Presentation of Assignment 6

Quantum Information and Computing, A.Y. 2022/2023

Paolo Zinesi

13/12/2022

Theoretical introduction

Quantum system composed of N subsystems described by $\psi_i \in \mathcal{H}^D$. For convenience, we define the basis of $\mathcal{H}^D$ as $\{|0\rangle, |1\rangle, \ldots, |D-1\rangle\}$. The total wavefunction $|\Psi\rangle$ can be written as

$$|\Psi\rangle = \bigotimes_{i=1}^{N} |\psi_i\rangle = \left(\sum_{\alpha_1=0}^{D-1} \psi_{\alpha_1} |\alpha_1\rangle \right) \otimes \cdots \otimes \left(\sum_{\alpha_N=0}^{D-1} \psi_{\alpha_N} |\alpha_N\rangle \right) \qquad \text{if separable,}$$

$$|\Psi\rangle = \sum_{\alpha_1,\ldots,\alpha_N=0}^{D-1} \Psi_{\alpha_1,\ldots,\alpha_N} |\alpha_1\rangle \otimes \cdots \otimes |\alpha_N\rangle \qquad \text{otherwise,}$$

where the coefficients $\psi_{\alpha_i}, \Psi_{\vec{\alpha}} \in \mathbb{C}$ satisfy the normalization condition $\langle\Psi|\Psi\rangle = 1$.

Object	Storage (bytes)
Separable wavefunction $\|\Psi\rangle$	$16D \cdot N$
General N-body wavefunction $\|\Psi\rangle$	$16\,D^N$
General N-body density matrix $\hat{\rho}$	$16\,D^{2N}$

Table 1: Storage dimensions when $\mathrm{Re}(\alpha)$, $\mathrm{Im}(\alpha)$ are stored in double precision (i.e., in 8 bytes).

Testing - Pure wavefunction creation

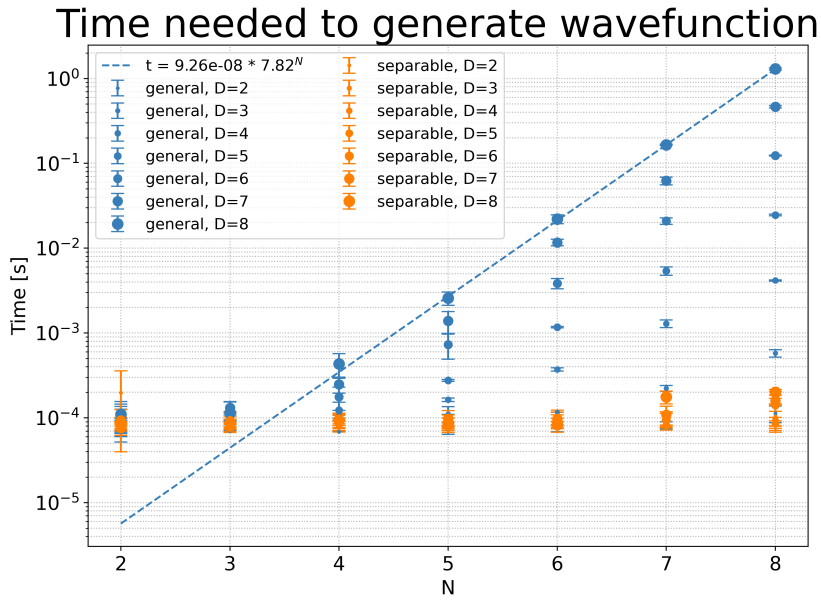
Separable state creation (manybody_functions.py)

