A COMPUTATIONAL APPROACH TO PHYSICS

M. Ebrahim Foulaadvand

A Computational Approach to Physics

A Computational Approach to Physics

By

M. Ebrahim Foulaadvand

Cambridge
Scholars
Publishing

A Computational Approach to Physics

By M. Ebrahim Foulaadvand

This book first published 2023

Cambridge Scholars Publishing

Lady Stephenson Library, Newcastle upon Tyne, NE6 2PA, UK

British Library Cataloguing in Publication Data
A catalogue record for this book is available from the British Library

ISBN (10): 1-5275-0126-4
ISBN (13): 978-1-5275-0126-3

Dedicated to the memory of *Jamshid Kashani* (Jamshid al-Kashi) (c1380-1429), the Persian (Iranian) mathematician and astronomer who invented the mechanical planetary computer which he called the Plate of Zones, and could graphically solve several planetary problems, including the prediction of the accurate positions in longitude of the Sun and Moon, and the planets in terms of elliptical orbits; the latitudes of the Sun, Moon, and planets; and the ecliptic of the Sun. Kashani computed the Pi number to the 16th decimal place and directed the Samarkand observatory.

Contents

Preface

The subject of *Computational Physics* has attracted much interest among physics and engineering students in recent years. It is a widely growing field thanks to the development and improvement of computational facilities namely, hard and software, clusters of computers, and parallel computation. Outstanding books on computational physics, such as the one written by Gould, Tobochnik, and Christian, from which I have learnt a lot, and the one by Landau, José Páez and Bordeianu have inevitably stimulated the students' interests to the subject. Physics is now profoundly and remarkably connected to numerical techniques and simulation methods. It requires a lot of computational skills for a proper understanding of its concepts and challenges. This has dramatically affected the traditional practice of training physics. The computational approach to physics is becoming an integral part of the physics educational programme. Many departments have included a computational physics course in their bachelor's or master's curricula. Students normally enjoy computational subjects because they realise how easier and more fascinating it will be to employ computational methods and techniques to solve complex problems in physics that are not amenable to exact analytical solutions. They mostly prefer to avoid the complexity of rigorous analytical solutions and instead enjoy solving the physical problems facilitated by packages and softwares numerically. The above opinion and viewpoint are based on the feedback I gained from students over a couple of years of experience in teaching computational physics. In many existing books on computational physics, the lack of a sufficient number of solved problems and examples is obvious. Many of these books heavily emphasise on teaching numerical techniques such as numerical differentiation, error analysis, root finding, etc., instead of applying

them to concrete physical problems. Another major shortcoming of many books with the terms "computation" and "physics" is their concentration on details of writing subroutines and codes without implementing them to sophisticated physical problems to be solved by these codes. Of course, there are exceptions, like the notable books written by Nicholas J. Giordano, and Alejandro Garcia, from which I have significantly learnt, borrowed, and benefited. On the contrary, the main goal of this book is to introduce students how to apply numerical techniques to problems of classical as well as modern physics. The book focuses and emphasizes on solving physics problems. In some sense, it can be considered as a *problem solver*. The distinguishing pedagogical feature of the book is two-fold. First, it provides many numerically solved problems and examples, and second, it briefly reviews analytical results and exact solutions as warming up before numerical involvement to problems that cannot be solved analytically. In this way, the students will achieve a better insight into the necessity of the numerical approach as a supplementary tool for the description of the underlying physics of the problem. The book level is upper intermediate and will be most appropriate for senior undergraduates, and junior postgraduate students. It is assumed that the potential reader has elementary programming knowledge. Therefore, I have avoided including the basics of computer programming and numerical methodologies such as differentiation, integration, etc. All the source codes and programme listings are in *C* programming language. You can access these codes and subroutines in the appendix, which appears online on the book's website: **http://www.znu.ac.ir/members/newpage/998/en**.

I am hopeful that the book will attract the attention of bright and interested students. Needless to say that the book is not free of typos and errors. It is highly appreciated to receive your comments and suggestions that can be sent to any of my email addresses: foolad@znu.ac.ir or ebrahim.foulaadvand@gmail.com.

Tehran, April 2023 M. Ebrahim Foulaadvand

Acknowledgement

I was attracted and inspired to Computational Physics during the "computer application in physics" undergraduate course that I passed with Reza Ejtehadi in 1998 at the former Aria Mehr (current Sharif) University of Technology in Tehran when I was a graduate student. He is gratefully appreciated. I am immensely thankful to the Institute for Research in Fundamental Sciences (IPM) at Tehran for providing me with a working office at which most of the manuscript was written. I am especially indebted to Somayyeh Belbasi, my former Ph.D. student, who contributed to several codes and subroutines. Some colleagues have provided essential advice and encouragement, including Andreas Schadschneider from the University of *Köln*, Gunter *Schütz* from the *Jülish* Forschungszentrum and Philipp Maass from the University of *Osnabrück*. Special thanks are dedicated to Ahmad Shariati and Amir Aghamohammadi from the Alzahra University of Tehran for invaluable help, advice, and technical support. Sincere thanks are given to anonymous referees who proposed valuable and constructive comments to the text. I appreciate the helps and feedback from many enthusiastic students during my lectures on Computational Physics that I gave at the University of Zanjan, Tarbiyat Modares university/Tehran, and the Institute for Advanced Studies at Basic Sciences (IASBS)/Zanjan from which this book has been evolved and emerged. It is the author's pleasure to thank the *Cambridge Scholar Publication* for the very enjoyable cooperation which made this book possible. Finally, the author deeply appreciates his wife, Azadeh, his son Aria Radin, and his daughter Arnika for their understanding, support, and patience during the preparation period of the manuscript.

Chapter 1

1st order differential equations in physics

The laws of physics are frequently formulated within the mathematical framework of differential equations. Therefore, it would be natural to begin our numerical investigations from such equations. For the sake of simplicity, let us start from the simplest case i.e.; first-order differential equations. There are a lot of topics that involve first-order equations. Examples include cooling of hot bodies, the Fourier law of heat transfer, decay of radioactive nuclei, etc. As our first topic let us numerically explore the problem of *coffee cooling* which is discussed in several computational textbooks.

1.1 First order differential equations

In this section, we try to introduce some physical problems which are mathematically formulated in terms of first-order ordinary differential equations. We begin with the old problem of coffee cooling which dates back to the time of Newton.

1.1.1 Coffee Cooling problem

If you put a cup of hot coffee on a table it will cool down until its temperature reaches the room's temperature. Experimental measurement of coffee temperature versus time proves that to a good approximation the temperature-time curve $T(t)$ obeys an exponential fall

(H. Gould and Chriastian, 2006). Energy transfer from the hot water in a cup of coffee to the surrounding air is complicated and in general involves convection, radiation, and conduction mechanisms. However, it can be shown that if the temperature difference between coffee and its surroundings is not too large, the rate of the coffee temperature change can be taken to be proportional to the temperature difference between coffee and its surrounding. Mathematically, one can formulate this statement by the following differential equation:

$$\frac{dT}{dt} = -r(T - T_s). \tag{1.1}$$

where $T(t)$ denotes the instantaneous coffee temperature and T_s denotes the surrounding (room) temperature. The constant r is the cooling constant. The minus sign in (1.1) implies that if $T > T_s$, the coffee temperature will decrease with time. The value of the cooling constant r depends on the heat transfer mechanism, the contact area with the surroundings, and the thermal properties of the water. Equation (1.1) is sometimes known as Newtons law of cooling. Although we all know the analytical solution of the linear first-order differential equation (1.1) let us re-obtain it. We show the initial coffee temperature by T_0 and perform a change of variable $T - T_s = Y$. Equation (1.1) then becomes $\frac{dY}{dt} = -rY$. In terms of the new variable Y, the initial condition becomes $Y(0) = T_0 - T_s$. We therefore, find $Y(t) = Y(0)e^{-rt}$ which in turn implies:

$$T(t) = T_0 e^{-rt} + (1 - e^{-rt})T_s \tag{1.2}$$

To solve (1.1) numerically we should be able to give $T(t)$ at discrete times t_n which are separated from each other by a fixed time interval Δt. In fact, the nth step of time will be $t_n = n\Delta t$. The next step is to write the time derivative in a finite difference form. This is simply achieved by Taylor expansion around t_n as follows:

$$T(t_n + \Delta t) = T(t_n) + \Delta t \frac{dT}{dt}(t_n) + \frac{1}{2}(\Delta t)^2 \frac{d^2T}{dt^2}(t_n) + O(\Delta t)^3 \tag{1.3}$$

Keeping only the term proportional to Δt in (1.3) one finds:

$$\frac{dT}{dt}(t_n) = \frac{T(t_n + \Delta t) - T(t_n)}{\Delta t} + O(\Delta t) \tag{1.4}$$

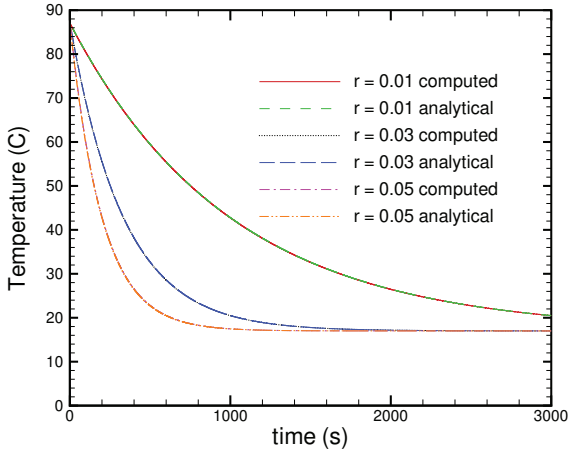

Figure 1.1: *Time evolution of coffee temperature for various values of cooling rate parameter r. Numerical results have been compared to analytical ones. The time step has been set to $\Delta = 0.1$ s. The initial and surrounding temperatures are $T_0 = 87\ ^0C$, $T_s = 17\ ^0C$.*

Putting this on the left-hand side of (1.1) and ignoring the truncation error, which is of order Δt, we simply find:

$$T_{n+1} = T_n - r\Delta t(T_n - T_s) \tag{1.5}$$

Note we have used T_n as a shorthand notation for $T(t_n)$. This is the basic equation that can be iterated from $n = 0$ to give the temperature T at subsequent steps $n = 1, 2, \cdots$. The routine `CoffeeCool` (see appendix 1.A) implements this algorithm which is known as the *Euler algorithm*. Figure (1.1) shows the numerical solution of (1.1) for some values of cooling rate r. The parameters have been set to $T_0 = 87\ ^0C$, $T_s = 17\ ^0C$, and $\Delta t = 0.1$ s. We have compared our numeric solution with the analytical one given by (1.2). You see there is a very good agreement between computed and analytical results. This shows that the Euler algorithm has a good performance in solving linear first-order differential equations.

