Week 8: Density Matrices

Alice Pagano

(Dated: November 29, 2020)

In this Report, we initialize both a generic quantum state and a separable one. We note that for a separable state the initialization is faster, since less coefficients are needed. After that, we develop an algorithm for computing the density matrix for a generic state $|\psi\rangle \in \mathcal{H}^{D^N}$. Then, we implement also an algorithm for computing the reduced density matrix over the subsystem K.

I. THEORY

A. General Pure State

Let us consider a system composed by N subsystems each described by its wave function $|\psi_i\rangle \in \mathcal{H}^D$, where $\mathcal{H}^D$ is a D-dimensional Hilbert space. Computational basis states of such a system are built by tensor products of each subsystems basis vectors, namely $\{|j\rangle_i\}_{i=1,\dots,N}$. A generic state $|\psi\rangle \in \mathcal{H}^{D^N}$ can be written as

$$|\psi\rangle = \sum_{\vec{\mathbf{j}}} c_{\vec{\mathbf{j}}} |j\rangle_1 \otimes |j\rangle_2 \otimes \cdots \otimes |j\rangle_N \qquad \text{with} \qquad \sum_{\vec{\mathbf{j}}} \left|c_{\vec{\mathbf{j}}}\right|^2 = 1 \tag{1}$$

where $c_{\vec{\mathbf{j}}}$ corresponds to a set of D^N coefficients. In particular, if the pure state is **separable** it can be written as follows:

$$\begin{aligned} |\psi\rangle &= \sum_{\vec{\mathbf{j}}} c_{j_1} c_{j_2} \dots c_{j_N} |j\rangle_1 \otimes |j\rangle_2 \otimes \cdots \otimes |j\rangle_N \\ &= \sum_{j_1} c_{j_1} |j\rangle_1 \otimes \sum_{j_2} c_{j_2} |j\rangle_2 \otimes \cdots \otimes \sum_{j_N} c_{j_N} |j\rangle_N \end{aligned} \tag{2}$$

where now $c_{\vec{\mathbf{j}}}$ corresponds to a set of $D \times N$ coefficients.

B. Density Matrix

Let us consider a **pure state** $|\psi\rangle \in \mathcal{H}^{D^N}$, the corresponding **density matrix** of dimension $D^N \times D^N$ can be written as:

$$\rho = |\psi\rangle\langle\psi| \tag{3}$$

It is **positive semi-definite**, **hermitian** with $\mathrm{Tr}(\rho) = 1$. Moreover, for a pure state the density matrix has the property that $\rho^2 = \rho$, i.e. the state is idempotent.

Now, let us consider again a system composed by $N = 2$ subsystems each with Hilbert space $\mathcal{H}^D$. Let the state of the composite system be $|\psi\rangle \in \mathcal{H}^{D^2}$. In general, there is no way to associate a pure state to the component system 1. However, it still is possible to associate a density matrix. Let us consider the density matrix of the system $\rho = |\psi\rangle\langle\psi|$. The state of 1 is the partial trace of ρ over the basis of system 2:

$$\rho_1 \equiv \sum_{j=1}^{D} \langle j|_2 \left(|\psi\rangle\langle\psi|\right) |j\rangle_2 = \mathrm{Tr}_2(\rho) \tag{4}$$

and the same for the state of 2:

$$\rho_2 \equiv \sum_{j=1}^{D} \langle j|_1 \left(|\psi\rangle\langle\psi|\right) |j\rangle_1 = \mathrm{Tr}_1(\rho) \tag{5}$$

these are called **reduced density matrix** of the system and in general have a dimension $D^{N-1} \times D^{N-1}$.

