

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

5,800

Open access books available

142,000

International authors and editors

180M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Chapter

Joint EigenValue Decomposition for Quantum Information Theory and Processing

*Gilles Burel, Hugo Pillin, Paul Baird, El-Houssain Baghious
and Roland Gautier*

Abstract

The interest in quantum information processing has given rise to the development of programming languages and tools that facilitate the design and simulation of quantum circuits. However, since the quantum theory is fundamentally based on linear algebra, these high-level languages partially hide the underlying structure of quantum systems. We show that in certain cases of practical interest, keeping a handle on the matrix representation of the quantum systems is a fruitful approach because it allows the use of powerful tools of linear algebra to better understand their behavior and to better implement simulation programs. We especially focus on the Joint EigenValue Decomposition (JEVD). After giving a theoretical description of this method, which aims at finding a common basis of eigenvectors of a set of matrices, we show how it can easily be implemented on a Matrix-oriented programming language, such as Matlab (or, equivalently, Octave). Then, through two examples taken from the quantum information domain (quantum search based on a quantum walk and quantum coding), we show that JEVD is a powerful tool both for elaborating new theoretical developments and for simulation.

Keywords: quantum information, quantum coding, quantum walk, quantum search, joint eigenspaces, joint eigenvalues, joint eigenvectors

1. Introduction

The field of quantum information is experiencing a resurgence of interest due to the recent implementation of secure transmission systems [1] based on the teleportation of quantum states in metropolitan networks and in the context of satellite transmissions, further underscored by the development of quantum computers. A new path for intercontinental quantum communication opened up in 2017 when a source onboard a Chinese satellite made it possible to distribute entangled photons between two ground stations, separated by more than 1000 km [2, 3]. Experiments using optical fibers [4] and terrestrial free-space channels [5] have also proved that the use of quantum entanglement can be achieved over large distances.

Quantum programming languages, such as Q# [6] have been developed to facilitate the design and simulation of quantum circuits. The underlying quantum theory is quite complex and often counter-intuitive due to the fact that it relies on linear algebra and tensor products—for instance, the state of a set of three independent qubits (quantum bits) is not described by a 3-dimensional vector, as would be the case for classical bits, but by a 2^3 -dimensional vector which lives in a Hilbert space constructed by tensor products of lower-dimensional spaces. Therefore, these programming languages are helpful for people who do not need to bother with the underlying theory.

However, since the quantum theory is fundamentally based on linear algebra, there are cases of practical interest for the researcher in which keeping a handle on the matrix representation of the quantum systems is a fruitful approach because it allows the use of powerful tools of linear algebra to better understand their behavior and to better implement simulation programs.

In this chapter, our objective is to illustrate how the concept of Joint EigenValue Decomposition (JEVD) can provide interesting results in the domain of quantum information. The chapter is organized as follows. In Section 2, we give some mathematical background and in Section 3, we provide basic elements to understand quantum information. Then, in Section 4, we show an example of the application of JEVD to quantum coding, more precisely we propose an algorithm, based on JEVD, to identify a quantum encoder matrix from a collection of given Pauli errors. Finally, in Section 5, we show that JEVD is a powerful tool for the analysis of a quantum walk search. More precisely, we prove that, while the quantum walk operates in a huge state space, there exists a small subspace that captures all the essential elements of the quantum walk, and this subspace can be determined thanks to JEVD.

2. Mathematical background

2.1 Matrices and notations

We note U^T the transpose of a matrix U and U^* the transpose conjugate of U . H is the normalized Hadamard 2×2 matrix and H_N the $N \times N$ Hadamard matrix obtained by the Kronecker product (defined in the next subsection):

$$H = \frac{1}{\sqrt{2}} \begin{pmatrix} 1 & 1 \\ 1 & -1 \end{pmatrix} \quad \text{and} \quad H_N = H^{\otimes n} \quad (N = 2^n) \quad (1)$$

I_N is the $N \times N$ identity matrix (which will sometimes be noted I when its dimension is implicit).

In the domain of quantum information processing, we mainly have unitary matrices. A square matrix U is unitary [7] if $U^* U = U U^* = I$. The columns of a unitary matrix are orthonormal and its eigenvalues are of norm 1. If the unitary matrix is real, its eigenvalues come by conjugate pairs.

We call “shuffle matrix” the permutation matrix $P_{a,b}$ which represents the permutation obtained when one writes elements row by row in an $a \times b$ matrix and reads them column by column. For instance, set $a = 2$ and $b = 3$. If one writes the elements 1, 2, 3, 4, 5, 6 row by row in a 2×3 matrix and reads them column by column, the order becomes 1, 4, 2, 5, 3, 6. Then the shuffle matrix is the permutation matrix such that $(1 \ 4 \ 2 \ 5 \ 3 \ 6) = (1 \ 2 \ 3 \ 4 \ 5 \ 6)P_{2,3}$. The inverse of $P_{a,b}$ is $P_{b,a} = (P_{a,b})^T$.

G_n is the $n \times n$ Grover diffusion matrix defined by [8]:

$$G_n = -I_n + 2\theta_n\theta_n^T \quad (2)$$

where θ_n the $(n \times 1)$ vector is defined by $\theta_n = [1 \ 1 \ \dots \ 1]^T / \sqrt{n}$. It is easy to see that $G_n\theta_n = \theta_n$. Therefore, θ_n is an eigenvector of G_n with eigenvalue $+1$. We can also see that for any vector v orthogonal to θ_n we have $G_nv = -v$. It follows that G_n has two eigenvalues, -1 and $+1$, and the dimensions of the associated eigenspaces are $n - 1$ and 1 .

2.2 Kronecker product

The Kronecker product, denoted by \otimes , is a bilinear operation on two matrices. If A is a $k \times l$ matrix and B is a $m \times n$ matrix, then the Kronecker product is the $km \times ln$ block matrix C below:

$$C = A \otimes B = \begin{pmatrix} a_{11}B & \dots & a_{1l}B \\ \vdots & \ddots & \vdots \\ a_{k1}B & \dots & a_{kl}B \end{pmatrix} \quad (3)$$

Assuming the sizes are such that one can form the matrix products AC and BD , an interesting property, known as the mixed-product property, is:

$$(A \otimes B)(C \otimes D) = (AC) \otimes (BD) \quad (4)$$

The Kronecker product is associative, but not commutative. However, there exist permutation matrices (the shuffle matrices defined in the previous subsection) such that, if A is an $a \times a$ square matrix and B a $b \times b$ square matrix, then [9]:

$$(A \otimes B)P_{a,b} = (B \otimes A)P_{b,a} \quad (5)$$

2.3 Singular value decomposition, image, and kernel

The Singular Value Decomposition (SVD) of an $m \times n$ matrix A is [7]:

$$A = USV^* \quad (6)$$

where U and V are unitary matrices, and S is diagonal. The diagonal of S contains the Singular Values, which are real nonnegative numbers, ranked by decreasing order. The sizes of the matrices are $U(m \times m)$, $S(m \times n)$ and $V(n \times n)$. The SVD is a very useful linear algebra tool because it reveals a great deal about the structure of a matrix.

