# QuEEN: Instruction Manual

Thomas Pope

July 23, 2015

# Contents

## 1 Introduction to QuEEN (***Qu**antum **E**volution of **E**xcitations in **N**etworks*)

QuEEN takes an $N \times N$ network Hamiltonian and uses it to solve the GKSL (Gorini, Kossakowski, Sudarshan an Lindblad) master equation [1–4] using the Monte-Carlo Quantum-Trajectories algorithm [5–8] in the presence of dissipative noise and stochastic and deterministic forms of dephasing. In the Monte-Carlo Quantum-Trajectories algorithm, the density matrix is decomposed into a wavefunction and evolved using the Schrödinger Equation, which is solved using a Runge-Kutta algorithm [9–11]. In addition, we employ decay processes between sites within the network and between network sites and an external site - the *sink*. QuEEN is parallel, so the trajectories are evenly distributed over a set of threads.

In addition, QuEEN can also add various forms of dephasing to the network site, both stochastic dephasing (static Gaussian noise, white noise [12, 13], filtered noise [14], and an Ornstein Uhlenbeck process [15–17]) and deterministic dephasing (plane wave signal, square wave signal, and FM signal). In the case of Static Noise and the Ornstein Uhlenbeck process, it is possible to allow correlation between sites. What follows is a list of the dephasing processes and a brief description of their function.

| | |
|---|---|
| Static Gaussian Noise | A random number is produced with a Gaussian distribution of a given width and added to the specified diagonal element of the Hamiltonian at the beginning of each trajectory. |
| White Noise | A continuous Markovian noise source of given rate is added to the specified site. This is done using the Master Equation formulation rather than adding the classical noise source directly to the Hamiltonian. |
| Filtered Noise | A discrete white noise process is generated and filtered through an on-the-fly finite impulse response (FIR) filter to suppress chosen frequencies in the signal for each specified diagonal element of the Hamiltonian. |
| Ornstein Uhlenbeck Process | An Ornstein Uhlenbeck process is generated for each specified diagonal element of the Hamiltonian with a given noise width and correlation time. |
| Plane Wave Signal | A sine wave is applied to each specified diagonal element of the Hamiltonian with a given frequency and amplitude. |
| Square Wave Signal | A square wave is applied to each specified diagonal element of the Hamiltonian with a given frequency and amplitude. The square wave is generated from a sequence of plane wave and the number of plane waves used is another variable. |
| FM signal | A frequency modulated signal is generated for each specified diagonal element of the Hamiltonian with given Amplitude, base frequency, signal frequency, and peak deviation. |

Additionally, the $QA$ algorithm allows the addition of a pulse matrix with a phase correction for the study of the complete STIRAP Hamiltonian.

QuEEN outputs the density matrix and site populations at each chosen time step. Optionally, the code can also output purity calculations and a convergence test that compares the cumulative change to the density matrix.

## 2 Monte-Carlo Jump Algorithm

We consider a single excitation propagating in a network of N sites. The network is given by the Hamiltonian,

$$\mathcal{H} = \sum_{i \neq j}^{N} \hbar \Delta_{ij} \, |i\rangle\langle j| + \sum_{i}^{N} \hbar \varepsilon_i \, |i\rangle\langle i|, \tag{2.1}$$

where $|i\rangle$ represents the excitation (e.g. an electron-hole pair) localized on the $i$-th site, $\varepsilon_i$ being the local site energies. Inter-site coupling, due for instance to electric dipole interaction, leads to tunnelling of excitations where $\Delta_{ij}$ is the tunnelling amplitude between sites $j$ and $i$. This input is given in the *.hamiltonian* file.

The dynamics of the density operator are evaluated using the GKSL (Gorini, Kossakowski, Sudarshan an Lindblad) master equation:

$$\dot{\rho}\left(t\right) = \overbrace{-\frac{i}{\hbar}\left[\mathcal{H}, \rho\left(t\right)\right]}^{\text{Hamiltonian component}} \underbrace{-\sum_{k=1}^{M} \frac{1}{2}\left\{\mathcal{L}_k^\dagger \mathcal{L}_k, \rho\left(t\right)\right\} + \sum_{k=1}^{M} \mathcal{L}_k \rho\left(t\right) \mathcal{L}_k^\dagger}_{\text{Dissipative component}}, \tag{2.2}$$

where,

$$\mathcal{L}_k = \sqrt{\Gamma_{ij}} \, |i\rangle\langle j| \tag{2.3}$$

and $\Gamma_{ij}$ is the decay rate from site $j$ to site $i$. This input is given in the *.dissipator* file.

