Quantum singular value estimation techniques for data representation

POLITECNICO
MILANO 1863

Armando Bellante

Advisor: Prof. Stefano Zanero

Co-Advisor: Dott. Alessandro Luongo
Prof. Maurizio Zamboni

School of Industrial and Information Engineering
Politecnico Di Milano

This dissertation is submitted for the degree of
Laurea Magistrale in Computer Science and Engineering

Politecnico Di Milano September 2020

To my family and friends.
To my grandparents.

Acknowledgements

Ringrazio di cuore tutte le persone che mi hanno accompagnato in questo viaggio intrapreso per passione, per amore di conoscenza e per curiosità delle infinite stranezze che questo universo ci rivela.

Per la realizzazione di questa tesi:

Ringrazio in particolar modo il mio relatore, Professor Stefano Zanero, per aver da subito supportato il mio interesse per la quantistica e per avermi guidato nell'intraprendere questo percorso di ricerca, ormai più di due anni fa.

Ringrazio con grande affetto Alessandro Luongo, che mi ha introdotto a questa materia. Il suo costante sostegno, la sua grande pazienza e disponibilità sono stati fondamentali per lo svolgimento di questo lavoro.

Ringrazio il Professor Maurizio Zamboni per aver condiviso questa tesi in qualità di co-relatore del Politecnico di Torino.

Sono grato a Patrick Rebentrost e al Professor Dagomir Kaszlikowski per i loro preziosi insegnamenti durante il periodo di studio trascorso alla National University of Singapore.

Desidero ringraziare Flavia Petruso, la mia compagna di corse sul lungomare di Capogallo durante questo periodo di Emergenza Covid-19, per le stimolanti conversazioni sull'argomento e il supporto morale in questi mesi.

Grazie anche a William Bonvini, a mio fratello Emanuele Bellante e a Giulio Gambino per avermi dato una mano durante la revisione di questo elaborato.

Un ultimo pensiero affettuoso va a Noemi Gozzi per essermi stata vicina nei momenti più difficili di questa ricerca.

Un sentito ringraziamento ancora al Professor Stefano Zanero, al Prorettore Vicario, Professoressa Donatella Sciuto, e al Rettore, Professor Ferruccio Resta, per avermi dato l'opportunità di proseguire la ricerca in questo ambito con un Dottorato presso il Politecnico di Milano.

Abstract

Quantum computing is a recent computing paradigm that promises substantial speed-ups in several computational tasks that are impractical with classical computers. In this work, we focus on extending the recent technique of Quantum Singular Value Estimation (QSVE) to provide quantum procedures that speed-up the computation of data representation models for machine learning. Firstly, we show how the problems of performing Principal Component Analysis (PCA), Correspondence Analysis (CA), and Latent Semantic Analysis (LSA) are closely related to computing the Singular Value Decomposition (SVD) of matrices obtained from the dataset. Then, we present a series of novel quantum algorithms that allow us to estimate the singular values and singular vectors in time that is sub-linear in the number of elements of the matrices. For each of the quantum algorithms, the computational cost is proven. Furthermore, we discuss how these procedures can be used to extract the representation models for PCA, CA, and LSA. In each of these three cases, we analyze the parameters of the run-time and prove the error bounds on the accuracy of each representation model. Finally, we simulate some of the quantum algorithms on the MNIST dataset, in order to show that the run-time parameters that do not depend on the matrix dimensions are often negligible and that the error bound on the computed model is reasonable and allows for competitive classification performances.

Sommario

Il quantum computing è un modello di computazione innovativo che permette di ottenere vantaggiosi speed-up nella risoluzione di problemi che i computer classici non riuscirebbero a risolvere in tempi pratici. L'idea alla base di questo modello è quella di computare come farebbe la natura, sfruttando le curiose proprietà delle particelle fondamentali della materia. Recenti sviluppi in questo ambito hanno mostrato come questo paradigma possa essere utilizzato per velocizzare di fattori polinomiali o addirittura esponenziali il tempo di esecuzione di algoritmi per il machine learning. In questo lavoro, ci proponiamo di estendere le ultime ricerche di quantum linear algebra per creare algoritmi quantistici che calcolino modelli di rappresentazione di dati per il machine learning. In particolare, introduciamo tre nuove procedure quantistiche che permettono di accedere classicamente ai singular value, ad una misura della loro importanza e ai singular vector di una matrice. Gli algoritmi che permettono di estrarre i singular value e la loro importanza hanno una dipendenza poli-logartmica nel numero di elementi della matrice, mentre l'algoritmo per estrarre i singular vector, dipendendo linearmente dalla dimensione dei vettori da estrarre, fornisce un vantaggio polinomiale sul metodo di scomposizione tradizionale. Dopo aver dimostrato formalmente le complessità computazionali di questi algoritmi, illustriamo come questi possano essere impiegati per velocizzare metodi di rappresentazione di dati come *Principal Component Analysis*, *Correspondence Analysis* e *Latent Semantic Analysis*. Per ciascuno dei tre metodi, forniamo e dimostriamo dei limiti teorici sull'errore di approssimazione dei modelli ottenuti ed analizziamo i parametri del run-time che non dipendono dalle dimensioni della matrice di input. Infine, investighiamo la distribuzione dei singular value in tre dataset per il machine learning: CIFAR-10, MNIST 784 e Research papers. In tutti e tre i casi analizzati, l'importanza dei singular value decresce esponenzialmente, mostrando che un numero ristretto degli stessi è molto più importante degli altri. Un ultimo esperimento è stato effettuato sul MNIST 784, simulando l'intera procedura di quantum PCA introdotta in questo lavoro. I risultati mostrano che i limiti teorici sull'errore di approssimazione di PCA sono contenuti e consentono prestazioni di classificazione competitive. Inoltre, mostriamo che i parametri che non dipendono dalle dimensioni del dataset in input hanno una grandezza ragionevole.

Table of contents

Nomenclature

Roman Symbols

$\mathbf{A}$	bold capital letters denote matrices
$\mathbf{a}$	bold lowercase letters denote vectors
a	lowercase letters denote scalars
X	capital letters denote either sets or unitary operators
$\mathscr{X}$	cursive capital letters denote random variables
$\mathbf{U}$	matrix of left singular vectors
$\mathbf{V}$	matrix of right singular vectors
$\mathbf{u}$	left singular vector of a matrix
$\mathbf{v}$	right singular vector of a matrix
p	sum of factor score ratios of some singular values
r	rank of a matrix

Greek Symbols

δ	precision parameter for a vector's estimator
ε	precision parameter for a scalar's estimator
κ	condition number of a matrix, defined as $\frac{\sigma_{min}}{\sigma_{max}}$
Λ	diagonal matrix containing the eigenvalues of a matrix
λ	eigenvalue of a matrix

γ precision parameter for a scalar's estimator

Σ diagonal matrix containing the singular values of a matrix

σ singular value of a matrix

θ threshold for the lowest singular value

ζ_i number of times i has been measured over the total number of measurements

Subscripts

$\cdot, j$ j^{th} column vector of a matrix

$i, \cdot$ i^{th} row vector of a matrix

i or j i^{th} component of a vector or i^{th} column vector of a matrix

i, j entry at the i^{th} row and j^{th} column of a matrix

Superscripts

T transpose of a matrix

$\dagger$ complex conjugate transpose of a matrix or vector

$*$ complex conjugate of a complex number

(k) best k-rank approximation of a matrix according to Eckart–Young–Mirsky

Other Symbols

$| S |$ number of elements in the set S

$| x |$ absolute value of a real or module of a complex number

Σ_i^n sum from $i = 0$ to $i = n - 1$

δ_{ij} Dirac's delta: equal to 1 if $i = j$, equal to 0 otherwise

$diag([x_1, \cdots, x_n])$ $n \times n$ diagonal matrix with entries $[x_1, \cdots, x_n]$

$\mathbf{1}_d$ column vector containing d entries, all set to 1

$\mu(\mathbf{A})$ function of a matrix: it is lower than the Frobenius norm and ℓ_∞ norm (see Section 3.2.2)

$O()$ time complexity of an algorithm

$\tilde{O}()$ time complexity of an algorithm, omitting the polylogarithmic terms

Acronyms / Abbreviations

CA Correspondence Analysis

KNN K-Nearest Neighbors classifier

LSA Latent Semantic Analysis

LSI Latent Semantic Indexing

PCA Principal Component Analysis

PC Principal Component

PIC Principal Inertia Components

SVD Singular Value Decomposition

SVE Singular Value Estimation

Chapter 1

Introduction

Quantum computation is a novel computing paradigm that promises substantial speed-ups in a plethora of tasks that are computationally hard for classical computers. The idea of exploiting the laws of quantum mechanics to propose a new computational paradigm started spreading in the '80s thanks to some research works from Paul Benioff [4], Richard Feynman [14], and Jurij Manin [31]. Since then, researchers have worked on designing quantum algorithms that provide polynomial or exponential speed-ups over their classical counterparts.

In 2009, a work from Harrow, Hassidim, and Lloyd [18] gave birth to the field of Quantum Machine Learning, presenting quantum procedures that solve a system of linear equations $\mathbf{Ax} = \mathbf{b}$, with $\mathbf{A} \in \mathbb{C}^{n \times m}$ and $\mathbf{b} \in \mathbb{C}^n$, in time that scales only logarithmically on the size of the matrix $\mathbf{A}$. This result has promoted further research on solving optimization problems and linear algebra tasks using quantum computers, leading to algorithms that perform linear regressions [3], re-implement Support Vector Machines [35], or execute k-means [20] with an exponential advantage over the best currently known classical counterparts.

Another critical task in Machine Learning, that can benefit from the latest quantum linear algebra results, is learning data representations. When handling big data, it is crucial to learn effective data representations that reduce the noise of the dataset and help the learner algorithm perform better on the task. For instance, it is known that reducing the number of features of highly dimensional datasets decreases the variance of machine learning models and helps the learners avoid overfitting.

However, data representation methods for machine learning, such as Principal Component Analysis [34], Slow Feature Analysis [22], or Latent Semantic Analysis [11] heavily rely on the computation of the Singular Value Decomposition for large matrices. This task is often prohibitive to perform: the traditional algorithm for SVD scales super-linearly on the number of matrix elements and costs $O(min(n^2m, nm^2))$ where n and m are the numbers of rows and columns of the input matrix [37].

In the last few decades, extensive effort has been spent on designing randomized classical methods. Those methods compute an approximation of the SVD by performing calculations on a few random samples from the input matrix, scaling sub-linearly in the input size. Among the most notable results, there are the Frieze-Kannan-Vempala [15] algorithm and some recent quantum-inspired algorithms [8]. Even though some of these methods claim exponential speed-ups over the traditional algorithm, their time complexity scales super-linearly in the approximation error, the rank of the matrix, and its condition number, losing the exponential speed-ups in real case scenarios [2].

In this work, we extend the recent quantum linear algebra techniques for singular value estimation to provide quantum algorithms that output a classical representation of the top-k singular values and singular vectors of a matrix, with run-times that scale only linearly or quadratically in the precision parameters and the smallest singular value of the desired rank-k approximation. In particular, we present and prove:

- an algorithm that allows sampling the top-k singular values σ_i of a matrix, up to ε precision, with probability proportional to their magnitude and that provides a γ-close estimate of their importance $\frac{\sigma_i^2}{\sum_j \sigma_j^2}$ in time $O\left(\frac{1}{\gamma^2}\frac{\mu(\mathbf{A})}{\varepsilon} polylog(nm)\right)$, where $\mu(\mathbf{A})$ is bounded by the maximum between the Frobenius norm of $\mathbf{A}$ and the maximum ℓ_1 norm of its rows;

- an algorithm that, given a threshold θ, estimates, to relative error η, the importance $p = \frac{\sum_{i:\sigma_i \geq \theta} \sigma_i^2}{\sum_j \sigma_j^2}$ of the singular values greater than θ in time $O\left(\frac{\mu(\mathbf{A})}{\varepsilon}\frac{1}{\eta\sqrt{p}} polylog(nm)\right)$;

- an algorithm that allows retrieving the top-k left or right singular vectors, such that the difference vector between the estimate and the real vector has ℓ_2 norm lower than δ, in times $O\left(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{kn\log(n+k)}{\delta^2}\right)$ and $O\left(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{km\log(m+k)}{\delta^2}\right)$ respectively.

The first two algorithms provide an exponential speed-up with respect to the traditional methods, while the third one provides a polynomial speed-up.

We show how these novel quantum procedures can be used to perform hybrid classical-quantum computations for data representation algorithms such as Principal Component Analysis, Correspondence Analysis, and Latent Semantic Analysis. For each of these three cases, we provide theoretical bounds on the accuracy of the retrieved representation models and we discuss the magnitude of the run-time parameters.

Finally, we reproduce the MNIST classification task using a simulation of the new quantum algorithms, to study the accuracy of the classification and the run-time parameters.

The remainder of the work is organized as follows. In Chapter 2, we introduce the concepts of eigenvalues and eigenvectors, singular values and singular vectors, and show how the data representation models for PCA, CA, and LSA relate to the singular value decomposition of the input data matrix. Then, in Chapter 3, we provide an introduction to quantum computation and to the techniques of quantum linear algebra that support the novel algorithms. At the end of the chapter, we give an overview of the related works on singular value decomposition, both using quantum computation and classical computation. In Chapter 4, we present our main contributions. We state and prove the novel quantum algorithms for singular values and singular vectors extraction and provide error bounds on the retrieval of approximate representations for PCA, CA, and LSA. Lastly, in Chapter 5 we conduct experiments simulating the use of the novel quantum algorithms in PCA for the MNIST classification task. Chapter 6 discusses the main limitations of this research, showing future research directions. Chapter 7 summarizes the conclusions.

Chapter 2

Premises and notation

2.1 Eigenvalues, eigenvectors and eigendecomposition of a matrix

Real matrices are important tools in Machine Learning as they allow to comfortably represent data and describe the operations to perform during an algorithm. Eigenvectors and Eigenvalues are fundamental linear algebra concepts that provide important information about a matrix.

Definition 1 (Eigenvalue and Eigenvectors)**.** *[40, Section 6.1 page 289] Let* $\mathbf{A}$ *be a* $\mathbb{R}^{n\times n}$ *square matrix,* $\mathbf{q} \in \mathbb{R}^n$ *a non-zero vector and* λ *a scalar. If the following equation is satisfied*

$$\mathbf{A}\mathbf{q} = \lambda\mathbf{q},$$

then $\mathbf{q}$ *is said to be an eigenvector of matrix* $\mathbf{A}$ *and* λ *is its associated eigenvalue.*

To have a geometric insight into what eigenvectors and eigenvalues are, we can think of any matrix as a linear transformation in the $\mathbb{R}^n$ space. Under this light, we can say that the eigenvectors of a matrix are those vectors of the space that, after the transformation, lie on their original direction and only get their magnitude scaled by a certain factor: the eigenvalue.

The eigenvalues reveal interesting properties of a matrix. For example, the trace of a matrix (i.e. the sum of the element along the main diagonal of a square matrix) is the sum of its eigenvalues

$$Tr[\mathbf{A}] = \sum_{i}^{n} \lambda_i,$$

and its determinant is equal to the product of the eigenvalues [40, Section 6.1 page 294]

$$det(\mathbf{A}) = \prod_{i}^{n} \lambda_i.$$

Moreover, a matrix $\mathbf{A}$ with eigenvalues $\{\lambda_1, ..., \lambda_k\}$ has an inverse only if all the eigenvalues are not zero. The inverse has eigenvalues $\{\frac{1}{\lambda_1}, ..., \frac{1}{\lambda_k}\}$.

Generally, one eigenvalue can be associated with multiple eigenvectors. There might be a set of vectors $E(\lambda) \subseteq \mathbb{R}^n$ such that for all those vectors $\mathbf{q} \in E(\lambda) : \mathbf{Aq} = \lambda\mathbf{q}$. That is why for each eigenvalue we talk about an eigenspace.

Definition 2 (Eigenspace)**.** *[30, Definition 7.1.5 page 108] Let* $\mathbf{A}$ *be a* $\mathbb{R}^{n\times n}$ *square matrix and* λ *be an eigenvalue of* $\mathbf{A}$*. The eigenspace of* $\mathbf{A}$ *related to* λ *is the space defined over the set of vectors* $E(\lambda) = \{\mathbf{x} : \mathbf{Ax} = \lambda\mathbf{x}\}$.

For each eigenspace, through the Gram-Schmidt procedure, starting from linearly independent vectors it is possible to identify a set of orthogonal eigenvectors that constitute a basis for the space. The basis that spans the space where all the eigenvectors of a matrix lie is called eigenbasis.

Definition 3 (Eigenbasis)**.** *A basis for the space where all the eigenvectors of a matrix lie is called eigenbasis.*

An important result is that vectors in different eigenspaces are linearly independent.

Lemma 1 (Linear independence of eigenvectors)**.** *[30, Lemma 7.2.3 page 112] The set of vectors obtained by the union of the bases of the eigenspaces of a matrix is linearly independent.*

This means that if the sum of the dimensions of the eigenspaces $\sum_i dim(E(\lambda_i))$ equals n, it is possible to find n eigenvectors of $\mathbf{A}$ that form a basis for the $\mathbb{R}^n$ space. If that is the case, each vector that lies in $\mathbb{R}^n$ can be written as a linear combination of the eigenvectors of $\mathbf{A}$. Interestingly, matrices that have n linearly independent eigenvectors can be decomposed in terms of their eigenvalues and eigenvectors.

Theorem 1 (Eigendecomposition or Diagonalization)**.** *[40, Section 6.2 page 304] Let* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *be a square matrix with n linearly independent eigenvectors. Then, it is possible to decompose the matrix as*

$$\mathbf{A} = \mathbf{Q}\Lambda\mathbf{Q}^{-1}.$$

Where $\mathbf{Q} \in \mathbb{R}^{n \times n}$ *is an square matrix and* $\Lambda \in \mathbb{R}^{n \times n}$ *is a diagonal matrix. In particular, each* i^{th} *column of* $\mathbf{Q}$ *is an eigenvector of* $\mathbf{A}$ *and the* i^{th} *entry of* Λ *is its associated eigenvalue.*

The matrices that can be eigendecomposed are also said *diagonalizable*, as in practice the theorem above states that such matrices are *similar* to diagonal matrices. Unfortunately, not all the square matrices have enough independent eigenvectors to be diagonalized. The Spectral Theorem provides us with a set of matrices that can always be eigendecomposed.

Theorem 2 (Spectral theorem)**.** *[40, Spectral Theorem page 339] Every symmetric matrix is diagonalizable* $\mathbf{A} = \mathbf{Q}\Lambda\mathbf{Q}^{-1}$*. Furthermore, its eigenvalues are real and it is possible to choose the columns of* $\mathbf{Q}$ *so that it is an orthogonal matrix.*

Recall that a matrix $\mathbf{Q}$ is said to be orthogonal if $\mathbf{Q}\mathbf{Q}^T = \mathbf{Q}^T\mathbf{Q} = \mathbb{I}$, therefore $\mathbf{Q}^{-1} = \mathbf{Q}^T$. The Spectral theorem, together with the fact that matrices like $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}\mathbf{A}^T$ are symmetric, will come in handy in later discussions.

Being able to eigendecompose a matrix allows performing some computations faster than otherwise. Some examples of operations that gain speed-ups from the eigendecomposed representation are matrix inversion and matrix exponentiation. Indeed, if we have a matrix $\mathbf{A} = \mathbf{Q}\Lambda\mathbf{Q}^{-1}$ its inverse can be computed as $\mathbf{A}^{-1} = \mathbf{Q}\Lambda^{-1}\mathbf{Q}^{-1}$ where $\Lambda^{-1} = diag([\frac{1}{\lambda_1}, ...\frac{1}{\lambda_n}])$. It is easy to check that this matrix is the inverse of $\mathbf{A}$:

$$\mathbf{A}\mathbf{A}^{-1} = (\mathbf{Q}\Lambda\mathbf{Q}^{-1})(\mathbf{Q}\Lambda^{-1}\mathbf{Q}^{-1}) = \mathbf{Q}\Lambda\Lambda^{-1}\mathbf{Q}^{-1} = \mathbf{Q}\mathbf{Q}^{-1} = \mathbb{I}$$

$$\mathbf{A}^{-1}\mathbf{A} = (\mathbf{Q}\Lambda^{-1}\mathbf{Q}^{-1})(\mathbf{Q}\Lambda\mathbf{Q}^{-1}) = \mathbf{Q}\Lambda^{-1}\Lambda\mathbf{Q}^{-1} = \mathbf{Q}\mathbf{Q}^{-1} = \mathbb{I}.$$

At the same time, the eigendecomposition of a matrix allows performing matrix exponentiation much faster than through the usual matrix multiplication. In fact, it is true that $\mathbf{A}^p = \mathbf{Q}\Lambda^p\mathbf{Q}^{-1}$. For instance,

$$\mathbf{A}^3 = (\mathbf{Q}\Lambda\mathbf{Q}^{-1})(\mathbf{Q}\Lambda\mathbf{Q}^{-1})(\mathbf{Q}\Lambda\mathbf{Q}^{-1}) = \mathbf{Q}\Lambda(\mathbf{Q}^{-1}\mathbf{Q})\Lambda(\mathbf{Q}^{-1}\mathbf{Q})\Lambda\mathbf{Q}^{-1} = \mathbf{Q}\Lambda\Lambda\Lambda\mathbf{Q}^{-1} = \mathbf{Q}\Lambda^3\mathbf{Q}^{-1}.$$

Computing big matrix powers such as $\mathbf{A}^{100}$, with its eigendecomposed representation, only takes two matrix multiplications instead of a hundred.

Traditionally, the computational effort of performing the eigendecomposition of a $\mathbb{R}^{n \times n}$ matrix is in the order of $O(n^3)$ and may become prohibitive for large matrices [34].

2.2 Singular value decomposition

Eigenvalues and eigenvectors can be computed only on square matrices. Moreover, not all matrices can be eigendecomposed. For this reason, we introduce the concepts of *singular values* and *singular vectors*, that are closely related to the ones of eigenvalues and eigenvectors, and offer a decomposition for all the kind of matrices.

Theorem 3 (Singular Value Decomposition)**.** *[40, Sections 7.1, 7.2] Any matrix* $\mathbf{A} \in \mathbb{R}^{n \times n}$ *can be decomposed as*

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$$

where $\mathbf{U} \in \mathbb{R}^{n \times r}$ *and* $\mathbf{V} \in \mathbb{R}^{m \times r}$ *are orthogonal matrices and* $\Sigma \in \mathbb{R}^{r \times r}$ *is a diagonal matrix. In particular, each* i^{th} *column of* $\mathbf{U}$ *and* $\mathbf{V}$ *are respectively called left and right singular vectors of* $\mathbf{A}$ *and the* i^{th} *entry of* Σ *is their associated singular value. Furthermore,* r *is a natural number smaller then* m *and* n*.*

Similarly to how eigenvalues and eigenvectors have been defined previously, for each pair of left-right singular vector, and the associated singular value, the following equation stands:

$$\mathbf{A}\mathbf{v} = \sigma\mathbf{u}.$$

If we consider the Singular Value Decomposition (SVD) under a geometric perspective, we can see any linear transformation as the result of a rotation, a scaling, and another rotation. Indeed, if we compute the product between a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ and a vector $\mathbf{x} \in \mathbb{R}^m$

$$\mathbf{A}\mathbf{x} = \mathbf{U}\Sigma\mathbf{V}^T\mathbf{x} = (\mathbf{U}(\Sigma(\mathbf{V}^T\mathbf{x}))).$$

$\mathbf{U}$ and $\mathbf{V}^T$, being orthogonal matrices, only rotate the vector without changing its magnitude, while Σ, being a diagonal matrix, alters its length.

It is interesting to note that the singular values of $\mathbf{A}$ - denoted as $\{\sigma_1, ..., \sigma_r\}$ - are the square roots $\{\sqrt{\lambda_1}, ..., \sqrt{\lambda_r}\}$ of the eigenvalues of $\mathbf{A}\mathbf{A}^T$ (or $\mathbf{A}^T\mathbf{A}$) and that the left and right singular vectors of $\mathbf{A}$ - denoted as $\{\mathbf{u}_1, ..., \mathbf{u}_r\}$ and $\{\mathbf{v}_1, ..., \mathbf{v}_r\}$ - are respectively the eigenvectors of $\mathbf{A}\mathbf{A}^T$ and $\mathbf{A}^T\mathbf{A}$.

The fact that each matrix can be decomposed in terms of its singular vectors and singular values, as in the theorem above, makes the relationship between singular values - singular vectors of a matrix and eigenvalues - eigenvectors of its products with the transpose clearer:

$$\mathbf{A}\mathbf{A}^T = (\mathbf{U}\Sigma\mathbf{V}^T)(\mathbf{U}\Sigma\mathbf{V}^T)^T = \mathbf{U}\Sigma\mathbf{V}^T\mathbf{V}\Sigma\mathbf{U}^T = \mathbf{U}\Sigma^2\mathbf{U}^T;$$

$$\mathbf{A}^T\mathbf{A} = (\mathbf{U}\Sigma\mathbf{V}^T)^T(\mathbf{U}\Sigma\mathbf{V}^T) = \mathbf{V}\Sigma\mathbf{U}^T\mathbf{U}\Sigma\mathbf{V}^T = \mathbf{V}\Sigma^2\mathbf{V}^T.$$

Note that the matrices $\mathbf{A}\mathbf{A}^T$ and $\mathbf{A}^T\mathbf{A}$ are symmetric matrices and so, for the Spectral theorem, they can always be decomposed. Moreover, note that they have positive eigenvalues: being the square roots of real positive eigenvalues, the singular values of a real matrix are always real positive numbers.

As the left and right singular vectors are eigenvectors of symmetric matrices, they can be chosen to be orthogonal as well. In particular, the left singular vectors of a matrix span the row space of the matrix, and the right singular vectors span the column space.

Definition 4 (Column (Row) Space)**.** *[38, Definition 8.1 page 192] Let* $\mathbf{A}$ *be a* $\mathbb{R}^{n\times m}$ *matrix. The column (row) space of* $\mathbf{A}$ *is the space spanned by the column (row) vectors of* $\mathbf{A}$*. Its dimension is equal to the number of linearly independent columns (rows) of* $\mathbf{A}$*.*

The number r of singular values and singular vectors of a matrix is its rank.

Definition 5 (Rank of a matrix)**.** *[38, Definition 8.3, Proposition 8.4 page 193-194] The rank of a matrix is the number of linearly independent rows/columns of the matrix. If the matrix belongs to the* $\mathbb{R}^{n\times m}$ *space, the rank is less or equal than* $min(n,m)$*. A matrix is said to be full rank if its rank is equal to* $min(n,m)$*.*

2.2.1 Singular vectors for data representation

Singular values and singular vectors provide important information about matrices and allow to speed up certain kind of calculations. Many data analysis algorithms, such as Principal Component Analysis, Correspondence Analysis, and Latent Semantic Analysis that will be further investigated in the following sections, are based on the singular value decomposition of a matrix.

