Simulating the nonequilibrium thermodynamics of quantum dots
Introduction to Stochastic Thermodynamics and Stochastic Simulations

Bonamino Luca

Contents

1 Introduction **2**
- 1.1 Equilibrium and non-equilibrium thermodynamics 2
- 1.2 Stochastic thermodynamics 4
  - 1.2.1 Dynamics 4
  - 1.2.2 Ensemble-level description 6
  - 1.2.3 Trajectory-level description 9
- 1.3 Quantum dots 13
  - 1.3.1 Assumptions 13
  - 1.3.2 Evolution of occupation probability 14

2 Algorithms for Stochastic Simulations **15**
- 2.1 Discrete-time simulation algorithm 16
  - 2.1.1 Quantum dot in contact with one reservoir 18
- 2.2 Gillespie algorithm 21
  - 2.2.1 Sampling from a known distribution 21
  - 2.2.2 Algorithm 22
  - 2.2.3 Stationary single-level quantum dot in contact with two reservoirs 23

3 Thermodynamics of Quantum Dots **29**

Chapter 1

Introduction

1.1 Equilibrium and non-equilibrium thermodynamics

When dealing with equilibrium thermodynamics, we are only allowed to look at our system in an equilibrium state. The way to ensure constant equilibrium of it, is to make it in contact with a thermal bath called *reservoir* at temperature T .
The dimensions of the reservoir compared to the system's, ensures that any change in its internal state, will be instantly damped by its components and therefore its temperature and its macroscopic quantities will remain constant allowing constant equilibrium of the system in contact with it.
Only under these condition we are allowed to define macroscopic quantities on the system, which are the ones that we intend to study.

Allowing system-reservoir contact is however not enough to ensure equilibrium if the system undergoes an external driven transformation. We have to give it the time to adapt to the reservoir conditions and get back to equilibrium with it at each change. This is why we use quasi-static transformations, which are theoretically done in a time infinitely long for this to be satisfied. Quasi-static transformations allow us to work with variations of the macroscopic quantities and therefore formulate the first law of thermodynamics in therms or state functions

$$dE = \delta W + \delta Q = -pdV + \sum_i \mu_i dN_i + \delta Q \tag{1.1}$$

Because of the slowness of these transformations and the need to escape the restriction of the single reservoir, we find the need to take a step further. This can be done with the assumption of local equilibrium.
This assumption basis itself on the fact that even if the entire system is not in equilibrium as a whole, each infinitesimally small subdivisions of it can be

considered in local equilibrium with the subdivisions around it. This enables us to still talk about macroscopic quantities and therefore, to formulate the laws of thermodynamics in terms of quantities whose infinitesimal time steps are considered as in equilibrium.

The transformations that are able to satisfy these conditions are the linear irreversible transformations.
From these transformations the second law of thermodynamics can be formulated taking in to account the entropy production rate $\delta S_i > 0$ due to the fact that system is not in equilibrium through out the transformation.

$$dS = \delta S_i + dS_e \tag{1.2}$$

Linear irreversible transformations allow the presence of multiple reservoirs, but still they do not work for microscopical systems which are easily perturbed compared to the macroscopic one. Moreover in microscopic systems, the fluctuations are very large, which results in a lot a big loss of information only looking at the average behaviour, hence we need a theory that is able to account these oscillations.
Here is where stochastic thermodynamics come to our help. It enables us to do thermodynamics on microscopical systems far from equilibrium.

In this introduction, I will go through the main concept of stochastic thermodynamics of jump processes and introduce the *quantum dots* and explain why the are useful for our purpose.
In the next two chapters, I will discuss two main examples on stochastic thermodynamic systems using quantum dots and verify the validity of the results with computer simulations.

1.2 Stochastic thermodynamics

In this section I introduce the stochastic dynamics and reformulate the two laws of thermodynamics in such framework.

We consider a system where arbitrary many states are allowed. The system has a time-dependent probability to be in a state, we want to describe the evolution in time of this probability using stochastic dynamics.

1.2.1 Dynamics

Stochastic Process and Markov assumption

Any system that evolves probabilistically in time is governed by a stochastic dynamics.

The state of the system $n(t)$ is a random quantity called *stochastic process* that can assume any values in the set of discrete states S. $n(t)$ describes a trajectory that the system follows jumping from one state to another. It takes for example the measures $n_1, n_2, n_3, n_4, ...$ at timed $t_1, t_2, t_3, t_4, ...$
The joint probability on $n(t)$ is

$$p(n(t)) = p(n_1, t_1; n_2, t_2; ....) \tag{1.3}$$

and the conditional probability

$$p(n_k, t_k; n_{k-1}, t_{k-1}; .....; n_j, t_j | n_{j-1}, t_{j-1}; .....; n_1, t_1) = \frac{p(n(t))}{p(n_{j-1}, t_{j-1}; .....; n_1, t_1)}, \ n_k, n_j \in S$$

Assuming the ordering $t_N \geq .... \geq t_k .... \geq t_j ... \geq t_2 \geq t_1$, a stochastic process is a *Markovian process* if the conditional probability is entirely determined by the most recent condition. The Markov assumption, reads

$$p(n_k, t_k; ....; n_j, t_j | n_{j-1}, t_{j-1}, ..., n_1, t_1) = p(n_k, t_k; ...., ; k_j, t_j | n_{j-1}, t_{j-1}) \tag{1.4}$$

Combining the conditional probability and the markov condition we find the *Champman-Kolmogorov equation*

$$p(n_k, t_k; n_1.t_1) = \sum_{i \in S} p(p_k, t_k | i, t_i) p(i, t_i | n_1, t_1) p(n_1, t_1) \tag{1.5}$$

which tell us that the probability to have a trajectory that present the states n_1 and n_k is given by the sum of the probabilities of transitions between all the intermediate states.