1.1.2 Nonlinear 1st order differential equations

In the coffee cooling problem, we encountered a linear first-order differential equation which was numerically solved by the Euler algorithm. The Euler algorithm can also be implemented to solve non-linear first-order equations in the following general form:

$$\frac{dx}{dt} = f(x,t) \tag{1.6}$$

Where $f(x,t)$ is a given function. Note the independent variable t need not necessarily be time and the dependent variable x need not be position. Let x_n and f_n be shorthand notations for $x(t_n)$ and $f(x_n,t_n)$ correspondingly where t_n is the $n-th$ timestep. One can obtain the numerical solution of (1.6) according to the Euler algorithm as follows:

$$x_{n+1} = x_n + \Delta t f(x_n,t_n) \tag{1.7}$$

If we want a better approximation we should resort to higher-order terms in the Taylor expansion. To this end, we have:

$$x_{n+1} = x(t_n+\Delta t) = x(t_n)+\Delta t\frac{dx}{dt}(t_n)+\frac{1}{2}(\Delta t)^2\frac{d^2x}{dt^2}(t_n)+O(\Delta t)^3 \tag{1.8}$$

From (1.6) we can replace $\frac{dx}{dt}(t_n)$ and $\frac{d^2x}{dt^2}(t_n)$ with $f(x_n,t_n)$ and $\frac{df}{dt}(x_n,t_n)$ respectively. To evaluate the latter one we proceed as follows:

$$\frac{df}{dt}(x_n,t_n) = \frac{\partial f}{\partial x}(x_n,t_n)\frac{dx}{dt}(x_n,t_n) + \frac{\partial f}{\partial t}(x_n,t_n) \tag{1.9}$$

Using (1.6) it can be simplified as follows:

$$\frac{df}{dt}(x_n,t_n) = \frac{\partial f}{\partial x}(x_n,t_n)f(x_n,t_n) + \frac{\partial f}{\partial t}(x_n,t_n) \tag{1.10}$$

Therefore, the second-order Taylor expansion gives:

$$x_{n+1} = x_n + \Delta t f(x_n,t_n) + \frac{1}{2}(\Delta t)^2[\frac{\partial f}{\partial x}(x_n,t_n)f(x_n,t_n)+$$

$$\frac{\partial f}{\partial t}(x_n,t_n)] + O(\Delta t)^3 \tag{1.11}$$

Given f all the terms on the right-hand side of (1.11) can be evaluated. We can go to higher terms in the Taylor expansion but the manipulations become more cumbersome. For more details, see other references (Pang, 2006; Scherer, 2010; Rubin H. Landau and Bordeianu, 2008).

1.1.3 Radioactive decay

Another example that involves a first-order linear differential equation is the decay of radioactive nuclei. Many unstable nuclei can decay into smaller atoms. This phenomenon is random in nature and we can only speak of the decay probability. A typical example is the nucleus of a Uranium isotope ^{238}U which can decay into a lighter atom ^{234}Th. Suppose we have $N(t)$ Uranium atoms at time t. If we denote the decay rate by λ, during the infinitesimal time interval $[t, t + \Delta t]$ on average a small number of Uranium atoms decay. This number will be proportional to the number of atoms $N(t)$ and the time interval Δt. The proportionality constant is the decay rate λ. Therefore we can write the following equation:

$$N(t) - N(t + \Delta t) = \lambda \Delta t N(t) \tag{1.12}$$

Even though the number of nuclei should be an integer number but we often can treat this number as a continuous variable and end up with the following first-order linear differential equation in the limit $\Delta t \to 0$:

$$\frac{N(t + \Delta t) - N(t)}{\Delta t} = \frac{dN(t)}{dt} = -\lambda N(t) \tag{1.13}$$

Equation (1.13) can simply and analytically be solved. The solution turns out to be:

$$N(t) = N_0 e^{-\lambda t} \tag{1.14}$$

In which N_0 is the initial number of uranium Nuclei at $t = 0$. Note the decay rate λ has the dimension of inverse time. We can solve (1.13) numerically as we did for the coffee cooling problem. Denoting $N(t_n)$ by N_n we can turn (1.13) into a finite difference equation:

$$\frac{N_{n+1} - N_n}{\Delta t} = -\lambda N_n \to N_{n+1} = N_n(1 - \lambda \Delta t) \tag{1.15}$$

The routine `NuclearDecay` (see appendix 1.B for details) implements the above Euler algorithm to find the numerical solution of (1.13). Figure (1.2) compares the numerical solutions with the analytical ones for some choices of decay rate λ. As you see there is an excellent agreement between numerical and analytical solutions. A useful concept in nuclear decay is half-life $T_{\frac{1}{2}}$ which is the time that half of the nuclei

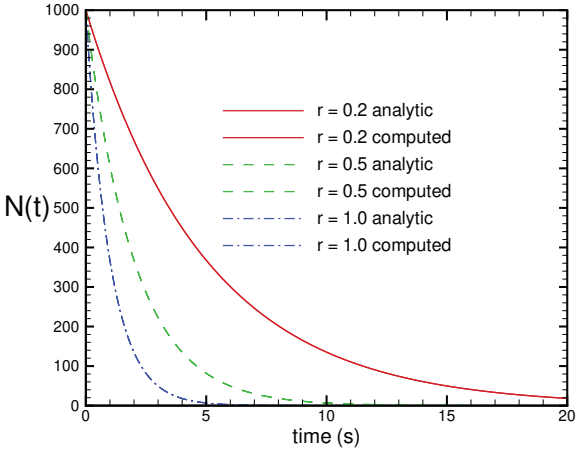

Figure 1.2: *Computed number of undecayed nuclei versus time for various values of the decay constant λ by Euler algorithm with $\Delta t = 0.01$. The number of radioactive nuclei has been initially set to $N_0 = 1000$. Comparison is made with the analytical solution.*

decay. Theoretically, it is related to decay rate λ as $T_{\frac{1}{2}} = \frac{ln2}{\lambda}$. Numerically we can compute $T_{\frac{1}{2}}$ via a simple `if` command in the subroutine. In the time loop, we can simply find the timestep t^* at which the condition $N(t^*) = \frac{N_0}{2}$ is satisfied. Half-life is then found to be $T_{\frac{1}{2}} = t^* \Delta t$. A more realistic situation is the double decay process. Let us pose this problem exactly as it is presented in problem four from chapter one of the book (Giordano and Nakanishi, 2006).

Problem:

Consider a radioactive decay problem involving two types of nuclei A and B with population $N_A(t)$ and $N_B(t)$. Suppose type A nuclei decay to type B which then also decay according to the following differential equations:

$$\frac{dN_A(t)}{dt} = -\lambda_A N_A(t) \tag{1.16}$$

$$\frac{dN_B(t)}{dt} = \lambda_A N_A(t) - \lambda_B N_B(t) \tag{1.17}$$

where λ_A and λ_B are decay constants for each type of nucleus. Use the Euler method to solve these coupled equations for $N_A(t)$ and $N_B(t)$ as a function of time. Explore the behaviour found for different values of $\frac{\lambda_A}{\lambda_B}$.

Before trying to solve the problem numerically the exact analytical solution is presented. Equations (1.16) and (1.17) form a linear set of first-order differential equations. Assume at $t = 0$ there are N_{0A} and N_{0B} nuclei respectively. Equation (1.16) simply gives:

$$N_A(t) = N_{0A} e^{-\lambda_A t} \tag{1.18}$$

Replacing $N_A(t)$ from (1.18) into (1.17) we find:

$$\frac{dN_B(t)}{dt} = \lambda_A N_{0A} e^{-\lambda_A t} - \lambda_B N_B(t) \tag{1.19}$$

Equation (1.19) is an inhomogeneous first-order differential equation for $N_B(t)$. Its solution comprises the sum of a special solution of the inhomogeneous equation plus an arbitrary solution of the homogeneous equation. To find the special solution we assume $N_B^s(t) = ae^{bt}$. Putting this into (1.19) we arrive at $abe^{bt} = \lambda_A N_{0A} e^{-\lambda_A t} - \lambda_B a e^{bt}$. The only way for this equation to be satisfied at an arbitrary time t is that $b = -\lambda_A$ to have a common exponential factor $e^{-\lambda_A t}$ on both sides. With this b the unknown a turns out to be $a = \frac{\lambda_A}{\lambda_B - \lambda_A} N_{0A}$ which then gives the special solution as follows: $N_B^s(t) = \frac{\lambda_A}{\lambda_B - \lambda_A} N_{0A} e^{-\lambda_A t}$. Having found the special solution, $N_B(t)$ can be written as follows:

$$N_B(t) = N_B^s(t) + Ce^{-\lambda_B t} \tag{1.20}$$

where the constant C is determined by requiring that $N_B(t)$ satisfies the initial condition $N_B(0) = N_{0B}$. Constant C simply is found to be: $C = N_{0B} - \frac{\lambda_A}{\lambda_B - \lambda_A} N_{0A}$. Putting this C into (1.20) $N_B(t)$ becomes:

$$N_B(t) = \frac{\lambda_A}{\lambda_B - \lambda_A} N_{0A} (e^{-\lambda_A t} - e^{-\lambda_B t}) + N_{0B} e^{-\lambda_B t} \tag{1.21}$$

To solve the problem numerically we show the number of nuclei A and B in timestep n by N_A^n and N_B^n respectively. The application of the

Euler algorithm with a timestep τ gives:

$$N_A^{n+1} = N_A^n - \tau\lambda_A N_A^n = N_A^n(1-\tau\lambda_A) \tag{1.22}$$

$$N_B^{n+1} = N_B^n + \tau(\lambda_A N_A^n - \lambda_B N_B^n) = N_B^n(1-\tau\lambda_B) + \tau\lambda_A N_A^n \tag{1.23}$$

Figure (1.3) sketches the computed N_A and N_B versus time obtained by the programme `DoubleNuclearDecay` (see appendix 1.C for details). Comparison to the analytical solution is also shown. The parameters are chosen as follows: $N_{0A} = 1000, N_{0B} = 200, \lambda_A = 0.2$ and $\lambda_B = 0.1$. As you see N_B increases first and then after reaching a maximum and then it decreases exponentially to zero. The reason is that we have chosen $\lambda_A > \lambda_B$ so that in the early stages of the decay process, the number of decayed nuclei of type A exceeds the number of B nuclei therefore $N_B(t)$ increases. After a sufficient time, there remain fewer A nuclei hence the source of B nuclei production smears off, and the decay process dominates the production. There is an excellent agreement between computed and analytical solutions. We see the Euler algorithm also shows a successful performance in dealing with a linear set of first-order differential equations.