To begin with, the SVD representation of a matrix allows us to fastly calculate some matrix norms, like the ℓ_2 norm and the Frobenius norm.

Definition 6 (ℓ_2 (or Spectral) norm)**.** *[40, Section 11.2 page 518] Let* $\mathbf{A} \in \mathbb{R}^{n\times m}$ *be a matrix. The* ℓ_2 *norm of* $\mathbf{A}$ *is defined as* $||\mathbf{A}||_2 = max_{\mathbf{x}}\frac{||\mathbf{A}\mathbf{x}||}{||\mathbf{x}||}$*.*

It is pretty easy to see that

$$||\mathbf{A}||_2 = \sigma_{max}$$

Where σ_{max} is the greatest singular value of $\mathbf{A}$. In particular, if we consider again the matrix $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$ as a linear transformation, we see that $\mathbf{U}$ and $\mathbf{V}^T$ only rotate vectors $||\mathbf{U}\mathbf{x}|| = ||\mathbf{x}||$, $||\mathbf{V}\mathbf{x}|| = ||\mathbf{x}||$ while Σ changes their magnitude $||\Sigma\mathbf{x}|| \leq \sigma_{max}||\mathbf{x}||$. For this reason, the ℓ_2

Norm of a matrix is also referred to as the Spectral Norm. During the rest of the work we will also use the notation $||\mathbf{A}||$ to refer to the Spectral Norm.

Another important matrix norm that benefits from SVD is the Frobenius norm, defined in the following way.

Definition 7 (Frobenius norm)**.** *[40, Section 11.2 page 518] Let $\mathbf{A} \in \mathbb{R}^{n\times m}$ be a matrix. The Frobenius norm of $\mathbf{A}$ is defined as $||\mathbf{A}||_F = \sqrt{\sum_i^n \sum_j^m a_{ij}^2}$.*

It can be shown that also this norm is related to the singular values.

Proposition 1. *The Frobenius norm of a matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$ is equal to the square root of the sum of squares of its singular values.*

$$||\mathbf{A}||_F = \sqrt{\sum_i^r \sigma_i^2}$$

Proof.

$$||\mathbf{A}||_F = \sqrt{\sum_i^n \sum_j^n a_{ij}^2} = \sqrt{Tr[\mathbf{A}\mathbf{A}^T]} = \sqrt{Tr[(\mathbf{U}\Sigma\mathbf{V}^T)(\mathbf{U}\Sigma\mathbf{V})^T]} =$$

$$\sqrt{Tr[\mathbf{U}\Sigma\mathbf{V}^T\mathbf{V}\Sigma\mathbf{U}^T]} = \sqrt{Tr[\mathbf{U}\Sigma\Sigma\mathbf{U}^T]} = \sqrt{Tr[\mathbf{U}\Sigma^2\mathbf{U}^T]} = \sqrt{\sum_{i=1}^n \sigma^2}$$

From the cyclic property of the trace $Tr[\mathbf{AB}] = Tr[\mathbf{BA}]$ it follows that $Tr[\mathbf{U}\Sigma^2\mathbf{U}^T] = Tr[\mathbf{U}^T\mathbf{U}\Sigma^2] = Tr[\Sigma^2]$, which is the sum of the squares of the singular values $\sum_{i=1}^n \sigma^2$. □

Another interesting result about the SVD of a matrix is known as the Eckart–Young–Mirsky theorem.

Theorem 4 (Best F-Norm Low Rank Approximation)**.** *[12][32] Let $\mathbf{A} \in \mathbb{R}^{n\times m}$ be a matrix of rank r and singular value decomposition $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$. The matrix $\mathbf{A}^{(k)} = \mathbf{U}^{(k)}\Sigma^{(k)}\mathbf{V}^{(k)T}$ of rank $k \leq r$, obtained by zeroing the smallest $r-k$ singular values of $\mathbf{A}$, is the best rank-k approximation of $\mathbf{A}$.*
Equivalently, $\mathbf{A}_k = argmin_{\mathbf{B}:rank(\mathbf{B})=k}(||\mathbf{A}-\mathbf{B}||_F)$.
Furthermore, $min_{\mathbf{B}:rank(\mathbf{B})=k}(||\mathbf{A}-\mathbf{B}||_F) = \sqrt{\sum_{i=k+1}^r \sigma_i}$.

To get a clearer understanding of this result, we could rewrite $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ and notice that matrix $\mathbf{A}$ is the sum of r matrices $\mathbf{u}_i\mathbf{v}_i^T$ each scaled by a scalar σ_i.

In practice, SVD decomposes matrix $\mathbf{A}$ in matrices of rank one, ordered by importance according to the magnitude of the singular values: the smaller the σ_i, the smaller is the

contribution that the rank-1 matrix gives to the reconstruction of **A**. When the smallest singular values are set to 0, we still reconstruct a big part of the original matrix, and in practical cases, we will see that matrices can be approximated with a relatively small number of singular values.

This theorem is particularly relevant for data reduction and it is widely used, for instance, in image compression [40, Section 7.1 page 364]. Figure 2.1 shows the effect of a low-rank approximation on a high resolution picture of Mondello, the seaside of Palermo.

(a) 5184x3456 JPG image of Mondello (PA); red channel; standardized. Image size = 26,7 MB

(b) Same image, retaining only the first 80 singular vectors-values. Image size = 17,4 MB

Fig. 2.1 Image compression using SVD.

Unfortunately though, calculating the singular vectors and singular values of a matrix is a computationally intensive task. Indeed, for a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ the cost of the exact SVD is $O\left(min(n^2m, nm^2)\right)$. Recently, there have been developed approximate methods that compute the Eckart-Young-Mirsky approximations of matrices in time $O(knm)$, where k is the rank of the output matrix [34], or in times that scale super-linearly on the desired rank and one dimension of the input matrix [5]. In Section 3.2.4, we discuss more recent randomized approaches that further improve the run-times of SVD in the case of low-rank matrices.

2.3 Data representation for machine learning

In this section, we present three techniques for data representation that are used in machine learning: Principal Component Analysis (PCA), Correspondence Analysis (CA), and Latent Semantic Analysis (LSA). Their use range from data visualization to dimensionality reduction and noise reduction, finding applications in tasks such as anomaly detection, clustering, and natural language processing.

We explain how these techniques are closely related to the Singular Value Decomposition of the input matrix and how to compute the representation models having the SVD representation of the input data.

2.3.1 Principal Component Analysis

Principal Component Analysis is a widely-used multivariate statistical method that finds many applications in Machine Learning, ranging from outlier detection to dimensionality reduction and data visualization.

Consider a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ that stores some information about n objects in its rows, describing them through m variables in its column. An example could be a matrix that stores information about some people, taking into consideration their age, height, and weight.

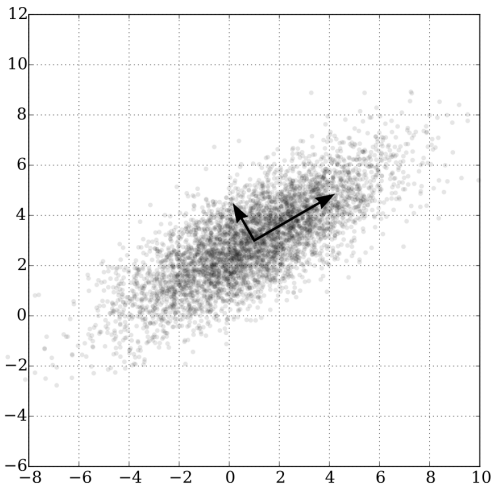


Fig. 2.2 The two principal components of a synthetic dataset sampled from a bivariate Gaussian distribution centered in (1,3) and with variance 3 and 1 along the two directions [10].

The main goal of PCA is to transform the original data matrix so to maximize the variance of each variable along the axis. During the process it is possible to reduce the number of variables taken into account, i.e. reducing $\mathbf{A} \in \mathbb{R}^{n \times m}$ to $\mathbf{A} \in \mathbb{R}^{n \times k}$ where $k \leq m$, and express the original data in terms of fewer latent variables that account for the majority of the variance of the original data. If it is possible to find a few latent variables that retain a large amount of variance of the original variables, it is possible to use PCA to visualize even high dimensional datasets.

The algorithm for PCA proceed as follows:
Start from the original data matrix

$$\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T \in \mathbb{R}^{n\times m};$$

compute the mean

$$\mu = \frac{1}{nm}\sum_{i}^{n}\sum_{j}^{m} a_{ij};$$

remove the mean from $\mathbf{A}$ before starting PCA

$$\mathbf{A} = \mathbf{A} - \mu\mathbf{1}_n\mathbf{1}_m^T;$$

compute the covariance matrix

$$\mathbf{X} = \frac{\mathbf{A}^T\mathbf{A}}{n-1};$$

compute the eigendecomposition of the covariance matrix

$$\mathbf{X} = \mathbf{Q}\Lambda\mathbf{Q}^{-1} = \frac{\mathbf{V}^T\Sigma^2\mathbf{V}}{n-1}.$$

Now, the eigenvectors of $\mathbf{X}$, and right singular values $\mathbf{V}$ of $\mathbf{A}$, are the principal directions or principal components. The eigenvalues of $\mathbf{X}$ represent the variance along the principal directions. To perform dimensionality reduction one computes the percentage of variance along each principal direction

$$\frac{\lambda_i}{\sum_j^n \lambda_j} = \frac{\sigma_i^2}{(n-1)}\frac{(n-1)}{\sum_j^n \sigma_j^2} = \frac{\sigma_i^2}{\sum_j^n \sigma_j^2}$$

and selects only the greatest k principal directions that account for a certain percentage of variance γ (usually is chosen in the range $[1/2, 1]$)

$$min(k) \text{ s.t. } \frac{\sum_i^k \lambda_i}{\sum_j^n \lambda_j} \geq \gamma$$

creating the matrix $\mathbf{V}^{(k)} \in \mathbb{R}^{m\times k}$.
Finally, the new data representation is computed by multiplying the initial data with the top-k principal directions

$$\mathbf{A}' = \mathbf{A}\mathbf{V}^{(k)} = (\mathbf{U}\Sigma\mathbf{V}^T)\mathbf{V}^{(k)} = \mathbf{U}^{(k)}\Sigma^{(k)}.$$

This last operation is a rotation whereby the vectors identifying the principal directions align with the axes.

In practice, to compute PCA and perform dimensionality reduction it is sufficient to compute the singular value decomposition of matrix $\mathbf{A}$ after the mean has been removed: one can compute the percentage of explained variance using the singular values and the new feature space by multiplying the top-k left singular vectors and singular values $\mathbf{U}^{(k)}\Sigma^{(k)}$.

2.3.2 Correspondence Analysis

Correspondence Analysis is a multivariate statistical tool used to explore relationships among categorical variables. CA belongs to the family of *factor analysis* methods, and it is similar to PCA, with the difference that it acts on categorical variables. The main idea of Correspondence Analysis is to decompose the chi-squared statistic of two categorical random variables into orthogonal factors [19].

Consider two categorical random variables $\mathscr{X}$, $\mathscr{Y}$ whose possible outcomes are respectively in $\{x_1,...,x_{|\mathscr{X}|}\}$, $\{y_1,...,y_{|\mathscr{Y}|}\}$ and consider some data D containing n observation of samples from X and Y jointly. One can imagine the possible outcomes of $\mathscr{X}$ and $\mathscr{Y}$ as the distinct values of two features of a dataset D.

For instance, if we consider a book of recipes, we can consider the outcomes of $\mathscr{X}$ as the distinct ingredients used in the book and those of $\mathscr{Y}$ as the distinct nationalities of the recipes. Correspondence analysis allows us to represent the ingredients and the nationalities in an Euclidean space that reveal important information about the correlation between the two variables. Starting from the recipes of the book it is possible to compute the contingency matrix $\mathbf{C} \in \mathbb{N}^{|\mathscr{X}|\times|\mathscr{Y}|}$, such that each cell c_{ij} contains the number of recipes in which the ingredient x_i and the nationality y_j appear together. The contingency table can be considered the input matrix of this algorithm.

From $\mathbf{C}$ it is possible to compute an estimate $\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}}$ of the joint probability of the two variables $\mathbf{P}_{\mathscr{X},\mathscr{Y}}$:

$$\hat{\mathbf{P}}_{X,Y} = \frac{\mathbf{C}}{\sum_{i=1}^{|\mathscr{X}|}\sum_{j=1}^{|\mathscr{Y}|} c_{ij}} = \frac{1}{n}\mathbf{C}.$$

The marginal probabilities are trivially computed as $\hat{\mathbf{p}}_{\mathscr{X}} = \hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}}\mathbf{1}_{|\mathscr{Y}|}$ and $\hat{\mathbf{p}}_{\mathscr{Y}} = \hat{\mathbf{P}}^T_{\mathscr{X},\mathscr{Y}}\mathbf{1}_{|\mathscr{X}|}$. We can define two matrixes $\mathbf{D}_{\mathscr{X}} := diag(\mathbf{p}_{\mathscr{X}})$ and $\mathbf{D}_{\mathscr{Y}} := diag(\mathbf{p}_{\mathscr{Y}})$, and compute:

$$\mathbf{A} := \mathbf{D}_{\mathscr{X}}^{-1/2}(\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}} - \hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}^T_{\mathscr{Y}})\mathbf{D}_{\mathscr{Y}}^{-1/2}.$$

Note that the term $(\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}} - \hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}_{\mathscr{Y}}^T)$ somehow expresses the deviation from independence of the variables $\mathscr{X}$ and $\mathscr{Y}$. Indeed, if the two variables were independent $\mathbf{P}_{\mathscr{X},\mathscr{Y}} = \mathbf{p}_{\mathscr{X}}\mathbf{p}_{\mathscr{Y}}^T$. Finally, let the singular value decomposition of $\mathbf{A}$ be $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$.

The orthogonal factors of $\mathscr{X}$ and $\mathscr{Y}$ respectively are $\mathbf{L}$ and $\mathbf{R}$, defined as follows:

$$\mathbf{L} := \mathbf{D}_{\mathscr{X}}^{-1/2}\mathbf{U}$$

$$\mathbf{R} := \mathbf{D}_{\mathscr{Y}}^{-1/2}\mathbf{V}.$$

The factor score λ_i associated to the i^{th} orthogonal factor is computed as:

$$\lambda_i = \sigma_i^2.$$

For each factor score λ_i we can compute the factor score ratio $\lambda_k^{(i)}$, which is a measure of how much variance ("correspondence") is captured by the corresponding factor:

$$\lambda_k^{(i)} = \frac{\lambda_i}{\sum_{j=1}^{r}\lambda_j}.$$

It is possible to represent the possible outcomes of the variables $\mathscr{X}$ and $\mathscr{Y}$ in an Euclidean space, called the *factoring space*, by exploiting the matrices $\mathbf{L}$ and $\mathbf{R}$. In particular, the outcome x_i is *represented* by the i^{th} row of $\mathbf{L}$ and the outcome y_i is *represented* by the i^{th} row of $\mathbf{R}$.

If the factor score ratios of the first two or three factors are significant, it is possible to graphically plot the output of CA to visualize the categorical values as points in $\mathbb{R}^2$ or $\mathbb{R}^3$.

The complexity of Correspondence Analysis depends on the size of the sample space of the random variables $\mathscr{X}$, $\mathscr{Y}$. As the number of possible outcomes of the analyzed variables increases, computing the SVD of $\mathbf{A}$ becomes prohibitive.

Hsiang et al. [19] have recently connected the problem of correspondence analysis to the problem of estimating the Principal Inertia Components (PIC) and have formulated the problem of solving Correspondence Analysis so that it can be solved with the aid of Neural Networks. The connection with PIC makes the problem of Correspondence analysis relevant even in tasks that concern privacy in machine learning [42].

2.3.3 Latent Semantic Analysis

Latent Semantic Analysis is a data representation method to represent words and documents as vectors in some Euclidean spaces, similarly to how Correspondence Analysis do, such that it is possible to define distance metrics among terms, among documents and among terms and documents. LSA's representation is similar to the one of Bag of Words or TF-IDF but it also aims to model latent semantics among words, and among documents as to better cope with problems of synonymy and polysemy.

Latent Semantic Analysis has firstly been introduced in a famous work by Deerwester et al. [11] as a method to perform automatic indexing and retrieval of text documents. When used for such purpose, it is usually referred to as Latent Semantic Indexing (LSI).

Consider a set D of documents containing words from a vocabulary V. Let the matrix $\mathbf{A} \in \mathbb{N}^{|V|\times|D|}$ be the *term-document matrix*, built in the same way of the contingency table of Correspondence Analysis: each component a_{ij} of the matrix is the number of times the i^{th} term of the vocabulary appears in the j^{th} document of the documents set.

The singular value decomposition of $\mathbf{A}$ is given by $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$.

Let $\mathbf{Q}$ be the a low-rank approximation of $\mathbf{A}$, obtained by keeping only the first k singular values of $\mathbf{C}$:

$$\mathbf{Q} = \mathbf{U}^{(k)}\Sigma^{(k)}\mathbf{V}^{(k)T}.$$

Given the above singular value decomposition, it is possible to use $\mathbf{U}$ and $\mathbf{V}$ to get representations of terms and documents that allow for comparisons.

Comparing two terms:

To evaluate whether two terms have similar patterns of occurrences across the set of documents, we can compute the dot product between the corresponding two rows of $\mathbf{Q}$. Observing that $\mathbf{QQ}^T = \mathbf{U}^{(k)}\Sigma^{(k)2}\mathbf{U}^{(k)T}$, we can obtain a representation space for the terms

$$\mathbf{L} = \mathbf{U}^{(k)}\Sigma^{(k)}$$

where the i^{th} row contains the coordinates of the i^{th} term.

Comparing two documents:

Analogously, to compare two documents with respect to the terms used, we can compute the dot product between the corresponding two columns of $\mathbf{Q}$. Observing that $\mathbf{Q}^T\mathbf{Q} = \mathbf{V}^{(k)}\Sigma^{(k)2}\mathbf{V}^{(k)T}$, we can obtain a representation space for the documents

$$\mathbf{R} = \mathbf{V}^{(k)}\Sigma^{(k)}$$

where the j^{th} row contains the coordinates of the j^{th} document.

Comparing a term and a document:

Finally, to compare the i^{th} term and the j^{th} document, one can check the value of q_{ij}, which can be obtained via the dot product of the i^{th} row of

$$\mathbf{L}' = \mathbf{U}^{(k)}\Sigma^{(k)1/2}$$

and the j^{th} row of

$$\mathbf{R}' = \mathbf{V}^{(k)}\Sigma^{(k)1/2}.$$

Using these semantic representations for documents and terms, it is possible to retrieve the most relevant documents according to a certain query (Latent Semantic Indexing).

A query is a sequence of words and is generally referred to as a *pseudo-document*.

Given a pseudo-document as a vector of terms $\mathbf{x}_q$ it is possible to derive its representation:

$$\mathbf{v}_q = \mathbf{x}_q^T\mathbf{U}^{(k)}\Sigma^{(k)-1}.$$

Appropriately scaling $\mathbf{v}_q$ with $\Sigma^{(k)}$ or $\Sigma^{(k)1/2}$ allows us to compare the query against other terms or documents.

To retrieve the l most relevant documents according to the query, one should compute the dot product with all the other documents and choose the closest l.

Chapter 3

Quantum background

3.1 Quantum computation

In this section, we introduce the fundamental notions of quantum computation. We begin by presenting the Dirac's formalism and then discuss the three fundamental postulates of Quantum Mechanics.

We present the formalism to represent the state of a single qubit and of systems of qubits. We discuss how such systems evolve and how they can be measured.

This section is not meant to be an exhaustive summary of the literature on quantum computation, but rather an introduction to the subject with a particular focus on the topics that are treated in this research.

Throughout the work, we disregard the physical implementation of qubits and treat them as mathematical entities that obey certain laws.

3.1.1 The Bra-ket notation

The formalism that we will adopt to describe quantum states is the usual Bra-ket notation, also known as Dirac's formalism.

Let $\mathbf{x} = [x_1, ..., x_n]^T \in \mathbb{C}^n$ be a vector in a complex space. The ket notation $|\mathbf{x}\rangle$ represents the complex column vector $\mathbf{x}$. The bra notation $\langle \mathbf{x}|$ represents the complex conjugate transpose vector $\mathbf{x}^\dagger$.

$$|\mathbf{x}\rangle = \begin{bmatrix} x_1 \\ \cdots \\ x_n \end{bmatrix}, \langle \mathbf{x}| = \begin{bmatrix} x_1^* & \cdots & x_n^* \end{bmatrix}.$$

Note that, in case the vector is real $\mathbf{x} \in \mathbb{R}^n$, its bra notation $\langle \mathbf{x}|$ is just the transpose of the vector, as the complex conjugate of a real number is the real number itself.

The bra-ket notation allows us to easily express operations on a complex vector space.

The *inner product* between two vectors $\mathbf{x}, \mathbf{y} \in \mathbb{C}^n$ can be written as

$$\langle \mathbf{x} | \mathbf{y} \rangle = \begin{bmatrix} x_1^* & \cdots & x_n^* \end{bmatrix} \begin{bmatrix} y_1 \\ \cdots \\ y_n \end{bmatrix} = \sum_i^n x_i^* y_i.$$

The inner product requires the two input vectors to lie on the same vector space and its result is a scalar.

Considering two vectors $\mathbf{x} \in \mathbb{C}^n, \mathbf{y} \in \mathbb{C}^m$ possibly lying in different vector spaces, their *outer product* can be written in the following way

$$|\mathbf{x}\rangle \langle \mathbf{y}| = \begin{bmatrix} x_1 \\ \cdots \\ x_n \end{bmatrix} \begin{bmatrix} y_1^* & \cdots & y_m^* \end{bmatrix} = \begin{bmatrix} x_1 y_1^* & \cdots & x_1 y_m^* \\ \cdots & \cdots & \cdots \\ x_n y_1^* & \cdots & x_n y_m^* \end{bmatrix}.$$

The result of an outer product between vectors in $\mathbb{C}^n, \mathbb{C}^m$ is a matrix in $\mathbb{C}^{n \times m}$.

Lastly, their *tensor product* is easily written as

$$|\mathbf{x}\rangle \otimes |\mathbf{y}\rangle = |\mathbf{x}\mathbf{y}\rangle = \begin{bmatrix} & \begin{bmatrix} y_1 \\ \cdots \\ y_m \end{bmatrix} \\ x_1 & \\ & \cdots \\ & \begin{bmatrix} y_1 \\ \cdots \\ y_m \end{bmatrix} \\ x_n & \end{bmatrix}.$$

The tensor product of two vectors in $\mathbb{C}^n$ and $\mathbb{C}^m$ is a vector in $\mathbb{C}^{nm}$.

3.1.2 Qubits and the Bloch sphere

Postulate 1 (State space)**.** *[33, Postulate 1 page 80] Associated to any isolated physical system is a complex vector space with inner product (that is, a Hilbert space) known as the state space of the system. The system is completely described by its state vector, which is a unit vector in the system's state space.*

As a bit is the fundamental unit of information in classical computing, a qubit is the fundamental information unit in quantum computing. A single qubit can be represented as a

unit ket vector $|\varphi\rangle$ in the inner product space $\mathbb{C}^2$. The main difference between a classical bit and a qubit is that a classical bit can only be in two states, either 0 or 1, while a qubit can be in a complex linear combination of two base states.

Conventionally, the two base states used to perform computations are states $|0\rangle$ and $|1\rangle$. They are also known as the vectors that form the *computational basis* or *z-basis*

$$|0\rangle = \begin{bmatrix} 1 \\ 0 \end{bmatrix}, |1\rangle = \begin{bmatrix} 0 \\ 1 \end{bmatrix}.$$

These states constitute an orthonormal basis for the $\mathbb{C}^2$ space. Indeed,

$$\langle 0|0\rangle = 1, \langle 1|1\rangle = 1$$

$$\langle 0|1\rangle = \langle 1|0\rangle = 0.$$

Since qubits are represented by unit ket vectors in the $\mathbb{C}^2$ space, the generic state of a qubit can be expressed as a *superposition* (i.e. complex linear combination) of the two states of the computational basis

$$|\varphi\rangle = \alpha\,|0\rangle + \beta\,|1\rangle = \alpha \begin{bmatrix} 1 \\ 0 \end{bmatrix} + \beta \begin{bmatrix} 0 \\ 1 \end{bmatrix} = \begin{bmatrix} \alpha \\ \beta \end{bmatrix}$$

where $\alpha, \beta \in \mathbb{C}$ are called amplitudes of the state and are complex numbers such that

$$|\alpha|^2 + |\beta|^2 = 1.$$

Because of this normalization constraint, it can be shown that each qubit can be rewritten in the following way

$$|\varphi\rangle = cos(\frac{\theta}{2})\,|0\rangle + e^{i\phi} sin(\frac{\theta}{2})\,|1\rangle$$

where $e^{i\phi}$ is the phase of the state. This notation allows us to visualize a qubit on a sphere, often referred to as the Bloch sphere, as shown in Figure 3.1.

All the states such that the sum of squares of their amplitudes sums to one are called *pure states* and lie on the surface of the Bloch sphere. There are cases in which a qubit is not an isolated system and its squared amplitudes do not sum to one. In this case, the qubit is said to be in a *mixed state*. Such states correspond to the points inside the Bloch sphere. Mixed states are usually described through the aid of density matrices rather then kets. In the rest of this work, we will only consider pure states, so we can stick to the ket vector representation.

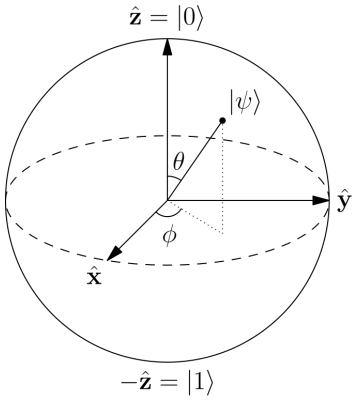


Fig. 3.1 The Bloch sphere visualization of a qubit state [9].

