Quantum version of the model in a trap

Lin Dong, September 1, 2015

We can write the 1D atom-cavity Hamiltonian in a hamonic trap as,

$$\begin{aligned}\mathcal{H}_{\text{eff}} &= \int dz \begin{pmatrix} \psi_\uparrow^\dagger(z) & \psi_\downarrow^\dagger(z) \end{pmatrix} \left[\frac{\hbar^2 k_z^2}{2m} + \frac{1}{2}m\omega^2 z^2 + \frac{\hbar^2}{m} q_r k_z \sigma_z + \delta\sigma_z\right] \begin{pmatrix} \psi_\uparrow(z) \\ \psi_\downarrow(z) \end{pmatrix} \\ &+ \int dz \frac{\Omega}{2}\psi_\uparrow^\dagger(z)\psi_\downarrow(z)c + \int dz \frac{\Omega}{2} c^\dagger \psi_\downarrow^\dagger(z)\psi_\uparrow(z) \\ &+ i\varepsilon_p(c^\dagger - c) - \hbar\delta_c c^\dagger c. \end{aligned} \tag{1}$$

Dissipation process is modeled by Liouvillean term $\mathcal{L}$ appearing in the master equation,

$$\dot{\rho} = \frac{1}{i\hbar}[\mathcal{H}_{\text{eff}}, \rho] + \mathcal{L}\rho \tag{2}$$

where

$$\mathcal{L}\rho = \kappa(2c\rho c^\dagger - c^\dagger c\rho - \rho c^\dagger c). \tag{3}$$

Then, we write the commutator explicitly as,

$$\begin{aligned}[\mathcal{H}_{\text{eff}}, \rho] &= \int dz \psi_\uparrow^\dagger(z)\left(\frac{k_z^2}{2m} + \frac{1}{2}m\omega^2 z^2 + \frac{q_r k_z}{m} + \delta\right)\psi_\uparrow(z)\rho - \int dz \rho\psi_\uparrow^\dagger(z)\left(\frac{k_z^2}{2m} + \frac{1}{2}m\omega^2 z^2 + \frac{q_r k_z}{m} + \delta\right)\psi_\uparrow(z) \\ &+ \int dz \psi_\downarrow^\dagger(z)\left(\frac{k_z^2}{2m} + \frac{1}{2}m\omega^2 z^2 - \frac{q_r k_z}{m} - \delta\right)\psi_\downarrow(z)\rho - \int dz \rho\psi_\downarrow^\dagger(z)\left(\frac{k_z^2}{2m} + \frac{1}{2}m\omega^2 z^2 - \frac{q_r k_z}{m} - \delta\right)\psi_\downarrow(z) \\ &+ \frac{\Omega}{2}\int dz \left(\psi_\uparrow^\dagger(z)\psi_\downarrow(z)c\rho + c^\dagger\psi_\downarrow^\dagger(z)\psi_\uparrow(z)\rho - \rho\psi_\uparrow^\dagger(z)\psi_\downarrow(z)c - \rho c^\dagger\psi_\downarrow^\dagger(z)\psi_\uparrow(z)\right) \\ &+ i\varepsilon_p\left(c^\dagger\rho - c\rho - \rho c^\dagger + \rho c\right) - \delta_c\left(c^\dagger c\rho - \rho c^\dagger c\right). \end{aligned}$$

To this end, we choose our basis states as $|n; q, \sigma\rangle$ where $n = 0, 1, 2, ...N$ and N is the truncation number of photon, $q = 0, 1, 2, ...Q$ and Q is the truncation number of harmonic oscillator energy levels and $\sigma = \uparrow, \downarrow$. Our goal is to calculate matrix elements of density operator under this basis $\langle m; p, \sigma|\rho|n; q, \sigma'\rangle \equiv \rho_{mn}^{p\sigma q\sigma'}$. Rules for creation and annilation operators are

$$\begin{aligned} c|n; q, \sigma'\rangle &= \sqrt{n}|n-1; q, \sigma'\rangle, \quad c^\dagger|n; q, \sigma'\rangle = \sqrt{n+1}|n+1; q, \sigma'\rangle \\ \langle m; q, \sigma|c &= \sqrt{m+1}\langle m+1; q, \sigma|, \langle m; q, \sigma|c^\dagger = \sqrt{m}\langle m-1; q, \sigma|. \end{aligned}$$