By substituting an effective Hamiltonian,

$$\mathcal{H}_{\mathbf{eff}} = \mathcal{H} - i\frac{\hbar}{2}\sum_k \mathcal{L}_k^\dagger \mathcal{L}_k, \tag{2.4}$$

we arrive at adjusted notation:

$$\dot{\rho}\left(t\right) = -\frac{i}{\hbar}\left(\mathcal{H}_{\mathbf{eff}}\rho\left(t\right) - \rho\left(t\right)\mathcal{H}_{\mathbf{eff}}^\dagger\right) + \sum_k \mathcal{L}_k \rho\left(t\right) \mathcal{L}_k^\dagger. \tag{2.5}$$

So,

$$\begin{aligned}
\rho\left(t_0 + dt\right) &= \rho\left(t_0\right) - \frac{i}{\hbar}\left(\mathcal{H}_{\mathbf{eff}}\rho\left(t_0\right) - \rho\left(t_0\right)\mathcal{H}_{\mathbf{eff}}^\dagger\right)dt + \sum_k \mathcal{L}_k \rho\left(t_0\right)\mathcal{L}_k^\dagger dt \\
&\approx \overbrace{\left(\mathcal{I} - \frac{i}{\hbar}\mathcal{H}_{\mathbf{eff}}dt\right)\rho\left(t_0\right)\left(\mathcal{I} + \frac{i}{\hbar}\mathcal{H}_{\mathbf{eff}}^\dagger dt\right)}^{\text{Evolution component}} + \underbrace{\sum_k \mathcal{L}_k \rho\left(t_0\right)\mathcal{L}_k^\dagger dt}_{\text{Jump component}}. \qquad (2.6)
\end{aligned}$$

From the jump component, it is possible to get the jump operators - generally defined as $\mathcal{L}_k = A\,|\phi_k\rangle\,\langle\phi\left(t_0\right)|$, where $A$ is a normalising factor; i.e. they move a state, $\phi\left(t_0\right)$, into a new state $\phi_k$. If a jump occurs with probability $dp_k$, then,

$$\sum_k \frac{\mathcal{L}_k\,|\phi\left(t_0\right)\rangle\,\langle\phi\left(t_0\right)|\,\mathcal{L}_k^\dagger dt}{dp_k} = \sum_k |\phi_k\rangle\,\langle\phi_k| \Rightarrow |\phi_k\rangle = \frac{\mathcal{L}_k\,|\phi\left(t_0\right)\rangle\sqrt{dt}}{\sqrt{dp_k}}, \tag{2.7}$$

where

$$\mathcal{L}_k = \sqrt{\frac{dp_k}{dt}}\,|\phi_k\rangle\,\langle\phi\left(t_0\right)|\,. \tag{2.8}$$

and

$$dp_k = \Gamma_{ij} \left|\langle j|\phi\left(t_0\right)\rangle\right|^2 dt. \tag{2.9}$$

And if no jump occurs, then, from the evolution component,

$$\frac{\left(\mathcal{I} - \frac{i}{\hbar}\mathcal{H}_{\mathbf{eff}}dt\right)\left|\phi\left(t_0\right)\right\rangle\left\langle\phi\left(t_0\right)\right|\left(\mathcal{I} + \frac{i}{\hbar}\mathcal{H}^{\dagger}_{\mathbf{eff}}dt\right)}{1 - \sum\limits_k dp_k} = \left|\phi_0\right\rangle\left\langle\phi_0\right| \Rightarrow \left|\phi_0\right\rangle = \frac{\left(\mathcal{I} - \frac{i}{\hbar}\mathcal{H}_{\mathbf{eff}}dt\right)\left|\phi\left(t_0\right)\right\rangle}{\sqrt{1 - \sum\limits_k dp_k}}. \tag{2.10}$$

Employing the Monte Carlo wave function approach, a random number, $\epsilon$, is generated from a uniform distribution within the interval $[0, 1)$. If $\epsilon < \sum_k dp_k$, a jump occurs. The jump identity, $\mu$, is chosen such that $\sum_k^{\mu} dp_k < \epsilon < \sum_k^{\mu+1} dp_k$. In this case, the wavefunction becomes $|\phi_\mu\rangle$. Otherwise, it becomes $|\phi_0\rangle$.