Master Equation

Choosing the system to be at the state n at the time t and at the state m at the time $t + dt$ for an infinitesimal dt and defining as W_{nm} the probability per unit time to jump from m to n ($W_{nm}(t) = 0,\ \forall t\ if\ n = m$), we compute the probability of a jump as

$$p(n, t + dt|m, t) = \left(1 - \sum_{n\neq m} W_{nm}(t)dt\right)\delta_{nm} + W_{nm}(t)dt \tag{1.6}$$

where the term $\sum_{n\neq m} W_{nm}(t)dt$ denotes the probability to have any jump from the state m.
Hence

$$p(n, t + dt|m, t) = \begin{cases} W_{nm}(t)dt, & \text{if } n \neq m \\ 1 - \sum_{n\neq m} W_{nm}(t)dt, & \text{if } n = m \end{cases}$$

Substitung in Eq (1.5)

$$\begin{aligned} p(n, t + dt) &= p(n, t + dt|n, t)p(n, t|n, t)p(n, t) + \sum_{m\neq n} p(n, t + dt|m, t)p(m, t|n, t)p(n, t) \\ &= p(n, t + dt|n, t)p(n, t) + \sum_{m\neq n} p(n, t + dt|m, t)p(m, t) \end{aligned}$$

Hence

$$p(n, t + dt) = \left(1 - \sum_{m\neq n} W_{mn}(t)dt\right) p(n, t) + \sum_{m\neq n} W_{nm}(t)p(m, t)dt \tag{1.7}$$

By rewriting (1.7)

$$\frac{p(n, t + dt) - p(n, t)}{dt} = -\sum_{m\neq n} W_{mn}(t)p(n, t) + \sum_{m\neq n} W_{nm}(t)p(m, t)$$

$$\iff \frac{dp(n, t)}{dt} = W_{n,n}(t)p(n, t) + \sum_{m\neq n} W_{nm}(t)p(m, t)$$

Hence, we find the *Master equation*

$$\frac{dp(n, t)}{dt} = \sum_{m} W_{nm}(t)p(m, t) \tag{1.8}$$

that describes the evolution of the probability of being in the state n in time.
To simply the notation, we note $p(n, t) \equiv p_n(t)$

We note that by normalisation of the probability $\sum_n p_n(t) = 1$, hence

$$\sum_n \dot{p}_n(t) = \sum_n \sum_m W_{nm} p_m(t) = \sum_m p_m(t) \sum_n W_{nm}(t) = 0$$

Hence

$$\sum_n W_{nm}(t) = 0, \ \forall t \tag{1.9}$$

In general, the jumps between states are due to different time-dependent external mechanisms that can be externally controlled with a mechanism λ_t

$$W_{nm}(\lambda_t) = \sum_\nu W^{(\nu)}_{nm}(\lambda_t) \tag{1.10}$$

The master equation (1.8) is the feature that enables us to say everything that is written in this document concerning the dynamics of the system we will consider.

1.2.2 Ensemble-level description

What we stated in the section 1.2.1 applies correctly to many stochastic models of different fields. To restrict ourselves to the world of physics we need to give a physical meaning to the mechanisms that are reflected by the rates. We think at these mechanisms as the reservoirs that try to impose its regime on the system. The rates are a result of this mechanisms and can be further controlled by an external protocol λ_t.
Hence, we think as the rate W^ν_{nm} as governed by the reservoir ν and by an external protocol λ_t.
We consider a system of ensemble energy $E(t) = \sum_n \epsilon_n(t) p_n(t)$ (the average over the energy of the levels with the probability of being in each level) and number of particles $N(t) = \sum_n N_n p_n(t)$ (the average over the number of particles in the levels with the probability of being in each level).
What really gives the physical meaning to the mathematical theory is the *local detail balance* assumption

$$\frac{W^\nu_{nm}(\lambda_t)}{W^\nu_{mn}(\lambda_t)} = e^{-\beta^\nu [\epsilon_n(\lambda_t) - \epsilon_m(\lambda_t) - \mu_\nu (N_n(t) - N_m(t))]}, \quad \beta^\nu = (k_B T_\nu)^{-1} \tag{1.11}$$

T_ν is the temperature of the reservoir ν and $\epsilon_m(\lambda_t) - \epsilon_n(\lambda_t) - \mu_\nu \left(N_m(t) - N_n(t)\right)$ is the elementary heat exchange in the process $m \to n$.
If the conditions are such that $p_n(t) = p_m(t) = p^{eq}$

$$W^\nu_{nm} p^{eq}_n = W^\nu_{mn} p^{eq}_m, \ \forall \nu \tag{1.12}$$

we say that the system is *detail balanced* - in equilibrium.

Absence of external driving force

We can justify the local detailed balance assumption in absence of any external driving force by saying that each reservoir tries to impose its equilibrium, from which

$$W^{\nu}_{nm} p^{\nu}_{n} = W^{\nu}_{mn} p^{\nu}_{m}$$

where p^{ν}_{n} is the gran canonical distribution due to the reservoir ν

$$p^{\nu}_{n} = \frac{e^{-\beta^{\nu}\epsilon_n - \mu_\nu N_n}}{Z^{\nu}}$$

where

$$Z^{\nu} = e^{-\beta^{\nu} G}$$

where $G = E - T_\nu S - \mu N$ is the Gibbs free energy.
Hence, the local detailed balance condition

$$\frac{W^{\nu}_{nm}}{W^{\nu}_{mn}} = \frac{p^{\nu}_{m}}{p^{\nu}_{n}} = e^{-\beta^{\nu}[\epsilon_n(t) - \epsilon_m(t) - \mu_\nu (N_n(t) - N_m(t))]}$$

Moreover, in absence of external force $W^{\nu}_{nm} \neq W^{\nu}_{nm}(t)$ a stationary solution p^{ss} (such that $\dot{p}^{ss} = 0$) exists.

First Law of Thermodynamics

The energy balance expression is found by time-deriving the ensemble energy E, hence

$$\dot{E}(t) = \sum_n \dot{\epsilon}_n(t) p_n(t) + \sum_n \epsilon_n(t) \dot{p}_n(t) \tag{1.13}$$

The first term of (1.13) takes into account variation of the energy of the levels, which is a quantity that is externally controlled. Hence, the quantity $\sum_n \dot{\epsilon}_n p_n$ corresponds to the mechanical work rate $\dot{w}(t)$ done on the system.
The second term takes into account the variation of the probability of the states $p_n(t)$ which is a consequence of transitions between the states that are due to the reservoirs. Hence the quantity $\sum_n \epsilon_n \dot{p}_n$ corresponds to the sum of a chemical work rate $\dot{w}_{chem}(t)$ done on the system and the heat exchange rate $\dot{Q}(t) = \sum_\nu \dot{Q}^{\nu}(t)$ between the reservoirs and the system.
By noticing that $W^{\nu}_{nm}(t) p_m(t)$ is the probability per unit time to have a transition induced by the reservoir ν from the state m to n being on in the state n and that $\epsilon_m(t) - \epsilon_n(t) - \mu_\nu \left(N_m(t) - N_n(t)\right)$ is the elementary heat system-reservoir exchange associated to the transition,

$$\dot{Q}^{\nu}(t) = \sum_{n,m} W^{\nu}_{nm}(t) p_m(t) \left([\epsilon_m(t) - \epsilon_n(t) - \mu_\nu \left(N_m(t) - N_n(t)\right)]\right) = \dot{Q}^{\nu}_{\epsilon}(t) - \dot{Q}^{\nu}_{matter}(t) \tag{1.14}$$

where

$$\dot{Q}^{\nu}_{\epsilon}(t) = \sum_{n,m} W^{\nu}_{nm}(t) p_m(t) \left(\epsilon_m(t) - \epsilon_n(t)\right)$$

$$\dot{Q}^{\nu}_{matter}(t) = \sum_{n,m} W^{\nu}_{nm}(t) p_m(t) \mu_\nu \left(N_m(t) - N_n(t)\right) = \dot{w}^{\nu}_{chem}(t)$$

