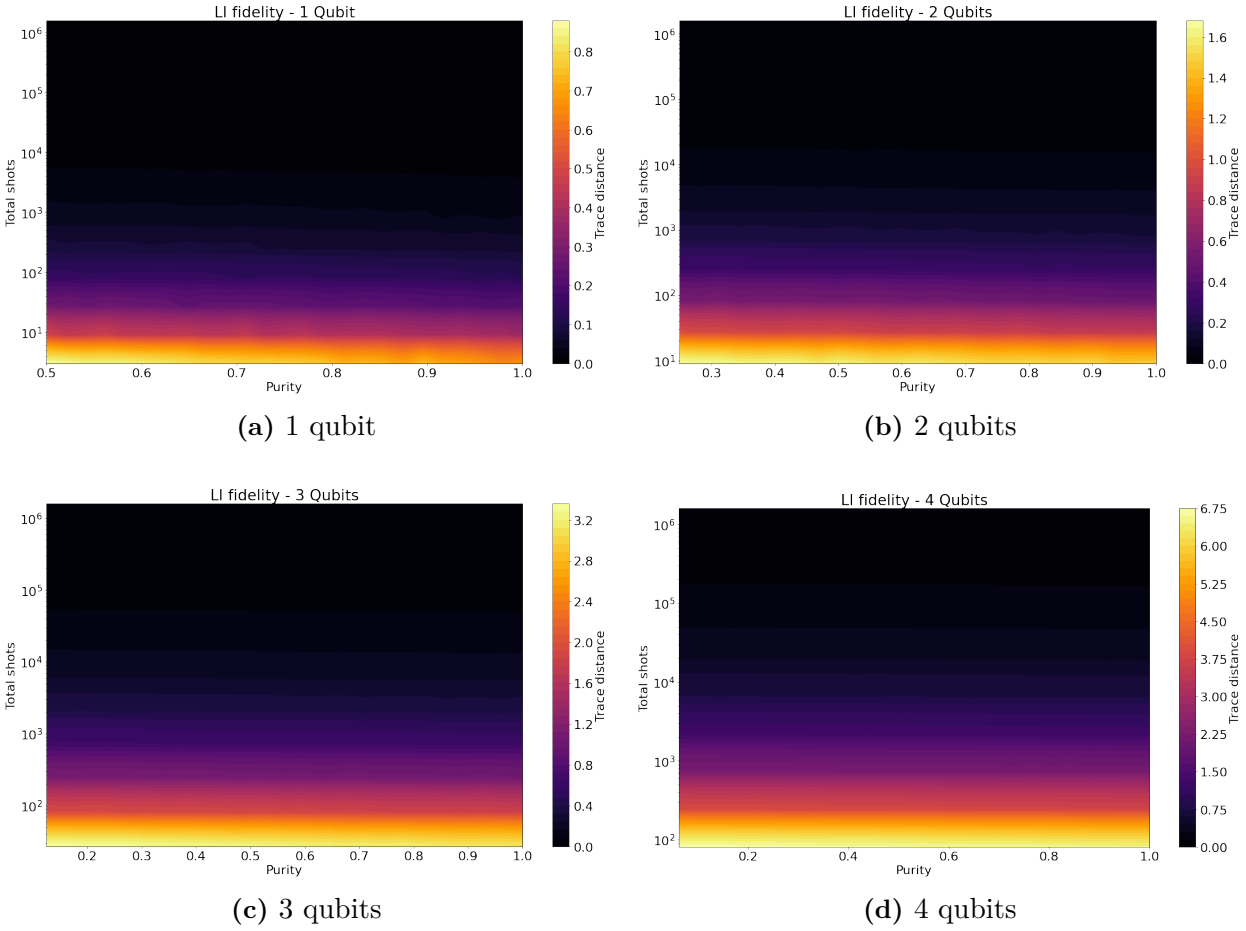
(c) 3 qubits **(d)** 4 qubits

Figure 4.2: Probability of LI Tomography returning a non physical state on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.

Figure 4.2a-4.2d shows increasing purity on the x axis and increasing shots on the y axis, while the color map indicates the likelihood of the state being non physical. Again each point is an average of 280 randomly generated states. The shots are distributed evenly on all the required measurements which scales as 3^n. Meaning that for 1 qubit, each basis measurement uses 1/3 of the shots, whilst for 4 qubits it is 1/81 shots pr basis measurement. Moreover the minimum purity scales as $1/2^n$ as in eq 2.10. There is a clear correlation between few shots and high purity, and a high likelihood of a non physical state. This effect is exacerbated when more qubits are added. Altough, especially with 1 and 2 qubits, there seems to be a quite large parameter space, where there is a near zero chance of the output being nonphysical, indicating that LI should not instantaneously be disregarded, due to its eigenvalue flaw.

4.2 Linear Inversion - Fidelity

Now that it has been established that LI is a viable QST option, a investigation of its fidelity would be needed.



(a) 1 qubit

(b) 2 qubits

(c) 3 qubits

(d) 4 qubits

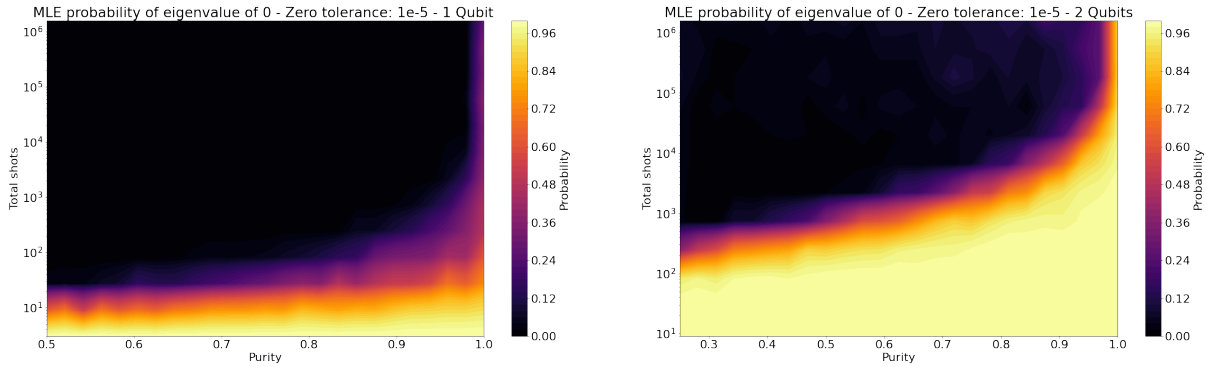
Figure 4.3: Trace distance between the generated state, and the state estimated by LI tomography is shown on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.