Of course, there are multiple orthogonal bases for the $\mathbb{C}^2$ Hilbert space. Other two famous states are the ones that form the so-called $x-basis$:

$$|+\rangle = \frac{|0\rangle + |1\rangle}{\sqrt{2}} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}, |-\rangle = \frac{|0\rangle - |1\rangle}{\sqrt{2}} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ -\frac{1}{\sqrt{2}} \end{bmatrix}.$$

We think it is useful to show the fact that these two vectors are orthonormal so to get the reader acquainted with the bra-ket arithmetic:

$$\langle +|+\rangle = \left(\frac{\langle 0| + \langle 1|}{\sqrt{2}}\right)\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) = \frac{\langle 0|0\rangle + \langle 0|1\rangle + \langle 1|0\rangle + \langle 1|1\rangle}{\sqrt{2}} = \frac{1+1}{2} = 1$$

$$\langle -|-\rangle = \left(\frac{\langle 0| - \langle 1|}{\sqrt{2}}\right)\left(\frac{|0\rangle - |1\rangle}{\sqrt{2}}\right) = \frac{\langle 0|0\rangle - \langle 0|1\rangle - \langle 1|0\rangle + \langle 1|1\rangle}{\sqrt{2}} = \frac{1+1}{2} = 1$$

$$\langle +|-\rangle = \langle -|+\rangle = \left(\frac{\langle 0| - \langle 1|}{\sqrt{2}}\right)\left(\frac{|0\rangle + |1\rangle}{\sqrt{2}}\right) = \frac{\langle 0|0\rangle + \langle 0|1\rangle - \langle 1|0\rangle - \langle 1|1\rangle}{\sqrt{2}} = \frac{1-1}{2} = 0.$$

Naturally, a quantum state expressed in the computational base can be written in other bases. For instance, in the $x-basis$:

$$|0\rangle = \frac{|+\rangle + |-\rangle}{\sqrt{2}}, |1\rangle = \frac{|+\rangle - |-\rangle}{\sqrt{2}}.$$

Therefore,

$$\begin{aligned} |\varphi\rangle = \alpha|0\rangle + \beta|1\rangle = \alpha\frac{|+\rangle + |-\rangle}{\sqrt{2}} + \beta\frac{|+\rangle - |-\rangle}{\sqrt{2}} = \\ \frac{\alpha+\beta}{\sqrt{2}}|+\rangle + \frac{\alpha-\beta}{\sqrt{2}}|-\rangle. \end{aligned} \tag{3.1}$$

3.1.3 Systems of multiple qubits

It is possible to extend the representation of a system of one qubit to the representation of a system of multiple qubits. If we consider n qubits, the Hilbert space we are considering grows to $\mathbb{C}^n$.

A system of 2 qubits, for example, can be described through ket vectors in the $\mathbb{C}^4$ vector space. The *computational basis* can then be rewritten as

$$|0\rangle = |00\rangle = |0\rangle \otimes |0\rangle = \begin{bmatrix} 1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ 0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}, |1\rangle = |01\rangle = |0\rangle \otimes |1\rangle = \begin{bmatrix} 1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ 0 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix},$$

$$|2\rangle = |10\rangle = |1\rangle \otimes |0\rangle = \begin{bmatrix} 0 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \\ 1 \begin{bmatrix} 1 \\ 0 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 1 \\ 0 \end{bmatrix}, |3\rangle = |11\rangle = |1\rangle \otimes |1\rangle = \begin{bmatrix} 1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \\ 1 \begin{bmatrix} 0 \\ 1 \end{bmatrix} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}.$$

These vectors constitute an orthonormal basis for $\mathbb{C}^4$. Every state of a two-qubit system can then be described as a linear combination of vectors in this basis

$$|\varphi\rangle = \alpha_0 |00\rangle + \alpha_1 |01\rangle + \alpha_2 |10\rangle + \alpha_3 |11\rangle .$$

It is possible to extend the concept of computational basis for a space $\mathbb{C}^n$. In the vector space $\mathbb{C}^n$, the computational basis is the set of vectors $\{|i\rangle = [1\delta_{1i}, \cdots, 1\delta_{(n-1)i}]^T, i \in [n]\}$ where δ_{ij} is the usual Dirac's delta as defined in the notations.

Therefore, a generic state of an n qubit system can be described in the following way

$$|\varphi\rangle = \sum_i^n \alpha_i |i\rangle . \tag{3.2}$$

The concept of pure states extends to multiple qubit systems. A qubit is in a pure state if it holds

$$\langle\varphi|\varphi\rangle = \sum_i^n |\alpha_i|^2 = 1.$$

This equation is also known as the *normalization condition* for state vectors.

The discussion about multiple orthogonal bases holds even for multiple qubit systems.

During the rest of the work, an $n-qubit$ system will also be referred to as a *quantum register*, as it is composed of multiple qubits just like a classical register is composed of several bits.

3.1.4 Unitary operations and quantum gates

Postulate 2 (Time evolution)**.** *[33, Postulate 2 page 81] The evolution of a* closed *quantum system is described by a* unitary transformation*. That is, the state $|\varphi\rangle$ of the system at time t_1 is related to the state $|\varphi'\rangle$ of the system at time t_2 by a unitary operator U which depends only on the times t_1 and t_2.*

$$|\varphi'\rangle = U|\varphi\rangle$$

A unitary operator is a matrix $U \in \mathbb{C}^{n\times n}$ such that $U^\dagger U = UU^\dagger = \mathbb{I}$. Unitary matrices are the complex equivalents of orthogonal matrices, they do not change the magnitude of vectors when applied to them. The application of a unitary operator on a quantum state preserves the normalization constraint, maintaining the state a unit vector, and corresponds to a rotation of the state on the Bloch sphere. Any unitary matrix corresponds to a valid quantum gate.

An example of a quantum gate is the quantum NOT gate. In the same way, a classical NOT gate brings a bit from the state 0 to the state 1 and vice-versa, the quantum NOT gate brings a qubit from a state of the computational basis to the other.

$$|0\rangle \mapsto |1\rangle, |1\rangle \mapsto |0\rangle.$$

When a qubit is in a superposition of the states of the computational basis, the quantum NOT gate acts linearly on the state.

$$\alpha|0\rangle + \beta|1\rangle \mapsto \beta|0\rangle + \alpha|1\rangle.$$

Visually, a quantum NOT gate is a rotation of 180° for the θ angle of the Bloch sphere. The unitary matrix representation of the 1-qubit NOT gate is the following matrix

$$X = |1\rangle\langle 0| + |0\rangle\langle 1|$$

$$X = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

In fact,

$$X\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}\begin{bmatrix}\alpha\\ \beta\end{bmatrix} = \begin{bmatrix}\beta\\ \alpha\end{bmatrix}.$$

Another quantum gate worth mentioning is the Hadamard gate

$$H = |+\rangle\langle 0| + |-\rangle\langle 1|$$

$$H = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}.$$

This gate brings the states of the computational basis to the states of the x-basis in the following way

$$H|0\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 1 \\ 0 \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \frac{|0\rangle + |1\rangle}{\sqrt{2}}$$

$$H|1\rangle = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix}\begin{bmatrix} 0 \\ 1 \end{bmatrix} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 \\ -1 \end{bmatrix} = \frac{|0\rangle - |1\rangle}{\sqrt{2}}.$$

Visually, one can imagine the Hadamard gate as a rotation of a qubit's state of 180° around the bisector of the x-z plane of the Bloch sphere. The Hadamard gate is particularly useful to create a uniform superposition of quantum states. Its extension to multiple-qubits gate allows preparing an n-qubit gate in a uniform superposition of all the states in the computational basis

$$H_n|0\rangle^{\otimes n} = \frac{1}{\sqrt{n}}\sum_{i}^{n}|i\rangle.$$

Similarly to how we have extended the single qubit systems to the many qubits systems using tensor products, it is possible to use tensor products of quantum gates to build gates that act independently on the qubits. For example, if we want a NOT gate that acts on the second qubit of a two qubits system we can build it as $\mathbb{I}\otimes X$

$$(\mathbb{I}\otimes X)(|\varphi\rangle|\gamma\rangle) = \mathbb{I}|\varphi\rangle\otimes X|\gamma\rangle.$$

In the same way, a 2-bit NOT that acts on the first of the two qubits is $X\otimes\mathbb{I}$ and n-qubit Hadamard gate is $H^{\otimes n}$.

It is easy to check that the tensor products of unitary matrices are still unitary matrices

$$(U\otimes\cdots\otimes U)(U\otimes\cdots\otimes U)^{\dagger} = (U\otimes\cdots\otimes U)(U^{\dagger}\otimes\cdots\otimes U^{\dagger}) =$$

$$UU^{\dagger}\otimes\cdots\otimes UU^{\dagger} = \mathbb{I}\otimes\cdots\otimes\mathbb{I} = \mathbb{I}.$$

There also exists quantum gates where one qubit of the system influences the others, like in the classical concepts of OR and AND. The most known multiple qubits gate is the c-NOT gate, which is a controlled-NOT gate. The basic 2-qubits c-NOT gate takes two qubits in

input, a *control* qubit and a *target* qubit. This gate is a conditional gate: the NOT operator is applied to the target qubit only if the control qubit is set to one. The output of a c-NOT are two qubits: the control one, that is left unchanged, and the target one that changes according to the control value. Mathematically, the 2-qubits c-NOT gate can be expressed as follows

$$\text{c-NOT} = |0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X$$

$$\text{c-NOT} = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix}.$$

Some examples of applications of the c-NOT are the following

$$\text{c-NOT}\,|0\rangle\,|\varphi\rangle = (|0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X)(|0\rangle\,|\varphi\rangle) = |0\rangle\langle 0|0\rangle \otimes \mathbb{I}\,|\varphi\rangle + |1\rangle\langle 1|0\rangle \otimes X\,|\varphi\rangle = |0\rangle\,|\varphi\rangle$$

$$\text{c-NOT}\,|1\rangle\,|\varphi\rangle = (|0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X)(|1\rangle\,|\varphi\rangle) =$$

$$|0\rangle\langle 0|1\rangle \otimes \mathbb{I}\,|\varphi\rangle + |1\rangle\langle 1|1\rangle \otimes X\,|\varphi\rangle = |1\rangle\,(X\,|\varphi\rangle)$$

$$\text{c-NOT}(\alpha\,|0\rangle + \beta\,|1\rangle)\,|\varphi\rangle = (|0\rangle\langle 0| \otimes \mathbb{I} + |1\rangle\langle 1| \otimes X)(\alpha\,|0\rangle\,|\varphi\rangle + \beta\,|1\rangle\,|\varphi\rangle) =$$

$$\alpha\,|0\rangle\langle 0|0\rangle \otimes \mathbb{I}\,|\varphi\rangle + \beta\,|0\rangle\langle 0|1\rangle \otimes \mathbb{I}\,|\varphi\rangle + \alpha\,|1\rangle\langle 1|0\rangle \otimes X\,|\varphi\rangle + \beta\,|1\rangle\langle 1|1\rangle \otimes X\,|\varphi\rangle =$$

$$\alpha\,|0\rangle\,|\varphi\rangle\,\beta\,|1\rangle\,(X\,|\varphi\rangle).$$

Of course, it is possible to build control gates different than the c-NOT. During the rest of the work, we will be referring to controlled-gates as conditional rotations.

All the quantum gates, because of the unitary constraint, must be reversible. That is the reason why to simulate classical gates we usually we have to pay an overhead by introducing ancillary qubits. For instance, the Toffoli gate, which is the quantum equivalent of the AND gate, takes three qubits in input and outputs three qubits. Thanks to the Toffoli gate it is possible to simulate all the other classical gates [33, Section 1.4.1 page 29].

In the end, all the quantum computing algorithms can be described by a series of unitary operations and some measurements.

3.1.5 Measurements of quantum systems

Postulate 3 (Quantum measurement)**.** *[33, Postulate 3 page 84] Quantum measurements are described by a collection* $\{M_m\}$ *of* measurements operators*. These are operators acting on the state space of the system being measured. The index m refers to the measurement outcomes that may occur in the experiment. If the state of the quantum system is* $|\varphi\rangle$ *immediately before the measurement then the probability that result m occurs is given by*

$$p(m) = \langle \varphi | M_m^\dagger M_m | \varphi \rangle \,, \tag{3.3}$$

and the state of the system after the measurement is

$$\frac{M_m |\varphi\rangle}{\sqrt{\langle \varphi | M_m^\dagger M_m | \varphi \rangle}}. \tag{3.4}$$

The measurement operators satisfy the completeness equation*,*

$$\sum_m M_m^\dagger M_m = \mathbb{I}. \tag{3.5}$$

The completeness equation expresses the fact that the probabilities sum to one:

$$1 = \sum_m p(m) = \sum_m \langle \varphi | M_m^\dagger M_m | \varphi \rangle \,. \tag{3.6}$$

We have introduced the concept of quantum states and described how they evolve over time. The last step to understand a quantum algorithm is to understand how a quantum state can be measured and what information can be retrieved.

Measuring a classical bit is conceptually rather easy, one can directly measure if the bit is in the 0 or 1 state. On the other hand, a qubit can be in a general superposition of two orthogonal states of the $\mathbb{C}^2$ vector space and it is not possible to instantly retrieve the information about the amplitudes of the state in a particular basis.

There exists a different type of measurements than the one discussed in this section. This type of measurements is generally referred to as POVM measurements and they still obey the third postulate. We omit the explanation of POVMs as it is possible to understand the rest of the work without this notion. An accurate description of POVMs can be found in [33, Section 2.2.6].

As already stated, we are not interested in the physical procedures used to measure a qubit, rather we are interested in the mathematical modeling of the measurement. To start with the kind of measurements that we will use later in the work, one has to choose an orthogonal basis for the Hilbert space where the quantum system lies. Once the basis is chosen, it is possible to measure the system of qubits in this base. The output of the measurement will be one quantum state among the states of the chosen basis.

In particular, once a generic quantum state is measured it loses its superposition properties and collapses to a vector of the chosen basis state. Each basis vector is likely to be measured with probability equal to the squared amplitude of the generic quantum state with respect to the particular basis vector.

To better clarify the statement above, let's discuss measurements on a single qubit considering outputs in the computational basis. In this particular case, the set of mathematical operators that are used to describe the measurements is the following:

$$\{M_0 = |0\rangle\langle 0|, M_1 = |1\rangle\langle 1|\}.$$

It is interesting to note the following equivalences, that can be shown to be valid for whichever orthogonal basis we choose

$$M_0^\dagger M_0 = |0\rangle\langle 0|0\rangle\langle 0| = |0\rangle\langle 0| = M_0$$

$$M_1^\dagger M_1 = |1\rangle\langle 1|1\rangle\langle 1| = |1\rangle\langle 1| = M_1$$

$$\sum_m M_m^\dagger M_m = M_0^\dagger M_0 + M_1^\dagger M_1 = M_0 + M_1 = |0\rangle\langle 0| + |1\rangle\langle 1| = \mathbb{I}.$$

If we consider the generic state

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle.$$

The probabilities of measuring the state $|\varphi\rangle$ in the computational basis and seeing it collapse into the states $|0\rangle$ or $|1\rangle$ are equal the square amplitudes of $|\varphi\rangle$ with respect to the basis vector

$$p(0) = \langle\varphi|M_0^\dagger M_0|\varphi\rangle = \langle\varphi|M_0|\varphi\rangle = (\alpha^*\langle 0| + \beta^*\langle 1|)|0\rangle\langle 0|(\alpha|0\rangle + \beta|1\rangle) =$$

$$\alpha^*\alpha\langle 0|0\rangle\langle 0|0\rangle + \alpha^*\beta\langle 0|0\rangle\langle 0|1\rangle + \beta^*\alpha\langle 1|0\rangle\langle 0|0\rangle + \beta^*\beta\langle 1|0\rangle\langle 0|1\rangle = |\alpha|^2$$

$$p(1) = \langle\varphi|M_1^\dagger M_1|\varphi\rangle = \langle\varphi|M_1|\varphi\rangle = (\alpha^*\langle 0| + \beta^*\langle 1|)|1\rangle\langle 1|(\alpha|0\rangle + \beta|1\rangle) =$$

$$\alpha^*\alpha\langle 0|1\rangle\langle 1|0\rangle + \alpha^*\beta\langle 0|1\rangle\langle 1|1\rangle + \beta^*\alpha\langle 1|1\rangle\langle 1|0\rangle + \beta^*\beta\langle 1|1\rangle\langle 1|1\rangle = |\beta|^2.$$

Because of the normalization constraint of a pure state, these probabilities sum to one as stated in Equation 3.6.

In general, if $|\varphi\rangle$ is a generic quantum state and $|\gamma\rangle$ is a state of the orthogonal basis in which the measurement is being performed, the probability of measuring $|\gamma\rangle$ is:

$$p(\gamma) = \langle\varphi|\gamma\rangle \langle\gamma|\gamma\rangle \langle\gamma|\varphi\rangle = \langle\varphi|\gamma\rangle \langle\gamma|\varphi\rangle = |\langle\gamma|\varphi\rangle|^2.$$

For instance, a measurement of the quantum state $|\varphi\rangle = \frac{1}{\sqrt{n}} \sum_i^n |i\rangle$ in the computational basis can output each state $|j\rangle, j \in [n]$ with uniform probability

$$p(j) = |\langle j|\varphi\rangle|^2 = |\langle j| \frac{1}{\sqrt{n}} \sum_i^n |i\rangle|^2 = |\frac{1}{\sqrt{n}} \langle j|j\rangle + \frac{1}{\sqrt{n}} \sum_i^n \langle j|i\rangle|^2 = |\frac{1}{\sqrt{n}}|^2 = \frac{1}{n}$$

and a measurement of the state $|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle$ in the x-basis outputs the $|+\rangle$ state with probability:

$$p(+) = |\langle +|\varphi\rangle|^2 = |\langle +| (\alpha|0\rangle + \beta|1\rangle)|^2 = |\langle +| (\frac{\alpha+\beta}{\sqrt{2}}|+\rangle + \frac{\alpha-\beta}{\sqrt{2}}|-\rangle)|^2 =$$

$$|\frac{\alpha+\beta}{\sqrt{2}} \langle +|+\rangle + \frac{\alpha-\beta}{\sqrt{2}} \langle +|-\rangle|^2 = |\frac{\alpha+\beta}{\sqrt{2}}|^2$$

taking advantage of Equation 3.1.

Finally, it is important to know that a fundamental theorem of quantum computation, known as the *no cloning* theorem, states that it is impossible to create a circuit that creates a copy of a generic quantum state.

Theorem 5 (No cloning)**.** *[33, Box 12.1 page 532] There is no quantum algorithm that can create a copy of a general quantum state.*

Since after being measured a quantum state collapses to one of the states of the measurement basis and it is not possible to copy the state before the measurement, to measure a state multiple times and get more information on its amplitudes it is necessary to run the algorithm that creates it multiple times.

Through measurements it is possible to reconstruct a quantum state, retrieving the information about the amplitudes, including the phase components. The problem of fully characterizing a quantum state through measurements is called *quantum tomography*.

Wald confidence interval

In this subsection, we focus on the problem of estimating the probability that a quantum state collapses on a particular basis state when measured. In particular, following the explanation of [39, Section 5.1.3], we discuss how the Wald confidence interval provides us with a number of measurements that allow us to estimate this probability within a certain error and with a certain confidence.

Consider one qubit measured in the computational basis

$$|\varphi\rangle = \alpha|0\rangle + \beta|1\rangle.$$

As discussed in the previous section, the probability of measuring $|0\rangle$ is $p(0) = |\alpha|^2$ and the probability of measuring $|1\rangle$ is $p(1) = |\beta|^2 = 1 - p(0)$. The problem that we are addressing in here is: How many times do we need to produce and measure the state $|\varphi\rangle$ to get an estimate $\hat{p}$ of $p(0)$ such that $||\hat{p} - p(0)|| \leq \varepsilon$?

We can model the qubit as a random variable that follows a Bernoulli distribution and the measurement process as a Bernoulli trial: each measurement of $|\varphi\rangle$ is equivalent to sampling from the distribution and it results in either $|0\rangle$ or $|1\rangle$; we can consider it a success if the state collapses on $|0\rangle$ and a failure otherwise.

Let's denote with $Q = \{q_1, \cdots, q_S\}$ the set of outcomes of the measurements, where each $q \in \{0,1\}$ is equal to 1 if the measurement is a success and 0 otherwise. Furthermore, let z be the z-value for the confidence level. We use the Greek letter ζ to denote the number of successful measurements $\zeta = \sum_i^S q_i$.

Define the estimator $\hat{p}$ for p as the frequentist estimator

$$\hat{p} = \frac{\zeta}{S}.$$

The Wald confidence interval guarantees that $||\hat{p} - p|| \leq \varepsilon$ with probability calculated according to the z confidence level. For instance, $z = 2.58$ guarantees that $||\hat{p} - p|| \leq \varepsilon$ with a confidence of 99%, meaning that if one repeat the estimation multiple times, 99% of the times the estimator is ε-close to the real probability.

The error of the estimator can be shown to be

$$\varepsilon = z\sqrt{\frac{\hat{p}(1-\hat{p})}{S}}$$

and it is maximized when $\hat{p} = 0.5$. So, in general, it holds that

$$\varepsilon \leq \frac{z}{2\sqrt{S}}.$$

Solving the above inequality for S we find the number of required measurements, in a function of ε and z:

$$S = \frac{z^2}{4\varepsilon^2}.$$

It is interesting to note that the z values are pretty small numbers and that overall the number of measurements is in $O(\frac{1}{\varepsilon^2})$.

When considering a quantum register made of several qubits

$$|\gamma\rangle = \sum_i^n \gamma_i |i\rangle$$

we might be interested in estimating the probabilities $\mathbf{p} = \begin{bmatrix} p_0 \\ \cdots \\ p_{n-1} \end{bmatrix}$ of measuring the $\{|i\rangle, i \in [n]\}$ states. It is still possible to leverage the theory of the Wald score interval if after the measurement process we consider n distinct Bernoulli trials where the i^{th} Bernoulli trial succeeds if the i^{th} is measured. In this way, with S total measurements it is possible to estimate each probability p_i such that $||\hat{p}_i - p_i|| \leq \varepsilon$ with a certain confidence.

Note that this is different than estimating $\hat{\mathbf{p}} = \begin{bmatrix} \hat{p}_0 \\ \cdots \\ \hat{p}_{n-1} \end{bmatrix}$ such that $||\hat{\mathbf{p}} - \mathbf{p}|| \geq \varepsilon$ with the chosen confidence. To have this kind of guarantees it is necessary to model a qubit as a random variable that follows a Multinomial distribution and use simultaneous confidence intervals.

As a final remark, the Wald confidence interval is not the only Binomial confidence interval. For instance, a different confidence interval is the Wilson score interval [45], which employs a different estimator $\hat{p}$ and promises more reliable estimates when the probabilities approach 0 or 1. Later in the work, we will see that we are not interested in strict guarantees on probabilities around 0 or 1. Moreover, experiments in Section 5.3.1 show that the frequentist estimator with $O(\frac{1}{\varepsilon^2})$ measurements works good in practice for our purposes.

3.2 Quantum machine learning

After introducing the notation and some notions of quantum computation, in this section, we describe some more recent results on how to efficiently store large amounts of data in quantum computers, process them, and retrieve them. Finally, in 3.2.4, we discuss the quantum research works that are more related to ours.

3.2.1 Quantum data representation

Throughout the work, we will be representing scalars, vectors, and matrices as pure quantum states.

Scalar values $\gamma \in \mathbb{R}$ are represented using their binary expansion in the same way a scalar is stored in a register of a classical computer. Let the n-bit representation of γ be $\gamma = [\gamma_0, \gamma_1, \cdots, \gamma_{n-1}]$ where $\gamma_i \in \{0,1\}$, then the quantum register that represents it is the following state of the $\mathbb{C}^n$ computational basis:

$$|\gamma\rangle = |\gamma_0, \gamma_1, \cdots, \gamma_{n-1}\rangle .$$

In terms of memory complexity, to store a scalar γ that can be represented in n bits we need a quantum register of size $\Theta(n)$.

Real vectors $\mathbf{x} = \begin{bmatrix} x_0 \\ \cdots \\ x_{m-1} \end{bmatrix} \in \mathbb{R}^m$ can be stored as a *state-vectors*

$$|\mathbf{x}\rangle = \frac{1}{||\mathbf{x}||} \sum_{i=0}^{m-1} x_i |i\rangle = \begin{bmatrix} \frac{x_0}{||\mathbf{x}||_2} \\ \cdots \\ \frac{x_{m-1}}{||\mathbf{x}||_2} \end{bmatrix}$$

where the i^{th} element of the vector is stored in the amplitude of the $|i\rangle$ state. The memory complexity required to store a $\mathbb{R}^m$ vector is $O(log(m))$, in fact it is possible to use a quantum register $|i_0 i_1 ... i_{\lceil \log m \rceil - 1}\rangle$ of $\lceil \log_2 m \rceil$ qubits and a classical register that stores $||\mathbf{x}||$. If m is not a power of 2, the amplitudes of the states $\{|i\rangle : i \geq m\}$ are just set to zero.

Finally, also real matrices $\mathbf{A} = \begin{bmatrix} a_{00} & \cdots & a_{0(m-1)} \\ \cdots & \cdots & \cdots \\ a_{(n-1)0} & \cdots & a_{(n-1)(m-1)} \end{bmatrix} \in \mathbb{R}^m$ can be stored in *state-vectors*

$$|\mathbf{A}\rangle = \frac{1}{||\mathbf{A}||_F} \sum_i^n \sum_j^m a_{ij} |i\rangle |j\rangle = \begin{bmatrix} \frac{a_{00}}{||\mathbf{A}||_F} \\ \cdots \\ \frac{a_{0(m-1)}}{||\mathbf{A}||_F} \\ \cdots \\ \frac{a_{(n-1)(0)}}{||\mathbf{A}||_F} \\ \cdots \\ \frac{a_{(n-1)(m-1)}}{||\mathbf{A}||_F} \end{bmatrix}.$$

In this case the memory complexity $O(log(nm))$. Indeed, it is possible to use a bigger quantum register $|i_0 i_1 ... i_{\lceil \log n \rceil - 1}\rangle |j_0 j_1 ... j_{\lceil \log m \rceil - 1}\rangle$ of size $\lceil \log_2 n \rceil + \lceil \log_2 m \rceil$ qubits and a classical register to store $||\mathbf{A}||_F$. Just like in the vector case, if nm is not a power of 2 then the remaining entries are set to zero.

As discussed in Section 3.1.4, to prepare a quantum state $|\varphi\rangle$ we need access to a unitary U_φ that prepares the state

$$|\varphi\rangle = U_\varphi |0\rangle .$$

Given a generic state $|\varphi\rangle$, we denote the time to prepare it as $T(U_\varphi)$.

A Quantum Random Access Memory is a data structure that allows efficient quantum access to classical data. We define the concept of having efficient quantum access in the following way.

Definition 8 (Efficient quantum access)**.** *Given a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$, we say that we have efficient quantum access to it if there exists a data structure that allows performing the following mappings*

- $U : |i\rangle |0\rangle \mapsto |i\rangle |\mathbf{a}_{i,\cdot}\rangle = |i\rangle \frac{1}{||\mathbf{a}_{i,\cdot}||} \sum_j^m a_{ij} |j\rangle$
- $V : |0\rangle |j\rangle \mapsto |\overline{\mathbf{A}}\rangle |j\rangle = (\frac{1}{||\overline{\mathbf{A}}||} \sum_i^n ||\mathbf{a}_{i,\cdot}|| \, |i\rangle) |j\rangle$, *where* $\overline{\mathbf{A}} \in \mathbb{R}^n$ *and* $\overline{a}_i = ||\mathbf{a}_{i,\cdot}||$

in time polylog(nm).