The image and the kernel of A are defined by:

$$Im(A) = \{y \in \mathbb{C}^m : y = Ax \text{ for some } x \in \mathbb{C}^n\} \quad (7)$$

$$ker(A) = \{x \in \mathbb{C}^n : Ax = 0\} \quad (8)$$

When used in an algorithm, the notation *null* will also be used for a procedure that computes a matrix whose columns are an orthonormal basis of the kernel of A .

The complement of a subspace \mathcal{A} within a vector space \mathcal{H} is defined by:

$$(\mathcal{A})^c = \{y \in \mathcal{H} : x^*y = 0 \text{ for all } x \in \mathcal{A}\} \quad (9)$$

In an algorithm, if the columns of A are an orthonormal basis of \mathcal{A} then the columns of $B = \text{null}(A^*)$ provide an orthonormal basis of $(\mathcal{A})^c$.

The rank of A is its number of nonzero singular values. When programmed on a computer determination of the rank must take into account finite precision arithmetic, which means that “zero” is replaced by “extremely small” (less than a given tolerance value). Let us note $r = \text{rank}(A)$. We have

$$\dim(\text{Im}(A)) = r \quad (10)$$

$$\dim(\text{ker}(A)) = n - r \quad (11)$$

An orthonormal basis of $\text{ker}(A)$ is obtained by taking the last $n - r$ columns of the matrix V .

2.4 Joint eigenspaces and joint eigenvalue decomposition (JEVD)

The eigenvalue decomposition of a unitary matrix A is:

$$A = VDV^* \quad (12)$$

where D is a diagonal matrix, the diagonal of which contains the eigenvalues, and V is a unitary matrix whose columns are the eigenvectors.

Let us note E_λ^A the eigenspace of an operator A associated with an eigenvalue λ . The joint eigenspace $E_{\lambda,\mu}^{A,B}$ is:

$$E_{\lambda,\mu}^{A,B} = E_\lambda^A \cap E_\mu^B \quad (13)$$

A property of great interest in quantum information processing is that within $E_{\lambda,\mu}^{A,B}$ (and even within any union of joint eigenspaces) the operators A and B commute.

Determination of the joint eigenspace on a computer may be determined through the complement, because:

$$E_\lambda^A \cap E_\mu^B = \left((E_\lambda^A)^c \cup (E_\mu^B)^c \right)^c \quad (14)$$

Using Matrix-oriented programming languages, such as Matlab or Octave, this requires only a few lines. Let us note A_λ and B_μ matrices whose columns are orthonormal bases of E_λ^A and E_μ^B and $[\cdot]$ the horizontal concatenation of matrices. The following computation procedure provides a matrix C whose columns are an orthonormal basis of $E_{\lambda,\mu}^{A,B}$:

$$C = \text{null} \left(\left[\text{null}(A_\lambda) \quad \text{null}(B_\mu) \right] \right) \quad (15)$$

However, it is not efficient in terms of complexity and in the next sections we will propose faster computational procedures, adapted to each context.

A lower bound on the dimension of a joint eigenspace can be obtained as follows. Let us note n the dimension of the full space. We have, obviously:

$$\dim E_\lambda^A \cup E_\mu^B \leq n \quad (16)$$

and we know that:

$$\dim E_\lambda^A \cup E_\mu^B = \dim E_\lambda^A + \dim E_\mu^B - \dim E_\lambda^A \cap E_\mu^B \quad (17)$$

Combining both equations, we obtain:

$$\dim E_{\lambda,\mu}^{A,B} \geq \dim E_\lambda^A + \dim E_\mu^B - n \quad (18)$$

3. Quantum information principles

A quantum system is described by a state vector $|\psi\rangle \in \mathbb{C}^N$, where N is the dimension of the system. Since in the quantum formalism states $|\psi\rangle$ and $\gamma|\psi\rangle$ are equivalent, for any nonzero complex number γ , the state is usually represented by a normed vector and the global phase is considered irrelevant.

As long as it remains isolated, the evolution of a quantum system is driven by the Schrödinger equation. The latter is a first-order differential equation operating on the quantum state. Its integration shows that the quantum states at times t_1 and t_2 are linked by a unitary matrix U such that $|\psi_2\rangle = U|\psi_1\rangle$. The norm is preserved because U is unitary.

The second kind of evolution, called “measurement,” may occur if the system interacts with its environment. A measurement consists of the projection of the state onto a subspace of \mathbb{C}^N . When the measurement is controlled, it consists in defining *a priori* a decomposition of the state space into a direct sum of orthogonal subspaces $\bigoplus_i \mathcal{H}_i$. The measurement randomly selects one subspace. The result of the measurement is an identifier of the selected subspace (for instance, its index i). After measurement, the state is projected onto \mathcal{H}_i . If P_i is the projection matrix onto \mathcal{H}_i , then the state becomes $P_i|\psi\rangle$ (which is then renormalized because the projection does not preserve the norm). The probability of \mathcal{H}_i being selected is the square norm of $P_i|\psi\rangle$.

It is worth noting that a measurement may destroy a part of quantum information (because usually, a projection is an irreversible process), while the unitary evolution is reversible, and as such, preserves quantum information. Consequently, measurements must be used with extreme caution—how to design the system and the measurement device to measure only what is strictly required and not more is one of the difficult problems encountered in quantum information processing.

Quantum systems of special interest for quantum information processing are qubits (quantum bits) and qubit registers. A qubit belongs to a 2D quantum system with state a normed vector of \mathbb{C}^2 . To highlight links with classical digital computation, it is convenient to note $|0\rangle$ and $|1\rangle$ for the orthonormal basis of \mathbb{C}^2 . Physically any 2D quantum system can carry a quantum bit. For instance, the spin of an electron is a 2D quantum system, and the spins up and down can be associated with the basic states $|0\rangle$ and $|1\rangle$. A general qubit has an expression:

$$|\psi\rangle = \alpha_0|0\rangle + \alpha_1|1\rangle \quad (19)$$

where α_0 and α_1 are complex numbers subject to $|\alpha_0|^2 + |\alpha_1|^2 = 1$.

A qubit register is a 2^n -D quantum system which, for convenience, is usually referred to as a standard orthonormal basis noted $\{|0\dots 00\rangle, |0\dots 01\rangle, |0\dots 10\rangle, \dots, |1\dots 11\rangle\}$ and then, by analogy with classical digital processing, n is the number of qubits. For instance, for $n = 2$ the basis is $\{|00\rangle, |01\rangle, |10\rangle, |11\rangle\}$, where $|ab\rangle = |a\rangle \otimes |b\rangle$, and the quantum state of the register is:

$$|\psi\rangle = \sum_{(a,b) \in \{0,1\}^2} \gamma_{ab} |ab\rangle \quad (20)$$

Note that, contrary to classical digital registers, the qubits are usually not separable, hence the register must be considered as a whole. We say that the qubits are entangled. However in the special case where the coefficients γ_{ab} can be decomposed in the form $\gamma_{ab} = \alpha_a \beta_b$ the state can be written as a tensor product of the states of two qubits, which can be considered separately. Then, we have:

$$|\psi\rangle = (\alpha_0|0\rangle + \alpha_1|1\rangle) \otimes (\beta_0|0\rangle + \beta_1|1\rangle) \quad (21)$$

4. Application of JEVD to quantum coding

4.1 Principle of quantum coding

The objective of quantum coding is to protect quantum information [10]. In the classical domain, the information can be protected using redundancy—for instance, if we want to transmit bit 0 on a noisy communication channel, we can instead transmit 000 (and, similarly, transmit 111 instead of 1). On the receiver side, if one error has occurred on the channel, for instance, if the second bit is false, we receive 010 instead of 000, from which we can still guess that the most probable hypothesis is that the transmitted word was 000. Of course, if there were two errors the transmitted word could have been 111, but it is assumed that the probability of error is low, hence two errors are less likely than one error. More elaborated channel codes have been proposed, but fundamentally they are all based on the idea of adding redundancy and assuming that the probability of channel error is low.