Figure 4.3 has the same axis as figure 4.2, except the color map indicates fidelity in the form of trace distance from the output state from the LI tomography, to the true state generated by the data simulation. The figures show that the purity of the state does not impact the models ability to estimate the output very much. In fact the trace distance seems to improve slightly at higher purity. However, almost solely the number of shots that each basis was measured is responsible for the final trace distance. Moreover, (at least for 1-4 qubits) anything above 10^4 shots seems to be enough to get a near zero trace distance. Although the chosen parameters should be compared to the negative eigenvalue figure to determine whether the outcome is both correct and physical. It should also be mentioned that the maximum trace distance in the Hilbert space is 1. But due to the non physical output states generated from LI being outside the Hilbert space, trace distances above 1 can be seen.

4.3 Maximum Likelihood Estimation - Zero eigenvalues

The MLE model eliminates the issue of output states having negative eigenvalues, but instead introduces the problem of ”polished” data, meaning eigenvalues equal to zero. This is not as problematic as having non physical output states, but it still does not reflect the reality of quantum states in a experimental context[6]. To determine the scale of this issue a similar experiment to section 4.1 was conducted. Only experiments for 1, 2 and 3 qubits

were done, due to the data collection for these figures being quite time consuming, but MLE tomography can be done on at least 4 qubits.



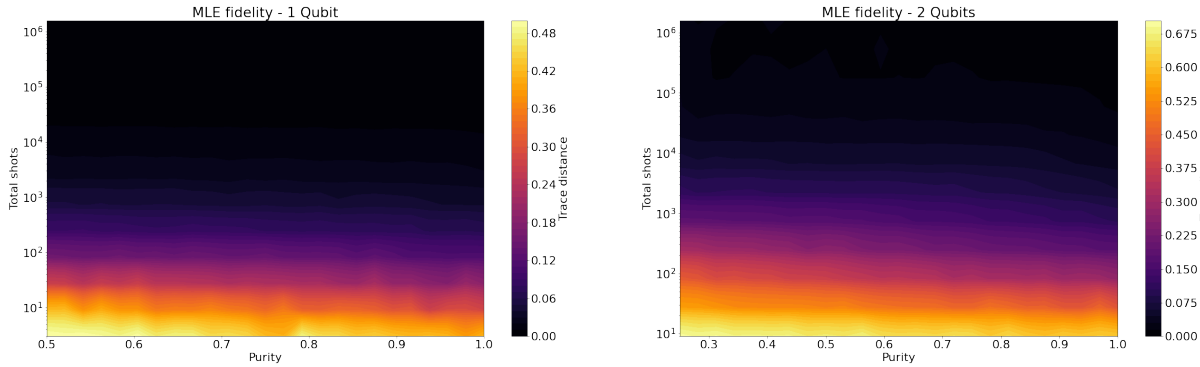
(a) 1 qubit **(b)** 2 qubit

Figure 4.4: Probability of MLE returning a state with one or more eigenvalues equal to zero is shown on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.

The x and y axis are the same, but this time the color map reflects the probability of the output state having one or more eigenvalues equal to zero. Now python does not exactly calculate eigenvalues of zero, but they are close. Due to this a tolerance of $\epsilon = 10^{-5}$ was set, meaning that any eigenvalue below the tolerance was set to zero. In appendix A.4 the same data, but with smaller tolerance values of 10^{-10}&10^{-15}, can be seen. But this introduced artifacting in the data. In figure 4.4 similar trends can be seen as in figure 4.2, in the form of a higher probability at higher purity and lower shots. If one increases the number of shots adequately the problem of eigenvalues of zero seems to be avoidable. Figures for 3 qubits can be seen in appendix, figures A.2, A.4e & A.4f.

4.4 Maximum Likelihood Estimation - Fidelity

The fidelity of MLE was investigated in a similar fashion to the LI model.



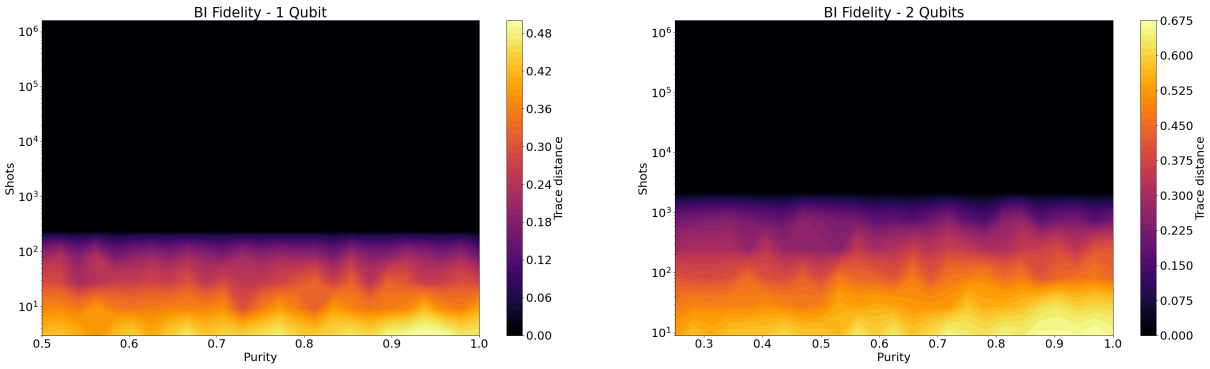
(a) 1 qubit **(b)** 2 qubits

Figure 4.5: Trace distance between the generated state, and the state estimated by MLE tomography is shown on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.

Again the x and y axis are the same, and the color map shows the fidelity in form of trace distance between the MLE generated state and the true state, generated in the data simulation step. The maximum trace distance seen is 0.5 and 0.675 respectively for 1 and 2 qubits. Which in comparison to the LI model is quite an improvement. See appendix figure A.3 for 3 qubits.

4.5 Bayesian Inference - Fidelity

The fidelity of BI Tomography was investigated in a similar fashion as LI and MLE.



(a) 1 qubit **(b)** 2 qubits

Figure 4.6: Trace distance between the generated state, and the state estimated by BI tomography is shown on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.