The chemical work correspond to the net system-resevoirs jumps.

$$\dot{w}_{chem}(t) = \sum_{\nu} \dot{w}^{\nu}_{chem}(t) = \sum_{n,m,\nu} W^{\nu}_{nm}(t) p_m(t) \mu_\nu \left(N_m(t) - N_n(t)\right) \tag{1.15}$$

Second Law of Thermodynamics

The entropy S of the system is given by the Shannon entropy multiplied by k_B to have the correct units

$$S = -k_B \sum_n p_n(t) \ln\left(p_n(t)\right)$$

$$\dot{S}/k_B = -\sum_n \dot{p}_n \ln\left(p_n(t)\right) - \sum_n p_n \frac{1}{p_n(t)} \dot{p}_n(t) = -\sum_n \dot{p}_n \ln\left(p_n(t)\right) - \sum_n \dot{p}_n(t)$$

Since by normalisation $\sum_n \dot{p}_n = 0$

$$\dot{S}/k_B = -\sum_n \dot{p}_n(t) \ln\left(p_n(t)\right) \tag{1.16}$$

$$\begin{aligned}
\sum_n \dot{p}_n(t) \ln\left(p_n(t)\right) &= -\sum_{n,m,\nu} W^{\nu}_{nm} p_m \ln\left(\frac{p_n}{p_m}\right) \\
&= \sum_{n,m,\nu} \frac{1}{2} \left[-W^{\nu}_{nm} p_m - W^{\nu}_{nm} p_m\right] \ln\left(\frac{p_n}{p_m}\right) \\
&= \sum_{n,m,\nu} \frac{1}{2} \left[W^{\nu}_{nm} p_m + W^{\nu}_{nm} p_m\right] \ln\left(\frac{p_m}{p_n}\right) \\
&= \sum_{n,m,\nu} \frac{1}{2} \left[W^{\nu}_{nm} p_m - W^{\nu}_{mn} p_n\right] \ln\left(\frac{p_m}{p_n}\right) \\
&= \sum_{n,m,\nu} \frac{1}{2} \left[W^{\nu}_{nm} p_m - W^{\nu}_{mn} p_n\right] \ln\left(\frac{W^{\nu}_{nm}}{W^{\nu}_{mn}}\right) \\
&\quad + \sum_{n,m,\nu} \frac{1}{2} \left[W^{\nu}_{nm} p_m - W^{\nu}_{mn} p_n\right] \ln\left(\frac{W^{\nu}_{nm} p_m}{W^{\nu}_{mn} p_n}\right)
\end{aligned}$$

By appling the local detailed balance condition (1.11), the first term can be written as

$$\sum_{n,m,\nu} \frac{1}{2} \left[W^{\nu}_{nm} p_m - W^{\nu}_{mn} p_n \right] \ln \left(\frac{W^{\nu}_{nm}}{W^{\nu}_{mn}} \right) = \sum_{\nu} \frac{\dot{Q}^{\nu}}{T_{\nu}} = \dot{S}_e(t)$$

which is the rate of the entropy exchange between the system and the reservoirs.
The second terms is the entropy production rate $\dot{S}_i$. The quantity $X^{\nu}(t) \equiv \ln \left(\frac{W^{\nu}_{nm} p_m}{W^{\nu}_{mn} p_n} \right)$ is the force pushing the system out of equilibrium and $J^{\nu}(t) \equiv W^{\nu}_{nm} p_m - W^{\nu}_{mn} p_n$ represents the net jumps between the states from the reservoir ν

We hence have

$$\dot{S}(t) = \dot{S}_e(t) + \dot{S}_i(t)$$

where

$$\begin{cases} S_e = \frac{1}{2} \sum_{n,m,\nu} J^{\nu}_{nm}(t) \ln \left(\frac{W^{\nu}_{mn}}{W^{\nu}_{nm}} \right) = \sum_{\nu} \frac{\dot{Q}^{\nu}}{T_{\nu}} \\ S_i = \frac{1}{2} \sum_{n,m,\nu} J^{\nu}_{nm}(t) X^{\nu}_{nm}(t) \end{cases} \tag{1.17}$$

Because $(x - y) \ln \left(\frac{x}{y} \right) \geq 0 \ \forall x, y$, $J^{\nu}_{nm}(t) X^{\nu}_{nm}(t) \geq 0$ and hence, as expected $\dot{S}_i \geq 0$

If the system undergoes a **quasi-static transformation**, the detailed balance condition is satisfied and hence the entropy production rate equals to zero. By expanding around equilibrium we see that differently than $\dot{S}_i$, $\dot{S}_e$ does not go to zero. Hence, as expended, if the system undergoes a quasi static transformation

$$\dot{S}(t) = \dot{S}_e(t) = \sum_{\nu} \frac{\dot{Q}^{\nu}}{T_{\nu}} \tag{1.18}$$

1.2.3 Trajectory-level description

The master equation describes the evolution of a time dependent probability. We are hence considering a time dependent macrostate that at each time corresponds to the mean value of the time dependent miscrostates. The macrostate that at each time is the mean value of the microstates at that time.
The time dependent microstates are hence "trajectories" and their mean value corresponds to the probability in the master equation.
In this section we give the expressions of the thermodynamic quantities on the single trajectories.