We write field operator $\psi_\sigma(z) = \sum_{q=1}^{Q}\varphi_q(z)a_{q\sigma}$ in second quantization, where $\varphi_q(z)$ is the eigenstate wavefunction of harmonic oscillator. Also, $k_z = -i\frac{\partial}{\partial z}$ serves as first quantization and only operates on wavefunction $\varphi_p(z)$. For arbitrary state, we have (where we have chosen trap unit by setting $\hbar = m = \omega = 1$)

$$\begin{aligned} \text{FirstTerm} \equiv & \langle m; p, \sigma| \textstyle\int dz \psi_\uparrow^\dagger(z)\left(\frac{k_z^2}{2} + \frac{1}{2}z^2 + q_r k_z + \delta\right)\psi_\uparrow(z)\rho|n; q, \sigma'\rangle \\ = & \langle m; p, \sigma| \textstyle\int dz \sum_{p'}\varphi_{p'}^*(z)a_{p'\uparrow}^\dagger\left(H_{\text{osc}} + \delta - iq_r\frac{\partial}{\partial z}\right)\sum_{q'}\varphi_{q'}(z)a_{q'\uparrow}\rho|n; q\sigma'\rangle \\ = & \textstyle\sum_{p'q'}\left[\int dz \varphi_{p'}^*(z)\left(H_{\text{osc}} + \delta - iq_r\frac{\partial}{\partial z}\right)\varphi_{q'}(z)\right]\langle m; p, \sigma|a_{p'\uparrow}^\dagger a_{q'\uparrow}\rho|n; q\sigma'\rangle \\ = & \textstyle\sum_{p'q'}\left[(q' + \frac{1}{2} + \delta)\int dz \varphi_{p'}^*(z)\varphi_{q'}(z) - iq_r\int dz \varphi_{p'}^*(z)\frac{\partial}{\partial z}\varphi_{q'}(z)\right]\langle m; p, \sigma|a_{p'\uparrow}^\dagger a_{q'\uparrow}\rho|n; q\sigma'\rangle \end{aligned}$$

Easy to have $\int dz \varphi_{p'}^*(z)\varphi_{q'}(z) = \delta_{p'q'}$ but more steps to follow for $-iq_r\int dz \varphi_{p'}^*(z)\frac{\partial}{\partial z}\varphi_{q'}(z)$. From

$$\varphi_q(z) = \mathcal{A}_q H_q(z)e^{-\frac{z^2}{2}}, \quad \mathcal{A}_q = \frac{1}{\sqrt{2^q q!}}\frac{1}{\pi^{1/4}}.$$

where $H_q(z)$ is the Hermite polynomials, we have

$$\begin{aligned}
-i\frac{\partial}{\partial z}\varphi_q(z) &= -i\mathcal{A}_q e^{-\frac{z^2}{2}}\left(H_q'(z) - zH_q(z)\right)\\
&= -i\mathcal{A}_q e^{-\frac{z^2}{2}}\left(2qH_{q-1}(z) - zH_q(z)\right)\\
&= -i\mathcal{A}_q e^{-\frac{z^2}{2}}\left(2qH_{q-1}(z) - \frac{1}{2}(H_{q+1}(z) + 2qH_{q-1}(z))\right)\\
&= -i\mathcal{A}_q e^{-\frac{z^2}{2}}\left(qH_{q-1}(z) - \frac{1}{2}H_{q+1}(z)\right)\\
&= -i\left(q\frac{\mathcal{A}_q}{\mathcal{A}_{q-1}}\varphi_{q-1}(z) - \frac{1}{2}\frac{\mathcal{A}_q}{\mathcal{A}_{q+1}}\varphi_{q+1}(z)\right)\\
&= -i\left(\sqrt{\frac{q}{2}}\varphi_{q-1}(z) - \sqrt{\frac{q+1}{2}}\varphi_{q+1}(z)\right)
\end{aligned}$$

where we have used recurrence relation for Hermite polynomials, including $H_n'(x) = 2nH_{n-1}(x)$ or $H_{n+1}(x) = 2xH_n(x) - H_n'(x)$ and $H_{n+1}(x) = 2xH_n(x) - 2nH_{n-1}(x)$. The above result is equivalent to replace $k_z = \frac{1}{\sqrt{2}i}(a - a^\dagger)$ and operate it on the wavefunction $\varphi_p(z)$, namely in one line we can prove,

$$\begin{aligned}
k_z\varphi_{q'}(z) &= \frac{1}{\sqrt{2}i}(a - a^\dagger)\varphi_{q'}(z)\\
&= -i\frac{1}{\sqrt{2}}\left(\sqrt{q'}\varphi_{q'-1}(z) - \sqrt{q'+1}\varphi_{q'+1}(z)\right)
\end{aligned}$$

Then, we should have $-iq_r\int dz\varphi_{p'}^*(z)\frac{\partial}{\partial z}\varphi_{q'}(z) = -iq_r\sqrt{\frac{1}{2}}\int dz\varphi_{p'}^*(z)(\sqrt{q'}\varphi_{q'-1}(z) - \sqrt{q'+1}\varphi_{q'+1}(z)) = -iq_r\sqrt{\frac{1}{2}}(\sqrt{q'}\delta_{p',q'-1} - \sqrt{q'+1}\delta_{p',q'+1})$. Also, note that

$$\begin{aligned}
\langle m; p, \sigma|a_{p'\uparrow}^\dagger a_{q'\uparrow}\rho|n; q\sigma'\rangle &= \delta_{p',p}\delta_{\sigma,\uparrow}\langle m; \mathrm{vac}|a_{q'\uparrow}\rho|n; q\sigma'\rangle\\
&= \delta_{p',p}\delta_{\sigma,\uparrow}\langle m; q'\uparrow|\rho|n; q\sigma'\rangle\\
&= \delta_{p',p}\delta_{\sigma,\uparrow}\rho_{mn}^{q'\sigma q\sigma'}
\end{aligned}$$

