Quantum Expectation-Maximization for Gaussian mixture models

Iordanis Kerenidis[1 2] **Alessandro Luongo**[3 1] **Anupam Prakash**[2]

1. Supplementary Material

We start by reviewing the classical EM algorithm for GMM, as in Algorithm 1.

Theorem 1.1 (Multivariate Mean Value Theorem (Rudin et al., 1964))**.** *Let U be an open set of $\mathbb{R}^d$. For a differentiable functions $f : U \mapsto \mathbb{R}$ it holds that $\forall x, y \in U$, $\exists c$ such that $f(x) - f(y) = \nabla f(c) \cdot (x - y)$.*

Lemma 1.2 (Componentwise *Softmax* function $\sigma_j(v)$ is Lipschitz continuous)**.** *For $d > 2$, let $\sigma_j : \mathbb{R}^d \mapsto (0, 1)$ be the* softmax *function defined as $\sigma_j(v) = \frac{e^{v_j}}{\sum_{l=1}^{d} e^{v_l}}$ Then σ_j is Lipschitz continuous, with $K \leq \sqrt{2}$.*

Proof. We need to find the K such that for all $x, y \in \mathbb{R}^d$, we have that $\left\|\sigma_j(y) - \sigma_j(x)\right\| \leq K \left\|y - x\right\|$. Observing that σ_j is differentiable and that if we apply Cauchy-Schwarz to the statement of the Mean-Value-Theorem we derive that $\forall x, y \in U$, $\exists c$ such that $\left\|f(x) - f(y)\right\| \leq \left\|\nabla f(c)\right\|_F \left\|x - y\right\|$. So to show Lipschitz continuity it is enough to select $K \leq \left\|\nabla \sigma_j\right\|_F^* = \max_{c \in \mathbb{R}^d} \left\|\nabla \sigma_j(c)\right\|$.

The partial derivatives $\frac{d\sigma_j(v)}{dv_i}$ are $\sigma_j(v)(1 - \sigma_j(v))$ if $i = j$ and $-\sigma_i(v)\sigma_j(v)$ otherwise. So $\left\|\nabla \sigma_j\right\|_F^2 = \sum_{i=1}^{d-1}(-\sigma(v)_i\sigma_j(v))^2 + \sigma_j(v)^2(1 - \sigma_j(v))^2 \leq \sum_{i=1}^{d-1} \sigma(v)_i\sigma_j(v) + \sigma_j(v)(1 - \sigma_j(v)) \leq \sigma_j(v)\sum_{i=0}^{d-1} \sigma_i(v) + 1 - \sigma_j(v) \leq 2\sigma_j(v) \leq 2$. In our case we can deduce that: $\left\|\sigma_j(y) - \sigma_j(x)\right\| \leq \sqrt{2}\left\|y - x\right\|$ so $K \leq \sqrt{2}$.

□

Definition 1 (Exponential Family (Murphy, 2012))**.** *A probability density function or probability mass function $p(v|\nu)$ for $v = (v_1, \cdots, v_m) \in \mathcal{V}^m$, where $\mathcal{V} \subseteq \mathbb{R}$, $\nu \in \mathbb{R}^p$ is said to be in the exponential family if can be written as:*

$$p(v|\nu) := h(v)\exp\{o(\nu)^T T(v) - A(\nu)\}$$

where:

- *$\nu \in \mathbb{R}^p$ is called the* canonical or natural *parameter of the family,*
- *$o(\nu)$ is a function of ν (which often is just the identity function),*
- *$T(v)$ is the vector of sufficient statistics: a function that holds all the information the data v holds with respect to the unknown parameters,*
- *$A(\nu)$ is the cumulant generating function, or log-partition function, which acts as a normalization factor,*
- *$h(v) > 0$ is the* base measure *which is a non-informative prior and de-facto is scaling constant.*

Lemma 1.3 (Error in the responsibilities of the exponential family)**.** *Let $v_i \in \mathbb{R}^n$ be a vector, and let $\{p(v_i|\nu_j)\}_{j=1}^k$ be a set of k probability distributions in the exponential family, defined as $p(v_i|\nu_j) := h_j(v_i)exp\{o_j(\nu_j)^T T_j(v_i) - A_j(\nu_j)\}$. Then, if we have estimates for each exponent with error ϵ, then we can compute each r_{ij} such that $|\overline{r_{ij}} - r_{ij}| \leq \sqrt{2k}\epsilon$ for $j \in [k]$.*

Proof. The proof follows from rewriting the responsibility of Equation (1) as:

$$r_{ij} := \frac{h_j(v_i)\exp\{o_j(\nu_j)^T T(v_i) - A_j(\nu_j) \ + \log\theta_j\}}{\sum_{l=1}^{k} h_l(v_i)\exp\{o_l(\nu_l)^T T(v_i) - A_l(\nu_l) \ + \log\theta_l\}} \tag{6}$$

In this form, it is clear that the responsibilities can be seen a *softmax* function, and we can use Theorem 1.2 to bound the error in computing this value.

Let $T_i \in \mathbb{R}^k$ be the vector of the exponent, that is $t_{ij} = o_j(\nu_j)^T T(v_i) - A_j(\nu_j) + \log\theta_j$. In an analogous way we define $\overline{T_i}$ the vector where each component is the estimate with error ϵ. The error in the responsibility is defined as $|r_{ij} - \overline{r_{ij}}| = |\sigma_j(T_i) - \sigma_j(\overline{T_i})|$. Because the function σ_j is Lipschitz continuous, as we proved in Theorem 1.2 with a Lipschitz constant $K \leq \sqrt{2}$, we have that, $|\sigma_j(T_i) - \sigma_j(\overline{T_i})| \leq \sqrt{2}\left\|T_i - \overline{T_i}\right\|$. The result follows as $\left\|T_i - \overline{T_i}\right\| < \sqrt{k}\epsilon$. □

[*] Equal contribution [1] Institute de Recherce en Informatique Fondamental - Paris, France [2] QCWare Corp. - Palo Alto, California [3] Atos Quantum Lab - Les Clayes sous bois, France. Correspondence to: Alessandro Luongo <aluongo@irif.fr>.

Proceedings of the 37^{th} International Conference on Machine Learning, Online, PMLR 119, 2020. Copyright 2020 by the author(s).

Algorithm 1 Expectation-Maximization for GMM

```
Require: Dataset $V$, tolerance $\tau > 0$.
Ensure: A GMM $\gamma^t = (\theta^t, \boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t)$ that maximizes locally
    the likelihood $\ell(\gamma; V)$ up to tolerance $\tau$.

Select $\gamma^0 = (\theta^0, \boldsymbol{\mu}^0, \boldsymbol{\Sigma}^0)$ using classical initialization
    strategies described in Subsection 1.2.
$t = 0$
repeat
    Expectation
    $\forall i, j$, calculate the responsibilities as:
        $r_{ij}^t = \frac{\theta_j^t \phi(v_i; \mu_j^t, \Sigma_j^t)}{\sum_{l=1}^k \theta_l^t \phi(v_i; \mu_l^t, \Sigma_l^t)}$    (1)
    Maximization
    Update the parameters of the model as:
        $\theta_j^{t+1} \leftarrow \frac{1}{n} \sum_{i=1}^n r_{ij}^t$    (2)
        $\mu_j^{t+1} \leftarrow \frac{\sum_{i=1}^n r_{ij}^t v_i}{\sum_{i=1}^n r_{ij}^t}$    (3)
        $\Sigma_j^{t+1} \leftarrow \frac{\sum_{i=1}^n r_{ij}^t (v_i - \mu_j^{t+1})(v_i - \mu_j^{t+1})^T}{\sum_{i=1}^n r_{ij}^t}$    (4)
    t=t+1
until
    $|\ell(\gamma^{t-1}; V) - \ell(\gamma^t; V)| < \tau$    (5)
Return $\gamma^t = (\theta^t, \boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t)$
```

1.1. Quantum procedures and further preliminaries

A qubit is a mathematical representation of a quantum mechanical object as a l_2 normalized vector of length 2 on $\mathbb{C}^2$. The state of a n-qubit system (a register of a quantum computer) is the tensor product of the single qubits: a unitary vector $|x\rangle \in H^{\otimes n} \simeq \mathbb{C}^{2^n}$. In other words, with $logn$ qubits we can describe a quantum state $|\psi\rangle = \sum_{i \in [n]} \alpha_i |i\rangle$ with $\sum_{i \in [n]} |\alpha_i|^2 = 1$. The values α_i are called amplitudes of the quantum state $|i\rangle$. The evolution of a quantum system is described by unitary matrices U. A matrix U is said to be unitary if $UU^\dagger = U^\dagger U = I$. From this fact it follows that unitary matrices are norm-preserving, and thus can be used as suitable mathematical description of pure quantum evolutions.

