Hints for the users that approach the MCTDHB code for the first time

Antonio Negretti
Institut für Quanteninformationsverarbeitung, Universität Ulm, D-89069 Ulm, Germany

For the MCTDHB Package (MCTDHB.org)

I. BRIEF REVIEW OF THE THEORY BEHIND THE MCTDHB METHOD

The method is designed to determine both the stationary state (by imaginary propagation) and the real time (forward) propagation of the many-body Schrödinger equation of N interacting structureless bosons:

$$i\hbar\partial_t|\psi\rangle = \hat{H}|\psi\rangle, \tag{1}$$

where

$$\hat{H} = \sum_{k,j} h_{k,j}\hat{b}_k^\dagger\hat{b}_j + \frac{1}{2}\sum_{k,j,q,s} W_{k,j,q,s}\hat{b}_k^\dagger\hat{b}_j^\dagger\hat{b}_q\hat{b}_s. \tag{2}$$

Here $h_{k,j}$ are the matrix elements of the one-body Hamiltonian, which contains the kinetic and potential (e.g., of the trap) energy terms of a single boson, whereas $W_{k,j,q,s}$ are the matrix elements of the two-body interaction between two bosons. The annihilation and creation operators satisfy the usual bosonic commutation relations $[\hat{b}_j, \hat{b}_k^\dagger] = \delta_{k,j}$.

The key idea of the MCTDHB method [1, 2] is that the many-body quantum state can be written as:

$$|\Psi(t)\rangle = \sum_{\vec{n}} C_{\vec{n}}(t)|\vec{n};t\rangle, \tag{3}$$

where $\vec{n} = (n_1, n_2, \ldots, n_M)$, $C_{\vec{n}}(t) \in \mathbb{C}$ are the expansion coefficients, and

$$|\vec{n};t\rangle = \frac{1}{\sqrt{n_1!\cdots n_M!}}[\hat{b}_1^\dagger(t)]^{n_1}\cdots[\hat{b}_M^\dagger(t)]^{n_M}|\mathrm{vac}\rangle \tag{4}$$

with $|\mathrm{vac}\rangle$ the vacuum state,

$$\hat{b}_k(t) = \int \mathrm{d}\mathbf{r}\phi_k^*(\mathbf{r},t)\hat{\Psi}(\mathbf{r},t), \tag{5}$$

and $\hat{\Psi}(\mathbf{r},t) = \sum_k \hat{b}_k(t)\phi_k(\mathbf{r},t)$ [$\mathbf{r} \equiv (x,y,z)$] is the usual bosonic quantum field operator. Here the modes (alternatively called, orbitals) $\phi_k(\mathbf{r},t)$ form an orthonormal basis in the Hilbert space of single-particle functions. Given this, the MCTDHB method truncates such a basis up to M states, which have to provide a faithful multimode description of the many-body quantum system dynamics. Since both $C_{\vec{n}}(t)$ and $|\vec{n};t\rangle$ are time-dependent quantities determined by the variational principle, such a truncation and the description of $|\Psi(t)\rangle$ is more effective, especially in view of a numerical computation. In addition to this, the occupancies n_k have to fulfill the following relation: $\sum_{k=1}^M n_k = N$. The mode populations n_k are the eigenvalues of the one-body density matrix $\rho(\mathbf{r},\mathbf{r}') = \langle\hat{\Psi}^\dagger(\mathbf{r}')\hat{\Psi}(\mathbf{r})\rangle$ (see Ref. [3] for notation and further definitions). Thus, in summary, the goal, when applying the MCTDHB method, is to solve the time-dependent equations of motion of the coefficients $C_{\vec{n}}(t)$ and of the orbitals $\phi_k(\mathbf{r},t)$, as illustrated in Refs. [1, 2]. The difficulty of the numerical computation is mainly due to the fact that the equations of motion of the orbitals form a set of coupled integro-differential equations (for more details see Refs. [1, 2]).

II. USE OF THE CODE

The code is provided in the folder `V2.2`, in which there are several files and directories. The MCTDHB numerical toolbox contained is such folder works for any number of modes M (in principle), particle number N, and for any kind of two-body interaction $W(\mathbf{r}_k - \mathbf{r}_j)$, that is, a regular and well-defined function within its support. The code is also designed to work in 2D and 3D, but, up to now, only in 1D is very well tested both for time-independent and time-dependent dynamics. The 2D version should be available (i.e., sufficiently tested to be considered reliable) by the end of 2012.

