Density Matrices & QuBits

Saverio Monaco

Quantum Information and Computing

Dipartimento di Fisica e Astronomia
"Galileo Galilei"

UNIVERSITÀ
DEGLI STUDI
DI PADOVA

December 16, 2021

Goal of the exercise: numerical handling of pure N-body wavefunctions.

Assume N particles each described by a d-dimensional Hilbert space $(\varphi_i \in \mathcal{H}^d)$.

There are two possible systems:

- (general) **Unseparable case:**

$$\Psi = \sum_{a_1,a_2,\ldots,a_N} C_{a_1,a_2,\ldots,a_N} |a_1\rangle \otimes |a_2\rangle \otimes \ldots \otimes |a_N\rangle \qquad dim \equiv d^N$$

- **Separable case:** (no interactions between particles)

$$\Psi = \bigotimes_{i=1}^{N} \varphi_i \qquad dim \equiv d \times N$$

It is possible to construct an operator $\hat{\rho}$ out of the total wavefunction Ψ described before:

$$\rho = |\Psi\rangle\langle\Psi| = \sum_{a_1,..,a_N} \sum_{a'_1,..,a'_N} C_{a_1,..,a_N} C^*_{a'_1,..,a'_N} |a_1,..,a_N\rangle \langle a'_1,..,a'_N|$$

Properties:

- hermitian $\leftrightarrow \quad \rho = \rho^\dagger$
- $Tr(\rho) = 1$
- Eigenvalues of ρ are: $\{1, 0, ..., 0\}$

Other properties:

- $\rho^2 = \rho$
- $< \rho > = Tr(\rho^2) \leq 1$

Custom derived type:

```
type qsystem
  integer :: N ! Number of systems
  integer :: d ! number of states
  logical :: separability ! Wether the whole system is separable
  double complex, dimension(:), allocatable   :: PSI ! Total WF
  double complex, dimension(:,:), allocatable :: rho ! Density matrix
end type qsystem
```

Related functions:

```
function densmat_pure_init(N, d, SEP, DEBUG) result(system)
subroutine densmat_genstates(system)
subroutine densmat_readcoeffs(system)
function densmat_computerho1(system,d) result(rho1)
```

A explicit way to visualize the total wavefunction (non-separable case):

$$\Psi = a_0 |0,0,...,0\rangle + a_1 |0,0,...,1\rangle + ...$$
$$... + a_{d-1} |0,0,...,d-1\rangle + a_d |0,0,...,1,0\rangle + ...$$

Getting the index of the array from the eigenvectors combination and viceversa is just a **change of base**:

```
function basechange_to(b_to, number, N)
                        result(number_b_to)

integer :: b_to, number, N, ii
integer, dimension(N) :: number_b_to

number_b_to = 0*number_b_to
do ii = 1, N, 1
  number_b_to(N - ii + 1) = modulo(number, b_to)
  number = number/b_to
end do
end function
```

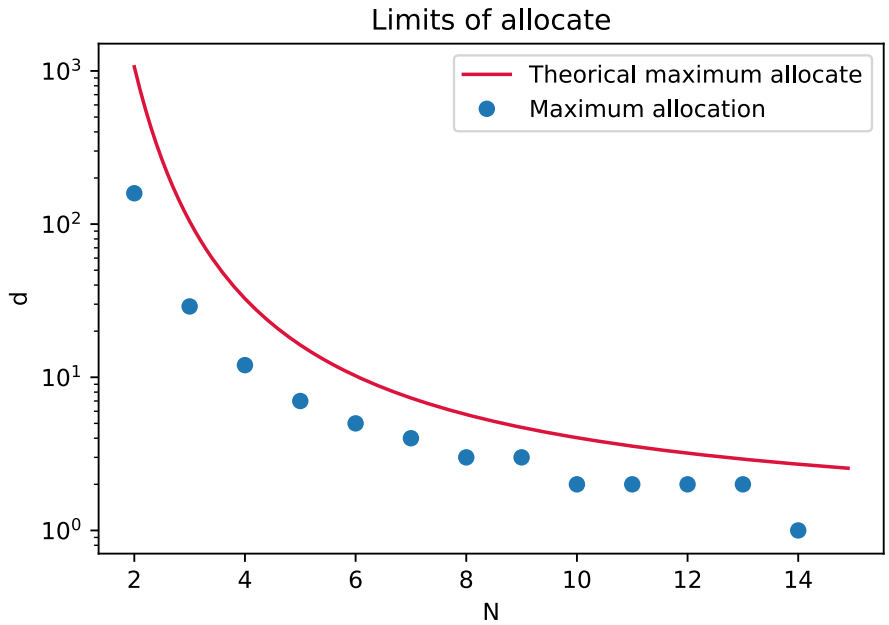
```
function basechange_from(b_from, number_from, N)
                          result(number_b10)

integer :: b_from, number_b10, N, ii
integer, dimension(N) :: number_from

do ii = 1, N, 1
  number_b10 = number_b10 +
               number_from(N - ii + 1)*
               b_from**(ii - 1)
end do
end function
```

Results

- **Correctness**: Trace and eigenvalues of ρ, (and it's partial trace as well) were computed to check the correctness. The results given match the theory up to some numerical error.
- **Stability**: The program may give errors when trying to allocate too much memory. The number of elements of ρ is d^{2N} and each element occupies 16 bytes (`double complex`)



Data was generated using a Ubuntu machine with 8GB of RAM