In the quantum domain, it is impossible to use redundancy because it is impossible to copy a quantum state (this is due to the “no-cloning theorem” [11]). However, we can use entanglement to produce the quantum equivalent of classical redundancy. The principle of quantum coding is shown in **Figure 1**. Assume we want to protect the quantum state $|\psi\rangle$ of a k -qubit register. We add r ancillary qubits initialized to $|0\rangle$ to form an n -qubit register ($n = k + r$). The encoder is represented by a unitary $2^n \times 2^n$

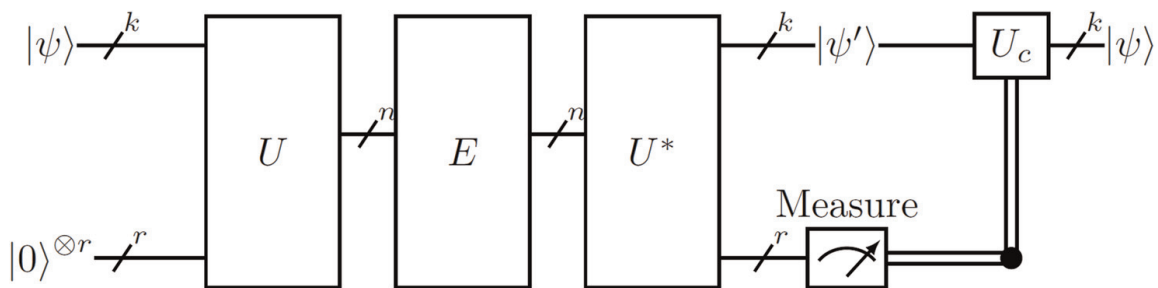


Figure 1.
Principle of quantum coding.

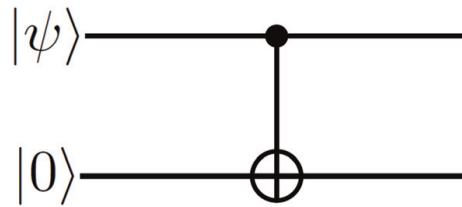


Figure 2.
 CNOT quantum gate.

matrix U . Then, errors may occur on the encoded state: they are represented by a unitary matrix E . The decoder is represented by another unitary matrix U^* which is the transpose conjugate of the encoding matrix. Finally, we measure the last r qubits of the decoded state, and, depending on the result of the measurement, we apply the appropriate restoration matrix U_c (which is a unitary matrix of size $2^k \times 2^k$) to the k -qubit register composed of the first k qubits of the decoded state.

As an illustration, let us consider $n = 2, k = 1$ and the very simple quantum encoder shown in **Figure 2**. It is a basic quantum circuit known as the CNOT quantum gate, and it is represented by the unitary matrix below:

$$U = \begin{pmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad (22)$$

A quantum error on a qubit is described by a 2×2 unitary matrix. It is convenient to decompose the error as a linear sum of the identity and the Pauli matrices below [12]:

$$Z = \begin{pmatrix} 1 & 0 \\ 0 & -1 \end{pmatrix} \quad X = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix} \quad Y = \begin{pmatrix} 0 & i \\ -i & 0 \end{pmatrix} \quad (23)$$

Let us consider that an error may appear on the first encoded qubit and that this error, if present, is represented by the unitary Pauli matrix X . Then, the error matrix which acts on the encoded state is:

$$E = X \otimes I = \begin{pmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{pmatrix} \quad (24)$$

It is easy to check that:

$$F = U^* E U = \begin{pmatrix} 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \end{pmatrix} = X \otimes X \quad (25)$$

The state at the input of the encoder is $[\alpha_0 \alpha_1]^T \otimes [10]^T = [\alpha_0 \ 0 \ \alpha_1 \ 0]^T$. The state at the output of the decoder is, therefore, $[0 \ \alpha_1 \ 0 \ \alpha_0]^T$.

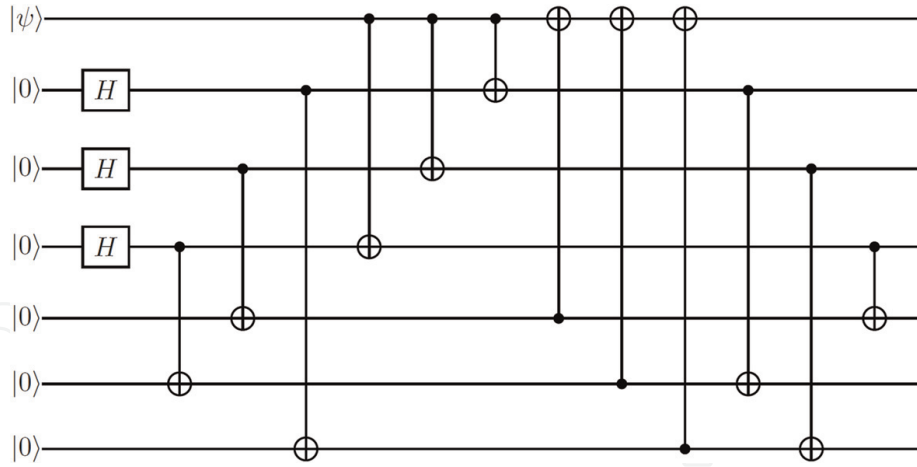


Figure 3.
Steane encoder.

Measuring the second qubit on the output of the decoder consists in decomposing the state space into a direct sum $\mathcal{H}_0 \oplus \mathcal{H}_1$ of two subspaces spanned by $\{|00\rangle, |10\rangle\}$ and $\{|01\rangle, |11\rangle\}$. The result of the measurement will be either 0 or 1 (index of the selected subspace), and by analogy with classical decoding, this result will be called the “syndrome.” The projections on these subspaces are $[00]^T$ and $[\alpha_1 \alpha_0]^T$. Then the probability to obtain syndrome 1 is 1.

The measurement then projects the state onto \mathcal{H}_1 . Note that in this particular case, the information is preserved by the projection. Then, applying the operator $U_c = X$ to the projected state restores the initial state.

Similarly, if there is no error, we can see that F is the identity matrix, then the projections on the subspaces are $[\alpha_0 \alpha_1]^T$ and $[00]^T$. In that case, the syndrome is 0 and the state is projected onto \mathcal{H}_0 . Correction is done by applying the operator I to the projected state, which is equivalent to doing nothing.