To avoid confusion, we have to point out one subtle difference between a and $a_{q\sigma}$, where a is the lowering operator for the harmonic oscillator state and $a_{q\sigma}$ is the atom number annhilation operator at oscillator state q and spin σ. Thus $a\varphi_q(z) = \sqrt{q}\varphi_{q-1}(z)$, but $a_{q\sigma}|n; p\sigma'\rangle = \delta_{q,p}\delta_{\sigma,\sigma'}|n; \mathrm{vac}\rangle$ and there is no $\sqrt{p}$ term as a prefactor! Wrapping it up, the first term is massaged into

$$\begin{aligned}
&\delta_{\sigma,\uparrow}\textstyle\sum_{p'q'}\left[\left(q' + \frac{1}{2} + \delta\right)\delta_{p'q'} - \frac{iq_r}{\sqrt{2}}\left(\sqrt{q'}\delta_{p',q'-1} - \sqrt{q'+1}\delta_{p',q'+1}\right)\right]\delta_{p',p}\rho_{mn}^{q'\sigma q\sigma'}\\
= \quad &\left(p + \tfrac{1}{2} + \delta\right)\rho_{mn}^{p\uparrow q\sigma'} - \tfrac{iq_r}{\sqrt{2}}\left(\sqrt{p+1}\rho_{mn}^{(p+1)\uparrow q\sigma'} - \sqrt{p}\rho_{mn}^{(p-1)\uparrow q\sigma'}\right)
\end{aligned}$$

and the rest of terms could be followed similarly.

$$-\langle m; p, \sigma|\int dz\rho\psi_\uparrow^\dagger(z)\left(\frac{k_z^2}{2m} + \frac{1}{2}m\omega^2z^2 + \frac{q_rk_z}{m} + \delta\right)\psi_\uparrow(z)|n; q, \sigma'\rangle = -\left(q + \frac{1}{2} + \delta\right)\rho_{mn}^{p\sigma q\uparrow} + \frac{iq_r}{\sqrt{2}}\left(\sqrt{q}\rho_{mn}^{p\sigma(q-1)\uparrow} - \sqrt{q+1}\rho_{mn}^{p\sigma(q+1)\uparrow}\right.$$

$$\langle m; p, \sigma|\int dz\psi_\downarrow^\dagger(z)\left(\frac{k_z^2}{2m} + \frac{1}{2}m\omega^2z^2 - \frac{q_rk_z}{m} - \delta\right)\psi_\downarrow(z)\rho|n; q, \sigma'\rangle = \left(p + \frac{1}{2} - \delta\right)\rho_{mn}^{p\downarrow q\sigma'} + \frac{iq_r}{\sqrt{2}}\left(\sqrt{p+1}\rho_{mn}^{(p+1)\downarrow q\sigma'} - \sqrt{p}\rho_{mn}^{(p-1)\downarrow q\sigma'}\right)$$

$$-\langle m; p, \sigma|\int dz\rho\psi_\downarrow^\dagger(z)\left(\frac{k_z^2}{2m} + \frac{1}{2}m\omega^2z^2 - \frac{q_rk_z}{m} - \delta\right)\psi_\downarrow(z)|n; q, \sigma'\rangle = -\left(q + \frac{1}{2} - \delta\right)\rho_{mn}^{p\sigma q\downarrow} - \frac{iq_r}{\sqrt{2}}\left(\sqrt{q}\rho_{mn}^{p\sigma(q-1)\downarrow} - \sqrt{q+1}\rho_{mn}^{p\sigma(q+1)\downarrow}\right.$$