Updating the density matrix after each step would require $\mathcal{O}\left(N^3\right)$ operations, where $N$ is the degrees of freedom of the Hamiltonian. Whereas, updating the wavefunction only requires $\mathcal{O}\left(N^2\right)$ operations. Summing over a range of $\mathcal{N}$ trajectories allows us to recover the probabilities obtained with the density operator. Hence, to obtain the same results requires $\mathcal{N} \times \mathcal{O}\left(N^2\right)$ operations. However, this technique is implicitly parrallelizable.

The wavefunction of the system, $|\Phi\left(t\right)\rangle$, can be described by the basis set, $|j\rangle$:

$$|\Phi\left(t\right)\rangle = \sum_{j=0}^{N} C_j\left(t\right)|j\rangle, \tag{2.11}$$

where $C_j$ are coefficients and $|j\rangle$ represent each excited state, $j = 1 \rightarrow N$, and the ground state $|0\rangle$. This can be substituted into the Schrödinger Equation, giving,

$$i\dot{C}_k\left(t\right) = \sum_{j=0}^{N} \langle k|\,\mathcal{H}_{\mathbf{eff}}\,|j\rangle\, C_j\left(t\right), \tag{2.12}$$

which, after substituting Eq. 2.4, yields,

$$i\dot{C}_k\left(t\right) = \sum_{j=0}^{N} \mathcal{H}_{jk}C_j\left(t\right) - i\frac{1}{2}\Gamma_k C_k\left(t\right), \tag{2.13}$$

where

$$\Gamma_k = \sum_{l=1}^{N} \Gamma_{lk}, \tag{2.14}$$

and $\Gamma_{lk}$ represents the probability of a jump from state $|k\rangle$ to state $|l\rangle$.

Using this notation, the density matrix can be decomposed to a set of wavefunction trajectories and recombined at the end of the simulation.

## 3 Compiling

To compile, copy the source code into a directory. A complete list of the source code files and a description of the contents follows,

1. **QuEEN.c**: Parent file, reads the command-line instructions and initializes the calculation
2. **Global-variables.h**: Declares all of the variables that a not thread-specific.
3. **jump_and_runge.h**: The main body of the code with the Monte-Carlo algorithm, Runge-Kutta algorithm, and all the dephasing processes. This is the parallel section of the calculation and contains all the thread-specific variables.
4. **read_input.h**: Reads the essential input files and any files that have been called from the command line using the flags (described in the next section).
5. **service_routines.h**: Misillaneous routines for printing output, declaring some matrices, etc.

Open a terminal in that directory and type: **gcc QuEEN.c -O3 -lm -fopenmp -o QuEEN**

1. **-O3**: Optimization flag. Purely optional, but recommended.
2. **-lm**: Necessary for the complex mathematics.
3. **-fopenmp**: Opens the parallel environment.
4. **-o QuEEN**: Defines an output location for the executable.

Once the code is compiled, you can generate an alias in your bash environment by adding the following line to the file ∼**/.bashrc**:

```
alias QuEEN='/path.../QuEEN'
```

Where the path is the location of the executable. Alternatively, you can call the executable explicitly every time you use it.

Once the input files, which are explain later, have been generated, for example:

1. STIRAP.hamiltonian
2. STIRAP.dissipator
3. STIRAP.initcon

to run the code in the terminal, write,

```
./QuEEN -label STIRAP
```

Further calculation options are available, eg.

```
./QuEEN -label STIRAP -sample 10000 -tmax 10 -fg 0.001 -cg 0.1 -wn -threads 4
```

These other options are explained below.