The very simple code used above, as an illustration, cannot correct more complex errors (for instance, an error Z on the first qubit). However, there exist efficient quantum codes, such as the Steane code [13], and the Shor code [14]. A remarkable result of quantum coding theory is that a linear combination of correctable errors is correctable [15].

Figure 3 shows the Steane Encoder, which is a $(n = 7, k = 1, t = 1)$ quantum encoder. This means that it encodes $k = 1$ qubits on $n = 7$ qubits and it is able to correct any error occurring on $t = 1$ encoded qubits. It is built with Hadamard (Eq. (1)) and CNOT (Eq. (22)) quantum gates. From this circuit description, it is possible to obtain the coding matrix U .

4.2 Determination of encoder matrix using JEVD

The problem we address can be stated as follows (see **Figure 1** for the notations)—given a list of n independent Pauli errors E_i with corresponding diagonal outer errors F_i , determine the unitary operator U (quantum encoder) such that:

$$U^* E_i U = F_i \quad \forall i \in \{1, \dots, n\} \quad (26)$$

This equation shows that the columns of U are the eigenvectors of E_i . Specification of the code by a small set of Pauli errors is very convenient and the interest of

E_i	F_i
$Z \otimes Z \otimes Z \otimes Z \otimes Z \otimes Z \otimes Z$	$Z \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I$
$X \otimes X \otimes I \otimes I \otimes I \otimes X \otimes X$	$I \otimes Z \otimes I \otimes I \otimes I \otimes I \otimes I$
$X \otimes I \otimes X \otimes I \otimes X \otimes I \otimes X$	$I \otimes I \otimes Z \otimes I \otimes I \otimes I \otimes I$
$X \otimes I \otimes I \otimes X \otimes X \otimes X \otimes I$	$I \otimes I \otimes I \otimes Z \otimes I \otimes I \otimes I$
$Z \otimes Z \otimes I \otimes I \otimes I \otimes Z \otimes Z$	$I \otimes I \otimes I \otimes I \otimes Z \otimes I \otimes I$
$Z \otimes I \otimes Z \otimes I \otimes Z \otimes I \otimes Z$	$I \otimes I \otimes I \otimes I \otimes I \otimes Z \otimes I$
$Z \otimes I \otimes I \otimes Z \otimes Z \otimes Z \otimes I$	$I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes Z$

Table 1.
Collection of Pauli errors.

automatic determination of matrix U is to allow further simulations of the behavior of the quantum code in various configurations.

To illustrate and validate the approach that will be developed below, let us consider the collection of $n = 7$ Pauli errors shown in **Table 1**. Here, to be able to check the results, this collection has been chosen to correspond to the Steane encoder (**Figure 3**), while in a standard application of the method, it would be given *a priori*. The interest is that here we can compute the encoder matrix from the circuit and this will allow us to check that our method produces the correct encoder matrix.

We use n independent equations in which each F_i is a tensor product of I and Z only (including only one Z). Therefore, matrices F_i are diagonal, and their diagonal elements are $+1$ and -1 in equal numbers.

Figure 4 shows the diagonals of the matrices F_i (each row corresponding to one diagonal). Values -1 and $+1$ are represented, respectively, by black and white dots.

Since matrix U does not depend on i in Eq. (26) its columns are joint eigenvectors of the E_i . For instance, in the example above, the 20th column of U is a joint eigenvector of E_1, E_2, \dots, E_7 associated to eigenvalues $+1, +1, -1, +1, +1, -1, -1$ (see **Figure 4**). In the general case, the set of n eigenvalues corresponding to the column c of U is easily obtained by taking the binary representation of $c - 1$ with the mapping $0 \rightarrow +1$ and $1 \rightarrow -1$.

Now, let us consider the determination of column c of U . We know that it is a vector spanning a joint eigenspace of the E_i corresponding to a given set of eigenvalues $\{\lambda_i, i = 1, \dots, n\}$. For each E_i let A_i denote the $2^n \times 2^{n-1}$ matrix whose orthonormal columns span the eigenspace associated to λ_i and B_i the $2^n \times 2^{n-1}$ matrix whose orthonormal columns span the kernel of A_i (which corresponds to the eigenspace associated to $-\lambda_i$).

Let \mathcal{Y}_k denote the joint eigenspace corresponding to eigenvalues $\{\lambda_j, j = 1, \dots, k\}$ with $k \in \{1, \dots, n\}$. We propose Algorithm 1 to efficiently compute the column of U . It computes a series of matrices Y_k whose columns are an orthonormal basis of \mathcal{Y}_k . Obviously, the searched column of U is Y_n . For the moment, let us consider that $K(c)=1$ (the optimal value will be discussed later).

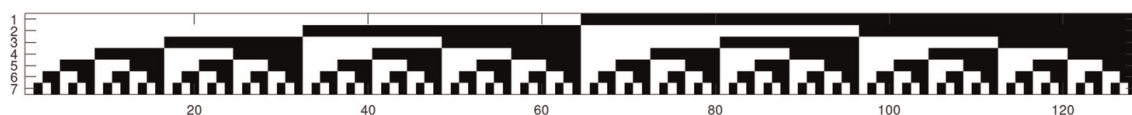


Figure 4.
Diagonals of matrices F_i .

```

if  $K(c) = 1$  then
  |  $Y_1 = A_1$ 
end
for  $k = K(c) + 1$  to  $n$  do
  |  $C_k = B_k^* Y_{k-1}$ 
  |  $Z_k = \text{null}(C_k)$ 
  |  $Y_k = Y_{k-1} Z_k$ 

```

Algorithm 1: Algorithm for determination of a joint eigenspace.

The sizes of the matrices are decreasing with k :

$C_k: 2^{n-1} \times 2^{n-k+1}$ $Z_k: 2^{n-k+1} \times 2^{n-k}$ $Y_k: 2^n \times 2^{n-k}$.

The intuitive ideas under the algorithm are the following:

- $C_k = B_k^* Y_{k-1}$: The orthonormal basis of \mathcal{Y}_{k-1} is projected on the kernel of A_k . The components of the projected vectors are expressed in the orthonormal basis B_k of that kernel. Consequently, $\text{Im}(C_k)$ is the projection of \mathcal{Y}_{k-1} on the kernel of A_k , expressed in that kernel.
 - A matrix Z_k whose columns are an orthonormal basis of the complement of this projection is determined.
 - Finally, the components of the basis vectors are restored to the original space by $Y_k = Y_{k-1} Z_k$
-

Let us prove that the matrices Y_k have orthonormal columns. This is obviously the case for $k = 1$. Then, by recursion, we have:

$$Y_k^* Y_k = Z_k^* Y_{k-1}^* Y_{k-1} Z_k = I \quad (27)$$

Now, let us prove, by recurrence, that $\text{Im}(Y_k) = \mathcal{Y}_k$.