$$\begin{aligned}
&\langle m; p, \sigma|\frac{\Omega}{2}\int dz\left(\psi_\uparrow^\dagger(z)\psi_\downarrow(z)c\rho + c^\dagger\psi_\downarrow^\dagger(z)\psi_\uparrow(z)\rho - \rho\psi_\uparrow^\dagger(z)\psi_\downarrow(z)c - \rho c^\dagger\psi_\downarrow^\dagger(z)\psi_\uparrow(z)\right)|n; q, \sigma'\rangle = \frac{\Omega}{2}\sum_{p'q'}\int dz\varphi_{p'}^*(z)\varphi_{q'}(z)\times\\
&\quad\left[\langle m; p, \sigma|a_{p'\uparrow}^\dagger a_{q'\downarrow}c\rho|n; q\sigma'\rangle + \langle m; p, \sigma|c^\dagger a_{p'\downarrow}^\dagger a_{q'\uparrow}\rho|n; q\sigma'\rangle - \langle m; p, \sigma|\rho a_{p'\uparrow}^\dagger a_{q'\downarrow}c|n; q\sigma'\rangle - \langle m; p, \sigma|\rho c^\dagger a_{p'\downarrow}^\dagger a_{q'\uparrow}|n; q\sigma'\rangle\right] =\\
&\quad\frac{\Omega}{2}\sum_{p'q'}\delta_{p'q'}\left[\sqrt{m+1}\delta_{pp'}\delta_{\sigma\uparrow}\rho_{(m+1)n}^{q'\downarrow q\sigma'} + \sqrt{m}\delta_{pp'}\delta_{\sigma\downarrow}\rho_{(m-1)n}^{q'\uparrow q\sigma'} - \sqrt{n}\delta_{qq'}\delta_{\sigma'\downarrow}\rho_{m(n-1)}^{p\sigma p'\uparrow} - \sqrt{n+1}\delta_{qq'}\delta_{\sigma'\uparrow}\rho_{m(n+1)}^{p\sigma p'\downarrow}\right] = \frac{\Omega}{2}\\
&\quad\sqrt{m+1}\delta_{\sigma\uparrow}\rho_{(m+1)n}^{p\downarrow q\sigma'} + \sqrt{m}\delta_{\sigma\downarrow}\rho_{(m-1)n}^{p\uparrow q\sigma'} - \sqrt{n}\delta_{\sigma'\downarrow}\rho_{m(n-1)}^{p\sigma q\uparrow} - \sqrt{n+1}\delta_{\sigma'\uparrow}\rho_{m(n+1)}^{p\sigma q\downarrow}
\end{aligned}$$

$$\langle m;p,\sigma|i\varepsilon_p\left(c^\dagger\rho - c\rho - \rho c^\dagger + \rho c\right) - \delta_c\left(c^\dagger c\rho - \rho c^\dagger c\right)|n;q,\sigma'\rangle = i\varepsilon_p\left(\sqrt{m}\rho^{p\sigma q\sigma'}_{(m-1)n} - \sqrt{m+1}\rho^{p\sigma q\sigma'}_{(m+1)n} - \sqrt{n+1}\rho^{p\sigma q\sigma'}_{m(n+1)} + \sqrt{n}\rho^{p\sigma q\sigma'}_{m(n-1)}\right) - \delta_c\left(m-n\right)\rho^{p\sigma q\sigma'}_{mn}$$

$$\kappa\langle m;p,\sigma|(2c\rho c^\dagger - c^\dagger c\rho - \rho c^\dagger c)|n;q,\sigma'\rangle = \kappa\left(2\sqrt{m+1}\sqrt{n+1}\rho^{p\sigma q\sigma'}_{(m+1)(n+1)} - (m+n)\rho^{p\sigma q\sigma'}_{mn}\right)$$

With above preparations, we write master equation Eq. 2 as,

$$\begin{aligned}
\frac{d}{dt}\rho^{p\sigma q\sigma'}_{mn} &= \frac{1}{i}\left[\left(p+\frac{1}{2}+\delta\right)\rho^{p\uparrow q\sigma'}_{mn} - \frac{iq_r}{\sqrt{2}}\left(\sqrt{p+1}\rho^{(p+1)\uparrow q\sigma'}_{mn} - \sqrt{p}\rho^{(p-1)\uparrow q\sigma'}_{mn}\right)\right] \\
&+ \frac{1}{i}\left[-\left(q+\frac{1}{2}+\delta\right)\rho^{p\sigma q\uparrow}_{mn} + \frac{iq_r}{\sqrt{2}}\left(\sqrt{q}\rho^{p\sigma(q-1)\uparrow}_{mn} - \sqrt{q+1}\rho^{p\sigma(q+1)\uparrow}_{mn}\right)\right] \\
&+ \frac{1}{i}\left[\left(p+\frac{1}{2}-\delta\right)\rho^{p\downarrow q\sigma'}_{mn} + \frac{iq_r}{\sqrt{2}}\left(\sqrt{p+1}\rho^{(p+1)\downarrow q\sigma'}_{mn} - \sqrt{p}\rho^{(p-1)\downarrow q\sigma'}_{mn}\right)\right] \\
&+ \frac{1}{i}\left[-\left(q+\frac{1}{2}-\delta\right)\rho^{p\sigma q\downarrow}_{mn} - \frac{iq_r}{\sqrt{2}}\left(\sqrt{q}\rho^{p\sigma(q-1)\downarrow}_{mn} - \sqrt{q+1}\rho^{p\sigma(q+1)\downarrow}_{mn}\right)\right] \\
&+ \frac{\Omega}{2}\frac{1}{i}\left[\sqrt{m+1}\delta_{\sigma\uparrow}\rho^{p\downarrow q\sigma'}_{(m+1)n} + \sqrt{m}\delta_{\sigma\downarrow}\rho^{p\uparrow q\sigma'}_{(m-1)n} - \sqrt{n}\delta_{\sigma'\downarrow}\rho^{p\sigma q\uparrow}_{m(n-1)} - \sqrt{n+1}\delta_{\sigma'\uparrow}\rho^{p\sigma q\downarrow}_{m(n+1)}\right] \\
&+ \varepsilon_p\left(\sqrt{m}\rho^{p\sigma q\sigma'}_{(m-1)n} - \sqrt{m+1}\rho^{p\sigma q\sigma'}_{(m+1)n} - \sqrt{n+1}\rho^{p\sigma q\sigma'}_{m(n+1)} + \sqrt{n}\rho^{p\sigma q\sigma'}_{m(n-1)}\right) \\
&+ i\delta_c\left(m-n\right)\rho^{p\sigma q\sigma'}_{mn} + \kappa\left(2\sqrt{m+1}\sqrt{n+1}\rho^{p\sigma q\sigma'}_{(m+1)(n+1)} - (m+n)\rho^{p\sigma q\sigma'}_{mn}\right)
\end{aligned} \tag{4}$$