The computational time of BI is significantly higher, resulting in only 20 states pr point on the heat map, explaining the artifacts. Which is also why only 1 and 2 qubit experiments have been run. BI Tomography can be run on up to at least 4 qubits. Nonetheless near zero trace distance can be achieved at slightly lower total shots, compared to the two other models. Anything above 10^2 and 10^3 for 1 and 2 qubits respectively, results in near zero trace distance.

4.6 Tomography on Soprano device

QST was applied experimentally on the Soprano device, which is a quantum processor from Quantware[10].

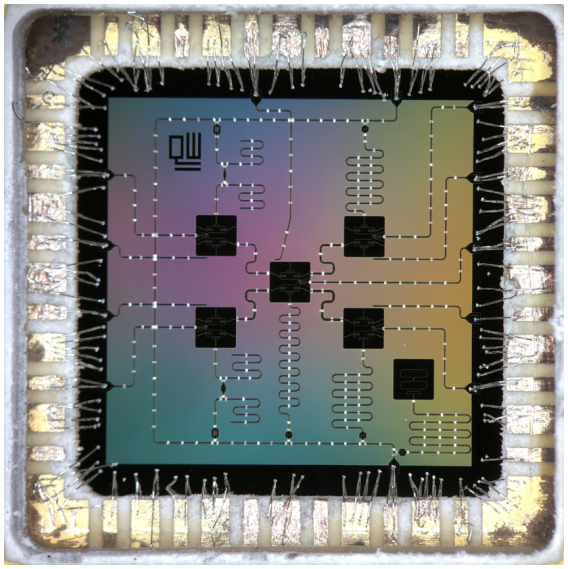


Figure 4.7: Soprano Device in the QCage

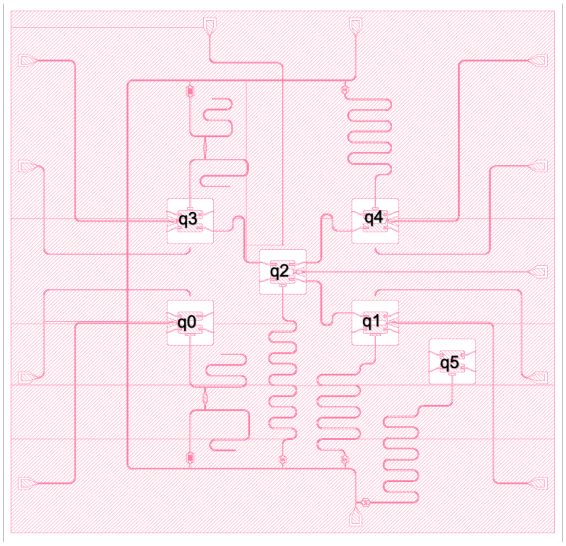


Figure 4.8: Schematic of the Soprano chip.

The chip contains 5 qubits with flux and drive lines allowing for control of both resonator and qubit frequency. Qubit 0, 1, 3 and 4 are all coupled to qubit 2, allowing for the possibility of two qubit gates. Additionally, there is an auxiliary qubit (q5) with only a flux line. When working with experimental readout data, the ground state, $|0\rangle$, and the excited state, $|1\rangle$ need to be defined and separated, before tomography can be done.

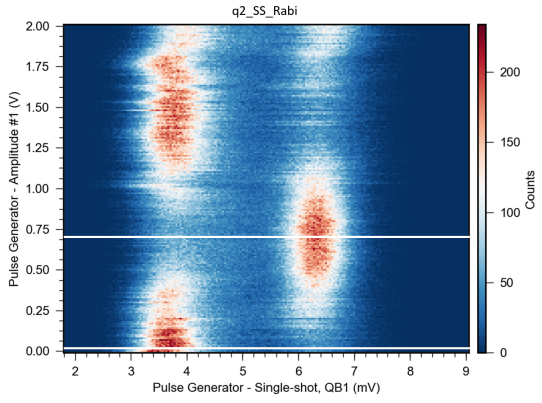


Figure 4.9: Qubit 2 - Rabi oscillation. Drive pulse amplitude on y-axis. High concentration of counts represent current state.

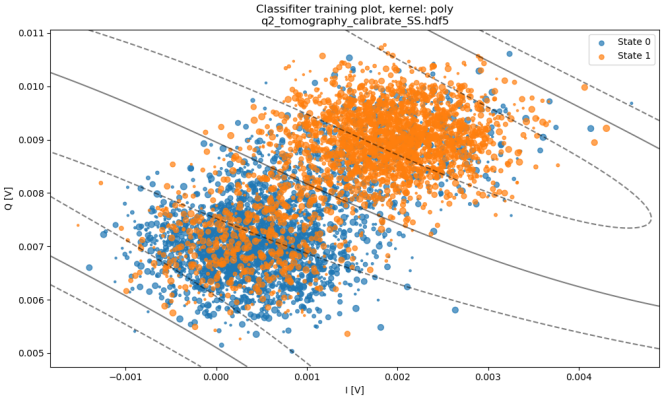


Figure 4.10: Qubit 2 - Classifier for state separation. In phase component on x-axis, out of phase component on y-axis.

In figure 4.9 qubit 2 was driven with a microwave signal at 5.0289 GHz, corresponding to the qubits frequency. The pulse amplitude is along the y-axis. The ground state is defined where the wave amplitude is $\sim 0V$, due to the qubit not being driven. The excited state is defined at $\sim 0.74V$ corresponding to the large concentration of counts on the right. The slices, marked by the white line, can then be used to create a classifier using the 'quantum fitter readout tools' package[21]. Resulting in figure 4.10, where the centered black line is

the separator for future measurements on this qubit in both the x, y and z basis. With the classifier calibrated, a custom sequence was designed for the measuring software Labber. The sequence has operations for 31 different states that will drive the qubit in a spiral from the ground state to the excited state. Each state was measured 15000 times total, divided between the three basis measurements.