The dataset is represented by a matrix $V \in \mathbb{R}^{n \times d}$, i.e. each row is a vector $v_i \in \mathbb{R}^d$ for $i \in [n]$ that represents a single data point. A matrix U is said to be unitary if $UU^\dagger = U^\dagger U = I$. The cluster centers, called centroids, at time t are stored in the matrix $C^t \in \mathbb{R}^{k \times d}$, such that the j^{th} row c_j^t for $j \in [k]$ represents the centroid of the cluster $\mathcal{C}_j^t$. We denote as $V_{\geq \tau}$ the matrix $\sum_{i=0}^{\ell} \sigma_i u_i v_i^T$ where σ_ℓ is the smallest singular value which is greater than τ. With $nnz(V)$ is the number of non-zero elements of the rows of V. Let $\kappa(V)$ be the condition number of V: that is the ratio between the biggest and the smallest (non-zero) singular value. We recommend Nielsen and Chuang (Nielsen & Chuang, 2002) for an introduction to the subject.

To prove our results, we are going to use the quantum procedures listed hereafter.

Claim 1.4. *(Kerenidis & Prakash, 2020) Let θ be the angle between vectors x, y, and assume that $\theta < \pi/2$. Then, $\|x - y\| \leq \epsilon$ implies $\||x\rangle - |y\rangle\| \leq \frac{\sqrt{2}\epsilon}{\|x\|}$. Where $|x\rangle$ and $|y\rangle$ are two unit vectors in ℓ_2 norm.*

We will also use Claim 4.5 from (Kerenidis et al., 2019a).

Claim 1.5. *(Kerenidis et al., 2019a) Let ϵ_b be the error we commit in estimating $|c\rangle$ such that $\||c\rangle - |\overline{c}\rangle\| < \epsilon_b$, and ϵ_a the error we commit in the estimating the norms, $|\,\|c\| - \overline{\|c\|}| \leq \epsilon_a \|c\|$. Then $\|\overline{c} - c\| \leq \sqrt{\eta}(\epsilon_a + \epsilon_b)$.*

Theorem 1.6 (Amplitude estimation and amplification (Brassard et al., 2002))**.** *If there is unitary operator U such that $U|0\rangle^l = |\phi\rangle = \sin(\theta)|x, 0\rangle + \cos(\theta)|G, 0^\perp\rangle$ then $\sin^2(\theta)$ can be estimated to multiplicative error η in time $O(\frac{T(U)}{\eta \sin(\theta)})$ and $|x\rangle$ can be generated in expected time $O(\frac{T(U)}{\sin(\theta)})$.*

We also need some state preparation procedures. These subroutines are needed for encoding vectors in $v_i \in \mathbb{R}^d$ into quantum states $|v_i\rangle$. An efficient state preparation procedure is provided by the QRAM data structures. We stress the fact that our results continue to hold, no matter how the efficient quantum loading of the data is provided. For instance, the data can be accessed through a QRAM, through a block encoding, or when the data can be produced by quantum circuits.

Theorem 1.7 (QRAM data structure (Kerenidis & Prakash, 2017))**.** *Let $V \in \mathbb{R}^{n \times d}$, there is a data structure to store the rows of V such that,*

1. *The time to insert, update or delete a single entry v_{ij} is $O(\log^2(n))$.*
2. *A quantum algorithm with access to the data structure can perform the following unitaries in time $T = O(\log^2 N)$.*
   (a) $|i\rangle|0\rangle \rightarrow |i\rangle|v_i\rangle$ for $i \in [n]$.
   (b) $|0\rangle \rightarrow \sum_{i \in [n]} \|v_i\| |i\rangle$.

In our algorithm we will also use subroutines for quantum linear algebra. For a symmetric matrix $M \in \mathbb{R}^{d\times d}$ with spectral norm $\|M\| = 1$, the running time of these algorithms depends linearly on the condition number $\kappa(M)$ of the matrix, that can be replaced by $\kappa_\tau(M)$, a condition threshold where we keep only the singular values bigger than τ, and the parameter $\mu(M)$, a matrix dependent parameter defined as

$$\mu(M) = \min_{p\in P}(\|M\|_F, \sqrt{s_{2p}(M)s_{2(1-p)}(M^T)}),$$

for $s_p(M) := \max_{i\in[n]} \|m_i\|_p^p$ where $\|m_i\|_p$ is the ℓ_p norm of the i-th row of M, and P is a finite set of size $O(1) \in [0,1]$. Note that $\mu(M) \le \|M\|_F \le \sqrt{d}$ as we have assumed that $\|M\| = 1$. The running time also depends logarithmically on the relative error ϵ of the final outcome state. (Chakraborty et al., 2018; Gilyén et al., 2018).

Theorem 1.8 (Quantum linear algebra (Chakraborty et al., 2018; Gilyén et al., 2018))**.** *Let $M \in \mathbb{R}^{d\times d}$ such that $\|M\|_2 = 1$ and $x \in \mathbb{R}^d$. Let $\epsilon, \delta > 0$. If we have quantum access to M and the time to prepare $|x\rangle$ is T_x, then there exist quantum algorithms that with probability at least $1 - 1/poly(d)$ return a state $|z\rangle$ such that $\||z\rangle - |Mx\rangle\| \le \epsilon$ in time $\widetilde{O}((\kappa(M)\mu(M) + T_x\kappa(M))\log(1/\epsilon))$.*

Theorem 1.9 (Quantum linear algebra for matrix products (Chakraborty et al., 2018))**.** *Let $M_1, M_2 \in \mathbb{R}^{d\times d}$ such that $\|M\|_1 = \|M\|_2 = 1$ and $x \in \mathbb{R}^d$, and a vector $x \in \mathbb{R}^d$ for which we have quantum access. Let $\epsilon > 0$. Then there exist quantum algorithms that with probability at least $1-1/poly(d)$ returns a state $|z\rangle$ such that $\||z\rangle - |Mx\rangle\| \le \epsilon$ in time $\widetilde{O}((\kappa(M)(\mu(M_1)T_{M_1} + \mu(M_2)T_{M_2}))\log(1/\epsilon))$, where T_{M_1}, T_{M_2} is the time needed to index the rows of M_1 and M_2.*

The linear algebra procedures above can also be applied to any rectangular matrix $V \in \mathbb{R}^{n\times d}$ by considering instead the symmetric matrix $\overline{V} = \begin{pmatrix} 0 & V \\ V^T & 0 \end{pmatrix}$.

The final component needed for the QEM algorithm is an algorithm for vector state tomography that will be used to recover classical information from the quantum states corresponding to the new centroids in each step. We report here two kind of vector state tomography. The ℓ_2 tomography has stronger guarantees on the output, while the ℓ_∞ is faster.

Theorem 1.10 (ℓ_∞ Vector state tomography)**.** *(Kerenidis et al., 2019b) Given access to unitary U such that $U|0\rangle = |x\rangle$ and its controlled version in time $T(U)$, there is a tomography algorithm with time complexity $O(T(U)\frac{\log d}{\delta^2})$ that produces unit vector $\widetilde{X} \in \mathbb{R}^d$ such that $\left\|\widetilde{X} - x\right\|_\infty \le \delta$ with probability at least $(1 - 1/poly(d))$.*

Theorem 1.11 (Vector state tomography (Kerenidis & Prakash, 2018))**.** *Given access to unitary U such that $U|0\rangle = |x\rangle$ and its controlled version in time $T(U)$, there is a tomography algorithm with time complexity $O(T(U)\frac{d\log d}{\epsilon^2})$ that produces unit vector $\widetilde{x} \in \mathbb{R}^d$ such that $\|\widetilde{x} - x\|_2 \le \epsilon$ with probability at least $(1 - 1/poly(d))$.*

Lemma 1.12 (Distance / Inner Products Estimation (Kerenidis et al., 2019a; Wiebe et al., 2014; Lloyd et al., 2013))**.** *Assume for a data matrix $V \in \mathbb{R}^{n\times d}$ and a centroid matrix $C \in \mathbb{R}^{k\times d}$ that the following unitaries $|i\rangle|0\rangle \mapsto |i\rangle|v_i\rangle$, and $|j\rangle|0\rangle \mapsto |j\rangle|c_j\rangle$ can be performed in time T and the norms of the vectors are known. For any $\Delta > 0$ and $\epsilon > 0$, there exists a quantum algorithm that computes*

$$|i\rangle|j\rangle|0\rangle \mapsto |i\rangle|j\rangle|\overline{d^2(v_i,c_j)}\rangle,$$

where$|\overline{d^2(v_i,c_j)} - d^2(v_i,c_j)| \leqslant \epsilon$ *with probability at least*$1-2\Delta$*, or*

$$|i\rangle|j\rangle|0\rangle \mapsto |i\rangle|j\rangle|\overline{(v_i,c_j)}\rangle,$$

where $|\overline{(v_i,c_j)} - (v_i,c_j)| \leqslant \epsilon$ *with probability at least* $1-2\Delta$ *in time* $\widetilde{O}\left(\frac{\|v_i\|\|c_j\|T\log(1/\Delta)}{\epsilon}\right)$.

