Statistical Methods and Thermodynamics

Chem 472: Lecture Notes

Prof. Victor S. Batista

Zoom: Meeting ID: 943 8610 8716, Passcode: victor

SCL 21 - Tuesday and Thursday 2:30 – 3:45 pm

Yale University - Department of Chemistry

Contents

CHEM 472/572 — Statistical Mechanics for Molecules and Materials

Statistical mechanics is the language that connects microscopic laws to macroscopic behavior. It underpins modern research in chemistry, materials science, physics, data science, and chemical engineering. This course introduces the core ideas of *quantum statistical mechanics* and shows how they are used to understand and predict the behavior of molecules and materials in gases, liquids, and solids.
The course emphasizes both physical insight and practical tools. You will learn how classical thermodynamics emerges from quantum mechanics, and how theoretical ideas translate into computational methods widely used in research and industry. By the end of the course, you will be able to analyze complex molecular systems using the same concepts and techniques found in current scientific literature.
Topics include statistical ensembles; Boltzmann, Fermi, and Bose statistics; density matrices; mean-field theories; phase transitions; chemical reaction dynamics; time-correlation functions; Monte Carlo methods; and molecular dynamics simulations.

Course Materials

Required Text: David Chandler, *Introduction to Modern Statistical Mechanics*, Oxford University Press (Chs. 3–8).
Additional References: Hill, McQuarrie, Reif, Kubo, Friedman, and Tuckerman (details on course website).
Lecture notes, assignments, and computational examples are available at: `https://batistalab.com/teaching/chem-472-572`

Who Should Take This Course

Designed for senior undergraduates and first-year graduate students with interests in molecular science, materials, theory, or computation. No prior experience with simulations is required—methods are introduced step by step.

Grading

Homework and assignments: 30% Midterm exam: 50% Final computational project: 20%
The final project replaces a traditional final exam and allows you to explore a topic of interest using computational tools introduced in class.

Tentative Schedule

- Jan. 12–21: Statistical Methods and Ensembles
- Jan. 26–Feb. 4: Ideal Systems
- Feb. 9–25: Phase Transitions
- Mar. 2–6: Monte Carlo Methods

Contact

Office hours by Zoom or appointment. Email: `victor.batista@yale.edu`

1 Introduction

Statistical mechanics builds a bridge between two descriptions of nature: macroscopic observables that we measure in the lab (temperature, pressure, energy, heat capacities, phase behavior), and the microscopic laws that govern electrons, atoms, and molecules. In this sense, it provides the molecular foundation of thermodynamics by explaining how macroscopic regularities emerge from microscopic physics.
A helpful analogy is a crowded city: we cannot track every person's motion, yet we can predict average traffic flow, density, and rush-hour patterns. Likewise, instead of following the exact trajectory of every particle in a macroscopic system, statistical mechanics focuses on the *statistics of accessible microscopic states*.

1.1 Classical microscopic states, ensembles, and time averages

Microscopic state (classical). For a classical system with N particles, a microscopic state is specified by a single point in *phase space*,

$$\Gamma \equiv (\mathbf{q}, \mathbf{p}) = (q_1, \dots, q_{3N}, p_1, \dots, p_{3N}),$$

i.e., the complete set of coordinates and conjugate momenta at an instant of time. You can think of Γ as the "full snapshot" of the system at that instant.

Ensemble (classical). An *ensemble* is a large collection of replicas of the system, each prepared under the same macroscopic conditions, but possibly occupying different microscopic states. In classical statistical mechanics, an ensemble is described by a probability density $\rho(\Gamma)$ over phase space (or by probabilities P_j in a discrete description). The key idea is that $\rho(\Gamma)$ quantifies our uncertainty about which microscopic state the system occupies.

Observables as time averages. An observable is a function of the microscopic state, $A(\Gamma)$ (for example, energy, pressure, or potential energy). A measurement performed over a time interval $[0, t_f]$ can be modeled as a time average along the trajectory $\Gamma(t)$:

$$\langle A \rangle_{\text{time}} = \frac{1}{t_f} \int_0^{t_f} A(\Gamma(t))\, dt = \frac{1}{t_f} \int_0^{t_f} A(t)\, dt.$$

Operationally: follow the system as it moves through phase space, evaluate A at each visited microscopic state, and average over the measurement time.

1.2 Numerical trajectories in phase space: Velocity Verlet

To compute $\Gamma(t)$ numerically, we integrate Newton's equations. For a one-dimensional coordinate q with momentum p and potential energy $V(q)$,

$$\dot{q} = \frac{p}{m}, \qquad \dot{p} = F(q) = -\frac{dV}{dq}.$$

Thus, once $V(q)$ is specified, the force $F(q)$ is determined, and the microscopic state $(q(t), p(t))$ can be propagated forward in time.
A widely used, stable, and time-reversible integrator is the **velocity Verlet** algorithm:

$$\begin{aligned}
q_{n+1} &= q_n + v_n \Delta t + \tfrac{1}{2} a_n \Delta t^2, \\
a_{n+1} &= \frac{F(q_{n+1})}{m}, \\
v_{n+1} &= v_n + \tfrac{1}{2}(a_n + a_{n+1}) \Delta t,
\end{aligned}$$

with $v = p/m$ and $a = F/m$. The algorithm is popular because it is simple, time-reversible, and typically conserves energy well over long times for conservative forces (the energy usually oscillates slightly around the correct value, rather than drifting steadily, when Δt is reasonable). Appendix B describes a Python implementation of the **velocity Verlet** algorithm as applied to the Harmonic oscillator example.

Why time-averaging becomes impossible for macroscopic systems. To compute $\langle A \rangle_{\text{time}}$ by explicit dynamics, we must know (i) the initial conditions $(\mathbf{q}(0), \mathbf{p}(0))$ and (ii) *all interactions*, encoded by the potential energy function $V(\mathbf{q})$. For a system with roughly one mole of particles ($\sim 10^{23}$), this becomes hopeless in practice: the state space is enormous and the exact interactions are far too complex to track particle-by-particle over relevant time scales.

Ensemble averages and the Boltzmann distribution. Statistical mechanics replaces time averages with *ensemble averages*. In the canonical ensemble (fixed N, V, T), the probability of occupying a microscopic state j with energy E_j is assumed to be

$$P_j \propto e^{-\beta E_j}, \qquad \beta = \frac{1}{k_B T}.$$

Normalization gives

$$P_j = \frac{e^{-\beta E_j}}{Z_\beta}, \qquad Z_\beta = \sum_j e^{-\beta E_j}, \qquad \sum_j P_j = 1.$$

Then the ensemble average of an observable is

$$\langle A \rangle_{\text{ens}} = \sum_j P_j A_j.$$

The message is: instead of following one system for a long time, we average over the distribution of states that are thermally accessible.

Ergodic hypothesis. The key assumption connecting these two ways of averaging is that, for an equilibrium system observed long enough,

$$\langle A \rangle_{\text{time}} = \frac{1}{t_f} \int_0^{t_f} A(t)\, dt \approx \langle A \rangle_{\text{ens}} = \sum_j P_j A_j.$$

This is the *Ergodic Hypothesis*: time averages along a single long trajectory can be estimated by ensemble averages over thermally populated microscopic states.

Exercise (for the motivated student). Find an example where the time average does *not* match the canonical ensemble average on accessible time scales (e.g., metastable systems, disconnected regions of phase space, glassy dynamics, or very high barriers).

The canonical partition function. Z_β is called the *canonical partition function*. Conceptually, Z_β is the normalization constant that makes probabilities add to one; physically, it is the central quantity from which equilibrium thermodynamic properties can be derived.

1.3 Example: Classical partition function of the harmonic oscillator

The harmonic oscillator plays a central role in molecular physics and chemistry. Near equilibrium, the potential energy surface of a molecule can be expanded about a stable configuration, and to leading order this expansion is quadratic in the nuclear displacements. As a result, molecular vibrations, lattice phonons, and small-amplitude fluctuations around equilibrium structures are all well described by collections of (independent or weakly coupled) harmonic oscillators. For this reason, the harmonic oscillator provides the simplest nontrivial setting in which to connect microscopic dynamics, statistical mechanics, and thermodynamic averages relevant to molecular systems.

For a *classical* one-dimensional harmonic oscillator with Hamiltonian

$$H(q,p) = \frac{p^2}{2m} + \frac{1}{2}kq^2,$$

the canonical partition function is obtained by integrating the Boltzmann factor over phase space:

$$Z_\beta = \int_{-\infty}^{\infty} dq \int_{-\infty}^{\infty} dp\; e^{-\beta H(q,p)}.$$

Because the Hamiltonian is quadratic in both q and p, the integrals factorize, yielding

$$Z_\beta = \left(\int_{-\infty}^{\infty} e^{-\beta\frac{p^2}{2m}}\,dp\right)\left(\int_{-\infty}^{\infty} e^{-\beta\frac{kq^2}{2}}\,dq\right) = \sqrt{\frac{2\pi m}{\beta}}\sqrt{\frac{2\pi}{\beta k}} = \frac{2\pi}{\beta\omega},$$

where $\omega = \sqrt{k/m}$. (In many treatments an additional factor such as $1/h$ is included to render Z_β dimensionless; the essential β-dependence, which controls the thermodynamics, is unchanged.)

Ensemble-average energy from the partition function. In molecular statistical mechanics, thermodynamic observables are obtained as ensemble averages. A key identity relates the mean energy to the partition function,

$$\langle E\rangle = -\frac{\partial}{\partial\beta}\ln Z_\beta.$$

For the classical harmonic oscillator, $\ln Z_\beta = \ln(2\pi/\omega) - \ln\beta$, and therefore

$$\langle E\rangle = \frac{1}{\beta} = k_BT.$$

This result reflects the equipartition theorem: each quadratic degree of freedom contributes $\frac{1}{2}k_BT$, giving $\langle K\rangle = \frac{1}{2}k_BT$ and $\langle V\rangle = \frac{1}{2}k_BT$.

Time averages versus ensemble averages. In molecular simulations, thermodynamic quantities are typically estimated from time averages along a single trajectory rather than explicit ensemble sampling. For the harmonic oscillator, the canonical ensemble predicts

$$\langle V\rangle_{\text{ens}} = \tfrac{1}{2}k_BT = \frac{\langle E\rangle}{2}.$$

In the numerical simulation discussed above, the total energy is fixed at $E = 0.5$, and the long-time average yields $\langle V\rangle_{\text{time}} \approx 0.25 = E/2$. This agreement illustrates, in the simplest molecular setting, how time averages generated by Hamiltonian dynamics reproduce ensemble predictions, providing a concrete example of the ergodic hypothesis that underlies molecular dynamics simulations.

Link to Dr. Uriel Morzan's Introduction to Statistical Mechanics

2 Postulates of Statistical Mechanics

In the previous section, we illustrated how to computed observables in terms of the probabilities $P_j = Z^{-1}e^{-E_j}$ of observing the system in the various possible microscopic states j of energy E_j, even when we might lack detailed knowledge of the microscopic time-evolution of the system during the time of the measurement.

This approach rests on just two postulates, known as the fundamental postulates of Statistical Mechanics.

$\boxed{\textbf{First Postulate :}}$ *The experimentally measured value of an observable in a macroscopic system is equal to its ensemble average.*

$\boxed{\textbf{Second Postulate :}}$ *A macroscopic system at equilibrium is described by the ensemble that maximizes the entropy, subject to the macroscopic constraints defining the system.*

In the next section, we introduce Gibbs' definition of entropy and we show that when the ensemble maximizes its entropy subject to the constraint of energy E, then $P_j = Z^{-1}e^{-E_j}$ and states with the same energy have the same probability. So, observables can be computed as averages over microscopic states with probabilities $P_j = Z^{-1}e^{-E_j}$, as illustrated in the previous section for the harmonic oscillator.

2.1 Ergodicity

An alternative approach is to compute observables as time averages:

$$\overline{O} = \lim_{\tau\to\infty} \frac{1}{\tau} \int_0^\tau O(t)\, dt, \tag{1}$$

after computing the trajectory of the system in phase-space, as done in molecular dynamics simulations, and obtaining $O(t)$ for each instantaneous configuration $x(t), p(t)$. However, for macroscopic systems, this is impossible since the number of degrees of freedom is enormous, and neither the initial conditions nor the detailed interactions with the environment are known.

The first postulate thus provides a powerful alternative. Instead of following one system in time, we imagine a large collection of identical, independently prepared copies (or *replicas*) of the system—an ensemble. We then compute the observable for each microscopic state and average over the ensemble,

$$\overline{O} = \sum_j P_j\, O(j), \tag{2}$$

where P_j is the probability that a replica of the system is in microscopic state j. This approach bypasses the need to track microscopic dynamics, provided we can determine how many replicas occupy each state.

The second postulate supplies exactly this missing information. By maximizing the entropy subject to macroscopic constraints (such as fixed energy or particle number), we find that $P_j = Z^{-1}e^{-E_j}$. This maximum-entropy principle implies that *all microscopic states with the same energy and particle number are equally probable*.

Returning to the dynamical picture, the second postulate implies that during the measurement, the system must spend equal fractions of time in all microscopic states with the same energy and when the ensemble average coincides with the time average the system is called *ergodic*. When expressed in this way, the second postulate is known as the *ergodic hypothesis* of statistical mechanics.

3 Examples of Ensemble Averages

3.1 Example 1: Spins Coupled to an External Field

We consider a system of two noninteracting spins with magnetic moments μ_1 and μ_2 in thermal equilibrium with a heat bath at temperature T. An external magnetic field B couples to each spin. Each spin can point either *up* $(+)$ or *down* $(-)$ relative to the field, with energies

$$E(i, \pm) = \pm\mu_i B, \qquad i = 1, 2. \tag{3}$$

Because the spins do not interact, the total energy is simply the sum of the individual spin energies.

Our goal is to compute, using the canonical ensemble:

(A) the average internal energy,

(B) the mean-square energy fluctuation,

(C) the entropy,

(D) the Helmholtz free energy,

(E) the average magnetization.

Solution

(A) Average internal energy

Each spin has two possible orientations, so the system has four microstates in total. The corresponding energies are

$$(-\mu_1-\mu_2)B, \quad (-\mu_1+\mu_2)B, \quad (\mu_1-\mu_2)B, \quad (\mu_1+\mu_2)B.$$

The partition function is therefore

$$Z = e^{-\beta B(-\mu_1-\mu_2)} + e^{-\beta B(-\mu_1+\mu_2)} + e^{-\beta B(\mu_1-\mu_2)} + e^{-\beta B(\mu_1+\mu_2)}. \tag{4}$$

Factoring this expression yields

$$Z = 4\cosh(\beta B\mu_1)\cosh(\beta B\mu_2). \tag{5}$$

The average internal energy follows from

$$\bar{E} = -\frac{\partial \ln Z}{\partial \beta},$$

which gives

$$\bar{E} = -B\mu_1\tanh(\beta B\mu_1) - B\mu_2\tanh(\beta B\mu_2). \tag{6}$$

Physical interpretation. Each spin contributes independently to the total energy. At high temperatures ($\beta B\mu_i \ll 1$), thermal fluctuations randomize the spins and $\bar{E} \to 0$. At low temperatures ($\beta B\mu_i \gg 1$), both spins align with the field and the energy approaches its minimum value $-B(\mu_1+\mu_2)$.

(B) Energy fluctuations

The mean-square fluctuation of the energy is

$$\overline{(\delta E)^2} = -\frac{\partial \bar{E}}{\partial \beta}.$$

Using the result above, we obtain

$$\overline{(\delta E)^2} = \frac{(B\mu_1)^2}{\cosh^2(\beta B\mu_1)} + \frac{(B\mu_2)^2}{\cosh^2(\beta B\mu_2)}. \tag{7}$$

Physical interpretation. Energy fluctuations are largest when thermal energy and magnetic alignment compete most strongly. They vanish at low temperatures, where the system is frozen into its ground state, and also at very high temperatures, where all microstates are nearly equally populated.

(C) Entropy

The entropy in the canonical ensemble is given by

$$S = k\beta\bar{E} + k\ln Z. \tag{8}$$

Substituting the expressions for $\bar{E}$ and Z yields

$$\begin{aligned} S = & -k\beta B\mu_1 \tanh(\beta B\mu_1) - k\beta B\mu_2 \tanh(\beta B\mu_2) \\ & + k\ln[4\cosh(\beta B\mu_1)\cosh(\beta B\mu_2)]\,. \end{aligned} \tag{9}$$

Physical interpretation. At high temperatures the four spin configurations are nearly equally likely, and the entropy approaches $k\ln 4$. As the temperature decreases, the magnetic field orders the spins, reducing the number of accessible microstates and hence the entropy.

(D) Helmholtz free energy
The Helmholtz free energy is

$$A = -kT\ln Z = -kT\ln[4\cosh(\beta B\mu_1)\cosh(\beta B\mu_2)]\,. \tag{10}$$

Physical interpretation. The free energy encodes the competition between minimizing energy and maximizing entropy. At low temperatures it is dominated by the internal energy of aligned spins, while at high temperatures the entropic contribution becomes increasingly important.

(E) Average magnetization
Each spin contributes $m_i(\pm) = \pm\mu_i$ to the total magnetization. The ensemble average magnetization is

$$\overline{M} = \sum_j p_j m_j, \qquad p_j = Z^{-1}e^{-\beta E_j}. \tag{11}$$

Evaluating the sum over the four microstates gives

$$\overline{M} = \mu_1 \tanh(\beta B\mu_1) + \mu_2 \tanh(\beta B\mu_2). \tag{12}$$

Physical interpretation. The magnetization measures the degree of spin alignment with the magnetic field. At high temperatures thermal agitation destroys alignment and $\overline{M} \to 0$. At low temperatures both spins align with the field, and the magnetization approaches its maximum value $\mu_1 + \mu_2$.

High-temperature limit and Curie's law
In the high-temperature (or weak-field) limit, $\beta B\mu_i \ll 1$, the hyperbolic tangent may be expanded to leading order:

$$\tanh(\beta B\mu_i) \simeq \beta B\mu_i. \tag{13}$$

Using this approximation, the average magnetization becomes

$$\overline{M} \simeq \beta B(\mu_1^2 + \mu_2^2) = \frac{\mu_1^2 + \mu_2^2}{kT}B. \tag{14}$$

Defining the magnetic susceptibility χ through $\overline{M} = \chi B$, we obtain

$$\chi = \frac{\mu_1^2 + \mu_2^2}{kT}. \tag{15}$$

Thus the susceptibility is inversely proportional to the temperature,

$$\chi \propto \frac{1}{T}, \tag{16}$$

which is Curie's law for a system of noninteracting magnetic moments.

Physical interpretation. At high temperatures thermal fluctuations dominate over magnetic alignment. The external field produces only a small imbalance between spin-up and spin-down states, leading to a magnetization that is linear in the field and decreases as $1/T$, characteristic of paramagnetic behavior.

3.2 Example 2: Helix–Coil Transition in Polypeptides

Small synthetic polypeptides composed of identical amino acids can undergo a structural transition between two distinct states:

- an α-helical state with energy $E_\alpha = -\Delta$,
- a coiled state with energy $E_c = +\Delta$.

A schematic illustration of these configurations is shown in Fig. 1. For long polypeptides, many helical and coiled segments may coexist along the chain.

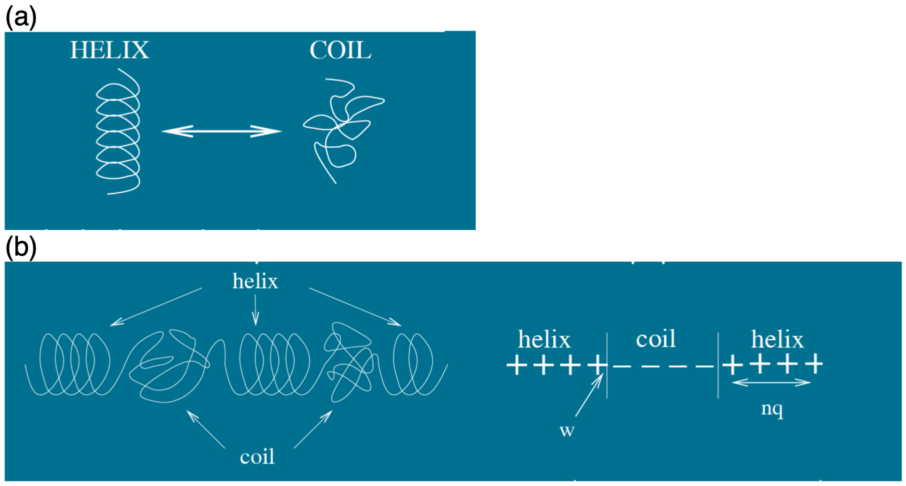


Figure 1: Helix–coil transition in polypeptides. (a) Helix–coil transition in a single segment. (b) Multiple helices and coils in a long polypeptide.

We consider a polypeptide consisting of a large number N of independent segments at thermal equilibrium with a heat bath at temperature T. Each segment can be in either the helix or coil state.

Our goal is to determine how thermodynamic quantities, such as internal energy, entropy and molar fraction of helical and coil domains depend on temperature.

(A) Ensemble average internal energy
Let $n_k^{(j)} = \pm 1$ denote the state of segment k in configuration j, where $n_k = -1$ corresponds to the α-helical state and $n_k = +1$ to the coiled state. The total energy of configuration j is

$$E_j = \sum_{k=1}^{N} n_k^{(j)} \Delta. \tag{17}$$

The canonical partition function is therefore

$$Z = \sum_j e^{-\beta E_j} = \prod_{k=1}^{N} \sum_{n_k=\pm 1} e^{-\beta n_k \Delta} = \left[2\cosh(\beta\Delta)\right]^N. \tag{18}$$

The ensemble average internal energy follows from

$$\bar{E} = -\frac{\partial \ln Z}{\partial \beta}, \tag{19}$$

which yields

$$\bar{E} = -N\Delta \tanh(\beta\Delta). \tag{20}$$

Physical interpretation. Each segment behaves like a two-level system. At high temperatures ($\beta\Delta \ll 1$), thermal fluctuations populate helix and coil states nearly equally, so $\bar{E} \approx 0$. At low temperatures ($\beta\Delta \gg 1$), most segments occupy the lower-energy helical state and the energy approaches its minimum value $-N\Delta$.

(B) Entropy

The entropy in the canonical ensemble is given by the Gibbs definition

$$S = -\sum_j p_j \ln p_j, \qquad p_j = Z^{-1} e^{-\beta E_j}. \tag{21}$$

Equivalently,

$$S = k\beta\bar{E} + k\ln Z. \tag{22}$$

Substituting the expressions for $\bar{E}$ and Z, we obtain

$$S = -\frac{\Delta N}{T}\tanh(\beta\Delta) + k\ln\left[(2\cosh(\beta\Delta))^N\right]. \tag{23}$$

Physical interpretation. The entropy measures the number of accessible helix–coil configurations. At high temperatures all 2^N configurations are nearly equally probable and $S \to k\ln(2^N) = Nk\ln 2$. As the temperature decreases, the system becomes increasingly ordered and the entropy decreases accordingly.

(C) Fraction of helical segments

Let $N_j^{(\alpha)}$ be the number of helical segments in configuration j. The total energy may be written as

$$E_j = N\Delta - 2\Delta N_j^{(\alpha)}. \tag{24}$$

The average number of helical segments can be obtained from the partition function via

$$\frac{\partial \ln Z}{\partial(\beta\Delta)} = 2\bar{N}^{(\alpha)} - N. \tag{25}$$

Defining the fraction of helical segments as

$$\chi_\alpha = \frac{\bar{N}^{(\alpha)}}{N}, \tag{26}$$

we find

$$\chi_\alpha = \frac{1}{2}\left[1 + \tanh(\beta\Delta)\right]. \tag{27}$$

Physical interpretation. The helix fraction interpolates smoothly between two limits. At high temperatures, $\chi_\alpha \to 1/2$, indicating equal probabilities of helix and coil states. At low temperatures, $\chi_\alpha \to 1$, meaning that nearly all segments are helical. The transition is gradual rather than sharp, reflecting the absence of cooperativity in this simple noninteracting model.

4 Entropy

Gibbs introduced the following definition of entropy:

$$S = -k \sum_j P_j \ln P_j, \tag{28}$$

where P_j is the probability of finding the system in microscopic state j and k is Boltzmann's constant. Remarkably, this is the *unique* function of the probabilities alone that is *extensive*: if a system is composed of independent subsystems, its total entropy is the sum of the entropies of the subsystems.

Extensivity is a fundamental requirement for a definition of entropy, and it plays a central role in establishing Eq. (28) as the correct definition, as shown in the following exercise.

4.1 Exercise: Entropy Extensivity

Show that the entropy defined in Eq. (28) is extensive. That is, for two uncorrelated subsystems A and B, prove that

$$S_{AB} = S_A + S_B.$$

Solution. If subsystems A and B are independent, the joint probability of finding A in configuration j_A and B in configuration j_B factorizes:

$$P_{j_A, j_B} = P_{j_A} P_{j_B}.$$

The entropy of the combined system is therefore

$$\begin{aligned} S_{AB} &= -k \sum_{j_A} \sum_{j_B} P_{j_A, j_B} \ln P_{j_A, j_B} \\ &= -k \sum_{j_A} \sum_{j_B} P_{j_A} P_{j_B} \ln\left(P_{j_A} P_{j_B}\right). \end{aligned} \tag{29}$$

Using the identity $\ln(ab) = \ln a + \ln b$, this becomes

$$\begin{aligned} S_{AB} &= -k \sum_{j_A} \sum_{j_B} P_{j_A} P_{j_B} \left[\ln P_{j_A} + \ln P_{j_B}\right] \\ &= -k \sum_{j_A} P_{j_A} \ln P_{j_A} \sum_{j_B} P_{j_B} - k \sum_{j_B} P_{j_B} \ln P_{j_B} \sum_{j_A} P_{j_A}. \end{aligned} \tag{30}$$

Since probabilities are normalized, $\sum_{j_A} P_{j_A} = \sum_{j_B} P_{j_B} = 1$, we obtain

$$S_{AB} = S_A + S_B,$$

as required.

One can go further and show that the logarithm is essentially the *only* function that satisfies this additivity property. Suppose a function $f(p)$ obeys

$$f(p_{j_A} p_{j_B}) = f(p_{j_A}) + f(p_{j_B}).$$

Taking the partial derivative with respect to p_{j_A} gives

$$\frac{\partial f(p_{j_A} p_{j_B})}{\partial (p_{j_A} p_{j_B})} p_{j_B} = \frac{\partial f(p_{j_A})}{\partial p_{j_A}}, \tag{31}$$

and similarly for p_{j_B},

$$\frac{\partial f(p_{j_A} p_{j_B})}{\partial (p_{j_A} p_{j_B})}\, p_{j_A} = \frac{\partial f(p_{j_B})}{\partial p_{j_B}}. \tag{32}$$

Combining these results yields

$$p \frac{df(p)}{dp} = c, \tag{33}$$

where c is a constant. Integrating,

$$f(p) = c \ln p,$$

showing that the logarithm is uniquely singled out by the requirement of extensivity.

4.2 Maximum Entropy Principle

We now show that maximizing the entropy leads directly to the Boltzmann distribution,

$$P_j = Z^{-1} e^{-\beta E_j}, \tag{34}$$

for systems with fixed extensive variables (such as volume V and particle number N) and fixed average energy,

$$E = \sum_j P_j E_j. \tag{35}$$

The derivation uses the method of Lagrange multipliers to maximize the entropy subject to the constraints of energy conservation and probability normalization. Equivalently, we minimize the functional

$$f = k \sum_j P_j \ln P_j + \gamma \left(\sum_j P_j E_j - E \right) + \gamma' \left(\sum_j P_j - 1 \right), \tag{36}$$

where γ and γ' are Lagrange multipliers.

The extremum condition,

$$\left. \frac{\partial f}{\partial P_j} \right|_X = 0, \tag{37}$$

yields

$$k \left(\ln P_j + 1 \right) + \gamma E_j + \gamma' = 0. \tag{38}$$

Solving for P_j, we obtain

$$P_j = \exp(-\beta E_j) \exp\left[- \left(\frac{\gamma'}{k} + 1 \right) \right], \tag{39}$$

where $\beta = \gamma / k$.

Finally, normalization $\sum_j P_j = 1$ determines the prefactor,

$$\exp\left[- \left(\frac{\gamma'}{k} + 1 \right) \right] = \left(\sum_j e^{-\beta E_j} \right)^{-1} \equiv Z^{-1},$$

where Z is the partition function. Thus,

$$P_j = Z^{-1} e^{-\beta E_j}, \tag{40}$$

which is the Boltzmann distribution.

5 Stationarity, Constancy, and Liouville's Theorem

As shown in the previous section, a classical system at equilibrium is described by an ensemble with the following phase-space probability density:

$$P(x,p) = Z^{-1} e^{-\beta H(x,p)}, \tag{41}$$

where $H(x,p)$ is the Hamiltonian, $\beta = (kT)^{-1}$, and

$$Z = \int dx\, dp\, e^{-\beta H(x,p)}$$

is the canonical partition function.

5.1 Classical Liouville Equation

The total time derivative of a phase-space density $P(x,p,t)$ along a trajectory is

$$\frac{dP}{dt} = \frac{\partial P}{\partial t} + \dot{x}\frac{\partial P}{\partial x} + \dot{p}\frac{\partial P}{\partial p}. \tag{42}$$

Using Hamilton's equations,

$$\dot{x} = \frac{\partial H}{\partial p}, \qquad \dot{p} = -\frac{\partial H}{\partial x}, \tag{43}$$

Eq. (42) becomes

$$\frac{dP}{dt} = \frac{\partial P}{\partial t} + \frac{\partial H}{\partial p}\frac{\partial P}{\partial x} - \frac{\partial H}{\partial x}\frac{\partial P}{\partial p} = \frac{\partial P}{\partial t} + \{P,H\}. \tag{44}$$

where we have introduced the *Poisson bracket* which for any two functions $A(x,p)$ and $B(x,p)$ is defined, as follows:

$$\{A,B\} \equiv \frac{\partial A}{\partial x}\frac{\partial B}{\partial p} - \frac{\partial A}{\partial p}\frac{\partial B}{\partial x}. \tag{45}$$

Poisson brackets encode the structure of classical Hamiltonian dynamics and provide a compact way to express time evolution.

Liouville's theorem, discussed in Sec. D, states that the probability density is conserved along classical trajectories:

$$\frac{dP}{dt} = 0. \tag{46}$$

Equation (44) therefore implies the *Liouville equation*,

$$\frac{\partial P}{\partial t} = -\{P,H\}. \tag{47}$$

5.2 Stationarity of the Equilibrium Distribution

For the equilibrium density in Eq. (41), the probability depends on phase-space coordinates only through the Hamiltonian, i.e. $P = f(H)$. Using the definition of the Poisson bracket,

$$\{P,H\} = \{f(H),H\} = f'(H)\{H,H\} = 0, \tag{48}$$

since the Poisson bracket of any function with itself vanishes. Substituting into Eq. (47) yields

$$\frac{\partial P}{\partial t} = 0. \tag{49}$$

Thus, we find that the equilibrium phase-space density is *stationary*.

5.3 Constancy Along Trajectories

Combining stationarity with Liouville's theorem, we obtain

$$\frac{dP}{dt} = \frac{\partial P}{\partial t} + \{P, H\} = 0.$$

Therefore, the equilibrium density $P(x, p)$ is not only time-independent, but also *constant along every Hamiltonian trajectory in phase space*. This result provides the classical foundation for equilibrium statistical mechanics.

6 Temperature

So far, the parameter

$$T \equiv \frac{1}{k\beta} = \frac{1}{\gamma}$$

has appeared simply as the inverse of the Lagrange multiplier γ introduced in the entropy maximization problem. At this stage, this definition is purely formal and does not yet justify calling T the *temperature*.

A deeper and more useful definition follows from substituting $P_j = Z^{-1}e^{-\beta E_j}$ into the definition of entropy, $S = -k_B \sum_j P_j \ln P_j$, giving

$$S = \frac{E}{T} + k_B lnZ, \tag{50}$$

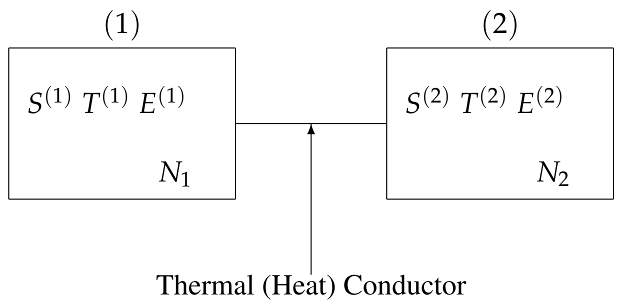
which relates T directly to the entropy:

$$\frac{1}{T} = \left(\frac{\partial S}{\partial E}\right)_N. \tag{51}$$

This equation tells us that temperature measures how sensitively the entropy changes when energy is added to the system, while keeping the number of particles fixed.

The goal of this section is to show that this quantity T has exactly the defining property of temperature in thermodynamics: *it is uniform throughout a system in thermal equilibrium.* In other words, when two systems can exchange energy freely, equilibrium requires that they share the same T.

Consider an isolated system at equilibrium, with a fixed total number of particles N and total energy E, in the state of maximum entropy. Imagine dividing the system into two subsystems (or compartments), labeled (1) and (2), which are separated by a wall that allows the transfer of energy (heat) but not particles. The situation is illustrated schematically below.



Each compartment is characterized by its entropy, energy, and temperature:

$$(S^{(1)}, E^{(1)}, T^{(1)}), \qquad (S^{(2)}, E^{(2)}, T^{(2)}),$$

with N_1 and N_2 particles, respectively.

Now imagine a small amount of energy δE being transferred from compartment (1) to compartment (2). This implies

$$\delta E^{(1)} = -\,\delta E, \qquad \delta E^{(2)} = \delta E. \tag{52}$$

Because the total system was initially in a state of *maximum entropy*, any such small redistribution of energy can only decrease the entropy. Therefore, the resulting change in entropy must satisfy

$$\delta S_{E,N} \leq 0. \tag{53}$$

The total change in entropy is the sum of the changes in each compartment:

$$\begin{aligned} \delta S &= \delta S^{(1)} + \delta S^{(2)} \\ &= \left(\frac{\partial S^{(1)}}{\partial E^{(1)}}\right)_{N_1} \delta E^{(1)} + \left(\frac{\partial S^{(2)}}{\partial E^{(2)}}\right)_{N_2} \delta E^{(2)} \\ &= \left(-\frac{1}{T^{(1)}} + \frac{1}{T^{(2)}}\right) \delta E \leq 0. \end{aligned} \tag{54}$$

Crucially, the energy transfer δE could be either positive or negative. The inequality in Eq. (54) must therefore hold for *any* infinitesimal exchange of energy. The only way this can be guaranteed is if

$$\boxed{T^{(1)} = T^{(2)}}.$$

We conclude that equilibrium requires equality of T across all parts of the system. This result justifies identifying the parameter T, defined through

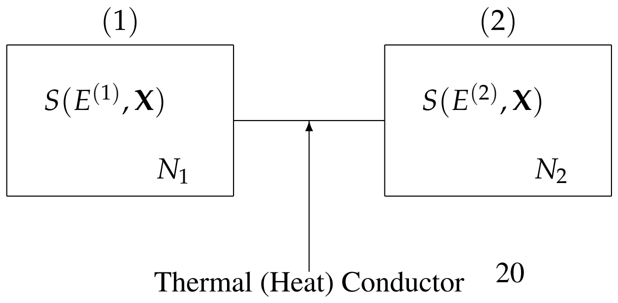
$$\frac{1}{T} = \left(\frac{\partial S}{\partial E}\right)_N,$$

with the thermodynamic temperature.

7 Minimum Energy Principle

The minimum energy principle follows directly from the maximum entropy principle. While the latter states that equilibrium corresponds to a maximum of entropy at fixed energy, the former describes the same equilibrium state from a complementary viewpoint: *at fixed entropy, equilibrium corresponds to a minimum of energy*.

To see this explicitly, consider a system at thermal equilibrium composed of two subsystems, labeled (1) and (2), which can exchange energy but not particles. Each subsystem is characterized by an entropy $S(E^{(i)}, \mathbf{X})$, where $\mathbf{X}$ denotes all other extensive variables held fixed (such as volume and particle number).



Now imagine imposing a *constraint* on the system by transferring a small amount of energy δE from subsystem (2) to subsystem (1). The new energy distribution becomes

$$E^{(1)} \to E^{(1)} + \delta E, \qquad E^{(2)} \to E^{(2)} - \delta E.$$

Because the system was initially in equilibrium, its entropy was maximal. Any such constrained redistribution of energy must therefore reduce the total entropy:

$$S(E^{(1)} + \delta E, \mathbf{X}) + S(E^{(2)} - \delta E, \mathbf{X}) < S(E^{(1)}, \mathbf{X}) + S(E^{(2)}, \mathbf{X}). \tag{55}$$

Now remove the constraint and allow the system to relax back to equilibrium while keeping the entropy fixed. Let E denote the total energy of the unconstrained equilibrium state. By definition,

$$S(E, \mathbf{X}) = S(E^{(1)} + \delta E, \mathbf{X}) + S(E^{(2)} - \delta E, \mathbf{X}). \tag{56}$$

Combining Eqs. (55) and (56), we obtain

$$S(E, \mathbf{X}) < S(E^{(1)}, \mathbf{X}) + S(E^{(2)}, \mathbf{X}). \tag{57}$$

Since temperature is defined by

$$\left(\frac{\partial S}{\partial E}\right)_{V,N} = \frac{1}{T} > 0, \tag{58}$$

entropy is a monotonically increasing function of energy. Therefore, the inequality in Eq. (57) implies

$$E < E^{(1)} + E^{(2)}. \tag{59}$$

We conclude that imposing internal constraints at fixed entropy forces the system away from equilibrium and increases its total energy. Conversely, when entropy is held fixed, the equilibrium state is the one that minimizes the internal energy:

$$dE\Big|_{S,V} \geq 0. \tag{60}$$

This statement is known as the *minimum energy principle*.

Physical Example

Consider two balloons filled with nitrogen gas at room temperature, placed in thermal contact. At equilibrium, they share the same temperature and collectively minimize their internal energy at fixed entropy.

Lowering the temperature would reduce the molecular kinetic energy and therefore the total internal energy. Because $\partial S/\partial E > 0$, this reduction in energy would also reduce the entropy.

However, entropy can also be reduced without changing the total energy. For example, imagine exchanging a small number of fast molecules from one balloon with slow molecules from the other, then separating the balloons. One balloon becomes slightly warmer and the other slightly colder. Although the total energy remains unchanged, the system is now out of equilibrium and has lower entropy.

If the balloons are then allowed to re-equilibrate while keeping the entropy fixed, the only way to return to equilibrium is by reducing the total internal energy. Thus, equilibrium can be reached either by maximizing entropy at fixed energy or by minimizing energy at fixed entropy—two equivalent descriptions of the same physical principle.

8 Canonical and Microcanonical Enseles

In the previous sections we showed that, at thermal equilibrium, the probability of finding a system in microstate j is given by the *Boltzmann probability distribution*

$$p_j = Z^{-1} \exp(-\beta E_j), \tag{61}$$

where $\beta = (kT)^{-1}$, T is the temperature of the ensemble, k is the Boltzmann constant, and Z is the partition function that ensures normalization of the probabilities.

This expression applies to a *canonical ensemble*, in which the system is in thermal contact with a heat reservoir and is therefore allowed to exchange energy. As a result, different microstates generally have different energies E_j, and lower-energy states are more probable than higher-energy ones.

A particularly important limiting case is the *microcanonical ensemble*. In this ensemble, the system is completely isolated: its energy, volume, and number of particles are all fixed. Consequently, every accessible microstate has exactly the same energy,

$$E_j = E.$$

Because the system has no way to exchange energy with its surroundings, there is no physical reason to favor one accessible microstate over another. The fundamental assumption of statistical mechanics is therefore that all such microstates are *equally likely*. If W denotes the total number of accessible microstates, then

$$p_j = \frac{1}{W}. \tag{62}$$

Substituting this uniform probability distribution into Gibbs' definition of entropy,

$$S = -k \sum_j p_j \ln p_j,$$

we immediately obtain

$$S = k \ln W. \tag{63}$$

This expression is the famous formula introduced by Boltzmann. It provides a direct and profound connection between macroscopic thermodynamics and microscopic physics: *entropy measures the logarithm of the number of accessible microscopic configurations of the system.*

In summary, the canonical ensemble describes systems at fixed temperature, where microstates are weighted according to their energy, while the microcanonical ensemble describes isolated systems, where all accessible microstates are equally probable. Despite their different assumptions, both ensembles lead to consistent definitions of entropy and equilibrium in the appropriate limits.

9 Equivalency of Ensembles

A central idea in statistical mechanics is that macroscopic properties of a system should not depend on the particular ensemble used to describe it, provided the system is sufficiently large and in equilibrium. In particular, the microcanonical and canonical ensembles must yield the same thermodynamic predictions. We now show how this equivalence emerges in a transparent and general way.

Consider the canonical ensemble, whose partition function is defined as

$$Z = \sum_k e^{-\beta E_k} = e^{-\beta A}, \tag{64}$$

Figure 2: Boltzmann's grave in Vienna, bearing the formula $S = k \ln W$.

where the sum runs over all microstates k, each with energy E_k, and A is the Helmholtz free energy.

Many distinct microstates may share the same energy. It is therefore natural to reorganize the sum by grouping together all states with identical energy. Let E_l denote a distinct energy level, and let $\Omega(E_l)$ be the number of microstates with that energy. Rewriting the partition function in terms of these energy levels gives

$$Z = \sum_l \Omega(E_l)\, e^{-\beta E_l}. \tag{65}$$

For macroscopic systems, the energy spectrum is extremely dense, and it is convenient to replace the discrete sum over energy levels by an integral. In this continuous limit, Eq. (65) becomes

$$Z = \int_0^\infty dE\, \Omega(E)\, e^{-\beta E}. \tag{66}$$

Equation (66) shows that the canonical partition function $Z(\beta)$ is the *Laplace transform* of the microcanonical density of states $\Omega(E)$. Since the Laplace transform is unique, there is a one-to-one correspondence between $Z(\beta)$ and $\Omega(E)$: knowing either function determines the other.

This result demonstrates that the canonical and microcanonical ensembles contain the same statistical information about the system. Although they describe equilibrium in different ways—one at fixed temperature and the other at fixed energy—they are mathematically equivalent and lead to identical thermodynamic predictions in the thermodynamic limit.

9.1 Example: Two-Level System

As a concrete illustration of the equivalence between the microcanonical and canonical ensembles, we compute the internal energy of a system of N noninteracting two-level particles. The purpose of this example is to show explicitly that the ensemble-average internal energy is identical in both descriptions.

Each particle can occupy either a ground state with energy 0 or an excited state with energy ϵ.

Microcanonical Ensemble

Let m denote the number of particles in the excited state. The total internal energy of the system is then

$$E = m\epsilon = \sum_{j=1}^{N} n_j \epsilon, \tag{67}$$

where $n_j = 0, 1$.

For a fixed value of m, the number of microstates is equal to the number of ways of choosing m excited particles out of N:

$$\Omega(E) = \binom{N}{m} = \frac{N!}{m!(N-m)!}. \tag{68}$$

The entropy is therefore

$$S = k \ln \Omega(E). \tag{69}$$

The temperature follows from the microcanonical definition

$$\frac{1}{T} = \left(\frac{\partial S}{\partial E}\right)_N = \frac{1}{\epsilon}\left(\frac{\partial S}{\partial m}\right)_N. \tag{70}$$

Using Stirling's approximation for large N,

$$\ln N! \simeq N \ln N - N, \tag{71}$$

and similarly for $m!$ and $(N-m)!$, we obtain

$$\frac{1}{T} = \frac{k}{\epsilon} \ln\left(\frac{N-m}{m}\right). \tag{72}$$

Introducing $\beta = (kT)^{-1}$, this relation can be written as

$$\beta\epsilon = \ln\left(\frac{N}{m} - 1\right). \tag{73}$$

Solving for m/N, we find

$$\boxed{\frac{m}{N} = \frac{1}{1+e^{\beta\epsilon}}} \quad \Rightarrow \quad \boxed{E = \frac{N\epsilon}{1+e^{\beta\epsilon}}}. \tag{74}$$

Canonical Ensemble

In the canonical ensemble, the system is in thermal contact with a heat reservoir at temperature T. Since the particles do not interact, the partition function factorizes:

$$Z = \prod_{j=1}^{N} \sum_{n_j=0}^{1} e^{-\beta\epsilon n_j} = \left(1 + e^{-\beta\epsilon}\right)^N . \tag{75}$$

Taking the logarithm,

$$\ln Z = N \ln\left(1 + e^{-\beta\epsilon}\right) . \tag{76}$$

The internal energy is obtained from

$$E = -\left(\frac{\partial \ln Z}{\partial \beta}\right)_{N,V} = \frac{N\epsilon}{1 + e^{\beta\epsilon}}. \tag{77}$$

This result is identical to the one obtained in Eq. (74) using the microcanonical ensemble. The two descriptions therefore yield the same equilibrium internal energy, explicitly demonstrating the equivalence of ensembles for this system.

10 Thermal Fluctuations

In the previous section we showed that, for a system of N two-level particles, the ensemble-average internal energy is the same whether it is computed in the microcanonical or canonical ensemble, provided N is sufficiently large. There is, however, an important conceptual difference between the two descriptions.

In the microcanonical ensemble, the internal energy is fixed by construction and does not fluctuate from one member of the ensemble to another. In contrast, in the canonical ensemble the system is in thermal contact with a heat reservoir, and its energy fluctuates among different replicas of the system.

The purpose of this section is to show that this apparent difference does not contradict the equivalence of ensembles. The key point is that, although energy fluctuations exist in the canonical ensemble, their *relative magnitude* becomes vanishingly small as N increases.

We begin by computing the mean-squared fluctuation of the internal energy,

$$(\delta E)^2 \equiv \langle (E - \langle E \rangle)^2 \rangle,$$

in the canonical ensemble. Using the density operator $\hat{\rho}$, this can be written as

$$(\delta E)^2 = \mathrm{Tr}\{\hat{\rho}(\hat{H} - \bar{E})^2\} = \mathrm{Tr}\{\hat{\rho}\left(\hat{H}^2 - 2\hat{H}\bar{E} + \bar{E}^2\right)\} , \tag{78}$$

where $\bar{E} \equiv \langle E \rangle$.

Using $\mathrm{Tr}\{\hat{\rho}\} = 1$ and $\mathrm{Tr}\{\hat{\rho}\hat{H}\} = \bar{E}$, this expression simplifies to

$$(\delta E)^2 = \mathrm{Tr}\{\hat{\rho}\hat{H}^2\} - \bar{E}^2. \tag{79}$$

Writing the trace explicitly in the energy eigenbasis yields

$$(\delta E)^2 = \sum_k p_k E_k^2 - \left(\sum_k p_k E_k\right)^2 , \tag{80}$$

where $p_k = Z^{-1}e^{-\beta E_k}$ and $Z = \mathrm{Tr}\{e^{-\beta\hat{H}}\}$ is the canonical partition function.

Substituting these expressions, we obtain

$$(\delta E)^2 = \frac{\mathrm{Tr}\{\hat{H}^2 e^{-\beta\hat{H}}\}}{Z} - \frac{\left(\mathrm{Tr}\{\hat{H}e^{-\beta\hat{H}}\}\right)^2}{Z^2}. \tag{81}$$

Noting that derivatives of Z with respect to β generate powers of the Hamiltonian, this expression can be rewritten as

$$(\delta E)^2 = \frac{1}{Z}\frac{\partial^2 Z}{\partial\beta^2} - \frac{1}{Z^2}\left(\frac{\partial Z}{\partial\beta}\right)^2. \tag{82}$$

This leads to a compact and useful result:

$$(\delta E)^2 = \frac{\partial^2 \ln Z}{\partial\beta^2} = -\left(\frac{\partial E}{\partial\beta}\right)_{N,V}. \tag{83}$$

We now examine the relative size of the energy fluctuations. Dividing the root-mean-square fluctuation by the mean energy gives

$$\frac{\sqrt{(\delta E)^2}}{E} = \frac{\sqrt{-\partial E/\partial\beta}}{E} \propto \frac{1}{\sqrt{N}}. \tag{84}$$

Thus, although the absolute energy fluctuations grow with system size, their *relative magnitude* decreases as $N^{-1/2}$. For macroscopic systems, where $N \sim 10^{23}$, these fluctuations are utterly negligible. This is why the canonical and microcanonical ensembles yield the same macroscopic predictions, despite their different treatments of energy.

Connection to Heat Capacity

As a by-product of this analysis, we obtain a remarkable and important result: *energy fluctuations in the canonical ensemble are directly related to the heat capacity of the system.*

Using Eq. (83) and the relation $\beta = (kT)^{-1}$, we find

$$(\delta E)^2 = -\left(\frac{\partial E}{\partial T}\right)_{N,V}\left(\frac{\partial T}{\partial\beta}\right) = kT^2\left(\frac{\partial E}{\partial T}\right)_{N,V}. \tag{85}$$

Defining the heat capacity at constant volume as

$$C_V \equiv \left(\frac{\partial E}{\partial T}\right)_{N,V}, \tag{86}$$

we obtain the final result

$$(\delta E)^2 = kT^2 C_V. \tag{87}$$

This relation shows that the system's response to changes in temperature is encoded in the size of its equilibrium energy fluctuations—a deep and general connection between thermodynamic response functions and statistical fluctuations.

10.1 Exercise: Probability of a Small Fluctuation

A closed system in thermal equilibrium with a reservoir at temperature T has (up to normalization) the energy probability density

$$P(E) \propto \Omega(E)\, e^{-\beta E}.$$

Estimate the probability of observing an energy fluctuation

$$E = \bar{E} + \delta E, \qquad \delta E = 10^{-6}\, \bar{E},$$

by computing the ratio $P(E)/P(\bar{E})$.

Take 10^{-3} moles of an ideal gas, for which

$$\bar{E} = \frac{3}{2} N k_B T, \qquad C_V = \frac{3}{2} N k_B.$$

Solution

A convenient way to estimate $P(E)/P(\bar{E})$ is to expand $\ln P(E)$ about its maximum at $E = \bar{E}$:

$$\ln P(E) = \ln P(\bar{E}) + \left.\frac{\partial \ln P}{\partial E}\right|_{\bar{E}} \delta E + \frac{1}{2} \left.\frac{\partial^2 \ln P}{\partial E^2}\right|_{\bar{E}} (\delta E)^2 + \cdots . \tag{88}$$

10.1.0.1 Step 1: The linear term vanishes at equilibrium. Since $\ln P(E) = \ln \Omega(E) - \beta E + \text{const}$,

$$\left.\frac{\partial \ln P}{\partial E}\right|_{\bar{E}} = \left.\frac{\partial \ln \Omega}{\partial E}\right|_{\bar{E}} - \beta = \frac{1}{k_B T} - \beta = 0, \tag{89}$$

because $\beta = (k_B T)^{-1}$. Thus the first nonzero correction is quadratic.

10.1.0.2 Step 2: Evaluate the curvature. Differentiate once more:

$$\left.\frac{\partial^2 \ln P}{\partial E^2}\right|_{\bar{E}} = \frac{\partial}{\partial E} \left.\left(\frac{\partial \ln \Omega}{\partial E}\right)\right|_{\bar{E}} = \frac{\partial}{\partial E} \left.\left(\frac{1}{k_B T}\right)\right|_{\bar{E}} \tag{90}$$

$$= -\frac{1}{k_B T^2} \left.\frac{\partial T}{\partial E}\right|_{\bar{E}} = -\frac{1}{k_B T^2\, C_V}, \tag{91}$$

since $C_V = \left(\frac{\partial E}{\partial T}\right)_{N,V}$ implies $\left(\frac{\partial T}{\partial E}\right)_{N,V} = 1/C_V$.

Therefore,

$$\ln \frac{P(E)}{P(\bar{E})} = \ln P(E) - \ln P(\bar{E}) = -\frac{1}{2} \frac{(\delta E)^2}{k_B T^2\, C_V}. \tag{92}$$

10.1.0.3 Step 3: Substitute ideal-gas expressions. With $C_V = \frac{3}{2} N k_B T$ and $\bar{E} = \frac{3}{2} N k_B T$, we have $C_V T = \bar{E}$ and

$$\ln \frac{P(E)}{P(\bar{E})} = -\frac{1}{2} \frac{(\delta E)^2}{k_B T^2\, C_V} = -\frac{1}{2} \frac{(10^{-6} \bar{E})^2}{k_B T^2\, C_V}. \tag{93}$$

Using $\bar{E} = \frac{3}{2} N k_B T$ gives

$$\ln \frac{P(E)}{P(\bar{E})} = -\frac{3}{4} N\, (10^{-6})^2. \tag{94}$$

Now compute N for 10^{-3} moles:

$$N = (10^{-3})N_A \approx 10^{-3} \times 6.023 \times 10^{23} = 6.023 \times 10^{20}.$$

Hence,

$$\ln \frac{P(E)}{P(\bar{E})} = -\frac{3}{4}(6.023 \times 10^{20})(10^{-12}) = -\frac{3}{4}(6.023 \times 10^{8}) \approx -4.52 \times 10^{8}. \tag{95}$$

10.1.0.4 Final answer.

$$\boxed{\frac{P(E)}{P(\bar{E})} = \exp\left(-4.52 \times 10^{8}\right) \approx 0} \tag{96}$$

In words: a 10^{-6} *fractional* energy fluctuation is astronomically unlikely even for a "small" macroscopic sample like 10^{-3} moles.

11 Grand Canonical Ensemble

In the canonical ensemble the system can exchange *energy* with a heat reservoir, so the temperature T is fixed while the particle number N is fixed. In many physical situations (e.g., adsorption on a surface, electrons in a metal exchanging with leads, molecules in a container connected to a gas supply), the system can exchange *both* energy and particles with its environment. The appropriate description is then the *grand canonical ensemble*.

Our goal is to maximize the Gibbs entropy

$$S = -k \sum_j p_j \ln p_j, \tag{97}$$

subject to the constraints that (at fixed volume V) the ensemble has a fixed *average* energy and a fixed *average* particle number:

$$E = \sum_j p_j E_j, \tag{98}$$

$$\bar{N} = \sum_j p_j N_j. \tag{99}$$

Physically, the grand canonical ensemble represents a collection of replicas in equilibrium with

- a *heat reservoir* that fixes the temperature T, and
- a *particle reservoir* that fixes the chemical potential μ (at the same temperature T).

The system is free to exchange energy and particles with these reservoirs, so E and N fluctuate from replica to replica, but their averages E and $\bar{N}$ are fixed.

11.1 Maximum-Entropy Derivation

We use Lagrange multipliers to enforce the constraints in the minimization of

$$\begin{aligned} f(p_1, p_2, \ldots) \equiv k\sum_j p_j \ln p_j + \gamma\left(\sum_j p_j E_j - E\right) + \gamma'\left(\sum_j p_j N_j - \bar{N}\right) \\ + \gamma''\left(\sum_j p_j - 1\right), \end{aligned} \tag{100}$$

where $\gamma, \gamma', \gamma''$ are Lagrange multipliers.

Taking the derivative with respect to p_j and setting it to zero gives

$$\frac{\partial f}{\partial p_j} = k(\ln p_j + 1) + \gamma E_j + \gamma' N_j + \gamma'' = 0. \tag{101}$$

Solving for p_j,

$$p_j = \exp\left(-1 - \frac{\gamma''}{k}\right) \exp\left(-\frac{\gamma}{k}E_j - \frac{\gamma'}{k}N_j\right). \tag{102}$$

It is conventional to introduce

$$\beta \equiv \frac{1}{kT} \qquad \beta \equiv \frac{\gamma}{k} \qquad \beta\mu \equiv -\frac{\gamma'}{k}$$

so that the probability distribution takes the familiar form

$$p_j = \frac{e^{-\beta E_j + \beta\mu N_j}}{\sum_j e^{-\beta E_j + \beta\mu N_j}} = \Xi^{-1} e^{-\beta E_j + \beta\mu N_j}. \tag{103}$$

The normalization factor

$$\Xi \equiv \sum_j e^{-\beta E_j + \beta\mu N_j} \tag{104}$$

is called the *grand canonical partition function*.