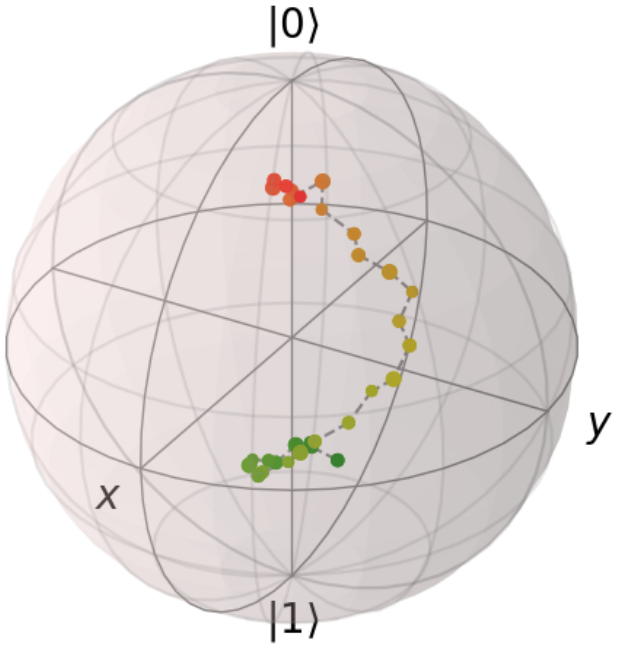
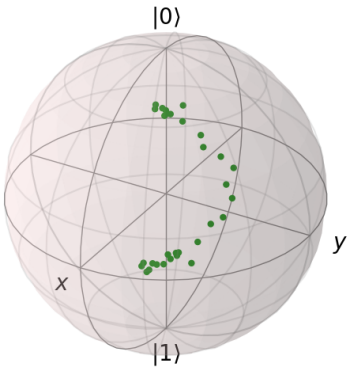
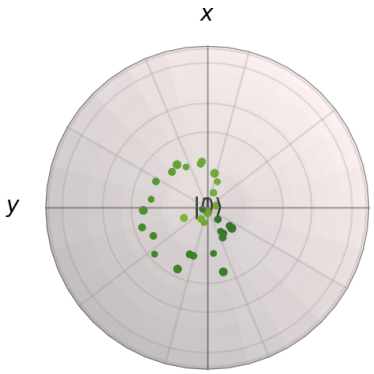


Figure 4.11: Bloch sphere with 31 measured states from qubit 2 on the Soprano device. Tomography done with LI.

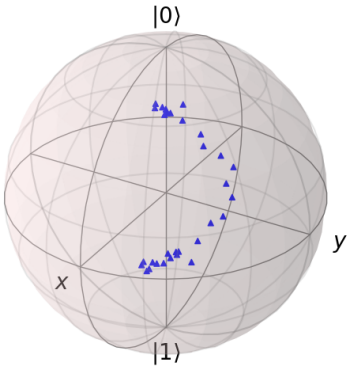
In figure 4.11 the resulting sequence of states can be seen. The measured values were defined as zero or one by the classifier, for each basis, and the resulting expectation values where used for QST using LI. The time progression is defined from the color red to the color green. The figure shows a clear spiral descent from the ground state to the excited state. Now if everything was done perfectly the states would be on the sphere surface, and not pushed into the sphere, indicating that the states are not pure. The average purity can be calculated according to eq 2.9. Resulting in the average purity of the generated states being 0.612. This can occur due to a multitude of factors in the experimental setup, such as a too hot cryostat. Nonetheless the execution of the respective parts of the process seems to have been reasonable. The same data was processed using the LI, MLE and BI models respectively. Figure 4.12 shows each model and all 3 combined on the Bloch sphere. Side view on the left side and top view on the right side. All the models seem to perform equally well. This is due to the fact that with the experiment was run with 15000 measurements, which, as can be seen from figures 4.3a, 4.5a & 4.6a, is more than enough for a near zero trace distance. Moreover, the risk of the LI model giving a non physical state or the MLE model giving a state with eigenvalues of zero, is avoided due to the low purity of the generated states, demonstrating that these models are applicable in a experimental context. See A.5 in the appendix for more examples of tomography done on the Soprano device.



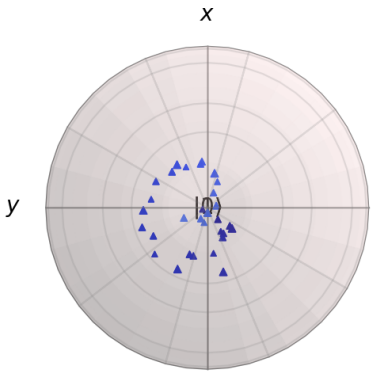
(a) LI - side view



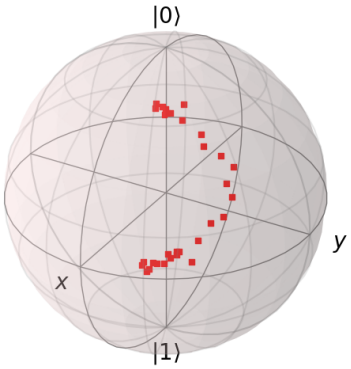
(b) LI - top view



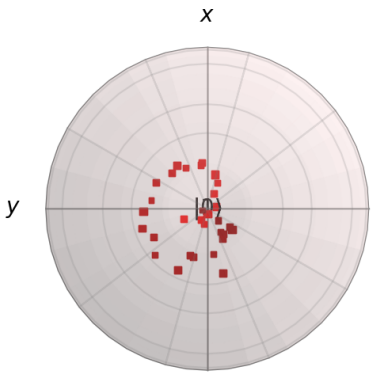
(c) MLE - side view



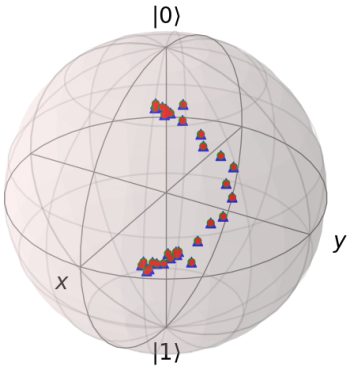
(d) MLE - top view



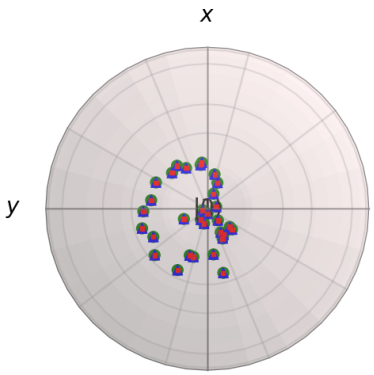
(e) BI - side view



(f) BI - top view



(g) All 3 models - side view



(h) All 3 models - top view

Figure 4.12: Comparison of the three types of tomography on the 31 states measured on qubit 2 of the Soprano device[10].