A possible data structure for the QRAM has been described in [23] [25] [22] and it is reported in the following subsection. We also include some basic notions about Block encodings, an advanced data encoding technique that promises substantial speed-ups in linear algebra tasks.

QRAM access model

The concept of QRAM has been widely discussed in quantum computing literature. In this work we refer to QRAM as a data structure that allows to efficiently creating quantum states from a classical data structure. In *Kerenidis et al.* [23] [25] they describe an efficient data structure that stores a dataset occupying linear space in the number of data points and that requires logarithmic time to perform an update/insertion/deletion of an already stored data point. Moreover, such a structure allows for the creation a quantum vector state in time polylogarithmic on the number of vector components.

Theorem 6. *[QRAM Data Structure] [23][25] Let* $\mathbf{A} \in \mathbb{R}^{n \times m}$ *be a matrix. Entries* (i, j, a_{ij}) *arrive in the system in arbitrary order and w is the number of entries already stored in the system. There exists a data structure to store the matrix* $\mathbf{A}$ *with the following properties:*

- *The size of the data structure is* $O(wlog^2(nm))$.
- *The time to store a new entry* (i, j, a_{ij}) *is* $O(log^2(nm))$
- *A quantum algorithm that has quantum access to the data structure can perform the mapping* $\tilde{U} : |i\rangle|0\rangle \mapsto |i\rangle|\mathbf{a}_{i,\cdot}\rangle = |i\rangle \frac{1}{||\mathbf{a}_{i,\cdot}||} \sum_j^m a_{ij}|j\rangle$, *for* $i \in [n]$, *corresponding to the rows of the matrix currently stored in memory and the mapping* $\tilde{V} : |0\rangle|j\rangle \mapsto |\tilde{\mathbf{A}}\rangle|j\rangle = \left(\frac{1}{||\tilde{\mathbf{A}}||} \sum_i^n ||\mathbf{a}_{\mathbf{i},\cdot}|| \, |i\rangle\right)|j\rangle$, *for* $j \in [m]$, *where* $\tilde{\mathbf{A}} \in \mathbb{R}^n$ *stores the norm of each row, in time* $O(polylog(nm))$.

Later in this work, we will assume that matrices have been stored in QRAM and that their greatest singular value will be smaller then one. In [25], *Kerenidis et al.* provide an efficient algorithm to normalize a matrix such that $||\mathbf{A}||_2 = \sigma_{max} \leq 1$. Such algorithm consists in performing the quantum *Spectral norm estimation*, reported in Section 3.2.3, to estimate $||\mathbf{A}||_2$ and divide all the components of the matrix by the spectral norm.

$$\mathbf{A}' = \frac{\mathbf{A}}{||\mathbf{A}||_2} = \mathbf{U}\frac{\Sigma}{\sigma_{max}}\mathbf{V}^T.$$

Since all the entries in Σ are $0 \leq \sigma_i \leq \sigma_{max}$, we have that all the singular values of $\mathbf{A}'$ are between 0 and 1 and the left and right singular vectors are the same of $\mathbf{A}$.

Block encodings

A recent research work from *Chakraborty et al.* [3] has formalized the framework of block encodings for quantum matrix representation, starting from the results of Low and Chuang

[29] in hamiltonian simulation. A block encoding of a matrix is a unitary embedding of the matrix.

Definition 9 (Block encoding)**.** *[26][3][16] Let* $\mathbf{A} \in R^{n\times n}$ *be a matrix. The l-qubit unitary matrix* $\mathbf{U} \in \mathbb{C}^{2^l\times 2^l}$ *is a* (α, l) *block encoding of* $\mathbf{A}$ *if* $\mathbf{U} = \begin{bmatrix} \mathbf{A}/\alpha & \cdot \\ \cdot & \cdot \end{bmatrix}$

As discussed by Chakraborty, Kerendis and Prakash have implicitly proven that it is possible to create a block encoding of a square matrix, stored in QRAM in time which is polylogarithmic in the number of rows of the matrix.

Theorem 7 (Block encoding using QRAM data structures)**.** *[26][3][23] [25] There exist QRAM data structures for storing vectors* $\mathbf{v}_i \in \mathbb{R}^n$*,* $i \in [m]$ *and matrices* $\mathbf{A} \in \mathbb{R}^{n\times n}$ *such that with access to these data structures one can implement a* $(\alpha(\mathbf{A}), 2\log n)$ *unitary block encoding for* $\mathbf{A}$ *with* $\alpha(\mathbf{A}) = min(||\mathbf{A}||_F/||\mathbf{A}||_2, s_1(\mathbf{A})/||\mathbf{A}||_2)$*, where* $s_1(\mathbf{A}) = max_i \sum_j |a_{ij}|$ *in time* $\tilde{O}(\log n)$*. Moreover, this block encoding can be constructed in a single pass over the matrix* $\mathbf{A}$*, and it can be updated in* $O(\log^2 n)$ *time per entry.*

Block encodings have a huge potential in linear algebra tasks and quantum machine learning, as this unitary representation of matrices allows performing fast matrix arithmetic calculations and singular values transformations [16]. In this work we will mainly use the QRAM access method to encode matrices in quantum states. However, we will discuss how block encodings could be used to improve the algorithms introduced in this research.

3.2.2 Quantum singular value estimation

Many quantum computing papers have been focusing on the problem of the quantum singular value decomposition of a matrix with different approaches [28] [36]. [23] [16] The problem of singular value estimation (SVE) , as formalized in [23], consists in mapping a vector $|\mathbf{x}\rangle$, that can be expressed as a linear combination of right (or left) singular vector of a matrix $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ as $|\mathbf{x}\rangle = \sum_i \alpha_i |\mathbf{v}_i\rangle$, to a vector $|\mathbf{x}\rangle = \sum_i \alpha_i |\mathbf{v}_i\rangle |\sigma\rangle$ such that the last register stores the singular values of $\mathbf{A}$.

Theorem 8 (Singular Value Estimation)**.** *[25][23] Let* $\mathbf{A} \in \mathbb{R}^{n\times m}$ *be a matrix with singular value decomposition* $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ *and* $r = min(n, m)$ *stored in QRAM. Let* ε *be a precision parameter. It is possible to create an algorithm* U_{SVE} *that performs the mapping*

$$|b\rangle = \sum_i \alpha_i |\mathbf{v}_i\rangle \mapsto \sum_i \alpha_i |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle$$

where $|\overline{\sigma}_i - \sigma_i| \leq \varepsilon$ *for all i with probability* $(1 - 1/poly(m))$ *in running time* $O(\frac{\mu(\mathbf{A})}{\varepsilon} polylog(nm))$, *where*

- $\mu(\mathbf{A}) = min_{p\in[0,1]}(||A||_F, \sqrt{s_{2p}(\mathbf{A})s_{2(p-1)}(\mathbf{A}^T)})$,
- $s_p(\mathbf{A}) = max_i||\mathbf{a}_{i,\cdot}||_p^p = max_i(\sum_j^m \mathbf{a}_{i,j}^p)^{1/p}$.

This theorem, firstly reported in [23] and later refined in [25], leverages on a famous quantum algorithm known as the *phase estimation* algorithm [33, Section 5.2 page 221].

Because of this algorithm, the error on the estimates $|\overline{\sigma}\rangle$ is not consistent and, potentially, can change at every run of the algorithm. In order to have the same estimations over different execution of the algorithms, it is possible to exploit a new version of the phase estimation algorithm that makes the error consistent.

Theorem 9 (Consistent Phase Estimation)**.** *[25][41] Let U be a unitary operator with eigenvectors* $|\mathbf{v}_i\rangle$ *and eigenvalues* $e^{i\theta_j}$ *for* $\theta_j \in [-\pi, \pi]$. *There exists a quantum algorithm with running time* $O(T(U)\frac{logn}{\varepsilon})$ *that transforms*

$$|\Phi\rangle = \sum_j^k \alpha_j |\mathbf{v}_j\rangle \mapsto \sum_j^k \alpha_j |\mathbf{v}_j\rangle |\overline{\theta}_j\rangle$$

where $\overline{\theta}_j$ *is an estimate such that* $|\overline{\theta}_j - \theta_j| \leq \varepsilon$ *for all* $j \in [k]$ *with probability at least* $1 - 1/poly(k)$. *The values of* $\overline{\theta}_j$ *are consistent across multiple executions of the algorithm.*

Using this modified version of the phase estimation algorithm, that has a consistent error across different runs, it is possible to derive a consistent singular value estimation algorithm. The consistent SVE algorithm still has the same complexity of the algorithm in Theorem 8 and has been sketched in [20] [1]. Throughout the rest of the work, when referring to singular value estimation we will be meaning its consistent version.

Consider that when applying SVE to a generic vector $|\mathbf{x}\rangle = \sum_i^r \alpha_i |\mathbf{v}_i\rangle \rightarrow \sum_i^r \alpha_i |\mathbf{v}_i\rangle |\sigma_i\rangle$ and measuring the last register, each $|\sigma_i\rangle$ has a probability of being measured $p(\sigma_i) = |\alpha_i|^2$. In practice, it can happen that the vector $|\mathbf{x}\rangle$ is almost perpendicular to one of the singular vectors corresponding to the highest singular values and that $|\alpha_i|^2$ is a very small probability. An interesting observation that we will exploit in the next chapter, initially introduced in [28] under the name of self-tomography, is that one can apply the SVE to the matrix state itself. Recall, from Proposition 1 and Theorem 3, that $||\mathbf{A}||_F = \sqrt{\sum_i^n \sum_j^m a_{ij}^2} = \sqrt{\sum_i^r \sigma_i^2}$ where $r = rank(\mathbf{A})$ and that a matrix can be decomposed in terms of its singular values and singular vectors as $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$.

Therefore, the following quantum states are equivalent:

$$|\mathbf{A}\rangle = \frac{1}{||\mathbf{A}||_F}\sum_i^n\sum_j^m a_{ij}\,|i\rangle\,|j\rangle = \frac{1}{\sqrt{\sum_i^r \sigma_i^2}}\sum_i^r \sigma_r\,|\mathbf{u}_i\rangle\,|\mathbf{v}_i\rangle\,. \tag{3.7}$$

Applying SVE to the matrix $|\mathbf{A}\rangle$ itself produces the following state

$$|\mathbf{A}\rangle = \frac{1}{\sqrt{\sum_i^r \sigma_i^2}}\sum_i^r \sigma_r\,|\mathbf{u}_i\rangle\,|\mathbf{v}_i\rangle \rightarrow \frac{1}{\sqrt{\sum_i^r \sigma_i^2}}\sum_i^r \sigma_r\,|\mathbf{u}_i\rangle\,|\mathbf{v}_i\rangle\,|\sigma_i\rangle$$

allowing each singular value to be measured with a probability $p(\sigma_i) = \frac{\sigma_i^2}{\sqrt{\sum_i^r \sigma_i^2}}$.

The SVE algorithm has been used in several quantum machine learning tasks over the last couple of years because of its polylogarithmic dependency on the number of matrix elements [23] [25] [20] [22]. Some major applications of SVE have been matrix algebra tasks such as matrix multiplication, inversion, and subspace projection in time polylogarithmic on the number of elements of the matrices. However, because of the phase estimation, such algorithms have a linear dependency on the error parameter.

Recently, matrix algebra techniques have been further improved by the works of Chakraborty and Gilyen [3][16] by means of the qubitization technique and the block encoding framework. The matrix algebra operations described in these papers do not rely on explicitly computing the singular value estimation in a register, but, instead, perform operations on the singular values of a matrix directly. This further reduces the linear dependency on the error parameter to a logarithmic dependency.

Theorem 10 (Matrix Algebra [22][3][16])**.** *Let* $\mathbf{A} = \sum_i^k \sigma \mathbf{u}_i \mathbf{v}_i \in \mathbb{R}^{n\times n}$ *such that* $||\mathbf{A}||_2 = 1$*, and a vector* $\mathbf{x} \in \mathbb{R}^n$ *stored in QRAM. There exist quantum algorithms that with probability at least* $1 - 1/poly(n)$ *return:*

- *a state* $|\mathbf{z}\rangle$ *such that* $||\mathbf{z}\rangle - |\mathbf{A}\mathbf{x}\rangle| \leq \varepsilon$ *in time* $\tilde{O}(\kappa(\mathbf{A})\mu(\mathbf{A})log(1/\varepsilon))$*;*
- *a state* $|\mathbf{z}\rangle$ *such that* $||\mathbf{z}\rangle - |\mathbf{A}^{-1}\mathbf{x}\rangle| \leq \varepsilon$ *in time* $\tilde{O}(\kappa(\mathbf{A})\mu(\mathbf{A})log(1/\varepsilon))$.

It is possible to also get estimates of the norms with multiplicative error η *by increasing the running time by a factor* $1/\eta$.

Where $\kappa(\mathbf{A})$ is the condition number of the matrix, defined as $\kappa(\mathbf{A}) = \frac{\sigma_{max}}{\sigma_{min}}$.

Even though some matrix algebra tasks can be executed more efficiently without computing the SVE explicitly, if one is interested in classically accessing the singular values of a matrix the SVE seems a necessary step.

3.2.3 Retrieving data from quantum computers

Now that we know how to encode data in quantum states and that we have introduced the quantum singular value estimation algorithm, the final step is to provide the reader with the theorems that will be used to classically retrieve the output of quantum algorithms.

The first theorem that we introduce is the amplitude amplification and estimation theorem. This theorem, provided an algorithm whose output is a state $\alpha |\mathbf{x},0\rangle + \beta |\mathbf{G},0^{\perp}\rangle$ where $|\mathbf{x}\rangle$ is the state in which we are interested and $|\mathbf{G}\rangle$ is a garbage state, allows to prepare the good state $|\mathbf{x}\rangle$ or to estimate $|\alpha|^2$.

Theorem 11. *[Amplitude amplification and estimation] [26][6] Given a unitary that performs the mapping* $U_x : |0\rangle \mapsto sin(\theta) |\mathbf{x},0\rangle + cos(\theta) |\mathbf{G},0^{\perp}\rangle$ *in time* T_U*, then* $sin(\theta)^2$ *can be estimated to multiplicative error* η *in time* $O(\frac{T(U_x)}{\eta sin(\theta)})$ *and* $|\mathbf{x}\rangle$ *can be generated in expected time* $O(\frac{T(U_x)}{sin(\theta)})$.

An example of algorithm that has been developed thanks to singular value estimation and amplitude estimation is an algorithm that estimates the spectral norm $||\mathbf{A}||_2$ of a matrix stored in QRAM. This procedure takes advantage of SVE with self-tomography, conditional rotations and amplitude estimation to compute the spectral norm of a matrix in time $\tilde{O}\left(\frac{log(\frac{1}{\varepsilon})}{\varepsilon\eta}\right)$, where $\eta = ||\mathbf{A}||_2/||\mathbf{A}||_F$, with absolute error $\varepsilon||\mathbf{A}||_F$ [25, Proposition 4.8 page 24].

Spectral norm estimation [25] Given a matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$ stored in QRAM, it gives an estimation of $||\mathbf{A}||_2/||\mathbf{A}||_F$.

Require:

Input matrix $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T \in \mathbb{R}^{n\times m}$ stored in QRAM.

Ensure:

An estimate for $\eta = ||\mathbf{A}||_2/||\mathbf{A}||_F$ with additive error ε.

1: Let $l = 0$ and $u = 1$ be lower and upper bounds for η and let $\tau = (l+u)/2$ an estimate for η to be refined using binary search in the next steps over $log(1/\varepsilon)$ iterations.

2: Prepare $|\mathbf{A}\rangle = \frac{1}{||\mathbf{A}_F||}\sum_i^n \sum_j^m a_{ij}|i\rangle|j\rangle = \frac{1}{\sqrt{\sum_j^r \sigma_j^2}}\sum_i^r \sigma_i |\mathbf{u}_i\rangle|\mathbf{v}_i\rangle$ and apply SVE with precision ε to get the state $\frac{1}{\sqrt{\sum_j^r \sigma_j^2}}\sum_i^r \sigma_i |\mathbf{u}_i\rangle|\mathbf{v}_i\rangle|\overline{\sigma_i}\rangle$ where $|\overline{\sigma_i} - \frac{\sigma_i}{||\mathbf{A}||_F}| \leq \varepsilon$.

3: Append a single qubit register $|R\rangle$ and perform a conditional rotation to set it to $|1\rangle$ if $\overline{\sigma} \geq \tau$ and $|0\rangle$ otherwise. Uncompute the SVE from step 2.

$$\frac{1}{||\mathbf{A}||_F}\sum_{i:\overline{\sigma}_i \geq \tau} \sigma_i |\mathbf{u}_i\rangle|\mathbf{v}_i\rangle|1\rangle + \frac{1}{||\mathbf{A}||_F}\sum_{j:\overline{\sigma}_j < \tau} \sigma_j |\mathbf{u}_j\rangle|\mathbf{v}_j\rangle|0\rangle$$

4: Perform amplitude estimation on the state above conditioned on the last register being 1 to estimate $\sum_{i:\overline{\sigma}_i \geq \tau} \sigma_i^2/||\mathbf{A}||_F$ to relative error $(1+\delta)$.

5: If the estimate in the step above is 0, then $u \to \tau$ else $l \to \tau$. Set $\tau = (u+l)/2$ and start again from 2.

Thanks to the structure of the QRAM, the Frobenius norm $||\mathbf{A}||_F$ of a matrix can be retrieved in constant time and the spectral norm is then computed as $||\mathbf{A}||_2 = \tau||\mathbf{A}||_F$. The detailed proof of the running time and error analysis of the algorithm can be found in [25].

Finally, the last statement we report allows us to retrieve the entries of a vector stored in the amplitudes of a quantum vector state.

Theorem 12 (Efficient vector state tomography)**.** *[26][24] Given a unitary mapping $U_x : |0\rangle \mapsto |\mathbf{x}\rangle$ in time $T(U_x)$ and $\delta > 0$, there is an algorithm that produces an estimate $\overline{\mathbf{x}} \in \mathbb{R}^n$ with $||\overline{\mathbf{x}}|| = 1$ such that $||\mathbf{x} - \overline{\mathbf{x}}|| \leq \delta$ with probability at least $(1 - 1/d^{0.83})$ in time $O(T(U_x)\frac{d\log d}{\delta^2})$.*

3.2.4 Related works

One of the first papers that faced the problem of performing the eigendecomposition of a quantum-encoded matrix is a work by Lloyd et al. [28]. In this work, the authors assume to have quantum access to a matrix in the form of a density matrix and develop a method for fast density matrix exponentiation that allows preparing the eigendecomposition of the full input matrix in time logarithmic on its dimensions. Hoever, this algorithm requires that the input matrix to be square, symmetric, and sparse or low-rank.

More recently, the works of Kerenidis et al. on Recommendation systems [23] and Least-squares [25] have used a different definition of quantum access to a matrix (the one used throughout this work) and defined the task of singular value estimation. Their singular value decomposition scales better with respect to the error parameters, eliminates the dependency on the condition number and does not require the input matrix to be square, symmetric and sparse or low-rank.

Several recent works, such as [27] [36] [17], have tried to improve or to extend the quantum singular value decomposition techniques, but very few of them have tried to sketch an algorithm that allows classical access to the singular values and singular vectors.

In the last few months, there have been attempts at creating near-term quantum algorithms for singular value decomposition. The authors of these works propose quantum circuits for singular value decomposition of quantum states on Noisy Intermediate-Scale Quantum (NISQ) devices using variational circuits [7] [44]. However, the complexity of such methods is still not clear and recent works have questioned the efficacy of the speed-ups of variational quantum algorithms due to noise-induced barren plateaus in the optimization landscape that often nullify the speed-ups [43].

The realization of quantum procedures that provide exponential speed-ups in linear algebra tasks has given inspiration for the realization of classical quantum-inspired algorithms that try to achieve the same run-time as their quantum counterparts. The process of transforming a quantum algorithm in a classical algorithm with a similar speed-up is usually referred to as de-quantization.

Most of this works are based on a famous algorithm by Frieze, Kannan, and Vempala which computes a low-rank approximation of a matrix in time which is sub-linear in the number of its elements [15] [8] [2]. Such algorithms promise exponential speed-ups over the traditional SVD algorithm for low-rank matrices. However, the high polynomial dependency of the run-times on the condition number, the rank, and the estimation error makes them advantageous only for matrices of extremely large dimensions, with low ranks and small condition numbers.

The research described in [2] suggests that the dependencies like $O\left(\frac{k^6}{\varepsilon^6}\right)$, where k is the rank of the low-rank approximation and ε is the precision parameter, are fundamental and can not be lowered further.

Chapter 4

Novel quantum algorithms for data representation

Building from the linear algebra knowledge discussed in Chapter 2 and the quantum computing procedures introduced in Chapter 3, we present and analyze a series of novel quantum algorithms that allow us to classically retrieve the singular value decomposition of a matrix loaded in QRAM.

After introducing the two main procedures and proving their runtime, we discuss how it is possible to exploit these algorithms to perform Principal Component Analysis, Correspondence Analysis, and Latent Semantic Analysis. For each of these three scenarios, we discuss the parameters that characterize the computational complexity of the quantum procedures and provide an error bound for the model computed using quantum routines.

4.1 Factor score ratios estimation

The first algorithm that we introduce in this chapter is a quantum routine that, given access to a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ of rank r stored in QRAM, allows us to retrieve the most relevant singular values (or eigenvalues of $\mathbf{A}^T\mathbf{A}$, $\mathbf{A}\mathbf{A}^T$) together with the amount of variance that they explain.

With a bit of abuse of terminology, we use the notation of [19] for Correspondence Analysis and we refer to the eigenvalues of $\mathbf{A}^T\mathbf{A}$ and $\mathbf{A}\mathbf{A}^T$ as factor scores $\lambda_i = \sigma_i^2$, and to the quantity $\frac{\lambda_i}{\sum_j^r \lambda_j} = \frac{\sigma_i^2}{\sum_j^r \sigma_j^2}$ as factor score ratio for the factor score λ_i.

Algorithms such as Principal Component Analysis and Correspondence Analysis are often used for data visualization or dimensionality reduction purposes: when this is the case, the assumption is that the greatest factor scores have a significant factor score ratio. The procedure that we provide in here is a fast procedure that allows verifying if this is the case

for the input matrix. The most interesting result is that the runtime of this subroutine is polylogarithmic on the number of elements of $\mathbf{A}$ and does not depend on its rank.

The main intuition behind this algorithm is that it is possible to create the state

$$\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i \left|\mathbf{u}_i\right\rangle \left|\mathbf{v}_i\right\rangle \left|\overline{\sigma_i}\right\rangle .$$

and that the third register, when measured in the computational base, is a random variable that outputs the estimates $\left|\overline{\sigma}_i\right\rangle$ of the singular values each with probability equal to the factor score ratio. In practice, it is possible to sample a certain number of times from this register to get an intuition of the distribution of the factor scores and their ratios.

When a matrix has a huge number of negligible singular values and only a few of them that are very big, the greatest ones will surely appear many times during the measurements, while the negligible ones are not likely to be measured. This intuition has been sketched in [28] for sparse or low rank square symmetric matrices, but has never been formalized.

We formalize it in Algorithm 1 and Theorem 13 using more recent techniques.

Quantum Factor Score Ratio Estimation - Algorithm 1 Compute an estimate of the distribution of the Factor Scores.

Require:

Input matrix $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ of rank r and size $n \times m$ stored in QRAM. Confidence level z, factor score estimation error ε, factor score ratio estimation error γ.

Ensure:

An estimate $\overline{\sigma}_i^2$ of the factor scores of $\mathbf{A}$ and an estimate $\hat{f}_i$ of their Factor Score Ratio, such that $|\sigma_i^2 - \overline{\sigma}_i^2| \leq \varepsilon$ with probability $1 - 1/polylog(n)$ and $|\frac{\sigma_i^2}{\sum_j^r \sigma_j^2} - \hat{f}_i| \leq \gamma$ with confidence level z.

1: $S = 0$
2: **while** $S < \frac{z^2}{4\gamma^2}$ **do**
3: Create the quantum state $\frac{1}{\sqrt{n}} \sum_i^n |i\rangle |0\rangle$ and query the QRAM to get the state

$$\frac{1}{||\mathbf{A}||_F} \sum_i^n \sum_j^m a_{ij} |i\rangle |j\rangle = \frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle$$

4: Apply consistent SVE with error $\gamma = \frac{\varepsilon}{2}$ on the state above to get the state

$$\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle$$

5: Measure the register containing $|\overline{\sigma}_i\rangle$ and store a counter of how many times this value has been measured
6: $S = S + 1$
7: **end while**
8: For each measured singular value $\overline{\sigma}_i$ classically compute the factor score $\overline{\sigma}_i^2$
9: For each measured singular value $\overline{\sigma}_i$ classically compute its factor score ratio estimator

$$\hat{f}_i = \frac{\zeta_{\overline{\sigma}_i}}{S}$$

where $\zeta_{\overline{\sigma}_i}$ is the number of times $|\overline{\sigma}_i\rangle$ has been observed among the measurements.

Theorem 13 (Quantum Factor Score Ratio Estimation)**.** *Let* $\mathbf{A} \in \mathbb{R}^{n\times m}$ *be a matrix with singular value decomposition* $\mathbf{A} = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ *stored in QRAM with* $|\sigma_{max}| \leq 1$*. Let* γ, ε *be precision parameters. There exists a quantum algorithm that estimates the factor scores* σ_i^2 *- or the singular values* σ_i *- and the factor scores ratios* $\frac{\sigma_i^2}{\sum_j^r \sigma_j^2}$ *of* $\mathbf{A}$ *in time* $O(\frac{1}{\gamma^2}\frac{\mu(\mathbf{A})}{\varepsilon} polylog(nm))$ *such that* $|\sigma_i^2 - \overline{\sigma}_i^2| \leq \varepsilon$ *with probability* $1 - 1/polylog(n)$ *and* $|\frac{\sigma_i^2}{\sum_j^r \sigma_j^2} - \hat{f}_i| \leq \gamma$ *with high confidence.*

Proof. The proof consists in proving the time complexity of Algorithm 1.

Creating the quantum state $\frac{1}{\sqrt{n}}\sum_i^n |i\rangle|0\rangle$ and querying the QRAM in step 3 has cost $O(polylog(nm))$ where n and m are the dimensions of the matrix $\mathbf{A}$, as a consequence of Definition 8 and Theorem 6.

The SVE algorithm from Theorem 8 runs in time $O(\frac{\mu(\mathbf{A})}{\tau} polylog(nm))$ and provides estimates $|\sigma_i - \overline{\sigma}_i| < \tau$, where $\tau > 0$ is a precision parameter, with probability $1 - 1/poly(n)$.