Following the previous work, we define

$$[\rho^{p\sigma q\sigma'}_{mn}]_{(2N+2)(Q+1)\times(2N+2)(Q+1)} = \begin{pmatrix} [\rho^{p\uparrow q\uparrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} & [\rho^{p\uparrow q\downarrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} \\ [\rho^{p\downarrow q\uparrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} & [\rho^{p\downarrow q\downarrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} \end{pmatrix} \tag{5}$$

We columnize the matrix array by array. For instance, in the column of $[\rho^{p\uparrow q\uparrow}_{mn}]_{(N+1)^2Q^2\times 1}$, the kth element is accessed as $k = m(N+1)(Q+1)^2 + n(Q+1)^2 + p(Q+1) + q + 1$ where $m,n = 0,1,2,...N$ and $p,q = 0,1,2,...Q$. We then further write EOM of density matrix as

$$[\rho] = \begin{pmatrix} [\rho^{p\uparrow q\uparrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} \\ [\rho^{p\uparrow q\downarrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} \\ [\rho^{p\downarrow q\uparrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} \\ [\rho^{p\downarrow q\downarrow}_{mn}]_{(N+1)(Q+1)\times(N+1)(Q+1)} \end{pmatrix}_{(2N+2)^2(Q+1)^2\times 1} \tag{6}$$

$$\frac{d}{dt}[\rho] = \begin{pmatrix} [M^{p\uparrow q\uparrow}_{mn}] & [S^{1pq}_{mn}] & [S^{2pq}_{mn}] & 0 \\ [S^{3pq}_{mn}] & [M^{p\uparrow q\downarrow}_{mn}] & 0 & [S^{4pq}_{mn}] \\ [S^{5pq}_{mn}] & 0 & [M^{p\downarrow q\uparrow}_{mn}] & [S^{6pq}_{mn}] \\ 0 & [S^{7pq}_{mn}] & [S^{8pq}_{mn}] & [M^{p\downarrow q\downarrow}_{mn}] \end{pmatrix}_{(2N+2)^2(Q+1)^2\times(2N+2)^2(Q+1)^2} [\rho] \tag{7}$$

We denote the RHS matrix of Eq. 7 as A in the code, and we access the element (i,j) as $i = (N+1)^2(Q+1)^2 r + m(N+1)(Q+1)^2 + n(Q+1)^2 + p(Q+1) + q + 1$ (Note: MATLAB index starts from 1 and C++ code index starts from 0, so attention should be paid here for the final value of 1. No such value is needed in C++) where $r,c = 0,1,2,3$, $m,n = 0,1,2,...N$ and $p,q = 0,1,2,...Q$

To benchmark the results, we first consider the case without pumping and decay, with zero photon inside the cavity and atom being the excited state.

- If we further set $q_r = 0$, then orbital degree becomes a good quantum number. In this simplified case, we have the small subspace $\{|n, q, \sigma\rangle\}$ where $n = 0, 1$ and $\sigma = \uparrow, \downarrow$, q is only a constant number. We use simplified notation of density operator and write the master equation as,

$$\begin{aligned}
\frac{d}{dt}\rho_{mn}^{q\sigma q\sigma'} &= \frac{1}{i}\left[\left(q+\frac{1}{2}+\delta\right)\rho_{mn}^{q\uparrow q\sigma'} - \left(q+\frac{1}{2}+\delta\right)\rho_{mn}^{q\sigma q\uparrow} + \left(q+\frac{1}{2}-\delta\right)\rho_{mn}^{q\downarrow q\sigma'} - \left(q+\frac{1}{2}-\delta\right)\rho_{mn}^{q\sigma q\downarrow}\right] \\
&+ \frac{\Omega}{2}\frac{1}{i}\left[\sqrt{m+1}\delta_{\sigma\uparrow}\rho_{(m+1)n}^{p\downarrow q\sigma'} + \sqrt{m}\delta_{\sigma\downarrow}\rho_{(m-1)n}^{p\uparrow q\sigma'} - \sqrt{n}\delta_{\sigma'\downarrow}\rho_{m(n-1)}^{p\sigma q\uparrow} - \sqrt{n+1}\delta_{\sigma'\uparrow}\rho_{m(n+1)}^{p\sigma q\downarrow}\right]
\end{aligned}$$