5 Conclusion

Simulation of quantum state measurements was implemented for n-qubits and wide customizability of the quantum state is possible. QST was successfully implemented in three distinct models; LI, MLE and BI. All three models can do tomography on at least 4 qubits relatively efficient[17]. The inherent problems with LI and MLE where characterized and the parameter space where they are prevalent was determined. The conclusion being that both models are not obsolete due to negative and zero eigenvalues respectively. The requirement being above 10^3 shots total for 1 & 2 qubits and purity below unity. The purity was demonstrated to not currently pose a issue in a experimental context, due to the states not being pure. The fidelity of all three models was investigated and show that at few qubits, no more than a magnitude of 10^4 shots are needed to perform accurate tomography, with all models. When more qubits are added, one naturally needs to measure more. The models where shown to perform robustly on simulated as well as experimental data, where it was capable of doing its task at the end of the pipe line in quantum state readout. Reasonable control of qubit 2 on the Soprano device was shown and the aim is to increase the fidelity and hopefully perform two-qubit operations on the device. In conclusion, LI & MLE are more favorable to use than BI in a experimental context, if the purity of the states are sufficiently low. This is due to the quite significant increase in computational time for the BI tomography model.

6 Acknowledgements

I would like to thank Morten Kjærgaard for giving me the opportunity to write my Bachelors Project in his group, and working on a cutting edge qubit device. Moreover I thank him for helpful advice, discussion and for bringing in romkugler while his foot was in a cast. I would like to thank Jacob Hastrup for his expertise and advice in building my Python code and understanding the mathematical concepts behind it. In addition i thank the members of the CQED group for helpful advice in the lab. Lastly i would like to thank Malthe Nielsen for rewarding sparring sessions and discussions.

References

[1] John Preskill. *Quantum computing 40 years later*. 2021. DOI: 10.48550/ARXIV.2106.10522. URL: https://arxiv.org/abs/2106.10522.

[2] John Preskill. "Quantum Computing in the NISQ era and beyond". In: *Quantum* 2 (Aug. 2018), p. 79. DOI: 10.22331/q-2018-08-06-79.

[3] Engineering National Academies of Sciences and Medicine. *Quantum Computing: Progress and Prospects*. Ed. by Emily Grumbling and Mark Horowitz. Washington, DC: The National Academies Press, 2019. ISBN: 978-0-309-47969-1. DOI: 10.17226/25196. URL: https://nap.nationalacademies.org/catalog/25196/quantum-computing-progress-and-prospects.

[4] P. Krantz et al. "A quantum engineer's guide to superconducting qubits". In: *Applied Physics Reviews* 6.2 (June 2019), p. 021318. DOI: 10.1063/1.5089550. URL: https://doi.org/10.1063%2F1.5089550.

[5] Robin Blume-Kohout. "Optimal, reliable estimation of quantum states". In: *New Journal of Physics* 12.4 (Apr. 2010), p. 043034. DOI: 10.1088/1367-2630/12/4/043034. URL: https://doi.org/10.1088%2F1367-2630%2F12%2F4%2F043034.

[6] Christopher Ferrie and Robin Blume-Kohout. *Maximum likelihood quantum state tomography is inadmissible*. 2018. DOI: 10.48550/ARXIV.1808.01072. URL: https://arxiv.org/abs/1808.01072.

[7] Bo Qi et al. "Quantum State Tomography via Linear Regression Estimation". In: *Scientific Reports* 3.1 (2013). ISSN: 2045-2322. DOI: 10.1038/srep03496. URL: https://dx.doi.org/10.1038/srep03496.

[8] Ermes Toninelli et al. "Concepts in quantum state tomography and classical implementation with intense light: a tutorial". In: *Adv. Opt. Photon.* 11.1 (Mar. 2019), pp. 67–134. DOI: 10.1364/AOP.11.000067. URL: http://opg.optica.org/aop/abstract.cfm?URI=aop-11-1-67.

[9] D. S. Gonçalves et al. "Bayesian inference for quantum state tomography". In: *Journal of Applied Statistics* 45.10 (2018), pp. 1846–1871. DOI: 10.1080/02664763.2017.1401049.

[10] Quantware. *Soprano QPU from Quantware*. https://www.quantware.eu/product/soprano. 2022.

[11] N. David Mermin. *Quantum Computer Science: An Introduction*. USA: Cambridge University Press, 2007. ISBN: 0521876583.

[12] Andreas Ketterer. *Modular variables in quantum information*. PhD 2016.

[13] Pradosh K. Roy. *Density Matrix , Mixed States*. July 2020. DOI: 10.13140/RG.2.2.21228.39045.

[14] Gina Passante, Paul J. Emigh, and Peter S. Shaffer. "Student ability to distinguish between superposition states and mixed states in quantum mechanics". In: *Physical Review Special Topics-physics Education Research* 11 (2015), p. 020135.

[15] J. C. A. Barata et al. "Pure and Mixed States". In: *Brazilian Journal of Physics* 51.2 (Nov. 2020), pp. 244–262. DOI: 10.1007/s13538-020-00808-0. URL: https://doi.org/10.1007%2Fs13538-020-00808-0.

[16] Marco Paini et al. "Estimating expectation values using approximate quantum states". In: *Quantum* 5 (Mar. 2021), p. 413. ISSN: 2521-327X. DOI: 10.22331/q-2021-03-16-413. URL: https://doi.org/10.22331/q-2021-03-16-413.

[17] Casper W. *Quantum State Tomography*. https://github.com/cqed-at-qdev/Quantum-State-Tomography.git. 2022.

[18] P. D. Nation J. R. Johansson and F. Nori. *QuTiP 2: A Python framework for the dynamics of open quantum systems*. 2013. DOI: [DOI:10.1016/j.cpc.2012.11.019].

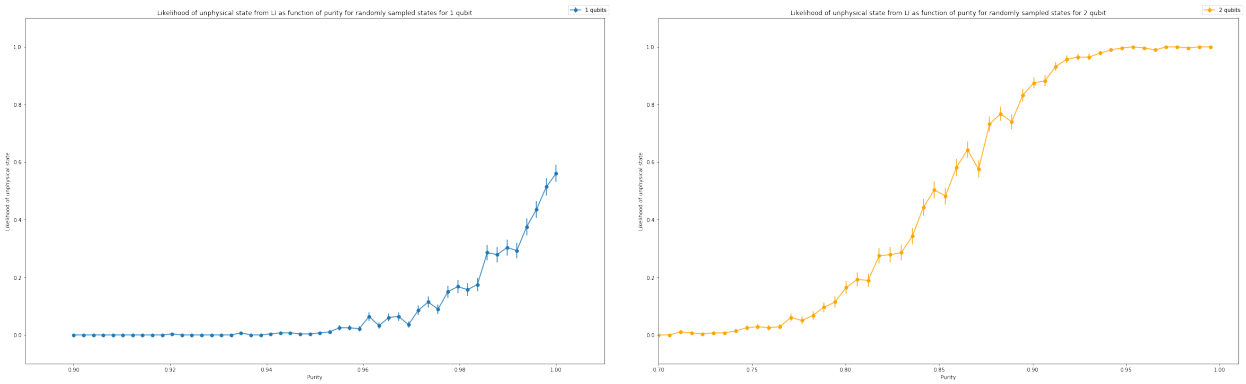
[19] Douglas S. Gonçalves et al. "Local solutions of Maximum Likelihood Estimation in Quantum State Tomography". In: (2011). DOI: 10.48550/ARXIV.1103.3682.