We consider doing a transformation on a system in contact with more reservoirs ν with maximum one particle that can fill the system's states. The transformation has a duration of τ
We define by $n(t)$ the occupied state at time t. The energy contribution at time t to the energy trajectory is

$$x_{n(t)} = \epsilon_{n(t)} - \mu(t) \tag{1.19}$$

We decretise the time interval $[0, \tau]$ in M steps, in such a way that

$$t_i = i\frac{\tau}{M}, \quad i = 0, 1, ..., M \tag{1.20}$$

The timesteps $dt = \frac{\tau}{M}$ must be small enough to catch small changes, but can not be too small, $dt \not\to 0$. If dt is too small no changes will we caught in the interval $(t, t + dt)$

With these considerations, we can formulate the two laws of thermodynamics for single trajectories

First Law

$$\Delta E = w_{mec} + w_{chem} + \sum_{\nu} Q^{\nu}$$

The contribution of mechanical work at each instant is the difference in energy of the occupied state. If the state is and remains occupied, an amount of $dw(t_i) = d\epsilon(t_i)$ enters the system if $d\epsilon(t_i) > 0$ [resp. exit the system if $d\epsilon(t_i) < 0$]. If the state is and remains unoccupied, no work is tranfered to [resp. extracted from] the system.

$$w_{mec} = \sum_{i=0}^{N} d\epsilon_{n(t_i)} = \sum_{i=1}^{N} \left[\epsilon_{n(t_i)}(t_i) - \epsilon_{n(t_i)}(t_{i-1})\right] \delta_{n(t_i),n(t_{i-1})} \tag{1.21}$$

Here we supposed that the energy of the level is not changed in $[t_0, t_0 + \delta]$ where $\delta < t_1 - t_0$, hence $d\epsilon_{n(t_0)} = 0$

The heat exchange contribution is nonzero only when jumps between the states occur and takes only the contribution by the reservoir ν from which the corresponding jump occur. Hence the heat contribution is the difference in effective energy between the states involved in the jump due only to the reservoir ν_i (the ν that corresponds to the reservoir from which we have a jump at the time t_i)

$$Q = \sum_{\nu} Q^{\nu} = \sum_{\nu} \sum_{i=1}^{N} \delta(\nu - \nu_i) \left[x^{\nu_i}_{n(t_i)-1}(t_i) - x^{\nu_i}_{n(t_i)}(t_i)\right] \tag{1.22}$$

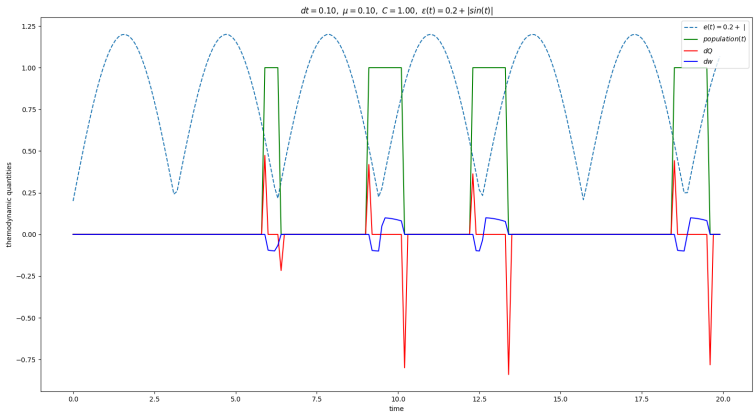

Figure 1.1: Single trajectory thermodynamic quantities of a single level quantum dot in contact with a single reservoir - **dashed line** $\rightarrow$ energy of the dot's level, **blue line** $\rightarrow$ mechanical work contribution, **red line** $\rightarrow$ heat exchange contribution

The chemical work contribution is the net particles exchanged in the process

$$w_{chem} = \sum_{\nu} \sum_{i=1}^{N} \delta(\nu - \nu_i)\mu_\nu \left[N^{\nu_i}_{n(t_i)-1}(t_i) - N^{\nu_i}_{n(t_i)}(t_i) \right] = \sum_{\nu} \sum_{i=1}^{N} \delta(\nu - \nu_i)\mu_\nu(\pm 1) \tag{1.23}$$

From Figure 1.1 we can see the evolution of these thermodynamic quantities contribution in time. The Figure represents the resulting plot from a simulation of a single level quantum dot in contact with one reservoir. We see that when the dot in empty, neither work not heat is exchanged, while when the dot is full, the work contribution (blue line) follow the behaviour of the energy (dashed line) as expected from Eq 1.21. The heat contribution (red line) only takes a nonzero value when a jump occurs: a positive contribution when an electron enters the quantum dot and a negative contribution when the electron exit the dot - as expected from Eq 1.22 and from the positivity of the energy $\epsilon(t)$ at each time (in the case of the transformation corresponding to the figure) .

Second Law

$$\Delta S = \sigma + \Delta S_e$$

where σ is the trajectory entropy production.
Taking $p(n(t) \to m) = W^{\nu}_{n(t),m} p_m = W^{\nu}_{n(t),m} \delta_{n(t),m}$ as the probability to jump from the current sate $n(t)$ to another stets m (we will justify this choice in the next chapter) and applying Eq 1.17, we see that

$$\Delta S = \sum_i^N k_B \ln \frac{p_{n(t_i)}(t_i)}{p_{n(t_i)}(t_i)} = k_B \ln \frac{p_{n(t_0)}(t_0)}{p_{n(t_N)}(t_N)} \tag{1.24}$$

Even if we are looking at trajectories, the entropy is still proportional to a probability.
The entropy production σ that is given by [2]

$$\sigma = k_B \ln \frac{P}{\overline{P}} \tag{1.25}$$

where P is the probability of the full trajectory forward in time and $\overline{P}$ is the probability of the full trajectory backwards in time. It is a measure of dissipation: it describes the difference between the probability if observing the trajectory forward in time and the probability of observing the trajectory backwards in time. If we do a quasi-static transformation $P = \overline{P}$ and hence, the transformation is time reversible.

The entropy exchange between the system and the reservoir is found by applying Eq 1.17 and Eq 1.22

$$\Delta S_e = \sum_i^N \sum_\nu \frac{Q^\nu}{T_\nu} = \sum_i^N \sum_\nu \frac{\delta(\nu - \nu_i)}{T_\nu} \left[x^{\nu_i}_{n(t_i)-1}(t_i) - x^{\nu_i}_{n(t_i)}(t_i) \right] \tag{1.26}$$

1.3 Quantum dots

A Quantum dot is a nanostructure of a semiconductors confined by another semiconductor with an higher energy gap. Because of the dimensions of the dots (from $2nm$ to $10nm$), their physics is governed by quantum mechanics and in particular, their confinement make them behave as a potential well. We are able to control their size and the energy of their levels which makes them great candidates for many applications.
They are commonly known for their propriety of emitting different color light depending on their dimensions rather than on the material they are made of. The dimension of the dot determines the size of the band gap, and hence its emission spectrum.

What is more interesting for a thermodynamic study is their atomic behavior and the possibility to estimate the population of their energy levels.
Putting the a dot in contact with one or more reservoirs, results in pumping or extracting electron by tunneling and hence a change in the dot's level population which then results in an heat flow and on a chemical work done on the system. Externally controlling the energy of its level results on a mechanical work on the system.

We can hence see that these nanostructures can be used to reproduce thermodynamic machines that in principle work far from equilibrium.
Using stochastic thermodynamics we are able to formally describe such processes and hence provide information on nonequilibrium thermodynamics on relatively simplifiable systems.