Remark 1. *The code has been tested very well on the bwGrid and the large cluster at the University of Stuttgart. Once the user is connected to the bwGrid, before he/she starts to use the code, the user has to run first the script* `source /.bashrc` *or simply* `bashrc` *from the home-directory. The file* `.bashrc` *contains several alias and, importantly, it loads the* `mpi` *and* `mkp` *libraries and the compiler for INTEL 11.1, which are needed in order to run the code.*

Remark 2. *In the current version of the code one can actually consider up to $M = 25$ mode functions (see also the folder* `V2.2/GENERATORS_CI/`*). Typically this is enough, since the user has to consider that the dimension of the Hilbert space grows like $(N+M-1)!/[N!(M-1)!]$ (see Ref. [4] for the size and the mapping of the configuration space).*

In the directory `V2.2/source/` are stored all sources files, while in the folder `V2.2/test/` there are some tests for the code that the user should run the first time that

the MCTDHB package is installed on his/her computer, in order to check that everything is working well (e.g., no problems occurred with the processor used, etc.). If those tests are fine, it means that all software has been installed in your own machine and that the code is ready to be used.

In the folder `V2.2/user_guesslib/` are stored four (Fortran) files:

1. `Get_InterParticle.F`;
2. `Guess_CI.F`;
3. `Guess_PSI.F`;
4. `VTRAP_EXT_TD.F`.

The first file is a subroutine to generate the two-body time-independent interaction $W(\mathbf{r}_k - \mathbf{r}_j)$, while the second and third one are subroutines to generate the initial set of coefficients $C_{\vec{n}}(t)$ and modes $\phi_k(\mathbf{r}, t)$ at time $t = 0$. These two subroutines are only needed in the case the initial stationary state has to be determined. Besides this, since such a state is obtained by imaginary propagation, the initial values of the coefficients and the initial set of modes is not relevant, because the state during the imaginary propagation has to be renormalized at each time step. Thus, usually, both `Guess_CI.F` and `Guess_PSI.F` should not be modified, unless the specific problem under investigations requires a proper choice. In the current version, however, the initial set of coefficients are chosen such that the first of them is 1 and all the others are set to zero. Regarding the initial set of modes, these are given as product of exponential functions and monomials. The important point here, is that they have to form an orthonormal basis, and therefore any set of which is (in principle) a good choice.

Finally, the file `VTRAP_EXT_TD.F` is a subroutine for the external trapping potential (needed for the computation of the $h_{k,j}$ matrix elements). Currently, it is written in such a way that it works not only when analytical expressions of the potential, both in time and in space, are provided, but also when it relies on some real-valued time-dependent function $d(t)$ (e.g., the trap position), for which no analytical formula is afforded. This situation is typical of optimal control problems, for which the time-dependence of $d(t)$ is computed numerically with some optimization algorithm (e.g., the Krotov method [5] or the recent developed one named CRAB [6]). If this is the case, in the file `VTRAP_EXT_TD.F` there is the subroutine called `USER_DATA`, which allows one to read the file that contains the data of the time-dependent function $d(t)$. More precisely, such a file has two-columns: the first one represents the time vector $\vec{t}$ and the second column represents the vector $\vec{d}$, that is, the time-dependent function evaluated at the times defined by the vector $\vec{t}$. Besides this, the subroutine enables the user to "process" these data for the needs of the integration of the equations of motion of the orbitals and coefficients $C_{\vec{n}}(t)$ (see later discussion about the adaptive time step). Indeed, while typically the elements of the data vector $\vec{d}$ are computed and stored at times t_k separated by a fixed time step (i.e., all temporal grid points are separated by $\Delta t = t_2 - t_1$ [8]), this is not the case for the (available) integrators for solving the (real time) dynamics of the many-body quantum system, where a time-adaptive scheme is used.

Given all this, the bottom line is that the file `VTRAP_EXT_TD.F` is the one that has typically to be modified accordingly to the specific investigated problem.

Remark 3. *Any time that one of the above four outlined files is modified, then the user has to compile the code by means of the line command* `V2.2/make` *or with* `V2.2/Makefile.BWGriD`.

In the folder `IN.FILES` there are two files:

1. `input.in`;
2. `properties.in`.

These two files are central for the run of the MCTDHB code, especially `input.in`, in which we provide the necessary information about the quantum system and the orbitals, like the nature of the two-body interaction, the number of particles, etc. Actually, the file `propeties.in`, is really needed when the user is interested in the computation of the first and second correlation functions both in coordinate and momentum spaces [3]. If this is not the case, the user does not need to copy such a file into the working directory.