## 4 Command Line Flags

The command line flags are listed below. Some flags should be followed with input variables, which are identified in these instructions as $i, $f, and $c - integer, float, and character variables respectively.

**Essential:**

| | |
|---|---|
| *-label* $c | The system label used by all the input/output files |

**Optional:**

| | |
|---|---|
| *-sample* $i | The number of quantum trajectories used (default: 10,000) |
| *-tmax* $i | Length of time each trajectory is run for (default: 10) |
| *-coarse_grid* $f<br>*-cg* $f | Time step in the coarse grid used for the output data (default: 1) |
| *-fine_grid* $f<br>*-fg* $f | Time step in the fine grid used for the calculation (default: 0.01) |
| *-threads* $i | Number of cpu threads to be used. If 0, it will use everything it can find (default: 1) |
| *-white_noise*<br>*-wn* | Add pure dephasing using the *.wnd file - Markovian |
| *-static_noise*<br>*-sn* | Add pure dephasing using the *.snd file - Static |
| *-correlated_static_noise*<br>*-csn* | Add pure dephasing using the *.snd & *.sitecorr files - Correlated Static |
| *-ornstein_uhlenbeck*<br>*-ou* | Add pure dephasing using the *.oud file - Ornstein-Uhlenbeck |
| *-correlated_ornstein_uhlenbeck*<br>*-cou* | Add pure dephasing using the *.oud & *.sitecorr files - Correlated Ornstein-Uhlenbeck |
| *-filtered_noise*<br>*-fn* | Add pure dephasing using the *.fnd file - Filtered |
| *-plane_wave*<br>*-pw* | Add pure dephasing using the *.pwd file - Plane wave deterministic signal |
| *-square_wave*<br>*-sq* | Add pure dephasing using the *.sqd file - Square wave deterministic signal |
| *-frequency_modulated*<br>*-fm* | Add pure dephasing using the *.fmd file - Frequency modulated deterministic signal |
| *-qa* | Employ the QA algorithm using the *.qa_a and *.qa_q files |

| | |
|---|---|
| *-purity* | Calculate the purity of the states at each point on the coarse grid - output into *.purity file |
| *-convergence*<br>*-con* | Calculate the relative convergence of the site population at each point on the coarse grid - output into *.convergence file |

# 5 Input Files

## 5.1 Essential Input

### 5.1.1 Hamiltonian

{label}.hamiltonian
$n$
i  j  $\langle i|\,\mathcal{H}\,|j\rangle$
$\vdots$  $\vdots$  $\vdots$

File contains the Hamiltonian of the isolated system with the shown format, where $n$ is the number of excited states in the system and $\mathcal{H}$ is the Hamiltonian.

### 5.1.2 Dissipation

{label}.dissipator
$n$
i  j  $\Gamma_{ij}$
$\vdots$  $\vdots$  $\vdots$

File contains the dissipative component of the GKSL (Gorini, Kossakowski, Sudarshan an Lindblad) master equation with the shown format, where $n$ is the number of excited states in the system and $\Gamma_{ij}$ represents the probability of a jump from state $|j\rangle$ to $|i\rangle$. If $i = j$, a jump is made into a *sink*.

### 5.1.3 Initial Condition

{label}.initcon
$n$
i  $\langle i|\phi_0\rangle$
$\vdots$  $\vdots$

File contains the initial condition of the system; i.e. the initial wavefunction with the shown format, where $n$ is the number of excited states in the system and $|\phi_0\rangle$ is the initial wavefunction. NB: The wavefunction can be automatically normalised.

## 5.2 Optional Input

### 5.2.1 Markovian Dephasing

Activated using the command line flag: *-wn*

| | |
|---|---|
| {label}.wnd<br>$n$<br>i $\gamma_i$<br>$\vdots$ $\vdots$ | File contains the master equation component, $\gamma_i$, for Markovian noise with the shown format, where $n$ is the number of excited states in the system. This noise is added to the master equation using |

$$\mathcal{L}_D(\rho) = \sum_{i=1}^{N} \gamma_i \left[ - \left\{ \sigma_i^+ \sigma_i^-, \rho \right\} + 2\sigma_i^+ \sigma_i^- \rho \sigma_i^+ \sigma_i^- \right]. \tag{5.1}$$