Equation (103) is the *generalized Boltzmann distribution*: microstates are weighted not only by their energy E_j, but also by their particle number N_j. The chemical potential μ plays the role of the "price" (in energy units) for adding a particle.

11.2 Entropy and the Grand Partition Function

Substituting Eq. (103) into the entropy definition (97) gives

$$S = -k\sum_j p_j \left(-\beta E_j + \beta\mu N_j - \ln \Xi\right). \tag{105}$$

Using $\sum_j p_j = 1$, $\sum_j p_j E_j = E$, and $\sum_j p_j N_j = \bar{N}$, this becomes

$$S = k\beta E - k\beta\mu\bar{N} + k\ln\Xi. \tag{106}$$

Solving for $\ln\Xi$ yields

$$\ln\Xi = \beta\left(-E + TS + \mu\bar{N}\right). \tag{107}$$

Since the Helmholtz free energy is $A \equiv E - TS$, we obtain

$$\ln \Xi = \beta \left(-A + \mu \bar{N}\right). \tag{108}$$

Important note. The identity above should be read as the definition of the *grand potential*

$$\Phi_G \equiv -kT \ln \Xi,$$

which satisfies $\Phi_G = A - \mu \bar{N}$ (and, in the thermodynamic limit, $\Phi_G = E - TS - \mu N$).

11.3 Relation Between Z and Ξ

A common and very useful relationship is obtained by expressing the grand canonical partition function as a sum over canonical partition functions at fixed particle number:

$$\boxed{\Xi(\beta, \mu, V) = \sum_{N=0}^{\infty} e^{\beta \mu N} \, Z_N(\beta, V)} \tag{109}$$

where $Z_N(\beta, V)$ is the canonical partition function for a system with exactly N particles.

We now derive Eq. (109) carefully, because it reveals what the grand canonical ensemble *really* is: a mixture of canonical ensembles with different N.

Step 1: Start from the definition of Ξ

By definition, the grand canonical partition function is a sum over *all* microstates j accessible when both energy and particle number are allowed to fluctuate:

$$\Xi(\beta, \mu, V) \equiv \sum_j e^{-\beta E_j + \beta \mu N_j}. \tag{110}$$

Here each microstate j has an energy E_j and a particle number N_j.

Step 2: Group microstates by particle number

The key observation is that the full set of microstates can be partitioned into disjoint groups labeled by N:

$$\{\text{all microstates}\} = \bigcup_{N=0}^{\infty} \{\text{microstates with particle number } N\},$$

and no microstate belongs to two different groups because N_j is a well-defined integer.

Therefore we can rewrite the sum over all microstates as an "outer" sum over N, and for each N an "inner" sum over microstates having that particle number:

$$\Xi(\beta, \mu, V) = \sum_{N=0}^{\infty} \sum_{j:\, N_j = N} e^{-\beta E_j + \beta \mu N}. \tag{111}$$

Step 3: Pull out the factor $e^{\beta \mu N}$

Inside the inner sum, N is fixed (all states in that inner sum have the same particle number), so the factor $e^{\beta \mu N}$ is constant with respect to j and can be pulled outside:

$$\Xi(\beta, \mu, V) = \sum_{N=0}^{\infty} e^{\beta \mu N} \sum_{j:\, N_j = N} e^{-\beta E_j}. \tag{112}$$

Step 4: Recognize the canonical partition function Z_N

But the inner sum is precisely the definition of the canonical partition function for fixed N:

$$Z_N(\beta, V) \equiv \sum_{j:\, N_j=N} e^{-\beta E_j}. \tag{113}$$

Substituting Eq. (113) into Eq. (112) immediately gives

$$\Xi(\beta, \mu, V) = \sum_{N=0}^{\infty} e^{\beta\mu N}\, Z_N(\beta, V), \tag{114}$$

which is exactly the boxed result in Eq. (109).

Physical interpretation. Equation (109) shows that the grand canonical ensemble is a weighted sum of canonical ensembles with different particle numbers. The weight of each N-sector is controlled by $e^{\beta\mu N}$:

- if μ is large and positive, larger N are favored;
- if μ is very negative, states with fewer particles dominate.

In this sense, μ plays the role of a "knob" that tunes the typical particle number in equilibrium.

Optional viewpoint (useful later). From Eq. (109), one can also read off the probability that the system contains exactly N particles:

$$P(N) = \frac{e^{\beta\mu N} Z_N(\beta, V)}{\Xi(\beta, \mu, V)}. \tag{115}$$

This makes the mixture interpretation completely explicit: the system samples different N with probability $P(N)$, and within each fixed N it behaves canonically.

11.4 Thermodynamic Identities in the Grand Canonical Ensemble

Once the grand canonical partition function

$$\Xi(\beta, \mu, V) = \sum_{N=0}^{\infty} e^{\beta\mu N} Z_N(\beta, V)$$

is known, all thermodynamic properties follow from simple derivatives of $\ln \Xi$.

Average particle number.

$$\boxed{\bar{N} = \frac{1}{\beta}\left(\frac{\partial \ln \Xi}{\partial \mu}\right)_{\beta, V}} \tag{116}$$

Average internal energy.

$$\boxed{E = -\left(\frac{\partial \ln \Xi}{\partial \beta}\right)_{\mu, V} + \mu\bar{N}} \tag{117}$$

Entropy. Using $S = -\left(\partial\Phi_G/\partial T\right)_{\mu,V}$ with $\Phi_G = -kT\ln\Xi$, one finds

$$\boxed{S = k\ln\Xi + k\beta E - k\beta\mu\bar{N}} \tag{118}$$

Grand potential and pressure.

The grand canonical ensemble is most naturally described in terms of the *grand potential*

$$\Phi_G(\beta, \mu, V) \equiv -kT \ln \Xi(\beta, \mu, V). \tag{119}$$

This quantity plays for the grand canonical ensemble the same role that the Helmholtz free energy $A = -kT \ln Z$ plays for the canonical ensemble: once Φ_G is known, all equilibrium thermodynamics follow from its derivatives.

Step 1: Identify the natural variables of Φ_G

From its definition, Φ_G is a function of (T, μ, V). Its differential can be obtained by combining the fundamental thermodynamic identity

$$dE = T\,dS - P\,dV + \mu\,dN$$

with the definition of the grand potential,

$$\Phi_G \equiv E - TS - \mu N.$$

Taking the differential,

$$\begin{aligned} d\Phi_G &= dE - T\,dS - S\,dT - \mu\,dN - N\,d\mu \\ &= (T\,dS - P\,dV + \mu\,dN) - T\,dS - S\,dT - \mu\,dN - N\,d\mu \\ &= -S\,dT - P\,dV - N\,d\mu. \end{aligned} \tag{120}$$

This shows that Φ_G has the natural variables (T, V, μ) and immediately gives the thermodynamic derivatives

$$S = -\left(\frac{\partial \Phi_G}{\partial T}\right)_{\mu,V}, \qquad N = -\left(\frac{\partial \Phi_G}{\partial \mu}\right)_{T,V}, \qquad P = -\left(\frac{\partial \Phi_G}{\partial V}\right)_{T,\mu}. \tag{121}$$

Step 2: Derive pressure in terms of Ξ

Using $\Phi_G = -kT \ln \Xi$ in Eq. (121), we obtain

$$\begin{aligned} P &= -\left(\frac{\partial \Phi_G}{\partial V}\right)_{T,\mu} = -\left(\frac{\partial}{\partial V}\left[-kT \ln \Xi(T, \mu, V)\right]\right)_{T,\mu} \\ &= kT \left(\frac{\partial \ln \Xi}{\partial V}\right)_{T,\mu}. \end{aligned} \tag{122}$$

So, in full generality,

$$\boxed{P = kT \left(\frac{\partial \ln \Xi}{\partial V}\right)_{T,\mu}.} \tag{123}$$

Step 3: Why $\Phi_G = -PV$ in the thermodynamic limit

For a homogeneous macroscopic system (no special boundaries, no long-range fields), thermodynamic potentials are *extensive*: they scale in proportion to the system size. In particular, at fixed T and μ, the only extensive variable in $\Phi_G(T, \mu, V)$ is V, so in the thermodynamic limit one must have

$$\Phi_G(T, \mu, V) = V\,\phi_G(T, \mu), \tag{124}$$

where $\phi_G(T,\mu)$ is an intensive quantity (independent of V).

Using Eq. (121),

$$P = -\left(\frac{\partial \Phi_G}{\partial V}\right)_{T,\mu} = -\left(\frac{\partial}{\partial V}\left[V\phi_G(T,\mu)\right]\right)_{T,\mu} = -\phi_G(T,\mu).$$

Substituting back into Eq. (124) gives

$$\boxed{\Phi_G(T,\mu,V) = -PV} \tag{125}$$

in the thermodynamic limit.

Step 4: The commonly used expression for P

Combining $\Phi_G = -kT\ln\Xi$ with $\Phi_G = -PV$ yields

$$-kT\ln\Xi = -PV \qquad \Rightarrow \qquad \boxed{P = \frac{kT}{V}\ln\Xi}. \tag{126}$$

Important remark. Equation (126) relies on the thermodynamic-limit scaling $\Phi_G \propto V$ (homogeneity/extensivity). When surface effects or strong finite-size effects are important, the more general derivative form in Eq. (123) should be used.

Summary. The grand canonical ensemble trades the fixed particle number N of the canonical ensemble for a fixed chemical potential μ. In return, thermodynamic observables are generated directly from $\ln\Xi$ by differentiation, making the grand canonical ensemble especially powerful for systems with particle exchange, quantum gases, and many-body problems.

12 Density Fluctuations

In this section we study fluctuations in the number of particles in a system at fixed volume V and temperature T. Our main goals are:

- to show that particle-number fluctuations can be computed directly from the grand canonical partition function, and
- to demonstrate that the *relative size* of these fluctuations scales as $1/\sqrt{\overline{N}}$ and therefore becomes negligible in macroscopic systems.

Particle-number fluctuations in the grand canonical ensemble

The ensemble-averaged squared fluctuation in the number of particles is defined as

$$\overline{(\delta N)^2} \equiv \overline{(N-\overline{N})^2} = \overline{N^2} - \overline{N}^2, \tag{127}$$

where the overline denotes an ensemble average. Writing this explicitly in terms of microstates j,

$$\overline{(\delta N)^2} = \sum_j p_j N_j^2 - \left(\sum_j p_j N_j\right)^2. \tag{128}$$

In the grand canonical ensemble, the probability of microstate j is

$$p_j = \Xi^{-1} e^{-\beta E_j + \beta\mu N_j},$$

where Ξ is the grand partition function. Substituting this expression into Eq. (128) gives

$$\overline{(\delta N)^2} = \Xi^{-1} \sum_j N_j^2 e^{-\beta E_j + \beta\mu N_j} - \left(\Xi^{-1} \sum_j N_j e^{-\beta E_j + \beta\mu N_j} \right)^2 . \tag{129}$$

This expression can be written compactly in terms of derivatives of Ξ:

$$\overline{(\delta N)^2} = \left(\frac{\partial^2 \ln \Xi}{\partial(\beta\mu)^2} \right)_V = \left(\frac{\partial \overline{N}}{\partial(\beta\mu)} \right)_V . \tag{130}$$

Physical interpretation. Equation (130) shows that number fluctuations are not independent quantities: they are directly tied to the system's response to changes in chemical potential. Large fluctuations correspond to a system that is highly sensitive to particle exchange with the reservoir, while small fluctuations indicate strong stability of the particle number.

12.1 Example: Uncorrelated particles in boxes

To gain intuition, consider a simple model in which uncorrelated particles are distributed among m boxes. Assume that the average occupancy of each box is very small, $\langle n_j \rangle \ll 1$, so that each box contains either zero or one particle: $n_j = 0$ or 1. The total number of particles is

$$N = \sum_{j=1}^{m} n_j.$$

The squared fluctuation in N is

$$\overline{(\delta N)^2} = \overline{N^2} - \overline{N}^2 = \sum_{j=1}^{m} \sum_{k=1}^{m} \langle n_j n_k \rangle - \left(\sum_{j=1}^{m} \langle n_j \rangle \right)^2 . \tag{131}$$

Separating diagonal and off-diagonal terms,

$$\begin{aligned} \overline{(\delta N)^2} = & \sum_{j=1}^{m} \langle n_j^2 \rangle + \sum_{j \neq k} \langle n_j n_k \rangle \\ & - \sum_{j=1}^{m} \langle n_j \rangle^2 - \sum_{j \neq k} \langle n_j \rangle \langle n_k \rangle . \end{aligned} \tag{132}$$

Since $n_j = 0, 1$, we have $\langle n_j^2 \rangle = \langle n_j \rangle$. Because the particles are uncorrelated, $\langle n_j n_k \rangle = \langle n_j \rangle \langle n_k \rangle$ for $j \neq k$. These terms cancel, leaving

$$\overline{(\delta N)^2} = \sum_{j=1}^{m} \langle n_j \rangle - \sum_{j=1}^{m} \langle n_j \rangle^2 = \sum_{j=1}^{m} \langle n_j \rangle (1 - \langle n_j \rangle). \tag{133}$$

For $\langle n_j \rangle \ll 1$, this simplifies to

$$\overline{(\delta N)^2} = m \langle n_1 \rangle = \overline{N}. \tag{134}$$

Therefore,

$$\frac{\sqrt{\overline{(\delta N)^2}}}{\overline{N}} = \frac{1}{\sqrt{\overline{N}}}. \tag{135}$$

Physical interpretation. The absolute size of particle-number fluctuations grows as $\sqrt{\overline{N}}$, but the *relative* size of fluctuations decreases as $1/\sqrt{\overline{N}}$. For macroscopic systems with $\overline{N} \sim 10^{23}$, relative fluctuations are extraordinarily small, explaining why thermodynamic variables appear sharply defined in everyday systems.

Connection to thermodynamics
Combining Eqs. (130) and (134), we obtain

$$\overline{N} = \left(\frac{\partial \overline{N}}{\partial(\beta\mu)}\right)_V. \tag{136}$$

This implies

$$\left(\frac{\partial \ln \overline{N}}{\partial(\beta\mu)}\right)_V = 1. \tag{137}$$

Integrating and dividing by the volume V, we find

$$\ln\left(\frac{N}{V}\right) = \beta\mu + c, \tag{138}$$

where c is a constant.

Physical interpretation. This result expresses the fundamental relationship between particle density, temperature, and chemical potential. It shows that the chemical potential controls the average density of particles in the system, providing the thermodynamic link between microscopic particle exchange and macroscopic observables.

13 Pure States

13.1 From classical ensembles to quantum states.

The classical framework developed above provides a powerful and intuitive description of equilibrium systems in terms of phase-space trajectories, probability densities, and ensemble averages. For many molecular systems at sufficiently high temperatures, this picture offers quantitatively accurate predictions. However, at the microscopic scale, nature is fundamentally quantum mechanical. Energy levels become discrete, phase-space points are replaced by state vectors, and observables are represented by operators rather than functions of (q, p).

Despite these differences, the core questions of statistical mechanics remain unchanged: how do we assign probabilities to microscopic states, how do we compute expectation values of observables, and how do macroscopic thermodynamic laws emerge from microscopic dynamics? To address these questions in the quantum regime, we must first understand the most complete description available for an individual quantum system—the *pure state*. This provides the foundation upon which quantum ensembles and statistical mixtures are built.

A *pure state* represents the most complete description that quantum mechanics allows for a physical system. It is described by a state vector (or *ket*) $|\psi\rangle$. Intuitively, a pure state corresponds to a situation in which we have maximal information about the system's quantum state.[1]

[1]If you are becoming comfortable with bra–ket notation, you may find the notes at this link helpful.

The time evolution of a pure state is governed by the time-dependent Schrödinger equation,

$$i\hbar \frac{\partial|\psi\rangle}{\partial t} = \hat{H}|\psi\rangle, \tag{139}$$

where $\hat{H}$ is the Hamiltonian operator. This equation is *deterministic*: once the initial state $|\psi(0)\rangle$ is specified, the state at all future times is fully determined. In this sense, $|\psi\rangle$ encodes the maximum possible information about the system.

13.2 Expanding a pure state in an eigenbasis

To connect the abstract state vector to physical measurements, it is useful to expand $|\psi\rangle$ in the eigenstates of an observable. Let $\hat{o}$ be a Hermitian operator representing an observable (such as position, momentum, or energy), with eigenstates $|\phi_k\rangle$ and eigenvalues o_k,

$$\hat{o}|\phi_k\rangle = o_k|\phi_k\rangle. \tag{140}$$

The state $|\psi\rangle$ can then be written as

$$|\psi\rangle = \sum_k a_k|\phi_k\rangle, \tag{141}$$

where the complex coefficients a_k specify how much each eigenstate contributes to $|\psi\rangle$.
It is convenient to express these coefficients in terms of amplitudes and phases,

$$a_k = \sqrt{p_k}\, e^{i\theta_k}, \tag{142}$$

where $p_k \geq 0$ and $\sum_k p_k = 1$. The quantity p_k has a direct physical meaning: it is the probability of obtaining the measurement outcome o_k when the observable $\hat{o}$ is measured in the state $|\psi\rangle$.

13.3 Expectation values

The *expectation value* of the observable $\hat{o}$ in the state $|\psi\rangle$ is

$$\langle\psi|\hat{o}|\psi\rangle = \sum_k p_k\, o_k. \tag{143}$$

This result is a probability-weighted average of the possible measurement outcomes.
More generally, consider an operator $\hat{A}$ that does not share the same eigenstates as $\hat{o}$. Its expectation value can still be computed using the $\{|\phi_k\rangle\}$ basis,

$$\langle\psi|\hat{A}|\psi\rangle = \sum_k p_k\langle\phi_k|\hat{A}|\phi_k\rangle + \sum_k \sum_{j\neq k} \sqrt{p_k p_j}\, e^{i(\theta_j-\theta_k)}\langle\phi_k|\hat{A}|\phi_j\rangle. \tag{144}$$

The first term looks like a classical average over states. The second term depends on the *relative phases* $\theta_j - \theta_k$ and represents *quantum coherence* (interference) between different basis states. These coherence terms are responsible for uniquely quantum phenomena such as interference and superposition.

13.4 Pure states and ensembles

Now imagine preparing a large number N of identical replicas of the system, each described by the same state vector $|\psi\rangle$. This collection is still described by a *pure state*. In this case, ensemble averages coincide with quantum expectation values:

$$\langle\hat{o}\rangle = \langle\psi|\hat{o}|\psi\rangle, \qquad \langle\hat{A}\rangle = \langle\psi|\hat{A}|\psi\rangle.$$

This equivalence becomes especially important when we later contrast pure states with *mixed states*, where incomplete information requires a different description.

13.5 Example: Two-level system in NMR (spin-$\frac{1}{2}$ nucleus)

In nuclear magnetic resonance (NMR), many nuclei of chemical interest (e.g., ^{1}H, ^{13}C, ^{15}N) behave, to an excellent approximation, as spin-$\frac{1}{2}$ particles. When placed in a strong static magnetic field B_0, such a nucleus has two Zeeman energy levels corresponding to spin alignment parallel or antiparallel to the field. These two levels form a natural realization of a quantum two-level system.

Let $\{|\alpha\rangle, |\beta\rangle\}$ denote the eigenstates of the Zeeman Hamiltonian,

$$\hat{H}_Z = -\gamma\hbar B_0 \hat{I}_z,$$

with eigenvalues E_α and E_β. These states are eigenstates of $\hat{I}_z$ and play the role of the computational basis $\{|0\rangle, |1\rangle\}$ in a qubit description.

A general pure state of a single nuclear spin can be written as

$$|\psi\rangle = a_\alpha|\alpha\rangle + a_\beta|\beta\rangle = \sqrt{p_\alpha}\,|\alpha\rangle + \sqrt{p_\beta}\,e^{i\theta}|\beta\rangle, \tag{145}$$

where $p_\alpha + p_\beta = 1$ and θ is the relative quantum phase between the Zeeman states.

If one measures the spin projection along the magnetic field, corresponding to the observable $\hat{I}_z$, the possible outcomes are $\pm\frac{1}{2}\hbar$ with probabilities p_α and p_β. The expectation value is therefore

$$\langle \hat{I}_z \rangle = p_\alpha \left(\tfrac{1}{2}\hbar\right) + p_\beta \left(-\tfrac{1}{2}\hbar\right), \tag{146}$$

which depends only on the level populations and is insensitive to the relative phase θ.

By contrast, NMR experiments typically probe transverse spin components, such as $\hat{I}_x$ or $\hat{I}_y$, which do *not* commute with $\hat{I}_z$. For a general observable $\hat{A}$ that couples the two Zeeman states, the expectation value is

$$\langle \hat{A} \rangle = p_\alpha\langle\alpha|\hat{A}|\alpha\rangle + p_\beta\langle\beta|\hat{A}|\beta\rangle + 2\sqrt{p_\alpha p_\beta}\,\mathrm{Re}\left[e^{i\theta}\langle\alpha|\hat{A}|\beta\rangle\right]. \tag{147}$$

The final term depends explicitly on the relative phase θ and represents *quantum coherence*. In NMR, this coherence corresponds to transverse magnetization and gives rise to the observable NMR signal. In the language of quantum information, it represents the off-diagonal elements of a qubit's density matrix, which are manipulated by control pulses and ultimately determine the system's ability to store and process quantum information.

14 Statistical Mixtures of States

In practice, it is often impossible to prepare a quantum system in exactly the same pure state every time. Instead, we consider a large collection of independently prepared replicas of the system. This collection is called an *ensemble*. Each replica is prepared under the same macroscopic conditions, but may occupy different microscopic quantum states.

An ensemble of N replicas is said to be in a *statistical mixture* of states $\{|\phi_k\rangle\}$ if n_k members of the ensemble are found in state $|\phi_k\rangle$. The probability of finding a replica in state $|\phi_k\rangle$ is then

$$p_k = \frac{n_k}{N}, \qquad \sum_k p_k = 1.$$

The essential point is conceptual: *each individual replica is in a definite state* $|\phi_k\rangle$. The randomness is not a quantum superposition within one system, but rather our uncertainty about which state a given replica occupies.

Ensemble averages for mixed states

Because the replicas are not coherently superposed, ensemble averages are computed as probability-weighted sums of expectation values within each state. For an observable $\hat{A}$,

$$A = \sum_k p_k \langle \phi_k | \hat{A} | \phi_k \rangle, \tag{148}$$

and for an observable $\hat{o}$ with eigenvalues o_k,

$$o = \sum_k p_k \langle \phi_k | \hat{o} | \phi_k \rangle = \sum_k p_k o_k. \tag{149}$$

It is useful to compare these expressions with the pure-state expectation value in Eq. (144). The ensemble average o in Eq. (149) has the same form as the corresponding pure-state expression in the $\{|\phi_k\rangle\}$ basis. However, the average A in Eq. (148) differs from the pure-state result because Eq. (148) contains only "diagonal" contributions. The "off-diagonal" interference terms present in Eq. (144) are absent.
Physical interpretation. A statistical mixture has *no phase coherence* between different states. Therefore, *there are no interference contributions to ensemble averages* coming from pairs of states $|\phi_k\rangle$ and $|\phi_j\rangle$ with $k \neq j$. In this sense, a statistical mixture behaves like a classical probability distribution over quantum states.

Incoherent superpositions

The statistical mixture above is also equivalent to an ensemble of replicas described by wavefunctions of the form

$$|\psi(\xi)\rangle = \sum_k \sqrt{p_k}\, e^{i\theta_k(\xi)} |\phi_k\rangle, \tag{150}$$

where ξ labels the members of the ensemble and the phases $\theta_k(\xi)$ are randomly and uniformly distributed.
Why this reproduces a mixture. When phases vary randomly from replica to replica, the interference terms average to zero in the ensemble. As a result, ensemble averages computed using Eq. (150) reduce to the mixture formula in Eq. (148). This provides a useful intuition: mixed states can be viewed as *incoherent* superpositions, in contrast to pure states which involve *coherent* superpositions with well-defined relative phases.

Common statistical ensembles

In what follows we introduce the most important statistical ensembles, focusing first on systems with a single molecular species.

- **Canonical ensemble (NVT).** All replicas are in thermal equilibrium with a heat reservoir at temperature T. This ensemble is the natural framework for laboratory conditions at fixed N, V, and T, and it is central to *Monte Carlo simulations*.

- **Microcanonical ensemble (NVE).** All replicas have the same energy E and the same number of particles N. Although less directly matched to typical laboratory conditions, it is central to *molecular dynamics simulations*, where energy is conserved by construction.

- **Grand canonical ensemble (μVT).** The replicas exchange both energy and particles with reservoirs at temperature T and chemical potential μ. This ensemble is useful for open systems (e.g., adsorption, exchange with a bath) where N fluctuates.

Exercise 1. Compute the ensemble average $\bar{A}$ associated with the incoherent superposition of states in Eq. (150) and verify that it coincides with Eq. (148).

15 Density Operator

The *density operator* provides a single, compact formalism for computing ensemble averages for both pure states and statistical mixtures. The key result is

$$A = \mathrm{Tr}\{\hat{\rho}\hat{A}\}, \tag{151}$$

where $\hat{\rho}$ encodes the state (pure or mixed) of the ensemble.

Definition of the density operator

For a statistical mixture of states $\{|\phi_k\rangle\}$ with probabilities $\{p_k\}$, the density operator is

$$\hat{\rho} = \sum_k p_k |\phi_k\rangle\langle\phi_k|. \tag{152}$$

For a pure state $|\psi\rangle$, this reduces to the projector

$$\hat{\rho} = |\psi\rangle\langle\psi|. \tag{153}$$

What $\hat{\rho}$ does for us. Instead of keeping track of either a state vector (pure state) or a list of states with probabilities (mixed state), we work with a single object $\hat{\rho}$. The trace formula in Eq. (151) then automatically produces the correct expectation values: it includes interference terms for pure states, and excludes them for mixtures.

Why the trace formula works (sketch)

To see why Eq. (151) reproduces pure-state expectation values, consider a pure state $|\psi\rangle = \sum_k a_k|\phi_k\rangle$ in a complete orthonormal basis $\{|\phi_k\rangle\}$, with $\langle\phi_{k'}|\phi_k\rangle = \delta_{kk'}$. Then

$$A = \mathrm{Tr}\{\hat{\rho}\hat{A}\} = \sum_{k'}\langle\phi_{k'}|\psi\rangle\langle\psi|\hat{A}|\phi_{k'}\rangle. \tag{154}$$

Substituting the expansion of $|\psi\rangle$ yields

$$A = \sum_{k'}\sum_j\sum_k\langle\phi_{k'}|\phi_k\rangle\, a_k a_j^*\langle\phi_j|\hat{A}|\phi_{k'}\rangle, \tag{155}$$

and using $\langle\phi_{k'}|\phi_k\rangle = \delta_{kk'}$ gives

$$A = \sum_k p_k\langle\phi_k|\hat{A}|\phi_k\rangle + \sum_k\sum_{j\neq k}\sqrt{p_k p_j}\, e^{i(\theta_k-\theta_j)}\langle\phi_j|\hat{A}|\phi_k\rangle, \tag{156}$$

which is identical to Eq. (144).
For a mixed state with $\hat{\rho}$ given by Eq. (152),

$$A = \mathrm{Tr}\{\hat{\rho}\hat{A}\} = \sum_{k'}\sum_k p_k\langle\phi_{k'}|\phi_k\rangle\langle\phi_k|\hat{A}|\phi_{k'}\rangle = \sum_k p_k\langle\phi_k|\hat{A}|\phi_k\rangle, \tag{157}$$

which reproduces Eq. (148).

Purity, coherence, and decoherence

The quantity $\mathrm{Tr}\{\hat{\rho}^2\}$ measures how "pure" a state is:

- $\mathrm{Tr}\{\hat{\rho}\} = 1$ for any physical state (pure or mixed).
- $\mathrm{Tr}\{\hat{\rho}^2\} = 1$ for pure states.
- $\mathrm{Tr}\{\hat{\rho}^2\} < 1$ for mixed states.

Thus, $\mathrm{Tr}\{\hat{\rho}^2\}$ can be used as a measure of coherence: as interference is lost (decoherence), the state becomes more mixed and $\mathrm{Tr}\{\hat{\rho}^2\}$ decreases.
Exercise 2.

(A) Show that $\mathrm{Tr}\{\hat{\rho}\} = 1$ for both mixed and pure states.

(B) Show that $\mathrm{Tr}\{\hat{\rho}^2\} = 1$ for pure states.

(C) Show that $\mathrm{Tr}\{\hat{\rho}^2\} \leq 1$ for mixed states.

16 Time Evolution of Ensembles

The time evolution of both pure and mixed states can be written in a single equation for the density operator:

$$\frac{\partial \hat{\rho}}{\partial t} = -\frac{[\hat{\rho}, \hat{H}]}{i\hbar}. \tag{158}$$

This is the *Liouville–von Neumann equation*. It is the density-matrix analogue of the Schrödinger equation: instead of evolving $|\psi\rangle$, we evolve $\hat{\rho}$.
Exercise 3. Using the Schrödinger equation (Eq. (139)), show that Eq. (158) describes the time evolution of $\hat{\rho}$ for a pure state.
Exercise 4. Using the linearity of Eq. (139), show that Eq. (158) also describes the time evolution of $\hat{\rho}$ for a mixed state.

Example: Two-level mixed state (contrast with a qubit pure state)

For a two-level system with basis states $\{|0\rangle, |1\rangle\}$, a pure state can be written as

$$|\psi\rangle = \sqrt{p_0}\,|0\rangle + \sqrt{p_1}\,e^{i\theta}|1\rangle, \qquad p_0 + p_1 = 1.$$

Here the relative phase θ is meaningful and affects expectation values of observables that do not commute with $\hat{o}$.
A mixed state with the same populations is instead described by

$$\hat{\rho} = p_0|0\rangle\langle 0| + p_1|1\rangle\langle 1|.$$

This state cannot be represented by a single $|\psi\rangle$ unless $p_0 = 1$ or $p_1 = 1$.
For an observable diagonal in this basis, both descriptions give the same average. However, for an observable that couples the two levels, the mixed state produces only diagonal contributions (no interference term). In other words, the mixed state does not retain a well-defined relative phase between $|0\rangle$ and $|1\rangle$.

Pure vs. mixed states: key differences

Property	Pure State	Mixed State
Mathematical description	$\|\psi\rangle$	$\hat{\rho} = \sum_k p_k \|\phi_k\rangle\langle\phi_k\|$
Information content	maximal	incomplete (statistical uncertainty)
Coherence between states	yes	no
Interference terms	present	absent
$\mathrm{Tr}\{\hat{\rho}^2\}$	$= 1$	< 1
Typical realization	isolated, well-controlled	thermal/open systems, noise

This comparison highlights why the density operator is essential in statistical mechanics: it naturally describes statistical uncertainty and decoherence, which are unavoidable in realistic physical systems.

17 Classical Analogue

Many core ideas of quantum statistical mechanics have direct classical counterparts. Seeing these parallels helps clarify what changes when we move from classical to quantum descriptions.

Microscopic states

In *quantum* statistical mechanics, microscopic states live in Hilbert space and are specified by quantum numbers (or by a state vector $|\psi\rangle$).
In *classical* statistical mechanics, a microscopic state is a point in phase space,

$$(x_1, \dots, x_f;\ p_1, \dots, p_f),$$

where f is the number of degrees of freedom. At any instant, the system occupies one such point.

Ensembles

Quantum ensembles are described by the density operator $\hat{\rho}$. Classical ensembles are described by a phase-space density

$$\rho = \rho(x_1, \dots, x_f, p_1, \dots, p_f),$$

where $\rho\, d\mathbf{x}\, d\mathbf{p}$ is the probability of finding the system in the corresponding phase-space volume element.

Time evolution of ensembles: Liouville's theorem

The quantum evolution equation for $\hat{\rho}$ (Eq. (158)) has a classical analogue known as *Liouville's theorem*:

$$\frac{\partial \rho}{\partial t} = -\left(\rho, H\right), \tag{159}$$

where ρ is the phase-space density and $(\cdot, \cdot)$ denotes the Poisson bracket. This equation expresses conservation of probability in phase space: as trajectories flow, probability is transported without being created or destroyed.

Poisson brackets and commutators

The classical analogue of the quantum commutator $[G,F]/(i\hbar)$ is the Poisson bracket,

$$(G,F) \equiv \sum_{j=1}^{f} \left(\frac{\partial G}{\partial x_j}\frac{\partial F}{\partial p_j} - \frac{\partial G}{\partial p_j}\frac{\partial F}{\partial x_j} \right). \tag{160}$$

In the appropriate classical limit, commutators reduce to Poisson brackets, linking quantum and classical dynamics.
Exercise 5. Prove Eq. (159) using Hamilton's equations,

$$\frac{dp_j}{dt} = -\frac{\partial H}{\partial x_j}, \qquad \frac{dx_j}{dt} = \frac{\partial H}{\partial p_j}, \tag{161}$$

where

$$H = \sum_{j=1}^{f} \frac{p_j^2}{2m_j} + V(x_1, \ldots, x_f).$$

Ensemble averages

Quantum ensemble averages are computed via

$$\langle \hat{A} \rangle = \mathrm{Tr}\{\hat{\rho}\hat{A}\}.$$

The classical analogue is

$$\bar{A} = \frac{\int d\mathbf{x} \int d\mathbf{p}\, \rho(\mathbf{x},\mathbf{p})\, A(\mathbf{x},\mathbf{p})}{\int d\mathbf{x} \int d\mathbf{p}\, \rho(\mathbf{x},\mathbf{p})}, \tag{162}$$

where $d\mathbf{x}\, d\mathbf{p}$ is a phase-space volume element.

Worked example: Classical harmonic oscillator in phase space

Consider the one-dimensional harmonic oscillator

$$H(x,p) = \frac{p^2}{2m} + \frac{1}{2}kx^2.$$

A microscopic state is a point (x,p) in phase space. Hamilton's equations,

$$\dot{x} = \frac{p}{m}, \qquad \dot{p} = -kx,$$

generate periodic motion. Eliminating time gives

$$\frac{p^2}{2m} + \frac{1}{2}kx^2 = E,$$

which is an ellipse in phase space. Thus, at fixed energy E, the system repeatedly visits the same closed curve of microscopic states.
Now consider an ensemble of oscillators with the same energy E but different initial phases. The phase-space density $\rho(x,p)$ is supported on the energy ellipse and is transported along it without distortion, in accordance with Liouville's theorem.

The time average of any observable $A(x,p)$ over one period is

$$\langle A \rangle_{\text{time}} = \frac{1}{T}\int_0^T A(x(t),p(t))\,dt.$$

For example,

$$\langle V \rangle_{\text{time}} = \left\langle \frac{1}{2}kx^2 \right\rangle = \frac{E}{2},$$

because the oscillator exchanges energy equally between kinetic and potential forms over a cycle.
This example illustrates: microscopic states as phase-space points, deterministic trajectories from Hamilton's equations, conservation of phase-space probability, and the relationship between time averages and ensemble averages in simple systems.

18 Entropy

The *entropy* $\bar{S}$ of an ensemble can be defined in terms of the density operator $\hat{\rho}$ as follows,

$$S \equiv -kTr\{\hat{\rho}\ \ln\hat{\rho}\}, \tag{163}$$

where k is the Botzmann constant. Equation (22) is the Von Neumann definition of entropy. This is the most fundamental definition of S because it is given in terms of the density operator $\hat{\rho}$, which provides the most complete description of an ensemble. In particular, the *Gibbs entropy formula*,

$$S = -k\sum_k p_k \ln p_k, \tag{164}$$

can be obtained from Eq. (163) by substituting $\hat{\rho}$ in accord with Eq. (152).

From Eq. (164) one can see that the entropy of a pure state is zero, while the entropy of a statistical mixture is always positive. Therefore,

$$S \geq 0, \tag{165}$$

which is the *fourth law of Thermodynamics*.

18.1 Exercise: Entropy Extensivity

Show that the definition of entropy, introduced by Eq. (164), fulfills the requirement of extensivity (*i.e.*, when dividing the system into fragments A and B, the entropy of the system including both fragments S_{AB} equals the sum of the entropies of the fragments S_A and S_B).

Solution: We consider that the fragments are independent so the joint probability p_{j_A,j_B} of configurations j_A and j_B of fragments A and B is equal to the product of the probabilities p_{j_A} and p_{j_B} of the configurations of each fragment. Therefore, $S_{AB} = \sum_{j_A}\sum_{j_B} p_{j_A,j_B}\ln(p_{j_A,j_B}) = \sum_{j_A}\sum_{j_B} p_{j_A}p_{j_B}\ln(p_{j_A}p_{j_B})$, with $S_A = \sum_{j_A} p_{j_A}\ln(p_{j_A})$ and $S_B = \sum_{j_B} p_{j_B}\ln(p_{j_B})$. So, $S_{AB} = S_A + S_B$, since $\ln(p_{j_A}p_{j_B}) = \ln(p_{j_A}) + \ln(p_{j_B})$.

We can show that there is no other function but the logarithm that fulfills that condition, as follows. Consider a function that fulfills the following condition: $f(p_{j_A}p_{j_B}) = f(p_{j_A}) + f(p_{j_B})$ and compute the partial derivative with respect to p_{j_A}, as follows:

$$\frac{\partial f(p_{j_A}p_{j_B})}{\partial(p_{j_A}p_{j_B})}p_{j_B} = \frac{\partial f(p_{j_A})}{\partial p_{j_A}}. \tag{166}$$

Analogously, we compute the partial derivatives with respect to p_{j_B}, as follows:

$$\frac{\partial f(p_{j_A} p_{j_B})}{\partial(p_{j_A} p_{j_B})} p_{j_A} = \frac{\partial f(p_{j_B})}{\partial p_{j_B}}. \tag{167}$$

Therefore,

$$p_{j_A} \frac{\partial f(p_{j_A})}{\partial p_{j_A}} = p_{j_B} \frac{\partial f(p_{j_B})}{\partial p_{j_B}} = c, \tag{168}$$

where c is a constant.

Therefore, $\frac{\partial f(p_{j_A})}{\partial p_{j_A}} = \frac{c}{p_{j_A}}$ and $\int dp_{j_A} \frac{\partial f(p_{j_A})}{\partial p_{j_A}} = c \int dp_{j_A} \frac{1}{p_{j_A}}$, giving $f(p_{j_A}) = c \ln(p_{j_A})$.

19 Maximum-Entropy Density Operator

The goal of this section is to obtain the density operator $\hat{\rho}$, with $Tr\{\hat{\rho}\} = 1$, that maximizes the entropy $S = -k\text{Tr}\{\hat{\rho}\ln\hat{\rho}\}$ of a system characterized by an ensemble average internal energy

$$E = \text{Tr}\{\hat{\rho}\hat{H}\}, \tag{169}$$

and fix extensive properties X such as $X = (V, N)$ (i.e., canonical and microcanonical ensembles).
This is accomplished by implementing the method of Lagrange Multipliers to maximize the function

$$f(\hat{\rho}) \equiv -k\text{Tr}\{\hat{\rho}\ln\hat{\rho}\} + \gamma(E - Tr\{\hat{\rho}\hat{H}\}) + \gamma'(1 - Tr\{\hat{\rho}\}), \tag{170}$$

where γ and γ' are Lagrange Multipliers. We, therefore, solve for $\hat{\rho}$ from the following equation

$$\left.\frac{\partial f}{\partial \hat{\rho}}\right)_X = 0, \tag{171}$$

and we obtain that the density operator that satisfies Eq. (171) must satisfy the following equation:

$$Tr\{-k\ln\hat{\rho} - k - \gamma\hat{H} - \gamma'\} = 0. \tag{172}$$

Therefore,

$$-\ln\hat{\rho} = 1 + \frac{\gamma}{k}\hat{H} + \frac{\gamma'}{k}. \tag{173}$$

Exponentiating both sides of Eq. (173) we obtain

$$\hat{\rho} = \exp(-(1 + \frac{\gamma'}{k}))\exp(-\frac{\gamma}{k}\hat{H}), \tag{174}$$

and, since Tr$\{\hat{\rho}\}$=1,

$$\exp(-(1 + \frac{\gamma'}{k})) = \frac{1}{Z}, \tag{175}$$

where Z is the *partition function*

$$Z \equiv Tr\{exp(-\beta\hat{H})\}, \tag{176}$$

with $\beta \equiv \gamma/k$.

Substituting Eqs. (176) and (175) into Eq. (174), we obtain that the density operator that maximizes the entropy of the ensemble, subject to the contraint of average ensemble energy $\bar{E}$, is

$$\hat{\rho} = Z^{-1}\exp(-\beta\hat{H}). \tag{177}$$

Note that

$$\frac{\partial\hat{\rho}}{\partial t} = 0, \tag{178}$$

when $\hat{\rho}$ is defined according to Eq. (177) and, therefore, the system is at *equilibrium*.
Exercise 6: Use Eqs. (158) and (177) to prove Eq. (178).

20 Internal Energy and Helmholtz Free Energy

Substituting Eqs. (176) and (175) into Eq. (169) we obtain that the internal energy E can be computed from the partition function Z as follows,

$$E = -\frac{\partial \ln Z}{\partial \beta}\Big)_X . \tag{179}$$

Furthermore, substituting Eqs. (176) and (175) into Eq. (163) we obtain

$$S = -k\mathrm{Tr}\{\hat{\rho}(-\beta\hat{H} - \ln Z)\} = k\beta E + k\ln Z. \tag{180}$$

In the next section we prove that the parameter $T \equiv (k\beta)^{-1}$ can be identified with the *temperature* of the ensemble. Therefore,

$$A = E - TS = -kT\ln Z, \tag{181}$$

is the *Helmholtz free energy*, that according to Eq. (179) satisfies the following thermodynamic equation,

$$E = \frac{\partial(\beta A)}{\partial \beta}\Big)_X . \tag{182}$$

21 Notes for the Inquisitive Mind: Measures of Information

21.1 Shannon Entropy

The goal of this section is to show that the *Shannon entropy*,

$$S = -\sum_j P_j \ \log_2 P_j, \tag{183}$$

is a measure of an amount of information (*i.e.*, information = data). In fact, in this section, we show that if we consider a message composed of symbols x_j with probabilty P_j (as obtained when transmitted through a noisy communication channel), then S gives the average number of bits necessary to store each one of the symbols of that message. Such fundamental observation has given birth to the field of information theory and is known as the **Shannon's source coding theorem**, as follows: *It is impossible to compress the data of a long stream of a random variable such that the average number of bits per symbol is less than the Shannon entropy of the source without loosing information.*

As an example, consider that the 'message' is a set of n snapshots reporting the position of a fluctuating particle. Storing each symbol x_j requires specifying K bits (*e.g.*, an array of K elements that can be either 1 or 0, indicating for example which position is occupied by the particle when the coordinate range is discretized into 2^K boxes). The more snapshots (*i.e.*, 'symbols'), the more information the message has with regards to the particle whereabouts. Therefore, the amount of information S scales with the number n of symbols: $S = Kn$.

The number of bits required to store a symbol (*i.e.*, K) can be given in terms of the total number of possible symbols $\Omega = 2^K$, as follows: $K = -log_2(1/\Omega)$, where $1/\Omega$ is the probability of a symbol when randomly picked out of Ω equally probable choices. More generally, when considering two messages, one with n_1 symbols from a library of $\Omega_1 = 2^{K_1}$ possibilities, and the other one with n_2 symbols chosen from $\Omega_2 = 2^{K_2}$ possibilities, the amount of information in the two messages is the same when the number of possibilities for the two messages is equal, so $\Omega_1^{n_1} = \Omega_2^{n_2}$, since then $n_1 \log_2 \Omega_1 = n_2 \log_2 \Omega_2$ as pointed out by Hartley in the [*The Bell System Technical Journal*, July 1928, p. 535] so the total number of bits required to store the two messages is the same: $K_1 n_1 = K_2 n_2$ and $K_1/\log_2 \Omega_1 = K_2/\log_2 \Omega_2$. This relation holds for all values of Ω only when $K = K_0 \log \Omega$, where the arbitrary constant K_0 can be omitted since the logarithmic base is also arbitrary. The particular base of the log fixes the size of the unit of information. With the base 2, the resulting units are called binary digits, or more briefly *bits* as suggested by [J.W. Tukey]. If the base 10 is used, the units may be called decimal digits, with $\log_2 \Omega = \log_{10} \Omega / \log_{10} 2$.

The amount of information (*i.e*, the number of bits) of each symbol is therefore equal to the logarithm of the number of possible symbols. The amount of information of a message is the number of possible messages (which is the number of symbols times the number of possible symbols: $S = n \log_2 \Omega$). It is, therefore, clear that for a message composed of symbols with probability P_j, the average number of bits necessary to store one of its symbols is given by Eq. (183), as discussed by Claude E. Shannon in his seminal work on communication theory published in [*The Bell System Technical Journal* **27**: 379-423, 1948], with symbols drawn with probability P_j as randomly picked from $\Omega_j = 1/P_j$ equally probable choices.

21.2 Majorization

When comparing the entropies of two sources with probabilities $\mathbf{p} = (p_1, p_2, ..., p_n)$ with $p_{j+1} < p_j$ and $\mathbf{q} = (q_1, q_2, ..., q_n)$ with $q_{j+1} < q_j$, it is possible to show that $S(\mathbf{p}) < S(\mathbf{q})$ when $\mathbf{p}$ majorizes $\mathbf{q}$, as follows:

$$\sum_{i=1}^{j} p_i > \sum_{i=1}^{j} q_i, \tag{184}$$

for any $j < n$ with

$$\sum_{i=1}^{n} q_i = \sum_{i=1}^{n} p_i = 1. \tag{185}$$

This can be shown by substracting Eq. (184) from Eq (185) to obtain:

$$\sum_{i=j+1}^{n} q_i > \sum_{i=j+1}^{n} p_i. \tag{186}$$

Therefore,

$$\sum_{j=1}^{n} \left[\sum_{i=j+1}^{n} q_i \right] \log\left(\frac{q(i)}{q(j+1)} \right) > \sum_{j=1}^{n} \left[\sum_{i=j+1}^{n} p_i \right] \log\left(\frac{q(i)}{q(j+1)} \right). \tag{187}$$

Changing the order of the sums in Eq. (187), we obtain:

$$\sum_{i=1}^{n} \left[\sum_{j=1}^{i-1} q_i \right] \log\left(\frac{q(i)}{q(j+1)} \right) > \sum_{i=1}^{n} \left[\sum_{j=1}^{i-1} p_i \right] \log\left(\frac{q(i)}{q(j+1)} \right), \tag{188}$$

and canceling equal terms within the sums over j, we obtain:

$$\begin{aligned} \sum_{i=1}^{n} q_i \log\left(\frac{q(1)}{q(i)} \right) &> \sum_{i=1}^{n} p_i \log\left(\frac{q(1)}{q(i)} \right), \\ \log(q(1)) - \sum_{i=1}^{n} q_i \log(q(i)) &> \log(q(1)) - \sum_{i=1}^{n} p_i \log(q(i)), \\ -\sum_{i=1}^{n} q_i \log(q(i)) &> -\sum_{i=1}^{n} p_i \log(q(i)). \end{aligned} \tag{189}$$

Furthermore, since $\log(x) < x - 1$, we obtain:

$$\begin{aligned} \log\left(\frac{q(i)}{p(i)} \right) &< \frac{q(i)}{p(i)} - 1, \\ p(i)\log\left(\frac{q(i)}{p(i)} \right) &< q(i) - p(i), \\ \sum_{i=1}^{n} p(i)\log\left(\frac{q(i)}{p(i)} \right) &< 0, \\ \sum_{i=1}^{n} p(i)\log(q(i)) &< \sum_{i=1}^{n} p(i)\log(p(i)), \\ -\sum_{i=1}^{n} p(i)\log(q(i)) &> -\sum_{i=1}^{n} p(i)\log(p(i)). \end{aligned} \tag{190}$$

Therefore, according to Eqs. (189) and (190), we obtain:

$$-\sum_{i=1}^{n} q(i)\log(q(i)) > -\sum_{i=1}^{n} p(i)\log(p(i)). \tag{191}$$

21.3 Maximum Entropy Image Reconstruction

Maximum entropy reconstruction is a widely applicable technique for generating images with maximum information, from noisy data, as discussed by Skillin and Bryan in the [*Mon. Not. R. Astr. Soc.* (1984) **211**, 111-124]. The image is regarded as a set of positive numbers $f_1, \cdots, f_N$ giving the intensity of pixels which are to be determined for image reconstruction (e.g., the image of a galaxy, or cell culture). The observational constraints on permitted reconstructions come from data points D_k that are related to the actual image in some known way by means of the measuring device, and subject to some form of noise due to interference with the background. In a linear experiment, $D_k = \sum R_{kj} f_j + n_k \sigma_k$, where R is the instrument response, σ_k is the standard error on the data and n_k is a random variable of zero mean and unit variance. Naively, one may attempt to recover the image P from the data D by applying R^{-1} but this usually fails whenever the data are incomplete since R^{-1} is not uniquely defined, or when R^{-1} is badly conditioned as in most deconvolution problems.

The reconstruction is set up by comparing the noisy data D_k with the simulated data F_k which would be obtained in the absence of noise: $F_k = \sum R_{kj} f_j$. A reconstruction is said to be *feasible* if the simulated data agree with the actual data to within the noise. The misfit is usually measured by the chi-squared value: $\chi^2 = \sum_k (F_k - D_k)^2/\sigma_k^2$, where the summation is over the observed data points k.

According to the maximum entropy criteria, one selects the feasible image with greatest entropy: $S = -\sum_k P_k \log_2 P_k$, where $P_k = f_k / \sum_j f_j$, by maximizing S subject to minimum χ^2. Using the Lagrange multiplier method for $Q = S - \lambda\chi^2 - \mu(1 - \sum_k P_k)$, with some value of the Lagrange multipliers λ and μ, we make $\partial Q/\partial f_j = 0$ to obtain f_j.

As pointed out by Skilling and Bryan, the resulting maximum entropy fitting provides a unique approach to reconstruct an image without introducing correlations beyond those which are required by the data. Maximizing the entropy can also be justified in terms of information. Given an image radiating with intensity pattern f_j, the entropy measures the average number of bits needed to define (or store) the position j of a single radiated photon coming from the image. Maximizing S, subject to observational constraints, involves seeking a maximally non-biased answer to the fundamental question 'Where would the next photon come from?'. The practical merit of maximizing entropy is that the resulting image has minimum configurational information, so that there must be evidence in the data for any structure which is seen, and the displayed structure is uniquely easy to comprehend.

21.4 Fisher Index

Another measure of information is the Fisher index I, defined as follows:

$$\begin{aligned} I &= \int_{-\infty}^{\infty} dx P(x) \left(\frac{1}{P(x)} \frac{\partial P(x)}{\partial x} \right)^2, \\ &= \int_{-\infty}^{\infty} dx P(x) \left(\frac{\partial \log[P(x)]}{\partial x} \right)^2 \end{aligned} \tag{192}$$

For an ergodic system, $P(x)$ is determined by the time the variable is found at x over the total observation time, so the total probability is normalized:

$$\begin{aligned}1 &= \int_{-\infty}^{\infty} dx P, \\ &= \lim_{t_f\to\infty} \frac{1}{t_f} \int_{-t_f/2}^{t_f/2} dt, \\ &= \lim_{t_f\to\infty} \frac{1}{t_f} \int_{-\infty}^{\infty} dx \left|\frac{\partial t}{\partial x}\right|, \\ &= \lim_{t_f\to\infty} \frac{1}{t_f} \int_{-\infty}^{\infty} dx \frac{1}{\sqrt{v^2}},\end{aligned} \tag{193}$$

from where we identify $1/P = t_f\sqrt{v^2}$, with $t_f \to \infty$. Therefore,

$$\begin{aligned}I &= \lim_{t_f\to\infty} t_f \int_{-\infty}^{\infty} dx\sqrt{v^2} \left(\frac{\partial P}{\partial x}\right)^2, \\ &= \lim_{t_f\to\infty} t_f \int_{-\infty}^{\infty} dx v^2 \left|\frac{\partial t}{\partial x}\right| \left(\frac{\partial P}{\partial x}\right)^2, \\ &= \lim_{t_f\to\infty} t_f \int_{-\infty}^{\infty} dx \left|\frac{\partial t}{\partial x}\right| \left(\frac{\partial P}{\partial t}\right)^2, \\ &= \lim_{t_f\to\infty} t_f \int_{-\infty}^{\infty} dt \left(\frac{\partial P}{\partial t}\right)^2,\end{aligned} \tag{194}$$

In the third row of Eq. (194), we have used that $\partial P/\partial x = -1/v\, \partial P/\partial t$, since according to the continuity equation $dP/dt = \partial P/\partial t + v\partial P/\partial x$ and $dP/dt = 0$ for the system at equilibrium. Furthermore,

$$\begin{aligned}\frac{\partial P}{\partial t} &= \frac{1}{t_f}\frac{\partial}{\partial t}(v^2)^{-1/2}, \\ &= \frac{1}{t_f}\left(-\frac{1}{2}\right)(v^2)^{-3/2} 2v\frac{\partial v}{\partial t} \\ &= -\frac{1}{t_f}\frac{1}{v^2}\frac{\partial v}{\partial t} = -\frac{1}{t_f}\frac{a}{v^2} = -\frac{1}{t_f m}\frac{F}{v^2}.\end{aligned} \tag{195}$$

Therefore,

$$\begin{aligned}I &= \lim_{t_f\to\infty} \frac{1}{t_f} \int_{-t_f/2}^{t_f/2} dt \frac{F^2}{(mv^2)^2}, \\ &= \left\langle \frac{F^2}{(mv^2)^2} \right\rangle,\end{aligned} \tag{196}$$

21.5 Mutual Information

The goal of this section is to introduce measures of mutual information (or uncertainty) shared by variables x_1 and x_2. Such measures estimate the amount of uncertainty about a variable x_1 reduced by measuring a variable x_2. In particular, when x_1 and x_2 are completely uncorrelated, knowing one of the two variables does not provide any information about the other and the joint probability distribution $P(x_1, x_2)$ is equal to the product of the marginal probability distributions $P_1(x_1) = \int dx_2 P(x_1, x_2)$ and $P_2(x_2) = \int dx_1 P(x_1, x_2)$.

The most common measure of mutual information is the index I_{x_1,x_2}, defined as follows:

$$I_{x_1,x_2} = \int dx_1 dx_2 P(x_1,x_2) \left(\log[P(x_1,x_2)] - \log[P_1(x_1)P_2(x_2)]\right). \qquad (197)$$

Note that in the particular case of uncorrelated variables, $I_{x_1,x_2} = 0$.