1.2 Nonlinear differential equation: Population Growth

As another application of first-order differential equations, we consider the growth and extinction of populations. In contrast to previous examples, the growth problem often involves nonlinear differential equations. Let us begin with a simple solvable nonlinear problem which is posed as a problem in chapter one of (Giordano and Nakanishi, 2006). Suppose the number of individuals in a population is $N(t)$ at time t. The population number can grow over time through the birth process. The number of newly born individuals per time unit is assumed to be proportional to $N(t)$ with proportionality constant a. The population number can decrease over time due to the death process. The death rate can be proportional to $N^2(t)$ to allow for the fact that food will become harder to find when the population number becomes larger. One can then write the following nonlinear rate equation:

$$\frac{dN(t)}{dt} = aN(t) - bN^2(t) \tag{1.24}$$

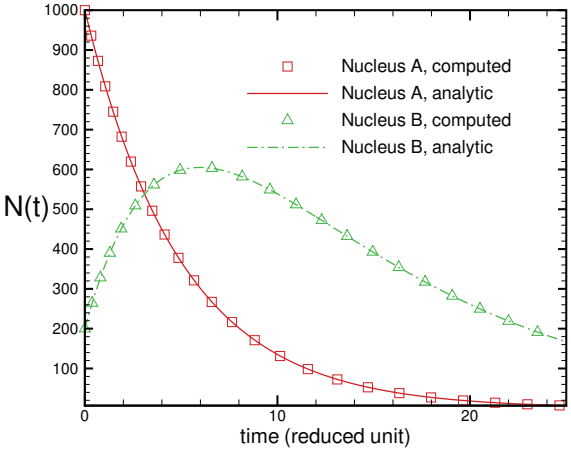

Figure 1.3: *Number of undecayed nuclei A and B versus time. The Euler algorithm with a timestep $\tau = 0.01$ has been implemented. Comparison is made with analytical solution. Decay rates are $\lambda_A = 0.2$ and $\lambda_B = 0.1$.*

Hopefully, equation (1.24) can be solved exactly. For this purpose, we simply write it as $\frac{dN}{aN-bN^2} = dt$ and integrate it from both sides. Supposing that at $t = 0$ the population number is $N(0) = N_0$. We have:

$$\frac{dN}{aN - bN^2} = dt \Rightarrow \int_{N_0}^{N(t)} \frac{dN}{aN - bN^2} = \int_0^t dt = t \tag{1.25}$$

Concerning the identity $\frac{1}{aN-bN^2} = \frac{1}{aN} + \frac{b}{a^2-abN}$ we find:

$$t = \int_{N_0}^{N(t)} (\frac{dN}{aN} + \frac{bdN}{a^2 - abN}) = \frac{1}{a}[ln\frac{N(t)}{N_0} - ln\frac{a - bN(t)}{a - bN_0}] \tag{1.26}$$

Equation (1.26) gives: $at = ln\frac{(a-bN_0)N(t)}{N_0[a-bN(t)]}$ which then implies:

$$\frac{N_0}{a - bN_0}e^{at} = \frac{N(t)}{a - bN(t)} \tag{1.27}$$

A simple algebra gives $N(t)$ as follows:

$$N(t) = \frac{aN_0 e^{at}}{a + bN_0(e^{at} - 1)} \tag{1.28}$$

In the long-time limit $t \gg \frac{1}{a}$ one finds $N \to N_s = \frac{a}{b}$. Note that we could obtain the stationary solution by setting the left-hand side of (1.24) equal to zero. Now let us see to what extent the numerical solution agrees with the exact one. As usual, we incorporate the Euler algorithm which is implemented as $\frac{N^{n+1}-N^n}{\tau} = aN^n - b(N^n)^2$. We find:

$$N^{n+1} = N^n(1 + a\tau - b\tau N^n) \tag{1.29}$$

Figure (1.4) shows the numerical solution, based on the iteration in (1.29). It has been obtained by the programme `PopulationGrowth` (see Appendix 1.D for more details). The numerical solution is also compared to the analytical solution. As you see the agreement between computed and analytical results is still very good. We have chosen the time unit such that $a = 1$ and have varied the death rate b. The initial population number is set to $N_0 = 1000$. Let us see what happens if N_0 becomes smaller. Figure (1.5) shows $N(t)$ for $N_0 = 5$. Here you see a different behaviour. Since the initial population is small the death term i.e.; $-bN^2$ would be very small in the early stages of the dynamics and therefore the birth term gives rise to the population increase. After a sufficient time, when the population has grown considerably, the death rate becomes large and does not allow for further population increase. As a result of competition between birth and death processes, the system comes to a stationary behaviour at $N_s = \frac{a}{b}$.

1.3 Master equation

Another common application of first-order differential equations appears in random processes. Suppose the outcome of an event can be realised in M different ways. The best example is throwing a die. In this case, the outcome event is the number shown by the top face of the dice. The number can be one to six ($M = 6$). When the stochastic dynamics occur in continuous time one can speak from the rate which is the probability of occurrence of an event per time unit. Let $P(n, t)$ denote the probability that at time t the nth outcome occurs

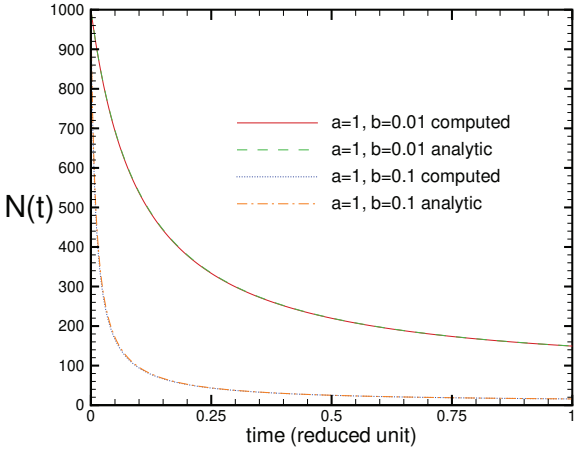

Figure 1.4: *Number of population individuals versus time with* $N_0 = 1000$. *The Euler algorithm with a timestep* $\tau = 0.001$ *has been used. Comparison is made with analytical solution.*

$(n = 1, 2, \cdots, M)$. Let $w_{n,m}$ denote the probability of transition from state n to state m $(m \neq n)$ during the infinitesimal interval $[t, t + \delta t]$. One can write the following first-order differential equation for $P(n, t)$ (van Kampen, 1992):

$$\frac{\partial P(n,t)}{\partial t} = \sum_{m \neq n} [P(m,t)w_{m,n} - P(n,t)w_{n,m}] \tag{1.30}$$

To see the derivation of (1.30) you may refer to any standard textbook on stochastic dynamics and statistical physics. I can suggest a very good one: (Reichel, 1998) (see chapter five). Equation (1.30), known as the *Master equation*, is linear and first order but the subtle point is that it is not local in the discrete state variable n and that is the point that makes its solution hard to find, if not impossible, in general. In a few special cases, we can analytically solve the Master equation. Despite our goal is not to solve the problems analytically but for didactic purposes, we prefer to start with the problems which are amenable to exact solutions. As one of these examples let us come

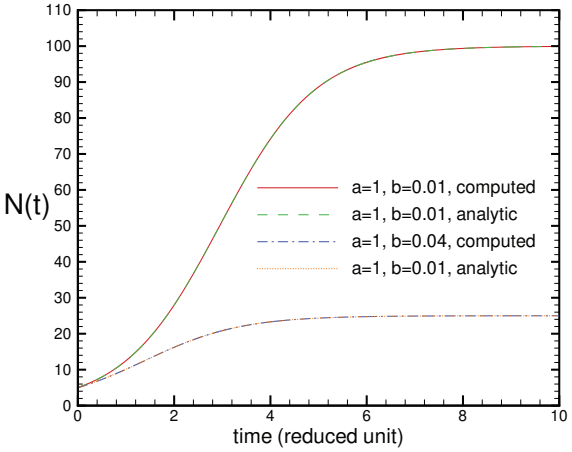

Figure 1.5: *Number of population individuals versus time with a smaller initial population number $N_0 = 5$. The Euler algorithm with a timestep $\tau = 0.001$ has been implemented.*

back again to the birth-death process. Suppose we have a population. In a more realistic description, the population number at time t can be any number $0 \leq n < \infty$. During the infinitesimal time interval δt the population number n can change by one number. It can increase to $n+1$ if a birth process occurs or decrease to $n-1$ if a death process happens. For simplicity, we assume that the birth and death rates depend linearly on the current population number n. The proportionality constants are taken as β and γ respectively. In terms of the transition rates, the only non-zero ones are those in which n and m differ by one. Therefore we have $w_{n,n+1} = \beta n$ and $w_{n,n-1} = \gamma n$. We can now write the Master equation for this random process:

$$\frac{\partial P(n,t)}{\partial t} = \beta(n-1)P(n-1,t)+\gamma(n+1)P(n+1,t)-(\beta+\gamma)nP(n,t) \tag{1.31}$$

