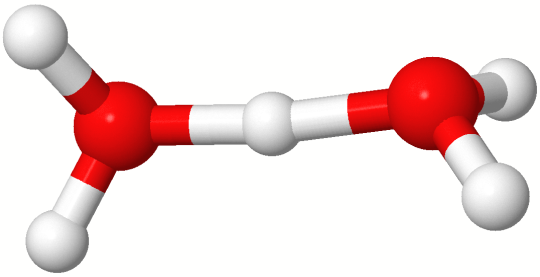
A Short Tutorial on Ab Initio Molecular Dynamics and Path Integral Molecular Dynamics

Using the i-PI/FHI-aims Infrastructure (2018)



(*Ab initio*) Molecular Dynamics
Manuscript for Exercise Problems

Prepared by Yair Litman, Nathaniel Raimbault and Mariana Rossi

Introduction

With this tutorial, we aim at introducing you to techniques used to solve the classical equations of motion for a system of interacting particles. The strategy we adopt is molecular dynamics (MD), a method that can be considered as old as modern physics itself, since in this framework one *propagates* in time a system of particles, given initial positions and momenta, by numerical integration of Newton's equations of motion. This technique allows one to sample the potential energy surface of a system at constant energy (microcanonical ensemble) and at constant temperature (canonical ensemble). As long as the *ergodic hypothesis* is valid, we will be, in principle, able to relate the time averaged parameters (or properties), to the thermodynamic averages over the corresponding statistical ensemble.

The kind of MD we are interested in is Born-Oppenheimer MD, where the degrees of freedom of the nuclei are considered to be decoupled from the degrees of freedom of the electrons (Born-Oppenheimer approximation). On top of this approximation, usually one treats the nuclei as classical particles. Therefore, the forces between them are evaluated by solving the (ground-state) electronic structure problem for a given number of electrons and for a given position of the nuclei. In this tutorial, the electronic structure problem is solved via Density Functional Theory (DFT) within the FHI-aims code and the nuclei problem is managed by the i-PI [1] code. i-PI works through a client-server paradigm, where the *ab initio* code, in this case FHI-aims, is the client that provides the calculation of interatomic forces, and i-PI is the server that provides the evolution of the equations of motion that sample the desired ensemble.

By the end of the tutorial, we will also show a small example on how one can also treat the nuclei as quantum particles, by performing what we call "path-integral molecular dynamics" (PIMD). This type of dynamics is based on the Feynman path integral formalism [2]. This formalism exploits the isomorphism between the partition function of a quantum particle with that of several repeated classical particles (beads) connected by harmonic springs and forming a closed loop (ring polymer). In PIMD, molecular dynamics is employed to drive phase-space statistical sampling. Static ensemble properties of the system can be accurately (in principle exactly in the limit of infinite beads) evaluated within this technique.

All the necessary theoretical background for a good understanding of this tutorial is given in the lectures of Luca Ghiringhelli (about statistical mechanics) and Mariana Rossi (about *ab initio* molecular dynamics).

The tutorial contains five exercises. All of them are based on a paradigmatic system, namely the gas-phase (isolated) $H_5O_2^+$, which some of you may recognize as the Zundel (cat)ion.

The **first**, and quick, introductory exercise presents the machinery we will use through the tutorial. In particular, you will learn how to setup the i-PI and FHI-aims inputs.

The **second** and **third** exercises pose the problem of choosing the right settings for obtaining a stable and reliable molecular dynamics trajectory. In particular we will investigate the effect of the self-consistent cycle and force convergence thresholds as well as the influence of the size of the time step.

The **fourth** exercise deals with the choice of a thermostat for simulating the canonical ensemble, i.e., the contact between your system of interest and a thermal reservoir. In this tutorial we will focus on the so-called stochastic thermostats. Other implementations,

for example based on extended Lagrangians exist as well, and will be briefly explained in the exercise. Also, we will see how one can use thermostats to break the fluctuation dissipation theorem and mimic the quantum nature of the nuclei by changing the momentum distribution on a "classical-nuclei" simulation.

In the **fifth** and last exercise we will compute free energy differences through the method of thermodynamic integration. In addition, we are going to compare results obtained from simulations that neglect quantum statistics for the nuclei with results from simulations that take into account these effects (PIMD).

Molecular Dynamics: a client/server approach

Exercise 1: Getting started

In this exercise we are going to familiarize ourselves with the important notations of i-PI input files. As mentioned previously, the i-PI program works through a client-server architecture. i-PI is a python code, and as such does not to be explicitly installed in the machine. It relies on the python and numpy being preinstalled. The code can be downloaded from http://ipi-code.org/, where one can find also more information regarding the whole project. It can use INTERNET or UNIX sockets, that allow the system to be simulated on the same machine or on a different machine, as long as the calculation can communicate with the server.

Let's take a look at the input files. The example simulation here will consist of one zundel molecule in the canonical (NVT) ensemble at 300K.

Server: An example of an input file for i-PI can be found in `exercise_1/input.xml`. It is an xml file, which is quite intuitive to learn. Please take your time to understand the keywords that are there and consult the i-PI manual, contained in the i-PI program package. We will be using UNIX sockets here, which is the most convenient way to use the code when running both servers and clients in small desktop (or laptop) machines. For that, we have to specify only the `<address>`, which can simply be a string containing a name of your choice. For internet sockets, one would have to provide the relevant IP address for the `<address>` field and a number for the `<port>` field. For example, if server and client would run in the same machine, one could set the mode of `<ffsocket>` to `inet` and write `localhost` in the address field. In HPC architecture this procedure can be more involved as often computing nodes do not have a network connection to the head node, or one cannot keep a program running in the head node for indefinite amounts of time. Solutions can be found through scripts that write the address on the relevant files only after a job starts running in a specific computing node, or through setting up tunnels to other machines. If you can't find a solution to your problem, do no hesitate to ask in the i-PI forum: http://ipi-code.org/resources/forum/.

Client: The keyword to add to the `control.in` file of FHI-aims is

```
use_pimd_wrapper UNIX:<address> <portnumber>
```

where `<address>` should be substituted by the name of the socket you choose, and the port can be any number since it does not play a role for UNIX sockets. The address that goes in the `control.in` file should match the one in the `input.xml` file of i-PI. The `geometry.in` is only provided for the initialization of FHI-aims, so in order to avoid inconsistency problems always check that the atoms are listed in the same order as in the i-PI input file. (Here, we did that for you ;))

Note that in all subsequent exercises you will be free to choose `<address>` as you want, and you should please change it in your input files (`control.in` and `input.xml`) so that your server and your client always match!

Let's run an i-PI+FHI-aims Molecular Dynamics simulation!