II. CODE DEVELOPMENT

In order to write the total wave function ψ of a system of `N` subsystem each of dimension `D`, write the density matrix of the state $\rho = |\psi\rangle\langle\psi|$ and the reduced density matrix over the subsystem `K`, we develop a program inside the file "density_matrix". The main steps of the program are:

1. the total number of subsystems `N` and their space dimension `D` is given as input;

2. then, we call the `SUBROUTINE` **`state_init`**`(N,D,sep,state)`, which in particular take as input a logical variable `sep` which is `TRUE` if the state is **separable** and `FALSE` if it is a **generic** one. In particular, if the state is separable we initialized $D \times N$ coefficients, while for a generic state we have to initialize D^N coefficients. The coefficients are initialized with random entries between $[-1 : 1]$ and at the end of the initialization they are normalized.

```fortran
subroutine state_init(N,D,sep,state)
    ...
    ! Check if separable or not separable state
    if(sep .eqv. .TRUE.) then
        dim = D * N
    else if(sep .eqv. .FALSE.) then
        dim = D**N
    end if
    ...
    ! initialize with system generated seed
    do ii = 1,dim
        call random_number(rand_re)
        call random_number(rand_im)
        state(ii) = cmplx(2*rand_re-1,2*rand_im-1)
    end do

    ! Normalization of the coefficients
    norm = sum( state(:)*conjg(state(:)) )
    state(:) = state(:)/sqrt(norm)

end subroutine state_init
```

3. after that, we compute the density matrix (which can be computed only for not separable states) of the state by calling the `FUNCTION` **`pure_density_matrix`**`(state)`, where `state` has dimension `dim`. In this function, we associate to a `ket` matrix of dimension `(dim,1)` the vector `state` and to a `bra` matrix of dimension `(1,dim)` the conjugate of the `state`. Then, to compute the density matrix `densMat` we make the multiplication between `ket` and `bra`;

```fortran
function pure_density_matrix(state) result(densMat)
    ...
    dim = size(state)
    allocate(densMat(dim,dim), bra(1,dim), ket(dim,1))

    ket(:,1) = state
    bra(1,:) = conjg(state)
    densMat = matmul(ket,bra)

    return
end function pure_density_matrix
```

4. at the end, we call `FUNCTION` **`reduced_density_matrix`**`(densMat,N,D,K)` to compute the reduced density matrix over the subsystem `K`. In order to do that, we access to the density matrix elements in an efficient way by encoding the wave function in a d-basis representation.

```fortran
function reduced_density_matrix(densMat,N,D,K) result(red_densMat)
...
dim = size(densMat,1)
allocate(red_densMat(D**(N-1),D**(N-1)))
...
do ii1=1,D**(K-1)
    do ii2=1,(D**(N-K))
        do jj1=1,D**(K-1)
            do jj2=1,(D**(N-K))
                Tr = COMPLEX(0.0d0,0.0d0)
                do kk=1,D
                    ind_row = (ii1-1) + 1 + ((kk-1) + (ii2-1)*D)*D**(K-1)
                    ind_col = (jj1-1) + 1 + ((kk-1) + (jj2-1)*D)*D**(K-1)
                    Tr = Tr + densMat(ind_row,ind_col)
                end do
                ! compute reduced density matrix
                ind_red_row = (ii1-1) + 1 + (ii2-1)*D**(K-1)
                ind_red_col = (jj1-1) + 1 + (jj2-1)*D**(K-1)
                red_densMat(ind_red_row, ind_red_col) = Tr
            end do
        end do
    end do
end do
end function reduced_density_matrix
```

