Documentation
Quantum Gaussian Information Toolbox

Igor Brandão*

Department of Physics, Pontifical Catholic University of Rio de Janeiro, Brazil

October 11, 2021

This file is an initial documentation for the Gaussian Quantum Information Numerical Toolbox in Python `https://github.com/IgorBrandao42/Gaussian-Quantum-Information-Numerical-Toolbox-python`. The toolbox is divided into two classes: one simulating gaussian states with methods for applying gaussian operations, measurements, entanglement criterias and and retrieving information from the state; and another class for calculating the open/closed dynamics of a given gaussian state following a set of quantum Langevin and Lyapunov equations, their steady state and semi-classical time evolution.

1 gaussian_state class

An instance of this class simulates a multimode gaussian state.

Definitions

Gaussian states are continuous variable state, whose Wigner function representation in phase-space is gaussian [1]. Let us briefly lay out some definitions:

For a N-modes continuous variable state, each mode is described by the annihilation ($\hat{a}_j$) and creation ($\hat{a}_j^\dagger$) operators, obeying bosonic commutation relations. These can be conventionally arranged in a $2N$-dimensional vectorial operator $\hat{\boldsymbol{b}} = (\hat{a}_1, \hat{a}_1^\dagger, \hat{a}_2, \hat{a}_2^\dagger, \ldots)^T$ whose commutation relations can be expressed as $\left[\hat{b}_j, \hat{b}_k\right] = \Omega_{jk}$ where $j, k = 1, \ldots, 2M$ and Ω is the $2N \times 2N$ symplectic form matrix given by

$$\Omega = \bigoplus_{k=1}^{M} \Omega_k \quad , \quad \Omega_k = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix} . \tag{1.1}$$

From these bosonic operators, we can define the corresponding quadrature operators $\hat{x}_j = \hat{a}_j^\dagger + \hat{a}_j$ and $\hat{p}_j = i(\hat{a}_j^\dagger - \hat{a}_j)$ and once again suitably arrange them into a $2M$-dimensional vectorial operator $\hat{\boldsymbol{X}} = (\hat{x}_1, \hat{p}_1, \hat{x}_2, \hat{p}_2, \ldots)^T$. It immediately follows from the bosonic commutation relations above that the quadratures must satisfy the canonical commutation relations $\left[\hat{X}_j, \hat{X}_k\right] = 2i\,\Omega_{jk}$.

From the definitions of gaussian states, they are completely characterized by their first moments,

$$\boldsymbol{R} \equiv \langle \hat{\boldsymbol{X}} \rangle = \mathrm{tr}\Big(\rho \hat{\boldsymbol{X}}\Big) ,$$

and second moments represented by the *covariance matrix* (CM), whose entries are given by

$$V_{j,k} = \frac{1}{2}\langle \hat{X}_j \hat{X}_k + \hat{X}_k \hat{X}_j \rangle - \langle \hat{X}_j \rangle \langle \hat{X}_k \rangle . \tag{1.2}$$

Class attributes

Gaussian states greatly simplify our treatment of continuous variable systems, as instead of dealing with high-dimensional density matrices/phase spaces, we need only to worry about $2N$-dimensional vectors and $2N \times 2N$ matrices. Thus, the internal variables of the gaussian_state class are

*igorbrandao@aluno.puc-rio.br

Class methods

In Table 1, we present the name and description of the methods of the gaussian_state class except one: the class constructor. It has the same name of the class and must be called to create an instance of it: a variable of type gaussian_state. There are essentially three ways the user can create such instance, dictated by the kind of input arguments:

- No arguments – default constructor returns a single mode vacuum state;
- Vector and matrix – constructor returns a multimode gaussian state with respective mean quadrature vector and covariance matrix;
- Name-pair value – the user provides a string with the name of a default single mode gaussian state and respective parameter:
    1. "vacuum" , – ;
    2. "coherent" , complex amplitude;
    3. "squeezed" , squeezing parameter;
    4. "thermal" , occupation number.