Practically, this adds a jump process and an extra term to Eq. 2.14:

$$\Gamma_i = \sum_{j=1}^{N} \Gamma_{ji} + \gamma_i. \tag{5.2}$$

### 5.2.2 Static Dephasing

Activated using the command line flag: *-sn*

| | |
|---|---|
| {label}.snd<br>$n$<br>i j $\sigma_{ij}^2$<br>$\vdots$ $\vdots$ $\vdots$ | File contains the noise width, $\sigma_{ij}^2$, for static Gaussian noise with the shown format, where $n$ is the number of excited states in the system. For each quantum trajectory, the sites are dephased by an uncorrelated random number with a gaussian distribution.<br><br>NB: this noise can be applied to the hopping elements as well as the site energies. |

### 5.2.3 Ornstein-Uhlenbeck Dephasing

Activated using the command line flag: *-ou*

| | |
|---|---|
| {label}.oud<br>$n$<br>i $\sigma_i^2$ $\tau_i$<br>$\vdots$ $\vdots$ $\vdots$ | File contains the noise width, $\sigma_i^2$, and correlation time, $\tau_i$, for an Ornstein-Uhlenbeck process with the shown format, where $n$ is the number of excited states in the system. The OU process is generated using |

$$\begin{aligned} X_0 &= \sigma_i Z_0 & (5.3) \\ X_n &= \kappa_n X_{n-1} + \sqrt{1-\kappa_n^2}\sigma_i Z_n, & (5.4) \end{aligned}$$

where the *correlation coefficient*, $\kappa_n = e^{-|t_m - t_n|/\tau_i}$ and $Z_i$ are uncorrelated random numbers with a gaussian distribution.
NB: this noise expensive since, unlike the *.dph and *.sgn, it is calculated for every point on the fine grid.

### 5.2.4 Filtered Dephasing

Activated using the command line flag: *-fn*

| | | | | |
|---|---|---|---|---|
| {label}.fnd | | | | |
| $n$ | | | | |
| i | $\sigma_j^2$ | $M_j$ | $f_j^L$ | $f_j^H$ |
| $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ | $\vdots$ |

File contains the noise width, $\sigma_j^2$, and filter parameters (order, $M_j$, and cutoff frequencies, $f_j^L$ and $f_j^H$), for filtered Gaussian noise with the shown format, where $n$ is the number of excited states in the system. The procedure is applied at every point on the fine grid.
If $f_j^L = f_j^H = 0$, then no filter is applied and a decrete white-noise process is produced (NB: this is unlike the white noise produced by the *-wn* flag, since it has a frequency spectra that depends on the fine grid).
If $f_j^L = 0 \neq f_j^H$, a highpass filter is applied with a frequency cutoff of $f_j^H$
If $f_j^L \neq 0 = f_j^H$, a lowpass filter is applied with a frequency cutoff of $f_j^L$
If $f_j^L > f_j^H$, a bandpass filter is applied and if $f_j^L < f_j^H$, a bandstop filter is applied. The filter is applied using the formula,

$$X_n = \sum_{m=1}^{M} h_m Z_{n-m}, \tag{5.5}$$

where $h_i$ is the frequency response corresponding to the selected filter and $Z_i$ are uncorrelated random numbers with a gaussian distribution. The frequency responses for each filter are give below.

| Type | $m = \frac{M}{2}$ | $m \neq \frac{M}{2}$ | Criteria |
|---|---|---|---|
| lowpass | $2dtf_j^L$ | $\frac{\sin 2\pi dt f_j^L \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)}$ | $f_j^L > 0,\ f_j^H = 0$ |
| highpass | $1 - 2dtf_j^H$ | $\frac{\sin \pi \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)} - \frac{\sin 2\pi dt f_j^H \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)}$ | $f_j^L = 0,\ f_j^H > 0$ |
| bandpass | $2dt\left(f_j^H - f_j^L\right)$ | $\frac{\sin 2\pi dt f_j^H \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)} - \frac{\sin 2\pi dt f_j^L \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)}$ | $f_j^L > f_j^H$ |
| bandstop | $1 - 2dt\left(f_j^H - f_j^L\right)$ | $\frac{\sin \pi \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)} - \frac{\sin 2\pi dt f_j^H \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)} + \frac{\sin 2\pi dt f_j^H \left(m - \frac{M}{2}\right)}{\pi\left(m - \frac{M}{2}\right)}$ | $f_j^L < f_j^H$ |

where $dt$ is the time step of the fine grid. Since the frequecy response is evaluated before the main body of the calculation, it add little expense. However, the filter itself is evaulated at every step, so high-order filters will be expensive.