1.2. Initialization strategies for EM

Unlike k-means clustering, choosing a good set of initial parameters for a mixture of Gaussian is by no means trivial, and in multivariate context is known that the solution is problem-dependent. There are plenty of proposed techniques, and here we describe a few of them. Fortunately, these initialization strategies can be directly translated into quantum subroutines without impacting the overall running time of the quantum algorithm.

The simplest technique is called *random EM*, and consists in selecting initial points at random from the dataset as centroids, and sample the dataset to estimate the covariance matrix of the data. Then these estimates are used as the starting configuration of the model, and we may repeat the random sampling until we get satisfactory results.

A more standard technique borrows directly the initialization strategy of *k-means++* proposed in (Arthur & Vassilvitskii, 2007), and extends it to make an initial guess for the covariance matrices and the mixing weights. The initial guess for the centroids is selected by sampling from a suitable, easy to calculate distribution. This heuristic works as following: Let c_0 be a randomly selected point of the dataset, as first centroid. The other $k-1$ centroids are selected by selecting a vector v_i with probability proportional to $d^2(v_i, \mu_{l(v_i)})$, where $\mu_{l(v_i)}$ is the previously selected centroid that is the closest to v_i in ℓ_2 distance. These centroids are then used as initial centroids for a round of k-means algorithm to obtain $\mu_1^0 \cdots \mu_j^0$. Then, the covariance matrices can be initialized as $\Sigma_j^0 := \frac{1}{|\mathcal{C}_j|}\sum_{i\in\mathcal{C}_j}(v_i - \mu_j)(v_i - \mu_j)^T$, where $\mathcal{C}_j$ is the set of samples in the training set that have been assigned to the cluster j in the previous round of k-means. The mixing weights are estimated as $\mathcal{C}_j/n$. Eventually Σ_j^0 is regularized to be a PSD matrix.

There are other possible choices for parameter initialization in EM, for instance, based on *Hierarchical Agglomerative Clustering (HAC)* and the *CEM* algorithm. In CEM we run one step of EM, but with a so-called classification step between E and M. The classification step consists in a hard-clustering after computing the initial conditional probabilities (in the E step). The M step then calculates the initial guess of the parameters (Celeux & Govaert, 1992). In the *small EM* initialization method we run EM with a different choice of initial parameters using some of the previous strategies. The difference here is that we repeat the EM algorithm for a small number of iterations, and we keep iterating from the choice of parameters that returned the best partial results. For an overview and comparison of different initialization techniques, we refer to (Blömer & Bujna, 2013; Biernacki et al., 2003).

Quantum initialization strategies For the initialization of γ^0 in the quantum algorithm we can use the same initialization strategies as in classical machine learning. For instance, we can use the classical *random EM* initialization strategy for QEM.

A quantum initialization strategy can also be given using the *k-means*++ initializion strategy from (Kerenidis et al., 2019a), which returns k initial guesses for the centroids $c_1^0 \cdots c_k^0$ consistent with the classical algorithm in time $\left(k^2 \frac{2\eta^{1.5}}{\epsilon\sqrt{\mathbb{E}(d^2(v_i,v_j))}}\right)$, where $\mathbb{E}(d^2(v_i,v_j))$ is the average squared distance between two points of the dataset, and ϵ is the tolerance in the distance estimation. From there, we can perform a full round of q-means algorithm and get an estimate for $\mu_1^0 \cdots \mu_k^0$. With q-means and quantum access to the centroids, we can create the state

$$|\psi^0\rangle := \frac{1}{\sqrt{n}}\sum_{i=1}^{n} |i\rangle \, |l(v_i)\rangle \, . \tag{7}$$

Where $l(v_i)$ is the label of the closest centroid to the i-th point. By sampling $S \in O(d)$ points from this state we get two things. First, from the frequency f_j of the second register we can have a guess of $\theta_j^0 \leftarrow |\mathcal{C}_j|/n \sim f_j/S$. Then, from the first register we can estimate $\Sigma_j^0 \leftarrow \sum_{i\in S}(v_i - \mu_j^0)(v_i - \mu_j^0)^T$. Sampling $O(d)$ points and creating the state in Equation (7) takes time $\widetilde{O}(dk\eta)$ by Theorem 1.12 and the minimum finding procedure described in (Kerenidis et al., 2019a).

Techniques illustrated in (Miyahara et al., 2019) can also be used to quantize the CEM algorithm which needs a hard-clustering step. Among the different possible approaches, the *random* and the *small EM* greatly benefit from a faster algorithm, as we can spend more time exploring the space of the parameters by starting from different initial seeds, and thus avoid local minima of the likelihood.

1.3. Special cases of GMM.

What we presented in the main manuscript is the most general model of GMM. For simple datasets, it is common to assume some restrictions on the covariance matrices of the mixtures. The translation into a quantum version of the model should be straightforward. We distinguish between these cases:

1. **Soft k-means**. This algorithm is often presented as a generalization of k-means, but it can actually be seen as special case of EM for GMM - albeit with a different assignment rule. In soft k-means, the assignment function is replaced by a softmax function with *stiffness* parameter β. This β represents the covariance of the clusters. It is assumed to be equal for all the clusters, and for all dimensions of the feature space. Gaussian Mixtures with constant covariance matrix (i.e. $\Sigma_j = \beta I$ for $\beta \in \mathbb{R}$) can be interpreted as a kind of soft or fuzzy version of k-means clustering. The probability of a point in the feature space being assigned to a certain cluster j is:

   $$r_{ij} = \frac{e^{-\beta\|x_i - \mu_i\|^2}}{\sum_{l=1}^{k} e^{-\beta\|x_i - \mu_l\|^2}}$$

   where $\beta > 0$ is the stiffness parameter.

2. **Spherical**. In this model, each component has its own covariance matrix, but the variance is uniform in all the directions, thus reducing the covariance matrix to a multiple of the identity matrix (i.e. $\Sigma_j = \sigma_j^2 I$ for $\sigma_j \in \mathbb{R}$).

3. **Diagonal**. As the name suggests, in this special case the covariance matrix of the distributions is a diagonal matrix, but different Gaussians might have different diagonal covariance matrices.

4. **Tied**. In this model, the Gaussians share the same covariance matrix, without having further restriction on the Gaussian.

5. **Full**. This is the most general case, where each of the components of the mixture have a different, SDP, covariance matrix.

1.4. Proofs

In this section we report all the proof of the theorems that we used to prove the main result. The numbering of the lemmas follows the numbering defined in the main manuscript.

Lemma 3.2 (Quantum Gaussian Evaluation)**.** *Suppose we have quantum access to a matrix $V \in \mathbb{R}^{n\times d}$, the centroid $\mu \in \mathbb{R}^d$ and the covariance matrix $\Sigma \in \mathbb{R}^{d\times d}$ of a multivariate Gaussian distribution $\phi(v|\mu,\Sigma)$, as well as an estimate for $\log(\det(\Sigma))$. Then for $\epsilon_1 > 0$, there exists a quantum algorithm that with probability $1-\gamma$ performs the mapping,*

- $U_{G,\epsilon_1} : |i\rangle|0\rangle \rightarrow |i\rangle|\overline{s_i}\rangle$ *such that* $|s_i - \overline{s_i}| < \epsilon_1$, *where* $s_i = -\frac{1}{2}((v_i-\mu)^T\Sigma^{-1}(v_i-\mu) + d\log 2\pi + \log(det(\Sigma)))$ *is the exponent for the Gaussian probability density function.*

The running time of the algorithm is,

$$T_{G,\epsilon_1} = O\left(\frac{\kappa^2(\Sigma)\mu(\Sigma)\log(1/\gamma)}{\epsilon_1}\eta\right). \tag{8}$$

Proof. We use quantum linear algebra and inner product estimation to estimate the quadratic form $(v_i-\mu)^T\Sigma^{-1}(v_i-\mu)$ to error ϵ_1. First, we decompose the quadratic form as $v_i^T\Sigma^{-1}v_i - 2v_i^T\Sigma^{-1}\mu + \mu^T\Sigma^{-1}\mu$ and separately approximate each term in the sum to error $\epsilon_1/4$.