1.3.1 Assumptions

The following assumptions are made on the quantum dots

- **We consider only one exited level of the dot interacting with the reservoir**
  The reservoir's electronic population is governed by the Fermi-Dirac distribution and we assume all the energy levels higher than the one we consider, to have an energy higher than the Fermi energy => the probability of finding electrons able to enter higher levels of the dot is low.
  We suppose all the levels with lower energy than the one we consider to have an energy far lower, and hence no electron interchange is possible with the considered state.
  This can be achieved by setting the right difference of potential on the dot.

- **Maximum one electron can populate the dot exited level**
  In order to enter a partially filled level, an electron must gain the potential energy of the electron already filling the level => The entering electron

must have an energy higher than the energy of the state and we suppose that energy to be higher than the Fermi energy.

From now on, **every time I will refer to the quantum dots in this document, I will suppose these conditions**.

1.3.2 Evolution of occupation probability

Defining the population of the dot by $n(t)$, we have that the population probability $p(t) = Prob(n(t) = 1)$ of the dot follows the differential equation

$$\dot{p}(t) = -W_- p(t) + W_+(1 - p(t)) \tag{1.27}$$

where

$W_+ dt = p(filled, t + dt|empty, t)$

$W_- dt = p(empty, t + dt|filled, t)$

If we consider the fact that the dot can be in two states: $state\ 2 \rightarrow empty$ and $state\ 1 \rightarrow filled$, with probabilities $p_2(t) = Prob(n(t) = 0)$ and $p_1(t) = Prob(n(t) = 1)$, setting

$p_1(t) = p(t)$

$p_2(t) = 1 - p(t)$

$W_{12} = W_+$

$W_{21} = W_-$

we can see the equivalence of the mater equation that reads

$$\begin{pmatrix} \dot{p}_1(t) \\ \dot{p}_2(t) \end{pmatrix} = \begin{pmatrix} -W_{21} & W_{12} \\ W_{21} & -W_{12} \end{pmatrix} \begin{pmatrix} p_1(t) \\ p_2(t) \end{pmatrix} \tag{1.28}$$

Because of the assumptions on the dot, the electron can enter the level only if the level is empty and can leave the level, only if the level is filled.
From now on in this document, **when referring to the the quantum dots, the indexing of the empty and filled states will be consistence**.

When referring to the quantum dots, the previous assumptions and the existence and the indexing of these two states will be assumed

Chapter 2

Algorithms for Stochastic Simulations

In this chapter I show how I did the simulations the two processes I treat in the next chapter.
I will just go through the dynamics, the thermodynamic quantities are a consequence of the dynamics and are found by applying the equations in section 1.2.3.
As explained in section 1.2.3, the objective is to simulate the stochastic trajectories and to get back the probability described by the master equation meaning over all the trajectories.
We start by considering a slightly more general case and we then restrict ourselves to the quantum dots with the assumptions given in section 1.3.1.

We consider a N state model with maximum one particle exchanged between the states. The particle has a probability $p_k(t)$ to be in the state k.
The evolution on p_k is governed by the master equation

$$\dot{p}_k(t) = \sum_k W^{\nu}_{km} p_m(t)$$

with a probability of transition

$$P(m \xrightarrow{\nu} k) = W^{\nu}_{km} p_m(t) \tag{2.1}$$

where, because we are dealing with trajectories and because only one particle can occupy a state, the probability of being in a state equals the population of that state. Hence, defining $a(t)$ as the occupied state at time t

$$p_m(t) = \delta_{a(t),m} \tag{2.2}$$

the trajectory probability of being in a state is one if we are in that state and zero if we are out of that state.

Therefore, knowing the rates and the population of a state at a certain time, enables us to determine the probabilities of the possible jumps the particle can do from that state in an interval around that time. Using a uniformly distributed variable, we can hence calculate the stochastic population of the states at each time, and hence construct a stochastic trajectory.

We therefore understand that there are two questions to which we have to answer to construct such stochastic trajectories which are

When does a jump occur?

Which jump occurs?

I did the simulation using two different algorithms depending on the situation I was in, that answer this two questions in two different ways.
For a time-dependent rates model, I used a method similar to the Euler algorithm, that I simply call *discrete-time simulation algorithm*, while for a time-independent rates model, I used the *Gillespie algorithm*.

2.1 Discrete-time simulation algorithm

I applied this algorithm to a setup with time-dependent rates.
The method is similar to the Euler algorithm for solving differential equations in the sense that I just iterate trough the times with a constant time step dt and check weather any jump occurs or not.
For each tie step, I do the following

1. check in which state I am by checking the population of the state
2. calculate the transition probabilities of all the possible jumps from the current state $a(t)$
3. sort the possible transition probabilities
4. draw a uniformly distributed number $r \in (0, 1)$
5. look where number r falls between the transition probabilities

For a model of N states k and M reservoirs ν, if at time t the sorted transition probabilities from the state $a(t)$ are described by the ordered array of the form [index $\Rightarrow$ value[index]]

$$trans = [trans_{k_i,\alpha}]_{\substack{i=1,2,..,N-1 \\ \alpha=0,1,....,N}}$$

where

$$trans_{k_i,\alpha} = \Big[\mu_j \Rightarrow P(a(t) \xrightarrow{j} k)\Big]_{\substack{j=1,2,...M \\ \mu_j=\alpha M+j}}$$

So that we have

$$trans = [1 \Rightarrow P(a(t) \xrightarrow{1} k_0), 2 \rightarrow P(a(t) \xrightarrow{2} k_0)..,M \Rightarrow P(a(t) \xrightarrow{M} k_0),$$
$$M+1 \Rightarrow P(a(t) \xrightarrow{1} k_1), M+2 \Rightarrow P(a(t) \xrightarrow{2} k_1), ..., 2M \Rightarrow P(a(t) \xrightarrow{M} k_1),$$
$$....$$
$$N+1 \Rightarrow P(a(t) \xrightarrow{1} k_{N-1}), N+2 \Rightarrow P(a(t) \xrightarrow{2} k_{N-1}), ..., N*M \Rightarrow P(a(t) \xrightarrow{M} k_{N-1})]$$

where μ_i is the index starting from 1 identifying the sorted probability transitions,
for that time t, we perform

---

if $r < P(a(t) \xrightarrow{1} k_0)$ **then**
  $p_{a(t)} = p_{a(t)} - 1$
  $p_{k_0} = p_{k_0} + 1$
  $a(t) = k_0$
....
else if $P(a(t) \xrightarrow{i} k_j) < r < P(a(t) \xrightarrow{i+1} k_j)$ **then**
  $p_{a(t)} = p_{a(t)} - 1$
  $p_{k_j} = p_{k_j} + 1$
  $a(t) = k_j$
....
else if $r < P(a(t) \xrightarrow{M} k_{N-1})$ **then**
  $p_{a(t)} = p_{a(t)} - 1$
  $p_{k_{N-1}} = p_{k_{N-1}} + 1$
  $a(t) = k_{N-1}$
else
  continue

---

where I recall that p_k is the probability=population of the state k.