Analogously, another measure of mutual information can be defined, as follows:

$$\tilde{I}_{x_1,x_2} = \int_{-\infty}^{\infty} dx_1 \int_{-\infty}^{\infty} dx_2 P \left(\frac{1}{P^2}\left(\frac{\partial P}{\partial x_1}\right)^2 + \frac{1}{P^2}\left(\frac{\partial P}{\partial x_2}\right)^2 - \frac{1}{P_1^2}\left(\frac{\partial P_1}{\partial x_1}\right)^2 - \frac{1}{P_2^2}\left(\frac{\partial P_2}{\partial x_2}\right)^2 \right). \qquad (198)$$

22 Bose-Einstein and Fermi-Dirac Distributions

Consider a system consisting of N quantum particles of a certain kind (e.g., bosons, or fermions with a certain spin). If the interaction of the particles is *weak* enough, each particle has its own motion which is independent of all others and the system is an ideal gas of quantum particles. The quantum states allowed for this individual motion are the one-particle states $|j>$ that satisfy the eigenvalue problem

$$H|j>=\epsilon_j|j>, \tag{199}$$

where ϵ_j are the eigenvalues. Since identical particles are indistiguishable in quantum mechanics, each quantum state $|\xi>$ for the complete system is completely specified when the number of particles occupying each one-particle state is specified —i.e., the quantum numbers ξ of the whole system are determined by the set of occupation numbers $n_1, n_2, n_3, ...$ The total energy of each quantum state $|\xi>$ is, therefore,

$$E_\xi = \sum_j n_j \epsilon_j. \tag{200}$$

Furthermore, since the quantum particles are indistinguishable,

$$\hat{P}_{jk}|\xi(1,2,3,...,j,j+1,...,k,k+1,...)>=\pm|\xi(1,2,3,...,k,j+1,...,j,k+1,...)>, \tag{201}$$

where $\hat{P}_{jk}$ is the operator that permutes particles j and k. The plus sign, in Eq. (201), corresponds to a system of *bosons* (i.e., integer spin particles) and the minus sign corresponds to a system of *fermions* (i.e., half-integer spin particles). The *Pauli Exclusion Principle* is a consequence of the symmetry requirement introduced by Eq. (201). Such principle establishes that in a system of fermions with the same spin there cannot be two particles occupying the same spatial state and, therefore, $n_j = 0, 1$. In a system of bosons, however, $n_j = 0, 1, 2, ...$ —i.e., there can be an arbitrary large number of particles in each state j.
The grand canonical partition function for a system of indistigushable particles is defined, according to Eq. (104),

$$\Xi = \sum_\xi exp(-\beta E_\xi + \beta\mu n_\xi), \tag{202}$$

or in terms of occupation numbers n_j,

$$\Xi = \sum_{n_1,n_2,...} exp(-\beta \sum_k \epsilon_k n_k + \beta\mu \sum_k n_k). \tag{203}$$

The grand canonical partition function for a system of fermions is

$$\Xi = \prod_k \sum_{n_k=0}^{1} exp(-\beta\epsilon_k n_k + \beta\mu n_k) = \prod_k (1+e^{\beta(\mu-\epsilon_k)}), \tag{204}$$

due to the Pauli Exclusion Principle and Eq. (203). Therefore, the average occupation number

$$\overline{n_k} = \Xi^{-1} \sum_{n_1,n_2,...} n_k e^{-\beta\sum_k \epsilon_k n_k + \beta\mu\sum_k n_k} = \partial \ln\Xi / \partial(-\beta\epsilon_j), \tag{205}$$

is given by the following expression

$$\overline{n_k} = \frac{e^{\beta(\mu-\epsilon_k)}}{e^{\beta(\mu-\epsilon_k)}+1} = \frac{1}{1+e^{\beta(\epsilon_k-\mu)}}, \tag{206}$$

which is the *Fermi-Dirac distribution*.

Analogously, the grand canonical partition function for a system of bosons is

$$\Xi = \prod_k \sum_{n_k=0}^{\infty} e^{-\beta(\epsilon_k-\mu)n_k} = \prod_k \frac{1}{1-e^{\beta(\mu-\epsilon_k)}}. \tag{207}$$

Therefore, the average occupation number is given by the following expression

$$\overline{n_k} = \frac{1}{e^{\beta(\epsilon_k-\mu)}-1}, \tag{208}$$

which is the *Bose-Einstein distribution*.

22.1 Chemical Potential

The chemical potential μ, introduced Eq. (105), defines the increase in free energy of the system upon addition of an infinitesimal amount of particles since

$$\begin{aligned} G = \mu N &= E - TS + Tk_B \ln\Xi, \\ &= H - TS, \\ &= E + PV - TS, \end{aligned} \tag{209}$$

so a change in extensive property gives

$$\begin{aligned} dG)_{T,P} &= \mu dN, \\ &= dE + PdV - TdS, \end{aligned} \tag{210}$$

Alternatively, we can define the chemical potential as the increase in internal energy due an increase in the average number of particles since according to Eq. (210), $\mu = \frac{\partial G}{\partial N}\Big)_{T,P} = \frac{\partial E}{\partial N}\Big)_{V,S}$.

Chemical Potential of Photons: The chemical potential of photons is zero since photons do not interact with each other and the number of photons (thermal radiation) is automatically adjusted by the walls of the container that emit and absorb photons until thermal equilibrium is reached. Noting that, $\ln\Xi = \ln Z + \beta\mu N$, and $\mu = 0$, we obtain that for photons $\Xi = Z$. Therefore, the partition function of a gas of photons is

$$Z = \prod_k \sum_{n_k=0}^{\infty} e^{-\beta\epsilon_k n_k} = \prod_k \frac{1}{1-e^{-\beta\epsilon_k}}. \tag{211}$$

and the average number of photons with energy ϵ_j is

$$\overline{n_j} = \frac{1}{e^{\beta\epsilon_j}-1}, \tag{212}$$

where $\epsilon_j = \hbar k_j c$, with $k_j = |\mathbf{k}_j|$ defined by the components of momenta $\mathbf{k}_j = (k_x^{(j)}, k_y^{(j)}, k_z^{(j)})$ of the electromagnetic standing waves in the container, with $k_{x,y,z}^{(j)} = n_{x,y,z}^{(j)}\pi/L_{x,y,z}$ defined by the dimensions of the container L_x, L_y and L_x, with quantum numbers $n_{x,y,z} = 1, 2, \cdots$.

The number of states with energy smaller that ϵ_j is given by the octant volume of the Hilbert space with positive quantum numbers within the sphere of radius n_j, which is $\frac{1}{8}\frac{4}{3}\pi n_j^3 = \frac{V}{6\pi^2}k_j^3$. Therefore, the number of states with momenta between k_j and $k_j + dk_j$ is $\frac{V}{2\pi^2} k_j^2 dk_j$. Considering 2 states of polarization per mode, we obtain:

$$
\begin{aligned}
E &= \frac{V}{\hbar^3 c^3 \pi^2} \int_0^\infty d\epsilon \frac{\epsilon^3}{e^{\beta\epsilon} - 1}, \\
&= \frac{V k_B^4 T^4}{\hbar^3 c^3 \pi^2} \int_0^\infty dx \frac{x^3}{e^x - 1}.
\end{aligned}
\tag{213}
$$

According to Eq. (275), $\frac{3}{x^3} \int_0^x d\gamma \frac{\gamma^3}{e^\gamma - 1} = \frac{\pi}{5x^3}$ when $x \gg 1$. Therefore, $\int_0^\infty dx \frac{x^3}{e^x-1} = \frac{\pi}{15}$ giving

$$
E = \frac{V k_B^4 T^4}{15 \hbar^3 c^3 \pi}. \tag{214}
$$

This equation can be used to obtain the Stefan-Boltzmann law, giving the energy radiated per unit time through a hole of unit area in the wall of the container:

$$
R = \sigma T^4, \tag{215}
$$

where σ is the Stefan-Boltzmann constant.

Furthermore, Eq. (213) allows us to obtain the energy per unit volume with frequency between w and $w + dw$, with $\epsilon = \hbar w = h\nu = hc/\lambda$, as follows:

$$
\rho(w, \beta) dw = \frac{\hbar}{c^3 \pi^2} \frac{w^3}{e^{\beta \hbar w} - 1} dw, \tag{216}
$$

which is the famous Planck's blackbody distribution law that agrees with the experimental spectral radiance and solves the paradox of the 'ultraviolet catastrophe' introduced by the Rayleigh-Jeans law of classical physics (incorrect prediction of spectral radiance proportional to λ^{-4} and T that significantly deviates from the experimental spectra in the ultraviolet range).
Exercise: Compute the derivative of the energy density with respect to λ and make it equal to zero to obtain that $\lambda_{max}(T)$ (the wavelength of maximum radiance) is inversely proportional to temperature –*i.e.*, Wien displacement law. Considering that the λ_{max} of the solar spectrum is close to 500 nm, estimate the temperature of the surface of the sun (you would get $T \approx 5270K$, so you can classify the sun as a yellow star. Analogously, we can estimate the temperature of other stars from their spectrum. The same law quantitatively describes the wavelength of light radiated by a nail as it gets red hot and then colorless as in gets hotter.

23 Classical limit of Quantum Statistical Distributions

According to Eqs. (206) and (208), the average number n_j of noninteracting quantum particles in state j is

$$
\overline{n_j} = \frac{1}{e^{\beta(\epsilon_j - \mu)} \mp 1}, \tag{217}
$$

where the minus sign corresponds to the Bose-Einstein distribution and the plus sign corresponds to the Fermi-Dirac distribution.

In the limit when $\overline{n_j} \to 0$ the system is very "dilute", since there are much more energetically accessible states than particles. This limit is achieved for all states of energy ϵ_j when

$$
e^{\beta(\epsilon_j - \mu)} >> 1. \tag{218}
$$

Therefore, in the limit when $\overline{n_j} \to 0$,

$$\overline{n_j} \approx e^{-\beta(\epsilon_j - \mu)}, \tag{219}$$

and the average number of particles is

$$\overline{N} = \sum_j \overline{n_j} \approx \sum_j e^{-\beta(\epsilon_j - \mu)}, \tag{220}$$

so that

$$\beta\mu = \ln\bar{N} - \ln\sum_j e^{-\beta\epsilon_j}. \tag{221}$$

Moreover, according to Eqs. (219) and (220),

$$\frac{\overline{n_j}}{\overline{N}} = \frac{e^{-\beta\epsilon_j}}{\sum_j e^{-\beta\epsilon_j}}, \tag{222}$$

which is the classical Boltzmann distribution. Therefore, *in the limit when $\overline{n_j} \to 0$ both the Fermi-Dirac and the Bose-Einstein distributions converge to the classical Boltzmann distribution.*

Furthermore, according to Eqs. (204) and (207),

$$\ln\Xi = \mp\sum_j \ln(1 \mp e^{-\beta(\epsilon_j - \mu)}). \tag{223}$$

and in the limit when $e^{\beta(\epsilon_j - \mu)} >> 1$,

$$\ln\Xi \approx \sum_j e^{-\beta(\epsilon_j - \mu)}, \tag{224}$$

since

$$\lim_{x\to 0} \ln(1 + x) = x. \tag{225}$$

Therefore, according to Eqs. (220) and (224),

$$\ln\Xi \approx \overline{N}, \tag{226}$$

and according to Eqs. (226) and (109),

$$\overline{N} \approx lnZ + \beta\mu\overline{N}. \tag{227}$$

Substituting Eq. (221) into Eq. (226), we obtain

$$\overline{N} \approx lnZ + \overline{N}\ln\overline{N} - \overline{N}\ln\sum_j e^{-\beta\epsilon_j}, \tag{228}$$

and according to the Stirling formula, introduced by Eq. (71),

$$Z = \frac{\left(\sum_j e^{-\beta\epsilon_j}\right)^{\overline{N}}}{\overline{N}!}, \tag{229}$$

where the $1/N!$ factor, in Eq. (229), indicates that quantum particles remain indistinguishable even in the classical limit!

24 Gibbs Paradox

The factor $1/N!$, in Eq. (229), is essential to resolve the paradox formulated by Gibbs before the discovery of Quantum Mechanics, stating that the entropy of a gas of N particles in a volume V at temperature T decreases by a factor of N k ln(2) when the system is divided in 2 by an internal membrane, although the thermodynamic quantities that define the state (i.e., N, T, V) remain constant.

To explain the paradox and its solution, consider a monatomic gas of N non-interacting molecules at temperature T and volume V, with internal energy $E = \frac{3}{2}NkT$ and entropy $S = \frac{E}{T} + kln(Z)$.

If we ignore the factor $1/N!$, in Eq. (229), we obtain:

$$\begin{aligned} Z &= \left(\sum_j e^{-\beta\epsilon_j}\right)^N \\ &= \left(\int d\mathbf{r} \int d\mathbf{p} e^{-\beta\frac{\mathbf{p}^2}{2m}}\right)^N \\ &= V^N \left(\frac{\pi 2m}{\beta}\right)^{3N/2} \end{aligned} \tag{230}$$

Therefore, $ln(Z) = Nln(V) + \frac{3}{2}Nln\left(\frac{\pi 2m}{\beta}\right)$ and the entropy is:

$$S = \frac{3}{2}Nk + Nkln(V) + \frac{3}{2}Nkln\left(\frac{\pi 2m}{\beta}\right). \tag{231}$$

Equation (231) leads to the Gibbs paradox when we consider twice the entropy $S_{1/2}$ of half of the system (*e.g.*, with half the number of particles N/2 and volume V/2):

$$2 \times S_{1/2} = \frac{3}{2}Nk + Nkln(V) + \frac{3Nk}{2}ln\left(\frac{\pi 2m}{\beta}\right) - Nkln(2), \tag{232}$$

since $2 \times S_{1/2} \neq S$ and the sum of the entropies of the halves should be equal to the total entropy.

In contrast, correcting Eq. (231) with the factor $1/N!$, using the Sterling approximation $ln(N!) \approx Nln(N) - N$, we obtain:

$$S = \frac{3}{2}Nk + Nkln(V) + \frac{3Nk}{2}ln\left(\frac{\pi 2m}{\beta}\right) - Nkln(N) + Nk \tag{233}$$

and

$$2 \times S_{1/2} = \frac{3}{2}Nk + Nkln(V) + \frac{3Nk}{2}ln\left(\frac{\pi 2m}{\beta}\right) - Nkln(2) - Nkln(N) + Nk + Nkln(2). \tag{234}$$

showing that $S = 2 \times S_{1/2}$.

The origin of the paradox is thus the misleading assumption that particles are distinguishable. However, that would require a method to determine whether a specific molecule is on one side of the dividing membrane or the other.

25 Example 1: Ideal Gas of Structureless Quantum Particles

An ideal gas of N non-interacting structureless paticles of mass m is described by the N-particle Hamiltonian

$$\hat{H} = \sum_{j=1}^{N} \hat{h}_j, \tag{235}$$

where $\hat{h}_j$ is the one-particle Hamiltonian

$$\hat{h}_j = -\frac{\hbar^2}{2m}\nabla^2_{\mathbf{R}_j} = -\frac{\hbar^2}{2m}\left(\frac{\partial^2}{\partial x_j^2} + \frac{\partial^2}{\partial y_j^2} + \frac{\partial^2}{\partial z_j^2}\right), \tag{236}$$

with $\mathbf{R}_j = (x_j, y_j, z_j)$. The eigenstates of $\hat{h}_j$ are the free-particle states

$$\phi_j(x,y,z) = Ae^{\mathbf{k}_j\cdot\mathbf{R}}, \tag{237}$$

where $\mathbf{k}_j = (kx_j, ky_j, kz_j)$, and A is a normalization constant determined by the volume of the box that contains the gas. The one-particle eigenstates satisfy the eigenvalue problem

$$\hat{h}_j|\phi_j >= \epsilon_j|\phi_j >, \tag{238}$$

with $\epsilon_j = (\hbar\mathbf{k}_j)^2/(2m_j)$. Note that since the volume of the box is V=Lx$\times$Ly$\times$Lz, and $|\phi_j >$ are stationary states, then Kx Lx = nx π, Ky Ly = ny π and Kz Lz = nz π, with nx,ny,nz=1,2,...
Therefore,

$$\sum_{nx,ny,nz} = \sum_{Kx,Ky,Kz}\frac{V}{\pi^3}\cdots$$

and

$$Z = \frac{1}{N!}\left(\sum_j e^{-\beta\epsilon_j}\right)^N = \frac{1}{N!\pi^{3N}}\left(V\int_0^\infty dK_x\int_0^\infty dK_y\int_0^\infty dK_z e^{-\frac{\beta\hbar^2}{2m}(K_x^2+K_y^2+K_z^2)}\right)^N. \tag{239}$$

Computing the Gaussian integrals analytically, we obtain

$$Z = \frac{1}{N!\pi^{3N}}\left(\frac{V}{2^3}\left(\frac{\pi 2m}{\beta\hbar^2}\right)^{3/2}\right)^N = \frac{V^N}{2^{3N}N!\pi^{3N}}\left(\frac{2\pi m}{\beta\hbar^2}\right)^{3N/2}, \tag{240}$$

since $\int_0^\infty e^{-\alpha x^2}dx = \frac{1}{2}\sqrt{\frac{\pi}{\alpha}}$. Therefore,

$$Z = \frac{V^N}{N!h^{3N}}\left(\frac{2\pi m}{\beta}\right)^{3N/2} \Rightarrow E = -\left.\frac{\partial \ln Z}{\partial\beta}\right)_{V,N} = \frac{3}{2}NkT. \tag{241}$$

In addition, according to Eq. (109),

$$\frac{\partial \ln}{\partial V}\Xi = \frac{\partial \ln}{\partial V}Z, \tag{242}$$

since $\beta\mu N$ depends only on T. Considering that $G = \mu N = H - TS = E + pV - TS = \beta^{-1}\ln\Xi - \beta^{-1}\ln Z$, with $TS = E + \beta^{-1}\ln Z$, or $E + pV - TS = \beta^{-1}\ln\Xi - TS + E$, we obtain $pV = \beta^{-1}\ln\Xi$, so we can define p as follows:

$$\beta p \equiv \left.\frac{\partial\ln\Xi}{\partial V}\right)_{T,N}. \tag{243}$$

and according to Eq. (242),

$$\beta p \equiv \left.\frac{\partial\ln Z}{\partial V}\right)_{T,N}. \tag{244}$$

Therefore, we obtain

$$\beta p = \frac{N}{V} \Rightarrow \boxed{pV = NkT}, \tag{245}$$

which is the equation of state for an ideal gas of structureless particles.

26 Example 2: Dilute Gas of Diatomic Molecules

In the previous example, we showed that the state of a structureless particle is completely defined by the vector $\mathbf{K} = (Kx, Ky, Kz)$ that specifies the momentum $\mathbf{K}\hbar$ of the center of mass. Specifying the state of a molecule, however, requires the specification of its internal state besides specifying the translation of the molecule as a whole, since the molecule can vibrate, rotate, or undergo electronic excitations while translating as specified by the vector $\mathbf{K}$.
Contrary to structureless particles, molecules are described by the Hamiltonian

$$\hat{H} = -\frac{\hbar^2}{2m}\nabla^2_{\mathbf{R}} + \hat{H}_{el}(\mathbf{R}, \mathbf{r}), \tag{246}$$

where, $\mathbf{R}$ and $\mathbf{r}$ are the nuclear and electronic coordinates.
A simple expression for the cononical partition function of an ideal gas of diatomic molecules in the ground electronic state can be obtained by factorizing the total energy into translational, rotational and vibrational modes (i.e., assuming that these modes are uncoupled from each other) and then modeling vibrations according to the harmonic approximation and rotations according to the rigid rotor approximation. The resulting partition function is

$$Z = \frac{(\sum_j e^{-\beta E_j})^N}{N!}, \tag{247}$$

Here, $E_j = E_{rot}(J(j)) + E_{transl}(n(i)) + E_{vib}(\nu(j))$, where $J(j)$ specifies the rotational quantum number, $\nu(j)$ the vibrational quantum number and $n(j)$ the translational modes of particle j. Therefore,

$$Z = \frac{(q_{transl} * q_{int})^N}{N!}, \text{ where } q_{int} = q_{rot} * q_{vib}, \tag{248}$$

with

$$q_{vib} = \sum_{\nu=0}^{\infty} e^{-\beta\hbar\omega_0(1/2+\nu)} = \frac{e^{-\beta\hbar\omega_0/2}}{1 - e^{-\beta\hbar\omega_0/2}}, \tag{249}$$

$$q_{transl} = \frac{V}{h^3}\left(\frac{2\pi m}{\beta}\right)^{3/2}, \tag{250}$$

and

$$\begin{aligned} q_{rot} &= \sum_{J=0}^{\infty}(2J+1)e^{-\beta\frac{\hbar^2}{2I_0}J(J+1)}, \\ &= \sum_{J=0}^{\infty} f(J), \\ &= \int_0^{\infty} f(J)dJ + \frac{1}{2}[f(\infty) + f(0)] + \sum_{j=1}^{\infty}(-1)^j \frac{B_j}{(2j)!}[f^{2j-1}(0) - f^{2j-1}(\infty)], \end{aligned} \tag{251}$$

where $f(J) = (2J+1)e^{-\beta\frac{\hbar^2}{2I_0}J(J+1)}$. In the third line, we have introduced the Euler-MacLaurin formula, where $B_1 = 1/6, B_2 = 1/30, B_3 = 1/42, \cdots$. Note that $f(0) = 1$, $f(\infty) = 0$ and $\int_0^{\infty} dJ f(J) = -\frac{2I_0}{\beta\hbar^2}\int_0^{\infty} dJ \frac{d}{dJ} e^{-\beta\frac{\hbar^2}{2I_0}J(J+1)} = \frac{2I_0}{\beta\hbar^2}$. Also, note that for simplicity we have ignored the internal structure of nuclei and the degeneracy factor associated with the permutation of indistinguishable nuclei.

27 Example 3: Phonons in a Solid Lattice

Having computed the partition function of a harmonic oscillator, we now compute the partition function of the normal modes of a solid at low temperature. According to the harmonic approximation, the Hamiltonian of the system is

$$\hat{H} = \sum_{\alpha=1}^{DN} \hat{h}_\alpha, \tag{252}$$

where DN is the number of normal modes, with D the dimensionality of the lattice and $\hat{h}_\alpha$ is the Hamiltonian of a harmonic oscillator with a frequency ω_α and eigenvalues

$$E_n(\alpha) = \hbar\omega_\alpha(\frac{1}{2} + n_\alpha), \tag{253}$$

with $n_\alpha = 1, 2...$
An arbitrary vibrational state ξ of the lattice can be specified by the DN normal mode frequencies ω_α and vibrational quantum numbers n_α. The energy of such state is

$$E_\xi = \sum_{\alpha=1}^{DN} [n_\alpha \hbar\omega_\alpha + \frac{\hbar}{2}\omega_\alpha]. \tag{254}$$

The canonical partition function for the lattice is

$$Z(\beta, N) = \sum_{n_1}\sum_{n_2}\sum_{n_3} ...\text{exp}\left(-\beta \sum_{\alpha=1}^{DN} n_\alpha \hbar\omega_\alpha + \frac{\hbar}{2}\omega_\alpha \right), \tag{255}$$

which according to Eq.(174) becomes,

$$Z(\beta, N) = \prod_{\alpha}^{DN} \frac{e^{-\beta\frac{\hbar\omega_\alpha}{2}}}{1 - e^{-\beta\hbar\omega_\alpha}} = \prod_{\alpha=1}^{DN} (e^{\beta\frac{\hbar\omega_\alpha}{2}} - e^{-\beta\frac{\hbar\omega_\alpha}{2}})^{-1}, \tag{256}$$

and

$$\text{ln}Z = -\sum_{\alpha=1}^{DN} \text{ln}(e^{\beta\frac{\hbar\omega_\alpha}{2}} - e^{-\beta\frac{\hbar\omega_\alpha}{2}}), \tag{257}$$

or in the continuous representation,

$$\text{lnZ} = -\int_0^\infty d\omega g(\omega) \text{ln}(e^{\beta\frac{\hbar\omega}{2}} - e^{-\beta\frac{\hbar\omega}{2}}) \tag{258}$$

where $g(\omega)$ is the density of states —i.e., the number of vibrational states with frequencies between ω and $\omega + d\omega$.

27.1 Einstein Model

The Einstein model assumes that all vibrational modes in a solid lattice have the same frequency ω_E and, therefore,

$$g(\omega) = DN\delta(\omega - \omega_E). \tag{259}$$

Substituting Eq. (259) into Eq. (258) we obtain

$$\text{lnZ} = -ND\text{ln}\left(e^{+\beta\hbar\omega_E/2} - e^{-\beta\hbar\omega_E/2} \right). \tag{260}$$

The average internal energy of the lattice can be computed from Eq. (260) as follows,

$$E = \frac{\partial \ln Z}{\partial(-\beta)} = +DN\frac{\hbar\omega_E}{2}\coth(\beta\hbar\omega_E/2). \tag{261}$$

Note that in the high temperature limit,

$$\lim_{\beta\to 0} E = \frac{DN}{\beta} = DNkT. \tag{262}$$

The heat capacity at constant volume C_v can also be obtained from Eq. (260) as follows,

$$C_v = \left.\frac{\partial E}{\partial T}\right)_v = \left(-\frac{ND}{kT^2}\right)\left(\frac{\hbar}{2}\omega_E\right)^2 \frac{\left(\left(e^{\beta\hbar\omega_E/2} - e^{-\beta\hbar\omega_E/2}\right)^2 - \left(e^{\beta\hbar\omega_E/2} + e^{-\beta\hbar\omega_E/2}\right)^2\right)}{\left(e^{\beta\hbar\omega_E/2} - e^{-\beta\hbar\omega_E/2}\right)^2}. \tag{263}$$

The expression introduced by Eq. (263) can be simplified to obtain

$$C_v = \left(\frac{\theta}{T}\right)^2 \frac{NDk}{\left(e^{-\frac{\theta}{2T}}(e^{\frac{\theta}{T}} - 1)\right)^2} = NDk\left(\frac{\theta}{T}\right)^2 \frac{e^{\frac{\theta}{T}}}{\left(e^{\frac{\theta}{T}} - 1\right)^2}, \tag{264}$$

with $\theta \equiv \frac{\hbar\omega_E}{2k}$.

Limiting cases

(i) At *high temperature*, $\theta << T$ and $e^{\frac{\theta}{T}} \approx 1 + \frac{\theta}{T}$. Therefore,

$$C_v = NDk\left(\frac{\theta}{T}\right)^2 \frac{1 + \frac{\theta}{T} + ...}{(1 + \frac{\theta}{T} - 1 + ...)^2} = NDk. \tag{265}$$

Eq. (265) is the Dulong-Petit's law —i.e., the experimental limit for C_v at high temperature.

(ii). At *low temperature*, $\theta >> T$. Therefore,

$$C_v = NDk\left(\frac{\theta}{T}\right)^2 e^{-\frac{\theta}{T}}. \tag{266}$$

It is important to note that Eq. (266) does not predict the correct limit for C_v at low temperature, since C_v should be proportional to T^3 when $T \to 0$.

27.2 Debye Model

Debye approximated the normal vibrations with the elastic vibrations of an isotropic continuous body where the number of vibrational modes with frequencies between ω and $\omega + d\omega$ is

$$g(\omega) = \begin{cases} \xi\omega^2, & \text{when } \omega < \omega_0, \\ 0, & \text{otherwise,} \end{cases} \tag{267}$$

where $\xi \equiv 3V/(2\pi^2 c^3)$ and

$$\int_0^{\omega_0} d\omega g(\omega) = 3N = \frac{V\omega_0^3}{2\pi^2 c^3}. \tag{268}$$

Therefore,

$$\xi = 9N2\pi^2 c^3/(\omega_0^3 2\pi^2 c^3). \tag{269}$$

According to Eqs.(268) and (258),

$$\text{lnZ} = -\int_0^{\omega_0} d\omega\xi\omega^2 \text{ln}\left(e^{\beta\hbar\omega/2} - e^{-\beta\hbar\omega/2}\right). \tag{270}$$

Therefore,

$$\text{lnZ} = -\int_0^{\omega_0} d\omega\xi\omega^2 \text{ln}e^{\beta\hbar\omega/2} - \int_0^{\omega_0} d\omega\xi\omega^2 \text{ln}(1 - e^{-\beta\hbar\omega}), \tag{271}$$

and

$$\text{lnZ} = -\frac{\beta\hbar}{2}\xi\frac{\omega_0^4}{4} - \int_0^{\omega_0} d\omega\xi\omega^2 \text{ln}(1 - e^{-\beta\hbar\omega}). \tag{272}$$

The internal energy E is computed from Eqs. (272) and (269) as follows,

$$E = \frac{\partial \text{lnZ}}{\partial(-\beta)} = \frac{\hbar}{2}\frac{9N\omega_0^4}{\omega_0^3 4} + \int_0^{\omega_0} d\omega\xi\omega^3\hbar\frac{e^{-\beta\hbar\omega}}{1 - e^{-\beta\hbar\omega}}, \tag{273}$$

and introducing the change of variables $\gamma \equiv \beta\hbar\omega$,

$$E = \frac{\hbar}{2}\frac{9N\omega_0}{4} + \frac{1}{\beta\hbar}\int_0^{\beta\hbar\omega_0} d\gamma\xi\frac{\gamma^3}{(\beta\hbar)^3}\hbar\frac{1}{(e^\gamma - 1)}. \tag{274}$$

Considering that

$$f(x) = \frac{3}{x^3}\int_0^x d\gamma\frac{\gamma^3}{e^\gamma - 1} = \begin{cases} 1 - \frac{3}{8}x + ... & x << 1 \\ \frac{\pi}{5x^3} + ... & x >> 1, \end{cases} \tag{275}$$

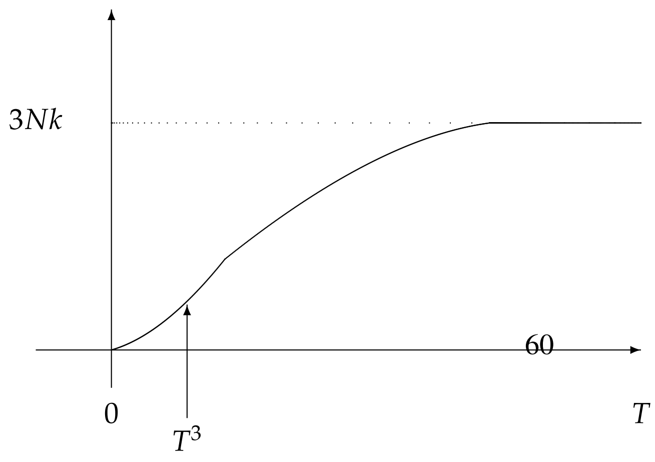
we obtain, according to Eqs. (274) and (275),

$$\tilde{E} = E - \frac{\hbar}{2}\frac{9N\omega_0}{4} = \frac{\omega_0^3 3N}{\beta\omega_0^3} \times \begin{cases} 1 - ... = 3NkT & \text{when } T \to \infty \\ \frac{\pi}{5(\beta\hbar\omega_0)^3} = \frac{(kT)^4\pi 3N}{5(\hbar\omega_0)^3} & \text{when } T \to 0. \end{cases} \tag{276}$$

Therefore the Debye model predicts the following limits for the heat capacity of a solid lattice,

$$C_v = \begin{cases} 3Nk & \text{when } T \to \infty \\ 3Nk^4\frac{\pi 4}{5(\hbar\omega_0)^3}T^3 & \text{when } T \to 0. \end{cases} \tag{277}$$

which are the correct high and low temperature limits, represented by the following diagram:



28 Example 4: Electrons in Metals

The goal of this section is to show that even at room temperature, the conducting electrons in metals can be modeled as an ideal gas of fermions contained in a box of volume $V = L^3$, where L defines the dimensions of the piece of metal. Such a goal is accomplished by comparing the kinetic energy of conducting electrons, modeled as an ideal gas of fermions, with typical energy fluctuations due to thermal motion.
The average number of electrons occupying the j-th energy state is

$$\overline{n_j} = \frac{1}{e^{\beta(\epsilon_j - \mu)} + 1}, \tag{278}$$

where

$$\epsilon_j = \frac{\hbar^2 K_j^2}{2m}, \tag{279}$$

and

$$K_j = (n_x(j), n_y(j), n_z(j))\pi/L \tag{280}$$

with $n_x(j), n_y(j), n_z(j) = 1, 2, ...$ Therefore, the average number of electrons is

$$\overline{N} = \sum_j \overline{n_j}, \tag{281}$$

$$\overline{N} = 2\sum_{n_x}\sum_{n_y}\sum_{n_z} \frac{1}{e^{\beta(\epsilon(n_x,n_y,n_z)-\mu)} + 1}, \tag{282}$$

or,

$$\overline{N} = 2\int_{-\infty}^{\infty} dk_x \int_{-\infty}^{\infty} dk_y \int_{-\infty}^{\infty} dk_z \left(\frac{L}{\pi}\right)^3 \frac{1}{e^{\beta(\frac{\hbar^2 K^2}{2m} - \mu)} + 1}. \tag{283}$$

In particular, at $T = 0$,

$$\frac{1}{1 + e^{\beta(\frac{\hbar^2 K^2}{2m} - \mu)}} = \begin{cases} 1, & \frac{\hbar^2 K^2}{2m} < \mu, \\ 0, & \frac{\hbar^2 K^2}{2m} > \mu, \end{cases} \tag{284}$$

therefore,

$$\overline{N} = 8\pi \int_0^{K_f} dK K^2 \left(\frac{L}{2\pi}\right)^3 = 8\pi \left(\frac{L}{2\pi}\right)^3 \frac{K_f^3}{3} = \frac{2V}{(2\pi)^3}\frac{4}{3}\pi K_f^3, \tag{285}$$

where K_f is the Fermi momentum defined as follows

$$\frac{\hbar^2 K_f^2}{2m} = \mu. \tag{286}$$

The value of K_f for a specific metal can be found, according to Eq. (285) and using the values of the density and atomic weight of the corresponding metal, assuming that each atom in the metal donates an electron to the conducting electron gas. Such value of K_f, can be used to compute the chemical potential according to Eq. (286). The calculation for Cu, with a density of 9 g/cm^3 and atomic weight of 63.5 g/mol gives

$$\mu/k \approx 80,000K, \tag{287}$$

which indicates that even at room temperature the ideal gas approximation is accurate.
Thermal Energy and Heat Capacity

The remaining of this section proves that at low temperature T the heat capacity C_v of electrons in metals is proportional T.
The thermal energy E of electrons in metals is

$$E = \sum_j 2\overline{n_j}\epsilon_j, \tag{288}$$

where index j specifies a one-electron quantum state with energy ϵ_j and $\overline{n_j}$ is the average number of electrons with one kind of spin in such state. The factor 2, introduced by Eq. (288) counts for the spin degeneracy. Substituting $\overline{n_j}$ according to Eq. (278) and changing the sum over j by a sum over energy levels we obtain

$$E = 2\int_0^\infty d\epsilon \frac{\rho(\epsilon)\epsilon}{e^{\beta(\epsilon-\mu)}+1}, \tag{289}$$

where $\rho(\epsilon)$ is the degeneracy of the energy level.
Eq. (289) can be integrated by parts, according to

$$\int_a^b u d\phi = u * \phi\Big|_a^b - \int_a^b \phi du, \tag{290}$$

defining $d\phi = \epsilon\rho(\epsilon)d\epsilon$ and $u(\epsilon) = 1/(exp(\beta(\epsilon-\mu))+1)$. Note that according to this choice of variables $\phi(\epsilon) = \int_0^\epsilon d\bar{\epsilon}\rho(\bar{\epsilon})\bar{\epsilon}$. We obtain

$$E = \lim_{\epsilon\to\infty} \frac{2\int_0^\epsilon d\epsilon'\rho(\epsilon')\epsilon'}{e^{\beta(\epsilon-\mu)}+1} - \lim_{\epsilon\to 0} \frac{2\int_0^\epsilon d\epsilon'\rho(\epsilon')\epsilon'}{e^{\beta(\epsilon-\mu)}+1} - 2\int_0^\infty d\epsilon[\int_0^\epsilon d\epsilon'\rho(\epsilon')\epsilon']\frac{(-e^{\beta(\epsilon-\mu)}\beta)}{(e^{\beta(\epsilon-\mu)}+1)^2}. \tag{291}$$

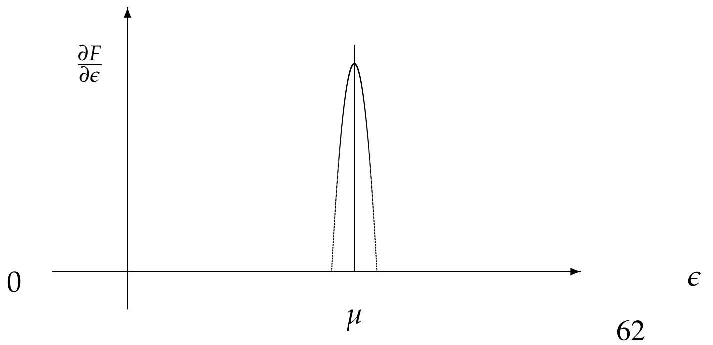
Note that the first term, introduced by Eq. (291) is equal to 0 since in the limit when $\epsilon \to \infty$ the denominator becomes extremely large. The second term introduced by Eq. (291) is also equal 0 since in the limit when $\epsilon \to 0$ the numerator is equal to 0. Therefore, introducing the definition

$$F(\epsilon) \equiv \frac{1}{e^{\beta(\epsilon-\mu)}+1}, \tag{292}$$

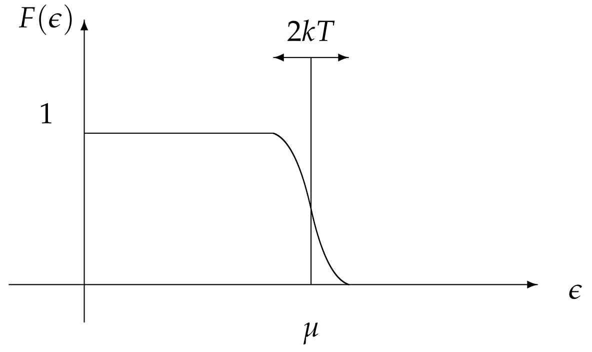
we obtain that Eq. (291) can be rewritten as follows,

$$E = -2\int_0^\infty d\epsilon\phi(\epsilon)\frac{\partial F}{\partial \epsilon}. \tag{293}$$

At this point, it is important to note that $\partial F/\partial\epsilon$ is a function peaked at $\epsilon = \mu$, as represented by the following diagram,



since $F(\epsilon)$ is a step function represented by the following diagram,



Therefore we can approximate $\phi(\epsilon)$, in Eq. (293), by its expansion around $\epsilon = \mu$,

$$\phi(\epsilon) = \phi(\mu + x) \approx \sum_{0}^{\infty} \left.\frac{\partial^n \phi}{\partial \epsilon^n}\right|_{\epsilon=\mu} \frac{(\epsilon - \mu)^n}{n!}, \tag{294}$$

where $x \equiv \epsilon - \mu$. Substituting Eq. (294) into Eq. (293) for a given spin, we obtain,

$$E = -\int_0^\infty d\epsilon \sum_{n=0}^{\infty} \left.\frac{\partial^n \phi}{\partial \epsilon^n}\right|_{\epsilon=\mu} \frac{(\epsilon - \mu)^n}{n!} \frac{\partial F}{\partial \epsilon}. \tag{295}$$

Considering the first few terms of Eq. (295) we obtain,

$$\begin{aligned} E &= \phi(\mu)\left(-\int_0^\infty d\epsilon \frac{\partial F}{\partial \epsilon}\right) + \left.\frac{\partial \phi}{\partial \epsilon}\right|_{\epsilon=\mu} \int_0^\infty d\epsilon (\epsilon - \mu) \frac{e^{\beta(\epsilon-\mu)}}{e^{2\beta(\epsilon-\mu)} + 2e^{\beta(\epsilon-\mu)} + 1} \\ &+ \left.\frac{\partial^2 \phi}{\partial \epsilon^2}\right|_{\epsilon=\mu} \int_0^\infty d\epsilon \frac{(\epsilon-\mu)^2}{2} \frac{e^{\beta(\epsilon-\mu)}}{e^{2\beta(\epsilon-\mu)} + 2e^{\beta(\epsilon-\mu)} + 1} + ..., \end{aligned} \tag{296}$$

where the first term is equal to $\phi(\mu)$ because $F(\infty) = 0$ and $F(0) = 1$. To show that the second term in Eq. (296) is equal to 0 we rewrite Eq. (296) in terms of the variable $\bar{x} = \beta x$,

$$\begin{aligned} E &= \phi(\mu) + \left.\frac{\partial \phi}{\partial \epsilon}\right|_{\epsilon=\mu} \int_{-\infty}^{\infty} d\bar{x} \frac{\bar{x}}{\beta} \frac{1}{e^{\bar{x}} + e^{-\bar{x}} + 2} \\ &+ \frac{(kT)^2}{2} \left.\frac{\partial^2 \phi}{\partial \epsilon^2}\right|_{\epsilon=\mu} \int_{-\infty}^{\infty} d\bar{x} \frac{\bar{x}^2}{e^{\bar{x}} + e^{-\bar{x}} + 2} + ..., \end{aligned} \tag{297}$$

where the lower integration limit has been changed from 0 to $-\infty$ since the integrand is approximately equal to zero whenever abs$(\epsilon - \mu)$ is large. Note that the integral introduced by the second term of Eq. (297) is equal to 0 because the integrand is odd. In addition,

$$\int_{-\infty}^{\infty} d\bar{x} \frac{\bar{x}^2}{e^{\bar{x}} + e^{-\bar{x}} + 2} = \frac{\pi^2}{3}, \tag{298}$$

therefore,

$$E = \phi(\mu) + \frac{(kT)^2}{2} \frac{\partial^2 \phi}{\partial \epsilon^2}\bigg)_{\epsilon=\mu} \left(\frac{\pi^2}{3}\right) + ..., \tag{299}$$

At sufficiently low T, higher order terms in Eq. (299) are negligible. Therefore, at low T

$$C_v \propto T, \tag{300}$$

as observed in experiments.

28.1 Continuous Approximation

The goal of this subsection is to show that the error introduced by approximating Eq. (282) according to Eq. (283) is negligible when L y sufficiently large. For simplicty, we show this for a 1-dimensional problem, where

$$\sum_{K_x=\pi/L}^{\infty} \frac{1}{e^{\beta(\frac{\hbar^2}{2m}K_x^2 - \mu)} + 1} = \sum_{K_x} f(K_x) \Delta K, \tag{301}$$

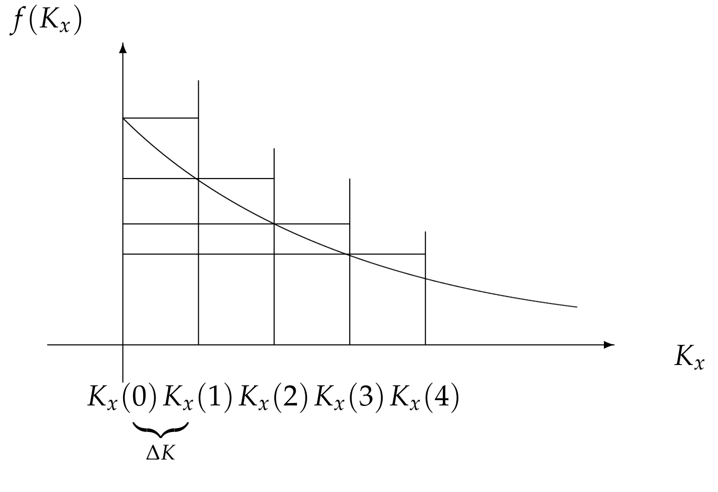
with

$$f(K_x) = \frac{1}{e^{\beta(\epsilon(K_x) - \mu)} + 1} \frac{L_x}{\pi}, \tag{302}$$

a decreasing function of K_x and

$$\Delta K = \frac{\pi}{L_x}, \tag{303}$$

Remember, that $K_x = K_x(n_x)$ is a function of the quantum number n_x, as defined by Eq. (280), where $n_x = 1, 2, ...,$ with $K_x = n_x \pi / L$.
The discrete sum, introduced by Eq. (301), can be represented by the following diagram,



The diagram shows that,

$$\sum_{n_x=1}^{\infty} f(K_x(n_x))\Delta K \leq \int_0^{\infty} dK f(K) \leq \sum_{n_x=0}^{\infty} f(K_x(n_x))\Delta K = \sum_{n_x=1}^{\infty} f(K_x(n_x))\Delta K + f(K_x(0))\Delta K, \quad (304)$$

since $f(K_x(n_x)$ is a decreasing function of K_x. So,

$$0 \leq \int_0^{\infty} dK f(K) - \sum_{n_x=1}^{\infty} f(K_x(n_x))\Delta K \leq f(K_x(0))\Delta K \leq 1. \quad (305)$$

Therefore, Eq. (305) shows that the discrete sum and the integral become equal to each other when L_x is very large since

$$0 \leq \int_0^{\infty} dK_x \frac{1}{e^{\beta(\epsilon(K_x)-\mu)}+1} - \sum_{K_x=\pi/L}^{\infty} \Delta K_x \frac{1}{e^{\beta(\epsilon(K_x(n_x))-\mu)}+1} \leq \frac{\pi}{L_x}, \quad (306)$$

so

$$\lim_{Lx\to\infty} \frac{L_x}{\pi} \int_0^{\infty} dK_x \frac{1}{e^{\beta(\epsilon(K_x)-\mu)}+1} = \sum_{n_x=1}^{\infty} \frac{1}{e^{\beta(\epsilon(K_x(n_x))-\mu)}+1}. \quad (307)$$

28.2 Joint Probabilities

The goal of this section is to show that the joint probability g_{ij} that an electron is in state i and another electron of the same spin is in state j is

$$g_{ij} = \overline{n_i n_j} - \delta_{ij}\overline{n_i}, \quad (308)$$

where $\overline{n_i}$ is the average population of state i. Note that the average $\overline{n_i n_j}$ thus provides information about *correlations* between different particles.
Consider the population of state i, n_i, in terms of the sum of occupation variables $n_i^{(\alpha)}$ over all electrons α,

$$n_i = \sum_{\alpha} n_i^{(\alpha)}, \quad (309)$$

where $n_i^{(\alpha)} = 1, 0$. Therefore, the probability that states i and j are populated is

$$\overline{n_i n_j} = \sum_{\beta}\sum_{\alpha} \overline{n_i^{(\alpha)} n_j^{(\beta)}} = \sum_{\alpha} \overline{n_i^{(\alpha)} n_j^{(\alpha)}} + \underbrace{\sum_{\alpha}\sum_{\beta\neq\alpha} \overline{n_i^{(\alpha)} n_j^{(\beta)}}}_{g_{ij}\text{by definition}}. \quad (310)$$

Note that

$$\sum_{\alpha} \overline{n_i^{(\alpha)} n_j^{(\alpha)}} = \sum_{\alpha} \overline{(n_i^{(\alpha)})^2}\delta_{ij}, \quad (311)$$

and that

$$\overline{n_i^2} = \sum_{\alpha}\sum_{\beta} \overline{n_i^{(\alpha)} n_i^{(\beta)}} = \sum_{\alpha} \overline{(n_i^{(\alpha)})^2} + \sum_{\alpha}\sum_{\beta\neq\alpha} \overline{n_i^{(\alpha)} n_i^{(\beta)}}, \quad (312)$$

where the double sum with $\beta \neq \alpha$ in Eq. (312) is equal to 0 because it corresponds to the joint probability that both particles α and β are in state i.
Substituting Eqs. (311) and (312) into Eq. (310), we obtain

$$\overline{n_i n_j} = \overline{n_i^2}\delta_{ij} + g_{ij}. \quad (313)$$

Eq. (313) is identical to Eq. (308) because $\overline{n_i^2} = \overline{n_i}$ when $n_i = 0, 1$.
Finally, note that according to Eq. (204),

$$
\begin{aligned}
\overline{n_j n_i} &= \frac{1}{\Xi} \frac{\partial^2 \Xi}{\partial(-\beta\epsilon_i)\partial(-\beta\epsilon_j)}\bigg|_{V,T} \\
&= \frac{\partial^2 \ln\Xi}{\partial(-\beta\epsilon_j)\partial(-\beta\epsilon_i)} + \frac{\partial \ln\Xi}{\partial(-\beta\epsilon_j)} \frac{\partial \ln\Xi}{\partial(-\beta\epsilon_i)} \\
&= \left(\frac{\partial}{\partial(-\beta\epsilon_j)}\bigg|_{V,T} + \overline{n_j} \right) \overline{n_i}, \\
&= \left(\delta_{ij} \frac{\partial}{\partial(-\beta\epsilon_i)}\bigg|_{V,T} + \overline{n_j} \right) \overline{n_i},
\end{aligned}
\tag{314}
$$

Therefore,

$$
g_{ij} = \left(\delta_{ij} \frac{\partial}{\partial(-\beta\epsilon_i)}\bigg|_{V,T} + \overline{n_j} - \delta_{ij} \right) \overline{n_i}. \tag{315}
$$

29 Chemical Equilibrium

The goal of this section is to derive the *law of mass action* and to obtain an expresion of the *equilibrium constant* for a chemical reaction in the gas phase in terms of the canonical partition function of the molecular constituents. To achieve these goals, we first obtain an expression of the chemical potential for the constituent molecules in terms of their canonical partition functions and then we derive the law of mass action by using the *minimum energy principle*. Finally, we combine both results and we obtain an expression of the equilibrium constant in terms of the molecular canonical partition functions.
The grand canonical ensemble of a multicomponent system is described by the density operator, introduced by Eq. (**??**) but where

$$
p_j = \frac{e^{-\beta E_j + \beta \sum_k \mu_k N_j(k)}}{\sum_j e^{-\beta E_j + \beta \sum_k \mu_k N_j(k)}} = \Xi^{-1} e^{-\beta E_j + \beta \sum_k \mu_k N_j(k)}, \tag{316}
$$

with μ_k the chemical potential of species k and $N_j(k)$ the number of particles of species k in quantum state j. Eq. (316) is obtained by maximizing the entropy of the system, introduced by Eq. (164), subject to the constraints of constant volume, average internal energy E_j and average number of particles $N_j(k)$ for all the different species k in the system.
Substituting Eq. (316) into Eq. (97), we obtain

$$
S = \frac{E}{T} - \frac{1}{T} \sum_k \mu_k \overline{N(k)} + k\ln\Xi. \tag{317}
$$

Therefore,

$$
G \equiv \sum_k \mu_k \overline{N(k)} = E - TS + Tk\ln\Xi, \tag{318}
$$

and since $G = H - TS = E + PV - TS$,

$$
PV = kT\ln\Xi. \tag{319}
$$

Eqs. (317)—(319) provide an expression for the *change in internal energy* dE due to changes in the extensive properties of the system such as changes of volume dV, number of particles $d\overline{N(k)}$ and entropy dS,

$$dE = TdS - PdV + \sum_k \mu_k d\overline{N(k)}. \tag{320}$$

According to Eq. (320), μ_k can be computed as follows,

$$\mu_k = \left.\frac{\partial E}{\partial \overline{N(k)}}\right)_{S,V}, \tag{321}$$

and since $A = E - TS$ and $dA = dE - TdS - SdT$,

$$\mu_k = \left.\frac{\partial A}{\partial \overline{N(k)}}\right)_{T,V}. \tag{322}$$

Furthermore, according to Eqs. (322) and (64),

$$\beta\mu_k = -\left.\frac{\partial \ln Z}{\partial \overline{N(k)}}\right)_{T,V}. \tag{323}$$

The canonical partition function,

$$Z = \prod_k \frac{(q_t(k) * q_{\text{int}}(k))^{\overline{N(k)}}}{\overline{N(k)}!}, \tag{324}$$

is computed according to Eq. (248), where $q_t(j) = V(2\pi m_j kT)^{3/2}/h^3$ and q_{int} are the translational and internal canonical partition functions of species j, respectively. The underlying assumption, when computing Z according to Eq. (324), is that the constituents of the systems in the gas phase do not interact with each other except when they undergo reactive collisions.
Substituting Eq. (324) into Eq. (323) and using the Stirling Formula, introduced by Eq. (71), we obtain

$$\boxed{\beta\mu_k = -\ln\left(\frac{q_t(k) * q_{\text{int}}(k)}{\overline{N(k)}}\right)}. \tag{325}$$

To derive the law of mass action, we consider the following chemical reaction,

$$aA + bB \rightleftharpoons cC + dD, \tag{326}$$

where the stoichiometric coefficients (c_k =a, b, c and d) determine the relative changes in the number of moles of the molecular constituents due to the chemical reaction, as follows:

$$\frac{dN(A)}{a} = \frac{dN(B)}{b} = -\frac{dN(C)}{c} = -\frac{dN(D)}{d}. \tag{327}$$

Substituting Eq. (327) into Eq. (320) we obtain an expression for the change in internal energy at constant entropy S and volume V,

$$\left. dE \right)_{S,V} = \sum_j \mu_j d\overline{N}(j) = d\overline{N(A)} \sum_j \mu_j \nu_j, \tag{328}$$

where $\nu_j = c_j/a$.

The *minimum energy principle* establishes that

$$dE\Big)_{S,V} \geq 0, \tag{329}$$

for all arbitrary changes $d\overline{N(A)}$ in a system that was initially at equilibrium. Therefore, according to Eqs. (328) and (329),

$$\boxed{\sum_j \beta\mu_j\nu_j = 0.} \tag{330}$$

Substituting Eq. (325) into Eq. (330), we obtain

$$\sum_j \ln\left(\left[\frac{q_{\text{int}} * (2\pi m_j kT)^{3/2}}{h^3}\right)^{\nu_j}\left[\frac{V}{\overline{\overline{N(j)}}}\right]^{\nu_j}\right) = 0, \tag{331}$$

and

$$\ln\prod_j \left[\frac{q_{\text{int}} * (2 * \pi * m_j * k * T)^{3/2}}{h^3}\right)^{\nu_j}\left[\frac{V}{\overline{\overline{N(j)}}}\right]^{\nu_j} = 0. \tag{332}$$

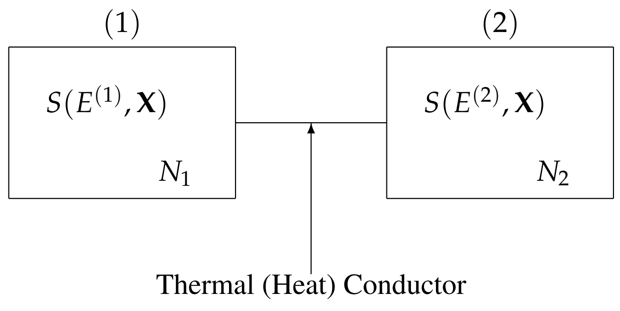
Therefore,

$$K(T) \equiv \prod_j \left(\frac{q_{\text{int}} * (2 * \pi * m_j * k * T)^{3/2}}{h^3}\right)^{-\nu_j} = \prod_j \left(\frac{V}{\overline{\overline{N(j)}}}\right)^{\nu_j}, \tag{333}$$

which is the *law of mass action*. Such law establishes that the concentrations of the constituent molecules in chemical equilibrium define an equilibrium constant $K(T)$ that depends only on the temperature of the system and on the nature of the chemical species. The first equality, in Eq. (333), provides a molecular expression of such equilibrium constant in terms of the canonical partition functions of the molecular constituents and T.

29.1 Minimum Energy Principle

The minimum energy principle, introduced by Eq. (329), is a consequence of the maximum entropy principle. This can be shown by considering the system at thermal equilibrium described by the following diagram:



Consider a small displacement of heat δE from compartment (2) to compartment (1). Since the system was originally at equilibrium, such a contraint in the distribution of thermal energy produces a constrained system whose entropy is smaller than the entropy of the system at equilibrium. Mathematically,

$$S(E^{(1)} + \delta E, \mathbf{X}) + S(E^{(2)} - \delta E, \mathbf{X}) < S(E^{(1)}, \mathbf{X}) + S(E^{(2)}, \mathbf{X}). \tag{334}$$

Now consider the system at equilibrium (i.e., without any constraints) with entropy $S(E,\mathbf{X})$ such that

$$S(E,\mathbf{X}) = S(E^{(1)}+\delta E,\mathbf{X}) + S(E^{(2)}-\delta E,\mathbf{X}). \tag{335}$$

Since, according to Eqs. (335) and (334),

$$S(E,\mathbf{X}) < S(E^{(1)},\mathbf{X}) + S(E^{(2)},\mathbf{X}), \tag{336}$$

and according to Eq. (51),

$$\left.\frac{\partial S}{\partial E}\right)_{V,N} = \frac{1}{T} > 0, \tag{337}$$

then

$$E < E^{(1)} + E^{(2)}. \tag{338}$$

Eq. (335) thus establishes that by imposing internal constraints at constant entropy the system that was initially at equilibrium with entropy $S(E,\mathbf{X})$ moves away from such equilibrium and its internal energy increases from E to $E^{(1)}+E^{(2)}$. Mathematically,

$$\left.dE\right)_{S,V} \geq 0, \tag{339}$$

which is the minimum energy principle.