If the goal is to estimate the singular values σ_i then trivially $\tau = \varepsilon$ and the complexity of this step is $\tilde{O}(\frac{\mu(\mathbf{A})}{\varepsilon})$.

On the other hand, if the goal is to estimate σ_i^2, we need to prove that $\tau \sim O(\varepsilon)$.

Running the SVE with precision τ means that for each singular value σ_i, with probablilty $1 - polylog(n)$, we have an estimate $\overline{\sigma}_i$ that in the worst case is $\overline{\sigma}_i = \sigma_i \pm \tau$. From the worst case estimate of σ_i, it follows that the worst case estimate of σ_i^2 is

$$\overline{\sigma}_i^2 = (\sigma_i \pm \tau)^2 = \sigma_i^2 \pm 2\sigma_i\tau + \tau^2$$

and since $|\sigma_i| \leq 1$, we can say that in the worst case

$$\overline{\sigma}_i^2 = \sigma_i^2 + (2\tau + \tau^2).$$

Solving the equation $2\tau + \tau^2 = \varepsilon$ for $\tau > 0$ leads to $\tau = \sqrt{1+\varepsilon} - 1$.

It is possible to show that $\tau \sim O(\varepsilon)$:

$$\tau = \sqrt{1+\varepsilon} - 1 = \frac{(\sqrt{1+\varepsilon}-1)(\sqrt{1+\varepsilon}+1)}{(\sqrt{1+\varepsilon}+1)} = \frac{1+\varepsilon-1}{\sqrt{1+\varepsilon}+1} = \frac{\varepsilon}{\sqrt{1+\varepsilon}+1} \sim \frac{\varepsilon}{2}.$$

This proves that, both when estimating the singular values σ_i and the factor scores σ_i^2, the cost of computing SVE at step 4 is $O\left(\frac{\mu(\mathbf{A})}{\varepsilon} polylog(nm)\right)$.

Finally, the third register of the state at step 5, when measured, can collapse to any state $|\overline{\sigma}_i\rangle$, each with probability $p_i = \frac{\sigma_i^2}{\Sigma_j^r \sigma_j^2}$. The next step of the proof is to understand how many times the state at step 5 has to be produced and measured in order to compute the estimates $\hat{f}_i$ with the guarantees stated in the theorem.

As already explained in Section 3.1.5, to solve the problem of estimating the factor scores ratios we can consider the measurement process of this state as doing r Bernoulli trials: one for each factor score, so that if we measure $\overline{\sigma}_i$ it is a success for the i^{th} Bernoulli trial and a failure for all the others.

To compute the number of samples that are necessary for a good estimation of a factor score, we can exploit the Wald confidence interval. If we choose $\hat{f}_i = \frac{\zeta_{\overline{\sigma}_i}}{S}$ as an estimator for $f_i = \frac{\sigma_i^2}{\Sigma_j^r \sigma_j^2}$, with $S = \frac{z^2}{4\gamma^2}$ measurements we have that $|\frac{\sigma_i^2}{\Sigma_j^r \sigma_j^2} - \hat{f}_i| \leq \gamma$ with confidence given by the z-score z, for each $\hat{f}_i$.

To sum up, the cost to prepare the state at step 5 is $O\left(\frac{\mu(\mathbf{A})}{\varepsilon} polylog(nm)\right)$ and the number of times it has to be measured is $O\left(\frac{1}{\delta^2}\right)$. It follows that the overall complexity of the algorithm is

$$O\left(\frac{1}{\delta^2}\frac{\mu(A)}{\varepsilon} polylog(nm)\right)$$

□

Claim 1. *If the matrix had spectral norm* $||\mathbf{A}||$ *before being stored in QRAM and normalized, then* $||\mathbf{A}||\sigma_i$ *and* $||\mathbf{A}||^2\sigma_i^2$ *estimate the un-normalized singular values and factor scores within error* $\varepsilon||\mathbf{A}||$ *and* $\varepsilon||\mathbf{A}||^2$ *respectively.*

Proof. The normalized version of $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$ is $\frac{\mathbf{A}}{||\mathbf{A}||} = \mathbf{U}\frac{\Sigma}{||\mathbf{A}||}\mathbf{V}^T$. The singular values $\{\sigma_i\}_i^r$ of the normalized matrix, whose estimation is provided by algorithm 1, are the entries of $\frac{\Sigma}{||\mathbf{A}||}$. Therefore, the singular values and factor scores of the original matrix $\mathbf{A}$ are respectively $\{\sigma_i||\mathbf{A}||\}_i^r$ and $\{\sigma_i^2||\mathbf{A}||^2\}_i^r$.
Algorithm 1 guarantees that

$$|\overline{\sigma}_i - \sigma_i| \leq \varepsilon.$$

We can multiply this inequality on both sides by the positive value $||\mathbf{A}||$

$$||\mathbf{A}||\,|\overline{\sigma}_i - \sigma_i| \leq \varepsilon||\mathbf{A}||$$

and let the term $||\mathbf{A}||$ enter the absolute value

$$|||\mathbf{A}||\overline{\sigma}_i - ||\mathbf{A}||\sigma_i| \leq \varepsilon||\mathbf{A}||.$$

The proof for $\left|||\mathbf{A}||^2\overline{\sigma}_i^2 - ||\mathbf{A}||^2\sigma_i^2\right| \leq \varepsilon||\mathbf{A}||^2$ proceeds similarly. □

In this formalization, the parameter γ is the one that controls big a factor score ratio should be for the singular value to be measured. If we choose γ bigger than the factor scores ratios in which we are not interested in, the estimator for the factor scores is likely to be 0 (indeed, since $f_i \leq \gamma$ then $|0 - f_i| \leq \gamma$ is a plausible estimation) and we are not seeing the factor score among the measurement outcomes. However, if the factor score ratio is significantly greater than γ, then the associated singular value is likely to be measured.

If the measurements reveal a uniform distribution of factor score ratios, it means that the assumption for few relevant factor scores falls and one can either drop the task and change methodology or increment the number of measurements according to how many relevant singular values he/she is willing to tolerate.

One limitation of this approach is that the Wald confidence interval does not provide an error guarantee for the estimation of the sum of the factor score ratios with the same confidence level z used in the procedure.

In other words, if we have k factor score ratios estimates $\hat{f}_i$ so that for each one $|\hat{f}_i - f_i| \leq \varepsilon$ with 99% confidence, we cannot state that $|\sum_i^k \hat{f}_i - \sum_i^k f_i| \leq k\varepsilon$ with 99% confidence, but with $(0.99)^k \cdot 100\%$ confidence. To overcome this problem one can either adjust the confidence level accordingly after a first run of the algorithm or use the procedure in Algorithm 2, Theorem 14. Moreover, in Chapter 6, we discuss an alternative approach to bound the sum of k factor score ratios to absolute error $k\gamma$ with probability $1 - 1/poly(r)$.

Quantum factor score sum check - Algorithm 2 Gives an estimate on the amount of variance captured by the singular values greater than a threshold θ.

Require:

Input matrix $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ of rank r and size $n \times m$ stored in QRAM. A threshold $\theta > 0$. Two error parameters ε, $\eta > 0$.

Ensure:

An estimator of the explained variance $\frac{\sum_{i:\overline{\sigma_i} \geq \theta} \sigma_i^2}{\sum_j^r \theta_j^2}$ with relative error η.

1: Create the quantum state $\frac{1}{\sqrt{n}} \sum_i^n |i\rangle$ and query the QRAM to get the state $\frac{1}{||\mathbf{A}||_F} \sum_i^n \sum_j^m a_{ij} |i\rangle |j\rangle = \frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle$

2: Apply SVE, with the same precision parameter ε used to establish the threshold θ, on the state above to get $\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma_i}\rangle$

3: Append a quantum register $|R\rangle$ to the state and perform a conditional rotation to get the state

$$\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_{i:\overline{\sigma_i} \geq \theta} \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle |0\rangle + \frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_{i:\overline{\sigma_i} < \theta} \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle |1\rangle$$

4: Uncompute the SVE to get the state

$$\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_{i:\overline{\sigma_i} \geq \theta} \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |0\rangle + \frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_{i:\overline{\sigma_i} < \theta} \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |1\rangle$$

5: Perform Amplitude Estimation with precision η on the last register $|0\rangle$, to estimate $p = \frac{\sum_{i:\overline{\sigma_i} \geq \theta} \sigma_i^2}{\sum_j^r \sigma_j^2}$.

Theorem 14 (Quantum check on explained variance)**.** *Let $\mathbf{A} \in \mathbb{R}^{n \times m}$ be a matrix with singular value decomposition $\mathbf{A} = \sum_i \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ stored in QRAM. Let η be a precision parameter, θ a threshold for the smallest singular value, ε a precision parameter for the singular values. There exists a quantum algorithm that estimates $p = \frac{\sum_{i:\overline{\sigma_i} \geq \theta} \sigma_i^2}{\sum_j^r \sigma_j^2}$ to relative error η in time $O\left(\frac{\mu(\mathbf{A})}{\varepsilon} \frac{1}{\eta \sqrt{p}} polylog(nm)\right)$.*

Proof. The proof consists in proving the time complexity of Algorithm 2.
As already discussed in the previous proof, the complexity of querying the QRAM in step 1 is $O(polylog(nm))$.

Step 2 can be performed using Theorem 8 in time $O\left(\frac{\mu(\mathbf{A})}{\varepsilon} polylog(nm)\right)$.

The use of a conditional rotation requires a quantum circuit that acts on the number of qubits of interest and it can be implemented such that its cost is negligible. Therefore, the complexity of step 3 can be considered in the order of $O(polylog(nm))$.

Uncomputing SVE has the same complexity of applying the SVE algorithm in step 2: if $\mathbf{U}_{SVE}$ performs singular value estimation, then $\mathbf{U}_{SVE}^T$ undoes it. Indeed $\mathbf{U}_{SVE}^T \mathbf{U}_{SVE} |\varphi\rangle = \mathbb{I} |\varphi\rangle$.
The cost of obtaining the quantum state at step 4 is $T(U_4) = O\left(\frac{\mu(\mathbf{A})}{\varepsilon} polylog(nm)\right)$.

Finally, the cost of amplitude estimation (Theorem 11), on the last register being $|0\rangle$ and with relative precision η, is equal to $O\left(T(U_4 \frac{1}{\eta\sqrt{p}})\right)$. Where p is the probability of measuring $|0\rangle$, which can be easily seen to be $p = \frac{\sum_{i:\overline{\sigma_i} \geq \theta} \sigma_i^2}{\sum_j^r \sigma_j^2}$ by applying the measurement operator $M_0 = \mathbb{I} \otimes \mathbb{I} \otimes |0\rangle \langle 0|$ on the state at 4, using the 3^{rd} Postulate of quantum mechanics stated in Section 3.1.5.

The final complexity of the algorithm is then:

$$O\left(\frac{\mu(\mathbf{A})}{\varepsilon} \frac{1}{\eta\sqrt{p}} polylog(nm))\right).$$

□

This procedure allows to compute the sum of the factor score ratios associated to all the singular values of $\mathbf{A}$ that are above a threshold δ. It is a very fast check as it runs in time $O\left(\frac{\mu(\mathbf{A})}{\varepsilon} \frac{1}{\eta\sqrt{p}} polylog(nm)\right)$ and we expect p to be a value closer to 1 rather then to 0. It can be used in combination with Algorithm 1: by running Algorithm 1, one can first have an idea of the threshold above which the singular value have a significant factor score ratio and then run Algorithm 2 to better check the total sum of the factor score ratios.

In practice, in problem such as PCA, the probability p is a number in the range $[1/2, 1]$ and the precision required is up to the second decimal digit $\eta = 0.01$. For this reason, we can consider the complexity of the above algorithm as $\tilde{O}\left(\frac{\mu\mathbf{A}}{\varepsilon}\right)$.

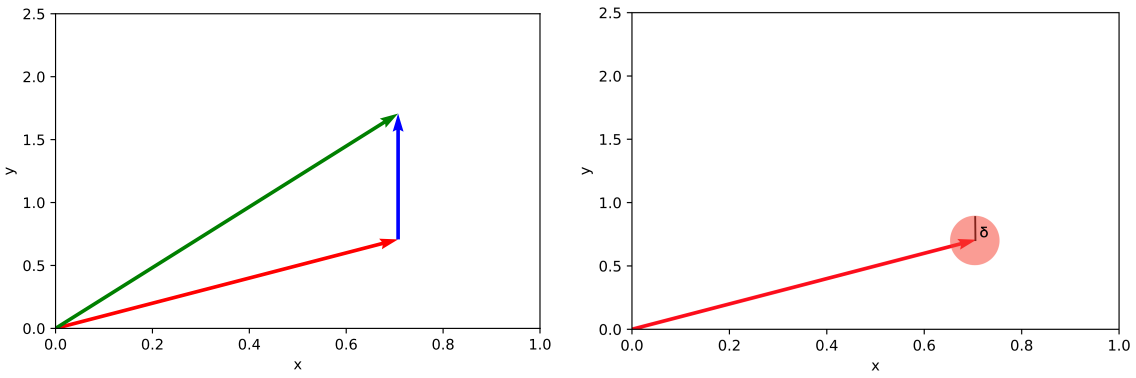
4.2 Top-k singular vectors extraction

The second algorithm that we present in this work is a quantum procedure that allows us to extract the top-k right or left singular vectors of a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ stored in QRAM and normalized so that $\sigma_{max} \leq 1$.

The inputs of this algorithm, other than the matrix, are a precision parameter δ for the estimation of each singular vector, a precision parameter ε for the singular value estimation, and a threshold θ such that the top-k singular values are greater or equal to that threshold.

Depending on the final application, the parameter θ can be either computed using the procedures of Theorems 13 and 14 or empirically determined according to the particular type of dataset $\mathbf{A}$.

This algorithm guarantees an estimate $\overline{\mathbf{u}}_i$ of each left (right) singular vector $\mathbf{u}_i$ such that $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$ and $||\overline{\mathbf{u}}_i|| = 1$.



(a) The blu vector is the *difference vector* between the green one and the red one.

(b) Let $\mathbf{b}$ be the red vector. The circle shows where all the vectors $\mathbf{x}$ such that $||\mathbf{a} - \mathbf{b}|| \leq \delta$ lie.

Fig. 4.1 Visualizing the meaning of $||\mathbf{a} - \mathbf{b}|| \leq \delta$ in $\mathbb{R}^2$.

Remembering that the ℓ_2 norm of a vector is the length of the vector, it is possible to have a visual understanding of this constraint in the $\mathbb{R}^2$ space by observing Figures 4.1a and 4.1b.

In practice, this constraint on the ℓ_2 norm of the *difference vector* ensures that the estimated vector has the same length and orientation of the original vector. In our specific case, the vector $\overline{\mathbf{u}}_i$ is a unit vector, so it can only lie at the intersection between the unit circle and the δ-circle around vector $\mathbf{u}_i$.

The algorithm is formalized as follows.

Quantum top singular vectors extraction - Algorithm 3 Given a $\mathbf{A} \in \mathbb{R}^{n \times m}$ matrix and a threshold δ, extract the top left (right) singular vectors.

Require:

Input matrix $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ of rank r and size $n \times m$ stored in QRAM. A threshold $\theta > 0$ and one error parameter ε.

Ensure:

An estimation $\overline{\mathbf{u}}_i$ of the left (right) singular vectors associated to singular values greater than θ, such that $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$.

1: Create the quantum state $\frac{1}{\sqrt{n}} \sum_{i=0}^{n-1} |i\rangle$ and query the QRAM to get the state $\frac{1}{||\mathbf{A}||_F} \sum_i^n \sum_j^m a_{ij} |i\rangle |j\rangle = \frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle$

2: Apply SVE with the precision parameter ε

3: Append a quantum register $|R\rangle$ to the state and perform a conditional rotation to get the state

$$\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_{i:\overline{\sigma}_i \geq \theta} \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}\rangle |0\rangle + \frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_{i:\overline{\sigma}_i < \theta} \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}\rangle |1\rangle$$

4: Perform Amplitude Amplification on the last register being $|0\rangle$

$$\frac{1}{\sqrt{\sum_j^k \sigma_j^2}} \sum_i^k \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}\rangle = \frac{1}{||\mathbf{A}^{(k)}||_F} \sum_i^k \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}\rangle$$

5: Append another ancillary register $|R'\rangle$ and perform the following controlled rotation

$$\frac{C}{||\mathbf{A}^{(k)}||_F} \sum_i^k \frac{\sigma_i}{\overline{\sigma}_i} |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle |0\rangle + \frac{1}{||\mathbf{A}^{(k)}||_F} \sum_i^k \sqrt{1 - \frac{C^2}{\overline{\sigma}_i^2}} |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle |1\rangle$$

6: Perform again Amplitude Amplification on the last register being $|0\rangle$

$$\frac{1}{\sqrt{k}} \sum_i^k |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}\rangle$$

7: Measure the last register and according to the measured value $|\overline{\sigma}_i\rangle$ apply vector tomography for the state vector $|\mathbf{u}_i\rangle$ ($|\mathbf{v}_i\rangle$).

Theorem 15 (Top-k singular vectors extraction)**.** *Let* $\mathbf{A} \in \mathbb{R}^{n\times m}$ *be a matrix with singular value decomposition* $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i$ *stored in QRAM with* $|\sigma_{max}| \leq 1$*. Let* $\delta > 0$ *be a precision parameter,* ε *the precision parameter for the singular values, and* θ *be a threshold such that* $\mathbf{A}$ *has k singular values greater than* θ*. Define* $p = \frac{\sum_j^k \sigma_j^2}{\sum_i^r \sigma_i^2}$*. Then, there exist a quantum algorithm that estimates*

- *The top k left singular vectors* $\{\mathbf{u}_i\}_i^k$ *of* $\mathbf{A}$ *with unit vectors* $\{\overline{\mathbf{u}_i}\}_i^k$ *such that* $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$ *with probability at least* $(1 - 1/n^{0.83})$*, in time* $\tilde{O}(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{kn\log(n+k)}{\delta^2})$.
- *The top k right singular vectors* $\{\mathbf{v}_i\}_i^k$ *of* $\mathbf{A}$ *with unit vectors* $\{\overline{\mathbf{v}_i}\}_i^k$ *such that* $||\overline{\mathbf{v}}_i - \mathbf{v}_i|| \leq \delta$ *with probability at least* $(1 - 1/m^{0.83})$*, in time* $\tilde{O}(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{km\log(m+k)}{\delta^2})$.

Proof. The proof consists in proving the time complexity of Algorithm 3.

Preparing the state $\frac{1}{\sqrt{n}}\sum_{i=0}^{n-1}|i\rangle$ and querying the QRAM in step 1 to get the matrix takes time $O(polylog(nm))$. Since we are using the $\tilde{O}()$ notation, the cost of this step is $\tilde{O}(1)$.

Step 2, executed with precision ε, has a computational cost of $\tilde{O}\left(\frac{\mu(\mathbf{A})}{\varepsilon}\right)$. The precision parameter ε has to be enough to recognize the singular values associated to the singular vectors in the correct order. We assume that the user already knows the singular values of $\mathbf{A}$ from Algorithm 1 or that has knowledge from the literature of the usual distances between the top singular values of the type of matrix under analysis.

The cost of the conditional rotations of steps 3 and 5 is negligible as it only scales with the size of the registers. For this reason, it is $\tilde{O}(1)$.

In the state at 3, the third register has a probability $p = \frac{\sum_j^k \sigma_j^2}{\sum_i^r \sigma_i^2}$ of collapsing to $|0\rangle$ when measured in the computational basis. Equation 3.4 from Postulate 3 in Section 3.1.5 ensures us that the state after the measurement with $M_0 = \mathbb{I}\otimes\mathbb{I}\otimes|0\rangle\langle 0|$ is

$$\frac{1}{\sqrt{\sum_j^k \sigma_j^2}}\sum_i^k \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}\rangle .$$

The cost $T(U_5)$ of preparing this state is the cost of running amplitude amplification on the state at 3 and is therefore $\tilde{O}\left(\frac{\mu(\mathbf{A})}{\varepsilon}\frac{1}{\sqrt{p}}\right)$.

The next step to analyze is the amplitude amplification at 6 on the last register being $|0\rangle$ in the state

$$\frac{1}{||\mathbf{A}^{(k)}||_F}\sum_i^k \sigma_i |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle \left(\frac{C}{\overline{\sigma}_i}|0\rangle + \sqrt{1-\frac{C^2}{\overline{\sigma}_i^2}}|1\rangle\right) =$$

$$\frac{C}{||\mathbf{A}^{(k)}||_F}\sum_i^k \frac{\sigma_i}{\overline{\sigma}_i} |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle |0\rangle + \frac{1}{||\mathbf{A}^{(k)}||_F}\sum_i^k \sqrt{1-\frac{C^2}{\overline{\sigma}_i^2}} |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle |1\rangle$$

The constant C is a normalization factor and it is in the order of $\tilde{O}(1/\kappa)$ where κ is the condition number $\kappa = \frac{\sigma_{max}}{\sigma_{min}}$ of the low-rank matrix $\mathbf{A}^{(k)}$. Since for construction its $\sigma_{max} \leq 1$ and $\sigma_{min} \geq \theta$, we can bound the condition number

$$\kappa \leq \frac{1}{\theta}.$$

From the famous work of Harrow, Hassidim and Lloyd [18] we know that applying amplitude amplification on the state above, with the condition of the third register being $|0\rangle$, would cost $\tilde{O}(\kappa T(U_5))$. In our case the cost is $\tilde{O}\left(\frac{1}{\theta}\frac{1}{\sqrt{p_f}}\frac{\mu(A)}{\varepsilon}\right)$.
Amplitude amplification leaves the registers in the state

$$\frac{1}{\sqrt{\sum_i^k \frac{\sigma_i^2}{\overline{\sigma}_i^2}}}\sum_i^k \frac{\sigma_i}{\overline{\sigma}_i} |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle \sim \frac{1}{\sqrt{k}}\sum_i^k |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle$$

since each $\overline{\sigma}_i = \sigma_i \pm \varepsilon$ and $\frac{\sigma_i}{\sigma_i \pm \varepsilon} \to 1$ as $\varepsilon \to 0$.

The last step of the proof is to compute the time complexity for the vector state tomography at step 7.

When measuring the last register of the state above in the computational basis and obtaining $|\overline{\sigma}_i\rangle$, the first two registers collapse in the state $|\mathbf{u}_i\rangle |\mathbf{v}_i\rangle$. On this $|\mathbf{u}_i\rangle |\mathbf{v}_i\rangle$ state it is possible to perform vector-state tomography using Theorem 12 either on the first register, to retrieve $\overline{\mathbf{u}}_i$, or on the second one, to retrieve $\overline{\mathbf{v}}_i$. Theorem 12 guarantees estimates such that $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$ with probability $(1-n^{0.83})$ and that $||\overline{\mathbf{v}}_i - \mathbf{v}_i|| \leq \delta$ with probability $(1-m^{0.83})$. Moreover, it guarantees that the estimates are with unit norm.

Recall that $\mathbf{u} \in R^n$ and $\mathbf{v} \in R^m$, so performing vector state tomography takes time $\tilde{O}(\frac{n\log n}{\delta^2})$ for the first register $|\mathbf{u}_i\rangle$ and time $\tilde{O}(\frac{m\log m}{\delta^2})$ for the second register $|\mathbf{v}_i\rangle$.

Using a Coupon Collector's argument [13], if the k states $|\overline{\sigma}_i\rangle$ are uniformly distributed, to get all the k possible states $|\mathbf{u}_i\rangle |\mathbf{v}_i\rangle$ at least once we would have needed $k\log k$ measurements

on average. In Section 5.3.3, we provide experiments that show how the Coupon Collector's run-time varies as ε increases.

To perform tomography correctly for each state, one should satisfy the coupon collector's problem the same number of times as the number of measurements needed by the vector state tomography procedure. Therefore, defining as $T(U_6)$ the cost of preparing the state at the end of step 6, the costs of the tomography for all the vectors $\overline{\mathbf{u}}_i$ and $\overline{\mathbf{v}}_i$ are:

$$\tilde{O}\left(T(U_6)\frac{k\log k \cdot n\log n}{\delta^2}\right), \tilde{O}\left(T(U_6)\frac{k\log k \cdot m\log m}{\delta^2}\right).$$

The following complexities are then proven:

$$\tilde{O}(\frac{1}{\theta}\frac{1}{\sqrt{p_f}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{k\log k \cdot n\log n}{\delta^2}), \tilde{O}(\frac{1}{\theta}\frac{1}{\sqrt{p_f}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{k\log k \cdot m\log m}{\delta^2}).$$

□

As a final remark, note that in most of the final applications, such as PCA or CA, one specifies θ such that the sum of factor score ratios of the top-k singular values is large, usually in the range $[1/2, 1]$. For this reason, the term $\frac{1}{\sqrt{p}}$ is often negligible.

4.3 Quantum Principal Component Analysis

In Chapter 2, Section 2.3.1, we have described the technique of Principal Component Analysis for dimensionality reduction and how this technique is related to the singular value decomposition if the input matrix. We have shown that the output of PCA on a matrix $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T \in \mathbb{R}^{n\times m}$ is the rank-k matrix $\mathbf{A}' = \mathbf{U}^{(k)}\Sigma^{(k)} \in \mathbb{R}^{n\times k}$, such that the k diagonal entries of $\Sigma^{(k)}$ capture a certain percentage of variance $\frac{\sum_i^k \sigma_i^2}{\sum_j^r \sigma_j^2} \geq p$, generally in the range $[1/2, 1]$. In this section, we propose an algorithm that leverages the two quantum procedures of Theorems 13 and 15 to compute the model of PCA and extract it. We discuss the time complexity of this algorithm and analyze the accuracy to which it is possible to approximate the model $\mathbf{A}'$. The algorithm is reported with the time complexity of each step.

Quantum principal component analysis - Algorithm 4 Given a $\mathbf{A} \in \mathbb{R}^{n\times m}$ matrix and a threshold δ, extract the top left (right) singular vectors.

Require:

Input matrix $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i^T$ of rank r, size $n \times m$, maximum singularr value $\sigma_{max} \leq 1$, and zero mean stored in QRAM. Percentage of explained variance p.

Ensure:

An approximation of $\frac{\mathbf{U}\Sigma}{||\mathbf{A}||}$.