We describe the procedure to estimate $\mu^T\Sigma^{-1}v_i = \left\|\Sigma^{-1}v_i\right\|\|\mu\|\langle\mu|\Sigma^{-1}v_i\rangle$, the other estimates are obtained similarly. We use the quantum linear algebra subroutines in Theorem 1.8 to construct $|\Sigma^{-1}v_i\rangle$ up to error $\epsilon_3 \ll \epsilon_1$ in time $O(\kappa(\Sigma)\mu(\Sigma)\log(1/\epsilon_3))$ and estimate $\left\|\Sigma^{-1}v_i\right\|$ up to error ϵ_1 in time $O(\kappa(\Sigma)\mu(\Sigma)\log(1/\epsilon_3)/\epsilon_1)$ which gives us the mapping $|i\rangle|0\rangle \mapsto |i\rangle|\|\Sigma^{-1}v_i\|\rangle$. We then use quantum inner product estimation (Theorem 1.12) to estimate $\langle\mu,\Sigma^{-1}v_i\rangle$ to additive error $\frac{\epsilon_1}{4\|\mu\|\|\Sigma^{-1}v_i\|}$. The procedure estimates $(\mu,\Sigma^{-1}v_i)$ within additive error $\epsilon_1/4$. The procedure succeeds with probability $1-\gamma$ and requires time $O(\frac{\kappa(\Sigma)\mu(\Sigma)\log(1/\gamma)\log(1/\epsilon_3)}{\epsilon_1}\|\mu\|\left\|\Sigma^{-1}v_i\right\|)$. Using similar estimation procedure for $v_i^T\Sigma^{-1}\mu$ and $\mu^T\Sigma^{-1}\mu$, we obtain an estimate for $\frac{1}{2}((v_i-\mu)^T\Sigma^{-1}(v_i-\mu)$ within error ϵ_1.

Recall that (through Lemma **??**) we also have an estimate of the log-determinant to error ϵ_1. Thus we obtain an approximation for $-\frac{1}{2}((v_i-\mu)^T\Sigma^{-1}(v_i-\mu) + d\log 2\pi + \log(\det(\Sigma)))$ within error $2\epsilon_1$. We have the upper bound, $\left\|\Sigma^{-1}v_i\right\| \leq \left\|\Sigma^{-1}\right\|\|v_i\| \leq \kappa(\Sigma)\|v_i\|$, as $\|\Sigma\| = 1$. Further observing that $\|u\| \leq \sqrt{\eta}$ and $\|v_i\| \leq \sqrt{\eta}$, the running time for this computation is $O\left(\frac{\kappa^2(\Sigma)\mu(\Sigma)\log(1/\gamma)\log(1/\epsilon_3)}{\epsilon_1}\eta\right)$.

□

Lemma 3.3 (Calculating responsibilities)**.** *Suppose we have quantum access to a GMM with parameters $\gamma^t = (\theta^t, \boldsymbol{\mu}^t, \boldsymbol{\Sigma}^t)$. There are quantum algorithms that can:*

1. *Perform the mapping $|i\rangle|j\rangle|0\rangle \mapsto |i\rangle|j\rangle|\overline{r_{ij}}\rangle$ such that $|\overline{r_{ij}} - r_{ij}| \leq \epsilon_1$ with probability $1-\gamma$ in time:* $T_{R_1,\epsilon_1} = \widetilde{O}(k^{1.5}\times T_{G,\epsilon_1})$
2. *For a given $j \in [k]$, construct state $|\overline{R_j}\rangle$ such that $\left\||\overline{R_j}\rangle - \frac{1}{\sqrt{Z_j}}\sum\limits_{i=0}^{n} r_{ij}|i\rangle\right\| < \epsilon_1$ where $Z_j = \sum\limits_{i=0}^{n} r_{ij}^2$ with high probability in time: $T_{R_2,\epsilon_1} = \widetilde{O}(k^2 \times T_{R_1,\epsilon_1})$*

Proof. For the first statement, let's recall the definition of responsibility: $r_{ij} = \frac{\theta_j\phi(v_i;\mu_j,\Sigma_j)}{\sum_{l=1}^k\theta_l\phi(v_i;\mu_l,\Sigma_l)}$. With the aid of U_{G,ϵ_1} of Lemma 3.2 we can estimate $\log(\phi(v_i|\mu_j,\Sigma_j))$ for all j up to additive error ϵ_1, and then using the current estimate of θ^t, we can calculate the responsibilities and create the state,

$$\frac{1}{\sqrt{n}}\sum_{i=0}^{n}|i\rangle\Big(\bigotimes_{j=1}^{k}|j\rangle|\overline{\log(\phi(v_i|\mu_j,\Sigma_j))}\rangle\Big)\otimes|\overline{r_{ij}}\rangle.$$

The estimate $\overline{r_{ij}}$ is computed by evaluating a weighted softmax function with arguments $\overline{\log(\phi(v_i|\mu_j,\Sigma_j)}$ for $j \in [k]$. The estimates $\overline{\log(\phi(v_i|\mu_j,\Sigma_j)}$ are then uncomputed. The runtime of the procedure is given by calling k times Lemma 3.2 for Gaussian estimation (the arithmetic operations to calculate the responsibilities are absorbed).

Let us analyze the error in the estimation of r_{ij}. The responsibility r_{ij} is a softmax function with arguments $\log(\phi(v_i|\mu_j,\Sigma_j))$ that are computed upto error ϵ_1 using Lemma 3.2. As the softmax function has a Lipschitz constant $K \leq \sqrt{2}$ by Lemma 1.3, we choose precision for Lemma 3.2 to be $\epsilon_1/\sqrt{2k}$ to get the guarantee $|\overline{r_{ij}} - r_{ij}| \leq \epsilon_1$. Thus, the total cost of this step is $T_{R_1,\epsilon_1} = k^{1.5}T_{G,\epsilon_1}$.

Now let's see how to encode this information in the amplitudes, as stated in the second claim of the Lemma. We estimate the responsibilities r_{ij} to some precision ϵ and perform a controlled rotation on an ancillary qubit to obtain,

$$\frac{1}{\sqrt{n}}|j\rangle\sum_{i=0}^{n}|i\rangle|\overline{r_{ij}}\rangle\Big(\overline{r_{ij}}|0\rangle + \sqrt{1-\overline{r_{ij}}^2}|1\rangle\Big). \tag{9}$$

We then undo the circuit on the second register and perform amplitude amplification on the rightmost auxiliary qubit being $|0\rangle$ to get $|\overline{R_j}\rangle := \frac{1}{\|\overline{R_j}\|}\sum_{i=0}^n\overline{r_{ij}}|i\rangle$. The runtime for amplitude amplification on this task is $O(T_{R_1,\epsilon}\cdot\frac{\sqrt{n}}{\|\overline{R_j}\|})$.

Let us analyze the precision ϵ required to prepare $|\overline{R_j}\rangle$ such that $\left\||R_j\rangle - |\overline{R_j}\rangle\right\| \leq \epsilon_1$. As we have estimates $|r_{ij} - \overline{r_{ij}}| < \epsilon$ for all i, j, the ℓ_2-norm error $\left\|R_j - \overline{R_j}\right\| = \sqrt{\sum_{i=0}^n|r_{ij} - \overline{r_{ij}}|^2} < \sqrt{n}\epsilon$. Applying Claim 1.4, the error for the normalized vector $|R_j\rangle$ can be bounded as $\left\||R_j\rangle - |\overline{R_j}\rangle\right\| < \frac{\sqrt{2n}\epsilon}{\|R_j\|}$. By the Cauchy-Schwarz inequality we have that $\|R_j\| \geq \frac{\sum_i^n r_{ij}}{\sqrt{n}}$. We can use this to obtain a bound $\frac{\sqrt{n}}{\|R_j\|} < \frac{\sqrt{n}}{\sum_i r_{ij}}\sqrt{n} = O(1/k)$, using the dataset assumptions in the main manuscript. If we choose ϵ such that $\frac{\sqrt{2n\epsilon}}{\|R_j\|} < \epsilon_1$, that is $\epsilon \leq \epsilon_1/k$ then our runtime becomes $T_{R_2,\epsilon_1} := \widetilde{O}(k^2 \times T_{R_1,\epsilon_1})$.