Instructions

- Open a terminal at the current directory and launch i-PI by typing

  ```
  i-pi input.xml
  ```

  At this point i-PI should start and parse the input file. At the bottom of the output on the screen it should say:

```
Created unix socket with address EX1
@ForceField: Starting the polling thread main loop.
```

This means i-PI has started properly, has created the UNIX socket, and is waiting for the communications from the clients that take care of the force evaluations.

- Now we can launch FHI-aims. Open a second terminal, either manually or by typing Ctrl+Shift+t, and enter the command

  `aims.ipi.x`

  Then FHI-aims should start and you will see some outputs.

- Now switch to the terminal where i-PI is running, notice that i-PI has built the connection with FHI-aims with the following message,

  ```
  @SOCKET:  Client asked for connection from . Now hand-shaking.
  @SOCKET:  Handshaking was successful. Added to the client list.
  ```

  and started the Molecular Dynamics simulation. It should also print on screen some information about the time cost of each MD step.

- What we are going to do now is to kill FHI-aims (don't worry, you shall not be sued for that). Simply switch to the terminal where FHI-aims is running and press Ctrl+c. Now look at whether i-PI is still running. Notice that although the evolution of the MD is paused, i-PI itself does not die off but instead continues to run and waits for a new client to take over. Now start FHI-aims again by typing:

  `aims.ipi.x`

  What happens to i-PI now?

- What if one stops i-PI? Kill i-PI by typing Ctrl+c where it is running, or create a file named EXIT in the folder where i-PI is running (you can use the bash command `touch EXIT`). Watch how i-PI responds, and how FHI-aims reacts. Think about what are the advantages of a clean exit when a MD program stops unexpectedly.

- Take a look at all the output files written by i-PI. You should have the file ex1.out that describes the system properties, ex1.pos.xyz that records the atomic trajectories, and RESTART that contains all the information to restart the simulation.

- Last (and tricky) question , we are dealing with a charged system. Can you find where the total charge in the input.xml file is specified? Why?

Timing: ∼20 minutes total

The microcanonical ensemble

Exercise 2: The importance of the SC convergence criteria

In this exercise, we will investigate the importance of the self-consistency convergence criteria when simulating the microcanonical ensemble. The the input files for i-PI can be found in the folder `exercise_2`. Please note that the initialization in i-PI is done with a previously thermalized geometry.

Instructions

- First, build an input file (`control.in`) for FHI-aims using the LDA (**pw-lda**) functional, no spin polarization (**spin none**), and using the "**light**" numerical and basis set standards for the species. Add the line corresponding to the i-PI communication that was shown in the previous exercise, as well as the following line to tell FHI-AIMS that you need to compute the forces:

  ```
  compute_forces .true.
  ```

  Please refer to the manual for the exact syntax of these flags.

  Do not add any flags that we do not mention. They are not needed and might hinder the performance of the calculation.

- We want to run a 0.15ps MD run in the microcanonical ensemble (NVE), using a 0.0005ps (Δt = 0.5fs) time step. For that, open the provided `input.xml` and change the lines `timestep` and `total_steps` accordingly. You also need to specify the `dynamics` mode to be 'nve'. The file chosen for the `initialization` is a checkpoint file containing positions, velocities, and some other settings from a previous simulation where this molecule was thermalized, using the 'nvt' mode [1] – that is what the 'chk' mode in i-PI means [2]. In this case, initializing from the checkpoint file means that i-PI will read positions and velocities from it.

- After everything is set up, start the simulation in the following way

[1] You will see how such a thermalization is done in `exercise 4`.

[2] You can open and read the checkpoint file if you wish in order to see the structure of it, which is actually also an .xml file very similar to the input of i-PI but containing more blocks.

In order to start the simulation type:

```
i-pi input.xml > output-i-PI &
```

(Wait 5 seconds in order to let i-PI do the initialization)

```
mpirun -np 8 aims.ipi.x > output-FHI-aims &
```

- The symbol `&` puts the run in the background, so that the output file is created, but the terminal is free for other use.
- If you would like anyway to have a dynamic view of what happens in your output, after starting the simulation you can type:

  `tail -f <name-of-output>`
  Press Ctrl+c to exit.
- ATTENTION: do not start another FHI-aims run simultaneously.
  That would slow down BOTH calculations considerably.

- When the previous calculation is over, run another simulation, keeping all parameters mentioned above but changing the name of the output and also changing to the following "loose" self-consistency convergence criteria:

```
sc_accuracy_rho 1E-2
sc_accuracy_eev 1E-1
sc_accuracy_etot 1E-2
sc_accuracy_forces 1E-1
```

- Finally, run another simulation with "accurate" self-consistency criteria:

```
sc_accuracy_rho 1E-5
sc_accuracy_eev 1E-4
sc_accuracy_etot 1E-6
sc_accuracy_forces 5E-4
```

IMPORTANT: Remember to change the output prefix in the i-PI input in order to not overwrite any output file.

REMINDER: Choose `<address>` in your input files (`control.in` and `input.xml`) so that your server and your client match.

When it is done, plot the total energy (`conserved` in the i-PI language for NVE. Why?) vs. the simulation time in `xmgrace`. [3]

For a short guide on how to plot files with multiple columns in `xmgrace` see the Appendix to this tutorial.

Can you see how the energy drifts with the "loose" settings? Find out what were the the default criteria for these thresholds that were applied in the first simulation of this exercise (hint: you can find them in the FHI-aims output). How do they compare with the "loose" and "accurate" settings with respect to this energy drift?

[3] If you are more confortable with, e.g., `gnuplot`, please feel free to use it instead!

Ideally, there should be no energy drift, since the energy is conserved in the micro-canonical ensemble. The reason for this drift is that we leave the true Born-Oppenheimer surface if we don't converge well our electronic structure; this leads to an unphysical (and undesirable) energy drift.

Timing: ∼20 minutes total

Exercise 3: The importance of the time step size

Here, we will investigate the effects of the time step size for the integration of the equations of motion in a microcanonical molecular dynamics simulation. For a better illustration, we will not only consider the $H_5O_2^+$ molecule, but also its heavier, deuterated counterpart $D_5O_2^+$ [4].

Checkpoint files for both $H_5O_2^+$ and $D_5O_2^+$ from previous simulations where the molecules were thermalized are provided in the folder `exercise_3`. Use the same `control.in` (the "accurate" one) and `geometry.in` from the previous exercise Both $H_5O_2^+$ and $D_5O_2^+$ have the exact same electronic structure. This means that for converging the DFT self consistent cycle, the answer will be the same no matter the name or mass of the nuclei [5]. Therefore, one can use the same FHI-aims input files for both molecules. What will matter, though, is that in the evolution of the nuclei through Newton's equation of motion, the mass of the atom enters in the definition of the force and the propagation of the equations of motion. Therefore we will use different masses (corresponding to hydrogen and deuterion) in the checkpoint file that i-PI is reading in to initialize. If you wish to have a look at where this is specified, open that file and look for the block named `<m shape=... >` in which the units are atomic mass units.