Table 1: Methods of the gaussian_state class

Method	Description	Reference
displace	Applies a displacement operator on a single mode gaussian state	[1]
squeeze	Applies a squeezing operator on a single mode gaussian state	[1]
rotate/phase	Applies a rotation operator on a single mode gaussian state	[1]
beam_splitter	Applies a beam splitter operator on a two mode gaussian state	[1]
two_mode_squeezing	Applies a two mode squeezing operator on a two mode gaussian state	[1]
tensor_product	Tensor product of two gaussian states	[2]
partial_trace	Partial trace over some modes	[2]
only_modes	Partial trace over all but some modes	[2]
matrix_element_coherent_basis	Calculates the density matrix elements on coherent state basis	[3]
matrix_element_number_basis	Calculates the density matrix elements on number states basis	[3]
purity	Purity	[1]
symplectic_eigenvalues	Symplectic eigenvalues of the covariance matrix	[1]
von_Neumann_Entropy	von Neumann entropy	[1]
mutual_information	Mutual information	—
occupation_number	Occupation number for each mode of the gaussian state	—
squeezing_degree	Ratio of the variance of the squeezed and antisqueezed quadratures	[4]
fidelity	Quantum Fidelity between the two gaussian states	[5]
coherence	Coherence of a multipartite gaussian state	[6]
number_operator_moments	Calculates means vector and covariance matrix of number operator	[3]
wigner	Wigner function over a 2D grid for a single mode gaussian state	[1]
q_function	Hussimi Q-function over a 2D grid for a single mode gaussian state	[3]
logarithmic_negativity	Logarithmic negativity for a bipartition of a gaussian state	[1]
duan_criteria	LHS of the Duan criteria for a bipartition of a multipartite gaussian state	[7]
measurement_general	Conditional state after a partial gaussian measurement	[8]
measurement_homodyne	Conditional state after a partial homodyne measurement	[8]
measurement_general	Conditional state after a partial heterodyne measurement	[8]
copy	Creates an identical copy	—

Table 2: Attributes of the gaussian_state class

Attributes	Description
R	Quadratures mean values
V	Covariance matrix
Omega	Symplectic form matrix
N_modes	Number of modes

2 gaussian_dynamics class

An instance of this class perform a time evolution on some initial gaussian_state according to a unconditional open quantum dynamics dictated by a set of quantum Langevin and Lyapunov equations.

Definitions

Langevin equation – We assume that the time evolution of the quadrature vector are dictated by a set of quantum Langevin equations of the form

$$\dot{\hat{\boldsymbol{X}}} = A(t)\hat{\boldsymbol{X}} + \hat{\boldsymbol{N}}\,,$$

where $A(t)$ is the drift matrix and D is the noise vector operator. We use take the expectation value of this equation to calculate the of the first moments of time evolved gaussian state dictated by

$$\dot{\boldsymbol{R}} = A(t)\boldsymbol{R} + \boldsymbol{N}\,,$$

where N is the mean noise vector.

Lyapunov equation – A direct consequence of the quantum Langevin equations above is that the time evolution of the covariance matrix of the initial state is given by a Lyapunov equation, with D the diffusion matrix,

$$\dot{V} = A(t)V + VA(t)^T + D\,,$$

Semi-classical Langevin equation – We may consider a semi-classical description of the system by performing a Monte Carlos simulation of the Langevin equations for the mean quadratures. At each iteration of this simulation, we perform a Euler-Maryuama integration of the semi-classical stochastic differential Langevin equation. The mean values of the noises are encompassed in $\boldsymbol{N}$ and its correlations (noise amplitude) are described in the diagonal elements of the diffusion matrix D. Analogously, the initial conditions are obtained from the mean values of the initial state $\boldsymbol{R}_0$ and the variances of their distribution in phase-space are given by the diagonal elements of its covariance matrix V_0.

Class properties

Class methods

In Table 4, we present the name and description of the methods of the gaussian_dynamics class except one: the class constructor. It has the same name of the class and must be called to create an instance of it: a variable of type gaussian_dynamics. There is only one way to call the constructor: passing as arguments the parameters for the time evolution described above:

1. A – the drift matrix
2. D – the diffusion matrix
3. $\boldsymbol{N}$ – the mean noise vector
4. Initial gaussian state (gaussian_state)

Table 3: Properties of the gaussian_dynamics class

Property	Description
A	Drift matrix
D	Diffusion matrix
N	Mean noise vector
t	Array with timestamps for the simulation
R	Array with mean quadratures for each timestamp
V	Cell with covariance matrix for each timestamp
state	Gaussian state for each timestamp
R_semi_classical	Array with semi-classical mean quadratures
is_stable	Boolean telling if the system is stable or not
N_time	Length of time array
Size_matrices	Size of covariance, diffusion and drift matrices