[20] Nicholas John Higham. "Analysis of the Cholesky Decomposition of a Semi-definite Matrix". In: 1990.

[21] A.M.N. Malthe. *Quantum_fitter*. https://github.com/qdev-dk/quantum_fitter/quantum_fitter.git. 2022.

A Appendix

A.1 Linear Inversion



(a) 1 qubit **(b)** 2 qubits

Figure A.1: Likelihood of linear inversion returning a non physical state on y-axis. Purity on x-axis. every point is 280 randomly generated quantum states, which each have been measured for 1000 shots.

A.2 Maximum Likelihood Estimation

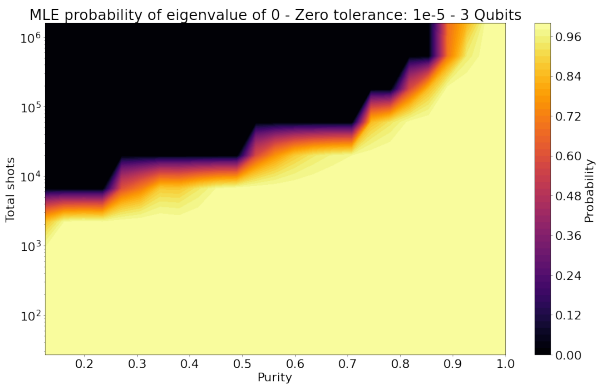


Figure A.2: Probability of 3 qubit MLE returning a state with one or more eigenvalues equal to zero is shown on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.

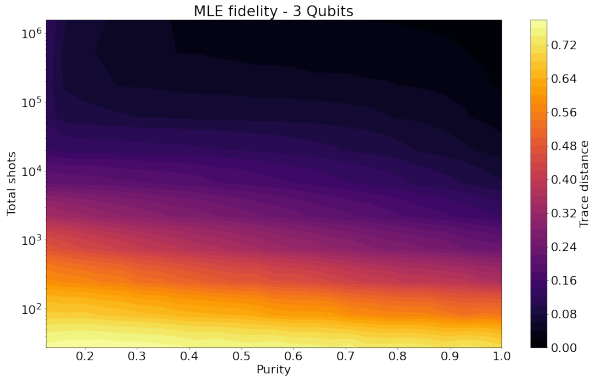
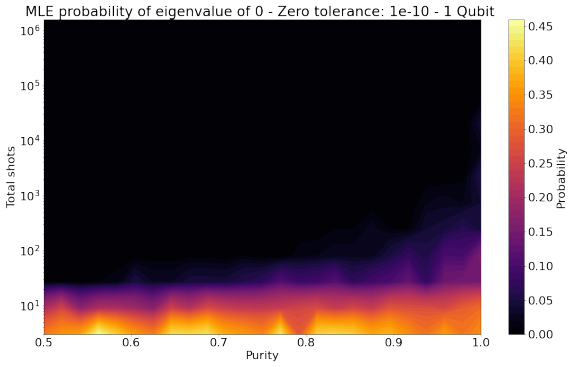
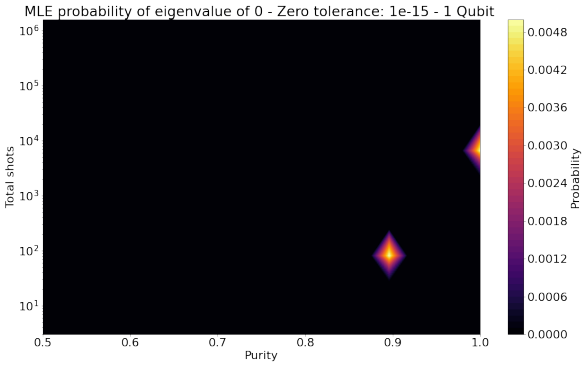


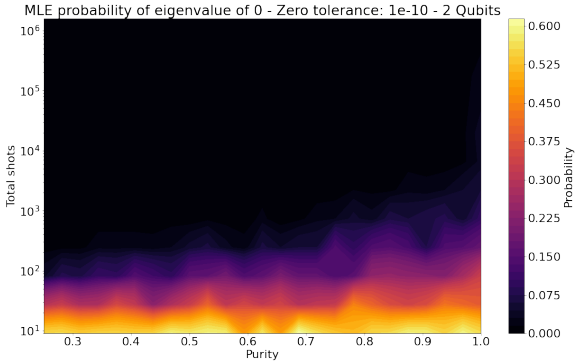
Figure A.3: Trace distance between the generated state, and the state estimated by 3 qubit MLE tomography is shown on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.