An `input.xml` file is also provided. Please read and complete it in a appropriate way for each of the following tasks.

In order to speed up the exercise, each group will simulate either $H_5O_2^+$ or $D_5O_2^+$. The suggested choice of the molecule will be made clear (announced) by the tutors at hand.

Instructions

- Copy the "accurate" `control.in` and the `geometry.in` files from the last exercise.

  Do not add any flags that we do not mention! They are not needed and might hinder the performance of the calculation.

- Modify the `input.xml` file to run a 0.40 ps MD run in the microcanonical ensemble, using a 0.0005 ps ($\Delta t = 0.5$ fs) time step and run it [6].

- When it is done, plot the total energy vs. the simulation time with `xmgrace`.

- Increase the time step to 0.001 ps and run the simulation again, take the necessary precautions to not overwrite the files.

- Increase the time step to 0.002 ps and run the simulation again, take the necessary precautions to not overwrite the files. If FHI-aims aborts the run, you can create a file named EXIT in the folder where i-PI is running to stop it in a clean way, by typing `touch EXIT` in your terminal window.

- Together with the plot for $\Delta t = 0.0005$ ps, plot the total energy vs. simulation time in `xmgrace` for $\Delta t = 0.001$ ps and $\Delta t = 0.002$ ps. If you're simulating $D_5O_2^+$ please try also $\Delta t = 0.003$ ps.

  How do the energy fluctuations develop? What is happening for the $\Delta t = 0.002$ ps (or $\Delta t = 0.003$ ps) run?

[4] The deuterium atom has one proton and one neutron in its nucleus, thus about twice as heavy as hydrogen.

[5] In extreme cases changing the mass of the nuclei could have an effect, but it is not the case for just the addition of one neutron.

[6] Hint: There should be 800 steps.

From a practical point of view, a larger time step is desirable, since it allows to assess longer trajectories in shorter computational times. Notice, however, that the $\Delta t = 0.002\,\mathrm{ps}$ simulation diverges for $H_5O_2^+$. In fact, the molecule dissociates. You can inspect the dynamics of the molecule by opening `ex1.pos.xyz` in VMD (or Jmol, or Molden, or whichever program you prefer). The reason for the dissociation is that the integrator is unable to deal with these "big" time steps. This integrator uses a simple Verlet algorithm [3], where the error in the integrator goes with Δt^4. If you are simulating $D_5O_2^+$, the $\Delta t = 0.002\,\mathrm{ps}$ simulation does not diverge, although the energy fluctuations become very large. You can also look at the dynamics of this molecule in VMD.

Although such large energy fluctuations would already not be accurate enough for a production run with this molecule, the fact that it does not explode illustrates an important point: the largest Δt that can be used in a particular integration algorithm depends on the highest vibrational frequency of the system. Since the D atoms, being heavier, have a larger vibrational period (Do you understand why this is obvious? Think about a harmonic oscillator), the Δt that can be used for the integrator can also be larger.

Timing: ∼25 minutes total

The canonical ensemble

Exercise 4: Testing thermostats

Most "real-life" experiments cannot be done in a situation where the energy is explicitly kept constant, but where other quantities like the average temperature or pressure are maintained instead. As you may know, an ensemble where the temperature is kept constant is called a canonical ensemble. In order to simulate such an ensemble, the system has to be coupled to a heat bath. From a statistical mechanics point of view, the average kinetic energy in the canonical ensemble follows the equipartition theorem, which says that it is equally distributed on the various degrees of freedom of the system. Therefore, the momenta $p = mv$ follow the Maxwell-Boltzmann (MB) distribution:

$$P(|p|) = \left(\frac{\beta}{2\pi m}\right)^{3/2} 4\pi p^2 e^{-\frac{\beta p^2}{2m}}. \tag{1}$$

The instantaneous temperature $\tilde{T}$ is given by the relation $\tilde{T} = \frac{\sum_i^N p_i^2/m_i}{3Nk_B}$, where m_i is the mass of atom i and N is the number of atoms. This means that the instantaneous temperature is not constant but can (and should) fluctuate around the average value. The theoretical standard deviation is $\sigma^2 = \frac{2T_0^2}{3N}$, where T_0 is the target temperature, which should also be equal to the mean of the distribution, i.e. $T_0 = 2\langle K\rangle/(3Nk_B)$, where $\langle K\rangle$ is the average value of the kinetic energy.

We will here focus on stochastic schemes to simulate thermostats in molecular dynamics. We will include a brief description of them below, as well as the idea behind other types of thermostats that will not be studied in this exercise, but that are nevertheless quite popular.

1. *Stochastic velocity-rescaling thermostat (svr)*[4].
   In this algorithm, which takes on ideas of much older stochastic thermostats[3, 5] and velocity rescaling schemes[6], a deviation of the instantaneous kinetic energy is corrected in the following way:

   $$dK = \left[\overline{K} - K(t)\right]\frac{dt}{\tau} + 2\sqrt{\frac{K(t)\overline{K}}{N_f\tau}}\xi(t) \tag{2}$$

   where $\overline{K}$ is the target kinetic energy, $K(t) = p^2(t)/2m$ is the instantaneous kinetic energy, τ is the relaxation time of the thermostat, N_f is the number of degrees of freedom, and ξ is a white noise term (the derivative of a Wiener process[7]) that obeys $\langle\xi(t)\xi(t')\rangle = \delta(t-t')$.

   In practice the trajectory is first propagated for one time step with e.g. a velocity-verlet integrator and the new velocities are calculated as usual. Then, the new kinetic energy K is evaluated and the velocities are rescaled by a factor α such that:

   $$\begin{aligned}\alpha^2 &= e^{-\Delta t/\tau} + \frac{\overline{K}}{N_f K}\left(1 - e^{-\Delta t/\tau}\right)\left(R_1^2 + \sum_{i=2}^{N_f} R_i^2\right) \\ &+ 2e^{-\Delta t/2\tau}\sqrt{\frac{\overline{K}}{N_f K}\left(1 - e^{-\Delta t/\tau}\right)}R_1\end{aligned}$$

[7] An example of a Wiener process $W(t)$ is the Brownian motion (you might have heard about it...). $W(t)$ has the following characteristics: $\xi(0) = 0$; $W(t)$ is continuous; the increments are independent and $W(t_2) - W(t_1)$ is a Gaussian with average 0 and $\sigma = t_2 - t_1$.

where the R_i's are independent random numbers from a Gaussian distribution with unitary variance[8].

For this thermostat a conserved pseudo-Hamiltonian $\tilde{H}(t)$ can be defined:

$$\tilde{H}(t) = H(t) - \int_0^t \left(\overline{K} - K(t')\right) \frac{dt'}{\tau} - 2\int_0^t \sqrt{\frac{K(t')\overline{K}}{N_f \tau}}\xi(t')$$

where $H(t)$ is the total energy of the atomic system.