Table 4: Properties of the gaussian_dynamics class

Method	Description
unconditional_dynamics	Unconditional time evolution of an initial states according to quantum Langevin and Lyapunov equa
conditional_dynamics	Conditional time evolution of an initial states according to quantum Langevin and Lyapunov equati
steady_state	Calculates the steady state of the system
floquet	Approximates up to first order the steady state of a system with periodic Hamiltonian
semi_classical_dynamics	Semi-classical trajectories of the system following a Monte Carlo method for the Langevin equations
langevin_semi_classical	Solve the semi-classical Langevin equation for the time evolved mean quadratures
steady_state	Calculates the steady state gaussian_state (at the moment it only works for constant drift matrix)

2.1 Example of usage

Let us see a basic example of usage of the Toolbox. First, we need to import the classes described above alongside numpy

```
import numpy as np
from quantum_gaussian_toolbox import *
```

Consider a harmonic oscillator in a thermal state, we can easily define its gaussian state through

```
nbar_0   = 1.0577e+05;                          # Initial particle occupation number
initial_state = gaussian_state("thermal", nbar_0); # Initial state
```

We may apply gaussian unitaries to this state. Maybe we want to squeeze and rotate it in phase space, which are done through

```
initial_state.squeeze(3);                       # Squeeze gaussian state
initial_state.rotate(-pi/4);                    # Rotate gaussian state
```

Now that we have defined some states for our particle, we may wonder what will happen to is once it starts to evolve in time. We now define the dynamics for the particle

```
omega = 2*np.pi*197e+3;                         # Particle natural frequency [Hz]
gamma = 2*np.pi*881.9730;                       # Damping constant [Hz] at 1.4 mbar pressure
nbar_env = 3.1731e+07;                          # Environmental    occupation number

A = np.array([[    0   ,  +omega ],             # Drift matrix for harmonic potential
              [ -omega ,  -gamma ]]);

D = np.diag([0, 2*gamma*(2*nbar_env+1)]);       # Diffusion matrix
N = np.zeros((2,1));                            # Mean noise vector
```

Now we can proceed to perform the numerical simulation of the time evolution for some timestamps

```
t = np.linspace(0, 2*pi/omega, 1000);           # Timestamps for simulation

simulation = gaussian_dynamics(A, D, N, initial_state)# Create simulation instance!
states = simulation.run(t);                     # Simulate and retrieve time evolved states
```

The variable 'states' is an array of gassian_state with the time evolution of our system. From it we can use the methods described in Table 1 and study, for example, how the mean number of photon has evolved in time:

```
nbar = np.zeros(size(t));                          # Variable to store occupation numbers

for i in range(len(states)):                       # For each time evolved state,
    nbar[i] = states[i].occupation_number();       # calculate its occupation number
```

References

[1] Christian Weedbrook, Stefano Pirandola, Raúl García-Patrón, Nicolas J. Cerf, Timothy C. Ralph, Jeffrey H. Shapiro, and Seth Lloyd. Gaussian quantum information. *Rev. Mod. Phys.*, 84:621–669, May 2012.

[2] Alessio Serafini. *Quantum Continuous Variables: A Primer of Theoretical Methods.* CRC Press, jun 2017.

[3] V. V. Dodonov, O. V. Man'ko, and V. I. Man'ko. Multidimensional hermite polynomials and photon distribution for polymode mixed light. *Phys. Rev. A*, 50:813–817, Jul 1994.

[4] Ond řej Černotík and Radim Filip. Strong mechanical squeezing for a levitated particle by coherent scattering. *Phys. Rev. Research*, 2:013052, Jan 2020.

[5] Leonardo Banchi, Samuel L. Braunstein, and Stefano Pirandola. Quantum fidelity for arbitrary gaussian states. *Phys. Rev. Lett.*, 115:260501, Dec 2015.

[6] Jianwei Xu. Quantifying coherence of gaussian states. *Phys. Rev. A*, 93:032111, Mar 2016.

[7] Lu-Ming Duan, G. Giedke, J. I. Cirac, and P. Zoller. Inseparability criterion for continuous variable systems. *Phys. Rev. Lett.*, 84:2722–2725, Mar 2000.

[8] Jinglei Zhang. *Quantum measurement and preparation of Gaussian states.* PhD thesis, Department of Physics and Astronomy, Aarhus University, 2018.