In the folder `Scripts` there are a number of scripts in order to handle the saved data, like the total energy, the spatial profile of the mode functions, the density, etc. Such scripts enable the user to make movies or to generate postscript files. Thus, the user has at his/her disposal a useful tool to make data analysis.

Finally in the folder `V2.2` there are some examples, for instance, `run-example1-PBS.sh`, which can be used to submit a job on the bwGrid cluster.

III. THE `INPUT.IN` FILE

Here, a detailed explanation of the content of the `input.in` is given. It is divided into three main blocks: PRM, ORB, and TM.

A. The PRM block

The first variable is `MB_JOB_TYPE`. This sets the "kind" of MCTDHB that the user is interested to apply. By typing `ALL` the user uses the standard MCTDHB method, where basically Eq. (2) is solved. In some circumstances, however, like for the Bose-Hubbard Hamiltonian used to describe the physics of ultracold atoms in an optical lattice, the interaction between bosons occurs only among nearest-neighbors or next-nearest-neighbors, and therefore several $h_{k,j}$, $W_{k,j,q,s}$ matrix elements are zeros. This

simplifies the problem quite a lot, and therefore the numerical integration of the equations of motion. By typing `FCI` the user selects the option for which special Hamiltonians like the Bose-Hubbard could be studied.

In the next four lines of the `input.in` file there are given the variables `Morb, Npar, xlambda_0, JOB_PreFac`. The variable `Morb` corresponds to the number of modes or orbitals, which can range from 1 to 25. The variable `Npar` represents the total number of bosonic particles and can be any positive integer number. The variable `xlambda_0` represents the strength of the two-body interaction potential. For instance, in the case of ultracold atoms, where the interaction between particles is well described by the pseudopotential, the variable `xlambda_0` is precisely g_{1D} in the expression $W(x_j - x_k) = g_{1D}\delta(x_j - x_k)$ (a similar applies in 2D and 3D). Finally the variable `JOB_PreFac` sets the type of time evolution: if the user sets (0,-1), then the real time forward propagation is performed, if the user types (0,1), then the real time backward propagation is performed, and if (-1,0) is chosen, then the imaginary time propagation is performed.

Remark 4. *In order to use the MCTDHB code, the Schrödinger equation (2) has to be written in dimensionless units. Thus, also* `xlambda_0` *has to be provided in such units. Besides, the kinetic energy operator is given by* $-\nabla^2/2$.

Remark 5. *In the special case of* $M = 1$*, the Gross-Pitaevskii equation for* N *particles is simulated. Such equation is written in a slightly different way (namely, the orbitals is normalized to 1 and a defined number of bosons* N *is assumed, see below) from the one typically used in the BEC community, that is, in the 1D case we have:*

$$-i\hbar\frac{\partial\phi(x,t)}{\partial t} = \hat{H}_{\rm gp}[\phi]\phi(x,t),$$
$$\hat{H}_{\rm gp}[\phi] = \left[-\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2} + V(x,t) + g_{1D}N'|\phi(x,t)|^2\right] \tag{6}$$

where $V(x,t)$ *is the external trapping potential. Here the nonlinear term is given by* $g_{1D}(N-1)|\phi(x,t)|^2$*, where* $N' = N - 1$ *appears. This equation can be derived by using the Hartree Ansatz for the ground state wave function, in which all particles are in the condensed mode, namely* $|\Psi\rangle = |N, 0, \ldots, 0\rangle \equiv |\psi\rangle \otimes \cdots \otimes |\psi\rangle$*. In such a formulation, also the Gross-Pitaevskii energy functional is slightly different, namely*

$$\frac{E}{N} = \int \mathrm{d}x \left[\frac{\hbar^2}{2m}\left|\frac{\partial\phi(x)}{\partial x}\right|^2 + V(x)|\phi(x)|^2 + \frac{g_{1D}N'}{2}|\phi(x)|^4\right]. \tag{7}$$