The `svr` thermostat yields the correct distribution of K, does not perturb the dynamics, and its accuracy and efficiency is rather independent of τ. Still, for (very) small non-ergodic isolated molecules it may still have some issues.

2. *Langevin thermostat*[7]

The Langevin thermostat is expressed through the following differential equation for the momenta (here in one dimension, without loss of generality):

$$\dot{p}(t) = -\frac{\partial V(R)}{\partial R} - \gamma p(t) + \sqrt{2m\gamma T}\xi(t) \tag{3}$$

where V is the potential energy, γ is the friction parameter and $\xi(t)$ is a stochastic variable distributed like a Gaussian white noise (just as in the svr thermostat). The friction term works like a drag that cools the system, while the random-noise term has a counter effect of heating the system. Since a system in the canonical ensemble must obey detailed-balance there is a relationship between the friction and the white noise that must be obeyed, often also called the fluctuation-dissipation relation which is given by $\langle \xi(t)\xi(0)\rangle = 2k_B T\gamma\delta(t)$. The Langevin equation describes, for example, Brownian motion – i.e. a Markovian (memoryless) stochastic differential equation. A Langevin thermostat disturbs dynamics considerably and is very sensitive on the value of γ, being hard to choose an optimal value for systems with degrees of freedom spanning a wide frequency range.

3. *Colored noise thermostats* [8]

Colored noise thermostats are an extension of Langevin thermostats; indeed they are also called Generalized Langevin Equation (GLE) thermostats. They are constructed via introducing auxiliary degrees of freedom $\mathbf{s}$ to the dynamics. These extra degrees of freedom model a Markovian process in higher dimensions, but give rise to non-Markovian dynamics when the fictitious degrees of freedom are integrated out. The equations of motion are:

$$\dot{R} = p/m \tag{4}$$

$$\begin{pmatrix}\dot{p}\\ \dot{s}\end{pmatrix} = \begin{pmatrix}-V'(R)\\ 0\end{pmatrix} - \mathbf{A}_p \begin{pmatrix}p\\ \mathbf{s}\end{pmatrix} + \mathbf{B}_p\left(\boldsymbol{\xi}\right), \tag{5}$$

where $\boldsymbol{\xi}$ is an array of uncorrelated Gaussian (white) noises, $V'(R)$ is the gradient of the potential and the $\mathbf{A}_p$ and $\mathbf{B}_p$ are matrices that obey the relation

[8] Note that $\sum_{i=2}^{N_f} R_i^2$ can be drawn directly from a suitable Gamma distribution

$$\mathbf{A}_p\mathbf{C}_p + \mathbf{C}_p\mathbf{A}_p^T = \mathbf{B}_p\mathbf{B}_p^T, \tag{6}$$

where $\mathbf{C}_p$ is the covariance matrix defined as $\mathbf{C}_p = \langle (p, \mathbf{s})^T (p,\ \mathbf{s}) \rangle$. By integrating out the $\mathbf{s}$ degrees of freedom, one gets dynamics of a non-Markovian process in the physical variables, with the EOM given by

$$\dot{R} = p/m \tag{7}$$

$$\dot{p} = -\frac{\partial V}{\partial R} - \int_{-\infty}^{t} Q(t-\tau)p(\tau) + \zeta(t), \tag{8}$$

where $\zeta(t)$ is a correlated noise and $Q(t-\tau)$ is a time (or frequency) dependent memory kernel which depends on the drift matrix $\mathbf{A}_p$. The fluctuation-dissipation theorem (and canonical sampling) is obeyed if $\langle \zeta(t)\zeta(0) \rangle = k_B T Q(t)$, or equivalently that $\mathbf{C}_p = k_B T \mathbf{1}$. Because these thermostats can directly act in different frequency ranges they can be more effective at thermalization of certain systems. One has to choose the parameters of the $\mathbf{A}_p$ matrix, which is fitted in the original papers [8] by optimizing functions that ensure, for example, a minimal correlation time for the potential energy on a wide frequency range.

4. *Extended Lagrangian approach: the Nosé-Hoover thermostat* [3, 9].
Even though we will not be simulating this kind of thermostats in the exercise, we show here a small explanation about this class of thermostats. Equations of motion derived from the Lagrangian of the system conserve the total energy of the system. One can write an *extended* Lagrangian, by adding fictitious degrees of freedom, such that the overall total energy is conserved but the atomic subsystem can span ensembles other than microcanonical. With the Nosé-Hoover Lagrangian, the atomic subsystem samples the canonical ensemble. The equations of motion of the Nose-Hoover thermostat are:

$$\dot{\mathbf{R}}_i = \mathbf{p}_i/m_i \tag{9}$$

$$\dot{\mathbf{p}}_i = -\frac{\partial V\left(\mathbf{R}^N\right)}{\partial \mathbf{R}_i} - \frac{\Pi \mathbf{p}_i}{Q} \tag{10}$$

$$\dot{\eta} = \frac{\Pi}{Q} \tag{11}$$

$$\dot{\Pi} = \left(\sum_i \frac{\mathbf{p}_i^2}{m_i} - \frac{g}{\beta} \right) \tag{12}$$

where g is the number of degrees of freedom of the system, V is the potential energy, Q the "thermostat mass", and $\mathbf{p}_i$ and m_i the momenta and masses of the ith particle of the system, respectively. The conjugated momentum Π of the extra coordinate η acts as a fluctuating drag parameter to the atomic subsystem. The conserved energy associated to the equations of motion is:

$$\mathscr{E} = \sum_i \frac{\mathbf{p}_i^2}{2m_i} + \mathscr{U}\left(\mathbf{r}^N\right) + \frac{1}{2}\frac{\Pi^2}{Q} + g\frac{\eta}{\beta} \tag{13}$$

In practice, most implementations of the Nosé-Hoover thermostat actually employ Nosé-Hoover chains, where several thermostats are coupled to each other giving rise

to coupled equations of motion with different masses Q_j. This scheme considerably ameliorates erogdicity issues present when a single Nosé thermostat is applied to the system in question [10].

5. *The "quantum" thermostat.* [8]
   As a last item in this exercise, we also present the so-called "quantum" thermostat, which is also a thermostat based on the GLE but is not a thermostat used to sample the canonical ensemble. Here, the fluctuation-dissipation theorem (FDT) is broken and one tries to enforce quantum statistics on the system (mainly trying to mimic zero-point energy contributions), based on for example, a harmonic approximation (note: FDT is equivalent to equipartition, thus breaking FDT implies breaking equipartition). Since the probability density distribution of the e.g. positions explored by a classical and a quantum harmonic oscillator are both Gaussian, one can transform one into the other by defining an effective temperature that is frequency dependent $T_{\mathrm{qm}} = \sum_i \frac{\hbar\omega_i}{2k_B} \coth\left(\frac{\hbar\omega_i}{2k_B T_{\mathrm{cl}}}\right)$. In order to ensure this different temperature distribution one has to input specifically both $\mathbf{A}_p$ and $\mathbf{C}_p (\neq k_B T \mathbf{1})$ matrices.