30 Exam 2

Exam 2 CHEM 430b/530b
Statistical Methods and Thermodynamics

Exercise 1

(**20 points**) Item (1.1):Consider an ideal gas of bosons with $\mu = 0$ at temperature $T = 1/(\beta k)$. Show that

$$\overline{\delta_{n_k}\delta_{n_j}} = \delta_{kj}\left.\frac{\partial \overline{n_k}}{\partial(-\beta\epsilon_k)}\right)_{V,T}, \tag{340}$$

where $\delta_{n_k} = n_k - \overline{n_k}$ and $\overline{n_k}$ is the average occupation of the one-boson energy level k.
(**20 points**) Item (1.2): Explain the minimum energy principle and show that such principle is a consequence of the maximum entropy principle.
(**20 points**) Item (1.3): Explain the classical limit of the quantum statistical distributions.

Exercise 2

Consider an ideal gas of O_2 molecules adsorbed on a surface of area S in thermal equilibrium at temperature $T = 1/(k\beta)$. Assume that each O_2 molecule in the gas can freely translate, vibrate and rotate but only on the 2-dimensional surface. Assume that the rotational motion of O_2 molecules can be described by a rigid rotor model where the rotational eigenstates have degeneracy $g(J) = 2$ for all values of J except for J=0 for which g(J)=1. Assume that the rotational states have eigenvalues $E_J = \frac{\hbar^2 J^2}{2I_0}$, with J=0, 1, 2, ..., where I_0 is the moment of inertia of the O_2 molecule.
(**10 points**) Item (2.1): Compute the rotational canonical partition function of an O_2 molecule as a function of its moment of inertia I_0 and β.

(**10 points**) Item (2.2): Compute the vibrational canonical partition function of an O_2 molecule as a function of its vibrational frequency ω_0 and β.
(**10 points**) Item (2.3): Compute the translational canonical partition function of an O_2 molecule as a function of its total mass m, β and the surface area S.
(**10 points**) Item (2.4): Compute the average internal energy E of the O_2 gas as a function of β, the O_2 mass m , the area of the surface S, the O_2 moment of inertia I_0 and the total number N of O_2 molecules on the surface.

Solution:
Exercise 1:
Item (1.1): Since $\delta_{n_k} = n_k - \overline{n_k}$,

$$\overline{\delta_{n_k}\delta_{n_j}} = \overline{n_k n_j} - \overline{n_k}\,\overline{n_j}, \tag{341}$$

where

$$\overline{n_j} = \frac{1}{e^{\beta\epsilon_j} - 1}, \tag{342}$$

because $\mu = 0$. Therefore,

$$\overline{n_k n_j} = \frac{1}{\Xi}\frac{\partial^2 \sum_{n_1=0}^{\infty}\sum_{n_2=0}^{\infty}...e^{-\beta(\epsilon_1 n_1 + \epsilon_2 n_2 + ...)}}{\partial(\beta\epsilon_j)\partial(\beta\epsilon_k)}, \tag{343}$$

or

$$\overline{n_k n_j} = \frac{1}{\Xi}\frac{\partial^2}{\partial(\beta\epsilon_k)\partial(\beta\epsilon_j)}\prod_j \frac{1}{1 - e^{-\beta\epsilon_j}}. \tag{344}$$

Computing the first partial derivative we obtain

$$\overline{n_k n_j} = \frac{1}{\Xi}\frac{\partial}{\partial(-\beta\epsilon_j)}\frac{e^{-\beta\epsilon_k}}{(1 - e^{-\beta\epsilon_k})^2}\prod_{l\neq k}\frac{1}{1 - e^{-\beta\epsilon_l}}, \tag{345}$$

and computing the second partial derivative we obtain

$$\begin{aligned}\overline{n_k n_j} \quad &= \tfrac{1}{\Xi}[\textstyle\prod_{l\neq k} \tfrac{1}{1-e^{-\beta\epsilon_l}}\delta_{kj}\tfrac{[(1-e^{-\beta\epsilon_k})^2 e^{-\beta\epsilon_k} + e^{-\beta\epsilon_k}2(1-e^{-\beta\epsilon_k})e^{-\beta\epsilon_k}]}{(1-e^{-\beta\epsilon_k})^4} \\ &+ \tfrac{e^{-\beta\epsilon_k}}{(1-e^{-\beta\epsilon_k})^2}\tfrac{e^{-\beta\epsilon_j}}{(1-e^{-\beta\epsilon_j})^2}\textstyle\prod_{l\neq j, l\neq k}\tfrac{1}{1-e^{-\beta\epsilon_l}}],\end{aligned} \tag{346}$$

where,

$$\Xi = \prod_j \frac{1}{1 - e^{-\beta\epsilon_j}}. \tag{347}$$

Therefore,

$$\overline{n_k n_j} = \delta_{kj}\frac{e^{-\beta\epsilon_k}}{(1 - e^{-\beta\epsilon_k})^2} + \frac{e^{-\beta\epsilon_k}}{(1 - e^{-\beta\epsilon_k})}\frac{e^{-\beta\epsilon_j}}{(1 - e^{-\beta\epsilon_j})}, \tag{348}$$

and

$$\overline{\delta_{n_k}\delta_{n_j}} = \delta_{kj}\frac{e^{-\beta\epsilon_k}}{(1 - e^{-\beta\epsilon_k})^2} = \delta_{kj}\frac{\partial}{\partial(-\beta\epsilon_k)}\frac{1}{e^{\beta\epsilon_k} - 1}, \tag{349}$$

which, according to Eq (264), gives

$$\boxed{\overline{\delta_{n_k}\delta_{n_j}} = \delta_{kj}\frac{\partial \overline{n_k}}{\partial(-\beta\epsilon_k)}.} \tag{350}$$

Item (1.2): See topic “Minimum Energy Principle” on page 55 of the lecture notes.
Item (1.3): See topic “Classical limit of Quantum Statistical Distributions” on page 36 of the lecture notes.

Exercise 2:
Item (2.1): The rotational canonical partition function of an O_2 molecule is

$$q_{\text{rot}} = \sum_{J=0}^{\infty} g(J)e^{-\beta\epsilon_J}. \tag{351}$$

Taking the continuous limit we obtain,

$$q_{\text{rot}} \approx \lim_{\epsilon\to 0}\int_{\epsilon}^{\infty} dJ g(J)e^{-\beta\epsilon_J} = \sqrt{\frac{\pi 2I_0}{\beta\hbar^2}}. \tag{352}$$

Item (2.2): The vibrational canonical partition function of an O_2 molecule is

$$q_{\text{vib}} = \sum_{\nu=0}^{\infty} e^{-\beta\epsilon_\nu} = \frac{e^{-\beta\hbar\omega_0/2}}{1-e^{-\beta\hbar\omega_0}}. \tag{353}$$

Item (2.3): The translational canonical partition function of an O_2 molecule is

$$q_{\text{transl}} = \frac{S}{\pi^2}\sum_{k_x}\sum_{k_y} e^{-\beta\frac{(k_x^2+k_y^2)\hbar^2}{2m}} \approx \frac{S}{\pi^2}\int dk_x e^{\frac{-\beta k_x^2\hbar^2}{2m}}\int dk_y e^{\frac{-\beta k_y^2\hbar^2}{2m}}. \tag{354}$$

Therefore,

$$q_{\text{transl}} \approx \frac{S\pi 2m}{\beta h^2}, \tag{355}$$

where $S = L_x \times L_y$, with L_x and L_y the lengths of the surface along the x and y directions, respectively.
Item (2.4): The total canonical partition function of the system is

$$Q = \frac{1}{N!}\prod_{j=1}^{N} q_{\text{rot}}q_{\text{vib}}q_{\text{transl}}. \tag{356}$$

Substituting the expressions for q_{rot}, q_{vib} and q_{transl} computed in items (2.1)—(2.3) we obtain,

$$Q = \frac{1}{N!}\prod_{j=1}^{N} S\frac{2\pi m}{\beta h^2}(e^{\beta\hbar\omega_0/2}-e^{-\beta\hbar\omega_0/2})^{-1}\sqrt{\frac{2\pi I_0}{\beta\hbar^2}} = \frac{1}{N!}\prod_{j=1}^{N}\frac{\pi x}{\beta^{3/2}(e^{\beta y}-e^{-\beta y})}. \tag{357}$$

Therefore, the average internal energy of the O_2 gas is

$$\overline{E} = \frac{\sum_{j=1}^{N}\partial \ln Q}{\partial\beta} = \frac{N}{Q}\frac{\partial Q}{\partial(-\beta)} = +\frac{\beta^{3/2}(e^{\beta y}-e^{-\beta y})N}{x}\frac{x(3/2\beta^{1/2}(e^{\beta y}-e^{-\beta y})+\beta^{3/2}y(e^{\beta y}+e^{-\beta y}))}{\beta^{3/2}(e^{\beta y}-e^{-\beta y})^2}, \tag{358}$$

which gives

$$\overline{E} = \sum_{j=1}^{N} \frac{3/2(e^{\beta y} - e^{-\beta y}) + \beta y(e^{\beta y} + e^{-\beta y})}{\beta(e^{\beta y} - e^{-\beta y})} = (\frac{3}{2\beta} + \frac{3y}{2}\frac{(e^{\beta y} + e^{-\beta y})}{(e^{\beta y} - e^{-\beta y})}) * N, \tag{359}$$

where $x = \frac{S2m\pi}{\hbar^2}\sqrt{\frac{2\pi I_0}{\hbar^2}}$ and $y = \frac{\hbar\omega_0}{2}$.

31 Quiz 2

Quiz 2 CHEM 430b/530b
Statistical Methods and Thermodynamics

(**30 points**) Exercise 1: Derive the Fermi-Dirac distribution.
(**30 points**) Exercise 2: Derive the Bose-Einstein distribution.
(**40 points**) Exercise 3: Derive an expression for the average density of electrons $\overline{N}/V$ in a metal at $T = 0$ K, as a function of the Fermi energy μ and the electron mass m.

Solution:
Exercise 1: See topic "Bose-Einstein and Fermi-Dirac distributions" on pages 34 and 35 of the lecture notes.
Exercise 2: See topic "Bose-Einstein and Fermi-Dirac distributions" on pages 34 and 35 of the lecture notes.
Exercise 3: According to Eq. (285),

$$\overline{N} = 8\pi \int_0^{K_f} dKK^2 \left(\frac{L}{2\pi}\right)^3 = 8\pi \left(\frac{L}{2\pi}\right)^3 \frac{K_f^3}{3} = \frac{2V}{(2\pi)^3}\frac{4}{3}\pi K_f^3, \tag{360}$$

where K_f is the Fermi momentum defined as follows

$$\frac{\hbar^2 K_f^2}{2m} = \mu. \tag{361}$$

Therefore,

$$\frac{\overline{N}}{V} = \frac{2}{(2\pi)^3}\frac{4}{3}\pi\left(\frac{2m\mu}{\hbar^2}\right)^{(3/2)}. \tag{362}$$

32 Ising Model

The goal of this section is to introduce the *Ising model* which is a simple model of systems with *interparticle interactions*. We compute its canonical partition function according to both the macroscopic approximation and the rigorous transfer matrix technique.

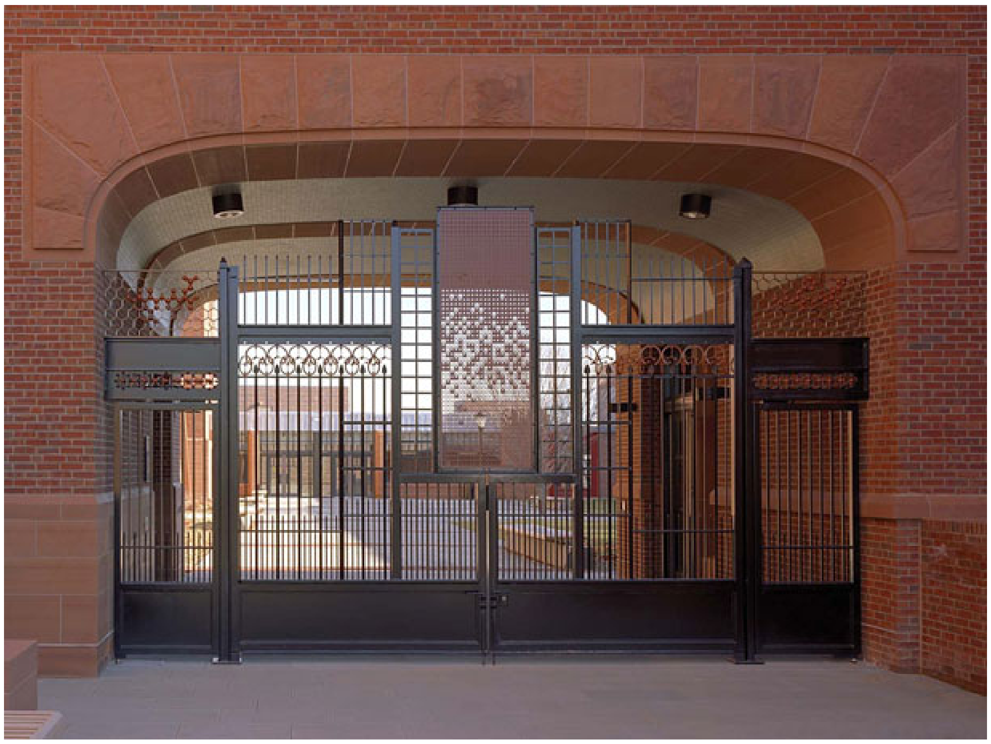
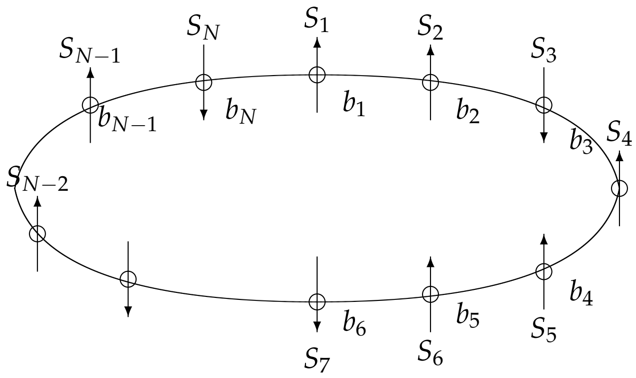


Figure 3: Chemistry Gate, Yale University. The central panel represents the Ising Lattice, a tribute to Nobel-prize winning Yale chemist Lars Onsager, who provided a proof of the Ising model for phase change.

The 1-dimensional Ising model is described by a system of N spins arranged in a ring as represented in the following diagram:



The Hamiltonian of the system is

$$H = -\bar{\mu} B \sum_j S_j - J \sum_{jk} S_j S_k, \tag{363}$$

where $\bar{\mu}$ is the magnetic dipole moment, B is an external magnetic field and J is the coupling constant between spins. The sum of products $S_j S_k$ defines the interaction between spins, including only nearest neighbors.
In the absence of an external magnetic field, the canonical partition function of the system is

$$Z = \sum_{S_1=-1}^{1} \sum_{S_2=-1}^{1} ... \sum_{S_N=-1}^{1} e^{\beta J S_1 S_2} e^{\beta J S_2 S_3} ... e^{\beta J S_N S_1}, \tag{364}$$

The partition function, introduced by Eq. (364), is *approximately* equal to

$$Z \approx \sum_{b_1=\pm 1} ... \sum_{b_N=\pm 1} e^{\beta J \sum_{j=1}^{N} b_j} = [2\cosh(\beta J)]^N, \tag{365}$$

where we have replaced the products of interaction $S_k S_j$ by bonds $b_j = \pm 1$ and we have assumed that all bonds are linearly independent. Note, however, that such approximation consist in assuming that N is sufficiently large (i.e., $N >> 1$) as to neglect the energy of one bond relative to the total energy of the system, since only N-1 bonds are actually linearly independent.
To perform a rigorous calculation of the canonical partition function introduced by Eq. (364), we define the *transfer function* in the absence of an external magnetic field as follows,

$$T(S_i, S_{i+1}) \equiv \exp(\beta J S_i S_{i+1}). \tag{366}$$

Substituting Eq. (366) into Eq. (364) we obtain

$$Z = \sum_{S_1=-1}^{1} \sum_{S_2=-1}^{1} ... \sum_{S_N=-1}^{1} T(S_1, S_2) T(S_2, S_3) ... T(S_N, S_1). \tag{367}$$

This expression corresponds to the trace of a product of N identical 2×2 matrices. To show this we introduce the *transfer matrix*,

$$\mathbf{T} \equiv \begin{pmatrix} T(1,1) & T(1,-1) \\ T(-1,1) & T(-1,-1) \end{pmatrix}. \tag{368}$$

Note that the element (j,k) of $\mathbf{T}^2$ is

$$\mathbf{T}^2(j,k) = \sum_{S_2=-1}^{1} T(j, S_2) T(S_2, k), \tag{369}$$

and therefore

$$Z = \sum_{S_1=-1}^{1} \mathbf{T}^N(S_1, S_1) = \mathrm{Tr}\{\mathbf{T}^N\}. \tag{370}$$

Thus the calculation of the canonical partition function for the 1-dimensional Ising model has been reduced to that of computing the trace of the Nth power of the transfer matrix.

We note that $\mathbf{T} = \Gamma \mathbf{T}_d \Gamma^\dagger$, with $\mathbf{T}_d$ the diagonal matrix of eigenvalues of $\mathbf{T}$, and Γ the matrix of eigenvectors of $\mathbf{T}$ with $\Gamma \Gamma^\dagger = I$, so $\mathbf{T}^N = \Gamma \mathbf{T}_d^N \Gamma^\dagger$. Furthermore, the $\mathrm{Tr}\{\mathbf{T}^N\} = \mathrm{Tr}\{\Gamma \mathbf{T}_d^N \Gamma^\dagger\} = \mathrm{Tr}\{\Gamma^\dagger \Gamma \mathbf{T}_d^N\} = \mathrm{Tr}\{\mathbf{T}_d^N\}$ since the trace is invariant under cyclic permutations (*i.e.*, Tr(ABC)=Tr(CAB) since $\sum_{j,k} (AB)_{jk} C_{kj} = \sum_{j,k} C_{kj} (AB)_{jk}$).

Now, the trace of a matrix is the sum of its eigenvalues and the eigenvalues of $\mathbf{T}^N$ are $\lambda_\pm^N$, where $\lambda_\pm$ are the eigenvalues of $\mathbf{T}$ determined by the equation

$$\begin{vmatrix} e^{\beta J} - \lambda & e^{-\beta J} \\ e^{-\beta J} & e^{\beta J} - \lambda \end{vmatrix} = 0, \tag{371}$$

with solutions

$$\lambda_\pm = e^{\beta J} \pm e^{-\beta J}. \tag{372}$$

Hence, the partition function is simply,

$$Z = \lambda_+^N + \lambda_-^N = 2^N(\cosh^N(\beta J) + \sinh^N(\beta J)). \tag{373}$$

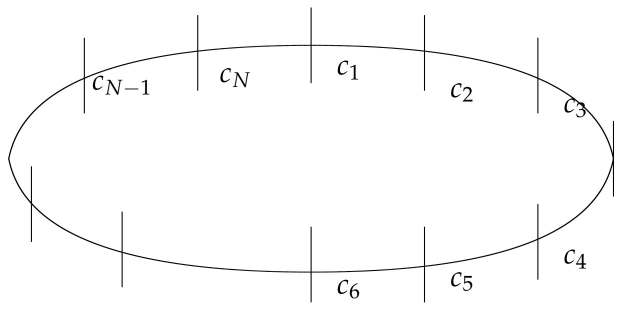
Note that when N is sufficiently large, $\sinh^N(\beta J) << \cosh^N(\beta J)$ and Eq. (373) coincides with Eq. (365). In the presence of a magnetic field, however,

$$Z = \sum_{S_1=-1}^{1} \sum_{S_2=-1}^{1} \dots \sum_{S_N=-1}^{1} e^{\beta J S_1 S_2 + \beta \mu B(S_1+S_2)/2} e^{\beta J S_2 S_3 + \beta \mu B(S_2+S_3)/2} \dots e^{\beta J S_N S_1 + \beta \mu B(S_N+S_1)/2}, \tag{374}$$

Exercise: 1-dimensional Ising Model
Compute the canonical partition function introduced by Eq. (374) by implementing the transfer matrix approach.

33 Lattice Gas

The goal of this section is to show that with a simple change of variables, the *Ising model* can be mapped into the *lattice gas* which is a simple model of density fluctuations and liquid-gas transformations. The 1-dimensional lattice gas model is described by the following diagram:



The lattice divides space into cells $c_1, c_2, ..., c_N$. Each cell has an occupation number $n_j = 0, 1$. The interaction between particles occupying the cells is modeled by assuming that the energy associated with a pair of occupied nearest neighbor cells is $-\epsilon n_j n_k$ and the total energy of the system is

$$E = -\mu \sum_j n_j - \epsilon \sum_j \sum_k n_j n_k, \tag{375}$$

where j is the index of the cell and μ is the chemical potential of a particle. The partition function of the lattice model is

$$\Xi = \sum_{n_1=0}^{1} \sum_{n_2=0}^{1} \dots \sum_{n_N=0}^{1} e^{\beta\mu \sum_{j=1}^{N} n_j + \beta\epsilon \sum_j \sum_k n_j n_k}. \tag{376}$$

To show the correspondence between the lattice gas and the Ising model, we make the variable transformation $n_j \equiv (S_j + 1)/2$ and we obtain

$$\Xi = \sum_{S_1=-1}^{1} \sum_{S_2=-1}^{1} \dots \sum_{S_N=-1}^{1} e^{\beta\frac{\mu}{2} \sum_j (S_j+1) + \beta\frac{\epsilon}{4} \sum_j \sum_k (S_j+1)(S_k+1)}, \tag{377}$$

Therefore the lattice model is isomorphic with the Ising model: "Spin up" in the Ising model corresponds to an occupied cell in the lattice model, "spin down" corresponds to an empty cell, the magnetic field in the Ising model corresponds (within constants) to the chemical potential in the lattice gas and the coupling constant in the Ising model is $\epsilon/4$ in the lattice gas.
The Ising model can also be mapped into many other problems in Chemistry and beyond, ranging from models of population dynamics to models of the brain.

34 Mean Field Theory

The goal of this section is to introduce the so-called *mean field theory* (also known as *self consistent field theory*) and to illustrate the theory by applying it to the description of the Ising model.
The main idea of the mean field theory is to focus on one particle and assume that the most important contribution to the interactions of such particle with its neighboring particles is determined by the *mean field* due to the neighboring particles.
In the 1-dimensional Ising model, for instance, the average force $\overline{F_k}$ exerted on spin S_k is

$$\overline{F_k} \equiv -\overline{\frac{\partial H}{\partial S_k}} = \bar{\mu} B + J \sum_j \overline{S_j}, \tag{378}$$

where the index j includes all the nearest neighbors of spin S_k. Therefore, the average magnetic field $\overline{B}$ acting on spin S_k is

$$\overline{B} \equiv \frac{\overline{F_k}}{\bar{\mu}} = B + \Delta B, \tag{379}$$

where

$$\Delta B = J 2 \overline{S_k} / \bar{\mu}, \tag{380}$$

is the contribution to the mean field due to the nearest neighbors. Note that $\overline{S_k} = \overline{S_j}$ when all spins are identical.

Eq. (379) defines the *self consistent* aspect of the theory, since according to such equation the mean field $\overline{B}$ acting on spin S_k is determined by its own mean value $\overline{S_k}$.
The assumption that the interactions of a spin with its neighboring spins can be approximately described by the *mean field*, introduced by Eq. (380), introduces an enormous simplification. Such mean field approximation simplifies the many body statistical mechanics problem to a *one-body* problem (i.e., Eq. (379) transforms the problem of N interacting spins influenced by an external magnetic field B to a problem of N non-interacting spins influenced by the mean field $\overline{B}$).
The partition function, under the mean field approximation, is

$$Z \approx \sum_{S_1} \sum_{S_2} \dots \sum_{S_N} e^{\beta \sum_j S_j (B + \Delta B) \bar{\mu}} = 2^N \cosh^N(\beta \bar{\mu} \overline{B}), \tag{381}$$

and the average value of S_k is

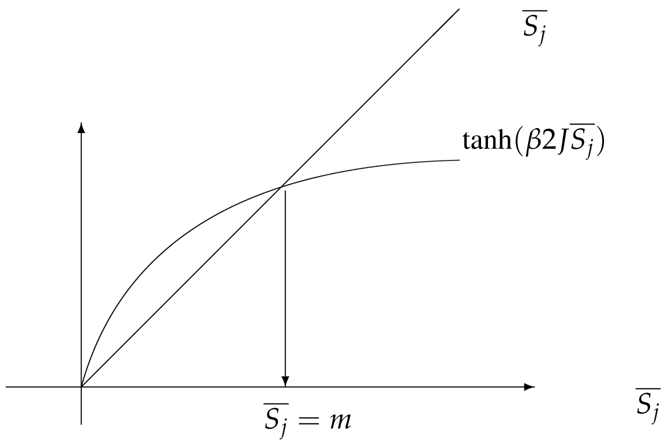
$$\overline{S_k} = \frac{1}{N}\sum_j p_j(\sum_l S_l(j)) = \frac{1}{N}Z^{-1}\sum_{S_1}\sum_{S_2}...\sum_{S_N}(\sum_l S_l)e^{\beta\sum_j S_j(B+\Delta B)\bar{\mu}}, \tag{382}$$

where p_j is the probability of state j. The average value of spin is

$$\overline{S_k} = \frac{1}{N}\frac{\partial \ln Z}{\partial(\beta\bar{\mu}\overline{B})} = \tanh(\beta\bar{\mu}(B + 2J\overline{S_k}/\bar{\mu})). \tag{383}$$

Note that Eq. (383) involves a transcendental equation. Its solution corresponds to the value of $\overline{S_k} = m$ for which the function on the left hand side of Eq. (383) (i.e., $\overline{S_k}$) equals the function on the right hand side of Eq. (383) (i.e., $\tanh(\beta\bar{\mu}(B + 2J\overline{S_k}/\bar{\mu}))$.

In the absence of an external magnetic field (i.e., when $B = 0$), Eq. (383) always has the trivial solution $S_k = 0$ and a non-trivial solution $S_k = m$ only when $\beta 2J > 1$. Such solution is represented by the following diagram:



The diagram shows that the mean field theory predicts *spontaneous magnetization* (i.e., magnetization in the absence of an external magnetic field) for the 1-dimensional Ising model at any temperature $T < 2J/k$, since there is a non-trivial solution $\overline{S_k} = m$ for which Eq. (383) is satisfied. Unfortunately, however, this result is erroneous! The 1-dimensional Ising model does not undergo spontaneous magnetization at any finite temperature, since each spin has only two nearest neighbors and the stabilization energy due to two nearest neighbors is not enough to overcome the randomization process due to thermal fluctuations. This simple example, however, illustrates the theory including the fact that it is sometimes inaccurate near critical points. The theory works better in higher dimensionality, e.g., in the 2-dimensional Ising model where the theory predicts spontaneous magnetization at a critical temperature $T_c = 4J/K$ that is close to the experimental value 2.3 J/K.

Exercise: Show that there is no spontaneous magnetization in the 1-dimensional Ising model at finite temperature by computing the average magnetization $M = \bar{\mu}\sum_j \overline{S_j}$ from the exact canonical partition function. Hint: Compute the average magnetization in the presence of an external magnetic field and show that in the limit when $B \to 0$ such magnetization becomes negligible.

34.1 Variational Mean Field Theory

The goal of this section is to introduce a variational approach for computing the optimum mean field determined by the Gibbs-Bogoliubov-Feynman equation and to illustrate such variational method by applying it to the description of the 1-dimensional Ising model.
Consider the task of computing the canonical partition function Z of the one-dimensional Ising model,

$$Z(K,N) = \sum_{S_1}\sum_{S_2}\cdots\sum_{S_N} e^{-\beta E(S_1,S_2,S_2,S_3,\ldots S_N)}, \tag{384}$$

where

$$E(S_1,S_2,S_2,S_3,\ldots S_N) = -\bar{\mu}B\sum_j S_j - J\sum_{jk} S_jS_k. \tag{385}$$

The mean field approximation, introduced by Eq. (381), is

$$Z_{MF}(K,N) = \sum_{S_1}\sum_{S_2}\cdots\sum_{S_N} e^{-\beta E_{MF}(S_1,S_2,S_2,S_3,\ldots S_N)} = 2^N\cosh^N(\beta\bar{\mu}(B+\Delta B)), \tag{386}$$

with

$$E_{MF}(S_1,S_2,S_2,S_3,\ldots S_N) = -\bar{\mu}(B+\Delta B)\sum_j S_j, \tag{387}$$

where $\Delta B = J2\overline{S_k}/\bar{\mu}$. Note that the mean field partition function Z_{MF}, introduced by Eq. (386) is an *approximation* to the actual partition function $Z(K,N)$, introduced by Eq. (384). The goal of the variational treatment is, therefore, to optimize the expression of the mean field ΔB in order for Z_{MF} to be as similar as possible to $Z(K,N)$.
To obtain a variational expression that involves both $Z_{MF}(K,N)$ and $Z(K,N)$ (i.e., the Gibbs-Bogoliubov-Feynman equation) we note that, according to Eqs. (384) and (386),

$$Z(K,N) = Z_{MF}(K,N)\frac{\sum_{S_1}\sum_{S_2}\cdots\sum_{S_N} e^{-\beta\Delta E}e^{-\beta E_{MF}}}{\sum_{S_1}\sum_{S_2}\cdots\sum_{S_N} e^{-\beta E_{MF}}} = Z_{MF}\langle e^{-\beta\Delta E}\rangle, \tag{388}$$

where $\Delta E = E - E_{MF}$, and $\langle\rangle$ indicates a mean field ensemble average. Furthermore, we note that

$$\langle e^{-\beta\Delta E}\rangle = \langle e^{-\beta\langle\Delta E\rangle}e^{-\beta(\Delta E-\langle\Delta E\rangle)}\rangle \geq e^{-\beta\langle\Delta E\rangle}, \tag{389}$$

since $\langle e^{-\beta\langle\Delta E\rangle}\rangle = e^{-\beta\langle\Delta E\rangle}$ and $e^x \geq 1+x$. Therefore,

$$Z(K,N) \geq Z_{MF}(K,N)e^{-\beta\langle\Delta E\rangle}, \tag{390}$$

which is the Gibbs-Bogoliubov-Feynman equation.
Eq. (390) allows us to find the optimum mean field by maximizing the right hand side (r.h.s.) of Eq. (390) with respect to ΔB.
Note that according to Eqs. (386) and (383),

$$\frac{\partial Z_{MF}}{\partial\Delta B} = Z_{MF}N\beta\bar{\mu}\langle s_k\rangle. \tag{391}$$

and according to Eq. (385) and (387),

$$\langle\Delta E\rangle = -J\sum_{j=1}^{N}\sum_k\langle s_js_k\rangle + \Delta B\bar{\mu}\sum_j\langle s_j\rangle = -J\frac{N}{2}2\langle s_j\rangle^2 + \Delta B\bar{\mu}N\langle s_j\rangle. \tag{392}$$

Therefore, computing the derivative of the r.h.s. of Eq. (390) with respect to ΔB and making such derivative equal to zero we obtain, according to Eqs. (391) and (392),

$$Z_{MF}N\beta\bar{\mu}\langle s_k\rangle e^{-\beta\langle\Delta E\rangle} + Z_{MF}e^{-\beta\langle\Delta E\rangle}(-\beta)(-J\frac{N}{2}2\frac{\partial\langle s_j\rangle}{\partial\Delta B}2\langle s_j\rangle + \bar{\mu}N\langle s_j\rangle) + \Delta B\bar{\mu}N\frac{\partial\langle s_j\rangle}{\partial\Delta B}) = 0. \quad (393)$$

Therefore, solving for ΔB in Eq. (393) we obtain

$$\Delta B = 2J\langle s_j\rangle/\bar{\mu}, \quad (394)$$

which is identical to the mean field introduced by Eq. (380). This means that the mean field introduced by Eq. (380) is the optimum field as determined by the Gibbs-Bogoliubov-Feynman equation (i.e., the mean field that maximizes the r.h.s. of Eq. (390)).

35 Renormalization Group Theory

The goal of this section is to introduce several concepts of *Renormalization Group Theory* and to illustrate such concepts with the 1-dimensional Ising model.
Consider the task of computing the canonical partition function Z of the one-dimensional Ising model in the absence of an external magnetic field. According to Eq. (364),

$$Z(K,N) = \sum_{S_1}\sum_{S_2}...\sum_{S_N} e^{K(S_1S_2+S_2S_3+...+S_NS_1)}, \quad (395)$$

where *coupling parameter* $K \equiv \beta J$ and N is the total number of spins. Note that according to Eq. (395),

$$\lim_{K\to 0} Z(K,N) = \prod_{j=1}^{N}\sum_{S_j=-1}^{1} 1 = 2^N. \quad (396)$$

The renormalization group strategy for the 1-dimensional Ising model can be described as follows.
Step (1). Sum over the even numbered spins in Eq. (395). Note that summing, e.g., over S_2 we obtain

$$Z(K,N) = \sum_{S_1,S_3,S_4,S_5,S_6,S_7...} [e^{K(S_1+S_3)} + e^{-K(S_1+S_3)}]e^{KS_3S_4}e^{KS_4S_5}e^{KS_5S_6}e^{KS_6S_7}..., \quad (397)$$

summing over S_2 and S_4 we obtain,

$$Z(K,N) = \sum_{S_1,S_3,S_5,S_6,S_7...} [e^{K(S_1+S_3)} + e^{-K(S_1+S_3)}][e^{K(S_3+S_5)} + e^{-K(S_3+S_5)}]e^{KS_5S_6}e^{KS_6S_7}..., \quad (398)$$

and summing over all even numbered spins we obtain

$$Z(K,N) = \sum_{S_1,S_3,S_5,S_7...} [e^{K(S_1+S_3)} + e^{-K(S_1+S_3)}][e^{K(S_3+S_5)} + e^{-K(S_3+S_5)}][e^{K(S_5+S_7)} + e^{-K(S_5+S_7)}].... \quad (399)$$

Step (2). Rewrite the remaining sum (i.e., the sum over odd numbered spins introduced by Eq. (399) by implementing the *Kadanoff transformation*

$$e^{K(S+S')} + e^{-K(S+S')} = f(K)e^{K'SS'}, \quad (400)$$

where both $f(K)$ and K' are functions of K. Substituting Eq. (400) into Eq. (399) we obtain

$$Z(K,N) = f(K)^{N/2}\sum_{S_1,S_3,S_5,S_7...} e^{K'S_1S_3}e^{K'S_3S_5}e^{K'S_5S_7}.... = f(K)^{N/2}Z(K',N/2). \quad (401)$$

Note that such transformation allows us to rewrite the partition function $Z(K,N)$ in terms of a *renormalized* partition function $Z(K',N/2)$ (i.e., a partition function with new parameters that describes an Ising model with half the number of spins and a different coupling parameter K'). To determine the *renormalization group equations* (i.e., K' and $f(K)$ as a function of K) and show that $K' < K$, we note that when $S = S' = \pm 1$, Eq. (400) gives

$$e^{2K} + e^{-2K} = f(K)e^{K'}, \tag{402}$$

and when $S = -S' = \pm 1$, Eq. (400) gives

$$2 = f(K)e^{-K'}. \tag{403}$$

Therefore, solving for $f(K)$ in Eq. (403) and substituting into Eq. (402) we obtain

$$K' = \frac{1}{2}\ln(\cosh(2K)), \tag{404}$$

and substituting Eq. (404) into Eq. (403) we obtain

$$f(K) = 2\cosh^{\frac{1}{2}}(2K). \tag{405}$$

Eqs. (404) and (405) are called *renormalization group equations* since they provide the renormalization scheme.

Step (3). Go to (1), replacing $Z(K,N)$ by $Z(K',N/2)$.

Step (3) is repeated each time on the subsequent (renormalized) partition function (i.e., $Z(K'',N/4)$, $Z(K''',N/8)$, $Z(K^{IV},N/16)$, $Z(K^{V},N/32)$, ... etc.) until the renormalized parameters become approximately constant (i.e., until the renormalized parameters reach a *fixed point* and become invariant under the Kadanoff transformation). Note that, according to Eq. (404), $K > K' > K'' > K'''$, etc., so after a few iterations the coupling parameter becomes negligibly small and the partition function can be approximated by using Eq. (396) as follows:

$$\begin{aligned}\ln Z(K,N) \approx\ & \tfrac{N}{2}\ln[2\cosh^{1/2}(2K)] + \tfrac{N}{4}\ln[2\cosh^{1/2}(2K')] + \tfrac{N}{8}\ln[2\cosh^{1/2}(2K'')] + \\ & \tfrac{N}{16}\ln[2\cosh^{1/2}(2K''')] + \tfrac{N}{32}\ln[2\cosh^{1/2}(2K^{IV})] + \tfrac{N}{64}\ln[2\cosh^{1/2}(2K^{V})] + \tfrac{N}{2^6}\ln 2.\end{aligned} \tag{406}$$

The renormalization group strategy thus involves computing the total sum, introduced by Eq. (395), step by step. The success of the approach relies on the fact that the sum converges to an expression that can be easily computed, after a few iterations.

Sometimes the partition function is known for a specific value of the coupling parameter (e.g., for $K' \approx 0$ in the 1-dimensional Ising model). The renormalization group theory can then be implemented to compute the partition function of the system for a different value K of the coupling constant. This is accomplished by inverting Eq. (404) as follows:

$$K = \frac{1}{2}\cosh^{-1}[\exp(2K')]. \tag{407}$$

and computing $Z(K,N)$ from $Z(K',N/2)$ according to Eq. (401).

One could also define the function $g(K)$ as follows

$$Ng(K) \equiv \ln Z(K,N), \tag{408}$$

and substituting Eq. (407) into Eq. (401) we obtain

$$Ng(K) = \frac{N}{2}\ln 2 + \frac{N}{2}\ln(\cosh^{\frac{1}{2}}(2K)) + \frac{N}{2}g(K'). \tag{409}$$

Therefore, given the partition function $Z(K', N)$ for a system with coupling constant K', one can compute $g(K')$ and K according to Eqs. (408) and (407), respectively. The partition function $Z(K, N) = \exp(Ng(K))$ is then obtained by substituting the values of $g(K')$ and K in Eq. (408).
Note that according to this procedure, $K > K'$ and the subsequent iterations give larger and larger values of K. This indicates that the flow of K has only two fixed points at K= 0 (e.g., at infinite temperature) and K= ∞ (e.g., at 0 K). Systems with phase transitions, however, have nontrivial fixed points at intermediate values of K. For instance, following a similar procedure, as the one described in this section, it is possible to show that the 2-dimensional Ising model has an additional fixed point K_c and that the heat capacity $C = \frac{d^2}{dk^2}g(k)$ divergers at K_c. Thus, K_c determines the critical temperature where the system undergoes a phase transition and spontaneosly magnetizes.
Hint for Exercise 5.17: From Eq. (c) solve for K to obtain Eq. (c'): $K = \frac{1}{4}\cosh^{-1}(e^{8K'/3})$. Then, substitute K defined according to (c') into (b) to obtain: $g(K') = 2g(K) - \ln\{2[\cosh(x/2)]^{1/2}e^{K'/3}\}$, with $x = \cosh^{-1}(e^{8K'/3})$. Next, note that $(\cosh(x/2))^2 = (e^x + e^-x + 2)/4 = (\cosh(x) + 1)/2$, so $[\cosh(x/2)]^{1/2} = (\cosh(x) + 1)^{1/4}/2^{1/4} = 2^{-1/4}(e^{8K'/3} + 1)^{1/4} = e^{K'/3}2^{-1/4}(e^{4K'/3} + e^{-4K'/3})^{1/4} = e^{K'/3}\cosh^{1/4}(4K'/3)$. Therefore, $g(K') = 2g(K) - \ln\{2e^{2K'/3}\cosh^{1/4}(4K'/3)\}$, giving Eq. (b'): $g(K) = \frac{1}{2}g(K') + \frac{1}{2}\ln\{2e^{2K'/3}\cosh^{1/4}(4K'/3)\}$.

36 Metropolis Monte Carlo Method

The goal of this section is to introduce the Metropolis Monte Carlo Method (*J. Chem. Phys.* **21**, 1087, *1953*) and to illustrate the algorithm as applied to the computation of canonical ensemble averages for the Ising model.
The Metropolis Monte Carlo method is a computational approach (i.e., an algorithm) for generating a set of N configurations of the system $\xi_1, \xi_2, \xi_3, ..., \xi_N$ such that

$$\lim_{N\to\infty} \frac{N_\xi}{N} = P(\xi), \tag{410}$$

where P(ξ) is a given probability distribution (e.g., the Boltzmann distribution P(ξ) = Z^{-1} exp[-β E(ξ)]) and N_ξ is the number of configurations ξ (e.g., the number of configurations generated with a particular arrangement of spins $S_1(\xi), S_2(\xi), ..., S_N(\xi)$ in the Ising model).

The Metropolis Monte Carlo algorithm can be described as follows:
Step (1): Pick a configuration ξ_n (the initial configuration can be *any* configuration of the system, e.g., any arrangement of spins in the Ising model).
Step (2): Pick a *trial configuration* ξ_t (usually a configuration similar to ξ_n) and compute the probability ratio $R = \frac{P(\xi_t)}{P(\xi_n)}$. Pick a random number p with value between 0 and 1. Make $\xi_{n+1} = \xi_t$ if $p \leq R$. Otherwise, make $\xi_{n+1} = \xi_n$.
Step (3): Go to (2) replacing ξ_n by ξ_{n+1}.
Step (3) is repeated N times, where N is a sufficiently large number. Note that, according to step (2), the probability of accepting a trial configuration ξ_t by making $\xi_{n+1} = \xi_t$ from a configuration ξ_n is

$$P_{\xi_n,\xi_t} = \begin{cases} R = \frac{P(\xi_t)}{P(\xi_n)}, & \text{when } P(\xi_t) < P(\xi_n), \\ 1, & \text{otherwise.} \end{cases} \tag{411}$$

The goal of the remaining of this section is to prove that such an algorithm indeed produces an ensemble of configurations that satisfies Eq. (410).
Consider an ensemble of N configurations with N(ξ) members of the ensemble in state ξ. Apply the Metropolis Monte Carlo algorithm to each member of the ensemble by setting $\xi_n = \xi$ and $\xi_t = \xi'$ in step (2), where ξ and ξ' are any two possible states. Note that by applying the algorithm the we generate more configurations and we therefore evolve the initial distribution. To show that the algorithm produces an ensemble of configurations that satisfies Eq. (410) we need to show that the any initial distribution $N(\xi)/N$ evolves towards the distribution $P(\xi) =$ and once such a distribution is reached it remains at *equilibrium*.
According to step (2), for any pair of states ξ and ξ', the number of configurations generated in state ξ' by applying the algorithm to the N(ξ) configurations in state ξ is $N(\xi)P_{\xi,\xi'}$, where $P_{\xi,\xi'}$ is the probability of accepting the trial configuration ξ' when $\xi_n = \xi$. In addition, the number of configurations generated in state ξ' by applying the algorithm to the N(ξ') configurations in state ξ' is (1-$P_{\xi',\xi}$) N(ξ'). Therefore, the total number $\overline{N}(\xi')$ of configurations generated in state ξ' due to any other state ξ is

$$\overline{N}(\xi') = N(\xi') + \Delta N(\xi'), \tag{412}$$

where

$$\Delta N(\xi') = N(\xi)P_{\xi,\xi'} - N(\xi')P_{\xi',\xi}, \tag{413}$$

is the net change in the number of configurations in state ξ', relative to $N(\xi')$.
According to Eqs. (411) and (413),

$$\Delta N(\xi') = N(\xi) - N(\xi')\frac{P(\xi)}{P(\xi')}, \tag{414}$$

when $P(\xi') > P(\xi)$ and

$$\Delta N(\xi') = N(\xi)\frac{P(\xi')}{P(\xi)} - N(\xi'), \tag{415}$$

when $P(\xi') < P(\xi)$. Therefore, according to Eqs. (414) and (415), $\Delta N(\xi') = 0$ when $N(\xi)/N = P(\xi)$ and $N(\xi')/N = P(\xi')$, i.e., *the algorithm does not alter the relative population of the states when the ensemble distribution is equal to the equilibrium distribution*. In addition, Eqs. (414) and (415) indicate that $\Delta N(\xi') > 0$ when $N(\xi')/N < P(\xi')$ (and $\Delta N(\xi') < 0$ when $N(\xi')/N > P(\xi')$), i.e., *the algorithm evolves any arbitrary distribution towards the equilibrium distribution* where $\frac{N_\xi}{N} = P(\xi)$.
Note: The most important aspect of the method is that the algorithm is able generate an ensemble of configurations with the probability distribution P(ξ) = Z^{-1} exp[-β E(ξ)], simply by computing the probability *ratios* $P(\xi')/P(\xi)$. Therefore, the method *avoids the need of computing the canonical partition function of the system* Z, a computational task that would be computationally intractable for most real applications. This numerical technique is thus extremely useful since it allows one to compute any canonical ensembles *without having to compute the canonical partition function of the system* as follows,

$$\langle A \rangle \approx \bar{A} = \frac{1}{N}\sum_{\xi} N_\xi A(\xi), \tag{416}$$

where $A(\xi)$ is the value of the observable A for state ξ and $\bar{A}$ is the Monte Carlo *estimator* of $\langle A \rangle$ associated with the finite number of configurations N.

Exercise:
Implement the Metropolis Monte Carlo Algorithm to generate an ensemble of configurations for a 2-dimensional Ising model with (20 $\times$ 20 spins) in the absence of an external field. Compute the average value of the magnetization at various different temperatures and show that the system exhibits spontaneous magnetization when $T < 2.3J/k$, where J is the coupling constant between spins.

37 Variance Reducing Techniques

The goal of this section is to introduce a few techniques commonly used for reducing the statistical error in Monte Carlo computations of ensemble averages.
According to the previous section, the Monte Carlo computation of the ensemble average

$$\langle A \rangle = Z^{-1} \int d\xi A(\xi) e^{-\beta E(\xi)}, \tag{417}$$

entails sampling an ensemble of random configurations ξ with probability distribution $P(\xi) = Z^{-1}\exp[-\beta E(\xi)]$, computing $A(\xi)$ for each configuration and finally averaging all of these values to obtained the unbiased estimator $\bar{A}$ introduced by Eq. (416). The convergence rate of such computation is determined by the *central limit theorem* (CLT) (see, e.g., K.L. Chung *A course in Probability Theory*, Academic Press, New York, 1974).
The CLT states that given a sequence of random variables $A(\xi_1), A(\xi_2), A(\xi_3), ...A(\xi_N)$ with expectation $\langle A \rangle$ and variance

$$\sigma^2 = \frac{1}{N} \sum_{\xi} N(\xi)(A(\xi) - \langle A \rangle)^2, \tag{418}$$

then the distribution of averages $\bar{A}$ obtained with different sequences of random variables tends to be a Gaussian distribution

$$G(\bar{A}) = \frac{1}{\sqrt{2\pi}\varepsilon} e^{-\frac{(\bar{A}-\langle A \rangle)^2}{2\varepsilon^2}}, \tag{419}$$

where

$$\varepsilon = \sigma / \sqrt{N}, \tag{420}$$

regardless of the dimensionality of the integral introduced by Eq. (417) and the nature of the probability function used to generate the sequences of random variables $A(\xi_1), A(\xi_2), A(\xi_3), ...A(\xi_N)$. The standard deviation ε of the distribution of the average is the *standard error* of the Monte Carlo computation. Therefore, results are reported as follows

$$\langle A \rangle = \bar{A} \pm \varepsilon. \tag{421}$$

Note that according to the definitions of the variance and the standard error, introduced by Eqs. (418) and (420), respectively, the standard error is large whenever the random variables $A(\xi_j)$ spread over a wide range of values. This is one of the main problems in calculations of high dimensional integrals, since the integrand $A(\xi)$ usually spreads over a very large range of values and the variance σ^2 is thus formidably large. In addition, depending on the observable of interest, the Boltzmann distribution might not sample the configurations of the system that contribute with the most to the ensemble average. These difficulties are sometimes overcome by implementing *variance reduction techniques* such as *importance sampling*, *correlated sampling*, *stratified sampling*, *adaptive sampling*, *control variates* and *umbrella sampling*. J.M. Hammersley and D.C. Handscomb *Monte Carlo Methods*, Chapter 5, John Wiley & Sons Inc., London, (1964) and J.S. Liu *Monte Carlo Strategies in Scientific Computing*, Chapter 2, Springer New York (2001) are recommended references for these methods. Here we limit our presentation to a concise description of some of them.

37.1 Importance Sampling

The *importance sampling* technique concentrates the distribution of sampled configurations in the parts of the integration range that are of most importance. Instead of computing the ensemble average

$$\langle A \rangle = \int d\xi P(\xi) A(\xi), \tag{422}$$

according to the estimator $\bar{A}$ introduced by Eq. (416), after sampling configurations ξ according to the probability distribution $P(\xi)$, configurations are sampled according to a different probability distribution $\tilde{P}(\xi)$ and the ensemble average is computed according to the estimator

$$\langle A \rangle \approx \overline{\frac{g}{\tilde{P}}} \equiv \frac{1}{N}\sum_{\xi} N(\xi)\frac{g(\xi)}{\tilde{P}(\xi)}, \tag{423}$$

where $g(\xi) \equiv P(\xi)A(\xi)$ and $\tilde{P}(\xi)$ is assumed to be normalized.
The variance of the estimator introduced by Eq. (423) is

$$\sigma^2 = \frac{1}{N}\sum_{\xi} N(\xi)\left(\frac{g(\xi)}{\tilde{P}(\xi)} - < A >\right)^2, \tag{424}$$

or

$$\sigma^2 = \frac{1}{N}\sum_{\xi} N(\xi)\frac{g(\xi)^2}{\tilde{P}(\xi)^2} - \left(\frac{1}{N}\sum_{\xi} N(\xi)\frac{g(\xi)}{\tilde{P}(\xi)}\right)^2. \tag{425}$$

Note that according to Eq. (425), $\sigma^2 = 0$, when $\tilde{P}(\xi) = g(\xi)$. Therefore, the variance can be reduced by choosing $\tilde{P}(\xi)$ similar to $|g(\xi)|$. Such choice of $\tilde{P}(\xi)$ concentrates the distribution of sampled configurations in the parts of the integration range that are of most importance. According to such distribution, the random variables $g(\xi)/\tilde{P}(\xi)$ spread over a modest range of values close to 1 and therefore the standard error of the Monte Carlo calculation is reduced.

The *umbrella sampling* technique is a particular form of importance sampling, specially designed to investigate rare events. Configurations are sampled according to the non-Boltzmann distribution P(ξ) $\propto$ exp[-β (E(ξ)+W(ξ))], where $W(\xi)$ is zero for the interesting class of configurations that defined the rare event and very large for all others.

37.2 Correlated Sampling

Consider the task of computing the integral

$$\Delta I = I_1 - I_2, \tag{426}$$

with

$$I_1 = \int dx g_1(x) f_1(x), \tag{427}$$

and

$$I_2 = \int dx g_2(x) f_2(x). \tag{428}$$

The procedure for correlated sampling can be described as follows:
Step (1). Sample random configurations $x_1, ..., x_N$ by using the sampling function $f_1(x)$ and evaluate the function g_1 for each of these configurations to obtain $g_1(x_1)$, $g_1(x_2)$, $g_1(x_3)$ ... $g_1(x_N)$. In addition, sample random configurations $y_1, ..., y_N$ by using the sampling function $f_2(y)$ and evaluate the function g_2 for each of these configurations to obtain $g_2(y_1)$, $g_2(y_2)$, $g_2(y_3)$ ... $g_2(y_N)$.
Step (2) Estimate ΔI according to

$$\Delta I = \frac{1}{N}\sum_{j=1}^{N} g_1(x_j) - g_2(y_j). \tag{429}$$

The variance of ΔI is

$$\sigma^2 = \frac{1}{N}\sum_{j=1}^{N}\left(g_1(x_j) - g_2(y_j) - (I_1 - I_2)\right)^2, \tag{430}$$

or

$$\sigma^2 = \frac{1}{N}\sum_{j=1}^{N}\left(g_1(x_j) - I_1\right)^2 + \frac{1}{N}\sum_{j=1}^{N}\left(g_2(y_j) - I_1\right)^2 - 2\frac{1}{N}\sum_{j=1}^{N}\left(g_1(x_j) - I_1\right)\left(g_2(y_j) - I_2\right), \tag{431}$$

where the first two terms on the r.h.s. of Eq. (431) are the variances σ_1^2 and σ_2^2 of the random variables g_1 and g_2, respectively, and the third term is the covariance $\text{cov}(g_1, g_2)$ of the two random variables. Note that when x_j and y_j are *statistically independent*, the $\text{cov}(g_1, g_2) = 0$ and

$$\sigma^2 = \sigma_1^2 + \sigma_2^2. \tag{432}$$

However, if the random variables are *positively correlated* then the $\text{cov}(g_1, g_2) > 0$ and the variance σ^2 is reduced. The key to reduce the variance is thus to ensure positive correlation between g_1 and g_2. This could be achieved by using *the same sequence of random numbers* for sampling both sets of random configurations x_j and y_j.

37.3 Control Variates

Consider the Monte Carlo computation of a multidimensional integral (e.g., an ensemble average),

$$\langle A \rangle = \int d\xi g(\xi), \tag{433}$$

and assume that the integral

$$\int d\xi \tilde{g}(\xi) = A_0, \tag{434}$$

can be analytically computed for an approximate expression of the integrand $\tilde{g}(\xi) \approx g(\xi)$. The function $\tilde{g}(\xi)$ is called the *control variate* for $g(\xi)$.
The control variates method is an approach that exploits the information provided by Eq. (434) to reduce the variance of the Monte Carlo computation. The integral, introduced by Eq. (433), is written in two parts,

$$\langle A \rangle = A_0 + \int d\xi (g(\xi) - \tilde{g}(\xi)), \tag{435}$$

where the first term on the r.h.s. of Eq. (435) is analytically computed and the second term is computed by correlated sampling Monte Carlo integration. Note that since $\tilde{g}(\xi)$ mimcs $g(\xi)$ and usually absorbs most of its variation, the error in the Monte Carlo computation of the second term in the r.h.s. of Eq. (435) is usually appreciably smaller than those of a Monte Carlo evaluation of the integral introduced by Eq. (433).