Going on in the `input.in` file we find the variable `GUESS`, which can assume three different values: `HAND, BINR, DATA`. When the user sets `HAND`, then the code uses as initial conditions for the coefficients $C_{\vec{n}}(t)$ and modes $\phi_k(\mathbf{r},t)$ the ones defined in the files `Guess_CI.F` and `Guess_PSI.F` of the folder `V2.2/user_guesslib/`. This option is typically used for the determination of the stationary state. The other two options, that is, `BINR, DATA`, are used when time-dependent dynamics is performed. When the option `BINR` is chosen, then we need to move/copy the files `CIc_bin` and `PSI_bin` into the working directory created for the time-dependent dynamics, that have been created when running the imaginary time propagation. Instead, if the option `DATA` is chosen, then we have to move the `xxxtime.dat` and `xxxcoef.dat` files into the corresponding working directory. Here `xxx` stands for the corresponding time at which the data have been stored. For example, if for the imaginary time propagation a total propagation time of $T = 20$ has been chosen, then the initial time for the real time forward propagation is given by T, that is, the data stored at that time. This is precisely the meaning of the variable `Binary_Start_Point_t`. If we set `Binary_Start_Point_t = 20.0d0`, then it starts the calculation with the coefficients and modes saved at that time, BUT only when `GUESS='BINR'`. On the other hand, if we choose `GUESS='DATA'`, then we have to provide which data file for the coefficients and modes have to be used and read. This is given in the next lines of the `input.in` file, where the variable `Time_Res_Orb_File_Name = 'xxxtime.dat'` and `Time_Res_CIc_File_Name = 'xxxcoef.dat'` have to be provided. Thus, for the previous example, `Time_Res_Orb_File_Name = '20.0000000time.dat'` and `Time_Res_CIc_File_Name = '20.0000000time.dat'`. This last option is relevant in the case, for example, when the initial stationary state has been already computed, but the initial condition (at time $t = 0$) of the real time propagation is given by the stationary state with a kick of the form $e^{i\mathbf{k}\cdot\mathbf{r}}$. In this case we first have to modify the modes such that $\phi_k(\mathbf{r}, t = 0) \mapsto e^{i\mathbf{k}\cdot\mathbf{r}}\phi_k(\mathbf{r}, t = 0)$, which is easier to do when data files are handled, and then perform the forward real time propagation.

The last variable is `ORB_DIAG`: if `ORB_DIAG=.T.`, then it uses as the initial set of modes the eigenstates of the trapping potential (see also file `Guess_PSI.F` in the folder `V2.2/user_guesslib/`); if `ORB_DIAG=.F.`, then it does not diagonalize and uses the aforementioned initial conditions for the modes defined in `Guess_PSI.F`.

B. The ORB block

Here the variable `DIM_MCTDH` sets the dimension of the problem, that is, 1, 2 or 3. Then the variables `NDX, NDY, NDZ` set the number of (DVR - Discrete Variable Representation) points, namely the grid points for the x, y, and z axis, respectively. The variables `Time_DVRMETHODX, Time_DVRMETHODY, Time_DVRMETHODZ` set the type of DVR-basis for the x, y, and z axis, respectively. If 1, then

the basis is the one defined by the eigenfunctions of the harmonic oscillator; if 3 the sin basis; if 4 the FFT routine is used, and if 5 the Exponential basis. The variable `Wxx_TYPE` sets the type of two-body particle interaction (0 means pseudopotential). The variables `Time_xint` and `Time_xfnl` define the lower and upper bounds, respectively, of the interval [`Time_xint`,`Time_xfnl`], namely the (spatial) support of the orbitals. The same applies for the y and z axes.

Finally, the variable `Time_mass` defines the mass in the Schrödinger equation in case the kinetic energy operator, after the transformation to dimensionless units, is not given by the usual expression $-\nabla^2/2$, but rather by $-\kappa\nabla^2/2$, where κ is some numerical, dimensionless constant (see also the previous Remark 4).

C. The TM block

Here the variables `Time_Bgn` and `Time_Fnl` correspond to the initial and final time of the propagation (both for real and imaginary and for forward and backward propagations). The variable `Time_MAX` sets an upper bound for the maximum computational time, beyond which it does not make sense anymore to perform the computation (no more memory available, etc). The variable `Time_print_step` fixes the time step at which the data files have to be printed in the working directory, while the variable `Time_ici_prt` has to be a positive integer and it defines how many times the `xxxcoef.dat` file have to be created. For instance, `Time_print_step=0.1d0` and `Time_ici_prt=10` means that the `xxxtime.dat` files are generated at the rate given by the time step 0.1, while the `xxxcoef.dat` files at the rate given by the time step 1, that is, 10 times more than `Time_print_step`. The variable `Time_tau` is the initial time step needed for the numerical integration of the equations of motions of the real time propagation.