Instructions

- You will find some templates for the input files in the Exercise 4 folder. We let you choose whether you want to simulate $H_5O_2^+$ or $D_5O_2^+$ – you can keep what you did in the previous exercise.

  You should use a time step of **0.5** fs, which corresponds to the following line in the i-pi input.xml file,

  ```
  <timestep units="femtosecond"> 0.5 </timestep>
  ```

  and we will be simulating all systems at 300 K:

  ```
  <temperature units="kelvin"> 300 </temperature>
  ```.

  Since we are simulating a canonical ensemble, the dynamics mode will be set to "nvt":

  ```
  <dynamics mode="nvt">
  ```

- The `control.in` file provided in this folder is tailored to run fast (and inaccurate) simulations, so that this exercise can be done within the time frame proposed here. **Please do not use this type of `control.in` file for real production calculations.**

- Each group will run only one of the four different thermostats discussed above, and the division will be made clear by the tutors at hand.

  Two of these thermostats, namely the Stochastic Velocity Rescaling and the Langevin ones, have parameters that you can play with. In order to obtain reasonable results, one should provide an educated guess for the value of these thermostats' parameters. In order to realize to which extent such user-given parameters can influence a simulation, you will be asked to try different values for these parameters, as explained below.

  1. svr
     ```
     <thermostat mode="svr">
     <tau units="femtosecond"> xx </tau>
     </thermostat>
     ```

Here tau is a relaxation time of the thermostat, and is given in femtoseconds. Its value has to be chosen by the user (i.e., you !). Even though in principle, for a sufficiently long simulation time, the thermalization will work, the performance of the thermostat depends on the value of tau[9]. In order to gauge the influence of this parameter, we ask you to test two different values for tau: one value which should yield a correct behavior, for example $\tau = 2$, and one more extreme value, for example $\tau = 500$ or $\tau = 0.005$.

2. langevin

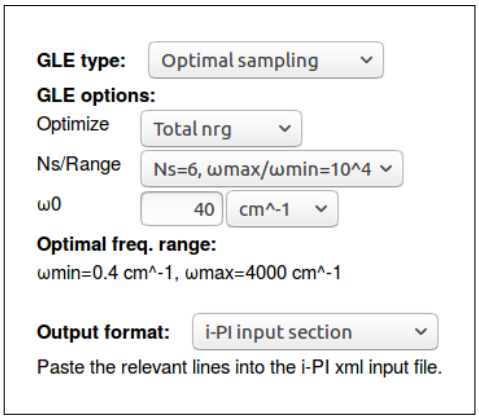
```
<thermostat mode="langevin">
<tau units="femtosecond"> xx </tau>
</thermostat>
```

Just like for the SVR thermostat, you have to choose a value for tau.

3. gle

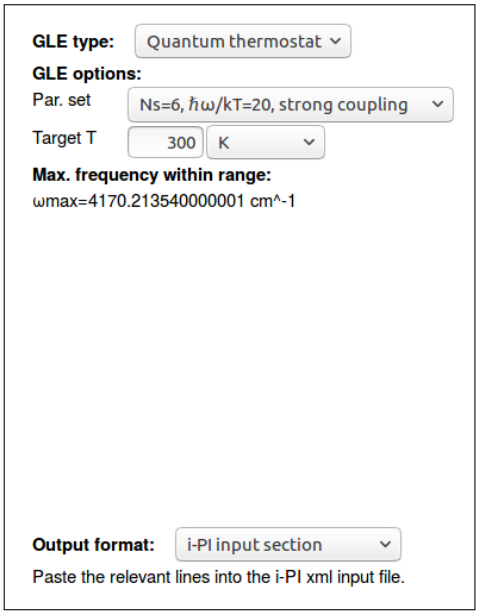
The Generalized Langevin Equation provides a very flexible framework to manipulate the dynamics of a classical system, improving sampling efficiency and obtaining quasi-equilibrium ensembles that mimic quantum fluctuations.

This thermostat takes a matrix as an input parameter. It can be generated at http://gle4md.org/index.html?page=matrix.
Choose the parameters in the website like it is shown in the picture, and copy them to the appropriate section of your input.xml file.



4. Colored noise thermostat

The parameters for this thermostat are quite complex. We will here use it to approximate nuclear quantum effects (that is, we will not simulate a canonical ensemble), so that the matrices $\mathbf{A}$ and $\mathbf{C}$ (Eq. 5) will have to be given as an input. They can be generated at http://gle4md.org/index.html?page=matrix.
Choose the parameters in the website like it is shown in the picture, and copy them to the input.xml file.



[9]Note that the influence of tau is similar to the mass parameter Q for the Nosé-Hoover thermostat.

- While waiting for the simulations to complete, you are challenged to demonstrate Eq. 1 and calculate what would be the temperature corresponding to a zero point energy ($\hbar\omega/2$) for a frequency $\omega_1 = 3000$ cm^{-1} and for $\omega_2 = 100$ cm^{-1} considering a system with only 1 degree of freedom. Also show which frequency ω corresponds to a temperature of 300K.
- Use the script get_properties.py to extract different properties, like the temperature, the kinetic energy and the potential energy:

  ```
  python get_properties.py 'property' 'ipi-output'
  ```

  where 'property' is a placeholder for the property you want to extract. You can choose from 'temperature', 'kinetic', 'potential' and''conserved'. This will output a single-column file containing the desired property at every step of the simulation. Plot each of these quantity. Is it what you were expecting ?
- Let us analyze a bit further the output. Use the script `histogram` in order to obtain the temperature distribution of your system:

  ```
  histogram -xi 0 -xf 600 < mean_temperature.dat > histo.dat
  ```

  (Here we assumed 'temperature.dat' is your single-column file containing the temperature of the system at each time step.) What is the shape of the distribution ? Why ?

  Another way to analyze the behavior of the temperature of your system is to look at the cumulative average, that is, the average temperature over time computed at each step of the simulation. You can do this by running the following script:

  ```
  python cumul_avg.py 'mean_temperature.dat'
  ```

  Can we assert that the system is correctly thermalized ?