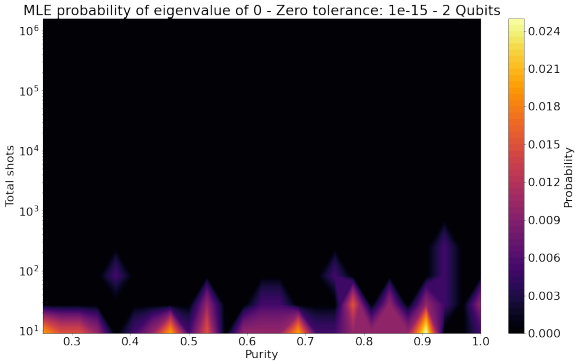
(a) 1 qubit - tol: 10^{-10}



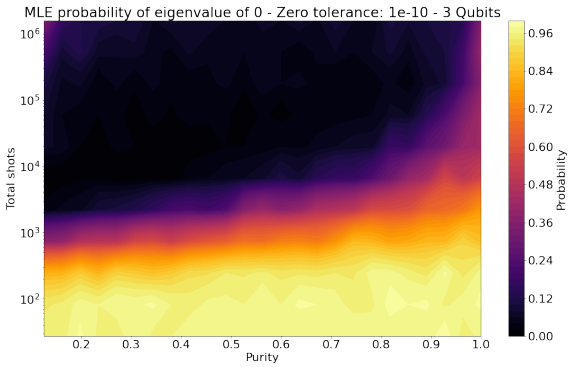
(b) 1 qubit - tol: 10^{-15}



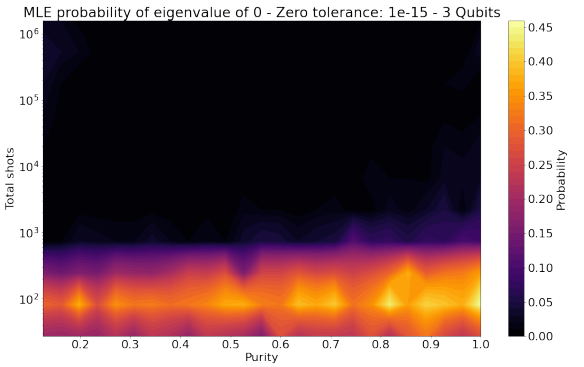
(c) 2 qubit - tol: 10^{-10}



(d) 2 qubit - tol: 10^{-15}



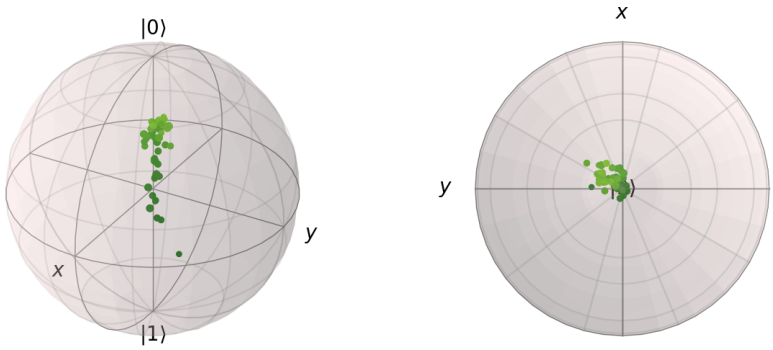
(e) 3 qubit - tol: 10^{-10}



(f) 3 qubit - tol: 10^{-15}

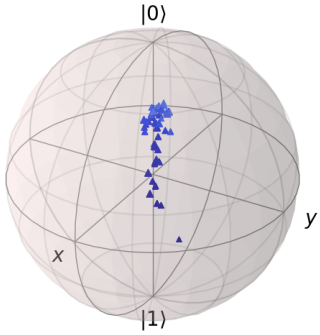
Figure A.4: Probability of MLE returning a state with one or more eigenvalues equal to zero is shown on the heat map. Purity of the generated state on the x-axis and number of measurement shots (total) on y-axis, distributed between the 3^n measurements.

A.3 Soprano data

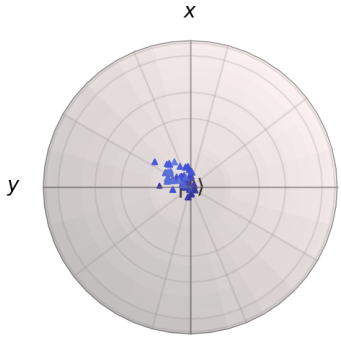


(a) LI - side view

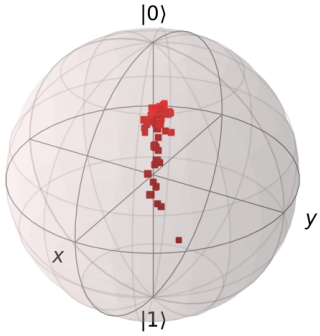
(b) LI - top view



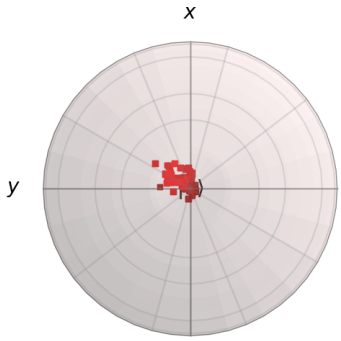
(c) MLE - side view



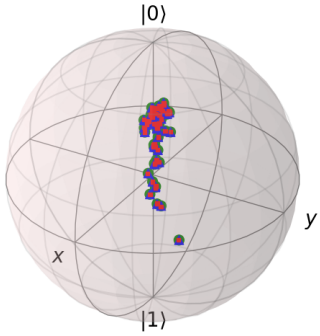
(d) MLE - top view



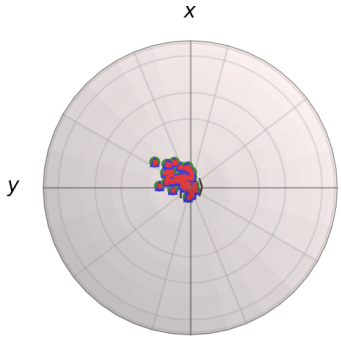
(e) BI - side view



(f) BI - top view

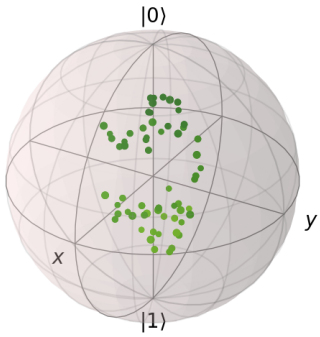


(g) All 3 models - side view

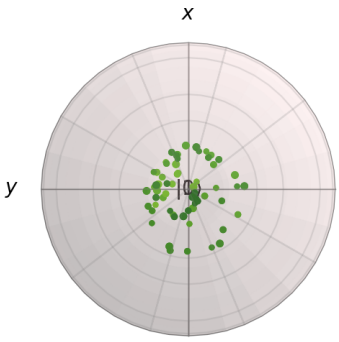


(h) All 3 models - top view

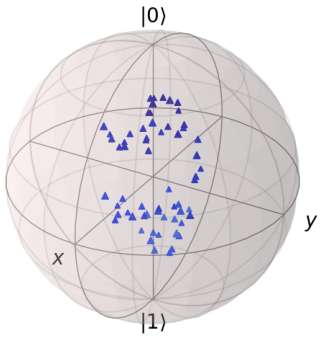
Figure A.5: Comparison of the three types of tomography on decay from excited to ground state, using measured data from Soprano device[10].