### 5.2.5 Plane Wave Dephasing

Activated using the command line flag: *-pw*

{label}.pwd
$n$
i $a_i$ $f_i$ $p_i$
$\vdots$ $\vdots$ $\vdots$ $\vdots$

File contains the Amplitude, $a_i$, signal frequency, $f_i$, and phase, $p_i$, of a plane wave signal with the shown format, where $n$ is the number of excited states in the system. The plane wave is generated using

$$x_{pw}(t) = a_i \sin\left[2\pi\left(f_i + p_i\right)t\right] \tag{5.6}$$

### 5.2.6 Square Wave Dephasing

Activated using the command line flag: *-sq*

{label}.sqd
$n$
i $a_i$ $f_i$ $M_i$
$\vdots$ $\vdots$ $\vdots$ $\vdots$

File contains the Amplitude, $a_i$, signal frequency, $f_i$, and fourier sample size, $M_i$, of a square wave signal with the shown format, where $n$ is the number of excited states in the system. The square wave is generated using

$$x_{sq}(t) = \frac{4a_i}{\pi}\sum_{k=1}^{M_i}\frac{\sin\left(2\pi\left(2k-1\right)f_i t\right)}{2k-1} \tag{5.7}$$

### 5.2.7 Frequency Modulated Dephasing

Activated using the command line flag: *-fm*

{label}.fmd
$n$
i $a_i$ $b_i$ $s_i$ $d_i$
$\vdots$ $\vdots$ $\vdots$ $\vdots$ $\vdots$

File contains the Amplitude, $a_i$, base frequency, $b_i$, signal frequency, $s_i$, and peak deviation, $d_i$, of a frequency-modulation signal with the shown format, where $n$ is the number of excited states in the system. The FM signal is generated using

$$x_{fm}(t) = a_i \cos\left(2\pi b_i t + \frac{d_i}{s_i}\cos\left(2\pi s_i t\right)\right). \tag{5.8}$$

### 5.2.8 Dephasing Correlation

Activated using the command line flags: *-csn -cou*

{label}.sitecorr

$$\begin{array}{lll} n & & \\ \text{i} & \text{j} & c_{ij} \\ \vdots & \vdots & \vdots \end{array}$$

File contains the correlation constant, $c_{ij}$, which correlates the noise in site $|j\rangle$ with the noise in site $|i\rangle$. The file has the shown format, where $n$ is the number of excited states in the system. Static noise and the OU process are correlated using

$$x_j = \sigma_j^2 \left[ (1 - c_{ij}) \, Z_j + c_{ij} Z_i \right], \tag{5.9}$$

where $Z_i$ is the route random number for site $i$.

### 5.2.9 QA Algorithm

Activated using the command line flag: *-qa*

{label}.qa_q

$$\begin{array}{lll} n & & \\ \text{i} & \text{j} & \mathcal{Q}_{ij} \\ \vdots & \vdots & \vdots \end{array}$$

{label}.qa_a

$$\begin{array}{llllllll} N_A & & & & & & & \\ A_\eta & \tau_\eta & T_\eta & \phi_\eta^a & \phi_\eta^b & \phi_\eta^c & \tau'_\eta & T'_\eta \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \end{array}$$