  Now use the script "autocorr" to calculate the autocorrelation function of the potential energy:

  ```
  autocorr -maxlag 1000 < 'potential.ex4.out.dat' > 'autocorr.dat'
  ```

  Plot the autocorrelation function against time. What is the decorrelation time ? (How) does it change with respect to tau ?
- Calculate the instantaneous temperature of each atomic species for each thermostat you tried by using the following script:

  ```
  python get_temp_species.py 'ipi-output' O H
  ```

  where 'O' and 'H' define the oxygen and hydrogen species. How do the different thermostats work in each case? Do the thermostats couple efficiently to the system? Can you understand the result for the quantum thermostat? (Hint: different vibrational frequencies involve different atoms and give rise to different zero point energies)
- The script "get_rdf_h5o2.py" writes the radial distribution function $g(r)$, decomposed into OH, OO, and HH pairs. Run the script on each **FHI-aims output** by typing:

  ```
  python get_rdf_h5o2.py 'output-aims'
  ```

  It will output three files, starting with 'rdf_oo', 'rdf_oh' and 'rdf_hh'. The first column is the distance (in Ångstroms) and the second column the count. Plot these files for each thermostat you used and compare the results for each $g(r)$. For which $g(r)$ of you see the larger differences between the classical and "quantum thermostat" distribution?

Time estimation: ∼1h

Exercise 5: Free energy estimation via Thermodynamic Integration.

While the potential energy surface is shaped by the $3N$ coordinates (degrees of freedom) that describe a molecule, the *free* energy, which is the quantity actually accessible in experiments, is a function of thermodynamical variables (temperature, pressure, entropy, volume, etc.). This is the quantity of fundamental interest for comparison with experiments and the one that rules all dynamics of the system. In order to perform the task of calculating free energies, it is necessary to define the partition function $Z(T)$ of the system of interest. With respect to the canonical partition function $Z(T)$, the Helmholtz free energy can be written as [11]:

$$F(T) = -k_B T \ln[Z(T)], \tag{14}$$

where k_B is the Boltzmann constant and T is the temperature.

In the harmonic-oscillator approximation, the partition function can be written as a product of the several vibrational energy levels, weighted according to the Bose-Einstein statistics to take into account the quantum nature of the nuclei

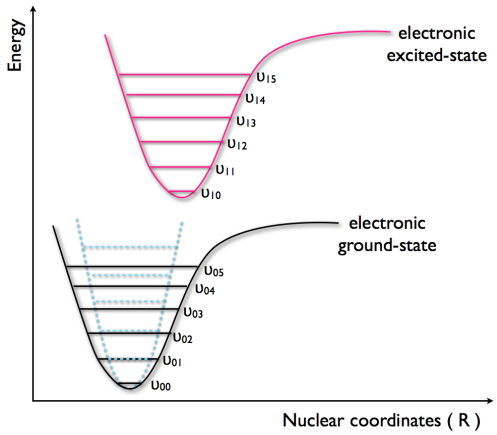


Figure 1: Schematic drawing of electronic and vibrational states of a hypothetical molecule. The Morse-like curves in black and pink represent the electronic ground-state and the first excited state, respectively. The levels drawn in each of them correspond to the possible vibrational frequencies v_{0n}, v_{1n}. The dashed parabola-like blue line represents the harmonic approximation, with equally spaced vibrational levels.

$$Z_{\text{vib}}^{\text{ha}}(T) = \prod_{i=1}^{3N-6} \frac{e^{-\frac{\hbar\omega_i}{2k_BT}}}{1 - e^{-\frac{\hbar\omega_i}{k_BT}}}, \tag{15}$$

where ω_i are the normal modes of vibration of the molecule and the product runs over all modes except the ones corresponding to translations and rotations [10]. Substituting 15 in 14, we get the following expression for the vibrational contributions to the harmonic free energy, $F_{vib}^{\text{ha}}(T)$:

$$F_{vib}^{\text{ha}}(T) = \sum_{i=1}^{3N-6} \left[\underbrace{\frac{\hbar\omega_i}{2}}_{\text{Zero Point Energy}} + k_BT \ln\left(1 - \exp^{-\frac{\hbar\omega_i}{k_BT}}\right) \right]. \tag{16}$$

There are limitations to the use of the harmonic approximation. It is not expected to be valid at all temperatures, due to its harmonic nature: When the atoms start to explore higher regions of the potential well, it cannot be approximated as a parabola. The "real" anharmonic modes are more closely spaced than what is estimated by the harmonic

[10] One can also include the contributions to the partition function and the free energy arising from rotations and translations (see Ref. [11] for possible approximations for these quantities), and in real systems these may not even be separable. In this exercise we will focus on only vibrational contributions, though.

approximation, as can be pictorically seen in Figure 1. One way to obtain anharmonic corrections to the harmonic free-energy will be the subject of this exercise.

From elementary thermodynamics, in the canonical (NVT) ensemble, one can write

$$\frac{\partial\left(\beta F\right)}{\partial\beta} = \langle U\rangle_{NVT}\,, \tag{17}$$

where $\beta = 1/(k_B T)$, F is the Helmholtz free energy, and U is the internal energy (potential plus kinetic energy) of the system and the brackets $\langle\ldots\rangle_{NVT}$ denote the canonical average. We can then evaluate the free energy at a temperature T by integrating Eq. 17

$$\frac{F(T)}{k_B T} \;=\; \frac{F_0(T_0)}{k_B T_0} + \int_{\beta_0}^{\beta} d\beta' \langle U\rangle_{\beta'}, \tag{18}$$

where $F_0(T_0)$ is the free energy at a reference temperature. The term $\langle U\rangle_{\beta'}$ can be written as a sum of two contributions, namely,

$$\langle U\rangle_\beta = \langle U^{\mathrm{ha}}\rangle_\beta + \langle \Delta U^{\mathrm{an}}\rangle_\beta, \tag{19}$$

where $\langle \Delta U^{\mathrm{ha}}\rangle_\beta$ is the ensemble average of the harmonic contributions to the internal energy (which we know how to solve analytically) and $\langle \Delta U^{\mathrm{an}}\rangle_\beta$ is the ensemble average of the anharmonic contributions.

So far, all the presented equations are exact. In practice, the integral in Eq. 18 is evaluated numerically by sampling $\langle U\rangle_\beta$ for a discrete set of temperatures from T_0 to the desired T. Then, a numerical integration can be performed. Now we make an approximation, namely that our system can be approximated as harmonic at T_0, so that $F_0(T_0) \approx F_0^{ha}(T_0)$. This approximation is best when T_0 is a low temperature. Then, one can use equations 18 and 19 to write

$$F(T) = F_{vib}^{\mathrm{ha}}(T) + k_B T \int_{\beta_0}^{\beta} d\beta' \langle \Delta U^{\mathrm{an}}\rangle_{\beta'}, \tag{20}$$

where $F_{vib}^{\mathrm{ha}}(T)$ is given by equation 16.[11]

We will consider two approaches to evaluate the ensemble average of ΔU^{an}. First, we will consider MD simulations with classical nuclei as we did in the previous exercises. Then, we will use path integral molecular dynamics (PIMD). As was discussed in the lecture, PIMD is classical molecular dynamics in an extended space: Each quantum particle is mapped onto a ring polymer of classical particles, which consists of n repetitions of the original system connected by harmonic springs. This method thus increases the cost of the calculation by n. Since the harmonic forces of the springs are easily and analytically evaluated, the method is also trivially parallelizable. Note that in the first case we make a further approximation, where we consider the anharmonic contributions in a classical framework, but add their effect to the "quantum" harmonic free energy of equation 16.