1: Use algorithm 1 to get an ε-estimate of the singular values and a γ-estimate of their explained variance. Select a threshold θ such that the sum of the explained variance for the top k singular values is greater or equal than p.

$$\tilde{O}\left(\frac{1}{\gamma^2}\frac{\mu(\mathbf{A})}{\varepsilon}\right)$$

2: Optionally check the explained variance using algorithm 2.

$$\tilde{O}\left(\frac{\mu(\mathbf{A})}{\varepsilon}\right)$$

3: Using algorithm 3, extract the top-k left singular vectors using threshold θ with precision δ.

$$\tilde{O}\left(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{klog(k)nlog(n)}{\delta^2}\right)$$

4: Compute the multiplication between singular values and singular vectors to obtain the matrix $\frac{\overline{\mathbf{U}\Sigma}}{||\mathbf{A}||}$.

$$O(nk)$$

The time complexity of the whole algorithm is:

$$O\left(\left(\frac{1}{\gamma^2}+\frac{1}{\theta}\frac{knlog(n+k)}{\delta^2}\right)\mu(\mathbf{A})\frac{polylog(nm)}{\varepsilon}\right).$$

Note that, as discussed in Section 4.2, in practice we can omit the term $\frac{1}{\sqrt{p}}$ as the probability is often a number greater than 0.5 (usually in the order of 0.8 or 0.9). Moreover, it is likely that even the term $\frac{1}{\theta}$ can be ignored, as we expect that in practice this value will not depend on the size of the matrix.

Finally, for matrices so that few singular values can capture a large amount of variance, the term $\frac{1}{\gamma^2}$ can be chosen lower than $min(n,m)$, and the asymptotic complexity becomes

$$O\left(\frac{\mu(\mathbf{A})knlog(n+k)}{\varepsilon\delta^2}polylog(nm)\right).$$

In conclusion, the overall complexity of using qPCA is given by the sum of the complexity required to store the normalized matrix $\mathbf{A}$ in the QRAM and the one required to perform the retrieval of the model

$$O\left(nmlog^2(nm)+\frac{log(1/\varepsilon)}{\varepsilon}\frac{||\mathbf{A}||_F}{||\mathbf{A}||}polylog(nm)+\frac{\mu(\mathbf{A})knlog(n+k)}{\varepsilon\delta^2}polylog(nm)\right)$$

In most of the real applications the bottleneck of this complexity is in the cost of storing the data in the QRAM and that this procedure runs sub-linearly on the number of elements of the input matrix when the matrices are large and suited for PCA, offering a polynomial speed-up over the traditional techniques.

To conclude, we claim that it is possible to reconstruct the model $\mathbf{A}' = \mathbf{U}^{(k)}\Sigma^{(k)}$ such that $||\overline{\mathbf{A}}' - \mathbf{A}'||_F \leq \sqrt{k}(\varepsilon+\delta)$. The experiments in Section 5.3.2 show that this is a reasonable error.

Lemma 2 (Accuracy of $\mathbf{U}\Sigma$ and $\mathbf{V}\Sigma$)**.** *Let $\mathbf{A} \in \mathbb{R}^{n\times m}$ be a matrix of rank k with singular value decomposition $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$ such that $\sigma_{max} \leq 1$ and let $\varepsilon, \delta > 0$ be two precision parameters. Given some approximate procedures that allow to retrieve an estimate $\overline{\sigma}_i$ of the singular values σ_i such that $|\overline{\sigma}_i - \sigma_i| \leq \varepsilon$ and an estimate $\overline{\mathbf{u}}_i$ of the left singular vectors $\mathbf{u}_i$ such that $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$, the error on the estimate $\overline{\mathbf{U}\Sigma}$ can be bounded as*

$$\left|\left|\overline{\mathbf{U}\Sigma} - \mathbf{U}\Sigma\right|\right|_F \leq \sqrt{k}(\varepsilon+\delta).$$

Similarly, if the approximate procedures output the singular values and the right singular vectors, the error bound for $\overline{\mathbf{V}\Sigma}$ *is*

$$\left|\left|\overline{\mathbf{V}\Sigma} - \mathbf{V}\Sigma\right|\right|_F \leq \sqrt{k}(\varepsilon + \delta).$$

Proof. Let's prove this result for $\left|\left|\overline{\mathbf{U}\Sigma} - \mathbf{U}\Sigma\right|\right|_F$.
We know by hypothesis that $|\overline{\sigma}_i - \sigma_i| \leq \varepsilon$, that $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$ and that $\sigma_{max} \leq 1$. Furthermore, since $\mathbf{U}$ is an orthonormal matrix, its columns have unitary norm $||\mathbf{u}_i|| = 1$.

The first step of the proof consists in bounding the error on the columns $||\overline{\sigma_i}\overline{\mathbf{u}}_i - \sigma_i\mathbf{u}_i||$.

$$||\overline{\sigma}_i\overline{\mathbf{u}}_i - \sigma_i\mathbf{u}_i|| \leq ||(\sigma_i \pm \varepsilon)\overline{\mathbf{u}}_i - \sigma_i\mathbf{u}_i|| =$$

$$||\sigma_i\overline{\mathbf{u}}_i \pm \varepsilon\overline{\mathbf{u}}_i - \sigma_i\mathbf{u}_i|| = ||\sigma_i(\overline{\mathbf{u}}_i - \mathbf{u}_i) \pm \varepsilon\overline{\mathbf{u}}_i||$$

Because of the triangular inequality

$$||\sigma_i(\overline{\mathbf{u}}_i - \mathbf{u}_i) \pm \varepsilon\overline{\mathbf{u}}_i|| \leq ||\sigma_i(\overline{\mathbf{u}}_i - \mathbf{u}_i)|| + ||\pm\varepsilon\overline{\mathbf{u}}_i|| =$$

$$\sigma_i\,||(\overline{\mathbf{u}}_i - \mathbf{u}_i)|| + \varepsilon\,||\overline{\mathbf{u}}_i||$$

Recall that by hypothesis, $||(\overline{\mathbf{u}}_i - \mathbf{u})|| \leq \delta$. Moreover, $||\overline{\mathbf{u}}_i|| = 1$ because of Theorem 12.

$$\sigma_i\,||(\overline{\mathbf{u}}_i - \mathbf{u}_i)|| + \varepsilon\,||\overline{\mathbf{u}}_i|| \leq \sigma_i\delta + \varepsilon$$

Finally, since $\sigma_i \leq \sigma_{max} \leq 1$, we get that $\sigma_i\delta + \varepsilon \leq \delta + \varepsilon$.
So an error bound on the columns is:

$$||\overline{\sigma}_i\overline{\mathbf{u}}_i - \sigma_i\mathbf{u}_i|| \leq \delta + \varepsilon$$

From the error bound on the columns and the fact that $f(x) = \sqrt{x}$ is an increasing monotone function, it is possible to prove the error bound on the matrices:

$$\left|\left|\overline{\mathbf{U}\Sigma} - \mathbf{U}\Sigma\right|\right|_F = \sqrt{\sum_i^n \sum_j^k (\left|\overline{\sigma}_j\overline{u}_{ij} - \sigma_j u_{ij}\right|)^2} = \sqrt{\sum_j^k \left(\sqrt{\sum_i^n (\overline{\sigma}_j\overline{u}_{ij} - \sigma_j u_{ij})^2}\right)^2} =$$

$$\sqrt{\sum_{j}^{k}\left(\left|\left|\overline{\sigma}_j\overline{\mathbf{u}}_j-\sigma_j\mathbf{u}_j\right|\right|\right)^2}\leq\sqrt{\sum_{j}^{k}(\varepsilon+\delta)^2}=\sqrt{k(\varepsilon+\delta)^2}=\sqrt{k}(\varepsilon+\delta)$$

The proof for $\left|\left|\overline{\mathbf{V}\Sigma}-\mathbf{V}\Sigma\right|\right|_F$ proceeds analogously. □

Claim 2. *If* $\mathbf{A}$ *has been normalized before retrieving the singular values and singular vectors and one wants to retrieve the representations of Lemma 2 for the un-normalized matrix, the estimated matrices are* $||\mathbf{A}||\overline{\mathbf{U}\Sigma}$ *and* $||\mathbf{A}||\overline{\mathbf{V}\Sigma}$ *and the bounds become*

- $\left|\left|||\mathbf{A}||\overline{\mathbf{U}\Sigma}-||\mathbf{A}||\mathbf{U}\Sigma\right|\right|_F\leq\sqrt{k}||\mathbf{A}||(\varepsilon+\delta)$
- $\left|\left|||\mathbf{A}||\overline{\mathbf{V}\Sigma}-||\mathbf{A}||\mathbf{V}\Sigma\right|\right|_F\leq\sqrt{k}||\mathbf{A}||(\varepsilon+\delta)$

Proof. To see that the desired data representations of the un-normalized matrix are $||\mathbf{A}||\mathbf{U}\Sigma$ and $||\mathbf{A}||\mathbf{V}\Sigma$ it is sufficient to recall, from the proof of Claim 1, that the singular values of the un-normalized matrix are just scaled by a factor $\frac{1}{||\mathbf{A}||}$, while the singular vectors remain the same.

Finally, to prove the bounds, we can just multiply both sides of the inequalities from Lemma 2 by the spectral norm $||\mathbf{A}||$:

$$\left|\left|||\mathbf{A}||\overline{\mathbf{U}\Sigma}-||\mathbf{A}||\mathbf{U}\Sigma\right|\right|_F\leq\sqrt{k}||\mathbf{A}||(\varepsilon+\delta),$$

$$\left|\left|||\mathbf{A}||\overline{\mathbf{V}\Sigma}-||\mathbf{A}||\mathbf{V}\Sigma\right|\right|_F\leq\sqrt{k}||\mathbf{A}||(\varepsilon+\delta).$$

□

4.4 Quantum Correspondence Analysis

Similarly to how we have provided a quantum procedure to speed-up Principal Component Analysis, we analyze a quantum procedure for Correspondence Analysis for data visualization - as presented in Section 2.3.2 - discussing its time complexity and providing an error bound on the computed model.

We assume that the input data is already classically available in the form of a contingency table $\mathbf{C}\in\mathbb{N}^{n\times m}$. All the operations required to compute

$$\mathbf{D}_{\mathscr{X}}^{1/2},\mathbf{D}_{\mathscr{Y}}^{1/2},\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}},\hat{\mathbf{p}}_{\mathscr{X}},\hat{\mathbf{p}}_{\mathscr{Y}}$$

and

$$\mathbf{A}=\mathbf{D}_{\mathscr{X}}^{1/2}(\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}}-\hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}_{\mathscr{Y}}^T)\mathbf{D}_{\mathscr{Y}}^{1/2}$$

can be performed in time $O(nm)$ and for this reason they can be executed classically.
The resulting matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ is stored in QRAM. Because matrix $\mathbf{A}$ is the result of arithmetics on matrices that express probabilities, it has some interesting properties that we state in the following claims.

Claim 3. *The entries of* $\mathbf{A}$ *are positive real numbers smaller than one. Moreover, the* ℓ_1 *norm of each of its rows and columns never exceeds one.*

Proof. We begin the proof by showing that the entries of $(\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}} - \hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}_{\mathscr{Y}}^T)$ and the ℓ_1 norms of its rows and columns are positive real numbers smaller than one.

Each entry p_{ij} of $\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}}$ represent the empiric probability $p(x_i, y_i)$ of seeing the outcomes x_i and y_i together and is therefore smaller than one. Moreover, the ℓ_1 norm of the i^{th} row vector is defined as

$$||\mathbf{p}_{i,\cdot}||_1 = \sum_{j}^{|\mathscr{Y}|} p_{ij}$$

which is the probability $p(x_i)$ of seeing the outcome x_i for the random variable $\mathscr{X}$. Similarly, the ℓ_1 norm of the j^{th} column vector is

$$||\mathbf{p}_{\cdot,j}||_1 = \sum_{i}^{|\mathscr{X}|} p_{ij}$$

which is the probability $p(y_j)$ of seeing the outcome $y_i j$ for the random variable $\mathscr{Y}$.

Each entry of the matrix $\hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}_{\mathscr{Y}}^T$ is equal to the joint probability $p(x_i, y_i) = p(x_i)p(y_i)$ of seeing the outcomes x_i and y_i together if they were independent. The ℓ_1 norms of its rows and columns are bounded by 1 for the same reasoning above. Since each entry of $\hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}_{\mathscr{Y}}^T$ is equal or smaller than the entries of $\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}}$, the matrix obtained by the difference of the these two matrices still has positive entries smaller than 1 and the ℓ_1 norm of its rows and columns stays smaller than 1.

To end the proof, it is sufficient to note that the entries of $D_{\mathscr{X}}^{1/2}$ and $D_{\mathscr{Y}}^{1/2}$ are square roots of probabilities, which are smaller than one, and that the scaling $\mathbf{D}_{\mathscr{X}}^{1/2}(\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}} - \hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}_{\mathscr{Y}}^T)\mathbf{D}_{\mathscr{Y}}^{1/2}$ cannot increase the value of the entries of $(\hat{\mathbf{P}}_{\mathscr{X},\mathscr{Y}} - \hat{\mathbf{p}}_{\mathscr{X}}\hat{\mathbf{p}}_{\mathscr{Y}}^T)$. Then, the properties are propagated to $\mathbf{A}$. □

Claim 4. *The maximum singular value of* $\mathbf{A}$ *is smaller than one.*

Proof. It is folklore knowledge that

$$||\mathbf{A}|| \leq \sqrt{||\mathbf{A}||_1 ||\mathbf{A}||_\infty} \leq 1$$

where

$$||\mathbf{A}||_1 = max_j(||\mathbf{a}_{\cdot,j}||) \text{ and } ||\mathbf{A}||_\infty = max_i(||\mathbf{a}_{i,\cdot}||).$$

Because of Claim 3, $||\mathbf{A}||_1$ and $||\mathbf{A}||_\infty$ are both bounded by 1.
It follows that $\sigma_{max} = ||\mathbf{A}|| \leq 1$. □

Claim 5. *The parameter $\mu(\mathbf{A})$ is negligible in all the Correspondence Analysis tasks.*

Proof. From [22] we know that

$$\mu(\mathbf{A}) \leq ||\mathbf{A}||_1$$

and from Claim 3 we know that $||\mathbf{A}||_1 \leq 1$. Therefore,

$$\mu(\mathbf{A}) \leq 1.$$

□

We can leverage the quantum routines of Theorems 13 and 15 to extract the factor scores $\{\sigma_i^2\}_i^k$, the factor score ratios $\{\frac{\sigma_i^2}{||\mathbf{A}||_F}\}_i^k$ and the orthogonal factors $\mathbf{L} = D_{\mathscr{X}}^{-1/2}\overline{\mathbf{U}}$ e $\mathbf{R} = D_{\mathscr{Y}}^{-1/2}\overline{\mathbf{V}}$. Just like in the PCA case, the factor $\frac{1}{\sqrt{p}}$, where p is the sum of the factor score ratios of the desired factor scores, is negligible as it is usually required to be greater than 0.40.
To extract the factor scores with precision ε and the factor score ratios with precision γ, it is possible to use Algorithm 1 and it takes time

$$\tilde{O}\left(\frac{1}{\varepsilon\gamma^2}\right)$$

To extract the left and right singular vectors it is possible to use Algorihtm 3 with precision δ and threshold θ for the lower singular value in time

$$\tilde{O}\left(\frac{1}{\theta}\frac{k(n+m)log(m+n+k)}{\delta^2}\right)$$

by modifying the algorithm so as to perform vector state tomography for the left and right singular values simultaneously. Moreover, when dealing with CA for data visualization, k is required to be equal to 2 or equal to 3, depending on the space ($\mathbb{R}^2$ or $\mathbb{R}^3$) where we want to represent the outputs of the categorical random variables $\mathscr{X}$ and $\mathscr{Y}$. For this reason, the complexity of this step can be considered to be:

$$\tilde{O}\left(\frac{1}{\theta}\frac{(n+m)log(m+n)}{\delta^2}\right).$$

The latter is certainly smaller than the cost of storing the matrix data $\mathbf{A}$ in QRAM, which is $O(nmlog^2(nm))$. The quantum procedures reduce the cost of CA to a linear cost on the number of elements of the contingency table $\mathbf{C} \in \mathbb{N}^{n\times m}$.

To conclude the analysis, we examine the error bound on the orthogonal factors $\mathbf{L}$ and $\mathbf{R}$ and we claim it to be $\sqrt{k}\delta$.

Lemma 3 (Accuracy of $\mathbf{L} = \mathbf{D}_{\mathscr{X}}^{1/2}\mathbf{U}$ and $\mathbf{R} = \mathbf{D}_{\mathscr{Y}}^{1/2}\mathbf{V}$)**.** *Given some estimates $\{\overline{\mathbf{u}}_i\}_i^k$ of the left singular vectors of matrix $\mathbf{A}$ such that $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$, it is possible to compute an estimate $\overline{\mathbf{L}} = \mathbf{D}_{\mathscr{X}}^{1/2}\overline{\mathbf{U}}$ of $\mathbf{L} = \mathbf{D}_{\mathscr{X}}^{1/2}\mathbf{U}$ such that $||\overline{\mathbf{L}} - \mathbf{L}||_F \leq \sqrt{k}\delta$. Similarly, $||\overline{\mathbf{R}} - \mathbf{R}||_F \leq \sqrt{k}\delta$.*

Proof. We proof the lemma for the $\left|\left|\overline{\mathbf{L}} - \mathbf{L}\right|\right|_F$ bound, by exploiting the fact that the entries $p_{\mathscr{X}i}$ of $\mathbf{D}_{\mathscr{X}}$ are smaller than one.

$$||\mathbf{D}_{\mathscr{X}}^{-1/2}\overline{\mathbf{U}} - \mathbf{D}_{\mathscr{X}}^{-1/2}\mathbf{U}||_F = \sqrt{\sum_i^n \sum_j^k (p_{\mathscr{X}i}^{1/2}\overline{u}_{ij} - p_{\mathscr{X}i}^{1/2}u_{ij})^2} =$$

$$\sqrt{\sum_i^n \sum_j^k (p_{\mathscr{X}i}^{1/2}\overline{u}_{ij} - p_{\mathscr{X}i}^{1/2}u_{ij})^2} = \sqrt{\sum_j^k \sum_i^n p_{\mathscr{X}i}(\overline{u}_{ij} - u_{ij})^2} \leq \sqrt{\sum_j^k \sum_i^n (\overline{u}_{ij} - u_{ij})^2} = \sqrt{k}\delta$$

In the same way we can prove $||\mathbf{D}_{\mathscr{Y}}^{-1/2}\overline{\mathbf{V}} - \mathbf{D}_{\mathscr{Y}}^{-1/2}\mathbf{V}||_F \leq \sqrt{k}\delta$ □

4.5 Quantum Latent Semantic Analysis

The last algorithm that we analyze is a quantum version of Latent Semantic Analysis. As discussed in Section 2.3.3, given the input data matrix $\mathbf{A} \in \mathbb{R}^{n\times m}$ of rank r, the data representation models for Latent Semantic Analysis are the following:

- $\mathbf{L} = \mathbf{U}^{(k)}\Sigma^{(k)}$
- $\mathbf{R} = \mathbf{V}^{(k)}\Sigma^{(k)}$
- $\mathbf{L}' = \mathbf{U}^{(k)}\Sigma^{(k)1/2}$
- $\mathbf{R}' = \mathbf{V}^{(k)}\Sigma^{(k)1/2}$.

In case we aim to perform Latent Semantic Indexing, we also need the matrix $\mathbf{Q} = \mathbf{U}^{(k)}\Sigma^{(k)-1}$ that will be used to prepare the query vectors for the documents.

The quantum-aided procedure to be performed consists in creating the matrix $\mathbf{A}$ using classical computation means, storing it in QRAM so that $\sigma_{max} \leq 1$, and running Algorithm 3 to extract the top-k singular values and singular vectors.

Differently from PCA and CA, in LSA one is not usually interested in retrieving the factor scores or the factor score ratios. For this reason, there is no need to run Algorithm 1. Instead, we can execute the modified version of Algorithm 3 sketched in Section 4.4 with threshold θ, precision ε for the singular values, and precision δ for the singular vectors.

In LSA, the number k of singular values and singular vectors to extract is determined empirically and it can be found in the literature. The original paper of Latent Semantic Indexing, for instance, suggests that $k = 100$ is a good number in several practical scenarios [11].

In the same way, we think that studies can be conducted to establish the threshold parameter θ, which we expect to be independent from the matrix size.

The complexity of the quantum steps of this algorithm is the one given by 15, modified to perform tomography both on the left and on the right singular vectors, and it is:

$$O\left(\left(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{k(n+m)log(n+m+k)}{\delta^2}\right)\mu(\mathbf{A})\frac{polylog(nm)}{\varepsilon}\right).$$

An interesting upper bound on $\mu(\mathbf{A})$, in the case of a matrix with spectral norm equal to one, is $\sqrt{r}$. Indeed, $\mu(\mathbf{A}) \leq ||\mathbf{A}||_F = \sqrt{\sum_i^r \sigma_i^2} \leq \sqrt{r}$.

In the case of low rank matrices, this complexity is significantly smaller than the cost of storing the input matrix in a QRAM and normalizing it, which is almost linear in the matrix size and is $O(nmlog^2(nm))$.

For what concerns the error bounds on the retrieved data representation models, we already know from Lemma 2 that it is possible to retrieve an approximation $\overline{\mathbf{L}}$ and $\overline{\mathbf{R}}$ such that $||\overline{\mathbf{L}} - \mathbf{L}||_F$ and $||\overline{\mathbf{R}} - \mathbf{R}||_F$ are smaller than $\sqrt{k}(\delta + \varepsilon)$, where δ is the precision on the singular vectors and ε the precision on the singular values. To provide bounds on the estimations of $\mathbf{L}', \mathbf{R}'$, and $\mathbf{Q}$ we present and prove Lemma 4 and Lemma 5.

Lemma 4 (Accuracy of $\mathbf{U}\Sigma^{1/2}$ and $\mathbf{V}\Sigma^{1/2}$)**.** *Let $\mathbf{A} \in \mathbb{R}^{n\times m}$ be a matrix of rank k with singular value decomposition $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$ such that $\sigma_{max} \leq 1$ and $\sigma_{min} \geq \theta$. Let $\varepsilon, \delta > 0$ be two precision parameters. Given some approximate procedures that allow to retrieve an estimate $\overline{\sigma}_i$ of the singular values σ_i such that $|\overline{\sigma}_i - \sigma_i| \leq \varepsilon$ and an estimate $\overline{\mathbf{u}}_i$ of the left singular*

vectors $\mathbf{u}_i$ *such that* $||\overline{\mathbf{u}}_i - \mathbf{u}_i|| \leq \delta$*, the error on the estimate* $\overline{\mathbf{U}\Sigma}^{1/2}$ *can be bounded as*

$$\left|\left|\overline{\mathbf{U}\Sigma}^{1/2} - \mathbf{U}\Sigma^{1/2}\right|\right|_F \leq \sqrt{k}\left(\delta + \frac{1}{2\sqrt{\theta}}\right).$$

Similarly, if the approximate procedures output the singular values and the right singular vectors, the error bound for $\overline{\mathbf{V}\Sigma}^{1/2}$ *is*

$$\left|\left|\overline{\mathbf{V}\Sigma}^{1/2} - \mathbf{V}\Sigma^{1/2}\right|\right|_F \leq \sqrt{k}\left(\delta + \frac{1}{2\sqrt{\theta}}\right).$$

Proof. We start the proof by bounding $\left|\sqrt{\overline{\sigma}_i} - \sqrt{\sigma_i}\right|$.

$$\left|\sqrt{\overline{\sigma}_i} - \sqrt{\sigma_i}\right| \leq \left|\sqrt{\sigma_i + \varepsilon} - \sqrt{\sigma_i}\right|$$

Let's define $\varepsilon = \gamma\sigma_i$ as a relative error:

$$\left|\sqrt{\sigma_i + \varepsilon} - \sqrt{\sigma_i}\right| = \left|\sqrt{\sigma_i + \gamma\sigma_i} - \sqrt{\sigma_i}\right| =$$

$$\left|\sqrt{\sigma_i(1+\gamma)} - \sqrt{\sigma_i}\right| = \left|\sqrt{\sigma_i}(\sqrt{1+\gamma} - 1)\right| =$$

$$\sqrt{\sigma_i}\left|\frac{(\sqrt{1+\gamma}-1)(\sqrt{1+\gamma}+1)}{\sqrt{1+\gamma}+1}\right| = \sqrt{\sigma_i}\left|\frac{\gamma+1-1}{\sqrt{1+\gamma}+1}\right| \leq \sqrt{\sigma_i}\frac{\gamma}{2}$$

Because of the definition of ε, we know that $\gamma = \frac{\varepsilon}{\sigma_i}$ and we can express the bound in ε again:

$$\left|\sqrt{\overline{\sigma}_i} - \sqrt{\sigma_i}\right| \leq \frac{\sqrt{\sigma_i}}{\sigma_1}\frac{\varepsilon}{2} = \frac{\varepsilon}{2\sqrt{\sigma_i}}$$

Furthermore, we can exploit the fact that $\sigma_{min} \geq \theta$ to relax the bound and make it uniform for each singular value:

$$\left|\sqrt{\overline{\sigma}_i} - \sqrt{\sigma_i}\right| \leq \frac{\varepsilon}{2\sqrt{\theta}}.$$

From the bound on the square root of the singular values, we can bound the columns of $\overline{\mathbf{U}\Sigma}^{1/2}$ in the following way:

$$\left|\left|\sqrt{\overline{\sigma}_i}\overline{\mathbf{u}}_i - \sqrt{\sigma_i}\mathbf{u}_i\right|\right| \leq \left|\left|\left(\sqrt{\sigma_i} + \frac{\varepsilon}{2\sqrt{\theta}}\right)\overline{\mathbf{u}}_i - \sqrt{\sigma_i}\mathbf{u}_i\right|\right| =$$

$$\left|\left|\sqrt{\sigma_i}(\overline{\mathbf{u}}_i - \mathbf{u}_i) + \frac{\varepsilon}{2\sqrt{\theta}}\overline{\mathbf{u}}_i\right|\right| \leq \sqrt{\sigma_i}\delta + \frac{\varepsilon}{2\sqrt{\theta}} \leq \delta + \frac{\varepsilon}{2\sqrt{\theta}}.$$

Finally, from the bound on the columns we derive the bound on the matrices:

$$\left|\left|\overline{\mathbf{U}\Sigma}^{1/2}-\mathbf{U}\Sigma^{1/2}\right|\right|_F=\sqrt{\sum_i^n\sum_j^k(\left|\sqrt{\overline{\sigma}_j}\overline{u}_{ij}-\sqrt{\sigma_j}u_{ij}\right|)^2}=\sqrt{\sum_j^k\left(\sqrt{\sum_i^n(\sqrt{\overline{\sigma}}_j\overline{u}_{ij}-\sqrt{\sigma_j}u_{ij})^2}\right)^2}=$$

$$\sqrt{\sum_j^k\left(\left|\left|\sqrt{\overline{\sigma}_j}\overline{\mathbf{u}}_j-\sqrt{\sigma_j}\mathbf{u}_j\right|\right|\right)^2}\leq\sqrt{\sum_j^k\left(\delta+\frac{\varepsilon}{2\sqrt{\theta}}\right)^2}=\sqrt{k\left(\delta+\frac{\varepsilon}{2\sqrt{\theta}}\right)^2}=\sqrt{k}\left(\delta+\frac{\varepsilon}{2\sqrt{\theta}}\right).$$

The proof for $\left|\left|\overline{\mathbf{V}\Sigma}^{1/2}-\mathbf{V}\Sigma^{1/2}\right|\right|_F\leq\sqrt{k}\left(\delta+\frac{1}{2\sqrt{\theta}}\right)$ is analogous. □

Lemma 5 (Accuracy of $\mathbf{U}\Sigma^{-1}$ and $\mathbf{V}\Sigma^{-1}$)**.** *Let $\mathbf{A}\in\mathbb{R}^{n\times m}$ be a matrix of rank k with singular value decomposition $\mathbf{A}=\mathbf{U}\Sigma\mathbf{V}^T$ such that $\sigma_{max}\leq 1$ and $\sigma_{min}\geq\theta$. Let $\varepsilon,\delta>0$ be two precision parameters. Given some approximate procedures that allow to retrieve an estimate $\overline{\sigma}_i$ of the singular values σ_i such that $|\overline{\sigma}_i-\sigma_i|\leq\varepsilon$ and an estimate $\overline{\mathbf{u}}_i$ of the left singular vectors $\mathbf{u}_i$ such that $||\overline{\mathbf{u}}_i-\mathbf{u}_i||\leq\delta$, the error on the estimate $\overline{\mathbf{U}\Sigma}^{-1}$ can be bounded as*

$$\left|\left|\overline{\mathbf{U}\Sigma}^{-1}-\mathbf{U}\Sigma^{-1}\right|\right|_F\leq\sqrt{k}(\frac{\delta}{\theta}+\frac{1}{\delta+\varepsilon}).$$

Similarly, if the approximate procedures output the singular values and the right singular vectors, the error bound for $\overline{\mathbf{V}\Sigma}^{-1}$ is

$$\left|\left|\overline{\mathbf{V}\Sigma}^{-1}-\mathbf{V}\Sigma^{-1}\right|\right|_F\leq\sqrt{k}(\frac{\delta}{\theta}+\frac{1}{\delta+\varepsilon}).$$

Proof. We prove the lemma for the $\left|\left|\overline{\mathbf{U}\Sigma}^{-1}-\mathbf{U}\Sigma^{-1}\right|\right|_F$ bound, the bound on $\left|\left|\overline{\mathbf{V}\Sigma}^{-1}-\mathbf{V}\Sigma^{-1}\right|\right|_F$ can be easily proved by repeating the same steps.