III. RESULTS

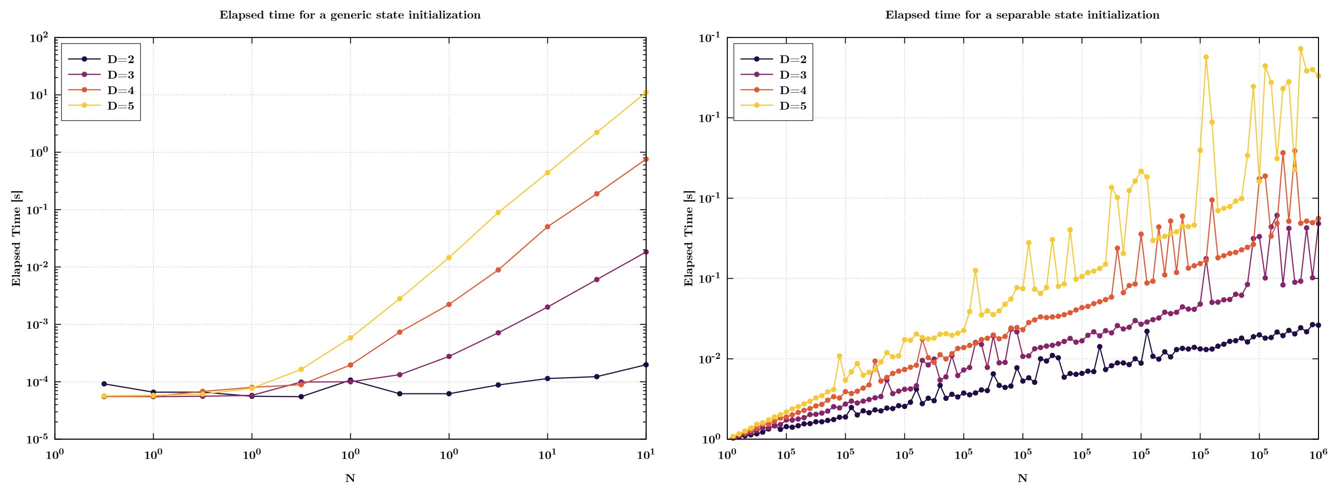

FIG. 1 Plot of the computational time required by the function for state coefficients initialization. **Right:** generic state. **Left:** separable state.

First of all, we test the performance of the subroutine for initializing a quantum state, i.e. finding random coefficients associated to the state. On the right of Fig. 1, we can see the results for a generic state initialization where the initialization of D^N coefficients is required. Instead, on the left, we see the computational time for initializing a separable state where only $D \times N$ coefficients has to be initialized. As far as a generic state is concerned, we see that as expected the computational time increases by increasing the space dimension of each subsystem D. For a separable state, we note instability in the curve which can be due to the fact that we are considering short time intervals in which other phenomena influence the execution of the code, such as the computer performance.

After that, we fix $N = 2$ and we compute the density matrix ρ associated to a generic state ψ. Then, we compute also the reduced density matrix both over the subsystem 1 ($\rho_1 = \mathrm{Tr}_2(\rho)$) and 2 ($\rho_2 = \mathrm{Tr}_1(\rho)$). For instance, starting from a generic two-qubits state ($D = 2$):

$$\psi = \begin{pmatrix} 0.531654 + 0.454865\, i \\ -0.041827 - 0.181760\, i \\ -0.428231 - 0.055035\, i \\ -0.515319 - 0.153920\, i \end{pmatrix}$$

we compute the following density matrix:

$$\rho = \begin{pmatrix} 0.489558 - 0.000000\, i & -0.104914 + 0.077608\, i & -0.252704 - 0.165528\, i & -0.343984 - 0.152569\, i \\ -0.104914 - 0.077608\, i & 0.034786 + 0.000000\, i & 0.027915 + 0.075533\, i & 0.049531 + 0.087226\, i \\ -0.252704 + 0.165528\, i & 0.027915 - 0.075533\, i & 0.186410 - 0.000000\, i & 0.229147 - 0.037553\, i \\ -0.343984 + 0.152569\, i & 0.049531 - 0.087226\, i & 0.229147 + 0.037553\, i & 0.289245 + 0.000000\, i \end{pmatrix}$$

The obtained reduced matrices are:

$$\rho_1 = \begin{pmatrix} 0.675968 - 0.000000\, i & 0.124233 + 0.040055\, i \\ 0.124233 - 0.040055\, i & 0.324032 + 0.000000\, i \end{pmatrix}, \qquad \rho_2 = \begin{pmatrix} 0.524344 - 0.000000\, i & -0.203173 - 0.078301\, i \\ -0.203173 + 0.078301\, i & 0.475656 - 0.000000\, i \end{pmatrix}$$

These results are in accordance with the theory, since we have that the diagonal entries of ρ, ρ_1 and ρ_2 are real numbers.

IV. SELF-EVALUATION

The code implemented works well and returns the results expected from the theory. We have seen how initialiazing a separable state can be faster than a generic one. In a further implementation of the code, it could be useful to compute the density matrix also for a separable state. For doing that, it is needed to recast the $D \times N$ coefficients into a state.