Obviously, this is the case for $k = 1$. Assume this is the case for $k - 1$. We have:

$$\text{Im}(Y_k) \subset \text{Im}(Y_{k-1}) = \mathcal{Y}_{k-1} \quad (28)$$

We have also:

$$B_k^* Y_k = B_k^* (Y_{k-1} Z_k) = (B_k^* Y_{k-1}) Z_k = C_k Z_k = 0 \quad (29)$$

Then

$$\text{Im}(Y_k) \subset \ker(B_k^*) = \text{Im}(A_k) \quad (30)$$

From $\text{Im}(Y_k) \subset \mathcal{Y}_{k-1}$ and $\text{Im}(Y_k) \subset \text{Im}(A_k)$ we deduce $\text{Im}(Y_k) \subset \mathcal{Y}_k$.

Conversely, assume that a vector x belongs to \mathcal{Y}_k . Because $\mathcal{Y}_k \subset \mathcal{Y}_{k-1}$ there exists a vector b such that $x = Y_{k-1} b$ and because $x \in \text{Im}(A_k)$ we have also $B_k^* x = 0$

E_i	F_i
$X \otimes X \otimes X \otimes X \otimes X \otimes X \otimes X$	$X \otimes I \otimes I \otimes I \otimes I \otimes I \otimes I$
$Z \otimes I \otimes Z \otimes Z \otimes Z \otimes Z \otimes Z$	$I \otimes X \otimes I \otimes I \otimes I \otimes I \otimes I$
$Z \otimes Z \otimes I \otimes Z \otimes Z \otimes Z \otimes Z$	$I \otimes I \otimes X \otimes I \otimes I \otimes I \otimes I$
$Z \otimes Z \otimes Z \otimes I \otimes Z \otimes Z \otimes Z$	$I \otimes I \otimes I \otimes X \otimes I \otimes I \otimes I$
$X \otimes I \otimes I \otimes I \otimes X \otimes I \otimes I$	$I \otimes I \otimes I \otimes I \otimes X \otimes I \otimes I$
$X \otimes I \otimes I \otimes I \otimes I \otimes X \otimes I$	$I \otimes I \otimes I \otimes I \otimes I \otimes X \otimes I$
$X \otimes I \otimes I \otimes I \otimes I \otimes I \otimes X$	$I \otimes I \otimes I \otimes I \otimes I \otimes I \otimes X$

Table 2.
 Additional Collection of Pauli errors.

Then

$$B_k^* Y_{k-1} b = 0 \Rightarrow C_k b = 0 \Rightarrow \exists a : b = Z_k a$$

Therefore $x = Y_{k-1} b = Y_{k-1} Z_k a = Y_k a \Rightarrow x \in \text{Im}(Y_k) \Rightarrow \mathcal{Y}_k \subset \text{Im}(Y_k)$.

After execution of the algorithm to determine each column of U , there remains an indetermination because the joint eigenvectors (i.e., the columns of U) are determined up to a phase factor. This has no consequence on the performance of the quantum code. However, if we want to fix this residual indetermination, we proposed a fast and simple procedure in ref. [16]. The procedure requires an additional set of n Pauli errors in which each additional F_i is a tensor product of I and X only. As an example, for the Steane code, we use **Table 2**.

After these remaining differences have been removed, we obtain an estimated matrix U that is equal to the true matrix, up to a global phase (**Figure 5**). However, this remaining indetermination does not matter because, as said before, the global phase has no significance in quantum physics. Here we have chosen the global phase so that the encoder matrix is real.

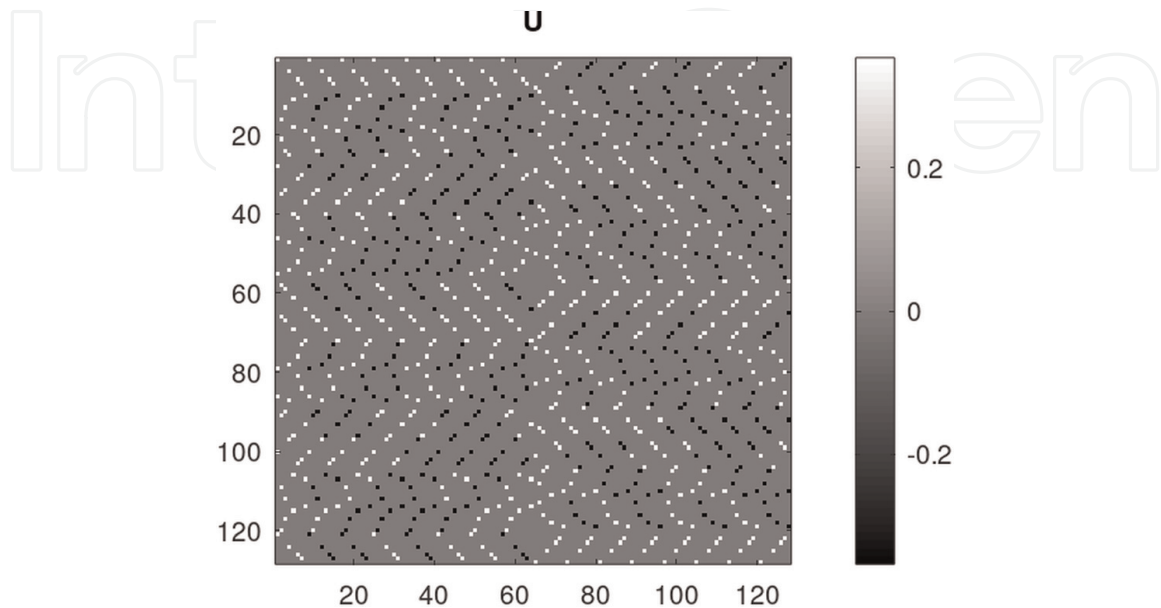


Figure 5.
 Estimated Matrix U for the Steane encoder.

Figure 5 shows the matrix computed by our method. We have checked that it is equal to the matrix directly computed from the circuit description.

The programmer may speed up the computation by taking into account the fact that when computing columns c of U , some matrices Y_k have already been computed for other columns and can be reused. For instance, in **Figure 4**, we see that the joint eigenvalues corresponding to columns 19 and 20 are the same, except the last one. Then, when computing column 20, we can set $K(20) = n$ in Algorithm 1 instead of the default value $K(20) = 1$, because the Y_{n-1} for column 20 is the same as for column 19. More generally, Algorithm 2 written in pseudo-Octave code computes the optimal values of the $K(c)$.

```

K = [1  1]
for k = 2 to n do
  | K = reshape( [K; k * ones(1, 2^{k-1})] [1 2^k] )
end

```

Algorithm 2: Algorithm for computation of the optimal values $K(c)$.

For instance, for $n = 3$ the algorithm produces $K = [1 \ 3 \ 2 \ 3 \ 1 \ 3 \ 2 \ 3]$.

5. Application of JEVD to quantum walk search

5.1 Principle of quantum walk search

Let us consider a particle that can move on a graph. In the classical world, at the time t this particle is localized at a node of the graph. It can then randomly choose one of the edges linked to this node to reach one of the adjacent nodes at a time $t + 1$. The repeated iteration of this process is the concept of classical random walk.

A quantum walk [17] relies also on a graph, but contrary to the classical walk, here the particle may be located at many nodes at the same time and can choose many edges simultaneously. At the time t , the state of the particle is then described by a state vector $|\psi_t\rangle$ and the evolution between times t and $t + 1$ is given by a unitary matrix $U = SC$ such that $|\psi_{t+1}\rangle = U|\psi_t\rangle$. The unitary matrices C and S represent, respectively, the choice of the edges and the movement to the adjacent nodes.