Similarly to the other proofs, we start by bounding $\left|\frac{1}{\overline{\sigma}_i-\frac{1}{\sigma_i}}\right|$.

$$\left|\frac{1}{\overline{\sigma}_i}-\frac{1}{\sigma_i}\right|\leq\left|\frac{1}{\sigma_i-\varepsilon}-\frac{1}{\sigma_i}\right|$$

Let's say $\varepsilon=\gamma\sigma_i$:

$$\left|\frac{1}{\sigma_i-\gamma\sigma_i}-\frac{1}{\sigma_i}\right|=\left|\frac{1}{\sigma_i(1-\gamma)}-\frac{1}{\sigma_i}\right|=\frac{1}{1-\gamma}$$

We know that $\gamma = \frac{\varepsilon}{\sigma_i}$ and that $\sigma_{max} \leq 1$, so the bound becomes

$$\frac{1}{1-\gamma} = \frac{\sigma_i}{\sigma_i - \varepsilon} \leq \frac{1}{\sigma_i - \varepsilon}$$

Finally, to avoid the dependency on the singular values, we can leverage the fact that $\sigma_{min} \geq \theta$:

$$\left| \frac{1}{\overline{\sigma}_i} - \frac{1}{\sigma_i} \right| \leq \frac{1}{\theta - \varepsilon}$$

From the bound on the inverse of the singular values, we can obtain the bound on the columns of the $\overline{\mathbf{V}\Sigma}^{-1}$ matrix:

$$\left|\left| \frac{1}{\overline{\sigma}_i}\overline{\mathbf{u}}_i - \frac{1}{\sigma_i}\mathbf{u}_i \right|\right| \leq \left|\left| \left(\frac{1}{\sigma_i} \pm \frac{1}{\delta+\varepsilon} \right)\overline{\mathbf{u}}_i - \frac{1}{\sigma_i}\mathbf{u}_i \right|\right| = \frac{1}{\sigma_i}\delta + \frac{1}{\delta+\varepsilon} \leq \frac{\delta}{\theta} + \frac{1}{\delta+\varepsilon}.$$

To end the proof, we can rewrite the bound on the matrices as:

$$\left|\left| \overline{\mathbf{U}\Sigma}^{-1} - \mathbf{U}\Sigma^{-1} \right|\right|_F = \sqrt{\sum_i^n \sum_j^k \left(\left| \frac{1}{\overline{\sigma}_j}\overline{u}_{ij} - \frac{1}{\sigma_j}u_{ij} \right| \right)^2} = \sqrt{\sum_j^k \left(\frac{1}{\overline{\sigma}_j}\overline{u}_{ij} - \frac{1}{\sigma_j}u_{ij} \right)^2} =$$

$$\sqrt{\sum_j^k \left(\left|\left| \frac{1}{\overline{\sigma}_j}\overline{\mathbf{u}}_j - \frac{1}{\sigma_j}\mathbf{u}_j \right|\right| \right)^2} \leq \sqrt{\sum_j^k \left(\frac{\delta}{\theta} + \frac{1}{\delta+\varepsilon} \right)^2} = \sqrt{k\left(\frac{\delta}{\theta} + \frac{1}{\delta+\varepsilon} \right)^2} = \sqrt{k}\left(\frac{\delta}{\theta} + \frac{1}{\delta+\varepsilon} \right)$$

□

To obtain the un-normalized representations of $\mathbf{L}$ and $\mathbf{R}$ for the un-normalized version of the input matrix $\mathbf{A}$, one can refer to Claim 2. Instead, for the un-normalized representation of $\mathbf{L}', \mathbf{R}'$, and $\mathbf{Q}$, we present and prove Claims 6 and 7.

Claim 6. *If* $\mathbf{A}$ *has been normalized before retrieving the singular values and singular vectors and one wants to retrieve the representations of Lemma 4 for the un-normalized matrix, the estimated matrices are* $\sqrt{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{1/2}$ *and* $\sqrt{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{1/2}$ *and the bounds become*

- $\left|\left| \sqrt{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{1/2} - \sqrt{||\mathbf{A}||}\mathbf{U}\Sigma^{1/2} \right|\right| \leq \sqrt{k||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}} \right)$
- $\left|\left| \sqrt{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{1/2} - \sqrt{||\mathbf{A}||}\mathbf{V}\Sigma^{1/2} \right|\right| \leq \sqrt{k||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}} \right)$

Proof. From the proof of Claim 1 we know that the singular values of the un-normalized matrix are the ones of the normalized matrix, scaled by a factor $\frac{1}{||\mathbf{A}||}$, while the singular vectors remain the same.

Since the entries of the un-normalized $\Sigma^{1/2}$ matrix are $\{\sqrt{||\mathbf{A}||\sigma_i}\}_i^k$, it is easy to see that the desired representations are $\sqrt{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{1/2}$ and $\sqrt{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{1/2}$.

To prove the bounds, it is sufficient to multiply both the inequalities of Lemma 4 by $\sqrt{||\mathbf{A}||}$:

$$\left|\left|\sqrt{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{1/2} - \sqrt{||\mathbf{A}||}\mathbf{U}\Sigma^{1/2}\right|\right| \leq \sqrt{k||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}}\right)$$

$$\left|\left|\sqrt{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{1/2} - \sqrt{||\mathbf{A}||}\mathbf{V}\Sigma^{1/2}\right|\right| \leq \sqrt{k||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}}\right)$$

□

Claim 7. *If* $\mathbf{A}$ *has been normalized before retrieving the singular values and singular vectors and one wants to retrieve the representations of Lemma 5 for the un-normalized matrix, the estimated matrices are* $\frac{1}{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{-1}$*, and* $\frac{1}{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{-1}$ *and the bounds become*

- $\left|\left|\frac{1}{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{-1} - \frac{1}{||\mathbf{A}||}\mathbf{U}\Sigma^{1/2}\right|\right| \leq \frac{\sqrt{k}}{||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}}\right)$
- $\left|\left|\frac{1}{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{-1} - \frac{1}{||\mathbf{A}||}\mathbf{V}\Sigma^{1/2}\right|\right| \leq \frac{\sqrt{k}}{||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}}\right)$

Proof. From the proof of Claim 1 we know that the singular values of the un-normalized matrix are the ones of the normalized matrix, scaled by a factor $\frac{1}{||\mathbf{A}||}$, while the singular vectors remain the same.

Since the entries of the un-normalized Σ^{-1} matrix are $\{\frac{1}{||\mathbf{A}||\sigma_i}\}_i^k$, it is easy to see that the desired representations are $\frac{1}{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{1/2}$ and $\frac{1}{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{1/2}$.

To prove the bounds, it is sufficient to multiply both the inequalities of Lemma 5 by $\frac{1}{||\mathbf{A}||}$:

$$\left|\left|\frac{1}{||\mathbf{A}||}\overline{\mathbf{U}\Sigma}^{-1} - \frac{1}{||\mathbf{A}||}\mathbf{U}\Sigma^{1/2}\right|\right| \leq \frac{\sqrt{k}}{||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}}\right)$$

$$\left|\left|\frac{1}{||\mathbf{A}||}\overline{\mathbf{V}\Sigma}^{-1} - \frac{1}{||\mathbf{A}||}\mathbf{V}\Sigma^{1/2}\right|\right| \leq \frac{\sqrt{k}}{||\mathbf{A}||}\left(\delta + \frac{1}{2\sqrt{\theta}}\right)$$

□

Chapter 5

Analysis and experiments

5.1 Simulation settings

The experiments have been conducted using Python. We report the code to allow the reader reproducing them.

5.1.1 Simulating a quantum state

In order to simulate the measurements of a quantum state, we have implemented a python class called QuantumState. This class simulates a basic quantum register

$$|\mathbf{x}\rangle = \frac{1}{||\mathbf{x}||}\sum_{i}^{n} x_i |i\rangle .$$

The object is initialized with a list of values that the register can assume and a list of amplitudes, one for each register. In case the amplitudes are not normalized, they get normalized when the object is initialized.

The quantum register is considered to be of arbitrary precision: we do not consider how many qubits are needed to represent a number in its binary expansion as this is not of interest for our experiments. The probability associated to each value of the registers is the square of their amplitudes, as discussed in Section 3.1.5.

The object QuantumState offers a method *get_state()* that returns a dictionary $\{register : probability\}$ and a method *measure()* that outputs one of the possible values of the registers, according to their probabilities. It is possible to call the method *measure()* with a parameter that specifies the number of measurements to be performed.

```
import random
import math
class QuantumState(object):
        """This class simulates a Quantum State"""
        def __init__(self, registers, amplitudes):
                super(QuantumState, self).__init__()
                self.registers = registers

                #Amplitudes must be normalized to have the right
                 ↪  probabilities for each register
                self.norm_factor = math.sqrt(sum([pow(x, 2) for x in
                 ↪  amplitudes]))
                self.amplitudes = [x/self.norm_factor for x in
                 ↪  amplitudes]

                #Each register_i appears with probability
                 ↪  amplitude_i^2
                self.probabilities = [pow(x, 2) for x in
                 ↪  self.amplitudes]
                assert (len(self.registers) == len(self.amplitudes))
                assert (abs(sum(self.probabilities) - 1) <
                 ↪  0.0000000001)

        def measure(self, n_times=1):
                return random.choices(self.registers,
                 ↪  weights=self.probabilities, k=n_times)

        def get_state(self):
                return {self.registers[i]:self.probabilities[i] for i
                 ↪  in range(len(self.registers))}
```

We also provide a method that estimates the probabilities of each value of a quantum register, given a list of measurements. The estimator is the usual frequentist estimator, discussed in the subsection of Section 3.1.5 about the Wald confidence interval.

```
from collections import Counter

def estimate_probabilities(measurements):
        counter=Counter(measurements)
        estimate={x:counter[x]/len(measurements) for x in counter}
        return estimate
```

5.1.2 Simulating the errors

To simulate the errors given by quantum measurements and discussed in the Theorems and Lemmas of Chapter 4, we perturb the original values by adding errors sampled from truncated Gaussian of zero mean and unit variance, truncated on the interval $[-error, error]$.

When simulating perturbations on the ℓ_2 norm of a vector, or on the Frobenius norm of a matrix, we spread the error equally to each component.

The implementation of the methods is the following.

```
from scipy.stats import truncnorm

#Given a scalar it makes it noisy by adding a truncated Gaussian error
def introduce_error(value, epsilon):
    return value + truncnorm.rvs(-epsilon,epsilon, size=1)

#Given a vector it makes it noisy by adding a truncated Gaussian error
 ↪  to each component
def make_noisy_vec(vec, noise):
    noise_per_component = noise/np.sqrt(len(vec))
    errors = truncnorm.rvs(-noise_per_component,noise_per_component,
     ↪  size=len(vec))
    new_vec = np.array([vec[i] + errors[i] for i in range(len(vec))])
    return new_vec

#Given a matrix it makes it noisy by adding a truncated Gaussian error
 ↪  to each component
def make_noisy_mat(A, noise):
    vector_A = A.reshape(A.shape[0]*A.shape[1])
    vector_B = make_noisy_vec(vector_A, noise)
    B = vector_B.reshape(A.shape[0],A.shape[1])
    return B
```

The graph in Figure 5.1 is the result of error simulations on a matrix and shows the distribution of the *samples* variable from the code below. The histograms are plotted in bins of 50 samples.

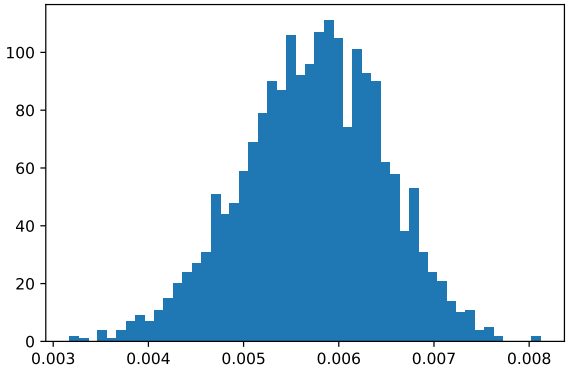


Fig. 5.1 Error simulation: generating a matrix $\mathbf{B} \in \mathbb{R}^{n \times m}$ that approximates a matrix $\mathbf{A} \in \mathbb{R}^{n \times m}$ such that $||\mathbf{A} - \mathbf{B}||_F \leq 0.01$ for 2000 times and plotting $||\mathbf{A} - \mathbf{B}||_F$. The error is introduced on each component using a truncated Gaussian with zero mean and unit variance, truncated for $[-\frac{\varepsilon}{\sqrt{nm}}, \frac{\varepsilon}{\sqrt{nm}}]$.

```
#Create matrix A
A = np.array([[3,4,2],
      [5,6,6],
      [9,4,7],
      [1,0,8]])
#Specify the max error on the Frobenius norm
noise = 0.01
samples = []
for i in range(2000):
    #Create a noisy copy of A
    B = make_noisy_mat(A, noise)
    #Append the Frobenius norm of A-B to the samples
    samples.append(np.linalg.norm(A-B))
```

The distribution of the error on matrices and vectors is still Gaussian, centered almost at the half of the specified error.

5.2 Singular values distribution in real data

Throughout the work, we have relied on the assumption that in real datasets for machine learning the distribution of the singular values is so that a few singular values are much bigger than the others.

To support this assumption we have selected three datasets and investigated the distribution of the factor score ratios $\frac{\sigma_i^2}{\sum_j^r \sigma_j^2}$ in all of them. The three datasets that we have selected are the famous CIFAR-10 and MNIST 784 datasets for image classification and the Research Paper dataset for text classification.

We briefly describe the three datasets and the preprocessing performed and show the distribution of the factor score ratios in Figure 5.2.

MNIST 784

The MNIST 784 dataset is probably the most used dataset in image classification. It is a collection of 70000 images composed of $28 \times 28 = 784$ pixels. Each image represents a black and white hand-written digit between 0 and 9. Each image is paired with a label that specifies the number represented in the image. Since the images are black and white, we need an array of only 784 values to represent an image. The dataset, excluding the labels, can be encoded in a matrix of size 70000×784. In this experiment we do not run any preprocessing on it, we compute the singular values and the factor score ratios on the original dataset.

CIFAR-10

The CIFAR-10 dataset is another widely used dataset in image classification. It contains 60000 colored images of 32×32 pixel, with the values for each of the 3 RGB colors. Each image represents an object of these 10 classes {airplane, automobile, bird, cat, deer, dog, frog, horse, ship, truck} and the dataset contains the information about the labels. We use all the images, reshaping the dataset to be of size 60000×3072 and we do not run any preprocessing on it.

Research Paper

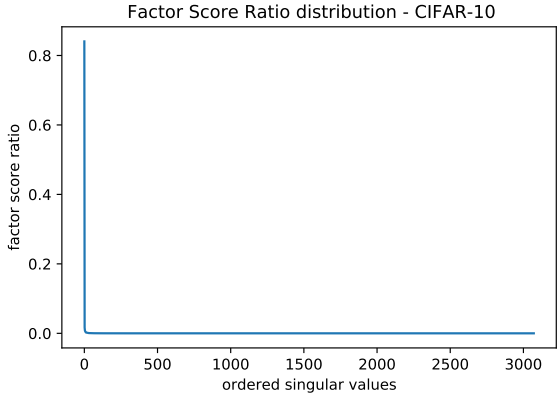
The Research Paper dataset is a dataset for text classification, released by Harun-Ur-Rashid, and available on Kaggle. The dataset is quite simple, it contains 2507 titles of papers together with the labels of the conferences where they have been published. The 5 possible labels are WWW, INFOCOM, ISCAS, SIGGRAPH, VLDB. We preprocess the titles

of the papers, to compute the contingency table, using the class *CountVectorizer* from *sklearn.feature_extraction.text* with the following parameters:

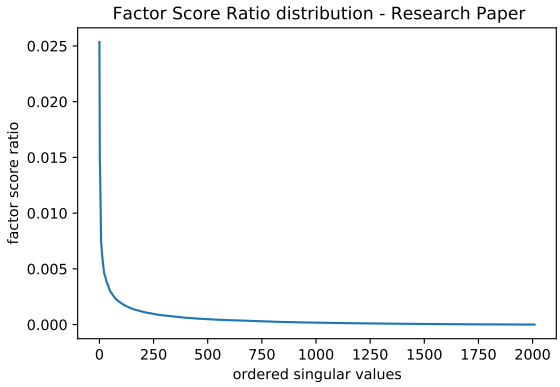
- max_df=0.5
- min_df=2
- stop_words='english'.

These parameters makes the preprocessing remove the English stop-words, the words that appear in only one document, and the words that appear in more than half the documents. The result is a contingency table of size 2507×2010.

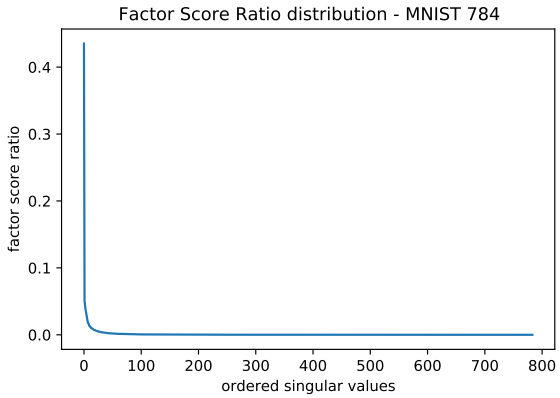
For all the three datasets, we compute the singular values and their factor score ratios and we plot them.



(a) CIFAR-10.



(b) Research Paper.



(c) MNIST 784.

Fig. 5.2 Factor score ratios of singular values in data for PCA and LSA.

As expected, the results of Figure 5.2 show an exponential decrease of the factor score ratios magnitudes in all the three datasets. It can be seen that the decrease in the singular value importance is much steeper in the two datasets containing images. However, even in the text classification data, the trend of the factor score ratios is decreasing exponentially.

5.3 MNIST classification with qPCA

To provide the reader with a clearer view of the algorithms described in Chapter 4 and their use in machine learning, we provide an experiment for the quantum PCA procedure introduced in Section 4.3. The experiment consists in reproducing the classical task of performing PCA on the MNIST dataset using the quantum procedures and then perform classification using a K-Nearest Neighbors (KNN) classifier.

We first reproduce the extraction of the singular values and the percentage of explained variance (factor score ratios) using the procedure from Theorem 13. Then, we simulate the error of Lemma 2 by introducing errors on the Frobenius norm of the correct PCA model to see how this error affects the classification accuracy.

Finally, we discuss the run-time parameters of qPCA with Theorems 13 and 15 using the data extracted from the first two experiment.

5.3.1 Number of Principal Components

The first step of the quantum algorithms consists in sampling the singular values and the amount of variance that they explain, using Theorem 13, so to decide the number of principal components to keep.

PCA assumes that the input matrix has already zero mean and that the matrix has been scaled so that the maximum singular value is less or equal than 1. The matrix *X_train_pca_scaled* is the matrix that we would like to have quantum access to. We refer to it, as usual, with $\mathbf{A}$, it has rank r and singular value decomposition $\mathbf{A} = \mathbf{U}\Sigma\mathbf{V}^T$.

We start the experiment by fixing a random seed so that it can be reproduced, we remove the mean from the input matrix, compute PCA with the maximum number of components $r = min(n,m) = min(70000, 784)$ and divide the matrix by the spectral norm, which is the largest singular value.

```
from sklearn import decomposition
import numpy as np

#Set a random seed to reproduce the experiment
random_seed = 1234
random.seed(random_seed)

#Retrieve MNIST Dataset
X, y = fetch_openml('mnist_784', version=1, return_X_y=True)

#Remove mean before applying PCA
X_mean0 = X - np.mean(X)
#Fit PCA
pca = decomposition.PCA()
pca.n_components = 784
pca_model = pca.fit(X_mean0)

#Transform the features
X_train_pca = pca_model.transform(X_mean0)
#Scale the matrix with the spectral norm
spectral_norm = pca_model.singular_values_[0]
X_train_pca_scaled = X_train_pca/spectral_norm
#Scale the singular values
pca_model.singular_values_ = pca_model.singular_values_/spectral_norm
```

Once all the singular values are computed and scaled in the interval $[0,1]$, we initialize a QuantumState object using the singular values both for the registers and the amplitudes

$$\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i |\sigma_i\rangle$$

to simulate the measurements on the third register of the quantum state at step 4 of Algorithm 1:

$$\frac{1}{\sqrt{\sum_j^r \sigma_j^2}} \sum_i^r \sigma_i |\mathbf{u}\rangle |\mathbf{v}\rangle |\overline{\sigma}_i\rangle .$$

We do not introduce error in the values of the register of the quantum state as in this step of the experiment we are interested in measuring the percentage of explained variance (factor score ratio) $\frac{\sigma_i^2}{\sum_j^r \sigma_j^2}$ of each singular value. We will deal with the error of the singular values at classification time, in Section 5.3.2.

After initializing the QuantumState object, we measure it $\frac{1}{\gamma^2} = 1000$ times and we estimate the factor score ratios. Measuring 1000 times guarantees us an error on each factor score ratio that is, at most, $\gamma = 0.03$. Finally, we sum the factor score ratios until the percentage of explained variance becomes greater than 85%.

```
q_state = QuantumState(registers=pca_model.singular_values_,
 ↪  amplitudes=pca_model.singular_values_)
estimations = estimate_probabilities(q_state.measure(1000))
exp_var = 0
i = 0
sv = sorted(estimations.keys())
while exp_var <= 0.85:
    exp_var += estimations[sv[-i]]
    i += 1
```

At the end of this code, the variable *i* contains the value 62 and the variable *exp_var* contains the value 85.10%. If we try the same calculations using the correct PCA model, we see that the top 61 singular values actually explain the 85.80% of the total variance and that the minimum number of principal components required to explain more than 85% is actually 59.

The number of principal components to use can be further refined using Algorithm 2 from Theorem 14. However, we believe that 62 is a perfectly acceptable number and we carry on the analysis using this number of principal components.

5.3.2 Classification error

Once that the number of principal components has been set to 62, the next step is to use Theorem 15 to extract the top-62 left singular vectors. To do so, we can retrieve the threshold θ from the previous step by checking the value of the 62^{th} smallest singular value, which is $\sigma_{61} = 0.1564$.

The next step of the experiment consists in checking how much of the error can be tolerated in the approximation of the PCA model $\overline{\mathbf{U}}^{(k=62)}\overline{\Sigma}^{(k=62)}$. We train again the PCA model specifying *pca.n_components = 62* and use the results from Lemma 2 to simulate the error $\sqrt{62}(\varepsilon+\delta)$, where ε is the error on the singular values and δ on the singular vectors.

We first compute the 10-fold Crossvalidation error using a KNN with $k=7$ on the correct model and then try the same classifier on several noisy PCA models.

```
knn = KNeighborsClassifier(n_neighbors=7)
score =
 ↪  cross_validate(knn,X_train_pca_scaled,y,cv=StratifiedKFold(n_splits=10,
 ↪  shuffle=True, random_state=random_seed))
expectation = np.average(score['test_score'])

experiments = []
k_sqrt = np.sqrt(pca.n_components)
errors= [k_sqrt*error for error in [0.8, 0.6, 0.4, 0.2, 0.1, 0.08,
 ↪  0.06, 0.04, 0.02, 0.01, 0.008, 0.006, 0.004, 0.002, 0.0008]]
for error in errors:
    B = make_noisy_mat(X_train_pca_scaled, error)
    f_norm = np.linalg.norm(X_train_pca_scaled-B)
    score =
     ↪  cross_validate(knn,B,y,cv=StratifiedKFold(n_splits=10,shuffle=True,
     ↪  random_state=random_seed))
    experiments.append([error, f_norm,
     ↪  np.average(score['test_score'])])
```

The results of the classifications are shown in Figure 5.3 and reported in Table 5.1.

The experiment shows that the error $(\varepsilon+\delta)$ can be around 0.1 without particular consequences on the classification accuracy. This is probably due to the fact that the bound on the Frobenius norm ensures that the geometry of the data points is preserved, in the same way as the ℓ_2 guarantees on a vector preserves the direction of the vector (see Figure 4.1).

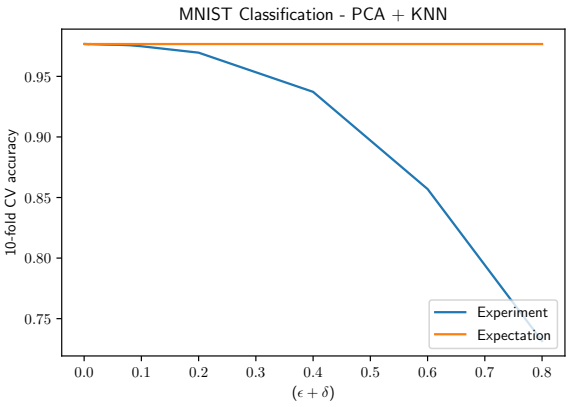


Fig. 5.3 MNIST classification with KNN as the error $\varepsilon + \delta$ on the PCA model (with k=62 PC) increases.