Because the rates are time-dependent, the order of the array *trans* may be different at each time. However, it is not enough construct a time-dependent array *trans* by sorting the transitions probability at each time and iterate on it. To extract useful information from the dynamics, such as the currents from each reservoir, for each jump occurring, we need to identify between which states and due to which reservoir it happened, we hence have to have a way to get back the non-numerical information about the last occurred jump, which would result in slow computation.
Thanks to simplifications and to physical intuition, we can set up our model in such a way that enables to know the order of the rates at each time, and hence to construct the algorithm ad hoc avoiding time consuming tests.

2.1.1 Quantum dot in contact with one reservoir

We investigate the situation of a quantum dot in contact with a reservoir. The dot is constrained to the assumption given in the section 1.3.1. As said in the section 1.3.2, this description corresponds to a two-level model, where the dot can be in two states: an empty state (state 2) and a filled state (state 1).
Because of the assumptions on the dot, the electron can enter the level only if the level is empty and can leave the level, only if the level is filled.
Defining by $p(t) = p_1(t)$ the population of the dot, we describe the possible jumps between the states by

$$P(2 \to 1) = (1 - p(t))\, W_{1,2} dt = \begin{cases} W_{1,2} dt & \text{if } p(t) = 0 \\ 0 & \text{if } p(t) = 1 \end{cases} \tag{2.3}$$

$$P(1 \to 2) = p(t) W_{2,1} dt = \begin{cases} W_{2,1} dt & \text{if } p(t) = 1 \\ 0 & \text{if } p(t) = 0 \end{cases} \tag{2.4}$$

Therefore, the algorithm for each iteration becomes

---

if $n(t) = 0$ **then**

  if $r < P(2 \to 1)$ **then**

    $p(t) = 1 - p(t)$

    $a(t) = 1$

  else

    continue

else if $n(t) = 1$ **then**

  if $r < P(1 \to 2)$ **then**

    $p(t) = 1 - p(t)$

    $a(t) = 2$

  else

    continue

Which results in the following python code for the dynamics of a trajectory

```
import numpy as np
import random

## Constants describing the reservoir
# and the reservoir-system interactions
# used for the calculations of the rates
global C, mu, T

MAXTIME = 10 # duration of the transformation
STEPS = 1000 # number of steps
dt = float(MAXTIME) / STEPS
time = np.linspace(0, MAXTIME, STEPS)

def trajectory(MAXTIME, time, dt):
    x = [0]
    t = 0
    n = 0
    for i in range(len(time)):
        t = time[i]
        e = energy(t)
        W_plus = rate(e,t,'up')
        W_down = rate(e,t,'down')
        r = random.random()
        if n == 0:
            if r < W_plus * dt:
                n = 1
            else:
                continue
        else:
            if r < W_down * dt:
                n = 0
            else:
                continue
        x.append(n)
    return x

def energy(t):
    '''Function calculating the energy of the dots level

    input: time : type: float
    return: energy :  type: float
    '''
    pass

def rate(e,t,type):
    '''Function calculating the rates of the transitions

    input: energy : type float , description: energy of the dot
    input: t : type float ,description: time of the transition
    input: type : type str , description: type of transition (up of
     down)
    return: rate : type float , description: rate of the transition
    '''
    pass
```

To describe the occupation probability of the macrostate, we mean over all the trajectories.

```
N = 10000 # Number of iterations
p = [[] for _ in range(N)]
avg_p = [0 for _ in range(MAXTIME)]
for n in range(N):
    p[n] = trajectory(MAXTIME, time, dt)
for i in range(MAXTIME):
    avg_p[i] = np.sum(p[:,i])/float(N)
```

For compactness reasons I chose the python language to illustrate the algorithm, the actual code of this simulation was written in C for performance reasons and due to personal preferences.
Note moreover that this is not the full code, the thermodynamic quantities are missing. The full code is found in the **Driven˙regime** directory.

Down-sides of the algorithm

The trajectory constructed using this algorithm is strongly dependent on how we descretise the time lane.
If we choose a time step dt too small, we wont find any transitions in the interval $(t, t+dt)$, and if we choose a dt to big the transition probability is not anymore the one we want to use to describe a process governed by the master equation because multiple jumps may occur in the interval $(t, t+dt)$.

2.2 Gillespie algorithm

I applied the Gillespie algorithm to a setup with time-independent rates.
Differently than the discrete-time method, here we do not descritise the time but, by sampling from the waiting times probability distribution, we calculate the instants at which the jumps occurs.
The jump times calculated with this method are the ones that we would find by solving the master equation. We hence don't do the approximations of the previous method and do not encounter its problems.

2.2.1 Sampling from a known distribution

To avoid going through it in the middle of the explanation of the Gillespie algorithm, in this section I show why we are allowed to sample from a probability distribution and how it is done.

Let X be a random variable governed by the probability distribution $p(x)$. We define by $F(x) \equiv P(X \leq x) \in [0,1]$ it's cumulative distribution function

$$F(x) = \int_0^x p\left(x'\right) dx'$$

A *sample* of the random variable X is the value of the cumulative distribution of X evaluated at a random value x assumed by X.
Hence, by defining the random variable $Y \equiv F(X)$, we see that Y is a sample of X.

Let q be the distribution of Y, we have that

$$\int_0^y q\left(y'\right) dy' = \int_0^x p\left(x'\right) dx'$$

hence

$$1 = \int q(y)dy = \int p(x)dx = \int p(x(y)) \mid \frac{dx}{dy} \mid dy$$

from which

$$q(y) = p(x(y)) \mid \frac{dx}{dy} \mid = p(x(y)) \left(\frac{dF(x)}{dx}\right)^{-1}_{x=x(y)} = \frac{p(x(y))}{p(x(y))} = 1 \tag{2.5}$$

We hence found that Y is a uniformly distributed.

Exponential distribution

We will see that the waiting times from the Gillespie algorithm are exponentially distributed. Therefore, here we sample from the an exponentially distributed

random variable X.