In the following, we will consider graphs associated with hypercubes [18]. We will note n the dimension of the hypercube and $N = 2^n$ the number of nodes. **Figure 6** shows the graph corresponding to a hypercube of dimension $n = 3$. It is convenient to label the nodes by binary words. In quantum language, these binary words κ are also used to label the basis vectors of the so-called position space \mathcal{H}^S .

The quantum state lives in a Hilbert space built by the tensor product of the position space \mathcal{H}^S (corresponding to the nodes) and the coin space \mathcal{H}^C (corresponding to the possible movements along the edges) $\mathcal{H} = \mathcal{H}^S \otimes \mathcal{H}^C$. The dimensions of these state spaces are $N_e = nN$, N and n .

It is usual to define C as [19]:

$$C = I_N \otimes G_n \tag{31}$$

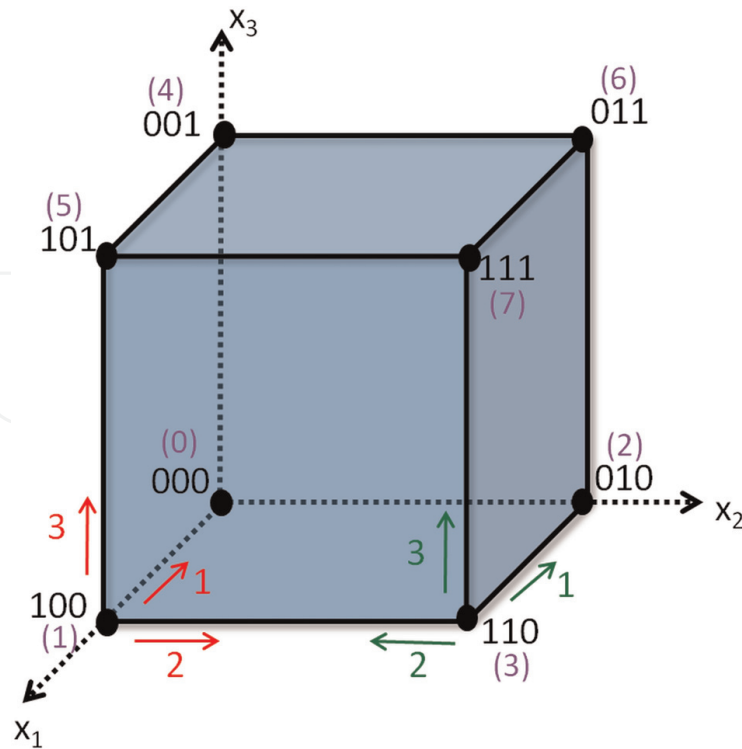


Figure 6.
 Hypercube for $n = 3$.

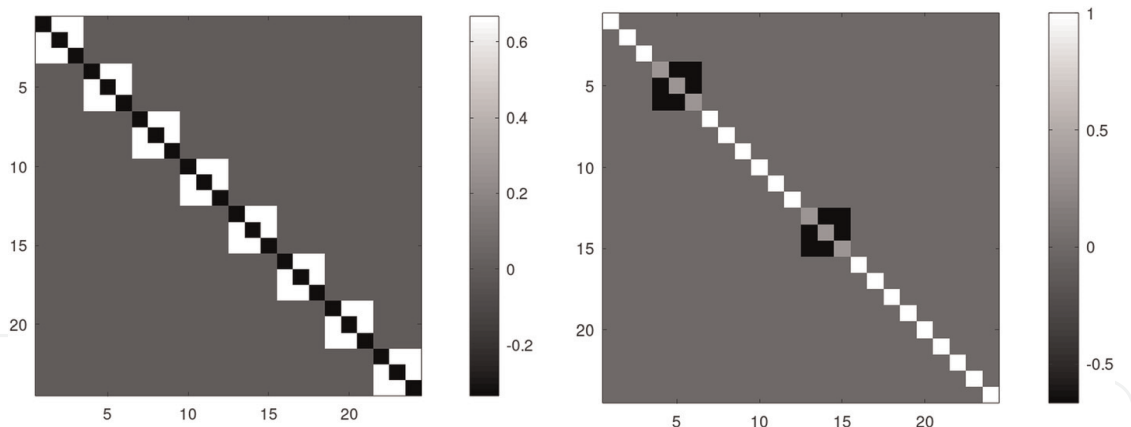


Figure 7.
 Matrices C (left) and O (right) for $n = 3$.

where G_n is the $n \times n$ Grover diffusion matrix defined in Section 2. Matrix C obtained for $n = 3$ is shown on **Figure 7**.

The structure of S is more complex. It is convenient to first define it in $\mathcal{H}^C \otimes \mathcal{H}^S$ and then to transpose it to \mathcal{H} using the shuffle matrix $P = P_{n,N}$ (defined in subsection 2.2). Then:

$$S = P\hat{S}P^T \quad (32)$$

where

$$\hat{S} = \text{diag}(\hat{S}_1, \dots, \hat{S}_n) \quad \text{and} \quad \hat{S}_d = I^{\otimes(n-d)} \otimes X \otimes I^{\otimes(d-1)} \quad (33)$$

The last equation just means that, because a movement along direction d corresponds to an inversion of the d^{th} bit in κ , the shift operator permutes the values associated to nodes that are adjacent along that dimension.

A quantum walk search can be described by repeated application of a unitary evolution operator Q , which can be written:

$$Q = UO \quad (34)$$

Here O is the oracle, which aims at marking the solutions. An example of oracle structure is shown in **Figure 7**. It is a block-diagonal matrix, whose blocks are $-G_n$ when they correspond to a solution and I_n otherwise. Denote M the number of solutions and assume that $M \ll N$ (otherwise the quantum walk search would serve no purpose because the probability of rapidly finding a solution with a classical search would be high). In the example shown in the figure, there are $M = 2$ solutions (located at positions 1 and 4 on **Figure 6**).

Let t denote the number of iterations until a measurement is performed. Starting from an initial state $|\psi_0\rangle$, repeated iterations lead to the state $|\psi_t\rangle = Q^t|\psi_0\rangle$ which is then measured. The theory of quantum walk search [19] shows that the probability of success (that is the probability of obtaining a solution by measurement) oscillates as a function of t . This means that theoretical tools which help to understand and simulate quantum walk search lead to the development of methods to determine the optimal time of measurement.

In the sequel, we will show that JEVD is a fruitful tool in this context. Indeed, set E to be the union of the joint eigenspaces of U and O , and \bar{E} its complement. Inside E , the operators commute. So, if we note with index E the restrictions of the operators to E , we have:

$$Q_E^2 = (U_E O_E)(U_E O_E) = U_E O_E^2 U_E = U_E^2 \quad (35)$$

Then, inside E , there is no significant difference between the effective quantum walk Q and the uniform quantum walk U , because, after each pair of successive iterations, the evolution is identical. Since the uniform quantum walk has no reason to converge to a solution, we deduce that the interesting part of the process lives in the complement of E , that is in \bar{E} .