We recall that equation (1.31) is not local in variable n and this makes the analytic solution cumbersome. However, it is possible to exactly find the mean number of the population and its deviation from the

mean value. This is elegantly done by the method of generating function in chapter five of (Reichel, 1998). We refer interested readers to this excellent book for details and only quote the main results here. The mean number of population $\langle n(t)\rangle$ is defined as follows:

$$\langle n(t)\rangle = \sum_{n=-\infty}^{\infty} nP(n,t) \tag{1.32}$$

It should be noted that the physical range for n starts from zero and we should simply set $P(n,t) = 0$ for $n < 0$. Defining a generating function $F(z,t)$:

$$F(z,t) = \sum_{n=-\infty}^{\infty} z^n P(n,t) \tag{1.33}$$

We simply find:

$$\langle n(t)\rangle = \frac{\partial F(1,t)}{\partial z}; \quad \langle n^2(t)\rangle - \langle n(t)\rangle = \frac{\partial^2 F(1,t)}{\partial z^2} \tag{1.34}$$

Higher moments are obtained similarly. With the help of the Master equation (1.31) and the definition of generating function $F(z,t)$ we can obtain the following first-order partial differential equation for $F(z,t)$ (see (Reichel, 1998) for details).

$$\frac{\partial F(z,t)}{\partial t} = (z-t)(\beta z - \gamma)\frac{\partial F(z,t)}{\partial z} \tag{1.35}$$

It is possible to solve (1.35) by the method characteristics (Sneddon, 1957). This method is explained in many mathematical physics books such as (Myint-U and Debnath, 2007). Another good reference is (Hassani, 2013). The application of this method to (1.35) with initial condition $P(n,0) = \delta_{m,n}$ gives:

$$F(z,t) = \left(\frac{\gamma(z-1)e^{(\beta-\gamma)t} - \beta z + \gamma}{\beta(z-1)e^{(\beta-\gamma)t} - \beta z + \gamma}\right)^m \tag{1.36}$$

Taking the partial derivatives of $F(z,t)$ with respect to variable z according to (1.34) gives:

$$\langle n(t)\rangle = me^{(\beta-\gamma)t} \tag{1.37}$$

$$\langle n^2(t)\rangle - \langle n(t)\rangle^2 = m(\frac{\gamma+\beta}{\gamma-\beta})e^{(\beta-\gamma)t}(1-e^{(\beta-\gamma)t}) \tag{1.38}$$

Now it is about time we solved the problem numerically and enjoy its beauty and convenience. Let us see if our Euler algorithm can give a satisfactory result or not. Because we have used n to denote the population number we denote the time step counter by k. Let P_n^k be a short notation for $P(n, k\tau)$ where τ is the timestep. The Master equation (1.31) turns into the following finite difference form:

$$P_n^{k+1} = P_n^k + \tau\beta(n-1)P_{n-1}^k + \tau\gamma(n+1)P_{n+1}^k - \tau(\beta+\gamma)nP_n^k \tag{1.39}$$

The discretised initial condition implies $P_n^0 = \delta_{m,n}$. Equation (1.39) allows for finding the probabilities at time step $k+1$ from their values at time step k. Nevertheless, to implement the Euler scheme to (1.39) we encounter a problem immediately! To evaluate P_0^1 we need to know the unphysical quantity P_{-1}^0. We need not worry because we know the physical range of n does not include negative numbers so we can remedy this problem by introducing a boundary condition $P(n,t) = 0$ whenever n becomes negative. But be careful! there is another problem in front of us. What should be done with the upper limit of n? In principle, n can go to infinity but we cannot treat this infinite limit by computer and will have to consider a finite upper limit for n. Let us suppose $n \leq M$ where M should be taken as large as possible. Now another problem arises when we want to evaluate P_M^1. In fact (1.39) gives the unphysical term P_{M+1}^0. A plausible assumption could be to set $P_{M+1}^0 = P_M^0$. More generally by setting:

$$P_{-1}^k = 0; \quad P_{M+1}^k = P_M^k \tag{1.40}$$

We can iterate (1.39) to any desired time step. The right-hand side of equation (1.40) can approximately be a finite difference form of the Neumann boundary condition $\frac{\partial P(\infty,t)}{\partial n} = 0$. Another possibility (method two) is to set $P_{M+1}^k = 0$. Specifying the right boundary condition we can iterate (1.39) in time and find $P_n^k \quad k = 1, 2, \cdots$ in the physical range $n = 0, 1, 2, \cdots$. The programme `PopulationMaster` (see appendix 1.E for details) implements the Euler algorithm with a time step $\tau = 0.01$ for solving the Master equation (1.31) numerically. Figure (1.6) exhibits the numerical solution for case $\beta = 0.1$ and $\gamma = 0.2$. We have set $M = 100$ and $m = 20$ (initial number of

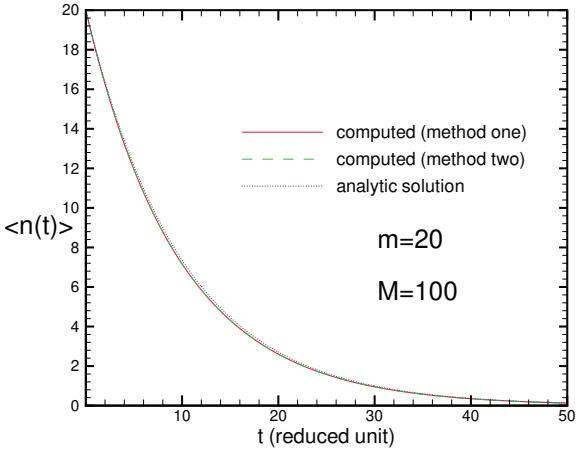

Figure 1.6: *The average number of population* $\langle n(t)\rangle$ *versus time. The Euler algorithm with a timestep* $\tau = 0.01$ *is used. The parameters are* $\beta = 0.1, \gamma = 0.2, M = 100$ *and* $m = 20$.

population). You see both methods one and two give a result that is in excellent agreement with the analytical solution (1.37). Now let us interchange the values of β and γ that is to set $\beta = 0.2$ and $\gamma = 0.1$. Figure (1.7) shows the result. We see entirely different behaviour. Now the computed answer deviates notably from the analytical one. The reason is certainly not due to the weakness of the Euler algorithm but to the inappropriateness of the right boundary condition. In the case where the birth rate is larger than the death rate, ($\beta > \gamma$) we know from the analytic solution that the average number of the population grows exponentially in time (see equation (1.37)). Accordingly, we can conclude that when t has increased the probability to have a larger population increases. This implies that for a fixed time t, the larger the n the larger $P(n,t)$ will be. Consequently, the boundary condition for P^k_{M+1} should be treated carefully. Can you suggest a more reasonable condition to reproduce the large-time behaviour of the exponentially growing population? As our final example of the birth-death process, we consider the following non-linear problem (Nicolis and Prigogine, 1977). This time let us formulate the problem in the context of chem-

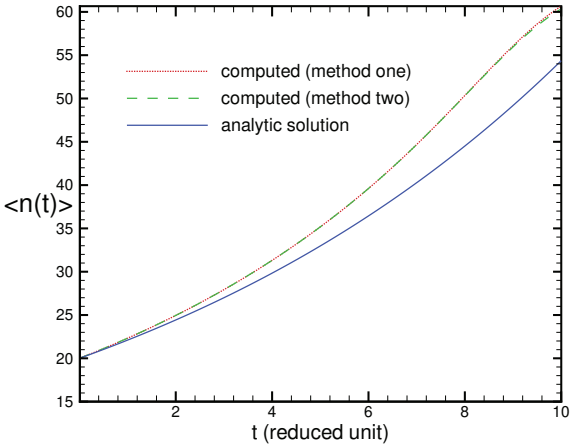

Figure 1.7: *The average number of population* $\langle n(t)\rangle$ *versus time. The Euler algorithm with a timestep* $\tau = 0.01$ *is used. The parameters are* $\beta = 0.2, \gamma = 0.1, M = 100$ *and* $m = 20$.

ical reactions. Consider a binary chemical reaction that occurs with the rate k (D. A. Mcquarrie and Russel, 1994).

$$2X \to B \tag{1.41}$$

The transition rate for this reaction is proportional both to the rate k and the number of different ways $\frac{1}{2}n_X(n_X-1)$ a pair of X molecules react to form a B molecule if there are n_X molecules in the system. Suppose before the reaction the number of X and $B-$type molecules are n_X+2 and n_B-1 respectively. After the reaction, we have n_X and n_B molecules of types X and B respectively. The transition rate turns out to be $\frac{k}{2}(n_X+2)(n_X+1)$. We can write the following Master equation for the probability to have n molecules of type X at time t:

$$\frac{\partial P(n,t)}{\partial t} = \frac{k}{2}(n+2)(n+1)P(n+2,t) - \frac{k}{2}n(n-1)P(n,t) \tag{1.42}$$

The natural boundary condition implies $P(-1,t) = 0$. Let us solve the problem numerically by the Euler method. Figure (1.8) shows the

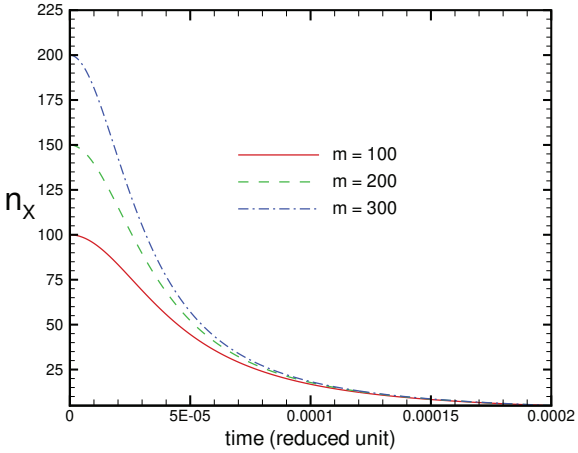

Figure 1.8: *Time dependence of X type molecules average number. The Euler algorithm with a timestep $\tau = 10^{-6}$ is used. The parameters are $k = 1$ and $m = 100, 200$ and 300.*

average number of X molecules versus time for some initial values m. It should be noted that we do not obtain the correct result unless the timestep is chosen very tiny. We have set $\tau = 10^{-6}$ to get a good result when the initial number of X molecules m is of order 10^2. For larger values of m, the timestep τ should be even smaller. The reason is due to the nonlinear term $n(n+2)$ in the Master equation. Before ending this chapter we should like to mention that the favourite Euler algorithm has shown very good performance in first-order differential and difference equations even when they are not local in their state variables or even nonlinear. I know you are curious to see what this simple algorithm can do when encountering higher-order differential equations. Chapter two gives you the answer!