Remark 6. *The imaginary propagation does not uses a time-adaptive step. For $M = 1$ both the imaginary and the real time propagations do not need to start with a small time step (0.1 is already sufficient), but for multimode simulations it is better to use, for the real time propagation, a smaller time step, like 10^{-6}.*

The variable `Time_TolError_Total` sets the (numerical) precision of the short iterative Lanczos (SIL) integrator, which is used for the integration of the equation of motion of the coefficients $C_{\vec{n}}(t)$, as well as the precision of the Adams-Bashforth-Moulton (ABM) predictor-corrector method, which is used in order to solve the equation of motion of the orbitals $\phi_k(\mathbf{r}, t)$ (see also Ref. [2]). The variables `Time_Min_Occ_Allowed, Time_minsil, Time_maxsil` are very specific of the SIL integrator, and therefore it is better to keep the values provided in the `V2.2` by the developers of the MCTDHB. The variable `Time_intgr` defines the type of integrator (BS - Bulirsch-Stoer method, ABM - Adams-Bashforth-Moulton predictor-corrector method, OMPABM - optimized version of ABM, RK - Runge Kutta routine, STIFF for problems in which the solution is a highly oscillating function). The variable `Time_intgr_max_step` sets the maximum time step to be used in the adaptive schemes. For instance, if we set `Time_intgr_max_step=0.1d0` and it turns out that the adaptation of the time step during the computation would set a time step of 0.2d0, then it sets, regardless the checks made for the time adaption, the time step to 0.1d0. The variable `PRINT_DATA` creates or not the `xxxtime.dat` and `xxxcoef.dat` files accordingly to the time steps defined previously. Thus, if `PRINT_DATA=.T.` it prints, if `PRINT_DATA=.F.` it does not. This last option might be useful if these data are not necessary for the data analysis. The variable `TD_Err_scale` defines the rate at which the time step $\Delta t'$ =`TD_Err_scale`$\times\Delta t$ has to be adjusted in the time-adaptive integrators. The variable `LZ` switch on (`LZ=.T.`) or off (`LZ=.F.`) the angular momentum in the z direction with angular velocity `OMEGAZ` for 2D problems. In the 1D scenario it has to be off.

Finally, the last variable in the `input.in` file, named `STATE`, defines which many-body quantum state has to be computed in the imaginary propagation. When the user sets `STATE=1`, it means that the ground state wavefunction is computed. When, as represented by the SIL subspace, `STATE=2` is set, then the first many-body excited state is computed (e.g., for a noninteracting many-body quantum system only one boson is promoted to the next high energy level while the other bosons are in the ground state), and so on for the next ones.

IV. WORKFLOW FOR THE MCTDHB

Here, the sequence of commands to be typed from the terminal are listed. Some of the listed commands, however, depend on the way the user structures the working directory (i.e., it relies on the users habits in numerical programming), and therefore some of them might not be necessarily used. Hence, what is shown below is basically an illustrative (educational) example.

The underlying idea of the below outlined sequence is to create in the user workspace location in the bwGrid a working directory named `mctdhb`, where all numerical computations are performed, while a copy of the MCTDHB source code is stored in a separated folder (i.e., `V2.2`) within the same workspace location. From here the user copies only the necessary files into the working directory (i.e., `mctdhb`). Within such a folder a subdirectory `relax` is created, where the imaginary time propagation is executed. Thus, within the subdirectory `relax` another folder is created (named `td`), where the real time forward propagation is performed.

The first two steps, however, are: connection to the bwGrid; make a copy of the `V2.2` folder provided by the MCTDHB developers in the workspace location of the bwGrid. For instance:

1. `>ssh -XY anegrett@themis.rz.uni-ulm.de`

2. `>cp V2.2 /scratch1/ws/anegrett-Negretti-0/`

Procedure for the imaginary time propagation:

1. `>bash`
2. `>cd /scratch1/ws/anegrett-Negretti-0/`
   (Goto user workspace location in the bwGrid)
3. `>mkdir mctdhb`
   (Create a folder for the simulation you wish to carry out, the so-called working directory)
4. `>cd V2.2/user_guesslib/`
   (Open the file `VTRAP_EXT_TD.F` with some text editor and modify the trapping potential)
5. `>cd ../`
6. `>make`
   (Compile the program and libraries)
7. `>cd ../mctdhb/`
8. `>mkdir relax`
9. `>cd relax/`
10. `>cp ../../V2.2/bin/* .`
    (Make a copy of the required files in the working directory, that is, in the folder `/mctdhb/relax/`)
11. `>cp ../../V2.2/IN.FILES/input.in .`
    (If needed, copy also the `properties.in` file)
12. `>vim input.in`
    (Open, for example with `vim`, the `input.in` file and fix all parameters you need to perform the imaginary time propagation or, alternatively called, relaxation)
13. `>./boson_MCTDHB_intel`
    (Run the code)