After establishing results about the dimensions of the eigenspaces of U and O , we will show that the concept of joint eigenspaces allows us to establish an upper bound on the dimension of the complement, with the remarkable result that this dimension grows only linearly with n . Then, we propose an algorithm for efficient computation of the joint eigenspaces and, finally, use it to check our theoretical upper bound.

5.2 Eigenspaces of U and O

Set

$$F = H_N \otimes I_n \quad (36)$$

Then matrix F diagonalizes S :

$$FSF = (H_N \otimes I_n)P_{N,n}\hat{S}P_{n,N}(H_N \otimes I_n) \quad (37)$$

$$= P_{n,N}(I_n \otimes H_N)\hat{S}(I_n \otimes H_N)P_{N,n} \quad (38)$$

$$= P^T \text{diag}(\dots, H_N \hat{S}_d H_N, \dots)P \quad (39)$$

The latter term is diagonal because the mixed product property, $H^2 = I$ and $HXH = Z$, shows that:

$$H_N \hat{S}_d H_N = I^{\otimes(n-d)} \otimes Z \otimes I^{\otimes(d-1)} \quad (40)$$

Once more, using the mixed product property, we can also prove that F keeps C unchanged, that is:

$$FCF = C \quad (41)$$

The diagonal of FSF is the concatenation of the binary representation of the numbers 0 to $N - 1$ with the mapping $0 \rightarrow (+1)$ and $1 \rightarrow (-1)$. That is:

$$FSF = \text{diag}(S_0, \dots, S_\kappa, \dots, S_{N-1}) \quad (42)$$

Note that the diagonal of S_κ contains k times -1 and $n - k$ times $+1$ (where k is the Hamming weight of κ).

Then, because $F^2 = I$, FUF is a block diagonal matrix:

$$FUF = (FSF)(FCF) \quad (43)$$

$$= \text{diag}(\dots, S_\kappa, \dots)C \quad (44)$$

Block κ is then

$$U_\kappa = S_\kappa G_n \quad (45)$$

We have:

$$\dim E_-^{U_\kappa} \geq \dim E_{+,-}^{S_\kappa, G_n} \quad (46)$$

$$\geq \dim E_+^{S_\kappa} + \dim E_-^{G_n} - n \quad (47)$$

$$\geq (n - k) + (n - 1) - n \quad (48)$$

$$\geq n - k - 1 \quad (49)$$

and

$$\dim E_+^{U_\kappa} \geq \dim E_{-,-}^{S_\kappa, G_n} \quad (50)$$

$$\geq \dim E_-^{S_\kappa} + \dim E_-^{G_n} - n \quad (51)$$

$$\geq k + (n - 1) - n \quad (52)$$

$$\geq k - 1 \quad (53)$$

Then, there is only room left for at most 2 eigenvalues, specifically, at most a pair of conjugate ones.

Assume that this pair of eigenvalues exists. Since the diagonal of G_n contains $-1 + \frac{2}{n}$, the trace of U_κ is:

$$\text{trace}(U_\kappa) = ((n - k)(+1) + k(-1)) \left(-1 + \frac{2}{n} \right) \quad (54)$$

$$= -n + 2k + 2 \left(1 - 2\frac{k}{n} \right) \quad (55)$$

The sum of the eigenvalues is equal to the trace and we already have eigenvalue -1 with multiplicity $n - k - 1$ and eigenvalue $+1$ with multiplicity $k - 1$. The sum of these $n - 2$ eigenvalues is $-n + 2k$. Then the sum of the two missing eigenvalues must be $2(1 - 2\frac{k}{n})$. Let us denote them by λ_k and λ_k^* . We must have $\text{Re}(\lambda_k) = 1 - 2\frac{k}{n}$. Then, since $|\lambda_k| = 1$ we have

$$\lambda_k = \left(1 - 2\frac{k}{n} \right) + i \frac{2}{n} \sqrt{k(n - k)} \quad (56)$$

Considering the eigenvalues of $-G_n$ and I_n it is trivial to show that the dimensions of the eigenspaces of the oracle are:

$$\dim E_-^O = M \quad \text{and} \quad \dim E_+^O = N_e - M \quad (57)$$

5.3 Upper bound on the dimension of the complement

The eigenvalues of U belong to $\{-1, +1, \lambda_k, \lambda_k^*\}$ where $k \in [1, n - 1]$. Then, there are $2 + 2(n - 1) = 2n$ eigenspaces of U .

For $j \in [1, 2n]$ let α_j be the dimensions of these eigenspaces and β_j the dimensions of their intersections with E_+^O . An eigenvector of U is in an intersection if and only if it is orthogonal to E_-^O . Then, because the dimension of E_-^O is M , we have $\beta_j \geq \alpha_j - M$.

Consequently

$$\sum_{j=1}^{2n} \beta_j \geq \sum_{j=1}^{2n} \alpha_j - 2nM \quad (58)$$

Obviously, we have $\sum_{j=1}^{2n} \alpha_j = N_e$, so that

$$\sum_{j=1}^{2n} \beta_j \geq N_e - 2nM \quad (59)$$

It follows that the dimension of the complement has an upper bound:

$$\dim E_c \leq 2nM \quad (60)$$

This is a remarkable result—despite the fact that the dimension of the Hilbert space grows exponentially ($N_e = n2^n$), the dimension of the complement grows only linearly with n .

5.4 Fast computation of the joint eigenspaces

5.4.1 Introduction

To check our theoretical upper bound, we propose an efficient algorithm for fast computation of the joint eigenspaces.

We have to compute orthonormal bases of joint eigenspaces of U and O . The dimension of E_-^O is small, hence, it makes sense to define it by an orthonormal basis generating the eigenspace. However, the dimension of E_+^O is large (greater than $N_e/2$). Hence, it is computationally more efficient to define it by an orthonormal basis of its complement (which is E_-^O). Indeed $\dim E_-^O \ll \dim E_+^O$. We then have to design an algorithm adapted to each case.

5.4.2 Intersection of two eigenspaces defined by orthonormal bases

Let us consider a matrix A whose columns are an orthonormal basis of an eigenspace of U , and a matrix B whose columns are an orthonormal basis of E_-^O . Set p and q to be the number of columns of these matrices (their number of rows being N_e). We want to compute an $N_e \times r$ matrix J whose columns are an orthonormal basis of the joint eigenspace (whose dimension we have set to be r). We propose the algorithm below, which is a straightforward adaptation of Theorem 1 in ref. [20].

First, we compute the $p \times q$ matrix C below:

$$C = A^* B \quad (61)$$

Then, we compute the SVD of C :

$$C = U_c S_c V_c^* \quad (62)$$

Denote by s_k the singular values (the diagonal elements of S_c) and determine r such that $s_k \geq 1 - \varepsilon$ for $k = 1, \dots, r$, and $s_k < 1 - \varepsilon$ for $k > r$. Here $\varepsilon \ll 1$ is a very small positive value introduced to take into account the presence of small errors due to computer finite precision arithmetic. Finally:

$$J = A U_c(:, 1:r) \quad (63)$$

Or, equivalently, $J = B V_c(:, 1:r)$.