37.4 Stratified Sampling

Consider the task of computing the multidimensional integral (e.g., an ensemble average),

$$\langle A \rangle = \int d\xi f(\xi). \tag{436}$$

The stratified sampling technique breaks the integration range into the union of k disjoint subregions $D_1, D_2, ..., D_k$, so that within each subregion the integrand is relatively constant. Then, we can sample m_j random configurations $\xi_j(1), \xi_j(2), ..., \xi_j(m_j)$ in the subregion D_j and approximate each subregional integral by

$$\int_{D_j} d\xi f(\xi) \approx A_j = \frac{1}{m_j}[f(\xi_j(1)) + f(\xi_j(2)) + ... + f(\xi_j(m_j))]. \tag{437}$$

The overall integral is computed as

$$\langle A \rangle \approx \bar{A} = A_1 + A_2 + ... + A_k, \tag{438}$$

whose variance is

$$\sigma^2 = \frac{\sigma_1^2}{m_1} + \frac{\sigma_2^2}{m_2} + ... + \frac{\sigma_k^2}{m_k}, \tag{439}$$

where σ_j^2 indicates the variation of the integrand in the subregion D_j. Note that only when the integrand is relatively constant within each subregion the variance introduced by Eq. (439) will be smaller than the variance of the estimator obtained by using a single region for the whole integration range, $\tilde{\sigma}^2/m$ where $m = m_1 + m_2 + ... + m_k$ and $\tilde{\sigma}$ is the overall variation of the integrand in the whole integration range.
If we look carefully we can see that the stratified sampling technique described in this section is a particular version of the importance sampling method.

37.5 Simulated Annealing

Consider the task of computing a thermodynamic ensemble average for a system with many degrees of freedom at low temperature (e.g., a large cluster, a polymer, or a protein). The challenge presented by these many-body systems is that in addition to their global minumum energy configuration they usually have many local energy minima separated by high energy barriers. A Metropolis Monte Carlo computation at low temperature that starts from a configuration that is far from the minimum energy geometry usually leads to erroneous results. The reason for this is that the configurations that make the most important contributions to an ensemble average are those that are close to the minimum energy configuration and the algorithm is inefficient at sampling configurations that are beyond high potential energy barriers. Reliable Monte Carlo calculations thus require obtaining first the minimum energy configuration of the system.
The *simulated annealing* algorithm (by S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi *Science* **220** 671-680, 1983) is an efficient technique to find the minimum energy configuration of the system. The algorithm can be described as follows:
Step (1): Initialize an arbitrary configuration ξ_0 for the system at temperature T_0, where T_0 is reasonably large.
Step (2): Starting from the configuration ξ_0, sample N configurations $\xi_1, \xi_2, ..., \xi_N$ by implementing the Metropolis Monte Carlo method with $P(\xi) = Z^{-1}\exp(-E(\xi)/(kT_0))$.
Step (3): Go to (2), replacing ξ_0 by ξ_N and T_0 by a lower temperature.
Step (3) is repeated each time on the subsequent configuration ξ_N until the temperature of the system is equal to 0.
It can be shown that the configuration that corresponds to the global minimum of $E(\xi)$ can be reached according to such algorithm whenever the temperature decreases at a logarithmic rate (e.g., see S. Geman and D. Geman *IEEE Transactions on Pattern Analysis and Machine Intelligence* **6**:721-741, 1984). In practice, however, a linear or even exponential temperature decrease schedule can often be implemented.
Exercise:
This computational assignment has been designed and organized by Dr. Jose A. Gascon.

1. Write a program for implementing the simulated annealing procedure and find the minimum energy geometry of a "cluster" of two atoms interacting according to the 12-6 Lennard-Jones potential

$$U(r) = 4\varepsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right], \tag{440}$$

   where $\varepsilon = 0.0104$ ev and $\sigma = 3.4$ Å. You can write you own code or modify the Fortran program *mclj.for* attached to the www page for the lecture notes of this course.

   Notes:

   (a) If you decide to use the *mclj.for* program, you will have to edit the code and write a few lines as specified in the *mclj.for* file. The missing lines should specify the Metropolis Monte Carlo procedure.

   (b) When running the *mclj.for* program you will be asked for the initial and final temperature. A reasonable value for the initial temperature is 10 K (just type 10). Since we are trying to find the global minimum the final temperature must be zero.

   (c) When asked "Is the initial geometry random (yes/no)?", type "yes" to have the program select an initial guess of the geometry. You can eventually put your own initial guess in which case type "no". To create an initial geometry you must create a file called *initial.xyz* in the standard xyz format, where the first line is the number of atoms, the second line is left blank or with any coment, and the following lines have the atom type and coordinates as shown below for a cluster of N argon atoms.

   N
   comment
   Ar x_1 y_1 z_1
   Ar x_2 y_2 z_2
   ...
   Ar x_N y_N z_N

   (d) The *mclj.for* program reports on the screen the numbers of steps, the average interatomic distance, the energy at that step and the ratio of accepted trials out of 100.

   (e) The final geometry is recorded in the output file *final.xyz*.

2. Visualize the final geometry using a molecular viewer program such as Rasmol or Molden. The file *movie.xyz* contains snapshots of the accepted geometries along the simulation. The movie with the sequence of accepted geometries can be visualized by using Molden. Compare the minimum energy geometry (i.e., the minimum energy distance between the two Ar atoms) found by the simulated annealing algorithm with the exact value computed from the expression of the 12-6 Lennard-Jones potential.

3. To appreciate the power of the simulated annealing method, find the minimum energy geometry of clusers with 3 and 13 argon atoms and report the values of the minimum energy. For the cluster with 13 atoms run the program with three different initial temperatures, 10 K, 20 K and 30 K. Compare the final results. Do the final energy and geometry depend on the initial temperature? Why, or why not?

4. How would you compute a thermodynamic average at a constant temperature using the program for simulating annealing ?

The link (http://ursula.chem.yale.edu/∼batista/classes/vaa/LJ.m) provides a Matlab solution to the computational assignment.

38 Kinetic Monte Carlo

This section introduces Monte Carlo simulations of dynamical processes. We consider the stochastic simulation of a complex dynamical phenomenon that involves N independent processes. As an example, consider the problem of modeling the time-dependent evolution of a 2-dimensional Ising model with N spins, where spins are flipped in time due to the influence of thermal fluctuations. For simplicity, consider that the time-evolution of a spin j is described by the first order differential equation $\frac{\partial s_j(t)}{\partial t} = -k_j^{(1)}(s_j(t) + s(0))$, with $s_j(t) = s_j(0)(2exp(-k_j^{(1)}t) - 1)$. We assume that the rate constants $k_j^{(1)}$ can take different values $k_j^{(1)}$, depending on the number of nearest neighbors (1–4) of j that have the same spin as j.

Accurate MC simulations should sample event j with probability $p_j = \frac{k_j^{(1)}}{k}$ where $k = \sum_{j=1}^{N} k_j^{(1)}$ is the total rate. This can be accomplished according to the so-called *kinetic Monte Carlo* method as follows:

1. Set the simulation time to zero, t = 0.

2. Pick a random number r_1 between 0 and k.

3. Select the process labeled by index that satisfies the condition,

$$\sum_{j=1}^{\alpha-1} k_j^{(1)} \leq r_1 \leq \sum_{j=1}^{\alpha} k_j^{(1)}. \tag{441}$$

4. Carry out event α.

5. Assign the time $t_\alpha = -\frac{1}{k} ln(r_2)$ to the selected event, where r_2 is a random number between 0 and 1, and advance the simulation time $t = t + t_\alpha$.

6. Update the values of $k_j^{(1)}$ that have changed due to event α.

7. Recompute k.

8. Goto (2).

Selecting the event α, according to Eq. (441), can be pictured as throwing a dart to a ruler of length k, subdivided by segments of length $k_1^{(1)}, k_2^{(1)}, k_3^{(1)}, ..., k_N^{(1)}$, as depicted in Fig. 1. The process α corresponds to the segment targeted by the dart (α= 7, in Fig. **??**).

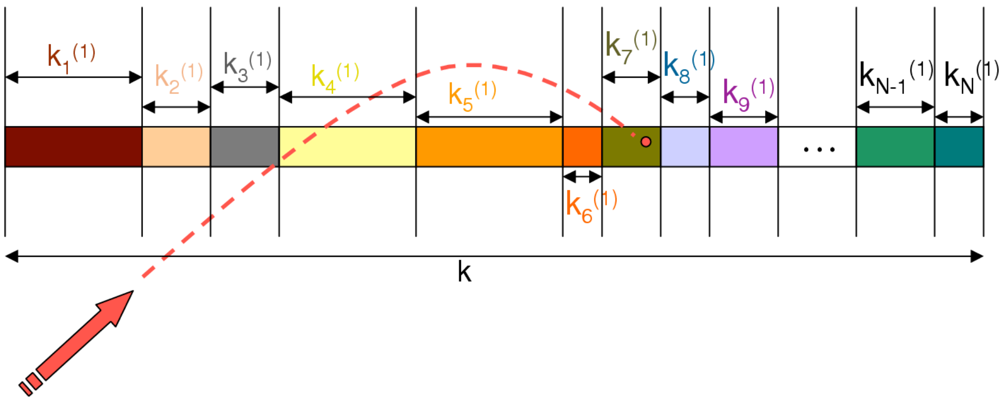


Figure 4: Dart and ruler scheme described in the text.

Unfortunately, finding out which segment was hit by the dart requires $O(N)$ operations when the search is based on a naive linear search procedure, since it is necessary to evaluate Eq. (441) with $\alpha = 1$, then with $\alpha = 2, \cdots$, etc., until finding the value of α that satisfies Eq. (441). Such a procedure is prohibitively expensive for most realistic applications. It is, therefore, necessary to implement an algorithm based on a binary tree where α can be selected in $O(log_2 N)$ steps, as described later in this section [J.L. Blue; I. Beichl, Phys. Rev. E (1995) 51, R867-R868].

Binary tree method: To explain the binary tree method, consider a system with only 8 possible events, where rate constants $k_1^{(1)}, \cdots, k_8^{(1)}$ define the roots of the tree depicted in Fig. (**??**). The next layer of nodes in the tree, $k_1^{(2)}, \cdots, k_4^{(2)}$, is defined by computing the partial sums of pairs of rate constants, as follows: $k_1^{(2)} = k_1^{(1)} + k_2^{(1)}, k_2(2) = k_3^{(1)} + k_4^{(1)}, \cdots, k_4^{(2)} = k_7^{(1)} + k_8^{(1)}$. The third layer of nodes, $k_1^{(3)}, \cdots, k_2^{(3)}$ is defined analogously, as follows:: $k_1^{(3)} = k_1^{(2)} + k_2^{(2)}, k_2^{(3)} = k_3^{(2)} + k_4^{(2)}$. Finally, the top of the tree is $k_1^{(4)} = k_1^{(3)} + k_2^{(3)} = k$.

To select the dynamical process α, we start from the top of the tree, where $k_1^{(4)} = k_j^{(n)}$, with $j = 1$ and $n = 4$, and we proceed as follows:

1. Generate a random number r_1.
2. If $r_1 \leq k_{2j-1}^{(n-1)} = k_1^{(3)}$, make $j = 2j - 1$. Otherwise, substract from r_1 and make $j = 2j$.
3. If $n = 1$, then $\alpha = j$. Otherwise, make $n = n - 1$ and go to (1).

Updating the binary tree, after carrying out the selected event, can also be done recursively from $n = 1$ to the top of the tree, by propagating the effects of the new rate constants.

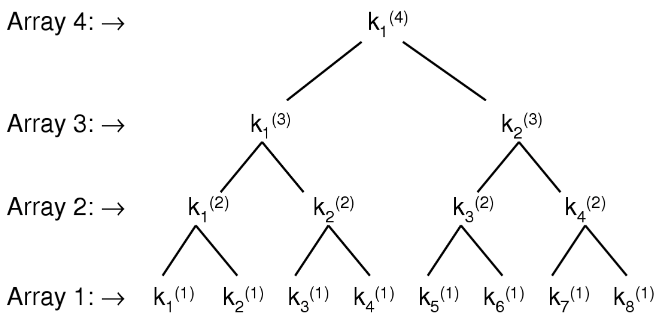


Figure 5: Binary tree explained in the text.

39 Exam 3

Exam 3 CHEM 430b/530b

Statistical Methods and Thermodynamics

Exercise 1

(**10 points**) Item (1.1): Explain the underlying approximation of Mean Field theory and illustrate it with a specific Hamiltonian.
(**20 points**) Item (1.2): Show that Mean Field theory predicts spontaneous magnetization for the 2-dimensional Ising model when $T < 4J/k$, where J is the coupling constant between spins.
(**20 points**) Item (1.3): Derive the Gibbs-Bogoliubov-Feynman inequality.
(**20 points**) Item (1.4): Derive the renormalization group equations for the 1-dimensional Ising model.

Exercise 2

(**30 points**) Compute the grand canonical partition function of a 1-dimensional lattice gas by implementing the transfer matrix approach.
Hint: Assume that the total energy for a given set of occupation numbers $\{n_j\}$ is

$$E = -\mu \sum_{j=1}^{N} n_j - \epsilon \sum_{j=1}^{N} \sum_{k} n_j n_k, \tag{442}$$

where μ is the chemical potential of the particles, the occupation numbers $n_j = 0, 1$. The indices k, in Eq. (442) label the cells that are next to cell j and N is the total number of cells.

Solution

Item (1.1):
The underlying approximation of Mean Field theory is to assume that the most important contribution to the interaction between each particle and its neighboring particles is determined by the *mean field* due to the neighboring particles. Section *Mean Field Theory* on page 65 of your lecture notes illustrates this concept as applied to the description of the 1-dimensional Ising model.

Item (1.2):
In the 2-dimensional Ising model, the average force $\overline{F_k}$ exerted on spin S_k is

$$\overline{F_k} \equiv -\overline{\frac{\partial H}{\partial S_k}} = \bar{\mu} B + J \sum_j \overline{S_j}, \tag{443}$$

where the index j includes all the nearest neighbors of spin S_k. Therefore, the average magnetic field $\overline{B}$ acting on spin S_k is

$$\overline{B} \equiv \frac{\overline{F_k}}{\bar{\mu}} = B + \Delta B, \tag{444}$$

where

$$\Delta B = J 4 \overline{S_k} / \bar{\mu}, \tag{445}$$

is the contribution to the mean field due to the nearest neighbors. Note that $\overline{S_k} = \overline{S_j}$ when all spins are identical.
The partition function, under the mean field approximation, is

$$Z \approx \sum_{S_1} \sum_{S_2} ... \sum_{S_N} e^{\beta \sum_j S_j (B + \Delta B) \bar{\mu}} = 2^N \cosh^N(\beta \bar{\mu} \overline{B}), \tag{446}$$

and the average value of S_k is

$$\overline{S_k} = \frac{1}{N}\frac{\partial \ln Z}{\partial(\beta\bar{\mu}\bar{B})} = \tanh(\beta\bar{\mu}(B + 4J\overline{S_k}/\bar{\mu})). \tag{447}$$

Note that Eq. (447) involves a transcendental equation. Its solution corresponds to the value of $\overline{S_k} = m$ for which the function on the left hand side of Eq. (447) (i.e., $\overline{S_k}$) equals the function on the right hand side of Eq. (447) (i.e., $\tanh(\beta\bar{\mu}(B + 4J\overline{S_k}/\bar{\mu}))$.
In the absence of an external magnetic field (i.e., when $B = 0$), Eq. (447) always has the trivial solution $S_k = 0$ and a non-trivial solution $S_k = m$ only when $\beta 4J > 1$.
Item (1.3):
See derivation of Eq. (390) on page 69 of your lecture notes.
Item (1.4):
See derivation of Eqs. (404) and (405) on your lecture notes.
Exercise 2:
The Hamiltonian of the system is

$$H = -\mu\sum_j n_j - \epsilon\sum_{jk} n_j n_k, \tag{448}$$

where the sum of products $n_j n_k$ defines the interaction between cells that are nearest neighbors.
The grand canonical partition function of the system is

$$\Xi = \sum_{n_1=0,1}\sum_{n_2=0,1}\cdots\sum_{n_N=0,1} e^{\beta(\mu(n_1+n_2)/2+\epsilon n_1 n_2)} e^{\beta(\mu(n_3+n_2)/2+\epsilon n_2 n_3)} \cdots e^{\beta(\mu(n_N+n_1)/2+\epsilon n_N n_1)}. \tag{449}$$

To perform a rigorous calculation of the grand canonical partition function introduced by Eq.(370), we define the *transfer function* as follows,

$$T(n_i, n_{i+1}) \equiv \exp(\beta(\mu(n_i + n_{i+1})/2 + \epsilon n_i n_{i+1})). \tag{450}$$

Substituting Eq.(371) into Eq.(370) we obtain

$$\Xi = \sum_{n_1=0,1}\sum_{n_2=0,1}\cdots\sum_{n_N=0,1} T(n_1,n_2)T(n_2,n_3)\dots T(n_N,n_1). \tag{451}$$

This expression corresponds to the trace of a product of N identical 2×2 matrices. Thus the calculation of the grand canonical partition function is reduced to that of computing the trace of the Nth power of the transfer matrix. Now, the trace of a matrix is the sum of its eigenvalues and the eigenvalues of $\mathbf{T}^N$ are $\lambda_\pm^N$, where $\lambda_\pm$ are the eigenvalues of $\mathbf{T}$ determined by the equation

$$\begin{vmatrix} e^{\beta(\mu+\epsilon)} - \lambda & e^{\beta\mu/2} \\ e^{\beta\mu/2} & 1 - \lambda \end{vmatrix} = 0, \tag{452}$$

with solutions

$$\lambda_\pm = \frac{1}{2}[e^{\beta(\mu+\epsilon)} + 1] \pm \sqrt{[e^{\beta(\mu+\epsilon)} - 1]^2/4 + e^{\beta\mu}}. \tag{453}$$

Hence, the partition function is simply,

$$Z = \lambda_+^N + \lambda_-^N, \tag{454}$$

where $\lambda_\pm$ are defined by Eq. (453).

40 Classical Fluids

To perform statistical mechanical calculations for classical fluids, we must be able to compute phase space averages,

$$\langle A \rangle = \sum_{\xi} A(\xi) \frac{e^{-\beta E(\xi)}}{Z}, \tag{455}$$

where $A(\xi)$ is the observable of interest evaluated in state ξ.
In a classical system of particles, the index ξ labels the microscopic phase point

$$\xi = (r_1, r_2, ..., r_n, p_1, p_2, ..., p_n) = (r^N, p^n), \tag{456}$$

i.e., the list of coordinates and momenta (r^N, p^n). Therefore, the ensemble average is

$$\langle A \rangle = \int dr^N \int dp^N f(r^N, p^N) A(r^N, p^N), \tag{457}$$

where

$$f(r^N, p^N) = \frac{e^{-\beta E(r^N, p^N)}}{\int dr^N \int dp^N e^{-\beta E(r^N, p^N)}}, \tag{458}$$

is the probability density at phase-point (r^N, p^N), where the system has total energy

$$E(r^N, p^N) = V(r^N) + K(p^N), \tag{459}$$

with total kinetic energy

$$K(p^N) = \sum_{j=1}^{N} \frac{p_j^2}{2m_j}, \tag{460}$$

and total potential energy of interaction between particles $V(r^N)$.
The factorization of the total energy into kinetic and potential energy terms, allows one to factorize the probability distribution function as follows,

$$f(r^N, \rho^N) = \Phi(r^N) P(\rho^N), \tag{461}$$

where

$$P(p^N) = exp(-\beta K(p^N)) / \int dp^N exp(-\beta K(p^N)), \tag{462}$$

and

$$\Phi(r^N) = exp(-\beta V(r^N)) / \int dr^N exp(-\beta V(r^N)). \tag{463}$$

Further factorization of the momentum distribution is possible, since the total kinetic energy of the system $K(p^N)$ is the sum of the kinetic energy of individual particles

$$\Phi(p^N) = \prod_{j=1}^{N} \phi(p_j), \quad \text{where} \quad \phi(p_j) = \frac{e^{-\beta p_j^2/2m_j}}{\int dp_j e^{-\beta p_j^2/2m_j}}. \tag{464}$$

Eqs. (463) and (464) thus define the probability distribution functions that are necessary to sample an ensemble of configurations of the fluid at thermal equilibrium.
Contrary to *clusters* of atoms or molecules, fluids with bulk properties usually involve an enormous number of particles $(i.e., \sim O(10^{23})$ particles where the fraction of particles near the walls of the container is negligibly small. However, computer memory and disk space requirements usually limit the capabilities of

computer simulations to model sytems with less than 10^6–10^7 particles. Contrary to macroscopic systems, these model systems have a significant fraction of particles near the walls of the container. Therefore, to simulate systems with bulk properties, it is necessary to implement a trick called *periodic boundary conditions* (p.b.c.) that minimizes surface effects.
Periodic Boundary Conditions: The implementation of p.b.c. for a one-dimensional "box" of length L converts the segment L into a circle of radius r, where $2\pi r = L$. The distance between any two particles is measured along the arc of such circle and hence the maximum separation between any two particles is L/2. One simple way of implementing p.b.c. for particles in a 1-dimensional box of length L is to add(subtract) L to the coordinate of any particle that crosses the left(right) boundary of the box after being slightly displaced to a new position. This can be implemented with an if-statement as follows:

```
if(x > L)
    x = x − L
else if(x < 0)                (465)
    x = x + L
end if
```

To compute the minimum distance dx between particles p1 and p2 at x1 and x2, respectively, we can write

```
dx = x1 − x2
if(dx > 0.5 ∗ L)
    dx = dx − L
else if(dx < −0.5 ∗ L)        (466)
    dx = dx + L
end if
```

Note that when programing Eqs. (465) and (466) you will need to implement the logic characters and the punctuation marks required by the specific programing language of choice. The generalization of Eqs. (465) and (466) to the implementation of p.b.c. in two and three dimensions is straightforward.

40.1 Radial Distribution Function

The goal of this section is to introduce the *radial distribution function* $g(\mathbf{r})$ (also known as *pair correlation function*), a central quantity in studies of fluids since it determines the average density of particles at a coordinate $\mathbf{r}$ relative to any particle in the fluid.
The radial distribution function is defined as follows

$$g(\mathbf{r}) = \rho^{2/N}(0,\mathbf{r})/\rho^2, \tag{467}$$

where $\rho = N/V$ is the density of a fluid of N particles in a container of volume V and $\rho^{2/N}(0,\mathbf{r})$ is the probability that a particle is at $\mathbf{r}$ when there is another particle at the origen of coordinates.
The probability $P^{2/N}(\mathbf{R}_1,\mathbf{R}_2)$ that particle 1 is found at $\mathbf{R}_1$ when particle 2 is at $\mathbf{R}_2$, in an N particle system, is

$$P^{2/N}(\mathbf{R}_1,\mathbf{R}_2) = \frac{\int dr^N \delta(\mathbf{r}_1-\mathbf{R}_1)\delta(\mathbf{r}_2-\mathbf{R}_2)e^{-\beta V(r^N)}}{\int dr^N e^{V(r^N)}} = \frac{\int dr^{N-2} e^{-\beta V(R_1,R_2,r_3,\ldots r_N)}}{\int dr^N e^{V(r^N)}}, \tag{468}$$

and the probability $\rho^{2/N}(\mathbf{R}_1,\mathbf{R}_2)$ that *a particle* (i.e., any particle) is found at $\mathbf{R}_1$ when *another one* (i.e., any other one) is at $\mathbf{R}_2$ is

$$\rho^{2/N}(\mathbf{R}_1,\mathbf{R}_2) = \frac{N!}{(N-2)!}P^{2/N}(\mathbf{R}_1,\mathbf{R}_2) = N(N-1)P^{2/N}(\mathbf{R}_1,\mathbf{R}_2). \tag{469}$$

In particular, in a fluid where the interaction between particles can be neglected,

$$P^{2/N}(\mathbf{R}_1,\mathbf{R}_2) = \frac{1}{V^2}, \tag{470}$$

or

$$\rho^{2/N}(\mathbf{R}_1,\mathbf{R}_2) = \frac{N(N-1)}{V^2} \approx \rho^2, \tag{471}$$

and $g(\mathbf{r}) = 1$.
In an atomic fluid (e.g., liquid argon), $g(\mathbf{r}_1,\mathbf{r}_2) = g(r)$, where $r =| \mathbf{r}_1 - \mathbf{r}_2 |$, since the fluid is uniform and isotropic. The presence of an atom at the origen of coordinates excludes other particles from all distances smaller than the radius of the first coordination shell where $g(r)$ has a maximum (see figure). The presence of the first coordination shell tends to exclude particles that are closer than the radius of the second coordination shell, where $g(r)$ has another maximum. This oscillatory form for $g(r)$ persists until r is larger than the *range of correlations* between the particles. At distances larger than the correlation length $g(r) = 1$, since $\rho^{2/N}(0,r) \approx \rho^2$ for *uncorrelated particles*. Therefore, $h(r) = g(r) - 1$ describes the deviation from the asymptotic limit of uncorrelated particles (i.e., an ideal gas) and the product $\rho g(r)$ describes the *average density of particles at a distance r from any particle in the fluid.*
Note that $\rho g(r) 4\pi r^2 dr$ is the average number of particles at a distance between r and r+dr from any particle in the fluid. Therefore, the calculation of $g(r)$ involves averaging the number of particles at a distance r from any particle in the system and dividing that number by the element of volume $4\pi r^2 dr$.
Exercise:
Compute the radial distribution function g(r) for a fluid of argon atoms at constant T,N,V using the program developed in the assignment of simulating annealing.

41 Reversible Work Theorem

The theorem states that *the radial distribution function g(r) determines the reversible work w(r) associated with the process by which two particles in a fluid are moved from an infinite separation to a separation r, as follows:*

$$g(r) = \exp(-\beta w(r)). \tag{472}$$

Note that since the process is reversible at constant T, N and V, $w(r) = \Delta A$, where A is the Helmholtz free energy.

Proof:
Consider the mean force between particles 1 and 2, averaged over the equilibrium distribution of all other particles,

$$- < \frac{dV(r^N)}{dr_1} >_{r_1 r_2} = \frac{-\int \frac{dV}{dr_1} e^{-\beta V(r^N)} dr_3 ... dr_N}{\int e^{-\beta V(r^N)} dr_3 ... dr_N}, \tag{473}$$

which gives

$$- < \frac{dV(r^N)}{dr_1} >_{r_1 r_2} = \frac{1}{\beta} \frac{d}{dr_1} \ln \int e^{-\beta V(r^N)} dr_3 ... dr_N, \tag{474}$$

or

$$- < \frac{dV(r^N)}{dr_1} >_{r_1 r_2} = +\frac{1}{\beta} \frac{d}{dr_1} \ln \frac{N(N-1)}{\rho^2 Z} + \frac{1}{\beta} \frac{d}{dr_1} \ln \int e^{-\beta V(r^N)} dr_3 ... dr_N, \tag{475}$$

since both the density ρ, the number of particles N in the system and the partition function Z are constants, independent of r_1.
Eq. (475) can be simplified according to Eqs. (467) and (468) as follows,

$$-<\frac{dV(r^N)}{dr_1}>_{r_1r_2}=\frac{1}{\beta}\frac{d}{dr_1}\ln g(r_1). \tag{476}$$

Integration of the expression for the average force, introduced by Eq. (476), gives the reversible work,

$$w(r_{12})=\int_{r_{12}}^{\infty}dr_1(-<\frac{dV(r^N)}{dr_1}>)=kT\int_{r_{12}}^{\infty}dr_1\frac{d}{dr_1}\ln g(r_1), \tag{477}$$

Therefore, the reversible work $w(r_{12})$ associated with the process by which particles 1 and 2 are moved from infinite separation to a relative separation r_{12} is

$$w(r_{12})=kT\ln g(\infty)-kT\ln g(r_{12})=-kT\ln g(r_{12}), \tag{478}$$

since $g(\infty)=1$.
Finally, note that

$$w(r)=kT\ln\frac{Z(r_{12}=\infty)}{Z(r_{12}=r)}=-(A(r_{12}=\infty)-A(r_{12}=r)), \tag{479}$$

where $Z(r_{12}=r)$ and $A(r_{12}=r)$ are the canonical partition function and the Helmholtz free energy of the system, subject to the constraint of fixed relative distance $r_{12}=r$ between particles 1 and 2.

42 Thermodynamic Properties of Fluids

The goal of this section is to show that the thermodynamic properties of fluids (e.g., the internal energy, the virial coefficient, etc.) can be computed in terms of the corresponding properties of an ideal gas plus a correction term that is determined by the radial distribution function of the fluid. This is illustrated by computing the internal energy of a classical fluid.
The ensemble average of the internal energy E is

$$\langle E\rangle=\langle K(p^N)\rangle+\langle V(r^N)\rangle, \tag{480}$$

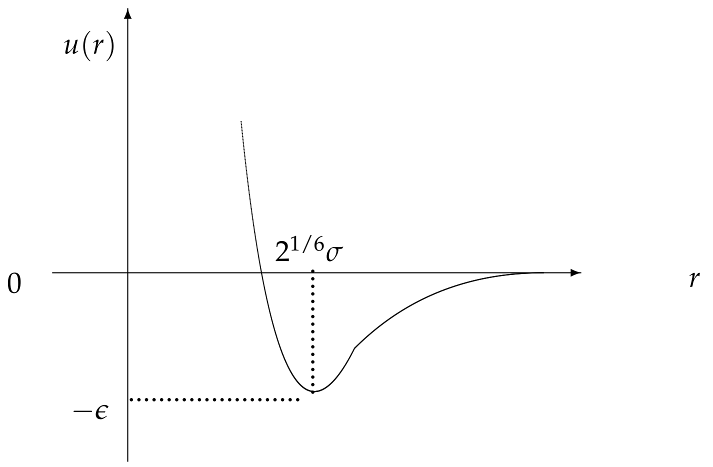
where $K(p^N)$ is the total kinetic energy. The simplest model for the potential energy $V(r^N)$ is the pairwise additive potential

$$V(r^N)=\sum_i\sum_{j<i}u(|r_i-r_j|), \tag{481}$$

where $u(r)$ is, for example, a Lennard-Jones potential

$$u(r)=4\varepsilon\left[\left(\frac{\sigma}{r}\right)^{12}-\left(\frac{\sigma}{r}\right)^{6}\right], \tag{482}$$

represented by the following diagram:



According to Eq. (481),

$$\langle E\rangle = N\langle\frac{p^2}{2m}\rangle + \sum_{j=1}^{N}\sum_{k\neq j}\frac{1}{2}\langle u(|r_j - r_k|)\rangle, \tag{483}$$

therefore,

$$\langle E\rangle = N\frac{3}{2}kT + \int dr_1 \int dr_2 u(|r_1 - r_2|)\frac{N(N-1)}{2}\frac{\int dr^{N-2}e^{-\beta V(r^N)}}{\int dr^N e^{-\beta V(r^N)}}. \tag{484}$$

Eq. (484) can be simplified, according to Eqs. (467) and (468), as follows:

$$\langle E\rangle = \frac{3}{2}NkT + \frac{1}{2}\int dr_1 \int dr_2 u(|r_1 - r_2|)\rho^{2/N}(r_1, r_2), \tag{485}$$

or

$$\langle E\rangle = \frac{3}{2}NkT + \frac{V\rho^2}{2}\int dr u(r) g(r). \tag{486}$$

Therefore, the ensemble average internal energy per particle

$$\frac{\langle E\rangle}{N} = \frac{3}{2}kT + \frac{\rho}{2}\int dr u(r) g(r), \tag{487}$$

is the sum of the internal energy per particle in an ideal gas (i.e., 3/2 k T) plus a correction term that can be obtained in terms of the radial distribution function g(r).

43 Solvation Free Energy: Thermodynamic Integration

The goal of this section is to show how to compute the *free energy of solvation* of structureless solute particles A dissolved in a solvent S. The solvation free energy is computed according to the *coupling parameter method* in terms of the pair correlation function $g(r_A, r_S)$, where r_A and r_S are the coordinates of the solute and solvent molecules, respectively.
Consider a solute-solvent mixture where solute particles with no internal structure interact with solvent molecules according to a pairwise additive potential V_{AS}. Assume that the solute concentration is so low that the interactions between solute particles can be neglected. The canonical partition function of the system is

$$Z_\lambda = \frac{Z_A^{(id)} Z_S^{(id)}}{V^{N_A+N_S}}\int dr^{N_A}\int dr^{N_S} e^{-\beta V_S(r^{N_S}) - \beta V_{SA}(r^{N_S}, r^{N_A})\lambda}, \tag{488}$$

where the *coupling parameter* $\lambda = 1$ and where

$$Z^{(id)} = \frac{1}{h^{3N}N!}\int dr^{3N}\int dp^{3N}e^{-\beta\frac{p^2}{2m}} = \frac{V^N}{N!h^{3N}}\int dp^{3N}e^{-\beta\frac{p^2}{2m}}. \tag{489}$$

In the absence of solute-solvent interactions (i.e., $\lambda = 0$), the energy of the system becomes factorizable into the solute and the solvent contributions to the total energy and the free energy of the system is $A^{(id)}(N_S, N_A, V, T) = -\beta^{-1}\ln Z_{\lambda=0}$. The change in free energy due to a differential change in λ is

$$-kT\frac{d\ln Z_\lambda}{d\lambda} = -kT\frac{\int dr^{N_A}\int dr^{N_S}(-\beta V_{SA}(r^{N_S}, rN_A))e^{-\beta V_S(r^{N_S})-\beta V_{SA}(r^{N_S},r^{N_A})\lambda}}{\int dr^{N_A}\int dr^{N_S}e^{-\beta V_S(r^{N_S})-\beta V_{SA}(r^{N_S},r^{N_A})\lambda}}. \tag{490}$$

and assuming that

$$V_{SA}(r^{N_S}, r^{N_A}) = \sum_{j=1}^{N_A}\sum_{k=1}^{N_S} u_{AS}(|r_j - r_k|), \tag{491}$$

we obtain

$$-kT\frac{d\ln Z_\lambda}{d\lambda} = N_A N_S\frac{\int dr^{(A_1)}\int dr^{(S_1)}u_{A_1S_1}(|r_{A_1} - r_{S_1}|)\int dr^{N_A-1}\int dr^{N_S-1}e^{-\beta V_S-\beta V_{SA}}}{\int dr^{N_A}\int dr^{N_S}e^{-\beta V_S-\beta V_{SA}}}. \tag{492}$$

Introducing the pair correlation function $g_{AS} = \rho_{AS}/\rho_A\rho_B$, where ρ_{AS} is the joint probability of finding a solute particle at r_{A_1} and a solvent molecule at r_{S_1},

$$\rho_{AS} = N_A N_S\int dr^{N_A-1}\int dr^{N_S-1}\frac{e^{-\beta V_S-\beta V_{SA}}}{\int dr^{N_A}\int dr^{N_S}e^{-\beta V_S-\beta V_{SA}}}, \tag{493}$$

we obtain

$$-kT\frac{d\ln Z_\lambda}{d\lambda} = \int dr_{A_1}\int dr_{S_1}u(|r_{A_1} - r_{S_1}|)\rho_A\rho_S g_{AS}, \tag{494}$$

or

$$-kT\frac{d\ln Z_\lambda}{d\lambda} = \rho_A\rho_S V\int dr u_{AS}(r)g_{AS}(r). \tag{495}$$

Therefore, the change in free energy due to a differential change in the coupling parameter λ is

$$-kT\frac{d\ln Z_\lambda}{d\lambda} = N_A\int dr u_{AS}(r)\rho_S g_{AS}(r), \tag{496}$$

and the total free energy of the system is

$$A(N_S, N_A, V, T) = A^{(id)}(N_S, N_A, V, T) + (\Delta A)_{solv}, \tag{497}$$

where the free energy of solvation $(\Delta A)_{solv}$ is,

$$(\Delta A)_{solv} = N_A\int_0^1 d\lambda\int dr u_{AS}(r)\rho_S g_{AS}^{(\lambda)}(r). \tag{498}$$

The approach implemented in this section, where a coupling parameter is introduced to investigate the contributions of specific interactions to an ensemble average, is called *coupling parameter method*.

43.1 Zwanzig Equation

The goal of this subsection is to show that the free energy change due to a change in the coupling parameter λ can be computed according to the equation that Zwanzig derived here at Yale, *J. Chem. Phys.* **22** 1420-1426 (1954), as follows:

$$A(\lambda_2) - A(\lambda_1) = -\beta^{-1}\ln\langle e^{-\beta[E(\lambda_2)-E(\lambda_1)]}\rangle_\beta, \tag{499}$$

where $\langle \cdots \rangle_\beta$ indicates the ensemble average at temperature $T = \beta^{-1}/k$.

Considering that

$$A(\lambda,\beta) = -\beta^{-1}\ln Z(\lambda,\beta), \tag{500}$$

where $Z(\lambda,\beta) = \sum_j exp(-\beta E_j(\lambda))$, we obtain:

$$\begin{aligned}\frac{Z(\lambda_2,\beta_2)}{Z(\lambda_1,\beta_1)} &= \frac{1}{Z(\lambda_1,\beta_1)}\sum_j e^{-\left[\beta_2 E_j(\lambda_2)-\beta_1 E_j(\lambda_1)\right]}e^{-\beta_1 E_j(\lambda_1)},\\ &= \langle e^{-[\beta_2 E(\lambda_2)-\beta_1 E(\lambda_1)]}\rangle_{\beta_1}.\end{aligned} \tag{501}$$

From (501), we obtain:

$$-\left[\beta_2 A(\lambda_2,\beta_2) - \beta_1 A(\lambda_1,\beta_1)\right] = \ln\langle e^{-[\beta_2 E(\lambda_2)-\beta_1 E(\lambda_1)]}\rangle_{\beta_1} \tag{502}$$

When $\beta_1 = \beta_2 = \beta$, Eq. (502) is identical to Eq. (499).

In addition, when $\lambda_1 = \lambda_2$, we obtain that the change in free energy per unit temperature due to a change in temperature from $T_1 = \beta_1^{-1}/k$ to $T_2 = \beta_2^{-1}/k$ is

$$\beta_2 A(\lambda,\beta_2) - \beta_1 A(\lambda,\beta_1) = -\ln\langle e^{-[\beta_2 E(\lambda)-\beta_1 E(\lambda)]}\rangle_{\beta_1} \tag{503}$$

which is computed as the ensemble average of $e^{-[\beta_2-\beta_1]E}$ at the temperature defined by β_1, as indicated in Problem 6.6 of the textbook.

Other properties could also be computed by thermodynamic integration, analogously, as implemented above for free energy calculations. As an example, consider the ensemble average of property $\hat{N}$:

$$\begin{aligned}\langle N(\lambda_1,\beta_1)\rangle &= \frac{\sum_j N_j e^{-\beta_1 E(\lambda_1)}}{\sum_j e^{-\beta_1 E(\lambda_1)}}\\ &= \frac{\sum_j e^{-\beta_2 E(\lambda_2)}}{\sum_j e^{-\beta_1 E(\lambda_1)}}\frac{\sum_j N_j e^{-\beta_1 E(\lambda_1)+\beta_2 E(\lambda_2)}e^{-\beta_2 E(\lambda_2)}}{\sum_j e^{-\beta_2 E(\lambda_2)}}\end{aligned} \tag{504}$$

where

$$\frac{\sum_j e^{-\beta_1 E(\lambda_1)}}{\sum_j e^{-\beta_2 E(\lambda_2)}} = \frac{\sum_j e^{-\beta_1 E(\lambda_1)+\beta_2 E(\lambda_2)}e^{-\beta_2 E(\lambda_2)}}{\sum_j e^{-\beta_2 E(\lambda_2)}} \tag{505}$$

Therefore,

$$\langle N(\lambda_1,\beta_1)\rangle = \frac{\sum_j p_j(\lambda_2,\beta_2)N_j e^{-\beta_1 E(\lambda_1)+\beta_2 E(\lambda_2)}}{\sum_j p_j(\lambda_2,\beta_2)e^{-\beta_1 E(\lambda_1)+\beta_2 E(\lambda_2)}} \tag{506}$$

44 Quiz 3

Quiz 3 CHEM 430b/530b
Statistical Methods and Thermodynamics

Metropolis Monte Carlo
(**25 points**) Item 1a: Describe the implementation of a Metropolis Monte Carlo algorithm to generate an ensemble of configurations with probability distribution $P(\xi)$.
(**25 points**) Item 1b: Prove that the Metropolis Monte Carlo algorithm described in item (1a) evolves any arbitrary distribution of configurations toward the equilibrium distribution $P(\xi)$.
Classical Fluids
(**25 points**) Item 2a: Explain how to compute the radial distribution function g(r) of liquid Ar, after having generated an ensemble of configurations of the system at thermal equilibrium.
(**25 points**) Item 2b: Derive an expression for the internal energy of liquid argon in terms of the radial distribution function g(r).

Solution:
Item 1a:
Read the description of the Metropolis Monte Carlo algorithm on page 74 of the lecture notes.
Item 1b:
Read the proof of the Metropolis Monte Carlo algorithm described on pages 75 and 76 of the lecture notes.
Item 2a:
The number of particles at a distance between r and r+dr from any particle in the fluid is

$$N(r) = \rho g(r) 4\pi r^2 dr. \tag{507}$$

Therefore, the calculation of $g(r)$ involves averaging the number of particles at a distance between r and $r + dr$ from any particle in the system and dividing that number by the element of volume $4\pi r^2 dr$.
Item 2b:
See derivation of Eq. (487) of the lecture notes.

45 Lars Onsager's Regression Hypothesis

The goal of this section is to introduce Lars Onsager's *regression hypothesis*, a consequence of the *fluctuation-dissipation theorem* proved by Callen and Welton in 1951. Here, we derive the regression hypothesis from the principles of statistical mechanics.
The regression hypothesis states that *the regression of microscopic thermal fluctuations at equilibrium follows the macroscopic law of relaxation of small non-equilibrium disturbances* (L. Onsager, *Phys. Rev.* **37**, 405 (1931); **38**, 2265 (1931)).
To understand this hypothesis, consider an observable A for a system at thermal equilibrium. Such property fluctuates in time with *spontaneous microscopic fluctuations*

$$\delta A(t) = A(t) - \langle A \rangle. \tag{508}$$

Here, $A(t)$ is the instantaneous value of the observable and $\langle A\rangle$ is the equilibrium ensemble average

$$\langle A\rangle = \frac{\mathrm{Tr}\{A(t)e^{-\beta H_0}\}}{\mathrm{Tr}\{e^{-\beta H_0}\}}. \tag{509}$$

The average correlation between $\delta A(t)$ and an instantaneous fluctuation at time zero $\delta A(0)$ is described by the correlation function

$$C(t) = \langle \delta A(t)\delta A(0)\rangle = \langle A(t)A(0)\rangle - \langle A\rangle^2, \tag{510}$$

where $\langle\rangle$ represents the *equilibrium ensemble average*. Note that at small times, instantaneous fluctuations are correlated and therefore,

$$\lim_{t\to 0} C(t) = \langle(\delta A(0))^2\rangle, \tag{511}$$

has a finite value. However, at large times $C(t)$ vanishes, i.e.,

$$\lim_{t\to\infty} C(t) = 0, \tag{512}$$

since $\delta A(t)$ becomes uncorrelated to $\delta A(0)$. This decay of correlations is the *regression of microscopic thermal fluctuations* referred to in Onsager's hypothesis. Therefore, Onsager's regression hypothesis can be formulated as follows

$$\frac{\overline{A}(t) - \langle A\rangle}{\overline{A}(0) - \langle A\rangle} = \frac{\langle \delta A(t)\delta A(0)\rangle}{\langle(\delta A(0))^2\rangle}, \tag{513}$$

where $\overline{A}(t)$ describes the *macroscopic relaxation* of the observable towards its equilibrium value $\langle A\rangle$, while the system evolves from an initial state that is not far from equilibrium to its final state in equilibrium with a thermal reservoir.
To derive Eq.(513) from the ensemble postulates of statistical mechanics, consider preparing the system in a state that is not far from equilibrium by applying to the system a weak perturbational field f (e.g., an electric field) that couples to the dynamical variable $A(t)$ (e.g., the instantaneous total dipole moment). The external field is assumed to be so weak that the perturbation Hamiltonian ΔH, written as an expansion in powers of f, can be approximated to be first order in the field as follows,

$$\Delta H = -fA(0). \tag{514}$$

Assume that such perturbational field was applied until the system equilibrated according to the total Hamiltonian

$$H = H_0 + \Delta H, \tag{515}$$

The macroscopic relaxation of the system is analyzed by switching off the external perturbational field and computing the evolution of the non-equilibrium ensemble average

$$\overline{A}(t) = \frac{\mathrm{Tr}\{A(t)e^{-\beta(H_0+\Delta H)}\}}{\mathrm{Tr}\{e^{-\beta(H_0+\Delta H)}\}}, \tag{516}$$

as the system evolves towards equilibrium. Since the perturbation ΔH is small, we can introduce the approximation

$$e^{-\beta(H_0+\Delta H)} \approx e^{-\beta H_0}(1-\beta\Delta H). \tag{517}$$

Substituting Eq. (517) into Eq. (516), we obtain

$$\overline{A}(t) \approx \frac{\mathrm{Tr}\{e^{-\beta H_0}(1-\beta\Delta H+...)A(t)\}}{\mathrm{Tr}\{e^{-\beta H_0}(1-\beta\Delta H+...)\}} = \frac{\mathrm{Tr}\{e^{-\beta H_0}A(t)\} - \beta\mathrm{Tr}\{e^{-\beta H_0}\Delta H A(t)\}}{\mathrm{Tr}\{e^{-\beta H_0}\} - \beta\mathrm{Tr}\{e^{-\beta H_0}\Delta H\}}, \tag{518}$$

or

$$\overline{A}(t) \approx \frac{\text{Tr}\{e^{-\beta H_0} A(t)\}}{\text{Tr}\{e^{-\beta H_0}\}(1-\beta\frac{\text{Tr}\{e^{-\beta H_0}\Delta H\}}{\text{Tr}\{e^{-\beta H_0}\}})} - \beta\frac{\text{Tr}\{e^{-\beta H_0}\Delta H A(t)\}}{\text{Tr}\{e^{-\beta H_0}\}(1-\beta\langle\Delta H\rangle)}, \tag{519}$$

which gives

$$\overline{A}(t) \approx \frac{\langle A(t)\rangle}{1-\beta\langle\Delta H\rangle} - \beta\frac{\langle\Delta H A(t)\rangle}{1-\beta\langle\Delta H\rangle}. \tag{520}$$

Introducing the approximation

$$1/(1-x) \approx 1+x, \tag{521}$$

for small x, in Eq. (520), we obtain

$$\overline{A}(t) \approx \langle A\rangle - \beta\langle\Delta H A(t)\rangle + \beta\langle\Delta H\rangle(\langle A\rangle - \beta\langle\Delta H A(t)\rangle). \tag{522}$$

Therefore, according to Eq. (522),

$$\overline{A}(t) \approx \langle A\rangle - \beta(\langle\Delta H A(t)\rangle - \langle\Delta H\rangle\langle A\rangle) + O((\Delta H)^2). \tag{523}$$

Substituting Eq. (514) into Eq. (523) and keeping only the terms that are first order in ΔH we obtain,

$$\overline{A}(0) - \langle A\rangle \approx f\beta(\langle A(0)^2\rangle - \langle A(0)\rangle^2), \tag{524}$$

and

$$\overline{A}(t) - \langle A\rangle \approx f\beta(\langle A(0)A(t)\rangle - \langle A(0)\rangle\langle A(t)\rangle). \tag{525}$$

Finally, Eqs. (524) and (525) complete the derivation of Eq. (512) in terms of the ensemble average postulates of statistical mechanics, since according to Eq. (508),

$$\langle\delta A(t)\delta A(0)\rangle = \langle A(t)A(0)\rangle - \langle A\rangle^2. \tag{526}$$

45.1 Response Function: Generalized Susceptibility

The goal of this section is to introduce the concept of *response function* $\chi(t,t')$, or *generalized susceptibility*, for the description of non-equilibrium disturbances.
According to Eqs. (525) and (526), the response to a perturbation ΔH in the linear regime (i.e., a perturbation that is linear in the field f) is

$$\Delta A(t,f) \approx f\beta\langle\delta A(t)\delta A(0)\rangle, \tag{527}$$

where $\Delta A(t,f) = \overline{A}(t) - \langle A\rangle$. Therefore,

$$\Delta A(t,\lambda f) = \lambda\Delta A(t,f). \tag{528}$$

The most general expression for a macroscopic response that is consistent with Eq. (528) is

$$\Delta A(t,f) = \int_{-\infty}^{\infty} dt'\chi(t,t')f(t') = \sum_j \chi(t,t_j)f(t_j)dt_j, \tag{529}$$

where $dt_j = (t_{j+1} - t_j)$ is the time increment, $f(t)$ is the external perturbational field and $\chi(t,t')$ is the *response function*. Note that the name *response function* is due to the fact that $\chi(t,t_0)$ is equal to the response at time t when the perturbational field is an impulse at time t_0 (i.e., $\chi(t,t_0) = \Delta A(t,\delta(t-t_0))$). In addition, any other perturbational field can always be written as a linear superposition of impulses since any function can be expanded as a linear combination of delta functions.

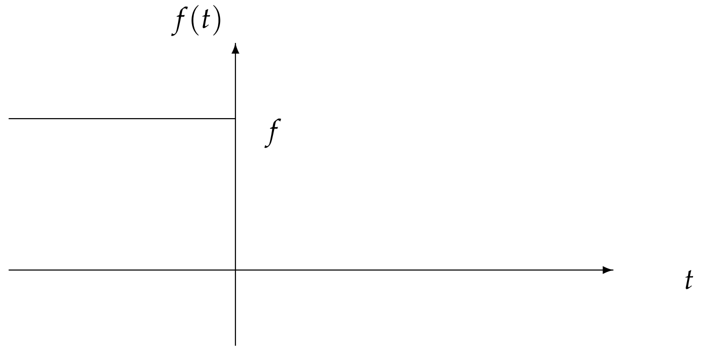
Another important observation is that according to Eq. (529),

$$\chi(t,t_j) = \frac{\partial \Delta \bar{A}(t)}{\partial f(t_j)}, \tag{530}$$

Therefore, $\chi(t,t_j)$ defines the first nonvanishing term in an expansion of $\Delta A(t,f)$ in powers of $f(t)$. The response function $\chi(t,t')$ is thus also called *generalized susceptibility*, since the electric susceptibility χ_e defines the first nonvanishing term in an expansion of the polarization $\vec{P}$ (i.e., the total electric dipole moment per unit volume) in powers of the electric field $\vec{E}$ as follows,

$$\vec{P} = \chi_e \vec{E}. \tag{531}$$

In analogy to the electric susceptibility χ_e, that is a property of the unperturbed system, the response function $\chi(t,t_j)$ is a function of the unperturbed system at equilibrium. To show this important aspect of the response function, consider the perturbational field represented by the following diagram:



which is defined as follows:

$$f(t) = \begin{cases} f, & \text{when } t \le 0, \\ 0, & \text{otherwise.} \end{cases} \tag{532}$$

The response at time $t > 0$ is

$$\Delta A(t) = f \int_{-\infty}^{0} dt' \chi(t-t') = -f \int_{\infty}^{t} dt'' \chi(t''), \tag{533}$$

where $t'' = t - t'$, $dt'' = -dt'$ and we have assumed that $\chi(t,t') = \chi(t-t')$. Therefore,

$$\frac{d\Delta A(t)}{dt} = -f\chi(t), \tag{534}$$

since, according to the second fundamental theorem of calculus,

$$\chi(t) = \frac{d}{dt}\int_a^t dt' \chi(t'), \tag{535}$$

where a is an arbitrary constant value. Substituting Eq. (527) into Eq. (534), we obtain

$$\chi(t) = -\beta \frac{d}{dt} < \delta A(0) \delta A(t) >, \tag{536}$$

where $t > 0$.

45.2 Linear Spectroscopy

The goal of this section is to show that the linear absorption spectrum $\sigma(\omega)$ of a system is determined by the regression of spontaneous polarization fluctuations at equilibrium as follows:

$$\begin{aligned}\sigma(\omega) &= 2\omega f_0^2 \bar{\epsilon}^2 \int_0^\infty \chi(t')\sin(\omega t')dt', \\ &= \beta 2\omega^2 f_0^2 \bar{\epsilon}^2 \int_0^\infty < \delta A(0)\delta A(t') > \cos(\omega t')dt',\end{aligned} \tag{537}$$

where $A(t)$ is the time-dependent polarization

$$A(t) = \sum_j p_j \langle \phi_j(t) \mid \hat{A} \mid \phi_j(t)\rangle, \tag{538}$$

where the functions $\phi_j(t)$ evolve according to the unperturbed system Hamiltonian H_0, as follows:

$$i\hbar \frac{\partial}{\partial t}\phi_j(t) = \hat{H}_0 \phi_j(t). \tag{539}$$

To derive Eq. (537), consider that the system is perturbed by a monochromatic electric field,

$$f(t) = f_0 \bar{\epsilon}(e^{-i\omega t} + e^{i\omega t}), \tag{540}$$

where f_0 is the amplitude of the field and $\bar{\epsilon}$ is the unit vector along the spatially uniform electric field. In the linear regime, the interaction between the system and the field is

$$\Delta\hat{H}(t) = -f(t)\hat{A}. \tag{541}$$

The total energy of the system is

$$E(t) = \sum_j p_j \langle \phi_j(t) \mid \hat{H}_0 + \Delta\hat{H}(t) \mid \phi_j(t)\rangle, \tag{542}$$

and the differential change of energy per unit time is

$$\dot{E}(t) = \sum_j p_j \langle \phi_j(t) \mid \frac{\partial \Delta\hat{H}(t)}{\partial t} \mid \phi_j(t)\rangle = -\dot{f}(t)A(t), \tag{543}$$

since,

$$\langle \frac{\partial \phi_j(t)}{\partial t} \mid \hat{H}_0 + \Delta\hat{H}(t) \mid \phi_j(t)\rangle = -\langle \phi_j(t) \mid \hat{H}_0 + \Delta\hat{H}(t) \mid \frac{\partial \phi_j(t)}{\partial t}\rangle. \tag{544}$$

Equation (544) can be proved by considering that $i\hbar|\dot{\phi}_j\rangle = \hat{H}_0|\phi_j\rangle$ and, therefore, $i\hbar\langle\phi_j|\hat{H}|\dot{\phi}_j\rangle = \langle\phi_j|\hat{H}\hat{H}_0|\phi_j\rangle$, and conjugating both sides of this equality gives $-i\hbar\langle\dot{\phi}_j|\hat{H}|\phi_j\rangle = \langle\phi_j|\hat{H}\hat{H}_0|\phi_j\rangle$ and therefore $\langle\dot{\phi}_j|\hat{H}|\phi_j\rangle = -\langle\phi_j|\hat{H}|\dot{\phi}_j\rangle$.