□

Lemma 1.13 (Computing θ^{t+1})**.** *We assume quantum access to a GMM with parameters γ^t and let $\delta_\theta > 0$ be a precision parameter. There exists an algorithm that estimates $\overline{\theta}^{t+1} \in \mathbb{R}^k$ such that $\left\|\overline{\theta}^{t+1} - \theta^{t+1}\right\| \leq \delta_\theta$ in time*

$$T_\theta = O\left(k^{3.5}\eta^{1.5}\frac{\kappa^2(\Sigma)\mu(\Sigma)}{\delta_\theta^2}\right)$$

Proof. An estimate of θ_j^{t+1} can be recovered from the following operations. First, we use Lemma 3.3 (part 1) to compute the responsibilities to error ϵ_1, and then perform the following mapping, which consists of a controlled rotation on an auxiliary qubit:

$$\frac{1}{\sqrt{nk}}\sum_{\substack{i=1\\j=1}}^{n,k} |i\rangle\,|j\rangle\,|\overline{r_{ij}}^t\rangle \mapsto$$

$$\frac{1}{\sqrt{nk}}\sum_{\substack{i=1\\j=1}}^{n,k} |i\rangle\,|j\rangle\left(\sqrt{\overline{r_{ij}}^t}\,|0\rangle + \sqrt{1-\overline{r_{ij}}^t}\,|1\rangle\right) \tag{10}$$

The previous operation has a cost of T_{R_1,ϵ_1}, and the probability of getting $|0\rangle$ is $p(0) = \frac{1}{nk}\sum_{i=1}^n\sum_{j=1}^k r_{ij}^t = \frac{1}{k}$.

Recall that $\theta_j^{t+1} = \frac{1}{n}\sum_{i=1}^n r_{ij}^t$ by definition. Let $Z_j = \sum_{i=1}^n \overline{r_{ij}}^t$ and define state $|\sqrt{R_j}\rangle = \left(\frac{1}{\sqrt{Z_j}}\sum_{i=1}^n \sqrt{\overline{r_{ij}}^t}\,|i\rangle\right)|j\rangle$. After amplitude amplification on $|0\rangle$ we have the state,

$$\begin{aligned}|\sqrt{R}\rangle &:= \frac{1}{\sqrt{n}}\sum_{\substack{i=1\\j=1}}^{n,k}\sqrt{\overline{r_{ij}}^t}\,|i\rangle\,|j\rangle\\ &= \sum_{j=1}^k\sqrt{\frac{Z_j}{n}}\left(\frac{1}{\sqrt{Z_j}}\sum_{i=1}^n\sqrt{\overline{r_{ij}}^t}\,|i\rangle\right)|j\rangle\\ &= \sum_{j=1}^k\sqrt{\overline{\theta_j}^{t+1}}\,|\sqrt{R_j}\rangle\,.\end{aligned} \tag{11}$$

The probability of obtaining outcome $|j\rangle$ if the second register is measured in the standard basis is $p(j) = \overline{\theta_j}^{t+1}$.

An estimate for θ_j^{t+1} with precision ϵ can be obtained by either sampling the last register, or by performing amplitude estimation to estimate each of the values θ_j^{t+1} for $j \in [k]$. Sampling requires $O(\epsilon^{-2})$ samples by the Chernoff bounds, but does not incur any dependence on k. In this case, as the number of cluster k is relatively small compared to $1/\epsilon$, we chose to do amplitude estimation to estimate all θ_j^{t+1} for $j \in [k]$ to error $\epsilon/\sqrt{k}$ in time,

$$T_\theta := O\left(k \cdot \frac{\sqrt{k}T_{R_1,\epsilon_1}}{\epsilon}\right). \tag{12}$$

Let's analyze the error in the estimation of θ_j^{t+1}. For the error due to responsibility estimation by Lemma 3.3 we have $|\overline{\theta_j}^{t+1} - \theta_j^{t+1}| = \frac{1}{n}\sum_i|\overline{r_{ij}}^t - r_{ij}^t| \leq \epsilon_1$ for all $j \in [k]$, implying that $\left\|\overline{\theta}^{t+1} - \theta^{t+1}\right\| \leq \sqrt{k}\epsilon_1$. The total error in ℓ_2 norm due to Amplitude estimation is at most ϵ as it estimates each coordinate of $\overline{\theta_j}^{t+1}$ to error $\epsilon/\sqrt{k}$.

Using the triangle inequality, we have the total error is at most $\epsilon + \sqrt{k}\epsilon_1$. As we require that the final error be upper bounded as $\left\|\overline{\theta}^{t+1} - \theta^{t+1}\right\| < \delta_\theta$, we choose parameters $\sqrt{k}\epsilon_1 < \delta_\theta/2 \Rightarrow \epsilon_1 < \frac{\delta_\theta}{2\sqrt{k}}$ and $\epsilon < \delta_\theta/2$. With these parameters, the overall running time of the quantum procedure is $T_\theta = O(k^{1.5}\frac{T_{R_1,\epsilon_1}}{\epsilon}) = O\left(k^{3.5}\frac{\eta^{1.5}\cdot\kappa^2(\Sigma)\mu(\Sigma)}{\delta_\theta^2}\right)$.

□

Lemma 1.14 (Computing $\boldsymbol{\mu}_j^{t+1}$)**.** *We assume we have quantum access to a GMM with parameters γ^t. For a precision parameter $\delta_\mu > 0$, there is a quantum algorithm that calculates $\{\overline{\mu_j}^{t+1}\}_{j=1}^k$ such that for all $j \in [k]$ $\left\|\overline{\mu_j}^{t+1} - \mu_j^{t+1}\right\| \leq \delta_\mu$ in time*

$$T_\mu = \widetilde{O}\left(\frac{kd\eta\kappa(V)(\mu(V) + k^{3.5}\eta^{1.5}\kappa^2(\Sigma)\mu(\Sigma))}{\delta_\mu^3}\right)$$

Proof. The new centroid μ_j^{t+1} is estimated by first creating an approximation of the state $|R_j^t\rangle$ up to error ϵ_1 in the ℓ_2-norm using part 2 of Lemma 3.3. We then use the quantum linear algebra algorithms in Theorem 1.8 to multiply R_j by V^T, and compute a state $|\overline{\mu_j}^{t+1}\rangle$ along with an estimate for the norm $\left\|V^TR_j^t\right\| = \left\|\overline{\mu_j}^{t+1}\right\|$ with error ϵ_{norm}. The last step of the algorithm consists in estimating the unit vector $|\overline{\mu_j}^{t+1}\rangle$ with precision ϵ_{tom} using tomography. Considering that the tomography depends on d, which we expect to be bigger than the precision required by the norm estimation, we can assume that the runtime of the norm estimation is absorbed. Thus, we obtain: $\widetilde{O}\left(k\frac{d}{\epsilon_{tom}^2}\cdot\kappa(V)\left(\mu(V) + T_{R_2,\epsilon_1}\right)\right)$.

Let's now analyze the total error in the estimation of the new centroids, which we want to be δ_μ. For this purpose, we use Claim 1.5, and choose parameters such that $2\sqrt{\eta}(\epsilon_{tom} + \epsilon_{norm}) = \delta_\mu$. Since the error ϵ_3 for quantum linear algebra appears as a logarithmic factor in the running time, we can choose $\epsilon_3 \ll \epsilon_{tom}$ without affecting the runtime.

Let $\overline{\mu}$ be the classical unit vector obtained after quantum tomography, and $\widehat{|\mu\rangle}$ be the state produced by the quantum linear algebra procedure starting with an approximation of $|R_j^t\rangle$. Using the triangle inequality we have $\left\| |\mu\rangle - \overline{\mu} \right\| < \left\| \overline{\mu} - \widehat{|\mu\rangle} \right\| + \left\| \widehat{|\mu\rangle} - |\mu\rangle \right\| < \epsilon_{tom} + \epsilon_1 < \delta_\mu/2\sqrt{\eta}$. The errors for the norm estimation procedure can be bounded similarly as $| \, \|\mu\| - \overline{\|\mu\|} | < | \, \|\mu\| - \widehat{\|\mu\|} | + | \overline{\|\mu\|} - \widehat{\|\mu\|} | < \epsilon_{norm} + \epsilon_1 \le \delta_\mu/2\sqrt{\eta}$. We therefore choose parameters $\epsilon_{tom} = \epsilon_1 = \epsilon_{norm} \le \delta_\mu/4\sqrt{\eta}$. Since the amplitude estimation step we use for estimating the norms does not depends on d, which is expected to dominate the other parameters, we omit the amplitude estimation step. Substituting for T_{R_2,δ_μ}, we have the more concise expression for the running time of:

$$\widetilde{O}\left(\frac{kd\eta\kappa(V)(\mu(V) + k^{3.5}\eta^{1.5}\kappa^2(\Sigma)\mu(\Sigma))}{\delta_\mu^3}\right) \tag{13}$$