File contains $\hat{\mathcal{Q}}$ matrix and pulse factor $\mathcal{A}(t)$, which are added to the Hamiltonian,

$$\mathcal{H}' = \mathcal{H} + \hat{\mathcal{Q}} \mathcal{A}(t). \tag{5.10}$$

$\hat{\mathcal{Q}}$ is of the form,

$$\hat{\mathcal{Q}} = \sum_{i,j}^{N} \mathcal{Q}_{ij} \, |i\rangle\langle j| \, , \tag{5.11}$$

and $\mathcal{A}(t)$ is of the form,

$$\mathcal{A}(t) = \sum_{\eta}^{N_A} A_\eta e^{-\left(\frac{t-\tau_\eta}{T_\eta}\right)^2} \cos\left(\phi_\eta(t)\right), \tag{5.12}$$

where

$$\phi_\eta(t) = \phi_\eta^a t - \phi_\eta^b \text{erf}\left(\frac{t - \tau'_\eta}{T'_\eta}\right) + \phi_\eta^c. \tag{5.13}$$

# 6 Output Files

## 6.1 Default Output

### 6.1.1 Density Matrix

{label}.dm

$$\begin{array}{ccccc} t & \rho_{00} & \rho_{01} & \dots & \rho_{NN} \\ \vdots & \vdots & \vdots & & \vdots \end{array}$$

On exit, the file contains the complete density matrix, $\rho$, at every point on the coarse grid with the shown format, where $t$ is time and $N$ is the size of the Hamiltonian.

### 6.1.2 Site Population

{label}.population

$$\begin{array}{ccccc} t & \rho_{00} & \rho_{11} & \dots & \rho_{NN} \\ \vdots & \vdots & \vdots & & \vdots \end{array}$$

On exit, the file contains the population of each state - or the diagonal components of the density matrix - at every point on the coarse grid with the shown format, where $t$ is time and $N$ is the size of the Hamiltonian.

## 6.2 Optional Output

### 6.2.1 State Purity

Activated using the command line flag: *-purity*

{label}.purity

$$\begin{array}{cc} t & \gamma \\ \vdots & \vdots \end{array}$$

On exit, the file contains the state purity, $\gamma$, at every point on the coarse grid with the shown format, where $t$ is time.

### 6.2.2 Convergence

Activated using the command line flag: *-con*

{label}.convergence

$$\begin{array}{ccccc} t & \sigma_0 & \sigma_1 & \dots & \sigma_N \\ \vdots & \vdots & \vdots & & \vdots \end{array}$$

On exit, the file contains the relative change to the density matrix after the final trajectory for each site population, $\sigma_i$, at every point on the coarse grid with the shown format, where $t$ is time.

## References

[1] A Kossakowski, *Reports on Mathematical Physics* **3**, 247 (1972).

[2] Goran Lindblad, *Communications in Mathematical Physics* **48**, 119 (1976).

[3] Vittorio Gorini, Andrzej Kossakowski, and Ennackal Chandy George Sudarshan, *Journal of Mathematical Physics* **17**, 821 (1976).

[4] Giuliano Benenti, Giuliano Strini, and Giulio Casati, *Principles of quantum computation and information* (World scientific, 2004).

[5] HJ Carmichael, *Physical review letters* **70**, 2273 (1993).

[6] R Dum, P Zoller, and H Ritsch, *Physical Review A* **45**, 4879 (1992).

[7] Jean Dalibard, Yvan Castin, and Klaus Mølmer, *Physical review letters* **68**, 580 (1992).

[8] Klaus Mølmer, Yvan Castin, and Jean Dalibard, *JOSA B* **10**, 524 (1993).

[9] Carl Runge, *Mathematische Annalen* **46**, 167 (1895).

[10] Wilhelm Kutta (1901).

[11] William H Press, Saul A Teukolsky, William T Vetterling, and Brian P Flannery (2002).

[12] H Haken and G Strobl, *Zeitschrift für Physik* **262**, 135 (1973).

[13] Alex W Chin, Animesh Datta, Filippo Caruso, Susana F Huelga, and Martin B Plenio, *New Journal of Physics* **12**, 065002 (2010).

[14] Saeed V Vaseghi, *Multimedia Signal Processing: Theory and Applications in Speech, Music and Communications* pp. 111–153.

[15] George E Uhlenbeck and Leonard Salomon Ornstein, *Physical review* **36**, 823 (1930).

[16] CW Gardiner, *Stochastic methods* (Springer-Verlag, Berlin–Heidelberg–New York–Tokyo, 1985).

[17] Lorenz Bartosch, *International Journal of Modern Physics C* **12**, 851 (2001).