5.4.3 Intersection of two eigenspaces, one of them being defined by an orthonormal basis of its complement

Let us consider a matrix A whose columns are an orthonormal basis of an eigenspace of U , and a matrix B whose columns are an orthonormal basis of the complement of E_+^O (that is E_-^O). First, we compute the $p \times q$ matrix C (Eq. (61)). Then, we compute the $p \times r$ matrix ($r \leq p$) Z below:

$$Z = \text{null}(C^*) \quad (64)$$

λ^O	λ^U	$\dim E_{\lambda^O, \lambda^U}^{O,U}$
-1	any λ_k or λ_k^*	0
+1	$\lambda_0 = +1$	321
+1	λ_1 or λ_1^*	4
+1	λ_2 or λ_2^*	18
+1	λ_3 or λ_3^*	32
+1	λ_4 or λ_4^*	32
+1	λ_5 or λ_5^*	18
+1	λ_6 or λ_6^*	4
+1	$\lambda_7 = -1$	321

Table 3.
Joint eigenspaces of O and U for $n = 7$ and $M = 3$ solutions located at nodes 2, 8, 9.

and we obtain an $N_e \times r$ matrix J whose columns are an orthonormal basis of $E_{\lambda,+}^{U,O}$ from:

$$J = AZ \tag{65}$$

The justification of the algorithm is as follows. The q columns of C are a basis of the projection of $Im(B)$ into $Im(A)$, the components being expressed in the basis of $Im(A)$. The complement of $Im(C)$ in $Im(A)$ is the desired intersection (expressed in $Im(A)$). The columns of Z are an orthonormal basis of this intersection. Finally, Eq. (65) restores the components in the original space.

5.5 Simulation results

Consider a hypercube of dimension $n = 7$ with $M = 3$ solutions located at nodes 2, 8, 9. The dimension of the state space is then $N_e = n2^n = 896$. From the discussion above, we know that the dimension of the complement is upper bounded by $2nM = 42$.

The algorithm gives us the dimensions of the joint eigenspaces of U and O (Table 3). The sum of the dimensions of the joint eigenspaces is then $\sum_{j=1}^{2n} \beta_j = 858$, from which we obtain the dimension of the complement:

$$\dim E_c = N_e - \sum_{j=1}^{2n} \beta_j = 38 \tag{66}$$

We can see that, as expected, this dimension ($\dim E_c = 38$) is much smaller than the dimension of the original state space ($N_e = 896$). We can also check that it is less than the theoretical upper bound ($2nM = 42$), as expected.

6. Conclusions

The recent growth of research on quantum communications and quantum information processing opens new challenges. In this chapter, we have shown that matrix

theory concepts, such as JEVD, are powerful tools to propose new theoretical results as well as efficient simulation algorithms.

In the domain of quantum coding, we have shown how to determine the encoding matrix of a quantum code from a collection of Pauli errors. On a more speculative note to be part of future work concerning interception of quantum channels, it might also be useful to identify the quantum coder used by a noncooperative transmitter.

In the domain of quantum walk search, thanks to JEVD we have proved that there exists a small subspace of the whole Hilbert space which captures the essence of the search process, and we have given an algorithm that allows us to check this result by simulation.

Acknowledgements

The authors thank the IBNM (Institut Brestois du Numérique et des Mathématiques), CyberIoT Chair of Excellence, for its support.

Abbreviations

[qubit]	Quantum bit
[JEVD]	Joint EigenValue Decomposition
[SVD]	Singular Value Decomposition

Author details


Gilles Burel^{1*}, Hugo Pillin^{1,2}, Paul Baird², El-Houssain Baghious¹ and Roland Gautier¹

1 Lab-STICC, University of Brest, Brest, France

2 LMBA, University of Brest, Brest, France

*Address all correspondence to: gilles.burel@univ-brest.fr

IntechOpen

© 2022 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Humble TS. Quantum security for the physical layer. *IEEE Communications Magazine*. 2013;**51**(8):56-62
- [2] Liao SK et al. Satellite-to-ground quantum key distribution. *Nature*. 2017;**549**:43-47
- [3] Ren JG et al. Ground-to-satellite quantum teleportation. *Nature*. 2017;**549**:70-73
- [4] Valivarthi R et al. Quantum teleportation across a metropolitan fibre network. *Nature Photonics*. 2016;**10**: 676-680
- [5] Yin J et al. Quantum teleportation and entanglement distribution over 100-kilometre freespace channels. *Nature*. 2012;**488**:185-188
- [6] Microsoft. The Q# programming language user guide [Internet] 2022 Available from: <https://docs.microsoft.com/en-us/azure/quantum/user-guide/?view=qsharp-preview> [Accessed: January 11, 2022]
- [7] Golub GH, Van Loan CF. *Matrix Computations*. 3rd ed. Baltimore and London: The John Hopkins University Press; 1996
- [8] Grover LK. Quantum mechanics helps in searching for a needle in a haystack. *Physical Review Letters*. 1997;**79**(2):325
- [9] D'Angeli D, Donno A. Shuffling Matrices, Kronecker Product and Discrete Fourier Transform. *Discrete Applied Mathematics*. 2017;**233**:1-18
- [10] Raussendorf R. Key ideas in quantum error correction. *Philosophical Transactions of the Royal Society A*. 2012;**370**:4541-4565
- [11] Wootters W, Zurek W. A single quantum cannot be cloned. *Nature*. 1982;**299**(5886):802-803. DOI: 10.1038/299802a0
- [12] Nielsen MA, Chuang IL. *Quantum Computation and Quantum Information*. Cambridge, UK: Cambridge University Press; 2010
- [13] Steane A. Multiple-particle interference and quantum error correction. *Proceeding of the Royal Society of London*. 1996;**452**(1954): 2551-2577. DOI: 10.1098/rspa.1996.0136
- [14] Shor PW. Scheme for reducing decoherence in quantum computer memory. *Physical Review A*. 1995;**52**(4): R2493-R2496. DOI: 10.1103/PhysRevA.52.R2493
- [15] Calderbank AR, Shor PW. Good quantum error-correcting codes exist. *Physical Review A*. 1996;**54**(2): 1098-1105. DOI: 10.1103/physreva.54.1098
- [16] Burel G, Pillin H, Baghious EH, Baird P, Gautier R. Identification Of Quantum Encoder Matrix From A Collection Of Pauli Errors. Ho Chi Minh city, Vietnam: Asia-Pacific Conference on Communications; 2019
- [17] Kempe J. Quantum random walks – An introductory overview. *Contemporary Physics*. 2003;**44**(4):307-327. DOI: 10.1080/00107151031000110776
- [18] Moore C, Russell A. Quantum Walks on the Hypercube. *Lecture Notes in Computer Science*. Vol. 2483. New York: Springer. DOI: 10.1007/3-540-45726-7_14
- [19] Shenvi N, Kempe J, Whaley KB. Quantum random-walk search

algorithm. *Physical Review A*. 2003;**67**:
052307

[20] Bjorck A, GolubG. Numerical
methods for computing angles between
linear subspaces. *Mathematics of
Computation*. 1973;**27**:123. DOI: 10.2307/
2005662

IntechOpen

IntechOpen