□

Lemma 1.15 (Computing Σ_j^{t+1})**.** *We assume we have quantum access to a GMM with parameters γ^t. We also have computed estimates $\overline{\mu_j}^{t+1}$ of all centroids such that $\left\|\overline{\mu_j}^{t+1} - \mu_j^{t+1}\right\| \le \delta_\mu$ for precision parameter $\delta_\mu > 0$. Then, there exists a quantum algorithm that outputs estimates for the new covariance matrices $\{\overline{\Sigma}_j^{t+1}\}_{j=1}^k$ such that $\left\|\Sigma_j^{t+1} - \overline{\Sigma}_j^{t+1}\right\|_F \le \delta_\mu\sqrt{\eta}$ with high probability, in time,*

$$T_\Sigma := \widetilde{O}\Big(\frac{kd^2\eta\kappa^2(V)(\mu(V') + \eta^2 k^{3.5}\kappa^2(\Sigma)\mu(\Sigma))}{\delta_\mu^3}\Big)$$

Proof. It is simple to check, that the update rule of the covariance matrix during the maximization step can be reduced to (Murphy, 2012, Exercise 11.2):

$$\Sigma_j^{t+1} \leftarrow \frac{\sum_{i=1}^n r_{ij}(v_i - \mu_j^{t+1})(v_i - \mu_j^{t+1})^T}{\sum_{i=1}^n r_{ij}} =$$
$$= \frac{\sum_{i=1}^n r_{ij}v_i v_i^T}{n\theta_j} - \mu_j^{t+1}(\mu_j^{t+1})^T = \Sigma'_j - \mu_j^{t+1}(\mu_j^{t+1})^T \tag{14}$$

First, note that we can use the estimates of the centroids to compute $\mu_j^{t+1}(\mu_j^{t+1})^T$ with error $\delta_\mu \|\mu\| \le \delta_\mu\sqrt{\eta}$ in the update rule for the Σ_j. This follows from the fact that $\overline{\mu} = \mu + e$ where e is a vector of norm δ_μ. Therefore $\left\|\mu\mu^T - \overline{\mu}\,\overline{\mu}^T\right\| < 2\sqrt{\eta}\delta_\mu + \delta_\mu^2 \le 3\sqrt{\eta}\delta_\mu$. It follows that we can allow an error of $\sqrt{\eta}\delta_\mu$ also for the left term in the definition of Σ_j^{t+1}. Let's discuss the procedure for estimating Σ'_j in Eq. (14). Note that $\text{vec}[\Sigma'_j] = (V')^T R_j$, so we use quantum matrix multiplication to estimate $|\text{vec}[\Sigma'_j]\rangle$ and $\left\|\text{vec}[\Sigma'_j]\right\|$. As the runtime for the norm estimation $\frac{\kappa(V')(\mu(V') + T_{R_2,\epsilon_1}))\log(1/\epsilon_{mult})}{\epsilon_{norms}}$ does not depend on d, we consider it smaller than the runtime for performing tomography. Thus, the runtime for this operation is:

$$O(\frac{d^2\log d}{\epsilon_{tom}^2}\kappa(V')(\mu(V') + T_{R_2,\epsilon_1}))\log(1/\epsilon_{mult})).$$

Let's analyze the error of this procedure. We want a matrix $\overline{\Sigma'_j}$ that is $\sqrt{\eta}\delta_\mu$-close to the correct one: $\left\|\overline{\Sigma'_j} - \Sigma'_j\right\|_F = \left\|\text{vec}\overline{[\Sigma'_j]} - \text{vec}[\Sigma'_j]\right\|_2 < \sqrt{\eta}\delta_\mu$. Again, the error due to matrix multiplication can be taken as small as necessary, since is inside a logarithm. From Claim 1.5, we just need to fix the error of tomography and norm estimation such that $\eta(\epsilon_{unit} + \epsilon_{norms}) < \sqrt{\eta}\delta_\mu$ where we have used η as an upper bound on $\|\Sigma_j\|_F$. For the unit vectors, we require $\left\||\Sigma'_j\rangle - \overline{|\Sigma'_j\rangle}\right\| \le \left\|\overline{|\Sigma'_j\rangle} - \widehat{|\Sigma'_j\rangle}\right\| + \left\|\widehat{|\Sigma'_j\rangle} - |\Sigma'_j\rangle\right\| < \epsilon_{tom} + \epsilon_1 \le \delta_\mu/2\sqrt{\eta}$, where $\overline{|\Sigma'_j\rangle}$ is the error due to tomography and $\widehat{|\Sigma'_j\rangle}$ is the error due to Lemma 3.3. For this inequality to be true, we choose $\epsilon_{tom} = \epsilon_1 < \delta_\mu/4\sqrt{\eta}$.

The same argument applies to estimating the norm $\|\Sigma'_j\|$ with relative error : $| \, \|\Sigma'_j\| - \overline{\|\Sigma'_j\|} | \le | \overline{\|\Sigma'_j\|} - \widehat{\|\Sigma'_j\|} | + | \widehat{\|\Sigma'_j\|} - \|\Sigma'_j\| \, | < \epsilon + \epsilon_1 \le \delta_\mu/2\sqrt{\eta}$ (where here ϵ is the error of the amplitude estimation step used in Theorem 1.8 and ϵ_1 is the error in calling Lemma 3.3. Again, we choose $\epsilon = \epsilon_1 \le \delta_\mu/4\sqrt{\eta}$. Note that $\kappa(V') \le \kappa^2(V)$. This can be derived from the fact that $\kappa(A \otimes B) = \kappa(A)\kappa(B)$, $\kappa(AB) \le \kappa(A)\kappa(B)$, and

$$V' := \begin{pmatrix} [e_1 \otimes e_1]^T \\ \vdots \\ [e_n \otimes e_n]^T \end{pmatrix} (V \otimes V).$$

Since the tomography is more costly than the amplitude estimation step, we can disregard the runtime for the norm estimation step. As this operation is repeated k times for the k different covariance matrices, the total runtime of the whole algorithm is given by $\widetilde{O}\Big(\frac{kd^2\eta\kappa^2(V)(\mu(V') + \eta^2 k^{3.5}\kappa^2(\Sigma)\mu(\Sigma))}{\delta_\mu^3}\Big)$.

Let us also note that for each of new computed covariance matrices, we use Lemma **??** to compute an estimate for their log-determinant and this time can be absorbed in the time T_Σ.

□

Lemma 1.16 (Quantum estimation of likelihood)**.** *We assume we have quantum access to a GMM with parameters γ^t. For $\epsilon_\tau > 0$, there exists a quantum algorithm that* *estimates* $\mathbb{E}[p(v_i;\gamma^t)]$ *with absolute error* ϵ_τ *in time*

$$T_\ell = \widetilde{O}\left(k^{1.5}\eta^{1.5}\frac{\kappa^2(\Sigma)\mu(\Sigma)}{\epsilon_\tau^2}\right)$$

Proof. We obtain the likelihood from the ability to compute the value of a Gaussian distribution and quantum arithmetic. Using the mapping of Lemma 3.2 with precision ϵ_1, we can compute $\phi(v_i|\mu_j,\Sigma_j)$ for all the Gaussians, that is $|i\rangle \bigotimes_{j=0}^{k-1}|j\rangle\,|\overline{p(v_i|j;\gamma_j)}\rangle$. Then, by knowing θ, and by using quantum arithmetic we can compute in a register the mixture of Gaussian's: $p(v_i;\gamma) = \sum_{j\in[k]}\theta_j p(v_i|j;\gamma)$. We now drop the notation for the model γ and write $p(v_i)$ instead of $p(v_i;\gamma)$. Doing the previous calculations quantumly leads to the creation of the state $|i\rangle\,|p(v_i)\rangle$. We perform the mapping $|i\rangle\,|p(v_i)\rangle \mapsto |i\rangle\left(\sqrt{p(v_i)}\,|0\rangle + \sqrt{1-p(v_i)}\,|1\rangle\right)$ and estimate $p(|0\rangle) \simeq \mathbb{E}[p(v_i)]$ with amplitude estimation on the ancilla qubit. To get a ϵ_τ-estimate of $p(0)$ we need to decide the precision parameter we use for estimating $\overline{p(v_i|j;\gamma)}$ and the precision required by amplitude estimation. Let $\overline{p(0)}$ be the ϵ_1-error introduced by using Lemma 3.2 and $\widehat{p(0)}$ the error introduced by amplitude estimation. Using triangle inequality we set $\left\|p(0)-\widehat{p(0)}\right\| < \left\|\widehat{p(0)}-\overline{p(0)}\right\| + \left\|\overline{p(0)}-p(0)\right\| < \epsilon_\tau$.