[11] The careful reader will realize that the free energy in Eq. 20 is the total free energy and not the vibrational one. This is because the second term in the right hand of that equation includes the rotational contribution. However, a term is still missing. We neglect it for simplicity and it does not change any conclusion to this exercise. If you are so curious that you have reached this point, could you guess which term is it before that you continue reading?
Find the answer in the reference [13]

Also, in this exercise we will use an interesting feature of i-PI, which is that several different clients can be connected to it, and they do not need to be electronic structure codes. In this case, we will use the `driver.x` code, which is distributed with i-PI and has many empirical or parametrized potentials coded into it. We will be using the potential from Ref. [12], that was parametrized on a calculated CCSD(T) surface for the Zundel cation, and provides much faster force evaluations than solving explicitly the Kohn-Sham equations as we were doing until now.

Finally, instead of providing you one file that performs all the post processing like a black box, we provide you several scripts (`exercise_5/scripts/`) that perform specific tasks so you understand the procedure step by step:

1. `F_ha.py`
   - **Function:** Computes the vibrational Helmholtz free energy up to a given temperature within the harmonic approximation
   - **Syntax:** `python proc.py <InputFreq_file> <Max_Temperature>`
   - **Input:** The <InputFreq_file> is the name of the file with the frequencies list (provided)
2. `U_ha.py`
   - **Function:** Computes the total internal energy (classical and quantum) up to a given temperature within the harmonic approximation
   - **Syntax:** `python proc.py <InputFreq_file> <Max_Temperature>`
   - **Input:** The <Input_Freq_file> is the name of the file with the frequencies list (provided)
3. `extract_mean.py`
   - **Function:** Computes averages value and standard deviation from an i-PI output file
   - **Syntax:** `python proc.py <name_input> <mode>`
   - **Mode options:** T (temperature), V (potential energy), K (Kinetic energy), K+V (Internal energy, U = K+V)
   - **Input**: i-PI output file
4. `integrate.py`
   - **Function:** Computes the second term of the equation 20 including the $k_B T$ factor
   - **Syntax:** `python proc.py <name_input> <number of points>`
   - **Input:** Two columns format. First column corresponds to the temperature and second column corresponds to $\langle \Delta U^{\mathrm{an}} \rangle_\beta$. (Every line starting with # is discarded)

Instructions

- Look at the folder `exercise_5/data/`. There you can find three files: `MD.dat`, `H5O2_PIMD.dat` and `D5O2_PIMD.dat`. (Why do we not specify `H5O2_MD.dat` or `D5O2_MD.dat` ?). Open one of them. You will see the data in two columns. The first one is the temperature T(K) and the second one the anharmonic contribution to the average internal energy $\langle \Delta U^{\mathrm{an}} \rangle_\beta$. You will notice that the table is not complete, your job is to complete it.

- Again, in order to speed up the exercise, each group will simulate either $H_5O_2^+$ or $D_5O_2^+$. The suggested choice of the molecule will be made clear (announced) by the tutors at hand.
- In order to compute the missing values you have to run MD or PIMD simulations. We provided the corresponding templates in the folder `exercise_5/skeleton/`. In particular, we provided one `input.xml` and one checkpoint file with an equilibrated geometry for each simulation. Please take your time in order read and understand the `input.xml` files: Which differences can you identify? Do you understand them?
- Here is how you will run i-PI and the `driver` code with the parametrized potential for the Zundel cation:

  In order to start the simulation type:

  ```
  i-pi input.xml > output-i-PI &
  ```

  (Wait 5 seconds in order to let i-PI do the initialization)

  Open a second terminal and start:

  ```
  driver.x -u -m zundel -h <address> &
  ```

  – ATTENTION: the `<address>` in the previous command MUST coincide with the address specified in the i-PI input file.

  – For PIMD simulations you can (should) run several instances of the client at the same time. That way each replica of your system will be calculated by different instances of the `driver` code. In order to do that please run

  ```
  for i in {1..8}; do driver.x -u -m zundel -h <address> > out_$i &
  done
  ```

- Go to `exercise_5/skeleton/1_MD/200K` and run the MD simulation. Then, go to `2_H5O2_PIMD/300K/` or `3_D5O2_PIMD/250K/` and run the PIMD simulation. **You can run both simulations at the same time.**
- *During the waiting time of this exercise, you may try to demonstrate Equation 20.*
- When all the simulations are over, you have to do the post processing
- Use the provided script `extract_mean.py` to compute the averages of internal energy for each simulation
- Use the provided script `U_ha.py` to compute the internal energy in the harmonic approximation. Since the MD simulations treat the nuclei as *classical* particles, the reference internal energy must be the harmonic classical energy, given by the equipartition theorem $U^{\mathrm{ha}}(T) = (3N-6)k_BT$. For the PIMD case the reference internal energy must be the quantum harmonic internal energy. Luckily the script provides you both of them, so take the appropriate value for each case.
- Calculate the $\langle \Delta U^{\mathrm{an}} \rangle_\beta$ (equation 19) and complete the tables
- Use the provided script `F_ha.py` to compute the Harmonic Free energy, $F_{\mathrm{vib}}^{\mathrm{ha}}(T)$

- Use the provided script `integrate.py` to perform the integrations . Remember that with the integration you are getting the anharmonic corrections, so to get the total free energy, $F(T)$, you have to add the harmonic contribution (equation 20)
- Plot $F(T)$ and $F_{\mathrm{vib}}^{\mathrm{ha}}(T)$ as a function of temperature (in xmgrace, for example)

  What are the differences? Can you estimate up to which temperature the harmonic approximation should be a good approximation? Are the anharmonicities important for these systems? And the nuclear quantum effects?
- Fun question if you want to try at home: Can you reproduce the result obtained with PIMD by instead performing MD with classical nuclei but employing the quantum thermostat of the previous exercise?

Timing: ≈1h20min total

References

[1] M. Ceriotti, J. More, D. E. Manolopoulos, Comput. Phys. Comm. **185** 1019 (2013).

[2] R. P. Feynman, and A. R. Hibbs *Quantum Mechanics and Path Integrals*; McGraw-Hill: New York, 1965.