Therefore, the total energy change σ due to the interaction with the external field for time T is

$$\sigma = \frac{1}{T}\int_0^T dt(-\dot{f}(t))A(t), \tag{545}$$

where, according to Eq. (540),

$$\dot{f}(t) = -if_0\bar{\epsilon}\omega[e^{-i\omega t} - e^{+i\omega t}]. \tag{546}$$

Substituting Eq. (546) into Eq. (545) we obtain,

$$\sigma = -\frac{i\omega}{T}\int_0^T f_0\bar{\epsilon}[e^{+i\omega t} - e^{-i\omega t}]\left(< A > + \int_{-\infty}^{\infty} \chi(t-t')f(t')dt'\right). \tag{547}$$

Note that

$$\int_{-\infty}^{\infty} \chi(t-t')f(t')dt' = \int_{-\infty}^{\infty} \chi(-t')f(t'-t)dt' = \int_{-\infty}^{\infty} \chi(t')f(-t'-t)dt', \tag{548}$$

thus Eq. (547) becomes

$$\sigma = -\frac{i\omega}{T}\int_0^T dt f_0\bar{\epsilon}[e^{+i\omega t} - e^{-i\omega t}]\left(< A > + \int_{-\infty}^{\infty} \chi(t')f(-t'-t)dt'\right). \tag{549}$$

To simplify Eq. (549), we note that

$$\lim_{T\to\infty}\frac{1}{T}\int_0^T e^{int'}dt' = \begin{cases} 1, & \text{when n=0} \\ 0, & \text{otherwise.}\end{cases} \tag{550}$$

Therefore, Eq. (549) becomes

$$\sigma = -\frac{i\omega}{T}\int_0^T dt f_0\bar{\epsilon}[e^{+i\omega t} - e^{-i\omega t}]\int_{-\infty}^{\infty} \chi(t')f_0\bar{\epsilon}[e^{+i\omega(t+t')} + e^{-i\omega(t+t')}]dt', \tag{551}$$

or

$$\sigma = -i\omega f_0^2\bar{\epsilon}^2\int_{-\infty}^{\infty} \chi(t')[e^{+i\omega t'} - e^{-i\omega t'}]dt'. \tag{552}$$

Therefore,

$$\sigma = 2\omega f_0^2\bar{\epsilon}^2\int_{-\infty}^{\infty} \chi(t')\sin(\omega t')dt'. \tag{553}$$

Substituting Eq. (536) into Eq. (553) we obtain

$$\sigma = -\beta 2\omega f_0^2\bar{\epsilon}^2\int_0^{\infty} \frac{d}{dt} < \delta A(0)\delta A(t') > \sin(\omega t')dt'. \tag{554}$$

Finally, integrating Eq. (554) by parts we obtain Eq. (537), since Eq. (554) can be written as $\int dt'u(t')dv/dt' = u(t')v(t') - \int dt'v(t')du/dt'$, with $u(t') = \sin(\omega t')$ and $v(t') =< \delta A(0)\delta A(t') >$.

46 Langevin Equation

In previous sections we have shown how to implement the regression hypothesis to describe the dissipation of macroscopic disturbances in terms of the regression of spontaneous fluctuations. As an example, we have analyzed the relaxation of a polarization disturbance A(t)-$\langle A\rangle$, in the linear regime, in terms of the regression of spontaneous polarization fluctuations $\langle\delta(t)\delta A(0)\rangle$. The goal of this section is to describe another application of this general theoretical approach to relate the fluctuations of a physical quantity with the dissipation of a macroscopic disturbance.
Consider the motion of a particle through a medium after being initially prepared in a certain state of motion (e.g., after being pushed by an external force). As a result of friction with the medium, the particle will be slowed down (i.e., its initial kinetic energy will be dissipated by heating up the medium). The motion of such particle is described by the *generalized Langevin equation*, which is derived in this section as follows.

Consider the Hamiltonian that describes a particle with coordinates $x(t)$, linearly coupled to the bath coordinates $y_j(t)$,

$$H = \frac{m\dot{x}^2}{2} + V(x) + H_b - \sum_j c_j y_j(t) x(t), \tag{555}$$

where c_j are the coupling constants that linearly couple $x(t)$ and $y_j(t)$. The terms $V(x)$ and H_b, introduced by Eq. (555), describe the interactions among system coordinates and among bath coordinates, respectively. The total force acting on the particle is

$$F(t) = -\frac{\partial V}{\partial x} + f(t), \tag{556}$$

where the fluctuating force $f(t)$ can be readily identified from Eq. (555),

$$f(t) = \sum_j c_j y_j(t). \tag{557}$$

Note that the motion of $y_j(t)$ depends on $x(t)$ since, according to Eq. (555), the force acting on $y_j(t)$ is $f_j = -\partial H/\partial y_j = -\partial H_b/\partial y_j + c_j x(t)$. Therefore, $f(t)$ is also a function of $x(t)$. Assuming that $f(t)$ is linear in $x(t)$,

$$f(t) = f_b(t) + \int_{-\infty}^{\infty} dt' \chi_b(t-t') x(t'), \tag{558}$$

where, according to Eq. (536),

$$\chi_b(t-t') = \begin{cases} -\beta \frac{dC_b(t-t')}{d(t-t')}, & \text{when } t > t', \\ 0, & \text{otherwise,} \end{cases} \tag{559}$$

with

$$C_b(t) = < \delta f(0) \delta f(t) > . \tag{560}$$

Therefore, the equation of motion for the particle is

$$m\ddot{x} = -\frac{dV}{dx} + f_b(t) + \int_{-\infty}^{t} dt' (-\beta) \frac{dC_b(t-t')}{d(t-t')} x(t'). \tag{561}$$

Changing the integration variable t' to $t'' = t - t'$, in Eq. (561), we obtain

$$m\ddot{x} = -\frac{dV}{dx} + f_b(t) - \int_{t}^{0} dt'' (-\beta) \frac{dC_b(t'')}{dt''} x(t-t''), \tag{562}$$

where the lower integration limit includes only values of $x(t-t'')$ with $(t-t'') > 0$. Integrating Eq. (562) by parts, we obtain

$$m\ddot{x} = -\frac{dV}{dx} + f_b(t) - [x(t-t'')\beta C_b(t'')|_{t''=t}^{t''=0} + \int_{t}^{0} dt'' (-\beta) C_b(t'') \frac{\partial x(t-t'')}{\partial t''}]. \tag{563}$$

Changing the integration variable t'' to $t' = t - t''$, in Eq. (563), we obtain

$$m\ddot{x} = -\frac{dV}{dx} + f_b(t) + [x(0)\beta C_b(t) - x(t)\beta C_b(0)] - \int_{0}^{t} \beta C_b(t-t') \dot{x}(t') dt'. \tag{564}$$

Eq. (564) is the *Generalized Langevin Equation*, which can be written in terms of the *potential of mean force*

$$\overline{V}(x) \equiv V(x) + \frac{1}{2} x^2 \beta C_b(0), \tag{565}$$

and the fluctuating force

$$\overline{F}(t) \equiv f_b(t) + \chi(0)\beta C_b(t), \tag{566}$$

as follows,

$$\boxed{m\ddot{x} = -\frac{\partial \overline{V}}{\partial x} + \overline{F}(t) - \int_0^t \beta C_b(t-t')\dot{x}(t')dt',} \tag{567}$$

where the third term on the r.h.s. of Eq. (567) is the generalized frictional force, a force that is *linear* in the velocity. The connection between the frictional force and the regression of thermal fluctuations of $f(t)$, introduced by Eq. (567), is known as the *second fluctuation-dissipation theorem*.

Note (Laplace transform method): Equation (567) can be used to obtain the equation of motion for the correlation function $c(t) = \langle x(t)x(0)\rangle$ when $V(x) = \frac{1}{2}m\omega^2 x^2$. Multiplying Eq. (567) by $x(0)$ and computing the ensemble average to obtain:

$$m\frac{d^2}{dt^2}c(t) = -m\bar{\omega}^2 c(t) - \int_0^t dt'\beta C_b(t-t')\dot{c}(t'), \tag{568}$$

where $\bar{\omega}^2 = \omega^2 + \beta C_b(0)/m$, since $\langle \bar{F}(t)x(0)\rangle = 0$.

To solve Eq. (568) and find $c(t)$, we implement the *Laplace transform method*, as follows. We compute the Laplace transform of both sides of Eq. (568) to obtain,

$$m\int_0^\infty dte^{-st}\frac{d^2}{dt^2}c(t) = -m\bar{\omega}^2\int_0^\infty dte^{-st}c(t) - \int_0^\infty dte^{-st}\int_0^t dt'\beta C_b(t-t')\frac{d}{dt'}c(t'). \tag{569}$$

Integrating by parts, we obtain:

$$\begin{aligned}\int_0^\infty dte^{-st}\dot{c}(t) &= c(t)e^{-st}|_0^\infty + s\int_0^\infty dte^{-st}c(t),\\ &= -\langle x(0)^2\rangle + s\tilde{c}(s),\end{aligned} \tag{570}$$

with $\tilde{c}(s) = \int_0^\infty dte^{-st}c(t)$ since $c(0) = \langle x(0)^2\rangle$. In addition,

$$\begin{aligned}\int_0^\infty dte^{-st}\frac{d^2}{dt^2}\langle x(0)x(t)\rangle &= \langle x(0)\dot{x}(t)\rangle e^{-st}|_0^\infty + s\int_0^\infty dte^{-st}\dot{c}(t),\\ &= s(-\langle x(0)^2\rangle + s\tilde{c}(s)),\end{aligned} \tag{571}$$

since $\langle x(0)\dot{x}(0)\rangle = 0$.

Substituting into Eq. (569), we obtain:

$$\begin{aligned}ms(-\langle x(0)^2\rangle + s\tilde{c}(s)) &= -m\bar{\omega}^2\tilde{c}(s) - \beta\int_0^\infty dte^{-st}\int_0^t dt' C_b(t-t')\dot{c}(t'),\\ &= -m\bar{\omega}^2\tilde{c}(s) - \beta\int_0^\infty dt'\dot{c}(t')\int_{t'}^\infty dte^{-st}C_b(t-t'),\\ &= -m\bar{\omega}^2\tilde{c}(s) - \beta\int_0^\infty dt'e^{-st'}\dot{c}(t')\int_0^\infty dt''e^{-st''}C_b(t''),\\ &= -m\bar{\omega}^2\tilde{c}(s) - \beta(-\langle x(0)^2\rangle + s\tilde{c}(s))\tilde{C}_b(s).\end{aligned} \tag{572}$$

Distributing and solving for $\tilde{c}(s)$, we obtain:

$$-ms\langle x(0)^2\rangle + ms^2\tilde{c}(s) = -(m\bar{\omega}^2 + \beta s\tilde{C}_b(s))\tilde{c}(s) + \beta\langle x(0)^2\rangle\tilde{C}_b(s), \tag{573}$$

and

$$\tilde{c}(s) = \langle x^2 \rangle \frac{s + \beta \tilde{C}_b(s)/m}{s^2 + \bar{\omega}^2 + \beta s \tilde{C}_b(s)/m}, \tag{574}$$

giving $c(t)$ according to the inverse Laplace transform $c(t) = \int_0^\infty ds e^{st} \tilde{c}(s)$.

Markovian Approximation

Changing the interation variable t', in Eq. (567), to $t'' = t - t'$ and *considering a time t much larger than the relaxation time scale for the correlation function* C_b (so that $C_b(t) = 0$ and $\partial x(t - t'')/\partial(t - t'') \approx \partial x(t)/\partial t$), we obtain

$$m\ddot{x} = -\frac{\partial \overline{V}}{\partial x} + f_b(t) - \beta \int_0^\infty C_b(t'') \dot{x}(t) dt''. \tag{575}$$

Note that Eq. (575) becomes the traditional *Langevin Equation*,

$$\boxed{m\ddot{x} = f_b(t) - \gamma \dot{x}(t),} \tag{576}$$

when $-\partial \overline{V}/\partial x = 0$. The friction coefficient γ is, therefore, determined by the regression of spontaneous thermal fluctuations as follows

$$\gamma = \beta \int_0^\infty C_b(t'') dt''. \tag{577}$$

The approximation implemented to obtain Eq. (575) involves considering that the relaxation time for fluctuating forces in the bath is much shorter than the time over which one observes the particle. Such approximation removes the "memory effects" from the equation of motion (note that Eq. (575) does not involve the nonlocality in time introduced by the time integral in Eq. (567)). This approximation is thus called *Markovian approximation* since it makes the instantaneous force independent of the state of the particle at any previous time.

Note that, according to Eq. (576),

$$m\langle \dot{\mathbf{v}} \rangle = -\gamma \langle \mathbf{v}(t) \rangle, \tag{578}$$

where $\mathbf{v} = \dot{x}$, since $\langle f(t) \rangle = 0$. The solution to Eq. (578) is,

$$\langle \mathbf{v}(t) \rangle = \langle \mathbf{v}(0) \rangle \exp(-\gamma t/m). \tag{579}$$

Eq. (579) indicates that the average initial momentum of the particle is dissipated into the bath at an exponential rate (i.e., the average velocity vanishes at an exponential rate). However, it is important to note that the condition $\langle \mathbf{v}(t) \rangle = 0$ at $t >> m/\gamma$ (e.g., at equilibrium) does not imply that the particle is at rest! At equilibrium, the fluctuating force $f_b(t)$ keeps buffeting the particle and the distribution of velocities is given by the Boltzmann distribution (see Eq. (462)). The average squared velocity for the particle is

$$\langle \mathbf{v}^2 \rangle = \frac{\int_{-\infty}^\infty dv_x \int_{-\infty}^\infty dv_y \int_{-\infty}^\infty dv_z (v_x^2 + v_y^2 + v_z^2) \exp(-\beta m (v_x^2 + v_y^2 + v_z^2)/2)}{\int_{-\infty}^\infty dv_x \int_{-\infty}^\infty dv_y \int_{-\infty}^\infty dv_z \exp(-\beta m (v_x^2 + v_y^2 + v_z^2)/2)} = \frac{3kT}{m}, \tag{580}$$

and the velocity autocorrelation function is

$$\langle \mathbf{v}(t)\mathbf{v}(0) \rangle = \langle \mathbf{v}(0)\mathbf{v}(0) \rangle \exp(-\gamma t/m) = \frac{3kT}{m} \exp(-\gamma t/m), \tag{581}$$

since Eq. (578) is valid not only for the dynamical variable $\mathbf{v}(t)$ but also for $\mathbf{v}(t)\mathbf{v}(0)$.

The motion of the particle is called *Brownian Motion*, in honor to the botanist Robert Brown who *observed* it for the first time in his studies of pollen. In 1828 he wrote "the pollen become dispersed in water in a great number of small particles which were perceived to have an irregular swarming motion". The theory of such motion, however, was

derived by A. Einstein in 1905 when he wrote: “In this paper it will be shown that ... bodies of microscopically visible size suspended in a liquid perform movements of such magnitude that they can be easily observed in a microscope on account of the molecular motions of heat ...”
To compute the average mean squared displacement $\langle \bar{x}^2 \rangle$ of the particle we substitute the variable x(t) in Eq. (576) by $\bar{x} = x(t) - x(0)$, we multiply both sides of Eq. (576) by such variable and we average over the ensemble distribution as follows,

$$m\langle \bar{x}\frac{\partial \dot{\bar{x}}}{\partial t}\rangle = -\gamma\langle \bar{x}\dot{\bar{x}}\rangle, \tag{582}$$

since $\langle \bar{x}f(t)\rangle = \langle \bar{x}\rangle\langle f(t)\rangle = 0$. Eq. (582) is equivalent to

$$m\langle \frac{\partial \bar{x}\dot{\bar{x}}}{\partial t}\rangle = -\gamma\langle \bar{x}\dot{\bar{x}}\rangle + m\langle \dot{\bar{x}}^2\rangle, \tag{583}$$

which, according to Eq. (580), gives

$$m\frac{\partial\langle \bar{x}\dot{\bar{x}}\rangle}{\partial t} = -\gamma\langle \bar{x}\dot{\bar{x}}\rangle + 3kT. \tag{584}$$

The solution to Eq. (584) is

$$\langle \bar{x}\dot{\bar{x}}\rangle = \frac{1}{2}\frac{\partial}{\partial t}\langle \bar{x}^2\rangle = -\frac{3kT}{\gamma}(\exp(-\gamma t/m) - 1). \tag{585}$$

Therefore, the mean squared displacement is

$$\boxed{\langle \bar{x}^2\rangle = -\frac{6kT}{\gamma}\left(-\frac{m}{\gamma}(\exp(-\gamma t/m) - 1) - t\right).} \tag{586}$$

At *short times* (i.e., when $\exp(-\gamma t/m) - 1 \approx -\gamma t/m + t^2/2 * \gamma^2/m^2$),

$$\langle \bar{x}^2\rangle = \frac{3kT}{m}t^2, \tag{587}$$

i.e., *the mean squared displacement at short times is quadratic in time*. This is the so-called *ballistic regime*, since it corresponds to ballistic motion (motion without collisions) for a particle with velocity equal to $\sqrt{3kT/m}$.
At *long times* (i.e., when $m/\gamma(\exp(-\gamma t/m) - 1) << t$),

$$\langle \bar{x}^2\rangle = \frac{6kT}{\gamma}t = 6Dt, \tag{588}$$

where the constant

$$D = \frac{kT}{\gamma}, \tag{589}$$

is the *diffusion coefficient*. Therefore, *at long times the mean squared displacement is linear in time*. This long time limit is the so-called *diffusional regime*.
The remaining of this section shows that the diffusion coefficient can be computed in terms of the velocity autocorrelation function $\langle \mathbf{v}(0)\mathbf{v}(t)\rangle$ as follows:

$$\boxed{D = \frac{1}{3}\int_0^t dt'\langle \mathbf{v}(0)\mathbf{v}(t')\rangle.} \tag{590}$$

Note that Eq. (589) can be readily obtained by substituting Eq. (581) into Eq. (590).
To prove Eq. (590), consider the particle displacement at time t,

$$\bar{x}(t) = \int_0^t dt' \mathbf{v}(t'), \tag{591}$$

and compute the time derivative of the squared displacement as follows

$$\frac{\partial}{\partial t}\bar{x}^2 = \frac{\partial}{\partial t}(\int_0^t dt' \mathbf{v}(t'))^2, \tag{592}$$

which according to Eq. (535) gives,

$$\frac{\partial}{\partial t}\langle \bar{x}^2 \rangle = 2\int_0^t dt' \langle \mathbf{v}(t)\mathbf{v}(t')\rangle. \tag{593}$$

Changing integration variables from t' to $t'' = t - t'$ we obtain

$$\frac{\partial}{\partial t}\langle \bar{x}^2 \rangle = 2\int_0^t dt'' \langle \mathbf{v}(t)\mathbf{v}(t-t'')\rangle = 2\int_0^t dt'' \langle \mathbf{v}(0)\mathbf{v}(-t'')\rangle, \tag{594}$$

since $C(t'+t-t) = \langle \mathbf{v}(t)\mathbf{v}(t'+t)\rangle$ is equal to $C(t'-0) = \langle \mathbf{v}(0)\mathbf{v}(t')\rangle$. Finally, since $C(t'') = C(-t'')$ we obtain

$$\frac{\partial}{\partial t}\langle \bar{x}^2 \rangle = 2\int_0^t dt'' \langle \mathbf{v}(0)\mathbf{v}(t'')\rangle, \tag{595}$$

Eq. (590) is obtained by substituting Eq. (588) into the l.h.s. of Eq. (595).

47 Velocity Verlet Algorithm

The goal of this section is to introduce the velocity Verlet algorithm for *molecular dynamics (MD) simulations* and to show how to implement it to compute time-correlation functions (e.g., $\langle v(t)v(0)\rangle$) and transport coefficients (e.g., the diffusion coefficient D).
Consider the task of computing the diffusion coefficient according to Eq. (590). The quantity of interest is, therefore, the equilibrium ensemble average of the velocity autocorrelation function

$$\langle v(0)v(t)\rangle = Tr\{\rho\ (v(0)v(t))\}. \tag{596}$$

The computation of $\langle v(0)v(t)\rangle$ thus requires sampling initial conditions according to the ensemble distribution described by ρ and for each initial condition compute the value of the particle velocity $v(t)$ at time t.
The *velocity Verlet algorithm* (W. C. Swope, H. C. Andersen, P. H. Berens, and K. R. Wilson, *J. Chem. Phys.* **76**, 637 (1982)) computes the particle velocity $v(t+\tau)$ and position $x(t+\tau)$ at time t+τ as follows:

$$\begin{aligned} x(t+\tau) &= x(t) + v(t)\tau + \tfrac{1}{2}\tau^2\tfrac{f(t)}{m}, \\ v(t+\tau) &= v(t) + \tfrac{f(t)+f(t+\tau)}{2m}\tau, \end{aligned} \tag{597}$$

where τ is a small time increment, m is the particle mass and $f(t)$ is the total force acting on the particle at time t. Given the initial conditions $x(0)$ and $v(0)$ one can compute $v(t)$ and $x(t)$ simply by applying Eqs. (597) successively n times, with $n = t/\tau$. Note that by implementing the algorithm one generates a *phase space trajectory* (i.e., a sequence of “snapshots” for the particle coordinates and velocities at all intermediate times $t_j = j * \tau$ (with j=1,2,...,n)).

We note that the evolution is time reversible since the velocities are evolved with a force that is symmetric in time (i.e., same force evolves velocities from t to $t+\tau$ than from $t+\tau$ to t). Time reversibility can be shown more formally by showing that the velocity Verlet algorithm is essentially the Trotter decomposition of the propagator, as shown by Tuckerman and Berne.

Molecular dynamics simulations thus provide the sequence of microscopic configurations through which the model system passes in time. Such detailed microscopic information allows one to compute the result of a measurement of an observable (i.e., an ensemble average) according to the time average introduced by Eq. (**??**) (i.e., simply by averaging the value of the observable throught the whole manifold of microscopic configurations generated during the time of the measurement). Therefore, another way of computing the ensemble average introduced by Eq. (596) is

$$\langle v(0)v(t)\rangle = \frac{1}{T}\int_0^T dt' v(t')v(t'+t), \tag{598}$$

where T is the time of the measurement of the diffusion constant D, a time that is much larger than the relaxation time of the velocity autocorrelation function.

Exercise:

Compute the velocity autocorrelation function for a fluid of argon atoms using the program developed for computing the radial distribution function. (Hint: substitute the Metropolis Monte Carlo algorithm by the velocity Verlet algorithm).

48 Thermal Correlation Functions

Equation 595 shows how to obtain transport coefficients, such as the diffusion coefficient by integration of the thermal correlation function corresponding to the velocity-velocity autocorrelation. Classical calculations can be performed by propagating Hamilton's equations according to the Velocity-Verlet algorithm. Coordinates and momenta $q(t)$ and $p(t)$ are propagated for a sufficiently long trajectory and classical correlation functions are obtained as follows:

$$C(t) = \langle A(0)B(t)\rangle = \frac{1}{\tau}\int_0^\tau dt' A(q(t'),p(t'))B(q(t'+t),p(t'+t)), \tag{599}$$

where $A(0)$ and $B(t)$ represent the quantities of interest at time 0 and t, respectively. [2]

The goal of this section is to show how to compute thermal correlation functions $C(t) = \langle A(0)B(t)\rangle$ for systems where quantum mechanical effects are important. The quantum mechanical expression of $C(t)$ is,

$$C(t) = Tr[\hat{\rho}\hat{A}\hat{B}(t)], \tag{600}$$

where $\hat{\rho} = Z^{-1}exp(-\beta\hat{H})$ is the density operator and the operators $\hat{A}$ and $\hat{B}(t)$ are defined so that $A(0) = \langle\Psi_0|\hat{A}|\Psi_0\rangle$ is the expectation value of A at $t=0$. In addition,

$$B(t) = \langle\Psi_0|\hat{B}(t)|\Psi_0\rangle = \langle\Psi_0|e^{(i/\hbar)\hat{H}t}\hat{B}e^{-(i/\hbar)\hat{H}t}|\Psi_0\rangle, \tag{601}$$

is the expectation value of $\hat{B}$ at time t when the system is initially prepared in state $|\Psi_0\rangle$ and evolves according to the Hamiltonian,

$$\hat{H} = \hat{p}^2/(2m) + \hat{V}, \tag{602}$$

as follows: $|\Psi_t\rangle = e^{-(i/\hbar)\hat{H}t}|\Psi_0\rangle$. Note that $\hat{B}(t) = e^{(i/\hbar)\hat{H}t}\hat{B}e^{-(i/\hbar)\hat{H}t}$ is the Heisenberg operator associated with quantity B.

[2] Note that calculations of $C(t)$ provide a description of any equilibrium property, $\langle A\rangle$, when $\hat{B}=1$, or dynamical ensemble average $\langle B(t)\rangle$, when $\hat{A}=1$, respectively.

Thermal correlation functions can therefore be expressed as,

$$C(t) = Z^{-1} Tr[e^{-\beta\hat{H}} \hat{A} e^{(i/\hbar)\hat{H}t} \hat{B} e^{-(i/\hbar)\hat{H}t}], \tag{603}$$

an expression that can be re-written in coordinate representation as follows:

$$\begin{aligned} &C(t) = Z^{-1} \int dx \int dx' \int dx'' \int dx''' \int dx'''' \langle x|e^{-\beta\hat{H}}|x'\rangle \langle x'|\hat{A}|x''\rangle \langle x''|e^{(i/\hbar)\hat{H}t}|x'''\rangle \\ &\langle x'''|\hat{B}|x''''\rangle \langle x''''|e^{-(i/\hbar)\hat{H}t}|x\rangle. \end{aligned} \tag{604}$$

Note that to compute $C(t)$ it is necessary to obtain expressions for the Boltzmann operator matrix elements $\langle x|e^{-\beta\hat{H}}|x'\rangle$ as well as for the forward and backward time-evolution operator matrix elements $\langle x|e^{-(i/\hbar)\hat{H}t}|x'\rangle$ and $\langle x|e^{(i/\hbar)\hat{H}t}|x'\rangle$, respectively.

48.1 Boltzmann Operator Matrix elements

To obtain the matrix elements of the Boltzmann operator at finite temperature $T = (k_B\beta)^{-1}$, we write the exponential operator as a product of a large number n of exponential operators,

$$\langle x_0|e^{-\beta\hat{H}}|x_n\rangle = \langle x_0|e^{-\epsilon\hat{H}} e^{-\epsilon\hat{H}} ... e^{-\epsilon\hat{H}}|x_n\rangle, \tag{605}$$

where $\epsilon \equiv \beta/n << 1$. Inserting the closure relation in between exponential operators we obtain,

$$\langle x_0|e^{-\beta\hat{H}}|x_n\rangle = \int dx_1 ... \int dx_{n-1} \langle x_0|e^{-\epsilon\hat{H}}|x_1\rangle ... \langle x_{n-1}|e^{-\epsilon\hat{H}}|x_n\rangle. \tag{606}$$

The high-temperature Boltzmann operator $e^{-\epsilon\hat{H}}$ can be written in the form of the Trotter expansion,

$$e^{-\epsilon\hat{H}} \approx e^{-\epsilon\hat{V}/2} e^{-\epsilon\hat{p}^2/(2m)} e^{-\epsilon\hat{V}/2}, \tag{607}$$

to second order accuracy. Therefore, matrix elements of the Boltzmann operator at high-temperature can be obtained as follows:

$$\langle x_0|e^{-\epsilon\hat{H}}|x_1\rangle = \int dx \int dp \int dx' \int dp' \langle x_0|e^{-\epsilon\hat{V}/2}|x'\rangle \langle x'|p'\rangle \langle p'|e^{-\epsilon\hat{p}^2/(2m)}|p\rangle \langle p|x\rangle \langle x|e^{-\epsilon\hat{V}/2}|x_1\rangle, \tag{608}$$

where

$$\langle x|p\rangle = \frac{1}{\sqrt{2\pi\hbar}} e^{\frac{i}{\hbar}xp}, \tag{609}$$

since

$$-i\hbar \frac{\partial}{\partial x} \langle x|p\rangle = p\langle x|p\rangle. \tag{610}$$

Furthermore,

$$\langle x|e^{-\epsilon\hat{V}/2}|x'\rangle = e^{-\epsilon V(x)/2} \delta(x - x'). \tag{611}$$

Therefore,

$$\begin{aligned} &\langle x_0|e^{-\epsilon\hat{H}}|x_1\rangle = \tfrac{1}{2\pi\hbar} \int dx \int dp \int dx' \int dp' e^{-\epsilon V(x')/2} \delta(x' - x_0) e^{\frac{i}{\hbar}x'p'} e^{-\epsilon p^2/(2m)} \delta(p - p') \\ &e^{-\frac{i}{\hbar}xp} \delta(x - x_1) e^{-\epsilon V(x_1)/2}, \end{aligned} \tag{612}$$

which gives,

$$\langle x_0|e^{-\epsilon\hat{H}}|x_1\rangle = \frac{1}{2\pi\hbar} e^{-\frac{\epsilon}{2}[V(x_0)+V(x_1)]} \int dp e^{-\epsilon p^2/(2m) + \frac{i}{\hbar}(x_0 - x_1)p}, \tag{613}$$

or,

$$\begin{aligned}\langle x_0|e^{-\epsilon\hat{H}}|x_1\rangle &= e^{-\frac{\epsilon}{2}(V(x_0)+V(x_1))}\sqrt{\frac{m}{2\pi\epsilon\hbar^2}}e^{-\frac{1}{2}m\left[\frac{(x_1-x_0)}{\hbar\epsilon}\right]^2\epsilon},\\ &\approx e^{-\epsilon V(x_1)}\sqrt{\frac{m}{2\pi\epsilon\hbar^2}}e^{-\frac{1}{2}m\left[\frac{(x_1-x_0)}{\hbar\epsilon}\right]^2\epsilon},\end{aligned} \tag{614}$$

Matrix elements of the Boltzmann operator at finite-temperature can be obtained by substituting Eq. (614) into Eq. (606):

$$\langle x_0|e^{-\beta\hat{H}}|x_n\rangle = \int dx_1 ... \int dx_{n-1}\left(\frac{m}{2\pi\epsilon\hbar^2}\right)^{n/2} e^{-\epsilon\sum_{j=1}^{n}V(x_j)+\frac{1}{2}m\omega^2(x_j-x_{j-1})^2}, \tag{615}$$

where $\omega = 1/(\hbar\epsilon)$. Note that the r.h.s of Eq. (615) corresponds to the partition function of a chain of n-harmonic oscillators with cordinates x_j under the influence of an external potential $V(x_j)$. Each chain of harmonic oscillators describes a path from x_0 to x_n.

The multidimentional integral, introduced by Eq. (615), can be computed by importance sampling Monte Carlo by sampling sets of coordinates $x_1, ..., x_{n-1}$ with sampling functions defined by the Gaussians associated with the linked harmonic oscillators. Such a computational approach for obtaining thermal equilibrium density matrices is called *Path Integral Monte Carlo*.

48.2 The Bloch Equation: SOFT Integration

The Boltzmann-operator matrix-elements are the solution of the Bloch equation,

$$\{\frac{\partial}{\partial\beta} - \frac{\hbar^2}{2m}\nabla^2_{\mathbf{x}} + V_0(\mathbf{x})\}\rho(\mathbf{x},\mathbf{x}';\beta) = 0, \tag{616}$$

with $\rho(\mathbf{x},\mathbf{x}';\beta) \equiv \langle\mathbf{x}|e^{-\beta\hat{H}_0}|\mathbf{x}'\rangle$ and $\hat{H}_0 = -\frac{\hbar^2}{2m}\nabla^2_{\mathbf{x}} + V_0(\mathbf{x})$. The Bloch equation, introduced by Eq. (616), is obtained by simply computing the derivative of $\langle\mathbf{x}|e^{-\beta\hat{H}_0}|\mathbf{x}'\rangle$ with respect to β and can be formally integrated, as follows:

$$\rho(\mathbf{x},\mathbf{x}';\beta) = \int d\mathbf{x}''\rho(\mathbf{x},\mathbf{x}'';\beta-\epsilon)\rho(\mathbf{x}'',\mathbf{x}';\epsilon), \tag{617}$$

with initial condition $\rho(\mathbf{x}'',\mathbf{x}';\epsilon)$ defined according to the high-temperature approximation, introduced by Eq. (614):

$$\rho(\mathbf{x},\mathbf{x}';\epsilon) = \left(\frac{m}{2\pi\epsilon}\right)^{1/2} e^{-\frac{\epsilon}{2}[V_0(\mathbf{x})+V_0(\mathbf{x}')]}e^{-\frac{m}{2\epsilon}(\mathbf{x}-\mathbf{x}')^2}. \tag{618}$$

Note: We note that the Bloch equation, introduced by Eq. (616), is the imaginary-time version of the time-dependent Schrödinger equation since introducing the change of variables $\beta \equiv it/\hbar$ and $\rho(x,x';\beta) = \psi_{x'}(x;\beta)$, we obtain:

$$\{-i\hbar\frac{\partial}{\partial t} - \frac{\hbar^2}{2m}\nabla^2_{\mathbf{x}} + V_0(\mathbf{x})\}\psi_{x'}(\mathbf{x};t) = 0. \tag{619}$$

Therefore, the same numerical methods developed for integration of the time-dependent Schrödinger equation can be used for integration of the Bloch equation by simply changing the variable t by $-i\hbar\beta$. One of those methods is the Split-Operator Fourier Transform (SOFT) method introduced in the next section.

48.3 SOFT Method

In this section we introduce the SOFT for integration of the Schrödinger equation, as necessary for wavepacket propagation. Next, we show that it can be analogously applied to obtain $\rho_{x';\beta}(x)$ from $\rho_{x';\epsilon}(x)$ by imaginary time propagation.

Integration of the Schrödinger equation: The SOFT method is a numerical approach for solving the time-dependent Schrödinger equation by using a grid-based representation of the time-evolving states and operators.

The essence of the method is to discretize the propagation time on a grid $t_k = (k-1)\tau$, with $k = 1, ..., n$ and time-resolution $\tau = t/(n-1)$, and obtain the wave-packet at the intermediate times t_k by recursively applying the evolution operator, as follows:

$$\Psi_{t_{k+1}}(x) = \int dx' \langle x|e^{-i\hat{H}\tau}|x'\rangle\langle x'|\Psi_{t_k}\rangle. \tag{620}$$

If τ is a sufficiently small time-increment (*i.e.*, n is large), the time-evolution operator can be approximated according to the Trotter expansion to second order accuracy,

$$e^{-i\hat{H}\tau} = e^{-iV(\hat{x})\tau/2}e^{-i\hat{p}^2\tau/(2m)}e^{-iV(\hat{x})\tau/2} + O(\tau^3), \tag{621}$$

which separates the propagator into a product of three operators, each of them depending either on $\hat{x}$, or $\hat{p}$.

Substituting Eq. (621) into Eq. (620) and inserting the closure relation $\hat{\mathbf{1}} = \int dp|p\rangle\langle p|$ gives,

$$\Psi_{t_{k+1}}(x) = \int dp \int dx' e^{-iV(\hat{x})\tau/2}\langle x|p\rangle e^{-ip^2\tau/(2m)}\langle p|x'\rangle e^{-iV(x')\tau/2}\Psi_{t_k}(x'). \tag{622}$$

By substituting $\langle p|x'\rangle$ and $\langle x|p\rangle$ as plane-waves, we obtain:

$$\Psi_{t_{k+1}}(x) = e^{-iV(\hat{x})\tau/2}\frac{1}{\sqrt{2\pi}}\int dp e^{ixp}e^{-ip^2\tau/(2m)}\frac{1}{\sqrt{2\pi}}\int dx' e^{-ipx'}e^{-iV(x')\tau/2}\Psi_{t_k}(x'). \tag{623}$$

According to Eq. (623), then, the computational task necessary to propagate $\Psi_t(x)$ for a time-increment τ involves the following steps:

1. Represent $\Psi_{t_k}(x')$ and $e^{-iV(x')\tau/2}$ as arrays of numbers $\Psi_{t_k}(x_j)$ and $e^{-iV(x_j)\tau/2}$ associated with a grid of equally spaced coordinates $x_j = x_{min} + (j-1)\Delta$, with finite resolution $\Delta = (x_{max} - x_{min})/(n-1)$.

2. Apply the potential energy part of the Trotter expansion $e^{-iV(x')\tau/2}$ to $\Psi_{t_k}(x')$ by simple multiplication of array elements:
$$\tilde{\Psi}_{t_k}(x_j) = e^{-iV(x_j)\tau/2}\Psi_{t_k}(x_j).$$

3. Fourier transform $\tilde{\Psi}_{t_k}(x_j)$ to obtain $\tilde{\Psi}_{t_k}(p_j)$, and represent the kinetic energy part of the Trotter expansion $e^{-ip^2\tau/(2m)}$ as an array of numbers $e^{-ip_j^2\tau/(2m)}$ associated with a grid of equally spaced momenta $p_j = j/(x_{max} - x_{min})$.

4. Apply the kinetic energy part of the Trotter expansion $e^{-ip^2\tau/(2m)}$ to the Fourier transform $\tilde{\Psi}_{t_k}(p)$ by simple multiplication of array elements:
$$\widetilde{\Psi}_{t_k}(p_j) = e^{-ip_j^2\tau/(2m)}\tilde{\Psi}_{t_k}(p_j).$$

5. Inverse Fourier transform $\widetilde{\Psi}_{t_k}(p_j)$ to obtain $\widetilde{\Psi}_{t_k}(x_j)$ on the grid of equally spaced coordinates x_j.

6. Apply the potential energy part of the Trotter expansion $e^{-iV(x')\tau/2}$ to $\widetilde{\Psi}_{t_k}(x')$ by simple multiplication of array elements,
$$\Psi_{t_{k+1}}(x_j) = e^{-iV(x_j)\tau/2}\widetilde{\Psi}_{t_k}(x_j).$$

48.4 Imaginary time propagation

Note that with the variable substitution $\tau \rightarrow -it$, with real t, the time evolution operator becomes a decaying exponential $e^{-\hat{H}t/\hbar}$ that reduces the amplitude of the initial wavepacket $\Psi_0(x) = \sum_j c_j\phi_j(x)$, as follows:

$$\Psi_t(x) = e^{-i\hat{H}\tau/\hbar}\Psi_0(x) = \sum_j c_j e^{-E_j t}\phi_j(x), \tag{624}$$

where $\hat{H}\phi_j(x) = E_j\phi_j(x)$. Terms with higher E_j are reduced more than those with smaller E_j. After renormalizing the resulting wavefunction $\Psi_t(x)$ (by dividing it by the square root of its norm), we get a state enriched with low energy components. The imaginary time propagation and renormalization procedure can be repeated several times until the function stops changing since it composed solely by the ground state $\phi_0(x)$, after removal of all other components (of higher energies) at a faster rate.

Having found ϕ_0, we can proceed to find ϕ_1 as done for ϕ_0 but including orthogonalization relative to ϕ_0, $\Psi_t(x) \rightarrow \Psi_t(x) - \langle\phi_0|\Psi_t\rangle\phi_0(x)$, after each propagation step, right before renormalization. Higher energy states are found analogously, by orthogonalization of the propagated state relative to all previously found eigenstates.

48.5 Ehrenfest Dynamics

The goal of this section is to show that the expectation values $\langle\hat{x}\rangle = \langle\psi|\hat{x}|\psi\rangle$ and $\langle\hat{p}\rangle = \langle\psi|\hat{p}|\psi\rangle$ are conjugate variables in the sense that they evolve according to the classical equations of motion (*i.e.*, Hamilton's equaitons):

$$\begin{aligned}\frac{d}{dt}\langle\hat{x}\rangle &= \langle\frac{\partial\hat{H}}{\partial\hat{p}}\rangle\\ &= \frac{\langle\hat{p}\rangle}{m},\\ \frac{d}{dt}\langle p\rangle &= -\langle\frac{\partial\hat{H}}{\partial\hat{x}}\rangle\\ &= -\langle V'\rangle,\end{aligned} \tag{625}$$

where $\hat{H} = \hat{p}^2/2m + V(\hat{x})$. This remarkable result, introduced by Eq. (625), is known as *Ehrenfest's theorem* and can be demonstrated, as follows.

First, we show that since ψ evolves according to the Schrödinger equation:

$$i\hbar\frac{\partial\psi}{\partial t} = \hat{H}\psi, \tag{626}$$

then

$$\langle\hat{p}\rangle = m\frac{d}{dt}\langle\hat{x}\rangle. \tag{627}$$

Using integration by parts, we obtain:

$$\begin{aligned}
\langle\hat{p}\rangle &= -i\hbar\langle\psi|\frac{\partial}{\partial x}|\psi\rangle, \\
&= -\frac{i\hbar}{2}\langle\psi|\frac{\partial}{\partial x}+\frac{\partial}{\partial x}|\psi\rangle, \\
&= -\frac{i\hbar}{2}\int dx\left[\psi^*\frac{\partial\psi}{\partial x}+\psi^*\frac{\partial\psi}{\partial x}\right], \\
&= -\frac{i\hbar}{2}\int dx\left[\psi^*\frac{\partial\psi}{\partial x}-\psi\frac{\partial\psi^*}{\partial x}\right], \\
&= m\int j\,dx, \\
&= -m\int x\frac{\partial j}{\partial x}\,dx,
\end{aligned} \tag{628}$$

where the current $j = -\frac{i\hbar}{2m}\left[\psi^*\frac{\partial\psi}{\partial x}-\psi\frac{\partial\psi^*}{\partial x}\right]$ satisfies the continuity equation,

$$\frac{d}{dt}\psi^*\psi+\frac{\partial j}{\partial x}=0. \tag{629}$$

Therefore,

$$\begin{aligned}
\langle\hat{p}\rangle &= m\int x\frac{d}{dt}\psi^*\psi\,dx, \\
&= m\frac{d}{dt}\int\psi^*x\psi\,dx, \\
&= m\frac{d}{dt}\langle\hat{x}\rangle.
\end{aligned} \tag{630}$$

Next, we show that

$$\frac{d}{dt}\langle\hat{p}\rangle = -\langle V'\rangle, \tag{631}$$

by substituting Eq. (626) into Eq. (628) and integrating by parts, as follows:

$$\begin{aligned}
\frac{d}{dt}\langle\hat{p}\rangle &= m\frac{d}{dt}\int_{-\infty}^{\infty}dx\,j, \\
&= -\frac{i\hbar}{2}\int_{-\infty}^{\infty}dx\left[\frac{d\psi^*}{dt}\frac{\partial\psi}{\partial x}+\psi^*\frac{\partial}{\partial x}\frac{d\psi}{dt}-\frac{d\psi}{dt}\frac{\partial\psi^*}{\partial x}-\psi\frac{\partial}{\partial x}\frac{d\psi^*}{dt}\right] \\
&= i\hbar\int_{-\infty}^{\infty}dx\left[\frac{d\psi}{dt}\frac{\partial\psi^*}{\partial x}+\frac{d\psi^*}{dt}\frac{\partial\psi}{\partial x}\right] \\
&= \int_{-\infty}^{\infty}dx\left[-\frac{\hbar^2}{2m}\frac{\partial^2\psi}{\partial x^2}\frac{\partial\psi^*}{\partial x}+V\psi\frac{\partial\psi^*}{\partial x}+c.c.\right] \\
&= \int_{-\infty}^{\infty}dx\left[-\frac{\hbar^2}{2m}\frac{\partial}{\partial x}\left(\frac{\partial\psi}{\partial x}\frac{\partial\psi^*}{\partial x}\right)+V\left(\psi\frac{\partial\psi^*}{\partial x}+\psi^*\frac{\partial\psi}{\partial x}\right)\right], \\
&= \int_{-\infty}^{\infty}dxV\left(\psi\frac{\partial\psi^*}{\partial x}+\psi^*\frac{\partial\psi}{\partial x}\right),
\end{aligned} \tag{632}$$

since $\frac{\partial\psi}{\partial x}\frac{\partial\psi^*}{\partial x} = 0$ when evaluated at $x = \pm\infty$. Therefore,

$$\begin{aligned}\frac{d}{dt}\langle\hat{p}\rangle &= \int_{-\infty}^{\infty} dx \frac{\partial\psi^*\psi}{\partial x} V, \\ &= -\int_{-\infty}^{\infty} dx \psi^* \frac{\partial V}{\partial x}\psi.\end{aligned} \tag{633}$$

Optional Exercise: Real and Imaginary Time Evolution

1. Write a Matlab code to simulate the evolution of a wavepacket bouncing back and forth on a harmonic well, described by the Hamiltonian $H = p^2/(2*m) + V(x)$, with $V(x) = 0.5*x^2$ after initializing the state according to the ground state displaced from its equilibrium position, as follows: $\psi(x,0) = exp(-(x-1)^2/2)/\sqrt[4]{\pi}$.

2. Compute the expectation values of position and momentum as a function of time $x(t)$ and $p(t)$ and compare them to the corresponding classical values obtained by integrating Hamilton's equation with the Velocity-Verlet algorithm:

$$\begin{aligned}p_{j+1} &= p_j + (F(x_j) + F(x_{j+1}))\tau/2, \\ x_{j+1} &= x_j + p_j\tau/m + F(x_j)\tau^2/(2m),\end{aligned} \tag{634}$$

with $x_0 = 1$ and $p_0 = 0$ the initial position and momentum of the harmonic oscillator and x_j and p_j the position and momentum at time $t = j*\tau$, while $F(x_j) = -V'(x_j) = -x_j$.

3. Compute the expectation values of position and momentum as a function of time $x(t)$ and $p(t)$ and compare them to the Ehrenfest trajectory obtained by integrating Hamilton's equation, using mean force:

$$\begin{aligned}\langle p\rangle_{j+1} &= \langle p\rangle_j + (\langle F(x)\rangle_j + \langle F(x)\rangle_{j+1})\tau/2, \\ \langle x\rangle_{j+1} &= \langle x\rangle_j + \langle p\rangle_j\tau/m + \langle F(x)\rangle_j\tau^2/(2m),\end{aligned} \tag{635}$$

with $\langle x\rangle_0 = 1$ and $\langle p\rangle_0 = 0$ the initial position and momentum of the harmonic oscillator and $\langle x\rangle_j$ and $\langle p\rangle_j$ the mean position and momentum at time $t = j*\tau$, while $\langle F(x)\rangle_j = -\langle V'(x)\rangle_j = -\langle x\rangle_j$.

4. Find the ground state of the harmonic well by propagating the wavepacket in imaginary time (*i.e.*, using the propagation time increment $\tau = -it$, with real t) and renormalizing the wave function after each propagation step.

5. Find the first excited state of the harmonic well by propagating the wavepacket in imaginary time (*i.e.*, using the propagation time increment $\tau = -it$, with real t), projecting out the ground state component and renormalizing the wave function after each propagation step.

6. Find the first 9 excited states, iteratively, by imaginary time propagation as in item 4, projecting out lower energy states and renormalizing after each propagation step.

7. Change the potential to that of a Morse oscillator $V(x) = De(1 - exp(-a(x - x_e)))^2$, with $x_e = 0$, $D_e = 8$, and $a = \sqrt{k/(2D_e)}$, where $k = m\omega^2$. Recompute the wave-packet propagation with $x_0 = -0.5$ and $p_0 = 0$ for 100 steps with $\tau = 0.1$ a.u. Compare the expectation values $x(t)$ and $p(t)$ to the corresponding classical and Ehrenfest trajectories obtained according to the Velocity-Verlet algorithm.

Solution: The link (http://ursula.chem.yale.edu/~batista/classes/vvv/HO570.tar) provides a Matlab implementation of the SOFT method as applied to the simulation of evolution of a wavepacket in a harmonic well in real time. In addition, the Matlab code implements the SOFT propagation method to find the lowest 10 eigenstates of the harmonic oscillator by 'evolution' in imaginary time.

Integration of the Bloch equation: The SOFT method can also be applied to integrate the Bloch equation, as implementation for the integration of the Schrödinger equation. Starting with the initial state

$\rho(\mathbf{x}'', \mathbf{x}'; \epsilon)$, defined according to the high-temperature approximation introduced by Eq. (614) we can obtain $\rho(\mathbf{x}'', \mathbf{x}'; 2\epsilon)$, as follows:

$$\rho(\mathbf{x}'', \mathbf{x}'; \tilde{\epsilon} + \epsilon) = \int d\mathbf{x} \langle \mathbf{x}'' | e^{-\tilde{\epsilon}\hat{H}_0} | \mathbf{x} \rangle \rho(\mathbf{x}, \mathbf{x}'; \epsilon). \tag{636}$$

Substituting $\langle \mathbf{x}'' | e^{-\tilde{\epsilon}\hat{H}_0} | \mathbf{x} \rangle$ by its Trotter expansion and following the same steps as before we obtain:

$$\rho(\mathbf{x}'', \mathbf{x}'; \tilde{\epsilon} + \epsilon)) = e^{-iV(\hat{x})\tau/2} \frac{1}{\sqrt{2\pi}} \int dp e^{ix''p} e^{-ip^2\tau/(2m)} \frac{1}{\sqrt{2\pi}} \int dx e^{-ipx} e^{-iV(x)\tau/2} \rho(\mathbf{x}, \mathbf{x}'; \epsilon). \tag{637}$$

Therefore, according to Eq. (637), $\rho(\mathbf{x}'', \mathbf{x}'; \tilde{\epsilon} + \epsilon))$ can be obtained for each $\mathbf{x}'$ by first representing $\rho(\mathbf{x}, \mathbf{x}'; \epsilon)$ on a grid of $\mathbf{x}$, then (i) multiply it by the function $e^{-iV(x)\tau/2}$; (ii) compute the Fourier transform of the resulting product; (iii) multiply it by $e^{-ip^2\tau/(2m)}$; (iv) inverse Fourier transform; and (v) multiply it by $e^{-iV(\hat{x})\tau/2}$. Repeating the sequence of steps (i)–(v) n times generates $\rho(\mathbf{x}'', \mathbf{x}'; \beta)$ when $\tilde{\epsilon} = (\beta - \epsilon)/n$.

Optional Exercise: Implement the SOFT method to obtain $\rho(\mathbf{x}'', \mathbf{x}'; \beta)$ for a Morse oscillator at $\beta = 1$. Compute the partition function $Z = Tr[\hat{\rho}] = \int d\mathbf{x} \rho(\mathbf{x}, \mathbf{x}; \beta)$ and the ensemble average energy $E = Z^{-1} Tr[\hat{\rho}\hat{H}] = Z^{-1} \int d\mathbf{x} d\mathbf{x}' \rho(\mathbf{x}, \mathbf{x}'; \beta) \langle \mathbf{x}' | \hat{H} | \mathbf{x} \rangle$ as a function of β. Then, compute the heat capacity as a function of temperature. Compare your results with the corresponding classical calculations.

48.6 Path Integral Monte Carlo and Ring Polymer Implementation

An alternative expression for the matrix elements of the Boltzmann operator can be obtained by introducing into Eq. (615) the following substitution:

$$\left(\frac{m}{2\pi\epsilon\hbar^2} \right)^{1/2} = (2\pi\hbar)^{-1} \int dp_j e^{-\epsilon \frac{p_j^2}{2m}}, \tag{638}$$

as follows:

$$\begin{aligned} \langle x_0 | e^{-\beta\hat{H}} | x_n \rangle &= (2\pi\hbar)^{-n} \int dp_1 ... \int dp_n e^{-\epsilon \sum_{j=1}^n \frac{p_j^2}{2m}} \\ &\times \int dx_1 ... \int dx_{n-1} e^{-\epsilon \sum_{j=1}^n V(x_j) + \frac{1}{2} m\omega^2 (x_j - x_{j-1})^2}, \end{aligned} \tag{639}$$

to obtain

$$\langle x_0 | e^{-\beta\hat{H}} | x_n \rangle = (2\pi\hbar)^{-n} \int dx_1 ... \int dx_{n-1} \int dp_1 ... \int dp_n \; e^{-\epsilon H_n(x_0; \mathbf{x}^n, \mathbf{p}^n)}, \tag{640}$$

where H_n is the n-bead polymer Hamiltonian

$$H_n(x_0; \mathbf{x}^n, \mathbf{p}^n) = \sum_{j=1}^n \frac{p_j^2}{2m} + V(x_j) + \frac{1}{2} m\omega^2 (x_j - x_{j-1})^2, \tag{641}$$

and

$$\begin{aligned} Z &= \int dx_0 \int dx_n \delta(x_0 - x_n) \langle x_0 | e^{-\beta\hat{H}} | x_n \rangle \\ &= (2\pi\hbar)^{-n} \int d\mathbf{x}^n \int d\mathbf{p}^n \; \delta(x_0 - x_n) e^{-\epsilon H_n(x_0; \mathbf{x}^n, \mathbf{p}^n)}. \end{aligned} \tag{642}$$

48.7 Optional Exercise

Compute $\langle x_0|e^{-\beta\hat{H}}|x_n\rangle$ for the Harmonic oscillator defined by the Hamiltonian

$$\hat{H} = \frac{\hat{p}^2}{2m} + \frac{1}{2}m\omega_0^2\hat{x}^2, \tag{643}$$

by using the Path Integral Monte Carlo method, with $n = 2, 4, 6, 8$ and 10 and show that for larger values of n the calculation converges to the analytic expression:

$$\langle x|e^{-\beta\hat{H}}|x'\rangle = \sqrt{\frac{m\omega}{2\pi\hbar sinh(\beta\hbar\omega)}}exp\left(-\frac{m\omega}{2\hbar sinh(\omega\beta\hbar)}\left[(x^2+x'^2)cosh(\omega\beta) - 2xx'\right]\right), \tag{644}$$

which in the free particle limit ($\omega \to 0$) becomes

$$\langle x|e^{-\beta\hat{H}}|x'\rangle = \sqrt{\frac{m}{2\pi\beta\hbar^2}}exp\left(-\frac{m}{2\beta\hbar^2}\left[(x-x')^2\right]\right), \tag{645}$$

since $sinh(\beta\hbar\omega) \to \beta\hbar\omega$ and $cosh(\beta\hbar\omega) \to 1$.

Matrix elements of the time-evolution operator $e^{-\frac{i}{\hbar}\hat{H}t}$ can be obtained by following the same methodology implemented for the Boltzmann matrix $e^{-\beta\hat{H}\tau}$. We first introduce the variable substitution $\epsilon \equiv i\tau/\hbar$ in Eq. (614) and then we obtain the short-time propagator as follows:

$$\langle x|e^{-\frac{i}{\hbar}\hat{H}\tau}|x'\rangle = \sqrt{\frac{m}{2\pi\hbar i\tau}}e^{\frac{i}{\hbar}\left(\frac{1}{2}m\left[\frac{(x-x')}{\tau}\right]^2 - \frac{1}{2}[V(x)+V(x')]\right)\tau}. \tag{646}$$

Then, we concatenate the short-time propagators introduced by Eq. (646) and we obtain the finite-time propagator,

$$\langle x_0|e^{-\frac{i}{\hbar}\hat{H}t}|x_n\rangle = \int dx_1 ... \int dx_{n-1}\left(\frac{m}{2\pi\hbar i\tau}\right)^{n/2} e^{\frac{i}{\hbar}\left(\sum_{j=1}^{n}\frac{1}{2}m\left[\frac{(x_j-x_{j-1})}{\tau}\right]^2 - \frac{1}{2}\left[V(x_j)+V(x_{j-1})\right]\right)\tau}, \tag{647}$$

which in the limit when $\tau \to 0$ and $n \to \infty$ with $t = n\tau$ becomes,

$$\langle x_0|e^{-\frac{i}{\hbar}\hat{H}t}|x_n\rangle = \int \mathfrak{D}[x(t)]e^{\frac{i}{\hbar}S_c(t)}, \tag{648}$$

where $S_c(t)$ is the classical action associated with the arbitrary trajectory $x(t)$,

$$S_c(t') \equiv \int_0^{t'} dt \quad \left[\frac{1}{2}m\left(\frac{\partial}{\partial t}x(t)\right)^2 - V(x(t))\right], \tag{649}$$

and $\mathfrak{D}[x(t)]$ is defined as follows,

$$\int \mathfrak{D}[x(t)]f(x(t)) \equiv \int dx_1 ... \int dx_{n-1}\left(\frac{m}{2\pi\hbar i\tau}\right)^{n/2} f(x(t)), \tag{650}$$

representing the integral over all paths $x(t)$ from x_0 to x_n, with intermediate coordinates $x_1, x_2, ..., x_{n-1}$ at times $\tau, 2\tau, ..., (n-1)\tau$, respectively.

49 Path Integral Molecular Dynamics and RPMD

The discrete path integral approach [D. Chandler and P. G. Wolynes J. Chem. Phys. 74, 4078 (1981)] implements Feynman's imaginary time path integral for computation of partition functions as discretized in coordinate space by time-slicing the Boltzmann operator and replacing the short-(imaginary)time propagators according to the Trotter approximation, as described in Sec. 48, to obtain the partition function of a 'ring polymer' with harmonically coupled beads:

$$Z_n = \left(\frac{\beta k}{2\pi}\right)^{n/2} \int dx_1 \cdots dx_n e^{-\beta \sum_{j=1}^{n} \left[\frac{1}{2}k(x_j - x_{j+1})^2 + \frac{1}{n}V(x_k)\right]_{x_{n+1}=x_1}}, \tag{651}$$

where $k = m\omega^2$ with $\omega = \sqrt{n}/(\beta\hbar)$ is the 'force constant' that harmonically couples the beads.