1.4 Problems

Problem 1.1 Numerically solve the coffee-cooling problem when the cooling rate r is not constant. It would be reasonable to assume that

$r(t)$ is a decreasing function of time. for instance, take $r(t) = r_0 e^{-\alpha t}$. Explore the solution for values $\alpha = 1, 2, 3$ per minute. Take r_0 the same as the constant r in figure (1.1). Compare your results with the one presented in figure (1.1).

Problem 1.2 Suppose you have K radioactive nuclei. Each nucleus can decay into a smaller one at its own rate. Denote the number of nuclei at time t by $N_1(t), \cdots, N_K(t)$ and the decay rates by $\lambda_1, \cdots, \lambda_K$.

(a) Try to analytically solve the following linear set of decay differential equations.

$$\frac{dN_1}{dt} = -\lambda_1 N_1(t); \ \frac{dN_2}{dt} = \lambda_1 N_1(t) - \lambda_2 N_2(t); \cdots$$

$$\frac{dN_K}{dt} = \lambda_{K-1} N_{K-1}(t) - \lambda_K N_K(t) \tag{1.43}$$

For the initial condition assume $N_1(0) = N_0$ and $N_k(0) = 0 \ \ k = 2, \cdots, N$. The analytical answer turns out to be (Foulaadvand, 2006):

$$N_i(t) = N_0 \lambda_{i-1} \lambda_{i-2} \cdots \lambda_1 \sum_{k=1}^{N} \frac{e^{-\lambda_k t}}{\prod_{j=1(j\neq k)}^{k} (\lambda_j - \lambda_k)} \tag{1.44}$$

(b) Numerically solve the set of equations by the Euler algorithm and compare your result with the analytical solution.

Problem 1.3 The master equation for chemical reaction kinetics $A + X \Leftrightarrow A + Y$ in which the number of molecules A remains constant turns out to be:

$$\dot{P}_n(t) = k_2(N-n+1)P_{n-1}(t) + k_1(n+1)P_{n+1}(t) - [k_1 n + k_2(N-n)]P_n(t) \tag{1.45}$$

N is the total number X molecules plus Y molecules (remains constant). If n denotes the number of X molecules,

(a) numerically solve this equation and obtain the mean number of n that is $\langle n \rangle$.

(b) Numerically obtain $\langle n^2 \rangle$ and then calculate the standard deviation of n.

(c) Can you solve the problem analytically?

Problem 1.4 Consider a linear birth-death process that includes the possibility of a change in the population due to immigration. Let α be the immigration rate which is the probability in unit time that an individual enters society. Assuming that at $t = 0$ there are m individuals in the society,

(a) write the master equation and obtain the generating function $F(z,t) = \sum_{n=-\infty}^{\infty} z^n P(n,t)$ for the process.

(b) Numerically compute the mean population number $\langle n \rangle$.

(c) Numerically obtain the process variance $\langle n^2 \rangle - \langle n \rangle^2$.

(d) Try to solve the problem analytically.

Problem 1.5 We intend to discuss the oscillatory and chaotic behaviours in chemical reactions. To obtain oscillations, it is essential to have a series of chemical reactions such that the products of some reactions are the reactants of others. In the following, we consider a simple set of reactions that can lead to oscillations under certain conditions (Lefever and Nicolis, 1971)

$$A \to X;\ B + X \to Y + D;\ 2X + Y \to 3X;\ X \to C \tag{1.46}$$

Assume that the density of constituents A and B are held constant and taking the rate constant be equal to unity, the corresponding rate equations turn out to be:

$$\frac{dX}{dt} = A - B - BX + X^2 Y;\ \frac{dY}{dt} = BX - X^2 Y \tag{1.47}$$

(a) Obtain the steady-state concentrations of molecules X and Y by setting the above equations equal to zero. Show that the steady values are $(X_s, Y_s) = (A, \frac{B}{A})$.

(b) By writing a programme, try to numerically solve Eq. (1.47) for given initial concentrations $X(0)$ and $Y(0)$ and fixed values of A and B.

Chapter 2

2nd order differential equations in physics

2.1 Introduction

In this chapter, we will consider second-order ordinary differential equations in physics particularly, in classical mechanics. Inarguably the most prominent second-order differential equation in physics is the Newton equation of motion for a point-like particle of mass m:

$$m\frac{d^2\mathbf{r}}{dt^2} = \mathbf{F}(\mathbf{r}, \mathbf{v}, t) \tag{2.1}$$

In equation (2.1) the force acting on the particle can depend, in its most general form, on the particle's instantaneous position $\mathbf{r}$, its velocity $\mathbf{v}$ as well as time t. Equation (2.1) is second-order in time and consequently, two initial conditions $\mathbf{r}(0)$ and $\mathbf{v}(0)$ should be specified to solve the equation. There are several examples in physics where you can exactly solve (2.1). You can refer to standard textbooks on classical mechanics to see these examples (Symon, 1971; Thornton and Marion, 2003; Fowles and Cassiday, 2004; Arya, 1997). Here we emphasize on the cases where exact analytical solutions are not exactly known. Of course, for pedagogical purposes, we normally begin with problems having analytical solutions so that we can compare our numerical solutions with analytical ones. It is better to turn

(2.1) into two first-order differential equations as follows:

$$\frac{d\mathbf{r}}{dt} = \mathbf{v} \tag{2.2}$$

$$\frac{d\mathbf{v}}{dt} = \mathbf{a} = \frac{\mathbf{F}}{m} \tag{2.3}$$

Equations (2.2) and (2.3) form a set of two by two linear system of first-order differential equations. It is possible to implement the Euler algorithm for discretising these equations and turning them into a finite difference form. Denoting the particle's position, velocity, and acceleration vectors at timestep n by $\mathbf{r}_n$, $\mathbf{v}_n$, and $\mathbf{a}_n$ the Taylor expansions of (2.2) and (2.3) up to the second term give:

$$\mathbf{r}_{n+1} = \mathbf{r}_n + \tau\mathbf{v}_n + o(\tau)^2 \tag{2.4}$$

$$\mathbf{v}_{n+1} = \mathbf{v}_n + \tau\mathbf{a}_n + o(\tau)^2 \tag{2.5}$$

In equations (2.4) and (2.5) τ is the timestep, $\mathbf{r}_n, \mathbf{v}_n$, and $\mathbf{a}_n$ are abbreviations for $\mathbf{r}(t_n), \mathbf{v}(t_n)$, and $\mathbf{a}(\mathbf{r}(t_n), \mathbf{v}(t_n), t_n)$ respectively. Having initial conditions i.e.; $\mathbf{r}_0$ and $\mathbf{v}_0$ we can iterate (2.4) and (2.5) to find the particle's position and its velocity in every time step. Let us begin our first example with the well-known problem of free fall near the Earth's surface.

2.2 Free fall near Earth's surface

Consider the motion of a particle of mass m in the gravitational field of the Earth. See figure (2.1). If we take into account the variation of the Earth's gravitational field with the distance from the Earth's centre, then the force on the particle is not constant. According to Newton's law of gravitation, the force due to the Earth on a particle of mass m is given by:

$$\mathbf{F} = -\frac{GmM}{(R+y)^2}\mathbf{j} \tag{2.6}$$

where y is measured from the Earth's surface, R is the Earth radius, M is the Earth mass and G is the gravitational constant. Note the

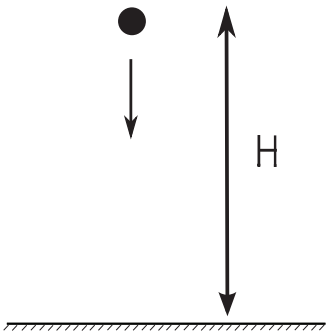

Figure 2.1: *A freely falling object in the Earth's gravitational field.*

positive direction of the y axis is taken outward from the Earth's surface. Showing $\frac{GM}{R^2}$ by g the Newton equation of motion becomes:

$$\mathbf{a} = \frac{d\mathbf{v}}{dt} = -\frac{g}{(1+\frac{y}{R})^2}\mathbf{j} \tag{2.7}$$

If we let $R \to \infty$ we will reach the simple equation $\mathbf{a} = -g\mathbf{j}$ and the equation of motion can be analytically solved easily. Keeping R finite there is no exact analytical solution for the Newton equation with (2.7) and it should be solved numerically. In this particular problem, the force does not depend on velocity and time but does depend on the particle's position y. Here we have $a_{yn} = -\frac{g}{(1+\frac{y_n}{R})^2}$. Suppose the particle is dropped from the height h above the Earth's surface with zero initial velocity. We intend to obtain the hit velocity to the ground. The programme `FreeFall` (see Appendix 2.A for details) numerically solves the motion of a free-falling body with the Euler algorithm. Figure (2.2) shows the dependence of the absolute value of the hit velocity difference $|\Delta V_{hit}| = |V_{hit}^{cons} - V_{hit}^{var}|$ on height h for the cases where the motion acceleration is constant g and position-dependent. As you see, by increasing the height h from which the particle has dropped the effect of variation of the gravitational field to the distance from the Earth's centre is amplified. The impact velocity itself is around 1000 m/s so the relative impact velocity difference is less than one percent even if the particle has dropped from fifty kilometres above the ground level. If the particle moves in the vicinity of the Earth we can neglect the variation of the gravitational field to distance from the Earth's centre hereafter unless otherwise stated.