Let $p(x) = \lambda e^{-\lambda x}, x \in [0, \infty)$ be the density of X

$$F(x) = \int_0^x p\,(x')\,dx' = 1 - e^{-\lambda x}$$

We define $U \in (0,1),\ U \equiv 1 - F(x)$. With r being its density, we see that

$$r(u) = \mid \frac{du}{dx} \mid_{x=x(u)} p(x(u)) = \frac{p(x(u))}{p(x(u))} = 1$$

Hence, U is also uniformly distributed, and hence we can set

$$U = 1 - F(X) = e^{-\lambda x} \iff X(U) = -\frac{1}{\lambda} \ln U \tag{2.6}$$

and hence

$$x(u) = -\frac{1}{\lambda} \ln u$$

2.2.2 Algorithm

As stated previously the objective is to find the next time at which a jump occurs. Once we have this knowledge we will be able to determine which of the possible jump occurs.

Jumping times

We define as $P(t + dt) = P(a(t + dt) = k \mid a(t) = k)$ the probability that no jumps occurs in the interval $(t, t+dt)$. Taking $a(t)$ as the current occupied state and k a general state

$$P(t + dt) = P(a(t + dt) = k \mid a(t) = k) \tag{2.7}$$

which from the master equation

$$P(a(t + dt) = k \mid a(t) = k) = 1 - \sum_{m,\nu} W^{\nu}_{mk} dt \tag{2.8}$$

by noticing that $1 = P(a(t) = k \mid a(t) = k) = P(t)$

$$\frac{P(t + dt) - P(t)}{dt} = \frac{dP(t)}{dt} = -\sum_{m,\nu} W^{\nu}_{mk} = -P(t) \sum_{m,\nu} W^{\nu}_{mk} dt \tag{2.9}$$

From which we can find the probability that no jump occur before a time τ by solving (2.9)

$$P(da(t) = 0 \mid t \leq \tau) = e^{-h\tau}, \quad h = \sum_{m,\nu} W^{\nu}_{m,k} \tag{2.10}$$

and we can find the probability of having a jump in τ

$$P(da(t) \neq 0) = 1 - e^{-h\tau} \tag{2.11}$$

From Eq (2.10), we see that the waiting times are exponentially distributed, we can hence sample trough them following Eq (2.6) found in the previous section, and hence setting a uniformly distributed variable $r_1 \in (0, 1)$ find the time of the next occurring jump by

$$t = \frac{1}{h} \ln\left(\frac{1}{r_1}\right) \tag{2.12}$$

Identify the type of jump

Once we have the knowledge of when a transition occurs, we can identify the type of jump by setting a uniformly distributed variable $r_2 \in (0, 1)$ and solving for the integer γ such that [4]

$$\frac{1}{h} \sum_{\rho=0}^{\gamma} W_\rho = r_2 = \frac{1}{h} \sum_{\rho=0}^{\gamma+1} W_\rho \tag{2.13}$$

where ρ runs over all the combinations of (k, ν) from the current state $a(t)$.

2.2.3 Stationary single-level quantum dot in contact with two reservoirs

We look how the Gillespie algorithm looks like in a model of a stationary state quantum dot in contact with two reservoirs.
We choose the again the empty state as the state 2 and the filled state as the state state 1 and we define by $p(t)$ population of the dot. The dots-reservoirs interactions are characterised by the rates

$$W_{1,2} = \sum_{\nu=1,2} W_{1,2}^{\nu} = W_{1,2}^{(1)} + W_{1,2}^{(2)}$$
$$W_{2,1} = \sum_{\nu=1,2} W_{2,1}^{\nu} = W_{2,1}^{(1)} + W_{2,1}^{(2)}$$

Where we suppose $W_{1,2}^{(1)} < W_{1,2}^{(2)}$ and that $W_{2,1}^{(2)} < W_{2,1}^{(1)}$

We have that

$$W_\rho = \begin{cases} (1 - p(t))\, W_{1,2} = (1 - p(t)) \left[W_{1,2}^{(1)} + W_{1,2}^{(2)}\right] \\ p(t) W_{2,1} = p(t) \left[W_{2,1}^{(1)} + W_{2,1}^{(2)}\right] \end{cases} \tag{2.14}$$

and

$$h = \sum_{\rho} W_\rho = (1 - p(t)) \left[W_{1,2}^{(1)} + W_{1,2}^{(2)}\right] + p(t) \left[W_{2,1}^{(1)} + W_{2,1}^{(2)}\right] \tag{2.15}$$

Hence

$$h = \begin{cases} W_{1,2}^{(1)} + W_{1,2}^{(2)} & \text{if } p = 0 \\ W_{2,1}^{(1)} + W_{2,1}^{(2)} & \text{if } p = 1 \end{cases} \tag{2.16}$$

We know how to calculate the times of the jumps by Eq. (2.12). Suppose that a jump occurs at the time τ, by knowing the rates we are able to identify which jump occurred and extract information on the currents I_1 and I_2 from the reservoirs 1 and 2 due to the transition using the following algorithm

---

$r_2 = random.uniform(0, 1)$
if $p(\tau) = 0$ **then**

if $r_2 < W_{1,2}^{(1)}/h$ **then**

$I_1(\tau) = 1$

$I_2(\tau) = 0$

else

$I_1(\tau) = 0$

$I_2(\tau) = 1$

else if $p(\tau) = 1$ **then**

if $r_2 < W_{2,1}^{(1)}/h$ **then**

$I_1(\tau) = -1$

$I_2(\tau) = 0$

else

$I_1(\tau) = 0$

$I_2(\tau) = -1$

$p(\tau) = 1 - p(\tau)$

The corresponding code for a full trajectory is

```
import numpy as np
import random

```

```
def trajectory(MAXTIME, rates):
    t = 0
    p = 0  # occupation: initially void quantum dot
    jumps_t = [0] # jump times list
    p_t = [p] # trajectory occupation list
    ## Rates calculated outside the function
    rate_u1 = rates['rate_up1']
    rate_u2 = rates['rate_up2']
    rate_d1 = rates['rate_down1']
    rate_d2 = rates['rate_down2']
    ## Currents
    I_1 = [0]
    I_2 = [0]

    while t < MAXTIME:
        rate_u = (1 - p) * (rate_u1 + rate_u2)
        rate_d = p * (rate_d1 + rate_d2)
        rate = rate_u + rate_d
        tau = - (1.0 / rate) * np.log(random.random())
        t += tau
        jumps_t.append(t)
        r = random.random()
        if p == 0:
            if r < rate_u1/rate_u:
                I_1.append(1)
                I_2.append(0)
            else:
                I_1.append(0)
                I_2.append(1)
        elif p == 1:
            if r < rate_d2/rate_d:
                I_1.append(0)
                I_2.append(-1)
            else:
                I_1.append(-1)
                I_2.append(0)
        p = 1-p
        p_t.append(p)

    return p_t, jumps_t, I_1, I_2
```

I stress on the fact that differently than the previous algorithm, we do not iterate through all the times and check if we have a jump or not, but we calculate the jumps occurred times and we update the dynamic quantities at that times.