The path integral molecular dynamics (PIMD) simulation method [M. Parrinello and A. Rahman, J. Chem. Phys. 80, 860 (1984); R. W. Hall and B. J. Berne, J. Chem. Phys. 81, 3641 (1984)], replaced the prefactor by Gaussians integrals over momentum, as follows:

$$Z_n = \int dx_1 \cdots dx_n dp_1 \cdots dp_n e^{-\beta H_n(\mathbf{x},\mathbf{p})}, \tag{652}$$

where $H_n(\mathbf{x}, \mathbf{p})$ is the Hamiltonian of the 'ring polymer':

$$H_n(\mathbf{x}, \mathbf{p}) = \sum_{j=1}^{n} \left[\frac{p_j^2}{2m'} + \frac{1}{2}m\omega^2(x_j - x_{j+1})^2 + \frac{1}{n}V(x_k) \right]_{x_{n+1}=x_1}, \tag{653}$$

PIMD samples configurations of the beads by using the primitive equation of motion for x_j and p_j:

$$\dot{x}_j = \frac{p_j}{m'}, \qquad \dot{p}_j = -m\omega^2(2x_j - x_{j+1} - x_{j-1}) - \frac{1}{n}\frac{\partial V}{\partial x_j} + \text{heat bath}. \tag{654}$$

The coupling to the heat bath makes it effectively a path integral Monte Carlo (PIMC) method. In PIMD/PIMC, m' is an arbitrary parameter, often called the 'fictitious mass' which could be defined in terms of m to improve convergence [M. E. Tuckerman, G. J. Martyna, M. L. Klein and B. J. Berne J. Chem. Phys. 99, 2796 (1993)] after transforming the primitive variables x_j to the 'staging modes' that diagonalize the harmonic coupling.

The 'path centroid' is the coordinate of the ring center-of-mass:

$$x_c(t) = \frac{1}{n}\sum_{j=1}^{n} x_j(t), \tag{655}$$

which experiences a force given by the 'potential of mean force' [Feynman and Kleinert Phys. Rev. A 34, 5080 (1986)]:

$$W(x_c) = -\beta^{-1}\ln\left[\int dx_1 \cdots dx_n dp_1 \cdots dp_n e^{-\beta H_n(\mathbf{x},\mathbf{p})}\right]. \tag{656}$$

The centroid molecular dynamics (CMD) method [J. Cao and G. A. Voth J. Chem. Phys. 99, 10070 (1993); ibid 100, 5106 (1994); ibid 101, 6168 (1994)], expresses the partition function as the path integral molecular dynamics of the centroid, as follows:

$$Z_n = \left(\frac{\beta k}{2\pi}\right)^{n/2} \int dx_c e^{-\beta W(x_c)}, \tag{657}$$

and samples configurations of the centroid by using the approximate dynamics defined by the potential of mean force, as follows:

$$\dot{x}_c = \frac{p_c}{m}, \qquad \dot{p}_c = -\frac{\partial W}{\partial x_c} + \text{heat bath.} \tag{658}$$

The ring polymer molecular dynamics method [I. R. Craig and D. E. Manolopoulos J. Chem. Phys. 121, 3368 (2004)] is essentially the PIMD method, typically carried out at n times the actual temperature (i.e., $\beta_n = \beta/n$), using a scaled Hamiltonian $nH_n(\mathbf{x}, \mathbf{p})$ and $m' = mn$. Originally, the heat bath was not applied.

Correlation functions are evaluated in terms of the Kubo transformed thermal correlation function,

$$K_{AB}(t) = Z^{-1} tr[\hat{A}^{\beta}_{Kubo}\ e^{i\hat{H}t}\hat{B}e^{-i\hat{H}t}], \tag{659}$$

where

$$\hat{A}^{\beta}_{Kubo} = \frac{1}{\beta}\int_0^{\beta} d\lambda e^{-\beta\hat{H}} e^{\lambda\hat{H}}\hat{A}e^{-\lambda\hat{H}}, \tag{660}$$

approximated by PIMD/RPMD as,

$$K_{AB}(t) \approx Z_n^{-1}\int d\mathbf{p}d\mathbf{x}A_n(0)B_n(t)e^{-\beta H_n(\mathbf{x},\mathbf{p})}, \tag{661}$$

where

$$A_n(0) = \frac{1}{n}\sum_{j=1}^{n} A_j(0), \qquad \text{and} \qquad B_n(t) = \frac{1}{n}\sum_{j=1}^{n} B_j(t), \tag{662}$$

or in terms of its centroid approximation, as follows:

$$K_{AB}(t) \approx Z_n^{-1}\int dp_c dx_c A(x_c(0))B(x_c(t))e^{-\beta\left(\frac{p_c^2}{2m}+W(x_c)\right)}. \tag{663}$$

which requires computing W by PIMD. The derivation of the path integral expression of K_{AB}, introduced by Eq. (661), is provided in Sec. 49.1.

Standard quantum correlation functions,

$$C_{AB}(t) = Z^{-1} tr[e^{-\beta\hat{H}}\hat{A}e^{i\hat{H}t}\hat{B}e^{-i\hat{H}t}], \tag{664}$$

are then obtained from the Fourier transform of the Kubo correlation function $\tilde{K}_{AB}(\omega)$ which is related to the Fourier transform of the standard correlation function $\tilde{C}_{AB}(\omega)$, as follows:

$$\tilde{C}_{AB}(\omega) = \left[\frac{\beta\omega}{1 - e^{-\beta\omega}}\right]\tilde{K}_{AB}(\omega), \tag{665}$$

as shown in the following section (Sec. 49.2).

Obtaining $C_{AB}(t)$ from $K_{AB}(t)$ is convenient since $K_{AB}(t)$ is purely real and invariant under time reversal, just like the classical time correlation functions, which are the natural output of both CMD and PIMD/RPMD calculations. In the linear regime, the response of the system is directly linked to such functions via the Kubo relations [R. Kubo, M. Toda, and N. Hashitsume. Statistical Physics II: Nonequilibrium Statistical Mechanics. Springer, New York, (1985)]. Furthermore, the PIMD $K_{AB}(t)$ reduces to its classical counterpart not only in the classical limit (*i.e.*, $\beta \to 0$) or when $t \to 0$ but also in harmonic systems for which they are exact. The CMD approximation to $K_{AB}(t)$ is also exact for $t = 0$ or for harmonic potentials.

49.1 PIMD/RPMD Expression of Kubo Transform Correlation Function

(This section was contributed by Kenneth Jung.)

Beginning with

$$K_{AB}(t) = \frac{1}{Z\beta}\int_0^\beta d\lambda\, Tr\left[e^{-(\beta-\lambda)\hat{H}}\hat{A}e^{-\lambda\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}\right], \tag{666}$$

where $\beta = 1/(k_BT)$ and Z is the partition function. The integral over lambda can be discretized as

$$\begin{aligned}
K_{AB}^{[N]}(t) &= \frac{1}{Z_N N}\sum_{k=1}^{N} Tr\left[e^{-\beta_N(N-k)\hat{H}}\hat{A}e^{-\beta_N k\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}\right] \\
&= \frac{1}{Z_N N}\sum_{k=1}^{N} Tr\left[\left(e^{-\beta_N\hat{H}}\right)^{N-k}\hat{A}\left(e^{-\beta_N\hat{H}}\right)^{k}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}\right] \\
&= \frac{1}{Z_N N}\sum_{k=1}^{N} Tr\left[\left(e^{-\beta_N\hat{H}}\right)^{N-k-1}e^{-\beta_N\hat{H}}\hat{A}\left(e^{-\beta_N\hat{H}}\right)^{k-1}e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}\right],
\end{aligned} \tag{667}$$

with $\beta_N = \beta/N$. We will now insert $N-1$ identities of the form

$$\hat{1}_t = e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}, \tag{668}$$

in Eq. (667) to give:

$$\begin{aligned}
K_{AB}^{[N]}(t) &= \frac{1}{Z_N N}\sum_{k=1}^{N} Tr\left[\left(e^{-\beta_N\hat{H}}\hat{1}_t\right)^{N-k-1}e^{-\beta_N\hat{H}}\hat{A}\hat{1}_t\left(e^{-\beta_N\hat{H}}\hat{1}_t\right)^{k-1}e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}\right] \\
&= \frac{1}{Z_N N}\sum_{k=1}^{N} Tr\Bigg[\left(e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\right)^{N-k-1}e^{-\beta_N\hat{H}}\hat{A}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar} \\
&\quad\times\left(e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\right)^{k-1}e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}\Bigg],
\end{aligned} \tag{669}$$

The trace will then be expanded in the position basis

$$\begin{aligned}
K_{AB}^{[N]}(t) &= \frac{1}{Z_N N}\sum_{k=1}^{N}\int dq''_N\langle q''_N|\left(e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\right)^{N-k-1}e^{-\beta_N\hat{H}}\hat{A}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar} \\
&\quad\times\left(e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\right)^{k-1}e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}|q''_N\rangle.
\end{aligned} \tag{670}$$

Now consider the jth term in the sum of Eq. (670):

$$\begin{aligned}
&\int dq''_N\langle q''_N|\left(e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\right)^{N-j-1}e^{-\beta_N\hat{H}}\hat{A}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar} \\
&\times\left(e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\right)^{j-1}e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}|q''_N\rangle \\
= &\int dq''_N\langle q''_N|\underbrace{e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\dots e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}}_{N-j-1\text{ terms}}e^{-\beta_N\hat{H}}\hat{A}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar} \\
&\times\underbrace{e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}\dots e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}e^{-i\hat{H}t/\hbar}}_{j-1\text{ terms}}e^{-\beta_N\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}e^{-i\hat{H}t/\hbar}|q''_N\rangle.
\end{aligned} \tag{671}$$

We will insert identities of the form

$$\hat{1}_{q'_l} = \int dq'_l |q'_l\rangle\langle q'_l|, \tag{672}$$

after each $e^{-\beta_N \hat{H}}$ term,

$$\hat{1}_{z_l} = \int dz_l |z_l\rangle\langle z_l|, \tag{673}$$

after each $e^{i\hat{H}t/h}$ term and

$$\hat{1}_{q''_l} = \int dq''_l |q''_l\rangle\langle q''_l|, \tag{674}$$

after each $e^{-i\hat{H}t/h}$ term (except the one next to $|q''_N\rangle$ since the expansion of the trace takes care of this term) to get

$$\begin{aligned}
&= \int dq''_N \langle q''_N|e^{-\beta_N\hat{H}}\hat{1}_{q'_1} e^{i\hat{H}t/\hbar}\hat{1}_{z_1}e^{-i\hat{H}t/\hbar}\hat{1}_{q''_1}\cdots e^{-\beta_N\hat{H}}\hat{1}_{q'_{N-j-1}}e^{i\hat{H}t/\hbar}\hat{1}_{z_{N-j-1}}e^{-i\hat{H}t/\hbar}\hat{1}_{q''_{N-j-1}} \\
&\quad\times e^{-\beta_N\hat{H}}\hat{1}_{q'_{N-j}}\hat{A}e^{i\hat{H}t/\hbar}\hat{1}_{z_{N-j}}e^{-i\hat{H}t/\hbar}\hat{1}_{q''_{N-j}}e^{-\beta_N\hat{H}}\hat{1}_{q'_{N-j+1}}e^{i\hat{H}t/\hbar}\hat{1}_{z_{N-j+1}}e^{-i\hat{H}t/\hbar}\hat{1}_{q''_{N-j+1}} \\
&\quad\times\cdots e^{-\beta_N\hat{H}}\hat{1}_{q'_{N-1}}e^{i\hat{H}t/\hbar}\hat{1}_{z_{N-1}}e^{-i\hat{H}t/\hbar}\hat{1}_{q''_{N-1}}e^{-\beta_N\hat{H}}\hat{1}_{q'_N}e^{i\hat{H}t/\hbar}\hat{1}_{z_N}\hat{B}e^{-i\hat{H}t/\hbar}|q''_N\rangle \\
&= \int d\boldsymbol{q}''\int d\boldsymbol{z}\int d\boldsymbol{q}'\,\langle q''_N|e^{-\beta_N\hat{H}}|q'_1\rangle\langle q'_1|e^{i\hat{H}t/\hbar}|z_1\rangle\langle z_1|e^{-i\hat{H}t/\hbar}|q''_1\rangle\cdots\langle q''_{N-j-2}|e^{-\beta_N\hat{H}}|q'_{N-j-1}\rangle \\
&\quad\times\langle q'_{N-j-1}|e^{i\hat{H}t/\hbar}|z_{N-j-1}\rangle\langle z_{N-j-1}|e^{-i\hat{H}t/\hbar}|q''_{N-j-1}\rangle\langle q''_{N-j-1}|e^{-\beta_N\hat{H}}|q'_{N-j}\rangle \\
&\quad\times\langle q'_{N-j}|\hat{A}e^{i\hat{H}t/\hbar}|z_{N-j}\rangle\langle z_{N-j}|e^{-i\hat{H}t/\hbar}|q''_{N-j}\rangle\langle q''_{N-j}|e^{-\beta_N\hat{H}}|q'_{N-j+1}\rangle\langle q'_{N-j+1}|e^{i\hat{H}t/\hbar}|z_{N-j+1}\rangle \\
&\quad\times\langle z_{N-j+1}|e^{-i\hat{H}t/\hbar}|q''_{N-j+1}\rangle\cdots\langle q''_{N-2}|e^{-\beta_N\hat{H}}|q'_{N-1}\rangle\langle q'_{N-1}|e^{i\hat{H}t/\hbar}|z_{N-1}\rangle\langle z_{N-1}|e^{-i\hat{H}t/\hbar}|q''_{N-1}\rangle \\
&\quad\times\langle q''_{N-1}|e^{-\beta_N\hat{H}}|q'_N\rangle\langle q_N|e^{i\hat{H}t/\hbar}|z_N\rangle\langle z_N|\hat{B}e^{-i\hat{H}t/\hbar}|q''_N\rangle.
\end{aligned}$$

We assume that $\hat{A}$ and $\hat{B}$ are functions of the position operator

$$\begin{aligned}
&= \int d\boldsymbol{q}''\int d\boldsymbol{z}\int d\boldsymbol{q}'\,A(q'_{N-j})B(z_N)\langle q''_N|e^{-\beta_N\hat{H}}|q'_1\rangle\langle q'_1|e^{i\hat{H}t/\hbar}|z_1\rangle\langle z_1|e^{-i\hat{H}t/\hbar}|q''_1\rangle\cdots\langle q''_{N-j-2}|e^{-\beta_N\hat{H}}|q'_{N-j-1}\rangle \\
&\quad\times\langle q'_{N-j-1}|e^{i\hat{H}t/\hbar}|z_{N-j-1}\rangle\langle z_{N-j-1}|e^{-i\hat{H}t/\hbar}|q''_{N-j-1}\rangle\langle q''_{N-j-1}|e^{-\beta_N\hat{H}}|q'_{N-j}\rangle \\
&\quad\times\langle q'_{N-j}|e^{i\hat{H}t/\hbar}|z_{N-j}\rangle\langle z_{N-j}|e^{-i\hat{H}t/\hbar}|q''_{N-j}\rangle\langle q''_{N-j}|e^{-\beta_N\hat{H}}|q'_{N-j+1}\rangle\langle q'_{N-j+1}|e^{i\hat{H}t/\hbar}|z_{N-j+1}\rangle \\
&\quad\times\langle z_{N-j+1}|e^{-i\hat{H}t/\hbar}|q''_{N-j+1}\rangle\cdots\langle q''_{N-2}|e^{-\beta_N\hat{H}}|q'_{N-1}\rangle\langle q'_{N-1}|e^{i\hat{H}t/\hbar}|z_{N-1}\rangle\langle z_{N-1}|e^{-i\hat{H}t/\hbar}|q''_{N-1}\rangle \\
&\quad\times\langle q''_{N-1}|e^{-\beta_N\hat{H}}|q'_N\rangle\langle q_N|e^{i\hat{H}t/\hbar}|z_N\rangle\langle z_N|e^{-i\hat{H}t/\hbar}|q''_N\rangle. \\
&= \int d\boldsymbol{q}''\int d\boldsymbol{z}\int d\boldsymbol{q}'\,A(q'_{N-j})B(z_N)\prod_{l=1}^{N}\langle q''_{l-1}|e^{-\beta_N\hat{H}}|q'_l\rangle\langle q'_l|e^{i\hat{H}t/\hbar}|z_l\rangle\langle z_l|e^{-i\hat{H}t/\hbar}|q''_l\rangle.
\end{aligned} \tag{675}$$

Where we have used the fact that $x_0 = x_N$, $x = \{q', q'', z\}$ to arrive at Eq. (675) and the following notation was introduced

$$\int d\boldsymbol{x} = \prod_{k=1}^{N}\int dx_k. \tag{676}$$

Now that we see how one term in the sum of Eq. (670) looks, it is easy to see that Eq. (670) can be rewritten using the result of Eq. (675) as

$$\begin{aligned}
K_{AB}^{[N]}(t) &= \int d\boldsymbol{q}'' \int dz \int d\boldsymbol{q}' \sum_{k=1}^{N} A(q'_{N-k}) B(z_N) \prod_{l=1}^{N} \langle q''_{l-1}| e^{-\beta_N \hat{H}} |q'_l\rangle \langle q'_l| e^{i\hat{H}t/\hbar} |z_l\rangle \langle z_l| e^{-i\hat{H}t/\hbar} |q''_l\rangle \\
&= \int d\boldsymbol{q}'' \int dz \int d\boldsymbol{q}' \sum_{k=1}^{N} A(q'_{k}) B(z_N) \prod_{l=1}^{N} \langle q''_{l-1}| e^{-\beta_N \hat{H}} |q'_l\rangle \langle q'_l| e^{i\hat{H}t/\hbar} |z_l\rangle \langle z_l| e^{-i\hat{H}t/\hbar} |q''_l\rangle \\
&= \int d\boldsymbol{q}'' \int dz \int d\boldsymbol{q}' A(\boldsymbol{q}') B(z_N) \prod_{l=1}^{N} \langle q''_{l-1}| e^{-\beta_N \hat{H}} |q'_l\rangle \langle q'_l| e^{i\hat{H}t/\hbar} |z_l\rangle \langle z_l| e^{-i\hat{H}t/\hbar} |q''_l\rangle. \qquad (677)
\end{aligned}$$

Since all the blocks are identical and can be permuted we can average the observable $B(z_N)$ over all the blocks to get

$$K_{AB}^{[N]}(t) = \int d\boldsymbol{q}'' \int dz \int d\boldsymbol{q}' A(\boldsymbol{q}') B(\boldsymbol{z}) \prod_{l=1}^{N} \langle q''_{l-1}| e^{-\beta_N \hat{H}} |q'_l\rangle \langle q'_l| e^{i\hat{H}t/\hbar} |z_l\rangle \langle z_l| e^{-i\hat{H}t/\hbar} |q''_l\rangle, \qquad (678)$$

where

$$O(\boldsymbol{x}) = \frac{1}{N} \sum_{k=1}^{N} O(x_k), \qquad (679)$$

is the block-averaged observable.

49.2 Relation of Kubo Transforms to standard TCF's

There is a general relationship between the Kubo correlation functions and the standard correlation functions through Fourier space which can be found by working in the energy representation of the Kubo transform.

$$\begin{aligned}
C_{AB}(t) &= \frac{1}{Z} tr\left[e^{-\beta\hat{H}} \hat{A}(0) \hat{B}(t)\right] \\
&= \frac{1}{Z} tr\left[e^{-\beta\hat{H}} \hat{A}(0) e^{i\hat{H}t/\hbar} \hat{B}(0) e^{-i\hat{H}t/\hbar}\right] \\
&= \frac{1}{Z} \sum_{n,m} \langle n| e^{-\beta\hat{H}} \hat{A}(0) |m\rangle \langle m| e^{i\hat{H}t/\hbar} \hat{B}(0) e^{-i\hat{H}t/\hbar} |n\rangle \qquad (680) \\
&= \frac{1}{Z} \sum_{n,m} e^{-\beta E_n} A_{nm} e^{iE_m t/\hbar} B_{mn} e^{-iE_n t/\hbar} \\
&= \frac{1}{Z} \sum_{n,m} e^{-\beta E_n} e^{i(E_m - E_n)t/\hbar} A_{nm} B_{mn}
\end{aligned}$$

where $A_{nm} = \langle n| \hat{A} |m\rangle$ and $B_{mn} = \langle m| \hat{B} |n\rangle$. This provides a straightforward way to evaluate the correlation function if the Hamiltonian can be efficiently diagonalized. Next we will Fourier transform the correlation function to obtain

$$
\begin{aligned}
\tilde{C}_{AB}(\omega) &= \int dte^{-i\omega t}C_{AB}(t) \\
&= \frac{1}{Z}\int dte^{-i\omega t}\sum_{n,m} e^{-\beta E_n}e^{i(E_m-E_n)t/\hbar}A_{nm}B_{mn} \\
&= \frac{1}{Z}\sum_{n,m} e^{-\beta E_n}A_{nm}B_{mn}\int dte^{-i\omega t}e^{i(E_m-E_n)t/\hbar} \\
&= \frac{1}{Z}\sum_{n,m} e^{-\beta E_n}A_{nm}B_{mn}\delta\left[\omega-(E_m-E_n)\right]
\end{aligned} \tag{681}
$$

Next we will evaluate the Kubo form in the eigen basis

$$
\begin{aligned}
K_{AB}(t) &= \frac{1}{Z\beta}\int_0^\beta d\lambda tr\left[e^{-(\beta-\lambda)\hat{H}}\hat{A}(0)e^{-\lambda\hat{H}}\hat{B}(t)\right] \\
&= \frac{1}{Z\beta}\int_0^\beta d\lambda tr\left[e^{-(\beta-\lambda)\hat{H}}\hat{A}(0)e^{-\lambda\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}(0)e^{-i\hat{H}t/\hbar}\right] \\
&= \frac{1}{Z\beta}\int_0^\beta d\lambda\sum_{n.m}\langle n|\, e^{-(\beta-\lambda)\hat{H}}\hat{A}(0)\,|m\rangle\,\langle m|\, e^{-\lambda\hat{H}}e^{i\hat{H}t/\hbar}\hat{B}(0)e^{-i\hat{H}t/\hbar}\,|n\rangle \\
&= \frac{1}{Z\beta}\int_0^\beta d\lambda\sum_{n.m} e^{-(\beta-\lambda)E_n}A_{nm}e^{-\lambda E_m}e^{iE_mt/\hbar}B_{mn}e^{-iE_nt/\hbar} \\
&= \frac{1}{Z\beta}\sum_{n.m}\int_0^\beta d\lambda e^{\lambda(E_n-E_m)}e^{-\beta E_n}e^{i(E_m-E_n)t/\hbar}A_{nm}B_{mn} \\
&= \frac{1}{Z\beta}\sum_{n.m}\frac{e^{\beta(E_n-E_m)}-1}{(E_n-E_m)}e^{-\beta E_n}e^{i(E_m-E_n)t/\hbar}A_{nm}B_{mn} \\
&= \frac{1}{Z\beta}\sum_{n.m}\frac{1-e^{-\beta(E_m-E_n)}}{(E_m-E_n)}e^{-\beta E_n}e^{i(E_m-E_n)t/\hbar}A_{nm}B_{mn}
\end{aligned} \tag{682}
$$

and again we Fourier transform to get

$$
\begin{aligned}
\tilde{K}_{AB}(\omega) &= \int dte^{-i\omega t}K_{AB}(t) \\
&= \frac{1}{Z\beta}\sum_{n.m}\frac{1-e^{-\beta(E_m-E_n)}}{(E_m-E_n)}e^{-\beta E_n}A_{nm}B_{mn}\int dte^{-i\omega t}e^{i(E_m-E_n)t/\hbar} \\
&= \frac{1}{Z\beta}\sum_{n.m}\frac{1-e^{-\beta(E_m-E_n)}}{(E_m-E_n)}e^{-\beta E_n}A_{nm}B_{mn}\delta\left[\omega-(E_m-E_n)\right]
\end{aligned} \tag{683}
$$

If we compare this result with the Fourier transform of the normal TCF and rename $(E_m - E_n)$ as $\hbar\omega$ we see that the Kubo TCF and the normal TCF are related through their Fourier transforms as

$$
\tilde{C}_{AB}(\omega) = \frac{\beta\hbar\omega}{1-e^{-\beta\hbar\omega}}\tilde{K}_{AB}(\omega) \tag{684}
$$

A Python and Colab

For this class, I recommend working in the Google Colab environment, so you can run your codes in Google's computers. In Colab, everything you need is already installed, or you can upload by mounting your Google drive.

I also recommend strengthening your python knowledge with the excellent tutorial provided in this section, adapted by Kevin Zakka for the Spring 2020 edition of cs231n, and available at Python_tt.ipynb.

Python is a great general-purpose programming language on its own, but with the help of a few popular libraries (numpy, scipy, matplotlib) it becomes a powerful environment for scientific computing. I expect that many of you will have some experience with Python and numpy; for the rest of you, this section will serve as a quick crash course both on the Python programming language and on the use of Python for scientific computing.

In this tutorial, we will cover:

- Basic Python: Basic data types (Containers, Lists, Dictionaries, Sets, Tuples), Functions, Classes
- Numpy: Arrays, Array indexing, Datatypes, Array math, Broadcasting
- Matplotlib: Plotting, Subplots, Images
- IPython: Creating notebooks, Typical workflows

A.1 A Brief Note on Python Versions

As of Janurary 1, 2020, Python has officially dropped support for `python2`. We'll be using Python 3.7 for this iteration of the course. You can check your Python version at the command line by running `python --version`. In Colab, we can enforce the Python version by clicking `Runtime -> Change Runtime Type` and selecting `python3`. Note that as of April 2020, Colab uses Python 3.6.9 which should run everything without any errors.

[6]:
```
!python --version
```

```
Python 3.6.9
```

A.1.1 Basics of Python

Python is a high-level, dynamically typed multiparadigm programming language. Python code is often said to be almost like pseudocode, since it allows you to express very powerful ideas in very few lines of code while being very readable. As an example, here is an implementation of the classic quicksort algorithm in Python:

[7]:
```
def quicksort(arr):
    if len(arr) <= 1:
        return arr
    pivot = arr[len(arr) // 2]
    left = [x for x in arr if x < pivot]
    middle = [x for x in arr if x == pivot]
    right = [x for x in arr if x > pivot]
    return quicksort(left) + middle + quicksort(right)

print(quicksort([3,6,8,10,1,2,1]))
```

```
[1, 1, 2, 3, 6, 8, 10]
```

A.1.1.1 Basic data types

A.1.1.1.1 Numbers Integers and floats work as you would expect from other languages:

```
[8]: x = 3
     print(x, type(x))
```

```
3 <class 'int'>
```

```
[9]: print(x + 1)    # Addition
     print(x - 1)    # Subtraction
     print(x * 2)    # Multiplication
     print(x ** 2)   # Exponentiation
```

```
4
2
6
9
```

```
[10]: x += 1
      print(x)
      x *= 2
      print(x)
```

```
4
8
```

```
[11]: y = 2.5
      print(type(y))
      print(y, y + 1, y * 2, y ** 2)
```

```
<class 'float'>
2.5 3.5 5.0 6.25
```

Note that unlike many languages, Python does not have unary increment (x++) or decrement (x–) operators.

Python also has built-in types for long integers and complex numbers; you can find all of the details in the documentation.

A.1.1.1.2 Booleans Python implements all of the usual operators for Boolean logic, but uses English words rather than symbols (`&&`, `||`, etc.):

```
[12]: t, f = True, False
      print(type(t))
```

```
<class 'bool'>
```

Now we let's look at the operations:

[13]:
```
print(t and f) # Logical AND;
print(t or f)  # Logical OR;
print(not t)   # Logical NOT;
print(t != f)  # Logical XOR;
```

```
False
True
False
True
```

A.1.1.1.3 Strings

[14]:
```
hello = 'hello'    # String literals can use single quotes
world = "world"    # or double quotes; it does not matter
print(hello, len(hello))
```

```
hello 5
```

[15]:
```
hw = hello + ' ' + world  # String concatenation
print(hw)
```

```
hello world
```

[16]:
```
hw12 = '{} {} {}'.format(hello, world, 12)  # string formatting
print(hw12)
```

```
hello world 12
```

String objects have a bunch of useful methods; for example:

[17]:
```
s = "hello"
print(s.capitalize())  # Capitalize a string
print(s.upper())       # Convert a string to uppercase; prints "HELLO"
print(s.rjust(7))      # Right-justify a string, padding with spaces
print(s.center(7))     # Center a string, padding with spaces
print(s.replace('l', '(ell)'))  # Replace all instances of one␣
 ↪substring with another
print('  world '.strip())  # Strip leading and trailing whitespace
```

```
Hello
HELLO
  hello
 hello
he(ell)(ell)o
world
```

You can find a list of all string methods in the documentation.

A.1.1.2 Containers Python includes several **built-in container types**: lists, dictionaries, sets, and tuples.

1. List item
2. List item
3. List item
4. List item

A.1.1.2.1 Lists A list is the Python equivalent of an array, but is resizeable and can contain elements of different types:

[18]:
```
xs = [3, 1, 2]    # Create a list
print(xs, xs[2])
print(xs[-1])      # Negative indices count from the end of the list;␣
 ↪prints "2"
```

```
[3, 1, 2] 2
2
```

Lists can be generated from arrays, as follows:

[20]:
```
import numpy as np

int_list = [] # list initialization
int_list = [0,0,1,2,3] # list with commas
int_list.append(4) # add 4 to end of the list
int_list.pop(2) # remove element with index 2

int_list2 = list(range(5)) # make list [0,1,2,3,4]
int_array = np.array(int_list) # make array [] with no commas: [0 1 2␣
 ↪3 4]
int_array2 = np.arange(5)  # make array [] with no commas: [0 1 2 3 4]
int_list2 = int_array.tolist()  # convert array to list

first = 0
last = 4
float_array = np.linspace(first,last,num=5)

print('int_list=',int_list)
print('int_list2=',int_list2)
print('int_array=',int_array)
print('int_array2=',int_array2)
print('float_array=',float_array)
```

```
int_list= [0, 0, 2, 3, 4]
int_list2= [0, 0, 2, 3, 4]
int_array= [0 0 2 3 4]
```

```
int_array2= [0 1 2 3 4]
float_array= [0. 1. 2. 3. 4.]
```

```
[21]: xs[2] = 'foo'     # Lists can contain elements of different types
      print(xs)
```

```
[3, 1, 'foo']
```

Lists have methods, including append, insert, remove, sort

```
[22]: xs.append('bar') # Add a new element to the end of the list
      print(xs)
```

```
[3, 1, 'foo', 'bar']
```

```
[23]: x = xs.pop()      # Remove and return the last element of the list
      print(x, xs)
```

```
bar [3, 1, 'foo']
```

As usual, you can find all the gory details about lists in the documentation.

A.1.1.2.2 Slicing In addition to accessing list elements one at a time, Python provides concise syntax to access sublists; this is known as slicing:

```
[24]: nums = list(range(5))     # range is a built-in function that creates␣
       ↪a list of integers
      print(nums)               # Prints "[0, 1, 2, 3, 4]"
      print(nums[2:4])          # Get a slice from index 2 to 4 (exclusive);␣
       ↪prints "[2, 3]"
      print(nums[2:])           # Get a slice from index 2 to the end; prints␣
       ↪"[2, 3, 4]"
      print(nums[:2])           # Get a slice from the start to index 2␣
       ↪(exclusive); prints "[0, 1]"
      print(nums[:])            # Get a slice of the whole list; prints ["0, 1,␣
       ↪2, 3, 4]"
      print(nums[:-1])          # Slice indices can be negative; prints ["0, 1,␣
       ↪2, 3]"
      nums[2:4] = [8, 9] # Assign a new sublist to a slice
      print(nums)               # Prints "[0, 1, 8, 9, 4]"
```

```
[0, 1, 2, 3, 4]
[2, 3]
[2, 3, 4]
[0, 1]
[0, 1, 2, 3, 4]
[0, 1, 2, 3]
[0, 1, 8, 9, 4]
```

A.1.1.2.3 Loops You can loop over the elements of a list like this:

```
[25]: animals = ['cat', 'dog', 'monkey']
      for animal in animals:
          print(animal)
```

```
cat
dog
monkey
```

If you want access to the index of each element within the body of a loop, use the built-in `enumerate` function:

```
[26]: animals = ['cat', 'dog', 'monkey']
      for idx, animal in enumerate(animals):
          print('#{}: {}'.format(idx + 1, animal))
```

```
#1: cat
#2: dog
#3: monkey
```

A.1.1.2.4 List comprehensions: When programming, frequently we want to transform one type of data into another. As a simple example, consider the following code that computes square numbers:

```
[27]: nums = [0, 1, 2, 3, 4]
      squares = []
      for x in nums:
          squares.append(x ** 2)
      print(squares)
```

```
[0, 1, 4, 9, 16]
```

You can make this code simpler using a list comprehension:

```
[28]: nums = [0, 1, 2, 3, 4]
      squares = [x ** 2 for x in nums]
      print(squares)
```

```
[0, 1, 4, 9, 16]
```

List comprehensions can also contain conditions:

```
[29]: nums = [0, 1, 2, 3, 4]
      even_squares = [x ** 2 for x in nums if x % 2 == 0]
      print(even_squares)
```

```
[0, 4, 16]
```

A.1.1.2.5 Dictionaries A dictionary stores (key, value) pairs, similar to a `Map` in Java or an object in Javascript. You can use it like this:

```
[30]: d = {}
      d = {'cat': 'cute', 'dog': 'furry'}  # Create a new dictionary with␣
       ↪some data
      print(d['cat'])        # Get an entry from a dictionary; prints "cute"
      print('cat' in d)      # Check if a dictionary has a given key; prints␣
       ↪"True"
```

```
cute
True
```

```
[31]: d['fish'] = 'wet'     # Set an entry in a dictionary
      print(d['fish'])       # Prints "wet"
```

```
wet
```

```
[32]: print(d['monkey'])  # KeyError: 'monkey' not a key of d
```

```
           ␣
 ↪---------------------------------------------------------------------------

       KeyError                                  Traceback (most␣
 ↪recent call last)
```

```
[33]: print(d.get('monkey', 'N/A'))  # Get an element with a default;␣
       ↪prints "N/A"
      print(d.get('fish', 'N/A'))    # Get an element with a default;␣
       ↪prints "wet"
```

```
N/A
wet
```

```
[34]: del d['fish']         # Remove an element from a dictionary
      print(d.get('fish', 'N/A')) # "fish" is no longer a key; prints "N/A"
```

```
N/A
```

You can find all you need to know about dictionaries in the documentation.
It is easy to iterate over the keys in a dictionary:

```
[35]: d = {'person': 2, 'cat': 4, 'spider': 8}
      for animal, legs in d.items():
          print('A {} has {} legs'.format(animal, legs))
```

```
A person has 2 legs
A cat has 4 legs
A spider has 8 legs
```

Add pairs to the dictionary

```
[36]: d['bird']=2
```

List keys

```
[37]: d.keys()
```

```
[37]: dict_keys(['person', 'cat', 'spider', 'bird'])
```

List Values

```
[38]: d.values()
```

```
[38]: dict_values([2, 4, 8, 2])
```

Query values from keys

```
[39]: d['bird']
```

```
[39]: 2
```

Dictionary comprehensions: These are similar to list comprehensions, but allow you to easily construct dictionaries. For example:

```
[40]: nums = [0, 1, 2, 3, 4]
      even_num_to_square = {x: x ** 2 for x in nums if x % 2 == 0}
      print(even_num_to_square)
```

```
{0: 0, 2: 4, 4: 16}
```

Convert array to list

A.1.1.2.6 Sets (like dictionaries but with no values, add & remove)

A set is an unordered collection of distinct elements. As a simple example, consider the following:

```
[41]: animals = {'cat', 'dog'}
      print('cat' in animals)    # Check if an element is in a set; prints␣
       ↪"True"
      print('fish' in animals)   # prints "False"
```

```
True
False
```

```
[42]: animals.add('fish')       # Add an element to a set
      print('fish' in animals)
      print(len(animals))        # Number of elements in a set;
```

```
True
3
```

```
[43]: animals.add('cat')       # Adding an element that is already in the␣
       ↪set does nothing
      print(len(animals))
      animals.remove('cat')    # Remove an element from a set
      print(len(animals))
```

```
3
2
```

Loops: Iterating over a set has the same syntax as iterating over a list; however since sets are unordered, you cannot make assumptions about the order in which you visit the elements of the set:

```
[44]: animals = {'cat', 'dog', 'fish'}
      for idx, animal in enumerate(animals):
          print('#{}: {}'.format(idx + 1, animal))
```

```
#1: fish
#2: dog
#3: cat
```

Set comprehensions: Like lists and dictionaries, we can easily construct sets using set comprehensions:

```
[45]: from math import sqrt
      print({int(sqrt(x)) for x in range(30)})
```

```
{0, 1, 2, 3, 4, 5}
```

A.1.1.2.7 Tuples A tuple is an (immutable) ordered list of values. A tuple is in many ways similar to a list; one of the most important differences is that tuples can be used as keys in dictionaries and as elements of sets, while lists cannot. Here is a simple example:

```
[46]: d = {(x, x + 1): x for x in range(10)}  # Create a dictionary with␣
       ↪tuple keys
      print(d)

      tt = ()  # initialization of empty tuple
      t1 = (66,)  # initialization of tuple with a single value
      t = (5, 6)       # Create a tuple
      tt = tt+t1+t
      print("tt=",tt)
      print("tt[2]=",tt[2])
      print("tt[1:3]=",tt[1:3])
      print("66 in tt", 66 in tt)

      print(type(t))
      print(d[t])
      print(d[(1, 2)])
```

```
{(0, 1): 0, (1, 2): 1, (2, 3): 2, (3, 4): 3, (4, 5): 4, (5, 6): 5, (6,␣
↪7): 6,
(7, 8): 7, (8, 9): 8, (9, 10): 9}
tt= (66, 5, 6)
tt[2]= 6
tt[1:3]= (5, 6)
66 in tt True
<class 'tuple'>
5
1
```

```
[47]: t[0] = 1
```

```
        ␣
 ↪---------------------------------------------------------------------------

        TypeError                                 Traceback (most␣
 ↪recent call last)

        <ipython-input-47-c8aeb8cd20ae> in <module>()
    ----> 1 t[0] = 1


        TypeError: 'tuple' object does not support item assignment
```

A.1.1.3 Functions

Python functions are defined using the `def` keyword. For example:

```
[48]: def sign(x):
          if x > 0:
              return 'positive'
          elif x < 0:
              return 'negative'
          else:
              return 'zero'

      for x in [-1, 0, 1]:
          print(sign(x))
```

```
negative
zero
positive
```

We will often define functions to take optional keyword arguments, like this:

```
[49]: def hello(name, loud=False):
          if loud:
              print('HELLO, {}'.format(name.upper()))
```

```
    else:
        print('Hello, {}!'.format(name))

hello('Bob')
hello('Fred', loud=True)
```

```
Hello, Bob!
HELLO, FRED
```

A.1.1.4 Classes A new class creates a new type of object, bounding data and functionality that allows new instances of the type made. Each class instance can have attributes attached to it, so we can make class instances as well as instances to variables and methods for maintaining the state of the class. Instances of the method can have attributes and can modify the state of the class, as clearly described by the documentation.

The syntax for defining classes in Python is straightforward:

[50]:
```
class Greeter:
    """ My greeter class """
    # Constructor (method of construction of class in a specific␣
 ↪state)
    v1 ='papa' # class variable shared by all instances
    def __init__(self, name_inp): # name_inp: argument given to␣
 ↪Greeter for class instantiation
        self.name = name_inp  # Create an instance variable␣
 ↪maintaining the state
                              # instance variables are unique to each␣
 ↪instance
    # Instance method
    # note that the first argument of the function method is the␣
 ↪instance object
    def greet(self, loud=False):
        if loud:
          print('HELLO, {}'.format(self.name.upper()))
          self.name = 'Haote'
        else:
          print('Hello, {}!'.format(self.name))
          self.name = 'Victor'

# Class instantiation (returning a new instance of the class assigned␣
 ↪to g):
# Constructs g of type Greeter & initialzes its state
# as defined by the class variables (does not execute methods)
g = Greeter('Fred')

# Call an instance method of the class in its current state:
# prints "Hello, Fred!" and updates state varible to 'Victor' since␣
 ↪loud=False
```

```
g.greet() # equivalent to Greeter.greet(g) since the first arg of␣
 ↪greet is g

# Call an instance method; prints "HELLO, VICTOR" and updates␣
 ↪variable to 'Haote'
g.greet(loud=True)    # equivalent to Greeter.greet(g,loud=True)
                      #since the first arg of greet is g
print(g.v1)
g.greet()             # Call an instance method; prints "Hello, Haote!"
                      # A method object is created by packing␣
 ↪(pointers to) the
                      # instance object g and the function object greet

g2 = Greeter('Lea')    # Class instance reinitializes variable to 'Lea'

g2.greet()             # Call an instance method; prints "Hello, Lea!"
g2.__doc__
g2.x=20                # Data attributes spring into existence upon␣
 ↪assignment
print(g2.x)
del g2.x               # Deletes attribute
g2.v1
```

```
Hello, Fred!
HELLO, VICTOR
papa
Hello, Haote!
Hello, Lea!
20
```

[50]:
```
'papa'
```

For loops (iterators). Behind the scenes, the for statement calls iter() on the container object.

[51]:
```
for element in [1,2,3]: # elements of list
  print(element)
for element in (1,2,3): # elements of tuple
  print(element)
for key in {'first':1, 'second':2, 'third':3}: # elements of␣
 ↪dictionary
  print('key=',key)
for char in '1234':
  print(char)
#for line in open(''myfile.txt)
#   print(line,end='')
```

```
1
2
```

```
3
1
2
3
key= first
key= second
key= third
1
2
3
4
```

A.1.1.5 Modules A module is a .py file containing Python definitions and statements that can be imported into a Python script, as described in the Python documentation.

As an example, after mounting your Google drive as described by the Navigating_tutorial.ipynb Jupyter notebook, use a text editor and write a module with the line:

```
[52]: greeting = "Good Morning!"
```

Save the document with the name mymod.py

Next, go the the folder where you saved that file and open a notebook with the lines:

```
[ ]: import mymod as my
     print(my.greeting)
```

you will see that the notebook has imported the variable greetingfrom the module `mymod.py` and has invoked the variable as an attribute of the module mymod that was imported as my when printing `Good Morning!!`.

Modules are very convenient since they allow you to import variables, functions and classes that you might have developed for previous projects, without having to copy them into each program. So, you can build from previous projects, or split your work into several files for easier maintenance.

Within a module, the module's name (as a string) is available as the value of the global variable `__name__`.

A.1.2 Numpy

Numpy is the core library for scientific computing in Python. It provides a high-performance multidimensional array object, and tools for working with these arrays. If you are already familiar with MATLAB, you might find this tutorial useful to get started with Numpy.

To use Numpy, we first need to import the `numpy` package:

```
[54]: import numpy as np
```

A.1.2.1 Arrays A numpy array is a grid of values, all of the same type, and is indexed by a tuple of nonnegative integers. The number of dimensions is the rank of the array; the shape of an array is a tuple of integers giving the size of the array along each dimension.

We can initialize numpy arrays from nested Python lists, and access elements using square brackets:

```
[55]: a = np.array([1, 2, 3])  # Create a rank 1 array
      print(type(a), a.shape, a[0], a[1], a[2])
      a[0] = 5                  # Change an element of the array
```

```
print(a)
```

```
<class 'numpy.ndarray'> (3,) 1 2 3
[5 2 3]
```

```
[56]: b = np.array([[1,2,3],[4,5,6]])    # Create a rank 2 array
print(b)
```

```
[[1 2 3]
 [4 5 6]]
```

```
[57]: print(b.shape)
print(b[0, 0], b[0, 1], b[1, 0])
```

```
(2, 3)
1 2 4
```

Numpy also provides many functions to create arrays:

```
[58]: a = np.zeros((2,2))   # Create an array of all zeros
print(a)
```

```
[[0. 0.]
 [0. 0.]]
```

```
[59]: b = np.ones((1,2))    # Create an array of all ones
print(b)
```

```
[[1. 1.]]
```

```
[60]: c = np.full((2,2), 7) # Create a constant array
print(c)
```

```
[[7 7]
 [7 7]]
```

```
[61]: d = np.eye(2)         # Create a 2x2 identity matrix
print(d)
```

```
[[1. 0.]
 [0. 1.]]
```

```
[62]: e = np.random.random((2,2)) # Create an array filled with random␣
 ↪values
print(e)
```

```
[[0.32071297 0.96986179]
 [0.32331846 0.50510489]]
```

A.1.2.2 Array indexing Numpy offers several ways to index into arrays.

Slicing: Similar to Python lists, numpy arrays can be sliced. Since arrays may be multidimensional, you must specify a slice for each dimension of the array:

[63]:
```
import numpy as np

# Create the following rank 2 array with shape (3, 4)
# [[ 1  2  3  4]
#  [ 5  6  7  8]
#  [ 9 10 11 12]]
a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])

# Use slicing to pull out the subarray consisting of the first 2 rows
# and columns 1 and 2; b is the following array of shape (2, 2):
# [[2 3]
#  [6 7]]
b = a[:2, 1:3]
print(b)
```

```
[[2 3]
 [6 7]]
```

A slice of an array is a view into the same data, so modifying it will modify the original array.

[64]:
```
print(a[0, 1])
b[0, 0] = 77    # b[0, 0] is the same piece of data as a[0, 1]
print(a[0, 1])
```

```
2
77
```

You can also mix integer indexing with slice indexing. However, doing so will yield an array of lower rank than the original array. Note that this is quite different from the way that MATLAB handles array slicing:

[65]:
```
# Create the following rank 2 array with shape (3, 4)
a = np.array([[1,2,3,4], [5,6,7,8], [9,10,11,12]])
print(a)
```

```
[[ 1  2  3  4]
 [ 5  6  7  8]
 [ 9 10 11 12]]
```

Two ways of accessing the data in the middle row of the array. Mixing integer indexing with slices yields an array of lower rank, while using only slices yields an array of the same rank as the original array:

[66]:
```
row_r1 = a[1, :]    # Rank 1 view of the second row of a
row_r2 = a[1:2, :]  # Rank 2 view of the second row of a
row_r3 = a[[1], :]  # Rank 2 view of the second row of a
print(row_r1, row_r1.shape)
print(row_r2, row_r2.shape)
print(row_r3, row_r3.shape)
```

```
[5 6 7 8] (4,)
[[5 6 7 8]] (1, 4)
[[5 6 7 8]] (1, 4)
```

```
[67]: # We can make the same distinction when accessing columns of an array:
col_r1 = a[:, 1]
col_r2 = a[:, 1:2]
print(col_r1, col_r1.shape)
print()
print(col_r2, col_r2.shape)
```

```
[ 2  6 10] (3,)

[[ 2]
 [ 6]
 [10]] (3, 1)
```

Integer array indexing: When you index into numpy arrays using slicing, the resulting array view will always be a subarray of the original array. In contrast, integer array indexing allows you to construct arbitrary arrays using the data from another array. Here is an example:

```
[68]: a = np.array([[1,2], [3, 4], [5, 6]])

# An example of integer array indexing.
# The returned array will have shape (3,) and
print(a[[0, 1, 2], [0, 1, 0]])

# The above example of integer array indexing is equivalent to this:
print(np.array([a[0, 0], a[1, 1], a[2, 0]]))
```

```
[1 4 5]
[1 4 5]
```

```
[69]: # When using integer array indexing, you can reuse the same
# element from the source array:
print(a[[0, 0], [1, 1]])

# Equivalent to the previous integer array indexing example
print(np.array([a[0, 1], a[0, 1]]))
```

```
[2 2]
[2 2]
```

One useful trick with integer array indexing is selecting or mutating one element from each row of a matrix:

```
[70]: # Create a new array from which we will select elements
      a = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
      print(a)
```

```
[[ 1  2  3]
 [ 4  5  6]
 [ 7  8  9]
 [10 11 12]]
```

```
[71]: # Create an array of indices
      b = np.array([0, 2, 0, 1])

      # Select one element from each row of a using the indices in b
      print(a[np.arange(4), b])  # Prints "[ 1  6  7 11]"
```

```
[ 1  6  7 11]
```

```
[72]: # Mutate one element from each row of a using the indices in b
      a[np.arange(4), b] += 10
      print(a)
```

```
[[11  2  3]
 [ 4  5 16]
 [17  8  9]
 [10 21 12]]
```

Boolean array indexing: Boolean array indexing lets you pick out arbitrary elements of an array. Frequently this type of indexing is used to select the elements of an array that satisfy some condition. Here is an example:

```
[73]: import numpy as np

      a = np.array([[1,2], [3, 4], [5, 6]])

      bool_idx = (a > 2)   # Find the elements of a that are bigger than 2;
                           # this returns a numpy array of Booleans of the␣
       ↪same
                           # shape as a, where each slot of bool_idx tells
                           # whether that element of a is > 2.

      print(bool_idx)
```

```
[[False False]
 [ True  True]
 [ True  True]]
```

```
[74]: # We use boolean array indexing to construct a rank 1 array
      # consisting of the elements of a corresponding to the True values
      # of bool_idx
      print(a[bool_idx])

      # We can do all of the above in a single concise statement:
      print(a[a > 2])
```

```
[3 4 5 6]
[3 4 5 6]
```

For brevity we have left out a lot of details about numpy array indexing; if you want to know more you should read the documentation.

A.1.2.3 Datatypes Every numpy array is a grid of elements of the same type. Numpy provides a large set of numeric datatypes that you can use to construct arrays. Numpy tries to guess a datatype when you create an array, but functions that construct arrays usually also include an optional argument to explicitly specify the datatype. Here is an example:

```
[75]: x = np.array([1, 2])  # Let numpy choose the datatype
      y = np.array([1.0, 2.0])  # Let numpy choose the datatype
      z = np.array([1, 2], dtype=np.int64)  # Force a particular datatype

      print(x.dtype, y.dtype, z.dtype)
```

```
int64 float64 int64
```

You can read all about numpy datatypes in the documentation.

A.1.2.4 Array math Basic mathematical functions operate elementwise on arrays, and are available both as operator overloads and as functions in the numpy module:

```
[76]: x = np.array([[1,2],[3,4]], dtype=np.float64)
      y = np.array([[5,6],[7,8]], dtype=np.float64)

      # Elementwise sum; both produce the array
      print(x + y)
      print(np.add(x, y))
```

```
[[ 6.  8.]
 [10. 12.]]
[[ 6.  8.]
 [10. 12.]]
```

```
[77]: # Elementwise difference; both produce the array
      print(x - y)
      print(np.subtract(x, y))
```

```
[[-4. -4.]
 [-4. -4.]]
[[-4. -4.]
 [-4. -4.]]
```

```
[78]: # Elementwise product; both produce the array
      print(x * y)
      print(np.multiply(x, y))
```

```
[[ 5. 12.]
 [21. 32.]]
[[ 5. 12.]
 [21. 32.]]
```

```
[79]: # Elementwise division; both produce the array
      # [[ 0.2         0.33333333]
      #  [ 0.42857143  0.5       ]]
      print(x / y)
      print(np.divide(x, y))
```

```
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
[[0.2        0.33333333]
 [0.42857143 0.5       ]]
```

```
[80]: # Elementwise square root; produces the array
      # [[ 1.          1.41421356]
      #  [ 1.73205081  2.        ]]
      print(np.sqrt(x))
```

```
[[1.         1.41421356]
 [1.73205081 2.        ]]
```

Note that unlike MATLAB, * is elementwise multiplication, not matrix multiplication. We instead use the dot function to compute inner products of vectors, to multiply a vector by a matrix, and to multiply matrices. dot is available both as a function in the numpy module and as an instance method of array objects:

```
[81]: x = np.array([[1,2],[3,4]])
      y = np.array([[5,6],[7,8]])

      v = np.array([9,10])
      w = np.array([11, 12])

      # Inner product of vectors; both produce 219
      print(v.dot(w))
      print(np.dot(v, w))
```

```
219
219
```

You can also use the `@` operator which is equivalent to numpy's `dot` operator.

[82]:
```
print(v @ w)
```

```
219
```

[83]:
```
# Matrix / vector product; both produce the rank 1 array [29 67]
print(x.dot(v))
print(np.dot(x, v))
print(x @ v)
```

```
[29 67]
[29 67]
[29 67]
```

[84]:
```
# Matrix / matrix product; both produce the rank 2 array
# [[19 22]
#  [43 50]]
print(x.dot(y))
print(np.dot(x, y))
print(x @ y)
```

```
[[19 22]
 [43 50]]
[[19 22]
 [43 50]]
[[19 22]
 [43 50]]
```

Numpy provides many useful functions for performing computations on arrays; one of the most useful is `sum`:

[85]:
```
x = np.array([[1,2],[3,4]])

print(np.sum(x))  # Compute sum of all elements; prints "10"
print(np.sum(x, axis=0))  # Compute sum of each column; prints "[4 6]"
print(np.sum(x, axis=1))  # Compute sum of each row; prints "[3 7]"
```

```
10
[4 6]
[3 7]
```

You can find the full list of mathematical functions provided by numpy in the documentation.

Apart from computing mathematical functions using arrays, we frequently need to reshape or otherwise manipulate data in arrays. The simplest example of this type of operation is transposing a matrix; to transpose a matrix, simply use the T attribute of an array object:

[86]:
```
print(x)
print("transpose\n", x.T)
```

```
[[1 2]
 [3 4]]
transpose
 [[1 3]
 [2 4]]
```

[87]:
```
v = np.array([[1,2,3]])
print(v )
print("transpose\n", v.T)
```

```
[[1 2 3]]
transpose
 [[1]
 [2]
 [3]]
```

A.1.2.5 Broadcasting Broadcasting is a powerful mechanism that allows numpy to work with arrays of different shapes when performing arithmetic operations. Frequently we have a smaller array and a larger array, and we want to use the smaller array multiple times to perform some operation on the larger array.

For example, suppose that we want to add a constant vector to each row of a matrix. We could do it like this:

[88]:
```
# We will add the vector v to each row of the matrix x,
# storing the result in the matrix y
x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
v = np.array([1, 0, 1])
y = np.empty_like(x)   # Create an empty matrix with the same␣
 ↪as x

# Add the vector v to each row of the matrix x with an explicit loop
for i in range(4):
    y[i, :] = x[i, :] + v

print(y)
```

```
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

This works; however when the matrix `x` is very large, computing an explicit loop in Python could be slow. Note that adding the vector v to each row of the matrix `x` is equivalent to forming a matrix `vv` by stacking multiple copies of `v` vertically, then performing elementwise summation of `x` and `vv`. We could implement this approach like this:

```
[89]: vv = np.tile(v, (4, 1))   # Stack 4 copies of v on top of each other
      print(vv)                  # Prints "[[1 0 1]
                                 #          [1 0 1]
                                 #          [1 0 1]
                                 #          [1 0 1]]"
```

```
[[1 0 1]
 [1 0 1]
 [1 0 1]
 [1 0 1]]
```

```
[90]: y = x + vv  # Add x and vv elementwise
      print(y)
```

```
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

Numpy broadcasting allows us to perform this computation without actually creating multiple copies of v. Consider this version, using broadcasting:

```
[91]: import numpy as np

      # We will add the vector v to each row of the matrix x,
      # storing the result in the matrix y
      x = np.array([[1,2,3], [4,5,6], [7,8,9], [10, 11, 12]])
      v = np.array([1, 0, 1])
      y = x + v  # Add v to each row of x using broadcasting
      print(y)
```

```
[[ 2  2  4]
 [ 5  5  7]
 [ 8  8 10]
 [11 11 13]]
```

The line `y = x + v` works even though `x` has shape `(4, 3)` and `v` has shape `(3,)` due to broadcasting; this line works as if v actually had shape `(4, 3)`, where each row was a copy of `v`, and the sum was performed elementwise.