  Due to conservation of excitation number, the Hilbert space is further reduced to the smaller subspace. For instance, if we have $|N-1, q, \uparrow\rangle$ as initial state, then we would only have population in another state $|N, q, \downarrow\rangle$ and these two states lead to four elements of density matrix and in solving the master equation, we are dealing with 4 by 4 ODE time evolution for $N = 1$,

$$\frac{d}{dt}\begin{pmatrix} \rho_{00}^{q\uparrow q\uparrow} \\ \rho_{01}^{q\uparrow q\downarrow} \\ \rho_{10}^{q\downarrow q\uparrow} \\ \rho_{11}^{q\downarrow q\downarrow} \end{pmatrix} = \begin{pmatrix} 0 & -\frac{\Omega}{2i} & \frac{\Omega}{2i} & 0 \\ -\frac{\Omega}{2i} & \frac{2\delta}{i} & 0 & \frac{\Omega}{2i} \\ \frac{\Omega}{2i} & 0 & -\frac{2\delta}{i} & -\frac{\Omega}{2i} \\ 0 & \frac{\Omega}{2i} & -\frac{\Omega}{2i} & 0 \end{pmatrix}\begin{pmatrix} \rho_{00}^{q\uparrow q\uparrow} \\ \rho_{01}^{q\uparrow q\downarrow} \\ \rho_{10}^{q\downarrow q\uparrow} \\ \rho_{11}^{q\downarrow q\downarrow} \end{pmatrix} \tag{8}$$

  and for arbitrary $N \geq 1$,

$$\frac{d}{dt}\begin{pmatrix} \rho_{N-1,N-1}^{q\uparrow q\uparrow} \\ \rho_{N-1,N}^{q\uparrow q\downarrow} \\ \rho_{N,N-1}^{q\downarrow q\uparrow} \\ \rho_{N,N}^{q\downarrow q\downarrow} \end{pmatrix} = \begin{pmatrix} 0 & -\frac{\Omega}{2i}\sqrt{N} & \frac{\Omega}{2i}\sqrt{N} & 0 \\ -\frac{\Omega}{2i}\sqrt{N} & \frac{2\delta}{i} & 0 & \frac{\Omega}{2i}\sqrt{N} \\ \frac{\Omega}{2i}\sqrt{N} & 0 & -\frac{2\delta}{i} & -\frac{\Omega}{2i}\sqrt{N} \\ 0 & \frac{\Omega}{2i}\sqrt{N} & -\frac{\Omega}{2i}\sqrt{N} & 0 \end{pmatrix}\begin{pmatrix} \rho_{N-1,N-1}^{q\uparrow q\uparrow} \\ \rho_{N-1,N}^{q\uparrow q\downarrow} \\ \rho_{N,N-1}^{q\downarrow q\uparrow} \\ \rho_{N,N}^{q\downarrow q\downarrow} \end{pmatrix} \tag{9}$$

- Also, we can consider case when $q_r \neq 0$ but $\Omega = 0$. In this case, excitation number is still a good quantum number but different oscillator q is coupled. The master equation is written as,

$$\begin{aligned}
\frac{d}{dt}\rho_{00}^{p\uparrow q\uparrow} &= \frac{1}{i}\left[\left(p+\frac{1}{2}+\delta\right)\rho_{00}^{p\uparrow q\uparrow} - \frac{iq_r}{\sqrt{2}}\left(\sqrt{p+1}\rho_{00}^{(p+1)\uparrow q\uparrow} - \sqrt{p}\rho_{00}^{(p-1)\uparrow q\uparrow}\right)\right] \\
&+ \frac{1}{i}\left[-\left(q+\frac{1}{2}+\delta\right)\rho_{00}^{p\uparrow q\uparrow} + \frac{iq_r}{\sqrt{2}}\left(\sqrt{q}\rho_{00}^{p\uparrow(q-1)\uparrow} - \sqrt{q+1}\rho_{00}^{p\uparrow(q+1)\uparrow}\right)\right]
\end{aligned}$$

  where we only need to consider Hilbert space spanned by $|0, q, \uparrow\rangle$ and $q = 0, 1, 2, ..., Q$ (if we choose initial state as zero photon and up spin and lowest oscillator state)