$(\varepsilon + \delta)$	$\sqrt{61}(\varepsilon + \delta)$	$\|\|\overline{\mathbf{U}}^{(61)}\overline{\Sigma}^{(61)} - \mathbf{U}^{(61)}\Sigma^{(61)}\|\|_F$	Accuracy
0.0	0.0	0.0	0.9767
0.0008	0.0062	0.0036	0.9767
0.002	0.0157	0.0090	0.9767
0.004	0.0314	0.0181	0.9767
0.006	0.0472	0.0272	0.9765
0.008	0.0629	0.0363	0.9764
0.01	0.0787	0.0454	0.9766
0.02	0.1574	0.0909	0.9764
0.04	0.3149	0.1819	0.9761
0.06	0.4724	0.2726	0.9760
0.08	0.6299	0.3636	0.9756
0.1	0.7874	0.4546	0.9748
0.2	1.5748	0.9094	0.9695
0.4	3.1496	1.8183	0.9372
0.6	4.7244	2.7289	0.8569
0.8	6.2992	3.6364	0.7314

Table 5.1 Classification accuracy using KNN with k=7 as the error on the MNIST PCA representation with 62 Principal Components increases.

5.3.3 Run-time parameters

As discussed in Section 4.3, the detailed run-time of this algorithm is

$$O\left(\left(\frac{1}{\gamma^2}+\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{knlog(n+k)}{\delta^2}\right)\mu(\mathbf{A})\frac{polylog(nm)}{\varepsilon}\right).$$

In this section, we discuss these precision parameters for the MNIST dataset. The main aim of this section is to show how to determine such parameters for a specific dataset. Indeed, the MNIST dataset is rather a simple dataset, having dimensions 70000×784 and the quantum procedures are not expected to show their speed-ups on a dataset of this size.

We show that Theorems 13 and 14 already give a polynomial advantage over classical methods on the MNIST dataset, scaling linearly on its dimensions, while Theorem 7 requires bigger dataset to show significant speed-ups.

When selecting the number of principal components, we have used an error of about $\gamma = 0.03$ on the percentage of explained variance of each singular value, performing $\frac{1}{\gamma} = 1000$ measurements of the quantum state. The results with this parameter are satisfactory as they lead to the selection of 62 principal components instead of 59.

It is worth it to recall that the parameter γ is not dependent on the size of the initial dataset, but only on how many principal components we are willing to accept. If we think that the number of relevant singular values is lower than a hundred, we expect that a thousand measurements can provide a good approximation.

Since the first 62 singular values explain the 85.80% of the total variance, the parameter $p = 0.8580$ and $\frac{1}{\sqrt{p}} = 1.0810$, which is negligible as for assumption.

The parameter θ, as already discussed, can be chosen to be equal to the smallest singular value, which is $\sigma_{61} = 0.1564$. With this value, we have that $\frac{1}{\theta} = 6.3938$. We also expect this number to remain fairly constant as the size of the dataset increases. Kerenidis and Luongo, in [22], show that increasing the dimensionality of the MNIST dataset by performing polynomial expansion on the features maintain this number almost constant.

To compute the parameter ε we have considered two requirements:

- ε has to be small enough for the threshold to cut out the undesired components;
- ε has to be small enough to preserve the ordering between the Principal Components, in case we wanted to change the model in the future.
- ε has to be small enough to ensure that the Coupon Collector's of Theorem 15 is satisfied.

In order to satisfy both the requirements, we have investigated the minimum difference between each pair of the top 62 singular values plus the first of the excluded ones. The code that we have run is pretty straightforward

```
epsilon = 1
index = 0
for i in range(0, 62):
    if pca_model.singular_values_[i] - pca_model.singular_values_[i+1]
     ↪ < epsilon:
        epsilon = pca_model.singular_values_[i] -
         ↪ pca_model.singular_values_[i+1]
        index = i
```

and it shows that the smallest difference is among the two singular values σ_{55}, σ_{56} and it is slightly greater than 0.0003.

At the same time, we have run experiments to check how the Coupon Collector's problem changes as ε increases. Recall that in Theorem 4.2 we state that

$$\frac{1}{\sqrt{\sum_i^k \frac{\sigma_i^2}{\overline{\sigma}_i^2}}} \sum_i^k \frac{\sigma_i}{\overline{\sigma}_i} |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle \sim \frac{1}{\sqrt{k}} \sum_i^k |\mathbf{u}_i\rangle |\mathbf{v}_i\rangle |\overline{\sigma}_i\rangle .$$

This is true only if ε is small enough to let the singular values distribute uniformly.

We check how the Singular Value Collection changes for several values of ε. For each error value that we test, we initialize a QuantumState object and we run the coupon collection problem a thousand times, taking the average number of required measurements as the result of the experiment.

```
def coupon_collect(quantum_state):
    counter = 0
    collection_dict = {value:0 for value in
     ↪ quantum_state.get_state().keys()}
    # Until you don't collect all the values, keep sampling and
     ↪ increment the counter
    while sum(collection_dict.values()) != len(collection_dict):
```

```
        value = quantum_state.measure()[0]
        if not collection_dict[value]:
            collection_dict[value] = 1
        counter += 1
    return counter

errors = [0.002, 0.004, 0.006, 0.008, 0.01, 0.02, 0.03, 0.04, 0.05,
 ↪  0.06, 0.07, 0.08, 0.09, 0.1]
n_trials = 1000

n = i

experiment_points = []
for e in errors:
    sve_singular_values = [introduce_error(singular_value, e) for
     ↪  singular_value in pca_model.singular_values_[:n]]
    amplitudes = [pca_model.singular_values_[i]/sve_singular_values[i]
     ↪  for i in range(0,n)]
    registers = [index for index in range(0,len(amplitudes))]

    quantum_state = QuantumState(registers,amplitudes)
    experiment_points.append(np.average([coupon_collect(quantum_state)
     ↪  for x in range(n_trials)]))

expected_points = [n*math.log(n,2.4) for i in range(len(errors))]
```

We have plotted the results of this code snippet in Figure 5.4 and reported the exact results in Table 5.2.

The experiments show that the Coupon Collector's argument is satisfied for $\varepsilon \leq 0.1$. Matching this result with the first two constraints, $\varepsilon = 0.003$ is a good choice.

From the experiments of Section 5.3.2, we know that $\varepsilon + \delta = 0.1$ and, since we do not have any other constraint on the precision δ, we can set it to $\delta = 0.1 - \varepsilon = 0.1 - 0.0003 = 0.0997$.

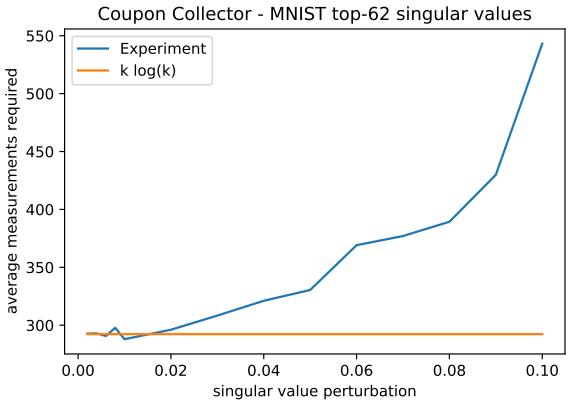


Fig. 5.4 MNIST SVE error for Coupon Collector.

ε	Average measurements
0.002	292.692
0.004	292.968
0.006	290.701
0.008	297.754
0.01	287.921
0.02	296.078
0.03	308.316
0.04	321.092
0.05	330.395
0.06	369.096
0.07	377.018
0.08	389.321
0.08	429.910
0.1	543.075

Table 5.2 Average measurements required to satisfy the Coupon Collector's argument as the error ε of the SVE increases.

To bound the parameter $\mu(\mathbf{A})$ favourably, we exploit the result of [25, Theorem 4.6]:

$$\mu(\mathbf{A}) \leq ||\mathbf{A}||_{\infty} = max_i(\sum_j^m |a_{ij}|) = 0.0726.$$

The number has been calculated by computing the maximum ℓ_1-norm on the rows of the PCA representation of $\mathbf{A}$ (obtained by running PCA with *pca.n_components* = *62*) via the following code snippet:

```
index = 0
l_infty = 0
for i in range(len(X_train_pca_scaled)):
    l1_norm = np.sum([abs(x) for x in X_train_pca_scaled[i]])
    if  l1_norm > l_infty:
        l_infty = l1_norm
        index = i
```

Finally, we want to consider also the running time of the optional algorithm from Theorem 14. This algorithm introduces the η error parameter, which is used to estimate the explained variance (factor score ratios sum) p with relative precision p. Since one percentage point is not a huge difference in PCA for data classification, we set $\eta = 0.01$.

We summarize all the run-time parameters in Table 5.3 and discuss the time complexity of the algorithms from Theorems 13, 14 and 15.

Parameter	Value	Run-time term	Value
γ	0.03	$\frac{1}{\gamma^2}$	1000
p	0.8580	$\frac{1}{\sqrt{p}}$	1.0795
θ	0.1564	$\frac{1}{\theta}$	6.3938
ε	0.0003	$\frac{1}{\varepsilon}$	3333.3333
δ	0.0997	$\frac{1}{\delta^2}$	100.6027
$\mu(\mathbf{A})$	0.0726	$\mu(\mathbf{A})$	0.0726
η	0.01	η	100

Table 5.3 Run-Time parameters for quantum PCA on MNIST.

To have a reference value for the time complexity, we give an idea of the time complexity of reading all the elements of the matrix once $O(nm)$ and of computing the SVD of a matrix

using its covariance matrix $O(n^2m, m^2n)$: [34][37][22]:

$$n \cdot m = 70000 \cdot 784 = 54880000 \simeq 5.4 \cdot 10^7$$

$$min(n^2 \cdot m, m^2 \cdot n) = 784 \cdot 784 \cdot 70000 = 43025920000 \simeq 4.2 \cdot 10^{10}$$

The algorithm from Theorem 13 runs in time:

$$O\left(\frac{\mu(\mathbf{A})}{\gamma^2\varepsilon} polylog(nm)\right).$$

Assuming that we have quantum access to the matrix $\mathbf{A}$ in time $log_2(nm)$, the exact time complexity is:

$$\frac{\mu(\mathbf{A})}{\gamma^2\varepsilon} log_2(nm) = 0.0726 \cdot 1000 \cdot 3333.3333 \cdot log_2(70000 \cdot 784) = 6221766.0070 \simeq 6.2 \cdot 10^6.$$

The algorithm from Theorem 14 runs in time:

$$O\left(\frac{\mu(\mathbf{A})}{\varepsilon}\frac{1}{\eta\sqrt{p}} polylog(nm)\right).$$

Assuming that we have quantum access to the matrix $\mathbf{A}$ in time $log_2(nm)$, the exact time complexity is:

$$\frac{\mu(\mathbf{A})}{\varepsilon}\frac{1}{\eta\sqrt{p}} log_2(nm) = 0.0726 \cdot 3333.3333 \cdot 100 \cdot 1.0795 \cdot log_2(70000 \cdot 784) =$$

$$671639.6404 \simeq 6.7 \cdot 10^5$$

The algorithm from Theorem 15 runs in time:

$$O\left(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{knlog(n+k)}{\delta^2} polylog(nm)\right).$$

Assuming that we have quantum access to the matrix $\mathbf{A}$ in time $log_2(nm)$, the exact time complexity is:

$$\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{knlog_2(n+k)}{\delta^2} log_2(nm) =$$

$$6.3938 \cdot 1.0795 \cdot 0.0726 \cdot 3333.3333 \cdot 100.6027 \cdot 62 \cdot 70000 \cdot log_2(70000+62) \cdot log_2(70000 \cdot 784) =$$

$$301801916741032.94 \simeq 3.0 \cdot 10^{14}.$$

These results show that Theorem 13 and 14 can already provide speed-ups on datasets as small as the MNIST dataset.

Even though their speed-up is not exponential, they still run sub-linearly on the number of elements of the matrix even though all the elements are taken into account during the computation, offering a polynomial speed-up with respect to their traditional classical counterparts. On big low-rank datasets that maintain a good distribution of singular values and $\mu()$ parameter, these algorithms are expected to show their exponential speed-up.

On the other hand, the algorithm from Theorem 15 performs worse than its traditional classical counterpart. We expect this Algorithm to show its speed-up asymptotically, with much larger datasets.

To this purpose, we leave as future work the study of these run-time parameters as the dataset's dimensions increase, following the experiment shown in this section and the experiments performed in [22].

Chapter 6

Limitations and future works

Even though our novel quantum procedures already scale well on the precision parameters for the data representations, we suspect that it is possible to exploit more advanced techniques based on qubitization to further reduce these dependencies. In particular, we intend to explore the recent frameworks of block encodings and singular value transformations [3] [16] to further improve the dependencies on the ε parameter.

The biggest limitation of this work is probably the fact that the algorithm from Theorem 15 gives a polynomial speed-up over the traditional SVD method rather than an exponential one.

It would be interesting to explore how the run-times of the theorems that we have provided would change if we use the novel ℓ_∞ vector state tomography procedure presented in [21]. This tomography allows to retrieve an approximation $\overline{\mathbf{x}} \in \mathbb{R}^n$ of a vector state $|\mathbf{x}\rangle$ such that $||\overline{\mathbf{x}} - \mathbf{x}||_\infty \leq \delta$ with probability $1 - 1/poly(n)$ in time $O\left(T(U_x)\frac{log(n)}{\delta^2}\right)$. Where $||\mathbf{x}||_\infty = max_i(|x_i|)$.

On the one hand, this algorithm provides an exponential advantage over the ℓ_2 tomography of Theorem 12. On the other hand, it also gives weaker approximation guarantees. We provide a formulation of Theorem 15 based on the ℓ_∞ tomography.

Theorem 16 (Top-k singular vectors extraction with ℓ_∞ guarantees)**.** *Let $\mathbf{A} \in \mathbb{R}^{n\times m}$ be a matrix with singular value decomposition $\mathbf{A} = \sum_i^r \sigma_i \mathbf{u}_i \mathbf{v}_i$ stored in QRAM with $|\sigma_{max}| \leq 1$. Let $\delta > 0$ be a precision parameter, ε the precision parameter for the singular values, and θ be a threshold such that $\mathbf{A}$ has k singular values greater than θ. Define $p = \frac{\sum_j^k \sigma_j^2}{\sum_i^r \sigma_i^2}$. Then, there exists a quantum algorithm that estimates*

- *The top k left singular vectors $\{\mathbf{u}_i\}_i^k$ of $\mathbf{A}$ with unit vectors $\{\overline{\mathbf{u}_i}\}_i^k$ such that $||\overline{\mathbf{u}}_i - \mathbf{u}_i||_\infty \leq \delta$ with probability at least $(1 - 1/poly(n))$, in time $\tilde{O}(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{k\log(n+k)}{\delta^2})$.*

- *The top k right singular vectors* $\{\mathbf{v}_i\}_i^k$ *of* $\mathbf{A}$ *with unit vectors* $\{\overline{\mathbf{v}_i}\}_i^k$ *such that* $||\overline{\mathbf{v}}_i - \mathbf{v}_i||_\infty \leq \delta$ *with probability at least* $(1 - 1/poly(m))$*, in time* $\tilde{O}(\frac{1}{\theta}\frac{1}{\sqrt{p}}\frac{\mu(\mathbf{A})}{\varepsilon}\frac{k\log(m+k)}{\delta^2})$.

Proof. The proof strictly follows the proof of Theorem 15, with the only difference that, in the last step, we perform the ℓ_∞ tomography from [21] instead of the vector state tomography from Theorem 12. □

Such tomography does not provide the same error guarantees of the ℓ_2 tomography and can lead to bad performances in practice. We leave the computation of the bounds for PCA, CA, and LSA and the experimental validation on the accuracy of the retrieved models for future works.

As a final remark, it is interesting to note that, using the ℓ_∞ tomography, it is possible to provide a different formulation of Theorem 13 that gives a simultaneous confidence interval on the error of the square roots of the factor score ratios.

Chapter 7

Conclusions

Contributions In this work, we have introduced three novel quantum procedures that extend the theory of quantum singular value estimation to achieve classical access to the top-k singular vectors and singular values of huge matrices. The algorithms that we propose for the estimation of the top-k singular values and the estimation of their factor score ratios scale polylogarithmically in the size of the input matrix. Instead, the algorithm for the extraction of the top-k singular vectors scales proportionally to the size and the number of the top-k singular vectors, resulting in a polynomial speed-up on the traditional algorithm. Furthermore, we have discussed how these algorithms can be used to speed-up data representations for machine learning. In particular, we have shown how to employ these algorithms for Principal Component Analysis, Correspondence Analysis, and Latent Semantic Analysis. For each of the three cases, we have provided an analysis of the run-time parameters and computed theoretical bounds on the accuracy of the retrieved models. The time complexities of such algorithms have reasonable dependencies on the approximation error and the smallest singular value of the desired k-rank approximation, never exceeding quadratic factors.

Experiments We have investigated the distribution of the factor score ratios in three different datasets: CIFAR-10, MINST 784 and Research papers. We have seen that in these three datasets the magnitudes of the factor score ratios decrease exponentially. Through the experiments on the MNIST dataset, we have shown that the error guarantees of our quantum PCA are tight enough to ensure the same classification performances of the non-approximated PCA representations. We have seen that the run-times parameters that should not depend on the number of elements in the matrix are of reasonable size. Moreover, we have seen that the first two algorithms scale sub-linearly even on small datasets.

Future works Even though we were not expecting huge speed-ups on the MNIST dataset, the fact that the extraction of the singular vectors did not scale sub-linearly in the experiment, together with the fact that we can still lower the approximation accuracy of the PCA representation without losing much classification accuracy, suggest that a possible future research direction is to prove the error bounds for the three data representation models using the ℓ_∞ tomography for the singular vectors estimation, as presented in Theorem 16, and run further experiments. This formulation introduces greater errors on the retrieved representation models but further reduces the computational complexity of the third algorithm, allowing to reduce the cost of vector state tomography from $O\left(\frac{n \log n}{\delta^2}\right)$ to $O\left(\frac{\log n}{\delta^2}\right)$. Thus, providing an exponential speed-up even in the singular vector extraction task. This improvement would allow our quantum routines to provide classical access to the singular value decomposition of large matrices with run-times that scale on error parameters and matrix properties in a way that classical algorithms could not provide for fundamental reasons [2]. Moreover, qubitization techniques from [3] and [16] could be used to further reduce the dependency on the singular value precision parameter from $O\left(\frac{1}{\varepsilon}\right)$ to $O\left(\log\left(\frac{1}{\varepsilon}\right)\right)$.

Conclusions We expect that, in the future, quantum algorithms will play a key role in many machine learning tasks, providing speed-ups that classical algorithms would not be able to achieve. Quantum procedures for representation learning have a huge potential and are worth being further investigated. We believe that many other representation algorithms, such as Independent Component Analysis and Factor Analysis, can substantially benefit from quantum advantage.

References

[1] Ambainis, A. (2012). Variable time amplitude amplification and quantum algorithms for linear algebra problems. In *29th International Symposium on Theoretical Aspects of Computer Science*, page 636. Citeseer.

[2] Arrazola, J. M., Delgado, A., Bardhan, B. R., and Lloyd, S. (2019). Quantum-inspired algorithms in practice. *arXiv*, pages arXiv–1905.

[3] Baier, C., Flocchini, P., and Leonardi, S. (2019). The power of block-encoded matrix powers: Improved regression techniques via faster hamiltonian simulation. In *46th International Colloquium on Automata, Languages, and Programming (ICALP 2019)*, volume 132, page 33. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik.

[4] Benioff, P. (1980). The computer as a physical system: A microscopic quantum mechanical hamiltonian model of computers as represented by turing machines. *Journal of statistical physics*, 22(5):563–591.

[5] Bhojanapalli, S., Jain, P., and Sanghavi, S. (2014). Tighter low-rank approximation via sampling the leveraged element. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 902–920. SIAM.

[6] Brassard, G., Hoyer, P., Mosca, M., and Tapp, A. (2002). Quantum amplitude amplification and estimation. *Contemporary Mathematics*, 305:53–74.

[7] Bravo-Prieto, C., García-Martín, D., and Latorre, J. I. (2020). Quantum singular value decomposer. *Physical Review A*, 101(6):062310.

[8] Chia, N.-H., Gilyén, A., Li, T., Lin, H.-H., Tang, E., and Wang, C. (2020). Sampling-based sublinear low-rank matrix arithmetic framework for dequantizing quantum machine learning. In *Proceedings of the 52nd Annual ACM SIGACT Symposium on Theory of Computing*, pages 387–400.

[9] Commons, W. (2019a). File:bloch sphere.svg — wikimedia commons,. [Online; accessed 24-agosto-2020].

[10] Commons, W. (2019b). File:gaussianscatterpca.svg — wikimedia commons, the free media repository. [Online; accessed 26-August-2020].

[11] Deerwester, S., Dumais, S. T., Furnas, G. W., Landauer, T. K., and Harshman, R. (1990). Indexing by latent semantic analysis. *Journal of the American society for information science*, 41(6):391–407.

[12] Eckart, C. and Young, G. (1936). The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218.

[13] Erdős, P. (1961). On a classical problem of probability theory.

[14] Feynman, R. P. (1982). Simulating physics with computers. *Int. J. Theor. Phys*, 21(6/7).

[15] Frieze, A., Kannan, R., and Vempala, S. (2004). Fast monte-carlo algorithms for finding low-rank approximations. *Journal of the ACM (JACM)*, 51(6):1025–1041.

[16] Gilyén, A., Su, Y., Low, G. H., and Wiebe, N. (2019). Quantum singular value transformation and beyond: exponential improvements for quantum matrix arithmetics. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing*, pages 193–204.

[17] Gu, L., Wang, X., and Zhang, G. (2019). Quantum higher order singular value decomposition. In *2019 IEEE International Conference on Systems, Man and Cybernetics (SMC)*, pages 1166–1171. IEEE.

[18] Harrow, A. W., Hassidim, A., and Lloyd, S. (2009). Quantum algorithm for linear systems of equations. *Physical review letters*, 103(15):150502.

[19] Hsu, H., Salamatian, S., and Calmon, F. P. (2019). Correspondence analysis using neural networks. In *The 22nd International Conference on Artificial Intelligence and Statistics*, pages 2671–2680.

[20] Kerenidis, I., Landman, J., Luongo, A., and Prakash, A. (2019a). q-means: A quantum algorithm for unsupervised machine learning. In *Advances in Neural Information Processing Systems*, pages 4134–4144.

[21] Kerenidis, I., Landman, J., and Prakash, A. (2019b). Quantum algorithms for deep convolutional neural networks. In *International Conference on Learning Representations*.

[22] Kerenidis, I. and Luongo, A. (2020). Classification of the mnist data set with quantum slow feature analysis. *Physical Review A*, 101(6):062327.

[23] Kerenidis, I. and Prakash, A. (2017). Quantum recommendation systems. In *8th Innovations in Theoretical Computer Science Conference (ITCS 2017)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik.

[24] Kerenidis, I. and Prakash, A. (2018). A quantum interior point method for lps and sdps. *arXiv*, pages arXiv–1808.

[25] Kerenidis, I. and Prakash, A. (2020). Quantum gradient descent for linear systems and least squares. *Physical Review A*, 101(2):022316.

[26] Kerenidis, I., Prakash, A., and Szilágyi, D. (2019c). Quantum algorithms for portfolio optimization. In *Proceedings of the 1st ACM Conference on Advances in Financial Technologies*, pages 147–155.

[27] Lin, J., Bao, W.-S., Zhang, S., Li, T., and Wang, X. (2019). An improved quantum principal component analysis algorithm based on the quantum singular threshold method. *Physics Letters A*, 383(24):2862–2868.

[28] Lloyd, S., Mohseni, M., and Rebentrost, P. (2014). Quantum principal component analysis. *Nature Physics*, 10(9):631–633.

[29] Low, G. H. and Chuang, I. L. (2019). Hamiltonian simulation by qubitization. *Quantum*, 3:163.

[30] Manara, M., Perotti, A., and Scapellato, R. (2007). *Geometria e algebra lineare*. Esculapio.

[31] Manin, Y. I. (1980). Vychislimoe i nevychislimoe (computable and noncomputable), moscow: Sov.

[32] Mirsky, L. (1960). Symmetric gauge functions and unitarily invariant norms. *The quarterly journal of mathematics*, 11(1):50–59.

[33] Nielsen, M. A. and Chuang, I. L. (2010). *Quantum Computation and Quantum Information*. Cambridge university press.

[34] Partridge, M. and Calvo, R. (1997). Fast dimensionality reduction and simple pca. *Intelligent data analysis*, 2(3):292–298.

[35] Rebentrost, P., Mohseni, M., and Lloyd, S. (2014). Quantum support vector machine for big data classification. *Physical review letters*, 113(13):130503.

[36] Rebentrost, P., Steffens, A., Marvian, I., and Lloyd, S. (2018). Quantum singular-value decomposition of nonsparse low-rank matrices. *Physical review A*, 97(1):012327.

[37] Ross, D. A., Lim, J., Lin, R.-S., and Yang, M.-H. (2008). Incremental learning for robust visual tracking. *International journal of computer vision*, 77(1-3):125–141.

[38] Schlesinger, E. (2011). *Algebra lineare e geometria*. Zanichelli.

[39] Schuld, M. and Petruccione, F. (2018). *Supervised Learning with Quantum Computers*. Springer.

[40] Strang, G. (2016). *Introduction to linear algebra*. Wellesley - Cambridge Press.

[41] Ta-Shma, A. (2013). Inverting well conditioned matrices in quantum logspace. In *Proceedings of the forty-fifth annual ACM symposium on Theory of computing*, pages 881–890.

[42] Wang, H., Vo, L., Calmon, F. P., Médard, M., Duffy, K. R., and Varia, M. (2019). Privacy with estimation guarantees. *IEEE Transactions on Information Theory*, 65(12):8025–8042.

[43] Wang, S., Fontana, E., Cerezo, M., Sharma, K., Sone, A., Cincio, L., and Coles, P. J. (2020a). Noise-induced barren plateaus in variational quantum algorithms. *arXiv e-prints*, pages arXiv–2007.

[44] Wang, X., Song, Z., and Wang, Y. (2020b). Variational quantum singular value decomposition. *arXiv*, pages arXiv–2006.

[45] Wilson, E. B. (1927). Probable inference, the law of succession, and statistical inference. *Journal of the American Statistical Association*, 22(158):209–212.

List of figures

List of tables