Broadcasting two arrays together follows these rules:

1. If the arrays do not have the same rank, prepend the shape of the lower rank array with 1s until both shapes have the same length.
2. The two arrays are said to be compatible in a dimension if they have the same size in the dimension, or if one of the arrays has size 1 in that dimension.
3. The arrays can be broadcast together if they are compatible in all dimensions.
4. After broadcasting, each array behaves as if it had shape equal to the elementwise maximum of shapes of the two input arrays.

5. In any dimension where one array had size 1 and the other array had size greater than 1, the first array behaves as if it were copied along that dimension

If this explanation does not make sense, try reading the explanation from the documentation or this explanation.

Functions that support broadcasting are known as universal functions. You can find the list of all universal functions in the documentation.

Here are some applications of broadcasting:

[92]:
```
# Compute outer product of vectors
v = np.array([1,2,3])  # v has shape (3,)
w = np.array([4,5])    # w has shape (2,)
# To compute an outer product, we first reshape v to be a column
# vector of shape (3, 1); we can then broadcast it against w to yield
# an output of shape (3, 2), which is the outer product of v and w:

print(np.reshape(v, (3, 1)) * w)
```

```
[[ 4  5]
 [ 8 10]
 [12 15]]
```

[93]:
```
# Add a vector to each row of a matrix
x = np.array([[1,2,3], [4,5,6]])
# x has shape (2, 3) and v has shape (3,) so they broadcast to (2, 3),
# giving the following matrix:

print(x + v)
```

```
[[2 4 6]
 [5 7 9]]
```

[94]:
```
# Add a vector to each column of a matrix
# x has shape (2, 3) and w has shape (2,).
# If we transpose x then it has shape (3, 2) and can be broadcast
# against w to yield a result of shape (3, 2); transposing this result
# yields the final result of shape (2, 3) which is the matrix x with
# the vector w added to each column. Gives the following matrix:

print((x.T + w).T)
```

```
[[ 5  6  7]
 [ 9 10 11]]
```

[95]:
```
# Another solution is to reshape w to be a row vector of shape (2, 1);
# we can then broadcast it directly against x to produce the same
# output.
```

```
print(x + np.reshape(w, (2, 1)))
```

```
[[ 5  6  7]
 [ 9 10 11]]
```

[96]:
```
# Multiply a matrix by a constant:
# x has shape (2, 3). Numpy treats scalars as arrays of shape ();
# these can be broadcast together to shape (2, 3), producing the
# following array:
print(x * 2)
```

```
[[ 2  4  6]
 [ 8 10 12]]
```

Broadcasting typically makes your code more concise and faster, so you should strive to use it where possible.

This brief overview has touched on many of the important things that you need to know about numpy, but is far from complete. Check out the numpy reference to find out much more about numpy.

A.1.3 Matplotlib

Matplotlib is a plotting library. In this section give a brief introduction to the `matplotlib.pyplot` module, which provides a plotting system similar to that of MATLAB.

[97]:
```
import matplotlib.pyplot as plt
```

By running this special iPython command, we will be displaying plots inline:

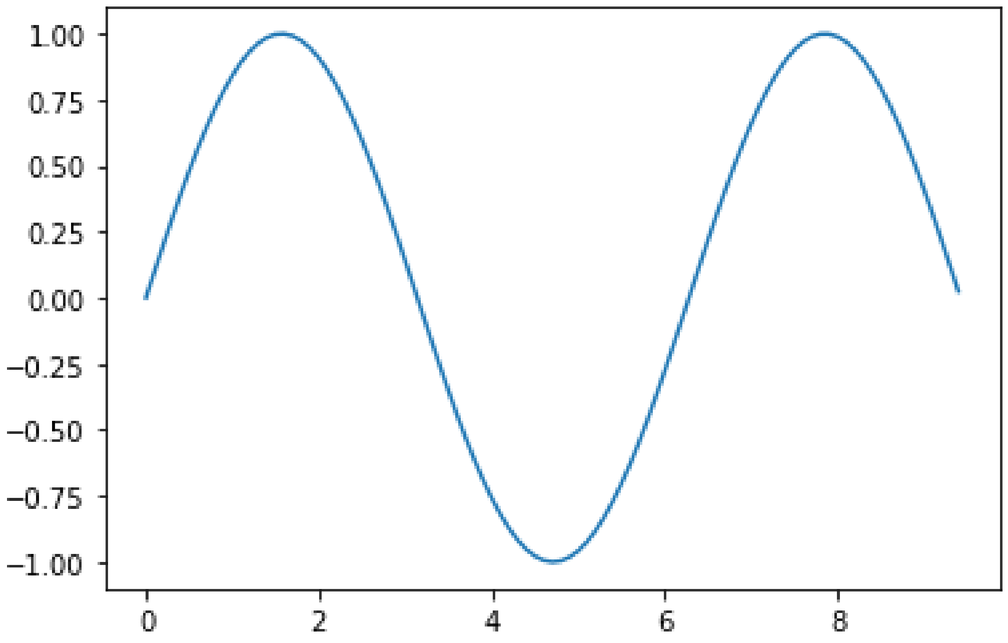
[98]:
```
%matplotlib inline
```

A.1.3.1 Plotting

The most important function in `matplotlib` is plot, which allows you to plot 2D data. Here is a simple example:

[99]:
```
# Compute the x and y coordinates for points on a sine curve
x = np.arange(0, 3 * np.pi, 0.1)
y = np.sin(x)

# Plot the points using matplotlib
plt.plot(x, y)
```

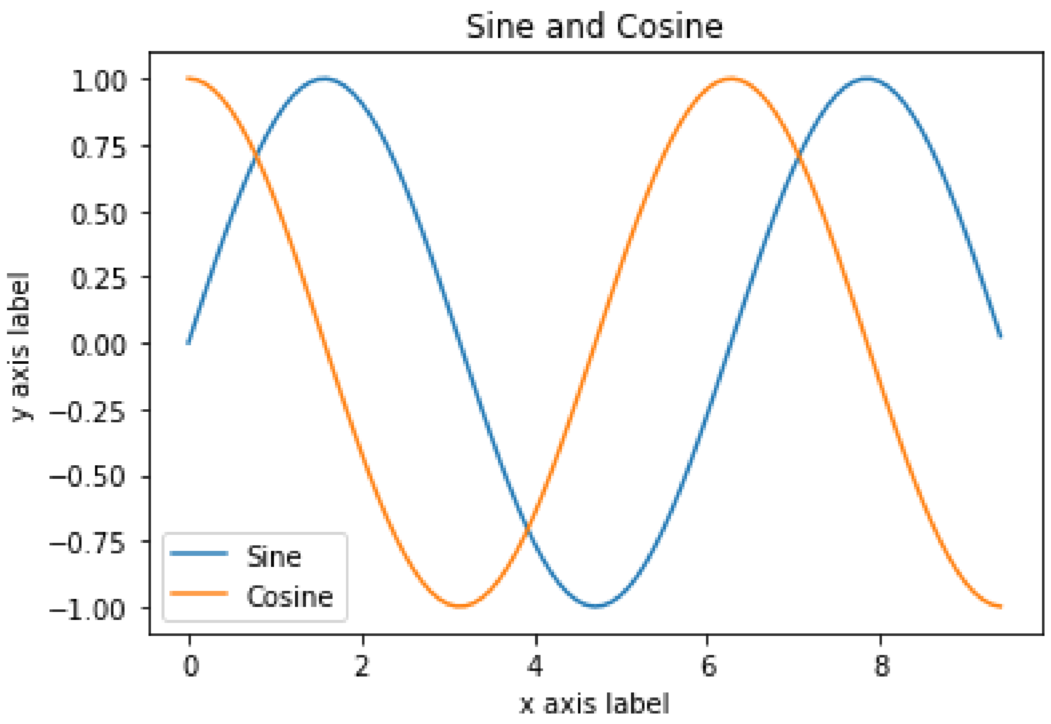
[99]:
```
[<matplotlib.lines.Line2D at 0x7f78639a1748>]
```



With just a little bit of extra work we can easily plot multiple lines at once, and add a title, legend, and axis labels:

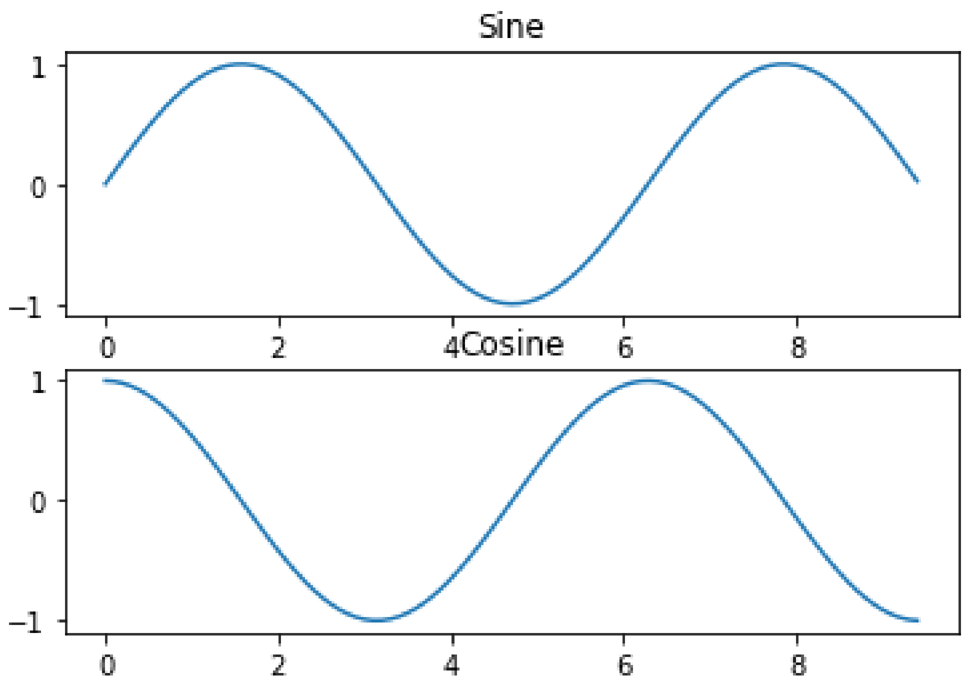
```
[100]: y_sin = np.sin(x)
       y_cos = np.cos(x)

       # Plot the points using matplotlib
       plt.plot(x, y_sin)
       plt.plot(x, y_cos)
       plt.xlabel('x axis label')
       plt.ylabel('y axis label')
       plt.title('Sine and Cosine')
       plt.legend(['Sine', 'Cosine'])
```

```
[100]: <matplotlib.legend.Legend at 0x7f78634f2860>
```



A.1.3.2 Subplots You can plot different things in the same figure using the subplot function. Here is an example:

[101]:
```
# Compute the x and y coordinates for points on sine and cosine curves
x = np.arange(0, 3 * np.pi, 0.1)
y_sin = np.sin(x)
y_cos = np.cos(x)

# Set up a subplot grid that has height 2 and width 1,
# and set the first such subplot as active.
plt.subplot(2, 1, 1)

# Make the first plot
plt.plot(x, y_sin)
plt.title('Sine')

# Set the second subplot as active, and make the second plot.
plt.subplot(2, 1, 2)
plt.plot(x, y_cos)
plt.title('Cosine')

# Show the figure.
plt.show()
```



You can read much more about the `subplot` function in the documentation.

B Numerical Dynamics and Phase–Space Statistics: Harmonic Oscillator

This section describes the implementation of the Velocity-Verlet algorithm for simulating the dynamics of a harmonic oscillator.

B.1 Model System

We consider a one–dimensional classical harmonic oscillator with Hamiltonian

$$H(q,p) = \frac{p^2}{2m} + \frac{1}{2}kq^2, \tag{685}$$

where q is the position, p the momentum, m the mass, and k the spring constant. The equations of motion follow from Hamilton's equations,

$$\dot{q} = \frac{p}{m}, \qquad \dot{p} = -kq. \tag{686}$$

This system is integrable and conserves the total energy exactly in continuous time. Numerically, however, the choice of integrator is critical for long–time stability.

B.2 Velocity Verlet Integrator

To evolve the dynamics, the following code implements the velocity Verlet scheme, a second–order, symplectic integrator widely used in molecular dynamics. Given (q_n, p_n) at time t_n, the updates are

$$q_{n+1} = q_n + \frac{p_n}{m}\Delta t + \frac{1}{2}a_n\Delta t^2, \tag{687}$$

$$p_{n+1} = p_n + \frac{1}{2}(a_n + a_{n+1})m\Delta t, \tag{688}$$

where the acceleration is

$$a_n = -\frac{k}{m}q_n. \tag{689}$$

Because the method is symplectic, it does not introduce secular energy drift. Instead, the energy oscillates around the exact value with bounded error, even over very long trajectories. The code explicitly monitors this by computing

$$\frac{\Delta E}{E_0} = \frac{E(t) - E(0)}{E(0)}. \tag{690}$$

B.3 Time Series and Phase–Space Trajectory

The arrays $q(t)$ and $p(t)$ produced by the integrator are used to generate:

- Time series plots of $q(t)$ and $p(t)$, showing sinusoidal motion with frequency $\omega = \sqrt{k/m}$.
- A phase–space trajectory $(q(t), p(t))$, which traces an ellipse corresponding to constant energy,

$$\frac{p^2}{2m} + \frac{1}{2}kq^2 = E. \tag{691}$$

These plots provide a direct visual diagnostic of both the physical dynamics and the numerical stability of the integrator.

B.4 Empirical Phase–Space Probability Density

Although the trajectory is deterministic, long–time sampling along the orbit can be used to construct an empirical phase–space probability density. After discarding the initial transient, the remaining points $\{(q_i, p_i)\}$ are binned into a two–dimensional histogram,

$$\rho(q, p) \approx \frac{1}{N}\sum_{i=1}^{N} \delta_\Delta(q - q_i)\,\delta_\Delta(p - p_i), \tag{692}$$

where δ_Δ represents binning over finite widths in q and p.

In the code, this is implemented using a normalized 2D histogram (`density=True`), ensuring that

$$\int \rho(q, p)\,dq\,dp \approx 1. \tag{693}$$

B.5 Three–Dimensional Visualization

The resulting discrete density $\rho(q, p)$ is plotted as a three–dimensional surface, with axes (q, p, ρ). For a single harmonic oscillator trajectory at fixed energy, the density forms a ridge concentrated along the constant–energy ellipse. This reflects Liouville's theorem: phase–space volume is preserved, and the dynamics remain confined to the energy shell.

B.6 Physical Interpretation

This construction illustrates the connection between deterministic Hamiltonian dynamics and statistical descriptions. While the instantaneous state occupies a single point in phase space, long–time sampling generates a stationary distribution supported on the energy manifold. Extensions to ensembles of initial conditions or weakly perturbed systems lead naturally to smooth phase–space densities and equilibrium statistical mechanics.

B.7 Running the Code in Google Colab.

The following Python script can be executed in Google Colab without local installation. First, navigate to `https://colab.research.google.com` and create a new notebook. Upload the script using `File → Upload notebook` or by dragging the `.py` file into the Colab workspace. Alternatively, the code may be pasted directly into a notebook cell. Required dependencies (`numpy` and `matplotlib`) are preinstalled in Colab. Once uploaded, run the notebook cells sequentially using `Runtime → Run all`. The figures are rendered inline, and printed diagnostics (energy drift and time-averaged energies) appear in the notebook output shown in Fig. 6.

```
import numpy as np
import matplotlib.pyplot as plt

def velocity_verlet(q0, p0, m, k, dt, nsteps):
    q = np.empty(nsteps + 1)
    p = np.empty(nsteps + 1)
    t = np.linspace(0, nsteps * dt, nsteps + 1)

    q[0] = q0
    p[0] = p0

    def force(q):
        return -k * q

    for i in range(nsteps):
        a = force(q[i]) / m
        q_next = q[i] + (p[i] / m) * dt + 0.5 * a * dt**2
        a_next = force(q_next) / m
        p_next = p[i] + 0.5 * (a + a_next) * m * dt

        q[i + 1] = q_next
        p[i + 1] = p_next

    K = p**2 / (2 * m)
    V = 0.5 * k * q**2
    E = K + V
    return q, p, t, E, K, V


# --- Parameters ---
m = 1.0
k = 1.0
q0 = 1.0
p0 = 0.0
dt = 0.05
```

```
nsteps = 20000

q, p, t, E, K, V = velocity_verlet(q0, p0, m, k, dt, nsteps)

# --- Energy diagnostics ---
E0 = E[0]
print(f"Max |ΔE/E0| ≈ {np.max(np.abs((E - E0) / E0)):.2e}")

# --- Time averages (second half of trajectory) ---
half = len(t) // 2
Tspan = t[-1] - t[half]

K_avg = np.trapezoid(K[half:], t[half:]) / Tspan
V_avg = np.trapezoid(V[half:], t[half:]) / Tspan
E_avg = np.trapezoid(E[half:], t[half:]) / Tspan

print(f"<K>_t ≈ {K_avg:.6f}")
print(f"<V>_t ≈ {V_avg:.6f}")
print(f"<E>_t ≈ {E_avg:.6f}")

# --- Phase-space density ---
q_s = q[half:]
p_s = p[half:]

bins = 200
H, q_edges, p_edges = np.histogram2d(q_s, p_s, bins=bins, density=True)

# =========================
# 2×2 plot layout
# =========================
fig, axs = plt.subplots(2, 2, figsize=(12, 10))

# (1) q(t) and p(t)
axs[0, 0].plot(t, q, label="q(t)")
axs[0, 0].plot(t, p, label="p(t)")
axs[0, 0].set_xlim(0, 100)
axs[0, 0].set_xlabel("t")
axs[0, 0].set_ylabel("value")
axs[0, 0].set_title("q and p vs time")
axs[0, 0].legend()

# (2) Phase-space trajectory
axs[0, 1].plot(q, p, lw=1)
axs[0, 1].set_xlabel("q")
axs[0, 1].set_ylabel("p")
axs[0, 1].set_title("Phase-space trajectory")

# (3) Energy vs time
axs[1, 0].plot(t, E)
axs[1, 0].set_xlabel("t")
axs[1, 0].set_ylabel("E(t)")
axs[1, 0].set_xlim(0, 100)
axs[1, 0].set_title("Total energy vs time")
```

```
# (4) Phase-space probability density
im = axs[1, 1].imshow(
    H.T,
    origin="lower",
    aspect="auto",
    extent=[q_edges[0], q_edges[-1], p_edges[0], p_edges[-1]]
)
axs[1, 1].set_xlabel("q")
axs[1, 1].set_ylabel("p")
axs[1, 1].set_title(r"Phase-space density $\rho(q,p)$")
fig.colorbar(im, ax=axs[1, 1])

plt.tight_layout()
plt.show()
```

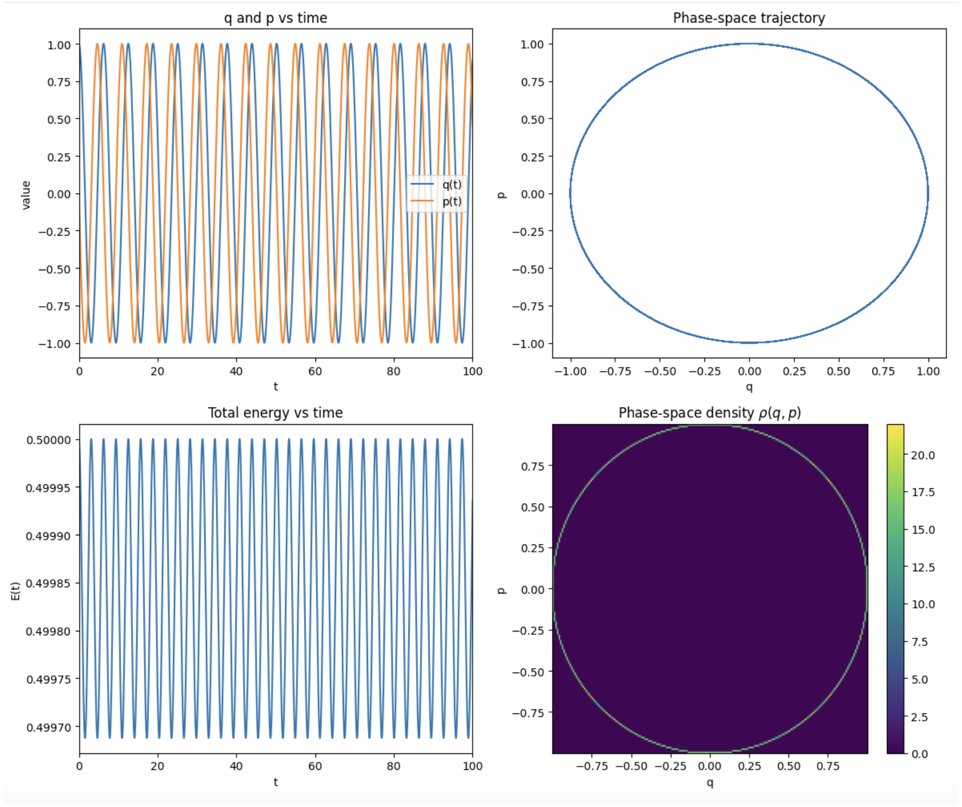


Figure 6: Classical dynamics of a one-dimensional harmonic oscillator. **(Top left)** Time evolution of the position $q(t)$ and momentum $p(t)$, showing periodic motion with a phase shift. **(Top right)** Phase-space trajectory (q, p), forming a closed ellipse corresponding to constant total energy. **(Bottom left)** Total energy $E(t) = K(t) + V(t)$ as a function of time, demonstrating excellent energy conservation of the velocity Verlet integrator. **(Bottom right)** Empirical phase-space probability density $\rho(q, p)$ obtained from long-time sampling of the trajectory, concentrated along the constant-energy ellipse in accordance with Liouville's theorem.

C Tutorial: Molecular Dynamics of Liquid Argon (Lennard–Jones) with RDF and Diffusion

This section walks through a compact molecular dynamics (MD) code for liquid argon modeled as a Lennard–Jones (LJ) fluid in *reduced LJ units*:

$$\sigma = 1, \quad \epsilon = 1, \quad m = 1, \quad k_B = 1.$$

We simulate an $N = 4n_{\text{cells}}^3$ particle system under periodic boundary conditions (PBC) using velocity-Verlet integration. From the *production* (NVE) trajectory we compute:

- the radial distribution function $g(r)$, and
- the diffusion constant D from the mean-squared displacement (MSD) using the Einstein relation.

C.1 Imports

We only require `numpy` for numerics and `matplotlib` for plotting.

Listing 1: Imports used in the tutorial.

```
import numpy as np
import matplotlib.pyplot as plt
```

C.2 Initializing a dense liquid: FCC lattice

Random placement at liquid density can create particle overlaps (huge LJ forces and numerical instability). A standard workaround is to start from a crystalline lattice (FCC here), then add small random displacements. The function below constructs an FCC lattice by tiling an FCC basis over a cubic box.

Listing 2: FCC lattice initialization in a cubic periodic box.

```
def fcc_lattice(n_cells, box):
    a = box / n_cells
    basis = np.array([
        [0,   0,   0],
        [0, 0.5, 0.5],
        [0.5, 0, 0.5],
        [0.5, 0.5, 0]
    ])
    pts = []
    for i in range(n_cells):
        for j in range(n_cells):
            for k in range(n_cells):
                cell = np.array([i, j, k], float)
                pts.append((cell + basis) * a)
    return np.vstack(pts) % box
```

C.2.0.1 What it does.

- Builds $4n_{\text{cells}}^3$ positions using an FCC basis scaled by lattice constant $a = \texttt{box}/n_{\text{cells}}$.
- Applies `% box` to ensure all coordinates lie in $[0, \texttt{box})$ for PBC.

C.3 Core MD engine: forces, temperature, and velocity-Verlet

The LJ pair potential is

$$U(r) = 4\epsilon \left[\left(\frac{\sigma}{r}\right)^{12} - \left(\frac{\sigma}{r}\right)^{6} \right],$$

with a cutoff $r_c = 2.5\sigma$. We use minimum-image PBC and a *shifted* potential $U(r) - U(r_c)$ so that $U(r_c) = 0$ (reduces discontinuity in energy at the cutoff).

C.3.1 LJ forces (minimum image + cutoff + potential shift)

Listing 3: Lennard–Jones forces with minimum-image PBC and potential shift.

```
def lj_forces(positions, box, epsilon=1.0, sigma=1.0, rcut=2.5):
    N = positions.shape[0]
    forces = np.zeros_like(positions)
    pe = 0.0
    rcut2 = (rcut*sigma)**2

    # potential shift U(rcut)=0
    inv_rc2 = (sigma**2)/rcut2
    inv_rc6 = inv_rc2**3
    inv_rc12 = inv_rc6**2
    U_shift = 4*epsilon*(inv_rc12 - inv_rc6)

    for i in range(N-1):
        rij = positions[i+1:] - positions[i]
        rij -= box * np.round(rij/box)            # minimum image
        r2 = np.sum(rij*rij, axis=1)
        mask = r2 < rcut2
        if not np.any(mask):
            continue

        r2m = r2[mask]
        rijm = rij[mask]

        inv_r2 = (sigma**2)/r2m
        inv_r6 = inv_r2**3
        inv_r12 = inv_r6**2

        pe += np.sum(4*epsilon*(inv_r12 - inv_r6) - U_shift)

        fac = 24*epsilon*(2*inv_r12 - inv_r6) / r2m
        fij = fac[:, None] * rijm

        forces[i] -= np.sum(fij, axis=0)
        forces[i+1:][mask] += fij

    return forces, pe
```

C.3.1.1 What it does.

- Computes all pair interactions in an $O(N^2)$ loop (fine for $N \approx 256$).

- Enforces periodicity with the *minimum-image convention*:

$$\mathbf{r}_{ij} \leftarrow \mathbf{r}_{ij} - L \cdot \mathrm{round}(\mathbf{r}_{ij}/L),$$

  where $L =$ `box`.

- Applies a cutoff $r < r_c$ and shifts the potential so $U(r_c) = 0$.
- Returns the total force on each particle and the potential energy.

C.3.2 Instantaneous temperature from velocities

We estimate temperature from kinetic energy (with $k_B = 1$ in reduced units). To avoid artificial heating from center-of-mass drift, we remove the mean velocity before computing the kinetic energy.

Listing 4: Temperature and kinetic energy from velocities with COM drift removal.

```
def temperature_from_vel(vel, mass=1.0):
    vel = vel - vel.mean(axis=0, keepdims=True)  # remove COM drift
    ke = 0.5 * mass * np.sum(vel**2)
    dof = 3*vel.shape[0] - 3
    T = (2*ke) / dof  # kB=1
    return T, ke, vel
```

C.3.2.1 What it does.

- Computes $K = \frac{1}{2} m \sum_i |\mathbf{v}_i|^2$ after removing COM motion.
- Uses degrees of freedom $\mathrm{dof} = 3N - 3$ (subtract 3 for removed COM velocity).
- Returns $(T, K, \mathbf{v})$ where `vel` is the drift-corrected velocity array.

C.3.3 Velocity-Verlet integrator

Velocity-Verlet is symplectic and widely used in MD because it provides good energy conservation in NVE when Δt is chosen appropriately.

Listing 5: Velocity-Verlet integration step.

```
def velocity_verlet(pos, vel, box, dt, mass=1.0, epsilon=1.0, sigma=1.0, rcut
    =2.5):
    F, pe = lj_forces(pos, box, epsilon, sigma, rcut)
    vel_half = vel + 0.5*dt*F/mass
    pos_new = (pos + dt*vel_half) % box
    F_new, pe_new = lj_forces(pos_new, box, epsilon, sigma, rcut)
    vel_new = vel_half + 0.5*dt*F_new/mass
    return pos_new, vel_new, pe_new
```

C.3.3.1 What it does.

- Updates velocities by a half step using current forces.
- Updates positions by a full step and wraps them into the periodic box.
- Recomputes forces at new positions and completes the velocity update.
- Returns updated $(\mathbf{r}, \mathbf{v})$ and the new potential energy.

C.4 Structure and dynamics analysis helpers

We now add analysis routines for $g(r)$ and for MSD/D.

C.4.1 Pair distances and RDF $g(r)$

The RDF measures local structure:

$$g(r) = \frac{1}{4\pi r^2 \rho} \left\langle \frac{dN(r)}{dr} \right\rangle,$$

where $\rho = N/V$ and $dN(r)$ counts neighbors in a spherical shell. In practice we histogram pair distances from snapshots and normalize by the ideal-gas expectation.

Listing 6: All unique pair distances under minimum-image PBC (used for RDF).

```
def pair_distances(positions, box):
    """All unique pair distances with minimum image. O(N^2) -- fine for N~256.
        """
    N = positions.shape[0]
    dists = []
    for i in range(N-1):
        rij = positions[i+1:] - positions[i]
        rij -= box * np.round(rij/box)
        d2 = np.sum(rij*rij, axis=1)
        dists.append(np.sqrt(d2))
    return np.concatenate(dists)
```

C.4.1.1 What it does.

- Computes distances for all unique pairs $(i < j)$ under PBC via minimum-image.
- Returns a 1D array of distances, suitable for histogramming.

Listing 7: Compute RDF from a list of sampled configurations.

```
def rdf_from_trajectory(pos_samples, box, nbins=100, rmax=None):
    """
    Compute radial distribution function g(r) from sampled configurations.
    pos_samples: list/array of (N,3) snapshots
    """
    pos_samples = list(pos_samples)
    N = pos_samples[0].shape[0]
    V = box**3
    rho = N / V
    if rmax is None:
        rmax = box/2  # max meaningful under PBC

    edges = np.linspace(0.0, rmax, nbins+1)
    hist = np.zeros(nbins, dtype=float)

    for pos in pos_samples:
        d = pair_distances(pos, box)
        hist += np.histogram(d, bins=edges)[0]
    # Normalize
    dr = edges[1] - edges[0]
    r = 0.5*(edges[:-1] + edges[1:])
    shell_vol = 4*np.pi*r**2 * dr

    # Expected counts per shell per snapshot for ideal gas:
    # N * rho * shell_vol / 2  (divide by 2 because we counted unique pairs)
    norm = len(pos_samples) * (N * rho * shell_vol / 2.0)
    g = hist / norm
    return r, g
```

C.4.1.2 What it does.

- Accumulates a histogram of pair distances over many snapshots.
- Uses $r_{\text{max}} = L/2$ by default (largest meaningful distance under minimum-image PBC).
- Normalizes by the ideal-gas expected counts per shell:

$$\text{norm} = N\rho\,4\pi r^2\,dr/2$$

  (the factor $1/2$ corrects for using unique pairs).
- Returns bin centers r and $g(r)$.

C.4.2 Unwrapping, MSD, and diffusion constant

Under PBC, wrapped coordinates jump discontinuously at the boundaries, which breaks long-time displacement. We therefore construct *unwrapped* trajectories by accumulating minimum-image displacements between frames.

Listing 8: Unwrap positions step-to-step to reconstruct continuous trajectories.

```
def unwrap_positions(pos, pos_prev, unwrapped_prev, box):
    """
    Unwrap positions given current wrapped pos and previous wrapped pos.
    """
    delta = pos - pos_prev
    delta -= box * np.round(delta/box)      # minimum-image displacement
    return unwrapped_prev + delta
```

C.4.2.1 What it does.

- Computes the minimum-image displacement between successive wrapped frames.
- Accumulates this displacement into an unwrapped coordinate array so particles move continuously across periodic boundaries.

Listing 9: Compute MSD relative to a reference time index from unwrapped snapshots.

```
def msd_from_unwrapped(unwrapped_samples, ref_index=0):
    """
    MSD(t) averaged over particles relative to reference time index.
    """
    r0 = unwrapped_samples[ref_index]
    msd = []
    for ut in unwrapped_samples:
        dr = ut - r0
        msd.append(np.mean(np.sum(dr*dr, axis=1)))
    return np.array(msd)
```

C.4.2.2 What it does.

- Computes
$$\mathrm{MSD}(t) = \left\langle |\mathbf{r}_i(t) - \mathbf{r}_i(0)|^2 \right\rangle_i$$
averaged over particles, using unwrapped positions.

- Uses a single reference time (index `ref_index`); for higher accuracy you can time-average over multiple reference origins, but this is often sufficient for a tutorial-scale run.

Listing 10: Estimate diffusion constant by linear fit of MSD in a late-time window.

```
def diffusion_from_msd(times, msd, fit_start_frac=0.4, fit_end_frac=0.9):
    """
    Fit MSD ~ 6 D t over a late-time window.
    """
    n = len(times)
    i0 = int(fit_start_frac*n)
    i1 = int(fit_end_frac*n)
    tfit = times[i0:i1]
    mfit = msd[i0:i1]

    # linear fit
    slope, intercept = np.polyfit(tfit, mfit, 1)
    D = slope / 6.0
    return D, slope, intercept, (i0, i1)
```

C.4.2.3 What it does.

- Fits a straight line to MSD in a late-time region (default 40%–90% of the sampled times).

- Uses the 3D Einstein relation at long times:
$$\mathrm{MSD}(t) \approx 6Dt \quad \Rightarrow \quad D = \frac{1}{6}\frac{d\,\mathrm{MSD}}{dt}.$$

- Returns D, the fitted slope/intercept, and the fit index window for plotting.

C.5 Running the simulation: equilibration (NVT-like) then production (NVE)

A common workflow is:

1. **Equilibration**: drive the system to the target temperature using simple velocity rescaling (a lightweight thermostat).
2. **Production**: switch off the thermostat and run NVE for collecting physically meaningful dynamics (MSD, diffusion).

In the provided driver code, snapshots are stored every `sample_every` steps for RDF and MSD analysis, while unwrapping is updated every step.

C.6 Computing and plotting $g(r)$ and MSD/D

Once `pos_samples` (wrapped snapshots) and `unwrapped_samples` are collected:

- Call `rdf_from_trajectory(pos_samples, box, ...)` to obtain $(r, g(r))$.
- Call `msd_from_unwrapped(unwrapped_samples)` to get MSD(t).
- Fit a late-time region via `diffusion_from_msd(times, msd)` and report D.

C.7 Optional: mapping reduced units to real argon units

If you choose LJ parameters for argon (one common set is $\sigma = 3.405\,\text{Å}$, $\epsilon/k_B = 119.8\,\text{K}$, $m = 39.948\,\text{amu}$), the LJ time unit is

$$\tau = \sigma\sqrt{\frac{m}{\epsilon}},$$

and diffusion converts as

$$D_{\text{SI}} = D_{\text{LJ}}\frac{\sigma^2}{\tau}.$$

This lets you translate the dimensionless D from the simulation into m^2/s (or cm^2/s).

C.8 Run MD: Equilibrate with velocity rescaling, then NVE

```
# --- Simulation parameters (reduced LJ units) ---
np.random.seed(0)

n_cells = 4                 # 4x4x4 FCC cells => N = 4*n_cells^3 = 256 atoms
N = 4 * n_cells**3

rho = 0.8                   # number density (typical LJ liquid-ish)
box = (N / rho)**(1/3)

dt = 0.004                  # time step (LJ units)
steps_eq = 2000             # equilibration steps
steps_prod = 3000           # production steps
rcut = 2.5

T_target = 1.0              # target temperature (LJ units)
mass = 1.0                  # LJ reduced mass
```

```
epsilon = 1.0
sigma = 1.0

# --- Initialize positions on FCC lattice ---
pos = fcc_lattice(n_cells, box)

# Small random jitter to break perfect symmetry (optional)
pos = (pos + 0.02*np.random.randn(N,3)) % box

# --- Initialize velocities (Maxwell-ish) ---
vel = np.random.randn(N,3)
T0, ke0, vel = temperature_from_vel(vel, mass=mass)
vel *= np.sqrt(T_target / T0)  # scale to target T

# --- Run ---
pes, kes, temps = [], [], []

# Equilibration: simple velocity rescaling thermostat
for step in range(steps_eq):
    pos, vel, pe = velocity_verlet(pos, vel, box, dt, mass, epsilon, sigma,
        rcut)
    T, ke, vel = temperature_from_vel(vel, mass)
    vel *= np.sqrt(T_target / T)  # rescale (thermostat)
    pes.append(pe); kes.append(ke); temps.append(T)

# Production: NVE (no thermostat)
for step in range(steps_prod):
    pos, vel, pe = velocity_verlet(pos, vel, box, dt, mass, epsilon, sigma,
        rcut)
    T, ke, vel = temperature_from_vel(vel, mass)
    pes.append(pe); kes.append(ke); temps.append(T)

E = np.array(pes) + np.array(kes)
print("N:", N, "box:", box)
print("T (last 1000 avg):", np.mean(temps[-1000:]))
print("Energy drift (prod):", E[-1] - E[steps_eq])
```

The output should read:

- N: 256 box: 6.839903786706787
- T (last 1000 avg): 0.9865451989895921
- Energy drift (prod): 0.03515588407606174

C.9 Plot temperature and energy traces

```
plt.figure(figsize=(10, 4))

plt.subplot(1, 2, 1)
plt.plot(temps)
plt.xlabel("Step")
plt.ylabel("Temperature (LJ units)")
plt.title("Temperature")
plt.subplot(1, 2, 2)
plt.plot(E)
plt.xlabel("Step")
plt.ylabel("Total Energy (LJ units)")
plt.title("Total Energy")

plt.tight_layout()
plt.show()
```

The plots should show equilibration and energy conservation, as show below in Fig. 7. You should see:

- Temperature stabilizes near T_{target} during equilibration
- During production (NVE), total energy is roughly conserved (small drift is normal in a simple code)

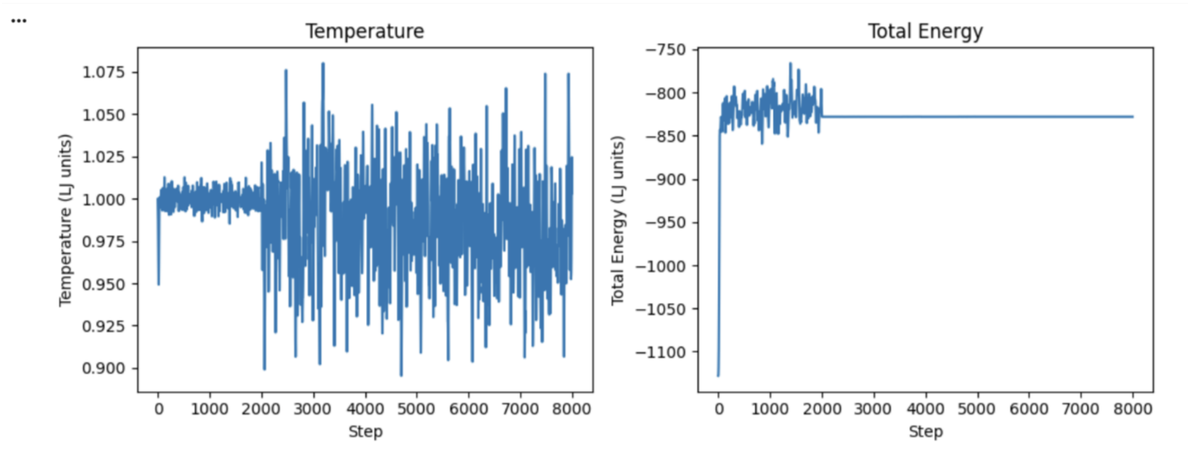


Figure 7: Instantaneous temperature (left) and total energy (right) as a function of MD step for the Lennard–Jones liquid argon simulation. During the initial equilibration stage, velocity rescaling maintains the temperature near the target value, while the total energy adjusts accordingly. After the switch to the NVE production run, the temperature exhibits natural fluctuations about equilibrium and the total energy remains constant, demonstrating good energy conservation and stable time integration.

C.10 Quick 2D snapshot

A scatter plot for a quick 2D visualization of the system can be generated, as follows:

```
plt.figure()
plt.scatter(pos[:,0], pos[:,1], s=10)
plt.xlim(0, box); plt.ylim(0, box)
plt.gca().set_aspect('equal', adjustable='box')
plt.xlabel("x"); plt.ylabel("y")
plt.title("Argon LJ MD snapshot (projection)")
plt.show()
```

producing the following picture:

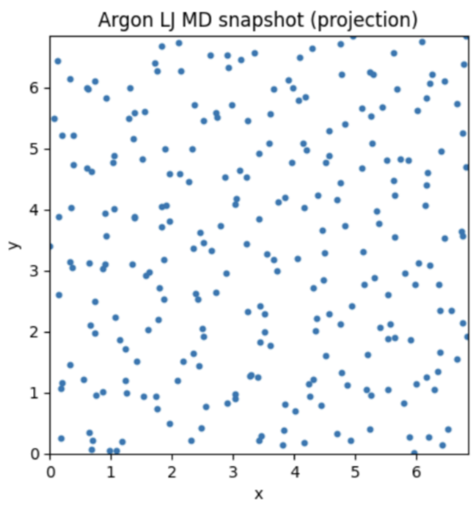


Figure 8: Two-dimensional projection of a representative snapshot from the Lennard–Jones molecular dynamics simulation of liquid argon. Each point corresponds to a particle position projected onto the x–y plane under periodic boundary conditions. The disordered spatial distribution is characteristic of a dense liquid state.

C.11 Run MD and collect samples

```
# --- Parameters (reduced LJ units) ---
np.random.seed(0)
n_cells = 4
N = 4 * n_cells**3

rho = 0.8
box = (N/rho)**(1/3)

dt = 0.004
steps_eq = 2000
steps_prod = 6000
rcut = 2.5

T_target = 1.0
mass = 1.0
epsilon = 1.0
sigma = 1.0

# sampling controls
sample_every = 10      # take a snapshot every N steps
nbins_rdf = 120

# --- Initialize ---
pos = fcc_lattice(n_cells, box)
pos = (pos + 0.02*np.random.randn(N,3)) % box

vel = np.random.randn(N,3)
T0, ke0, vel = temperature_from_vel(vel, mass)
```

```
vel *= np.sqrt(T_target / T0)

pes, kes, temps = [], [], []

# --- Equilibration (velocity rescaling thermostat) ---
for step in range(steps_eq):
    pos, vel, pe = velocity_verlet(pos, vel, box, dt, mass, epsilon, sigma,
        rcut)
    T, ke, vel = temperature_from_vel(vel, mass)
    vel *= np.sqrt(T_target / T)
    pes.append(pe); kes.append(ke); temps.append(T)

# --- Production (NVE), collect snapshots for RDF and unwrapped positions for
   MSD ---
pos_samples = []
unwrapped_samples = []

pos_prev = pos.copy()
unwrapped = pos.copy()   # start unwrapped == wrapped at t=0 of production
t_samples = []

for step in range(steps_prod):
    pos, vel, pe = velocity_verlet(pos, vel, box, dt, mass, epsilon, sigma,
        rcut)
    T, ke, vel = temperature_from_vel(vel, mass)
    pes.append(pe); kes.append(ke); temps.append(T)

    # unwrap every step (cheap)
    unwrapped = unwrap_positions(pos, pos_prev, unwrapped, box)
    pos_prev = pos.copy()

    if step % sample_every == 0:
        pos_samples.append(pos.copy())
        unwrapped_samples.append(unwrapped.copy())
        t_samples.append(step*dt)

E = np.array(pes) + np.array(kes)
print("N:", N, "box:", box, "rho:", rho)
print("T (prod avg):", np.mean(temps[-steps_prod:]))
print("Energy drift (prod):", E[-1] - E[-steps_prod])
print("Snapshots:", len(pos_samples))
```

The output should look, as follows:

- N: 256 box: 6.839903786706787 rho: 0.8
- T (prod avg): 0.985597823433789
- Energy drift (prod): 0.0011463063798373696
- Snapshots: 600

C.12 Compute and plot g(r)

```
r, g = rdf_from_trajectory(pos_samples, box, nbins=nbins_rdf, rmax=box/2)

plt.figure()
plt.plot(r, g)
plt.xlabel("r (LJ units)")
plt.ylabel("g(r)")
plt.title("Radial Distribution Function")
plt.show()
```

The figure should look, as follows:

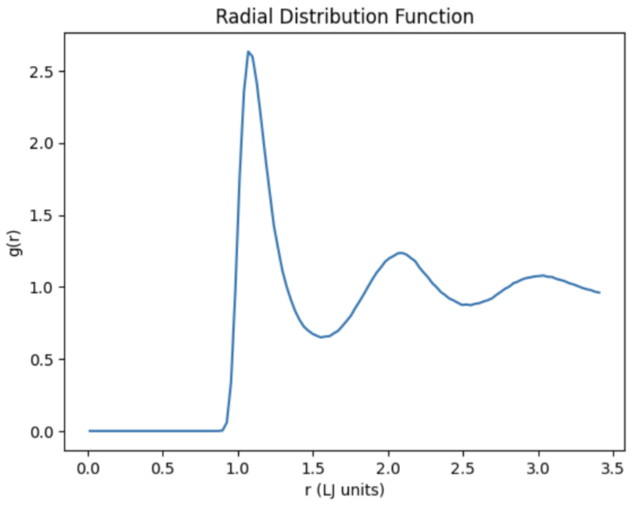
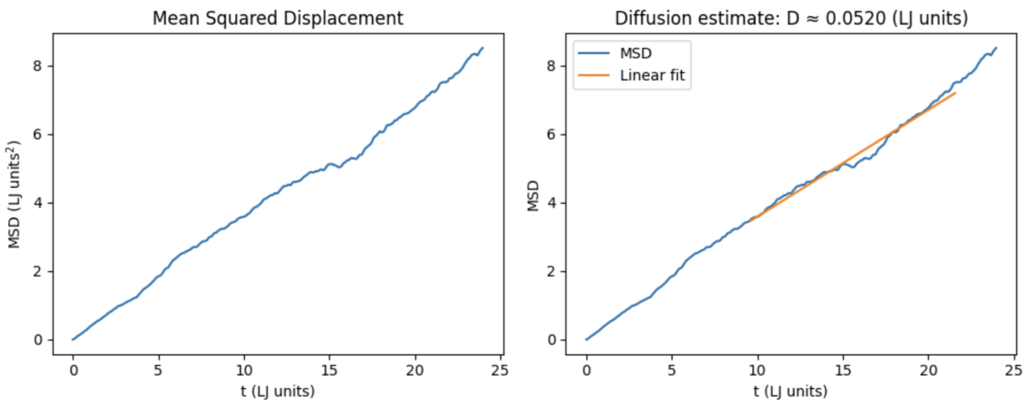


Figure 9: Radial distribution function $g(r)$ of the Lennard–Jones liquid argon system computed from the NVE production trajectory. The pronounced first peak near $r \approx \sigma$ indicates strong short-range ordering due to repulsive cores and nearest-neighbor coordination, while the damped oscillations at larger r reflect medium-range liquid structure and convergence toward the ideal-gas limit $g(r) \to 1$.

C.13 Compute MSD and diffusion constant D

```
times = np.array(t_samples)
msd = msd_from_unwrapped(unwrapped_samples)

D, slope, intercept, (i0, i1) = diffusion_from_msd(times, msd)

plt.figure(figsize=(10, 4))

# Left: MSD vs time
plt.subplot(1, 2, 1)
plt.plot(times, msd)
plt.xlabel("t (LJ units)")
plt.ylabel("MSD (LJ units$^2$)")
plt.title("Mean Squared Displacement")

# Right: MSD with linear fit
plt.subplot(1, 2, 2)
plt.plot(times, msd, label="MSD")
plt.plot(times[i0:i1], slope*times[i0:i1] + intercept, label="Linear fit")
plt.xlabel("t (LJ units)")
plt.ylabel("MSD")
plt.title(f"Diffusion estimate: D =  {D:.4f} (LJ units)")
plt.legend()

plt.tight_layout()
plt.show()

print(f"Estimated diffusion constant D = {D:.6f} (LJ units)")
```

The figures should look, as follows:



```
Estimated diffusion constant D ≈ 0.051979 (LJ units)
```

Figure 10: Mean-squared displacement (MSD) of Lennard–Jones liquid argon as a function of time (left) and MSD with a linear fit to the long-time regime (right). At late times the MSD grows linearly with time, consistent with diffusive behavior. The diffusion constant D is obtained from the slope of the fit using the Einstein relation $\mathrm{MSD}(t) \approx 6Dt$, yielding $D \approx 5.2 \times 10^{-2}$ in reduced LJ units.

C.14 Optional: Conversion of the Diffusion Constant to Real Argon Units

The diffusion constant obtained from the simulation is expressed in reduced Lennard–Jones (LJ) units. To convert this value to physical units (m^2/s) for argon, one may use a commonly adopted set of LJ parameters:

- Lennard–Jones length parameter:

$$\sigma = 3.405\ \text{Å} = 3.405 \times 10^{-10}\ \text{m},$$

- Lennard–Jones energy parameter:

$$\epsilon / k_B = 119.8\ \text{K} \quad \Rightarrow \quad \epsilon = 119.8\, k_B,$$

- Argon atomic mass:

$$m = 39.948\ \text{amu} = 39.948 \times 1.66054 \times 10^{-27}\ \text{kg}.$$

In LJ reduced units, the natural time scale is

$$\tau = \sigma\sqrt{\frac{m}{\epsilon}}. \tag{694}$$

Using this time unit, the diffusion constant in SI units is obtained from the reduced diffusion constant D_{LJ} via

$$D_{\mathrm{SI}} = D_{\mathrm{LJ}}\,\frac{\sigma^2}{\tau}. \tag{695}$$

This conversion yields D_{SI} in m^2/s; conversion to cm^2/s follows by multiplying by 10^4.

C.15 Computational Notes

The evaluation of the radial distribution function $g(r)$ and the mean-squared displacement (MSD) scales as $\mathcal{O}(N^2)$ per sampled configuration due to explicit pair-distance loops. For the system size used here ($N = 256$) and snapshot sampling every ten MD steps, this cost is negligible in typical Colab or laptop environments. For significantly larger systems (e.g., $N \gtrsim 800$), the use of cell lists or neighbor lists is recommended to reduce the computational complexity to approximately $\mathcal{O}(N)$ per step.

D Liouville's Theorem

Liouville's theorem is a fundamental result of classical statistical mechanics. It states that Hamiltonian dynamics preserves phase-space density along the motion.

Consider a classical system with phase-space coordinates (x, p) and Hamiltonian $H(x, p)$. The equations of motion are given by Hamilton's equations,

$$\dot{x} = \frac{\partial H}{\partial p}, \qquad \dot{p} = -\frac{\partial H}{\partial x}. \tag{696}$$

Let $\rho(x, p, t)$ be the phase-space density of an ensemble of systems. Conservation of probability implies a continuity equation in phase space (explained in Sec. D.1),

$$\frac{\partial \rho}{\partial t} + \nabla_\Gamma \cdot (\rho\,\dot{\Gamma}) = 0, \tag{697}$$

where $\Gamma = (x, p)$ and $\dot{\Gamma} = (\dot{x}, \dot{p})$.

Expanding the divergence term,

$$\frac{\partial \rho}{\partial t} + \dot{x}\frac{\partial \rho}{\partial x} + \dot{p}\frac{\partial \rho}{\partial p} + \rho\left(\frac{\partial \dot{x}}{\partial x} + \frac{\partial \dot{p}}{\partial p}\right) = 0. \tag{698}$$

The first three terms define the total derivative along a phase-space trajectory,

$$\frac{d\rho}{dt} = \frac{\partial \rho}{\partial t} + \dot{x}\frac{\partial \rho}{\partial x} + \dot{p}\frac{\partial \rho}{\partial p}.$$

Using Hamilton's equations,

$$\frac{\partial \dot{x}}{\partial x} = \frac{\partial^2 H}{\partial x\,\partial p}, \qquad \frac{\partial \dot{p}}{\partial p} = -\frac{\partial^2 H}{\partial p\,\partial x}.$$

Since mixed partial derivatives commute, these terms cancel and

$$\frac{\partial \dot{x}}{\partial x} + \frac{\partial \dot{p}}{\partial p} = 0.$$

The continuity equation therefore reduces to

$$\frac{d\rho}{dt} = 0. \tag{699}$$

Liouville's Theorem: The phase-space density is conserved along Hamiltonian trajectories. Equivalently, Hamiltonian flow is incompressible and preserves phase-space volume.

In Poisson-bracket form, Liouville's theorem can be written as

$$\frac{\partial \rho}{\partial t} = -\{\rho, H\}, \qquad \{A, B\} = \frac{\partial A}{\partial x}\frac{\partial B}{\partial p} - \frac{\partial A}{\partial p}\frac{\partial B}{\partial x}. \tag{700}$$

D.1 Derivation of the Continuity Equation in Phase Space

Let $\rho(\Gamma, t)$ be the phase-space probability density, with $\Gamma = (x, p)$. Consider an arbitrary region R of phase space. The probability that the system lies inside R at time t is

$$\mathcal{P}_R(t) = \int_R d\Gamma\, \rho(\Gamma, t), \qquad d\Gamma \equiv dx\, dp. \tag{701}$$

Step 1: Change in probability comes only from flux across the boundary. Probability is neither created nor destroyed, so $\mathcal{P}_R(t)$ can change only because probability flows through the boundary ∂R. Define the probability current in phase space as

$$\mathbf{J}(\Gamma, t) \equiv \rho(\Gamma, t)\, \dot{\Gamma}(\Gamma, t), \tag{702}$$

where $\dot{\Gamma} = (\dot{x}, \dot{p})$ is the phase-space velocity field induced by the dynamics.

The net probability leaving R per unit time is the outward flux of $\mathbf{J}$ through ∂R:

$$\frac{d\mathcal{P}_R}{dt} = -\oint_{\partial R} \mathbf{J} \cdot d\mathbf{S}. \tag{703}$$

(The minus sign ensures that outward flow decreases the probability inside R.)

Step 2: Express the same change using the time derivative of ρ. Differentiating Eq. (701) with respect to time (with R fixed in phase space) gives

$$\frac{d\mathcal{P}_R}{dt} = \int_R d\Gamma\, \frac{\partial \rho}{\partial t}. \tag{704}$$

Step 3: Convert the boundary flux into a volume integral. Using the divergence theorem in phase space,

$$\oint_{\partial R} \mathbf{J} \cdot d\mathbf{S} = \int_R d\Gamma\, \nabla_\Gamma \cdot \mathbf{J}, \qquad \nabla_\Gamma \equiv \left(\frac{\partial}{\partial x}, \frac{\partial}{\partial p}\right), \tag{705}$$

Eq. (703) becomes

$$\frac{d\mathcal{P}_R}{dt} = -\int_R d\Gamma\, \nabla_\Gamma \cdot \mathbf{J}. \tag{706}$$

Step 4: Equate the two expressions and localize. Equating Eqs. (704) and (706) yields

$$\int_R d\Gamma \left[\frac{\partial\rho}{\partial t} + \nabla_\Gamma \cdot \mathbf{J}\right] = 0. \tag{707}$$

Because R is arbitrary, the integrand must vanish pointwise. Therefore,

$$\frac{\partial\rho}{\partial t} + \nabla_\Gamma \cdot \mathbf{J} = 0. \tag{708}$$

Finally, substituting $\mathbf{J} = \rho\,\dot{\Gamma}$ from Eq. (702) gives the continuity equation in phase space:

$$\frac{\partial\rho}{\partial t} + \nabla_\Gamma \cdot (\rho\,\dot{\Gamma}) = 0, \tag{709}$$

which expresses local conservation of probability in phase space.