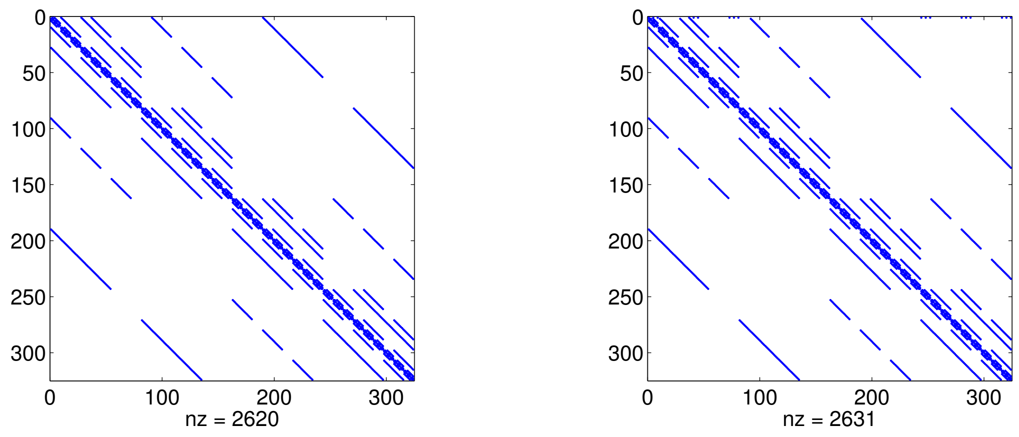

Fig. 1: Sparse view of the matrix pattern, before and after adding the normalization condition.

Algorithm 1 Mapping between global columnized vector and block and sub-block form of density operator

Here, we show the correspondence used in C++ code. Denote irow $= r(N+1)^2(Q+1)^2 + k$ and $k = m(N+1)(Q+1)^2 + n(Q+1)^2 + p(Q+1) + q$:

1. r runs from 0 to 3, corresponding to $\rho^{\uparrow\uparrow}$, $\rho^{\uparrow\downarrow}$,$\rho^{\downarrow\uparrow}$,$\rho^{\downarrow\downarrow}$. $r = \text{floor}(\frac{\text{irow}}{(N+1)^2(Q+1)^2})$;
2. $k = \text{irow} - r$;
3. $m = \text{floor}(\frac{k}{(N+1)(Q+1)^2})$;
4. $n = \text{floor}(\frac{k-m(N+1)(Q+1)^2)}{(Q+1)^2})$;
5. $p = \text{floor}(\frac{k-m(N+1)(Q+1)^2-n(Q+1)^2}{Q+1})$;
6. $q = k - m(N+1)(Q+1)^2 - n(Q+1)^2 - p(Q+1)$.

The rule behind this mapping is, we

1. arrange density matrix written in Eq. 5 as block rectangular form of $[\rho^{\uparrow\uparrow}, \rho^{\uparrow\downarrow},\rho^{\downarrow\uparrow},\rho^{\downarrow\downarrow}]^T$;
2. loop over photon number index m and $n$from 0 to N as a row major priority;
3. loop over orbital number index p and q from 0 to Q as a row major priority.

Note, for MATLAB code, row index starts from 1, so we need to add one to the q value.

Eq. 7 together with the normalization condition $\text{Tr}[\rho] = 1$ makes the steady state solution $\frac{d}{dt}\rho(t) = 0$ an over-determined problem. What we do in the MATLAB and C++ code is to put the first row of matrix A satisfy the condition $\text{Tr}[\rho] = 1$, which is explicitly written as $\sum_{m=0}^{N} \sum_{p=0}^{Q} \sum_{\sigma=\pm} \rho_{mm}^{p\sigma p\sigma} = 1$. What we have from Master Equation derivation in Eq. 4 is to relate every index in matrix $\rho_{mn}^{p\sigma q\sigma'}$ with their "neighering part", such as $\rho_{m(n+1)}^{p\sigma q\sigma'}$ etc. It is then natural to take advantage of the zeros in the matrix and their location. To exploit sparsity in solveing the general sparse linear system is to be economical in terms of both storage and computational effort. To visualize the sparse pattern of the matrix in Fig. 1, we use spy() function in MATLAB to view a matrix with cutoff chosen as $N = 2$ and $Q = 2$ before and after imposing the normalization condition.

Given the relation written in Eq. 4, we need to convert the language into C++ array form, which turns out to be a nontrivial task (total of two weeks coding/testing/debugging time). In short, it is essential to decompose a given row index in columnized vector Eq. 6 into correct corresponding $\{m, n, p, q, \sigma\}$ values. To achieve this goal, we use floor function (By definition, floor(x) Rounds x downward, returning the largest integral value that is not greater than x.) in both MATLAB and C++ in Algorithm 1. It can be checked straightforwardly, that the first index is 0(1) for $m = n = p = q = 0$ in C++(MATLAB) code, and the last index is $N(N+1)(Q+1)^2 + N(Q+1)^2 + Q(Q+1) + Q = 4(N+1)^2(Q+1)^2 - 1$ for C++ code or $N(N+1)(Q+1)^2 + N(Q+1)^2 + Q(Q+1) + Q + 1 = 4(N+1)^2(Q+1)^2$ for MATLAB code, which "fits right into the box".