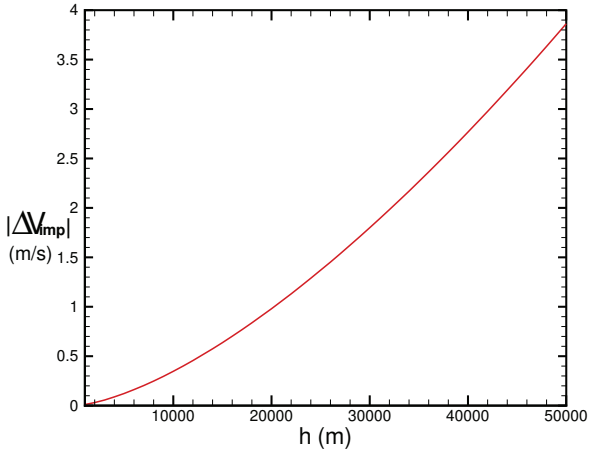

Figure 2.2: *Modulus of the hit velocity difference versus height. The Euler algorithm with a timestep $\tau = 10^{-3}$ is used. The particle initially dropped with zero velocity.*

2.3 Effect of air drag

Near the Earth, another important modification should be taken into consideration: *the effect of drag force due to air*. This resistive force depends on the particle's shape in a complicated manner if the object has a non-simple geometry. Furthermore, the motion of air plays an important role. We will restrict ourselves to simple cases where the particle has a simple geometry such as a sphere and the air motion is steady, uniform, and laminar. The direction of the drag force in simple cases where the object does not rotate will be opposite to the object's velocity and we can write:

$$\mathbf{F}_d(\mathbf{v}) = -F(v)\hat{\mathbf{v}} \tag{2.8}$$

where $v = |\mathbf{v}|$. Even with this approximation, $F(v)$ has a nontrivial dependence on the object's velocity modulus v. Two common dependence forms are normally discussed in the elementary literature. In the first case, $F(v)$ is linearly proportional to v which is $F(v) = c_1 v$, and in the second case which is phenomenologically ob-

served at high velocities, $F(v)$ is proportional to the square of velocity modulus: $F(v) = c_2 v^2$. In summary, we have:

$$\mathbf{F}_{1d}(\mathbf{v}) = -c_1\mathbf{v}: \quad \mathbf{F}_{2d}(\mathbf{v}) = -c_2 v\mathbf{v} \tag{2.9}$$

2.3.1 Linear air drag force

Now let us solve a problem in which the effect of air drag is assumed to have a linear dependence on velocity magnitude. The problem is: *Suppose we drop an object from height h above the ground level with zero initial velocity in the presence of air drag. What will be the hit velocity?*

As usual, we first try to solve the problem analytically. If there is no air resistance the answer is simply found to be: $v_{hit} = \sqrt{2gh}$. If there is air resistance the answer will be somewhat more difficult. In the case of linear drag force, the equation of motion becomes (positive y directs outward to the Earth):

$$\frac{dv}{dt} = -g - \frac{c_1}{m}v \tag{2.10}$$

Note here v is not the velocity modulus but the vertical component of velocity v_y. Initial conditions are $y(0) = h$ and $v(0) = 0$. Equation (2.10) is an inhomogeneous first-order linear differential equation and its solution turns out to be:

$$v(t) = \frac{gm}{c_1}(e^{-\frac{c_1 t}{m}} - 1) \tag{2.11}$$

As you see, for $t > 0$ the velocity component $v(t)$ is negative which indicates that the object is moving downward. To find the hitting velocity, we should be able to find the time the object hits the ground (flight time). Another integration from (2.11) gives:

$$y(t) = h - \frac{gm}{c_1}[\frac{m}{c_1}(e^{-\frac{c_1 t}{m}} - 1) + t] \tag{2.12}$$

To find the flight time we set $y = 0$ and reach the following transcendental equation:

$$\frac{c_1 h}{gm} = \frac{m}{c_1}(e^{-\frac{c_1 t}{m}} - 1) + t \tag{2.13}$$

Unfortunately, we are unable to exactly find the hitting time t from (2.13) and therefore cannot find the hitting velocity exactly. However, it is possible to proceed approximately. Suppose the quantity $\frac{c_1}{m}$ is small. Expanding $e^{-\frac{c_1}{m}t}$ up to the third order in (2.13) we arrive at the following cubic equation:

$$\frac{h}{g} = \frac{t^2}{2} - \frac{c_1 t^3}{6m}. \tag{2.14}$$

If there is no air drag we simply find $t = \sqrt{\frac{2h}{g}}$. When $c_1 \neq 0$ we avoid proceeding with the exact but complicated solution of the cubic equation. Alternatively, we consider that the hitting time has a power expansion in $\frac{c_1}{m}$. Keeping up to the third term we write t as:

$$t = \sqrt{\frac{2h}{g}} + \alpha\frac{c_1}{m} + \beta(\frac{c_1}{m})^2 \tag{2.15}$$

Replacing t from (2.15) in (2.14) and comparing the coefficients on both sides we find α and β as follows:

$$t = \sqrt{\frac{2h}{g}} + \frac{h}{3g}\frac{c_1}{m} + \frac{5\sqrt{2}}{36}(\frac{h}{g})^{\frac{3}{2}}(\frac{c_1}{m})^2 + o(\frac{c_1}{m})^3 \tag{2.16}$$

Putting the hitting time t from (2.16) into (2.11) one finds the hitting velocity. Now let us simulate the problem. We can easily implement the effect of air drag into our code. The programme `AirdragFall` (see Appendix 2.B for more details) does the job for us with the Euler algorithm. Figure (2.3) sketches the hitting velocity versus height h. Comparison is done to the free fall. We have also shown the simulation result by a better algorithm, namely the *Euler-Richardson* algorithm. We will soon explain this algorithm but for the moment you see the Euler algorithm performance is as good as this new algorithm. The air drag slows the object and hence the hit velocity is reduced.

2.3.2 Quadratic air drag force

Let us see the effect of the quadratic drag force on the problem. Since the motion is downward it is easier to take the y direction downward to

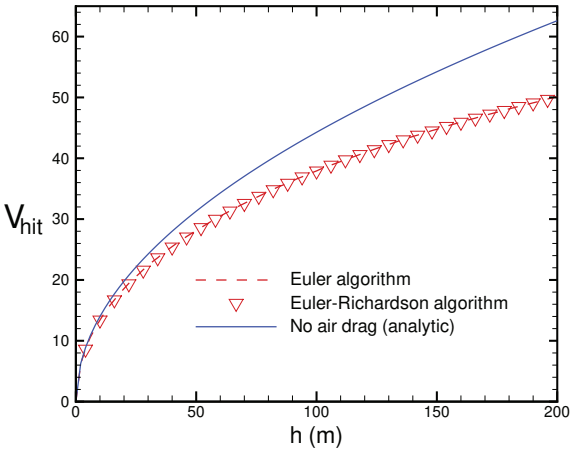

Figure 2.3: *Dependence of hitting velocity $v_{hit}(m/s)$ on the height h. The object has dropped with zero initial velocity. The Euler algorithm with a timestep $\tau = 0.01$ is used. The air drag constant c_1 has been set to* 0.1.

Earth and the origin at the initial position of the object. The equation of motion turns out to be:

$$\frac{dv}{dt} = g - \frac{c_2}{m}v^2 \tag{2.17}$$

where v is an abbreviation for the vertical component v_y of the velocity. We try to proceed analytically. An integration gives:

$$\int_0^{v(t)} \frac{dv}{g - \frac{c_2}{m}v^2} = t \tag{2.18}$$

Using the integral tables we find:

$$\int \frac{du}{a^2 - u^2} = \frac{1}{a}\coth^{-1}(\frac{u}{a}) + C \tag{2.19}$$

Implementing the initial conditions yields:

$$v(t) = \sqrt{\frac{mg}{c_2}}\coth(\sqrt{\frac{c_2 g}{m}}t) \tag{2.20}$$

Another integration gives (see integral tables):

$$y(t) = \frac{m}{c_2} ln[\sinh(\sqrt{\frac{c_2 g}{m}} t)] \tag{2.21}$$

By setting $y(t) = h$ one finds the hitting time:

$$t = \sqrt{\frac{m}{c_2 g}} \sinh^{-1}(e^{\frac{hc_2}{m}}) \tag{2.22}$$

By replacing the hitting time from (2.22) in (2.20) the hitting velocity v_h is found:

$$v_h = \sqrt{\frac{mg}{c_2}} \coth[\sinh^{-1}(e^{\frac{hc_2}{m}})] \tag{2.23}$$

Note the inverse of cosine hyperbolic is:

$$\sinh^{-1}(x) = ln(x + \sqrt{x^2 + 1}) \tag{2.24}$$

The interesting point is that we could solve the problem analytically when the air drag depends on v in a quadratic manner. Let us see how much the simulation results are close to the analytic ones. Figure (2.4) shows the dependence of the hit velocity on dropping height h for various values of the air drag coefficient c_2. The numerical results are in excellent agreement with the analytical result (2.23) (not shown in the graph). As you see, the larger the drag coefficient c_2 the less the magnitude of hitting velocity. Notice that if the drop height h is large enough, the particle reaches the terminal velocity $v_T = \sqrt{\frac{mg}{c_2}}$. For $c_2 = 0.1$ and 0.01 the height $h = 150\ m$ is quite enough for reaching the terminal velocity. Let us continue our numerical investigations by solving another problem on the one-dimensional motion of a particle in the Earth's gravitational field. This problem is put forward in (H. Gould and Chriastian, 2006) (problem 3-9).

Problem: *Suppose a pebble is thrown vertically upward with an initial velocity v_0. In the absence of air resistance, we know that the maximum height reached by the pebble is $\frac{v_0^2}{2g}$, its velocity upon return to the Earth equals v_0, the time of ascent equals the time of descent, and the total travel time is $\frac{2v_0}{g}$. We are interested to see the difference between the ascent and the descent times. Suppose there is an air drag*

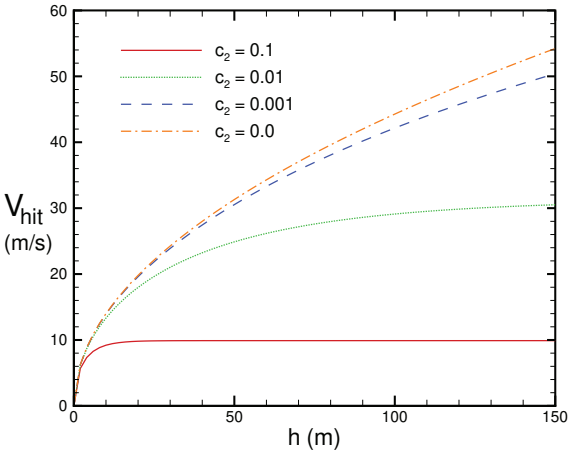

Figure 2.4: *Dependence of hitting velocity magnitude v_{hit} on dropping height h for the quadratic air drag. The object has dropped with zero initial velocity. The Euler algorithm with a timestep $\tau = 0.001$ is used.*

force that is proportional to the square of velocity magnitude.