To have $|p(0)-\widehat{p(0)}| < \epsilon_\tau$, we should set ϵ_1 such that $|\overline{p(0)}-p(0)| < \epsilon_\tau/4$, and we set the error in amplitude estimation and in the estimation of the probabilities to be $\epsilon_\tau/2$. The runtime of this procedure is therefore:

$$\widetilde{O}\left(k\cdot T_{G,\epsilon_\tau}\cdot\frac{1}{\epsilon_\tau\sqrt{p(0)}}\right) = \widetilde{O}\left(k^{1.5}\eta^{1.5}\cdot\frac{\kappa(\Sigma)\mu(\Sigma)}{\epsilon_\tau^2}\right)$$

□

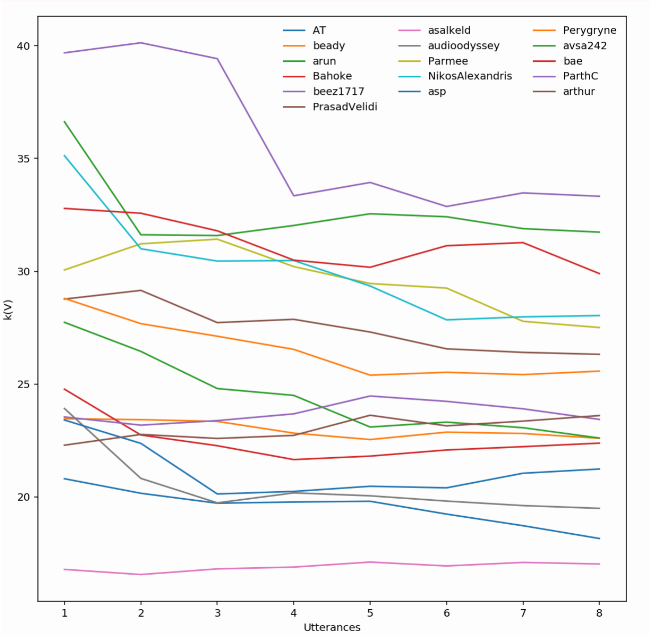
1.5. Experiments

We used a subset of the voices that can be found on VoxForge (Voxforge.org). The training set consist in at 5 speech utterances from 38 speakers. An utterance is a wav audio clips of a few seconds of voice speech. In order to perform speech recognition on raw wav audio files, we need to proceed with classical feature extraction procedures. In the speech recognition community is common to extract from audio the Mel Frequency Cepstrum Coefficients (MFCCs) features (Reynolds et al., 2000), and we followed the same approach. We selected $d = 40$ features for each speaker. This classical procedure, takes as input an audio file, and return a matrix where each row is a vector in $\mathbb{R}^{40}$, representing a small window of audio file of a few milliseconds. Due to the dissimilarities in the speakers' audio data, the different dataset $V_1 \ldots V_{38}$ are made of a variable number of points which ranges from $n = 2000$ to 4000. Then, each speaker is modeled with a mixture of 16 Gaussians with diagonal covariance matrix. The test set consists of other 5 (or more) unseen utterances of the same 38 speakers. The task is to correctly label the unseen utterances with the name of the correct speaker. This is done by testing each of the GMM fitted during the training against the new test sample. The selected model is the one with the highest likelihood. In the experiments, we compared the performances of classical and quantum algorithm, and measured the relevant parameters that govern the runtime of it. We used scikit-learn (Pedregosa et al., 2011) to run all the experiments.

We also simulated the impact of noise during the training of the the GMM fitted with ML estimate, so to assure the convergence of the quantum algorithm. For almost all GMM fitted using 16 diagonal covariance matrices, there is at least a Σ_j with bad condition number (i.e up to 2500 circa). As in (Kerenidis et al., 2019a; Kerenidis & Luongo, 2020) we took a threshold on the matrix by discarding singular values smaller than a certain value. Practically, we discarded any singular value smaller than 0.07. In the experiment, thresholding the covariance matrices not only did not make the accuracy worse, but had also a positive impact on it, perhaps because it has a regularizing effect on the model. Each of the model γ^t estimated with ML estimate has been perturbed at each iteration. For each model, the perturbation consists of three things. First we add to each of the components of θ some noise from the truncated Gaussian distribution centered in θ_i in the interval $(\theta_i - \delta_\theta/\sqrt{k}, \theta_i + \delta_\theta/\sqrt{k})$ with unit variance. This can guarantee that overall, the error in the vector of the mixing weights is smaller than δ_θ. Then we perturb each of the components of the centroids μ_j with Gaussian noise centered in $(\mu_j)_i$ on the interval $((\mu_j)_i - \frac{\delta_\mu}{\sqrt{d}}), (\mu_j)_i + \frac{\delta_\mu}{\sqrt{d}})$. Similarly, we perturbed also the diagonal matrices Σ_j with a vector of norm smaller than $\delta_\mu\sqrt{\eta}$, where $\eta = 10$. As we are using a diagonal GMM, this reduces to perturbing each singular value with Gaussian noise from a truncated Gaussian centered Σ on the interval $((\Sigma_j)_{ii} - \delta_\mu\sqrt{\eta}/\sqrt{d}, (\Sigma_j)_{ii} + \delta_\mu\sqrt{\eta}/\sqrt{d})$. Then, we made sure that each of the singular values stays positive, as covariance matrices are positive definite. Last, the matrices are thresholded, i.e. the eigenvalues smaller than a certain threshold σ_τ are set to σ_τ. This is done in order to make sure that the effective condition number $\kappa(\Sigma)$ is no bigger than a threshold κ_τ. With a $\epsilon_\tau = 7\times 10^{-3}$ and 70 iterations per initialization, all runs of the 7 different initialization of classical and quantum EM converged. Once the training has terminated, we measured all the values of $\kappa(\Sigma), \kappa(V), \mu(V), \mu(\Sigma), \log\det(\Sigma)$ for both ML and for MAP estimate. The results are in the Table on the main manuscript. Notably, taking a threshold on the Σ_j help to mitigate the errors of noise as it regularized the model. In fact, using classical EM with ML estimation, we reached an accuracy of 97.1%. With parameters of $\delta_\mu = 0.5$, $\delta_\theta = 0.038$, and a threshold on the condition number of the covariance matrices of Σ_j of 0.07, we reached an accuracy of 98.7%. [1]

We further analyzed experimentally the evolution of the condition number $\kappa(V_i)$ while adding vectors from all the utterances of the speakers to the training set V_i. As we can see from Figure 1, all the condition numbers are pretty stable and do not increase by adding new vectors to the various training sets $V_1, \dots V_n$.

Figure 1. Evolution of $\kappa(V_i)$ where V_i is the data matrix obtained by all the utterances available from the i-th speaker to the training set. For all the different speaker, the condition number of the matrix V_i is stable, and does not increase while adding vectors to the training set.



We leave for future work the task of testing the algorithm with further experiments (i.e. bigger and different types of datasets), and further optimizations, like procedures for hyperparameter tuning.

1.6. Quantum MAP estimate of GMM

Maximum Likelihood is not the only way to estimate the parameters of a model, and in certain cases might not even be the best one. For instance, in high-dimensional spaces, it is pretty common for ML estimates to overfit. Moreover, it is often the case that we have prior information on the distribution of the parameters, and we would like our models to take this information into account. These issues are often addressed using a Bayesian approach, i.e. by using a so-called Maximum A Posteriori estimate (MAP) of a model (Murphy, 2012, Section 14.4.2.8). MAP estimates work by assuming the existence of a *prior* distribution over the parameters γ. The posterior distribution we use as objective function to maximize comes from the Bayes' rule applied on the likelihood, which gives the posterior as a product of the likelihood and the prior, normalized by the evidence. More simply, we use the Bayes' rule on the likelihood function, as $p(\gamma; V) = \frac{p(V;\gamma)p(\gamma)}{p(V)}$. This allows us to treat the model γ as a random variable, and derive from the ML estimate a MAP estimate:

$$\gamma^*_{MAP} = \arg\max_{\gamma} \sum_{i=1}^{n} \log p(\gamma|v_i) \qquad (15)$$

Among the advantages of a MAP estimate over ML is that it avoids overfitting by having a kind of regularization effect on the model (Murphy, 2012, Section 6.5). Another feature consists in injecting into a maximum likelihood model some external information, perhaps from domain experts. This advantage comes at the cost of requiring "good" prior information on the problem, which might be non-trivial. In terms of labelling, a MAP estimates correspond to a *hard clustering*, where the label of the point v_i is decided according to the following rule:

$$y_i = \arg\max_j r_{ij} = \arg\max_j \log p(v_i|y_i = j; \gamma) + \log p(y_i = j; \gamma) \qquad (16)$$