[3] D. Frenkel and B. Smit, *Understanding Molecular Simulation: from algorithms to applications*, second edition, Academic Press 2002.

[4] G. Bussi, D. Donadio, and M. Parrinello, J. Chem. Phys. **126**, 014101 (2007).

[5] H. C. Andersen, J. Chem. Phys. **72**, 2384 (1980).

[6] H. J. C. Berendsen et al., J. Chem. Phys. **81**, 3684 (1984).

[7] M. E. Tuckerman, *Statistical Mechanics: Theory and Molecular Simulation*; Oxford University Press 2010.

[8] M. Ceriotti, G. Bussi, and M. Parrinello, J. Chem. Theory Comput. **6**, 1170 (2010).

[9] S. Nosé, J. Chem. Phys. **81**, 511 (1984). W.G. Hoover, Phys Rev. A **31**, 1695 (1985).

[10] G. Martyna, M. Klein, and M. Tuckerman, J. Chem. Phys.**97** 2635 (1992).

[11] D. McQuarrie, *Statistical Mechanics*, University Science Books, 1st. ed., (2000).

[12] X. Huang, B. J. Braams, and J. M. Bowman, J. Chem. Phys. **122**, 044308 (2005).

[13] The answer is the rotational contribution at T_0.

Plotting files in xmgrace

In order to plot files containing multiple Y columns, follow these steps:

1. Open xmgrace and click on Data → Import → ASCII (Figure 2).
2. Find the file that you want to plot and choose "Load as Block Data" in the dialogue box that will open (Figure 3).
3. Set X from column 1 and Y from whatever other column you want to plot in the new dialogue box that will open (Figure 4).
4. Press Apply and then press Close on all dialogues.

Properties of the plots can be changed by double clicking on the data or on the axis.

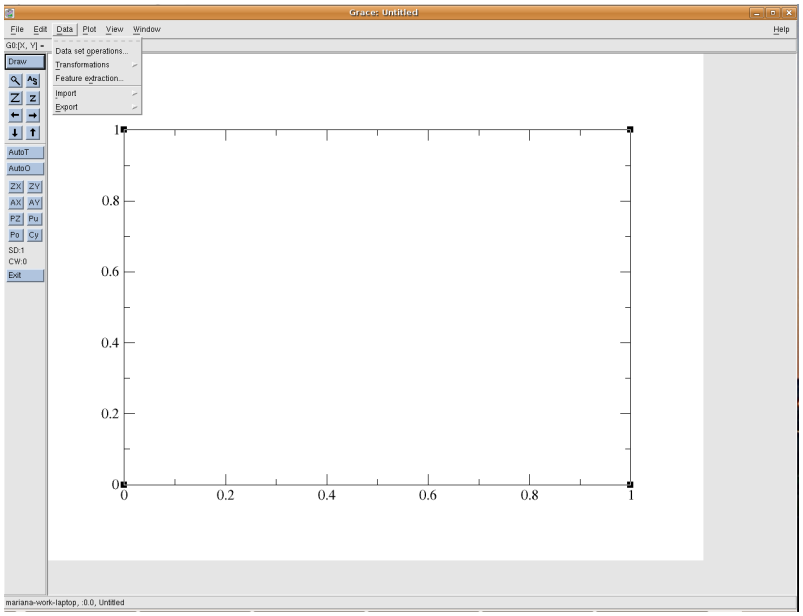


Figure 2: Step 1 - Open xmgrace and click on Data → Import → ASCII

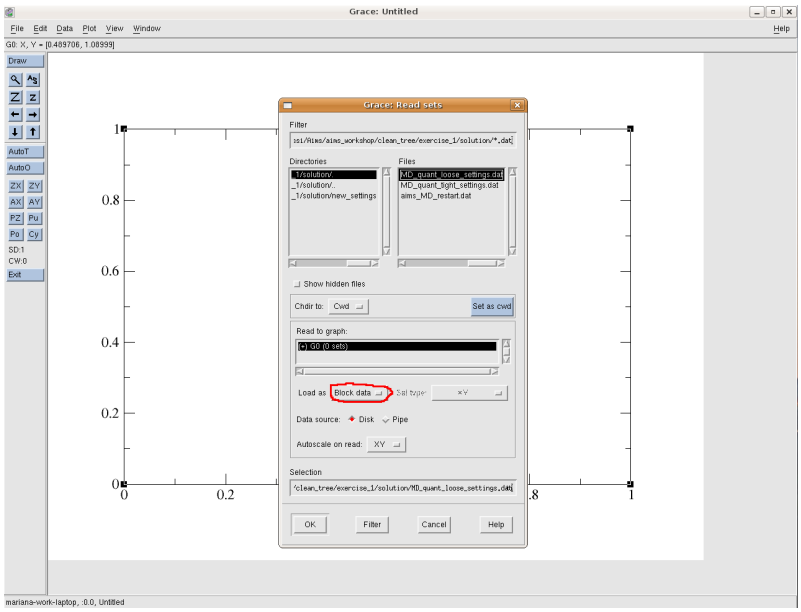


Figure 3: Step 2 - Find the file that you want to plot and choose “Load as Block Data” in the dialogue box that will open.

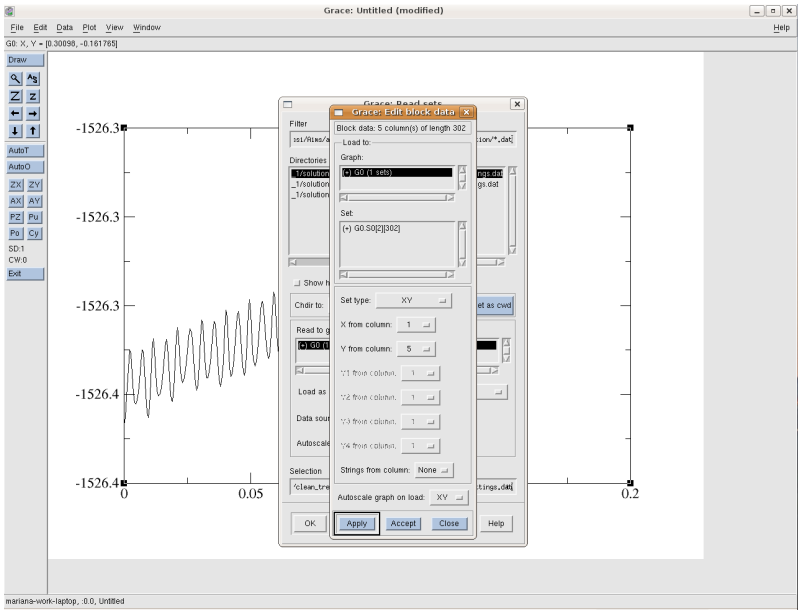


Figure 4: Step 3 - Set X from column 1 and Y from whatever other column you want to plot in the new dialogue box that will open.