Because the process is stochastic, the transition times are not equal for all trajectories and most of all they are not equally spaced . We hence understand that this code is not useful if we want to calculate an ensemble average - which is the objective.
To solve this problem, we define a "trajectory-general" time-vector and we build the trajectory population vectors by giving it the population values between the jumps.
The code then becomes

```
import numpy as np
```

```
import random

times = np.linspace(0, MAXTIME, resolution)

def trajectory(times, rates):
    """
    Stochastic simulation algorithm with fixed time resolution
    """
    t = 0
    ## Rates calculated outside the function
    rate_u1 = rates['rate_up1']
    rate_u2 = rates['rate_up2']
    rate_d1 = rates['rate_down1']
    rate_d2 = rates['rate_down2']

    p = 0 # initial population: initially void quantum dot
    resolution = len(times)
    p_t = [0 for i in range(resolution)]

    # Single trajectory quantities
    I_1 = 0
    I_2 = 0
    I_1_traj = [0]
    I_2_traj = [0]
    p_traj = []
    t_traj = [0]

    i = 0
    while i < resolution:
        p_old = p # Update each jump
        while t <= times[i]:
            rate_u = (1 - p) * (rate_u1 + rate_u2)
            rate_d = p * (rate_d1 + rate_d2)
            rate = rate_u + rate_d
            tau = - (1.0 / rate) * np.log(random.random())
            t += tau
            t_traj.append(t)
            r = random.random()
            if p == 0:
                if r < rate_u1/rate_u:
                    I_1 = 1
                    I_2 = 0
                else:
                    I_1 = 0
                    I_2 = 1

            elif p == 1:
                if r < rate_d2/rate_d:
                    I_1 = 0
                    I_2 = -1
                else:
                    I_1 = -1
                    I_2 = 0
            p = 1 - p
            I_1_traj.append(I_1)
            I_2_traj.append(I_2)
            p_traj.append(p)
```

```
        ## Find the index of the times list corresponding to
        # the nearest bigger value of the lst jump time
        # we are looking for the index k such that times[k] is
        # the first value such that t < times[k]
        time_idx = np.searchsorted(times > t, True)
        corr_idx = min(resolution - 1, time_idx)
        for j in range(i, corr_idx - 1):
            ## Fill the elements corresponding to the indexes
            # between the jumps with the old value
            p_t[j] = p_old
        # update previous jump index
        i = time_idx

    return p_t, I_1_traj, I_2_traj, t_traj, p_traj
```

In this way we are able to do a classic ensemble mean for the dot population. However this technique do not work for the quantities that are non-zero only at the times of the jumps such as the currents (we encounter the same issue with the heat exchange rate: in code in the directory **Stationary˙State**, it's seen that they are treated in the same way). Because the jumps are not equally spaced, the probability of finding more than one jump occurring in a small windows of time is extremely low, hence treating these quantities in the same way as the dot's population, drops them to zero. To have an idea of the behaviour of these quantities, we use the empirical cumulative function by integrating them. Because the process is markovian, and hence depends only on the just previous time, the empirical cumulative function should approach the expected values for a big number steps.

The full code of the dynamics is the following

```
import numpy as np
import random

N = 500
MAXTIME = 3000
resolution = 120000
times = np.linspace(0, MAXTIME, resolution)

######
# Here, calculation of the rates
# rate_u1, rate_u2, rate_d1, rate_d2
#####

rates = {'rate_up1' : rate_u1, 'rate_down1' : rate_d1 , 'rate_up2'
    : rate_u2, 'rate_down2': rate_d2}

## Initialising ensemble quantities
avg_p = [0 for i in range(res)]             # occupation
    distribution
second_moment = [0 for i in range(res)]     # second momentum of
    the occupation distribution

# Initialising single trajectory quantities
I1 = [0]
I2 = [0]
```

```
23 jumped_t = [0]
24
25 trajectory_flag = False
26 for j in range(N):
27     p_t, I1_traj, I2_traj, t_traj, p_traj = trajectory(times, rates
       )
28
29     if trajectory_flag == False:    # Pass here only once
30
31         # Calculation for the trajectory quantities
32         for i in range(len(t_traj)):
33             jumped_t.append(t_traj[i])
34         for i in range(len(t_traj)):
35             if times[i] == 0.0:
36                 dI1[i] = I1_traj[i]
37                 dI2[i] = I2_traj[i]
38             else:
39                 dI1[i] = sum(n_1_t[:i]) / t_traj[i]
40                 dI2[i] = sum(n_2_t[:i]) / t_traj[i]
41             I1.append(dI1[i])
42             I2.append(dI2[i])
43         trajectory_flag = True
44
45     ## Mean on N trajectoreis
46     for t in range(res):
47         avg_p[t] += p_t[t] / N
48         second_moment[t] += p_t[t]**2 / N
```

Chapter 3

Thermodynamics of Quantum Dots

In this chapter the actual project is introduced. The projects analyzes three cases faced in the three articles [3], [1] and [5].

We deal with three thermodynamic machines that are both in an out of equilibrium regime, but the constrains that keep them out of equilibrium are of two different nature - in both cases the cause are the rates, but for different reasons.
In the first case we are in a driven regime, hence the out of equilibrium status is caused by an external force acting on the system; in the second and third cases we are in a non equilibrium stationary regime that is due to the difference in the temperatures and chemical potential of the reservoirs, hence the cause is a chemical work.

We want to show that operating with a quantum dots in these regimes, we can extract useful information on how to make the machine more efficient, and by simulating them we can verify the validity of the theoretical results.
For the single quantum dots cases, an overview on what was done in the two articles [3] and [1] is given, while for the case of two couple quantum dots only the one setup described the article [5] is used. In the three setups, we show how the thermodynamic quantities are simulated at a trajectory level and analyse the results of the simulations.

References

[1] Massimiliano Esposito, Katja Lindenberg, and Christian Van den Broeck. "Thermoelectric efficiency at maximum power in a quantum dot". In: *EPL (Europhysics Letters)* 85.6 (2009), p. 60010.

[2] Massimiliano Esposito and Christian Van den Broeck. "Three faces of the second law. I. Master equation formulation". In: *Physical Review E* 82.1 (2010), p. 011143.

[3] Massimiliano Esposito et al. "Finite-time thermodynamics for a single-level quantum dot". In: *EPL (Europhysics Letters)* 89.2 (2010), p. 20003.

[4] Daniel T Gillespie. "Exact stochastic simulation of coupled chemical reactions". In: *The journal of physical chemistry* 81.25 (1977), pp. 2340–2361.

[5] Riccardo Rao and Massimiliano Esposito. "Conservation laws shape dissipation". In: *New Journal of Physics* 20.2 (2018), p. 023007.