Deriving the previous expression is straightforward using the Bayes' rule, and by noting that the softmax is rank-preserving, and we can discard the denominator of r_{ij} - since it does not depend on γ - and it is shared among all the other responsibilities of the points v_i. Thus, from Equation 15 we can conveniently derive Equation 16 as a proxy for the label. Fitting a model with MAP estimate is commonly done via the EM algorithm as well. The Expectation step of EM remains unchanged, but the update rules of the Maximization step are slightly different. In this work we only discuss the GMM case, for the other cases the interested reader is encouraged to see the relevant literature. For GMM, the prior on the mixing weight is often modeled using the Dirichlet distribution, that is $\theta_j \sim \text{Dir}(\boldsymbol{\alpha})$. For the rest of parameters, we assume that the conjugate prior is of the form $p(\mu_j, \Sigma_j) = NIW(\mu_j, \Sigma_j|\boldsymbol{m}_0, \iota_0, \nu_0, \boldsymbol{S}_0)$, where NIW$(\mu_j, \Sigma_j)$ is the Normal-inverse-Wishart distribution. The probability density function of the NIW is the product between a multivariate normal $\phi(\mu|m_0, \frac{1}{\iota}\Sigma)$ and a inverse Wishart distribution $\mathcal{W}^{-1}(\Sigma|\boldsymbol{S}_0, \nu_0)$. NIW has as support vectors μ with mean μ_0 and covariance matrices $\frac{1}{\iota}\Sigma$ where Σ is a random variable with inverse Wishart

[1] The experiments has been improved upon a previous version of this work, by adding more data, adding the noise during the training procedure, and finding better hyperparameters.

distribution over positive definite matrices. NIW is often the distribution of choice in these cases, as is the conjugate prior of a multivariate normal distribution with unknown mean and covariance matrix. A shorthand notation, let's define $r_j = n\theta_j = \sum_{i=1}^{n} r_{ij}$. As in (Murphy, 2012), we also denote with $\overline{x_j}^{t+1}$ and $\overline{S_j}^{t+1}$ the Maximum Likelihood estimate of the parameters $(\mu_j^{t+1})_{ML}$ and $(\Sigma_j^{t+1})_{ML}$. For MAP, the update rules are the following:

$$\theta_j^{t+1} \leftarrow \frac{r_j + \alpha_j - 1}{n + \sum_j \alpha_j - k} \tag{17}$$

$$\mu_j^{t+1} \leftarrow \frac{r_j \overline{x_j}^{t+1} + \iota_0 \boldsymbol{m}_0}{r_j + \iota_0} \tag{18}$$

$$\Sigma_j^{t+1} \leftarrow \frac{\boldsymbol{S}_0 + \overline{S_j}^{t+1} + \frac{\iota_0 r_j}{\iota_0 + r_j}(\overline{x_j}^{t+1} - \boldsymbol{m}_0)(\overline{x_j}^{t+1} - \boldsymbol{m}_0)^T}{\nu_0 + r_k + d + 2} \tag{19}$$

Where the matrix $\boldsymbol{S}_0$ is defined as:

$$\boldsymbol{S}_0 := \frac{1}{k^{1/d}} Diag(s_1^2, \cdots, s_d^2), \tag{20}$$

where each value s_j is computed as $s_j := \frac{1}{n}\sum_{i=1}^{n}(x_{ij} - \sum_{i=1}^{n} x_{ij}))^2$ which is the pooled variance for each of the dimension j. For more information on the advantages, disadvantages, and common choice of parameters of a MAP estimate, we refer the interested reader to (Murphy, 2012). Using the QEM algorithm to fit a MAP estimate is straightforward, since once the ML estimate of the parameter is recovered from the quantum procedures, the update rules can be computed classically.

Corollary 1.17 (QEM for MAP estimates of GMM)**.** *We assume we have quantum access to a GMM with parameters γ^t. For parameters $\delta_\theta, \delta_\mu, \epsilon_\tau > 0$, the running time of one iteration of the Quantum Maximum A Posteriori (QMAP) algorithm algorithm is*

$$O(T_\theta + T_\mu + T_\Sigma + T_\ell),$$

for

$$\begin{aligned}
T_\theta &= \widetilde{O}\left(k^{3.5}\eta^{1.5}\frac{\kappa^2(\Sigma)\mu(\Sigma)}{\delta_\theta^2}\right) \\
T_\mu &= \widetilde{O}\left(\frac{kd\eta\kappa(V)(\mu(V) + k^{3.5}\eta^{1.5}\kappa^2(\Sigma)\mu(\Sigma))}{\delta_\mu^3}\right) \\
T_\Sigma &= \widetilde{O}\Big(\frac{kd^2\eta\kappa^2(V)(\mu(V') + \eta^2 k^{3.5}\kappa^2(\Sigma)\mu(\Sigma))}{\delta_\mu^3}\Big) \\
T_\ell &= \widetilde{O}\left(k^{1.5}\eta^{1.5}\frac{\kappa^2(\Sigma)\mu(\Sigma)}{\epsilon_\tau^2}\right)
\end{aligned}$$

For the range of parameters of interest, the running time is dominated by T_Σ.

References

Arthur, D. and Vassilvitskii, S. k-means++: The advantages of careful seeding. In *Proceedings of the eighteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 1027–1035. Society for Industrial and Applied Mathematics, 2007.

Biernacki, C., Celeux, G., and Govaert, G. Choosing starting values for the EM algorithm for getting the highest likelihood in multivariate gaussian mixture models. *Computational Statistics & Data Analysis*, 41(3-4):561–575, 2003.

Blömer, J. and Bujna, K. Simple methods for initializing the EM algorithm for gaussian mixture models. *CoRR*, 2013.

Brassard, G., Høyer, P., Mosca, M., and Tapp, A. Quantum Amplitude Amplification and Estimation. *Contemporary Mathematics*, 305, 2002.

Celeux, G. and Govaert, G. A classification EM algorithm for clustering and two stochastic versions. *Computational statistics & Data analysis*, 14(3):315–332, 1992.

Chakraborty, S., Gilyén, A., and Jeffery, S. The power of block-encoded matrix powers: improved regression techniques via faster Hamiltonian simulation. *arXiv preprint arXiv:1804.01973*, 2018.

Gilyén, A., Su, Y., Low, G. H., and Wiebe, N. Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. *arXiv preprint arXiv:1806.01838*, 2018.

Kerenidis, I. and Luongo, A. Quantum classification of the MNIST dataset via Slow Feature Analysis. *Physical review letters A*, 101(6):062327, 2020.

Kerenidis, I. and Prakash, A. Quantum recommendation systems. *Proceedings of the 8th Innovations in Theoretical Computer Science Conference*, 2017.

Kerenidis, I. and Prakash, A. A quantum interior point method for LPs and SDPs. *arXiv:1808.09266*, 2018.

Kerenidis, I. and Prakash, A. Quantum gradient descent for linear systems and least squares. *Physical Review A*, 2020.

Kerenidis, I., Landman, J., Luongo, A., and Prakash, A. q-means: A quantum algorithm for unsupervised machine learning. In *Advances in Neural Information Processing Systems*, pp. 4136–4146, 2019a.

Kerenidis, I., Landman, J., and Prakash, A. Quantum algorithms for deep convolutional neural networks. *arXiv preprint arXiv:1911.01117*, 2019b.

Lloyd, S., Mohseni, M., and Rebentrost, P. Quantum algorithms for supervised and unsupervised machine learning. *arXiv*, 1307.0411:1–11, 7 2013. URL http://arxiv.org/abs/1307.0411.

Miyahara, H., Aihara, K., and Lechner, W. Quantum expectation-maximization algorithm. *Personal Communication*, 2019.

Murphy, K. P. *Machine learning: a probabilistic perspective*. MIT press, 2012.

Nielsen, M. A. and Chuang, I. Quantum computation and quantum information, 2002.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830, 2011.

Reynolds, D. A., Quatieri, T. F., and Dunn, R. B. Speaker verification using adapted gaussian mixture models. *Digital signal processing*, 10(1-3):19–41, 2000.

Rudin, W. et al. *Principles of mathematical analysis*, volume 3. McGraw-hill New York, 1964.

Voxforge.org. Free speech... recognition - voxforge.org. http://www.voxforge.org/. accessed 20/07/2019.

Wiebe, N., Kapoor, A., and Svore, K. M. Quantum Algorithms for Nearest-Neighbor Methods for Supervised and Unsupervised Learning. 2014. URL https://arxiv.org/pdf/1401.2142.pdf.