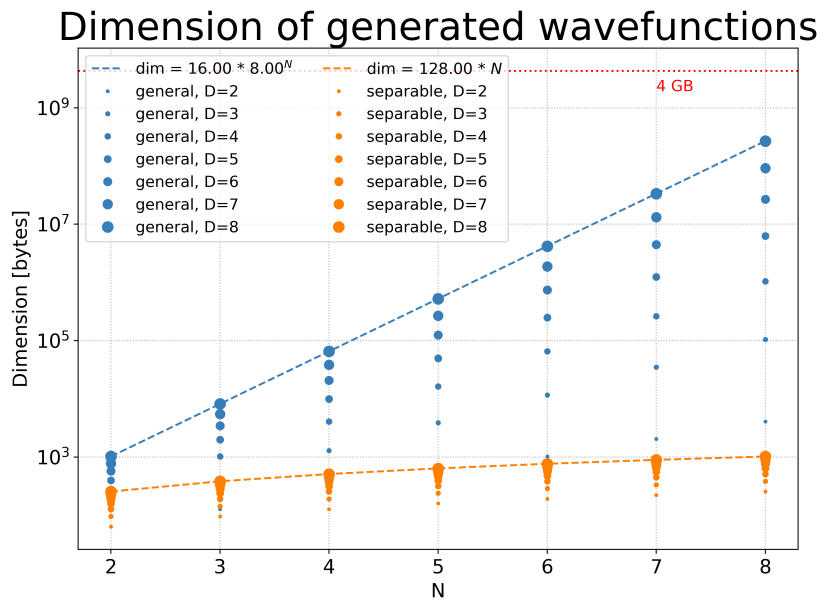
```
# the first terms are real (fix global phase)
WF = rng.standard_normal(size=(N,D,)) + \
     rng.standard_normal(size=(N,D,)) * 1.0j
WF[:,0] = np.real(WF[:,0])
WF /= np.sqrt(np.sum(np.abs(WF)**2, axis=1))\
        .reshape(-1,1)
```

General state creation (manybody_functions.py)

```
# the first term is real (fixes global phase)
WF = rng.standard_normal(size=(D**N,)) + \
     rng.standard_normal(size=(D**N,)) * 1.0j
WF[0] = np.real(WF[0])
WF /= np.sqrt(np.sum(np.abs(WF)**2))
```

The biggest limitation in the creation of a pure general wavefunction is the space needed to store its elements, which scales **exponentially** with the number of subsystems *N*. Instead, the storage dimension of a pure separable wavefunction scales only **linearly** with *N*.





Testing - Density matrix creation

Creation of a general state density matrix (manybody_functions.py)

```
WF, _, _ = generate_general_wf(N, D, rng)
rho = np.outer(np.conj(WF), WF)
```

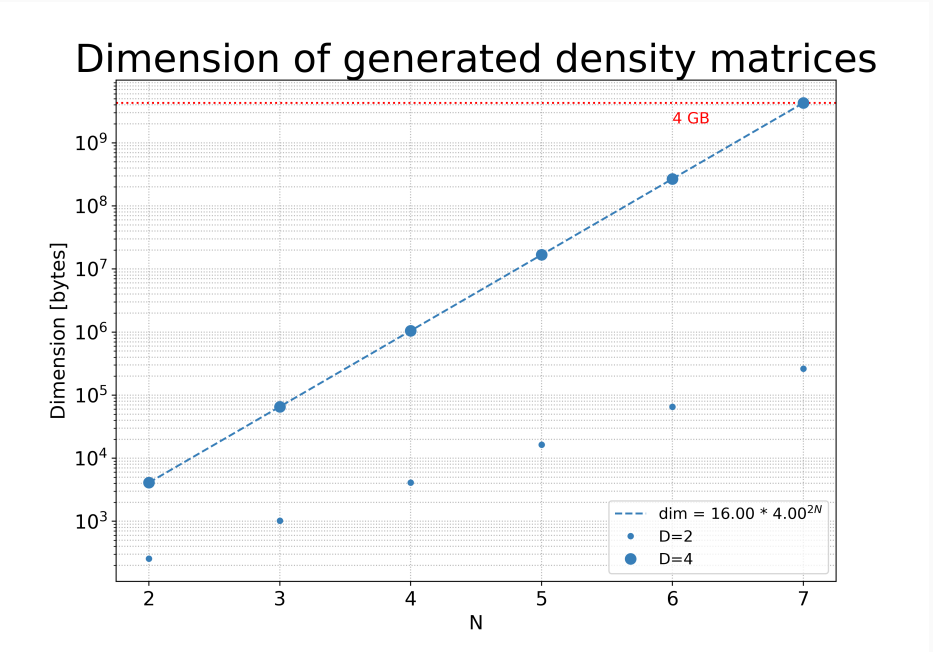
Density matrix of a general pure state $|\Psi\rangle$:

$$\hat{\rho} = |\Psi\rangle\langle\Psi|$$

Assuming a maximum storage of 4 GB in RAM for the density matrix, the maximum number of subsystems N_{max}, given the dimension D of $\mathcal{H}^D$, is

$$N_{max} = \frac{14}{\log_2 D}.$$

In particular, for $\mathbf{D = 2}$ the maximum number of qubits whose full density matrix can be stored in memory is $\mathbf{N_{max} = 14}$.



Testing - Partial trace of a generic density matrix

Example of multi-index handling functions for density matrices:

$$|01020\rangle \underset{\textit{vectorize\_idx}}{\overset{\textit{combine\_idxs}}{\rightleftharpoons}} 0 \cdot D^4 + 1 \cdot D^3 + 0 \cdot D^2 + 2 \cdot D^1 + 0 \cdot D^0$$

Partial trace function (manybody_functions.py)

```
Nred = len(traceout_indices) # list of indices to trace out
Nkeep = len(keep_indices) # list of indices that are kept

# allocation of reduced density matrix
rho_reduced = np.zeros(shape=(D**Nkeep,D**Nkeep), dtype=np.complex128)

# loop over all the elements of the reduced density matrix and fill them
for row in range(D**Nkeep):
    vec_row = vectorize_idx(row, N=Nkeep, D=D) # compute 'vectorized' row index
    for col in range(row, D**Nkeep):
        vec_col = vectorize_idx(col, N=Nkeep, D=D) # compute 'vectorized' column index

        # loop over all the indices to trace out, find the correct position in the
        # total density matrix rho and add element to the reduced density matrix
        for kk in range(D**Nred):
            vec_kk = vectorize_idx(kk, N=Nred, D=D)

            row_kk = combine_idxs(vec_kk, traceout_indices, vec_row, keep_indices, D=D)
            col_kk = combine_idxs(vec_kk, traceout_indices, vec_col, keep_indices, D=D)

            rho_reduced[row, col] += rho[row_kk, col_kk]

# the lower triangular matrix is simply the adjoint of the upper triangular matrix
rho_reduced = rho_reduced + np.conj(rho_reduced.T) - np.diag(np.diag(rho_reduced))
```

Testing - Partial trace of a generic density matrix

Test are performed on the two-dimensional Greenberger–Horne–Zeilinger (GHZ) state with N subsystems,

$$|GHZ\rangle = \frac{1}{\sqrt{2}}\left(|0\ldots0\rangle + |1\ldots1\rangle\right) = \\ = \frac{1}{\sqrt{2}}\left(|0\rangle^{\otimes N} + |1\rangle^{\otimes N}\right).$$

By tracing out a single subsystem k from $|GHZ\rangle\langle GHZ|$, the maximally mixed state

$$\hat{\rho}_{red} = \operatorname*{Tr}_k |GHZ\rangle\langle GHZ|$$

is obtained. In fact, the computed Von Neumann entropy is $S_{VN}(\hat{\rho}_{red}) > 1 - 10^{-16}$. The poor performances are due to the **three nested "for" loops** implied. Performance may be improved using dedicated Numpy functions (e.g. sum, reshape, einsum, etc.), but this approach is difficult to implement for a generic dimension D.