In the parallized C++ code, we distribute the long columnized vector (size of $4(N+1)^2(Q+1)^2$ by 1) of density operator into hundreds or even thousands cores of CPU, and each CPU communicates with one another collectively via

Message Passing Interface (MPI) to handle the portable and scalable large-scale parallel sparse linear algebra application. Of course, the matrix construction and operations are interfaced with PETSc, which is "a suite of data structures and routines for the scalable (parallel) solution of scientific applications modeled by partial differential equations".

We first developed MATLAB code with dense matrix construction, which is inherited from previous published work [Atoms 2015, 3(2), 182-194, http://www.mdpi.com/2218-2004/3/2/182], and benchmarked the physics in limiting cases where we know it has to work based on Mean Field Theory (which is a lot less computationally intensive). Then we are motivated to develop high performance C++ code by the fact that in the MATLAB prototype of code, it can only handle at most the cutoff value of $N = 5$ and $Q = 5$ in our 12GB and 6-cores workstation. Then we researched the open source market of available library solvers, and found out PETSc stands out as one of the best platform to deliver high performance solver and preconditioner algorithms.

For matrix from small to intermediate size, the matrix can be decomposed into lower and upper (LU) triangular decomposition, leading to direct-solver techniques. The advantage of the direct-solver approach is that once the decomposition is performed, it is efficiently to solve for multiple parameters using forward and backward substitutions, if the sparse pattern of the matrix remains unchanged. However, the time and memory complexities of LU factorization and its limited scalability on large-scale distributed memory platforms prevent use of the approach for large-scale problems.

Among all of these possible approaches, the iterative-solver approach theoretically has the best time complexity (here, "complexity" denotes how the computational cost of an algorithm grows with the size of the computational domain) if the number of iterations is independent of the physical parameters. For problems of substantial size, preallocation of matrix memory is crucial for attaining good performance. As user of PETSc, we do not need to preallocate exactly the correct amount of space, as long as a sufficiently close estimate is given, the high efficiency for matrix assembly will remain. We form one vector from scratch, duplicate as needed and let PETSc decide how many elements of the vector are stored on each processor, by identifying the starting and ending mesh points on each processor for the interior part of the mesh. Scaling up, we need to deploy the sparse format of the matrix and thus, a map that links back to dense matrix form for debugging purpose. Following Compressed Row Storage (CRS) convention, we pass in "local" size of the created vector to access each row of the sparse matrix and force the same parallel layout as the vector created above. Using PETSc interface, the matrix format can be specified at runtime. The linear system is distributed across the processors by chunks of contiguous rows, which correspond to contiguous sections of the mesh on which the problem is discretized. In the PETSc partitioning scheme, all the matrix columns are local while a number of global rows are distributed to different processors, which can be decided by PETSc automatically. For matrix assembly, each processor contributes entries for the part that it owns locally. In doing this, we loop over the global row index that is owned by the local processor, and assign nonzero values at computed column indices according to Eq. 7 and Algorithm 1. The matrix construction is essentially done in parallel and has been tested to achieve good efficiency.

After the code is finished with matrix construction, we deploy the object-oriented strategy used in PETSc linear solver context and set various options by default and/or at runtime. Most of the existing practical iterative techniques for solving large linear system of equations utilize a projection process in one way or another. A projection process represents a canonical way of extracting an approximation to the solution of a linear system from a subspace. The matrix we concern about is non-hermitian, and thus has strongly indefinite eigen-spectrum, which is known to be difficult to converge for Krylov iterative solvers without appropriately choosing preconditioners. It is possible to convert a non-hermitian matrix into a hermition one in theory. One such technique solves the equivalent linear system $A^\dagger Ax = A^\dagger b$ which is called the normal equation, where $\dagger$ denotes complex conjugate transposition. The coefficient matrix $A^\dagger A$ immediately becomes symmetric positive definite (SPD) and one can exploit highly successful Conjugate Gradient or Lanczos solvers. However, in practice, the coefficient matrix $A^\dagger A$ becomes much denser and less well-conditioned than the original matrix A. The general consensus is that solving the normal equations can be an inefficient approach in the case when A is poorly conditioned, but perhaps only in the extreme case when A is unitary, i.e. $A^\dagger A = I$ then the normal equation is clearly the best approach (the Conjugate Gradient method will converge in zero step).

We turn our major attention to the proper choice of combination in Krylov subspace solvers and preconditioning matrix. Preconditioning is a key ingredient for the success of Krylov subspace methods in most cases, since the lack of robustness is a widely recognized weakness of iterative solvers relative to direct solvers. Indeed, finding a good preconditioner to solve a given sparse linear system is often viewed as a combination of art and science.