Procedure for the real time forward propagation:

1. `>mkdir td`
2. `>cd td/`
3. `>cp ../../../V2.2/bin/* .`
   (Make a copy of the required files in the working directory, that is, in the folder `/mctdhb/relax/td/`)
4. `>cp ../input.in .`
5. `>cp ../CIc_bin .`
6. `>cp ../PSI_bin .`
7. `>vim input.in`
   (Open, for example with `vim`, the `input.in` file and fixe all parameters you need to perform the propagation)
8. `>./boson_MCTDHB_intel`
   (Run the code)

V. HOW TO MAKE POSTSCRIPT FILES AND MOVIES OF THE CREATED DATA

Assuming, for instance, the data to be in the folder where the time-dependent dynamics has been performed, namely in the folder `/mctdhb/relax/td/`, then, for the figures, the user has to proceed as follows:

1. `>cp ../../../V2.2/Scripts/sng_nat_occ_loop_M`

   (Here M has to be a positive integer. It corresponds precisely to the number of orbitals used in the simulation.)
2. `> ./sng_nat_occ_loop_M 5`
   (The script is executed, for instance, with $M = 5$.)
3. `gv fig_filename.ps`
   (Open one of the postscript files)

Instead, for the movies, the user has to proceed as follows:

1. `>cp ../../../V2.2/Scripts/sng_movie_MCTDHB_PSI_NO`

   (This is just an example for the wavefunction, but the user is asked to look in the `V2.2/Scripts/` directory for more details.)
2. `> ./sng_movie_MCTDHB_PSI_NO 3`
   (The script is executed, for instance, with 3 (out of M) natural orbitals.)
3. `mplayer filename.mpg`
   (Open the movie with some reader for movies, in this case `mplayer`)

VI. CONTENT OF `XXXTIME.DAT` FILE

A file like this has a variable number of columns, since such a number depends on the number of chosen orbitals. The first nine columns are always given and they represents:

1. x axis
2. y axis
3. z axis
4. DVR weight (see in this context Ref. [7] for more details)
5. $V(\mathbf{r})$ the trapping potential
6. $\rho_W(\mathbf{r})$ density in the working orbitals (W), that is, with the states defined in Eq. (4)
7. $\rho^*_W(\mathbf{r})$ complex conjugate of $\rho_W(\mathbf{r})$
8. $\rho_{NO}(\mathbf{r})$ density in the natural orbitals (NO), that is, the eigenstates of the reduced one-body density matrix

9. $\rho^*_{NO}(\mathbf{r})$ complex conjugate of $\rho_{NO}(\mathbf{r})$

Then there are $4M$ columns: the first $2M$ regard the working orbitals, whereas the last $2M$ concern the natural ones. Both the working and natural orbitals are collected in decreasing order with respect to the occupancy of the orbitals and for each orbital there are two associated columns: the first is the real part of the orbital and the second column represents its imaginary part.

VII. CONTENT OF NO_PR.OUT FILE

It contains $2+M$ columns: the first column represents the time vector t, whereas the last one the total energy of the system. For $M = 1$, that is, the Gross-Pitaevskii case, it is the energy functional defined in Eq. (7). In between there are M columns containing the occupancy of the orbitals with decreasing order.

[1] A. I. Streltsov, O. E. Alon, and L. S. Cederbaum, Phys. Rev. Lett. **99**, 030402 (2007).
[2] O. E. Alon, A. I. Streltsov, and L. S. Cederbaum, Phys. Rev. A **77**, 033613 (2008).
[3] K. Sakmann, A. I. Streltsov, O. E. Alon, and L. S. Cederbaum, Phys. Rev. A **78**, 023615 (2008).
[4] A. I. Streltsov, K. Sakmann, O. E. Alon, and L. S. Cederbaum, Phys. Rev. A **83**, 043604 (2011).
[5] V. F. Krotov, *Global Methods in Optimal control Theory*, vol. 195 (Marcel Dekker Inc., New York, 1996).
[6] T. Caneva, T. Calarco, and S. Montangero, Phys. Rev. A **84**, 022326 (2011).
[7] M. Beck, A. Jäckle, G. Worth, and H.-D. Meyer, Phys. Rep. **324**, 1 (2000).
[8] Here the time vector is defined as: $\vec{t} \equiv (t_1, t_2, \ldots, t_{N_t})$ with N_t the number of temporal grid points.