Let us first try to solve the problem analytically as usual. In the ascending part of the motion (the y axis is taken upward with the origin on the ground) the equation of motion becomes:

$$\frac{dv}{dt} = -g - \frac{c_2}{m}v^2 \tag{2.25}$$

An integration gives:

$$\int_{v_0}^{0} \frac{dv}{v^2 + \frac{mg}{c_2}} = -\frac{c_2}{m}t_{asc} \tag{2.26}$$

in which t_{asc} denotes the ascent time at which the particle's velocity becomes zero and its height from the ground becomes maximum. The integral in the left-hand side of (2.26) can exactly be evaluated by the following indefinite integral from integral tables (see (Spiegel, 1998)

for example):

$$\int \frac{du}{a^2+u^2} = \frac{1}{a}\tan^{-1}(\frac{u}{a}) \tag{2.27}$$

It turns out that:

$$t_{asc} = \sqrt{\frac{m}{c_2 g}}\tan^{-1}(\sqrt{\frac{c_2}{mg}}v_0) \tag{2.28}$$

You can easily verify that in the limit $c_2 \to 0$ the ascent time t_{asc} approaches $\frac{v_0}{g}$. To find the maximum height H we first find the time dependence of velocity in the ascent part of the motion. Setting the upper limit of the integral in (2.26) to $v(t)$ we find:

$$v(t) = \sqrt{\frac{mg}{c_2}}\tan[\tan^{-1}(\sqrt{\frac{c_2}{mg}}v_0) - \sqrt{\frac{c_2 g}{m}}t] \tag{2.29}$$

Another integration, using the indefinite integral $\int \tan u du = -ln(\cos u)$, gives:

$$y(t) = \frac{m}{c_2}\{ln\cos[\tan^{-1}(\sqrt{\frac{c_2}{mg}}v_0) - \sqrt{\frac{c_2 g}{m}}t] - ln\cos[\tan^{-1}(\sqrt{\frac{c_2}{mg}}v_0)]\} \tag{2.30}$$

Replacing t by t_{asc} in (2.30) we can find the maximum height H:

$$H = y(t_{asc}) = -\frac{m}{c_2}ln\cos[\tan^{-1}(\sqrt{\frac{c_2}{mg}}v_0)] \tag{2.31}$$

Having found the maximum height H we are now able to find the descent time t_{dsc} analytically. One simply should replace h in (2.22) by H. It turns out:

$$t_{dsc} = \sqrt{\frac{m}{c_2 g}}\sinh^{-1}(\{\cos[\tan^{-1}(\sqrt{\frac{c_2}{mg}}v_0)]\}^{-1}) \tag{2.32}$$

The above relations for t_{asc}, t_{dsc} and H will reduce to their known values when $c_2 \to 0$. Let us verify this explicitly for H. In (2.31) if we keep up to the first term in the Taylor series $\tan^{-1} x = x + \frac{x^3}{3} + \cdots$ and up to the second term for $\cos x$ we find;

$$H = -\frac{m}{c_2}ln(1 - \frac{c_2}{2mg}v_0^2) \tag{2.33}$$

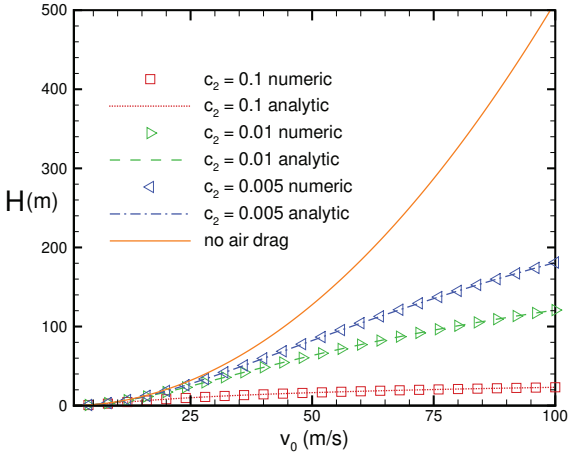

Figure 2.5: *The computed maximum height H versus v_0 for some values of c_2. The Euler algorithm with $\tau = 0.01$ is used. Comparison to analytic solution shows an excellent agreement.*

If we approximate $ln(1-x)$ by $-x$ we find the well-known result: $H = \frac{v_0^2}{2g}$. Figure (2.5) shows the dependence of the computed maximum height H on v_0 for some values of drag coefficient c_2 obtained by the programme `AirDragAscend` (see Appendix 2.C for more details). Comparison is made with the analytical solution (2.31) and also to the case with no air drag. In figure (2.6) we have sketched the difference between computed descent and ascent times $\Delta T = t_{dsc} - t_{asc}$ versus v_0 for some values of c_2. Comparison to the analytical solution shows excellent agreement.

2.4 Two-dimensional projectile motion

We are all familiar with the two-dimensional motion of a projectile in the absence of air resistance perhaps from high school physics. In a typical example, a cannonball is fired from the ground at angle θ_0 to the horizon with an initial velocity v_0. The maximum height H, the

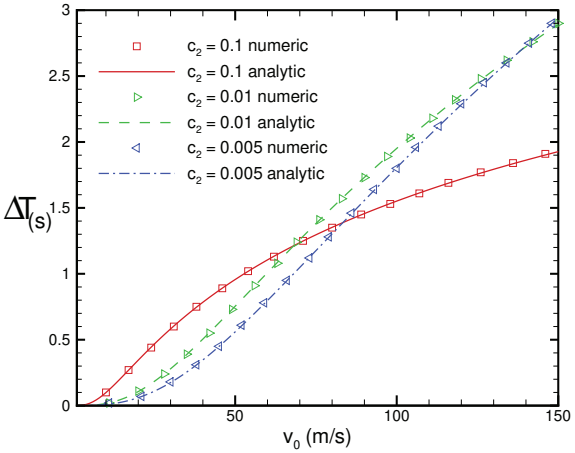

Figure 2.6: *The computed $\Delta T = t_{dsc} - t_{asc}$ versus v_0 for some values of c_2. Comparison to analytic solution shows an excellent agreement. The Euler algorithm with a timestep $\tau = 0.01$ is used.*

flight time T, and the range R turn out to be:

$$H = \frac{v_0^2 \sin^2 \theta_0}{2g};\ T = \frac{2v_0 \sin \theta_0}{g};\ R = \frac{v_0^2 \sin 2\theta_0}{g} \tag{2.34}$$

In this section, we intend to numerically study the air resistance effect on the characteristics of a two-dimensional projectile motion. Suppose the projectile mass is m and it is subjected to a constant gravitational field which is directed in the $-\mathbf{j}$ direction (positive y direction points upward to Earth). Moreover, there is an air drag force $\mathbf{F}_d$ which opposes the projectile velocity $\mathbf{v}$. See figure (2.7) for illustration.

2.4.1 Linear air drag force

If the magnitude of $\mathbf{F}_d$ is proportional to v, the Newton equation of motion becomes (positive y direction outward to Earth):

$$m\frac{d\mathbf{v}}{dt} = -mg\mathbf{j} - c_1 v\hat{v} = -mg\mathbf{j} - c_1\mathbf{v} \tag{2.35}$$

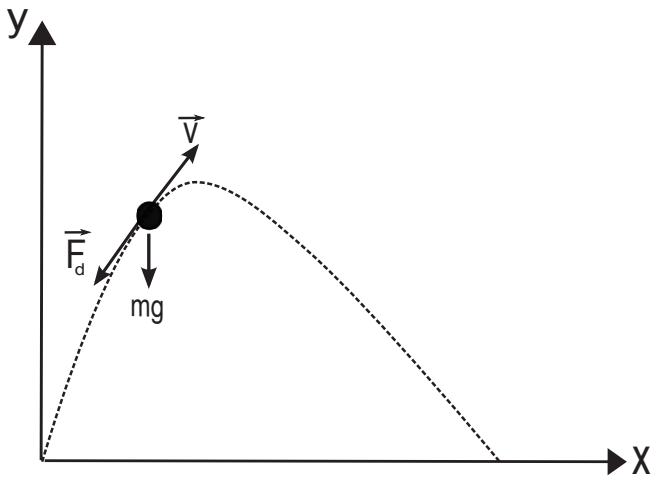

Figure 2.7: *The projectile trajectory in the Earth's gravitational field with air drag.*

In terms of components, we obtain the following set of linear uncoupled differential equations:

$$m\frac{dv_x}{dt} = -c_1 v_x \tag{2.36}$$

$$m\frac{dv_y}{dt} = -mg - c_1 v_y \tag{2.37}$$

The flight time is the sum of ascent time T_{asc} and descent time T_{dsc}. An integration gives the following equation for the ascent time:

$$\int_0^{v_0 \sin\theta_0} \frac{dv_y}{g + \frac{c_1}{m} v_y} = \int_0^{T_{asc}} dt = T_{asc} \tag{2.38}$$

The ascent time simply turns out to be:

$$T_{asc} = \frac{m}{c_1} ln(1 + \frac{c_1}{mg} v_0 \sin\theta_0) \tag{2.39}$$

Note in the limit of $c_1 \to 0$ (no air resistance) a Taylor expansion of the logarithm function leads to $T_{asc} = \frac{v_0 \sin\theta_0}{g}$ as expected. To find

the maximum height H we should integrate from $v_y(t)$. An integration of (2.37) gives:

$$v_y(t) = (v_0 \sin\theta_0 + \frac{mg}{c_1})e^{-\frac{c_1}{m}t} - \frac{mg}{c_1} \tag{2.40}$$

Another integration from (2.40) gives:

$$y(t) = \frac{m}{c_1}(v_0 \sin\theta_0 + \frac{mg}{c_1})(1 - e^{-\frac{c_1}{m}t}) - \frac{mg}{c_1}t \tag{2.41}$$