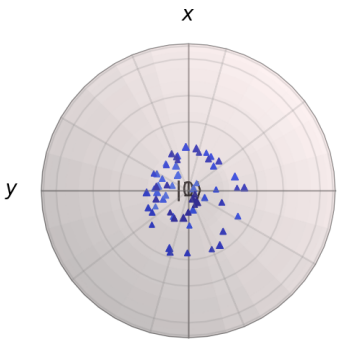
(a) LI - side view



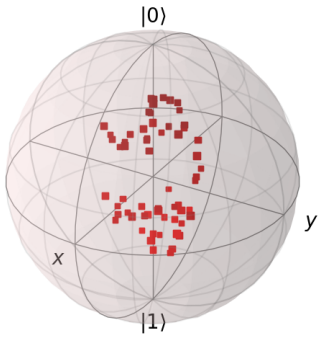
(b) LI - top view



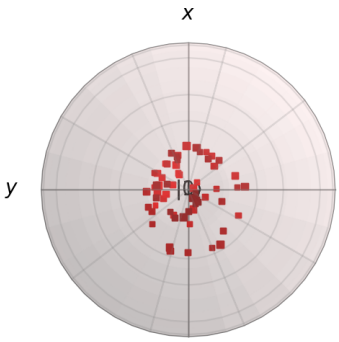
(c) MLE - side view



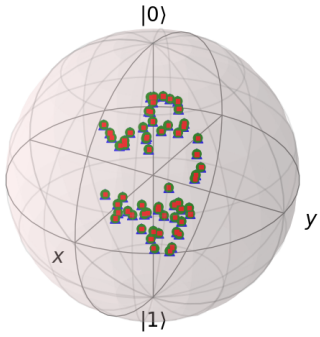
(d) MLE - top view



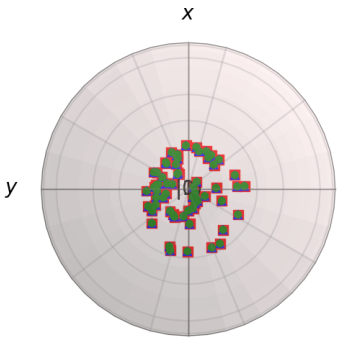
(e) BI - side view



(f) BI - top view



(g) All 3 models - side view



(h) All 3 models - top view

Figure A.6: Comparison of the three types of tomography on spiral around z axis from ground to excited, using measured data from Soprano device[10].

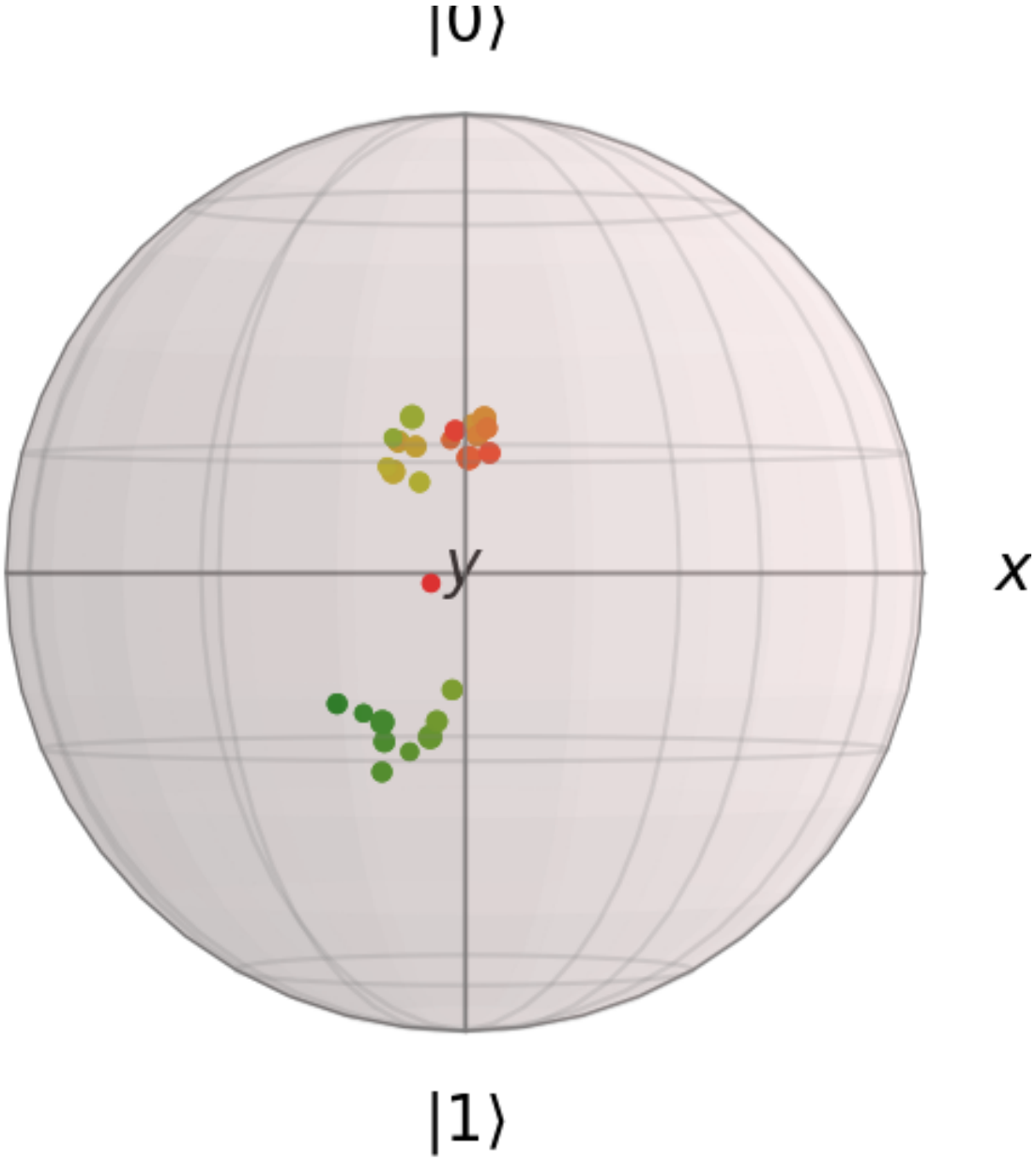


Figure A.7: QST using LI to construct a smiley using measured data from Soprano device[10].