To find H we should substitute t with T_{asc} in (2.41). A bit of algebra gives:

$$H = \frac{m}{c_1}(v_0 \sin\theta_0 + \frac{mg}{c_1})[1 - (1 + \frac{c_1}{mg}v_0 \sin\theta_0)^{-1}] -$$

$$\frac{m^2 g}{c_1^2} ln(1 + \frac{c_1}{mg}v_0 \sin\theta_0) \tag{2.42}$$

To find the range R we need to find the descent time T_{dsc}. However, according to (2.13) by replacing h with H we arrive at a transcendental equation for descending time that cannot be solved analytically. To find the range R we note that $R = x(T) = x(T_{asc} + T_{dsc})$. From (2.36) we have:

$$x(t) = \frac{mv_0 \cos\theta_0}{c_1}(1 - e^{-\frac{c_1}{m}t}) \tag{2.43}$$

2.4.2 Quadratic air drag force

Now let us see how analytically we can proceed if the air resistance depends quadratically on the projectile's velocity. The vectorial equation of motion becomes (y axis upward):

$$m\frac{d\mathbf{v}}{dt} = -mg\mathbf{j} - c_2 v^2 \hat{v} = -mg\mathbf{j} - c_2 v\mathbf{v} \tag{2.44}$$

where $v = \sqrt{v_x^2 + v_y^2}$ is the projectile's velocity magnitude. In terms of components, we obtain the following set of coupled differential equations:

$$m\frac{dv_x}{dt} = -c_2 v v_x \tag{2.45}$$

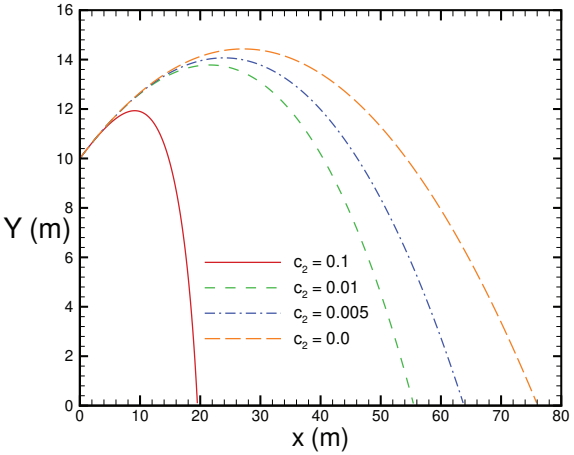

Figure 2.8: *Cannonball trajectory computed by the Euler algorithm for three values of c_2. Comparison is made with no air resistance situation. The simulation parameters are $h = 10\ m, v_0 = 30\ m/s, \theta_0 = 45^o$ and $\tau = 0.01$.*

$$m\frac{dv_y}{dt} = -mg - c_2 v v_y \tag{2.46}$$

Unfortunately, equations (2.45) and (2.46) are coupled and above that non-linear due to the term $v = \sqrt{v_x^2 + v_y^2}$. To the best of my knowledge, no analytical solution has been found up to now and we have to resort to numerical analysis. The programme `2dProjectile` (see Appendix 2.D for more details) simulates the two-dimensional motion of a projectile in the presence of air resistance. Figure (2.8) shows the computed cannonball trajectory of unit mass ($m = 1$) in the presence of quadratic air resistance drag force $F(v) = c_2 v^2$ for three values of drag coefficient c_2. Initial conditions are $h = 10\ m$, $\theta_0 = 45^o$ and $v_0 = 30\ m/s$. The Euler algorithm with $\tau = 0.01$ has been used. We are interested to know how the quantities: range, total flight time, and the maximum height H will vary when there is air resistance. In figure (2.9) we show the variation of maximum height H versus firing angle θ_0 for some values of c_2. Figure (2.10) shows the dependence of range R on θ_0. As expected by increasing the air resistance the range

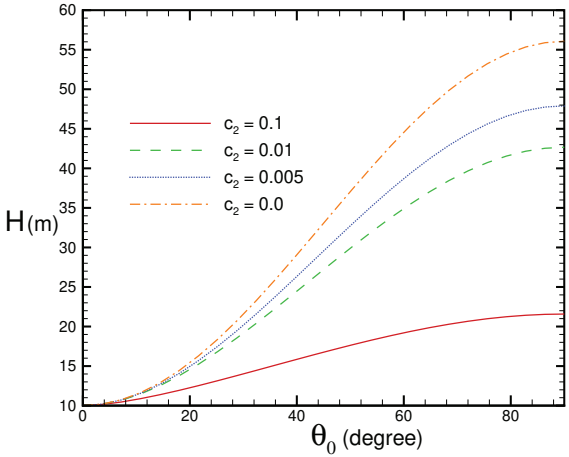

Figure 2.9: *The maximum height H computed by the Euler algorithm for three values of c_2. Comparison is made with no air resistance situation. The simulation parameters are $h = 10\ m, v_0 = 30\ m/s$, and $\tau = 0.01$.*

R decreases. We also notice that by increasing c_2 the angle at which the range is maximised will decrease. Now let us investigate how the motion characteristics vary when h is changed. Figure (2.11) shows the dependence of range R on h for fixed θ_0, v_0 and c_2. As our final example, we consider the following problem:

Consider the motion of two identical objects that both start their motion from height h. One object is dropped vertically from rest and the other is thrown with a horizontal velocity v_0. Which object reaches the ground first? Discuss three cases: no air resistance, air resistance with a linear dependence on velocity magnitude, and air resistance with quadratic dependence on velocity magnitude.

In the case of no drag force, both objects reach the ground simultaneously. The reason is that equation of motion for v_y does not depend on v_x for the object thrown horizontally. Equations of motion for v_x and v_y remain decoupled when we have a linear drag force. We recall that the equation of motion for the vertical velocity v_y is (2.37) and

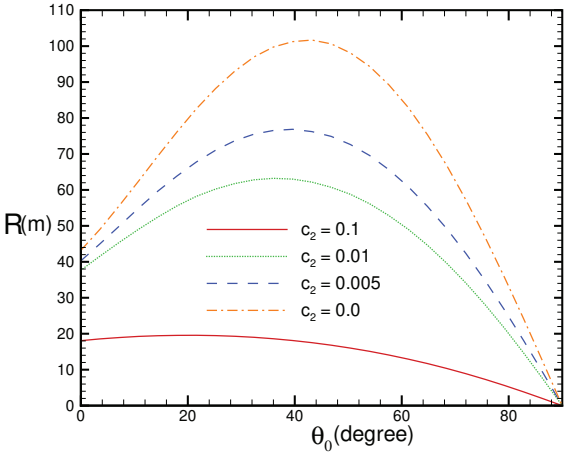

Figure 2.10: *The projectile range R computed by the Euler algorithm for three values of c_2. Comparison is made with no air resistance situation. The simulation parameters are $h = 10\ m, v_0 = 30\ m/s$, and $\tau = 0.01$.*

since this equation does not depend on v_x the initial value $v_x(0)$ does not affect it. However, for the third case i.e.; quadratic drag force the equations of motion for velocity components are coupled and the initial velocity $v_x(0)$ affects the problem. Figure (2.12) shows the falling time difference $\Delta T = T_2 - T_1$ versus h. Note T_1 corresponds to initial zero velocity whereas T_2 refers to non-zero initial horizontal velocity. We see that for large h the falling time difference ΔT does not depend on h. This is because for large h both objects reach the terminal velocity and the horizontally thrown object velocity becomes vertical. Consequently, they will have the same speed. The difference between falling time is associated with the early stages of the flight before the objects reach the terminal velocity. Before closing the chapter, we remind you that the Euler algorithm did a very good performance in dealing with second-order ordinary linear differential equations involving the free fall motion under the gravitational field. In the next chapter, we will encounter oscillatory motion and examine if the Euler algorithm can successfully be applied to this sort of motion.

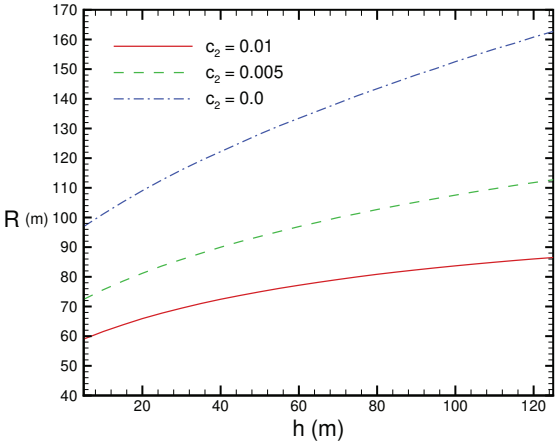

Figure 2.11: *The range R computed by the Euler algorithm versus height h for two values of c_2. Comparison is made with no air resistance situation. The simulation parameters are $\theta_0 = 45^o, v_0 = 30\ m/s$ and $\tau = 0.01$.*

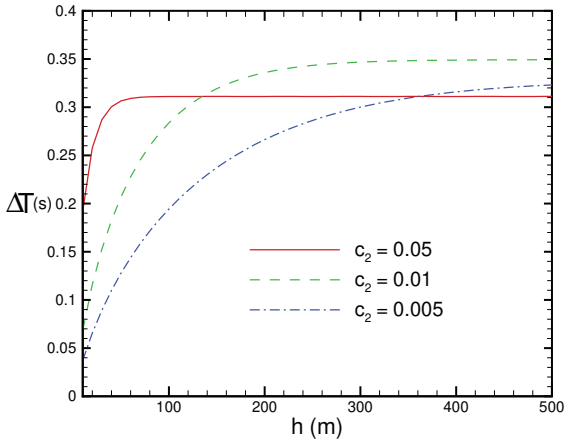

Figure 2.12: *The computed falling time difference ΔT by the Euler algorithm versus height h for two values of c_2. The simulation parameters are $v_{x0} = 30\ m/s$ and $\tau = 0.01$.*

2.5 Problems

Problem 2.1 Suppose a bicyclist of mass $m = 60\ kg$ generates a constant power P to ride her bicycle.

(a) Ignoring the damping effects caused by earth and air, obtain the velocity time dependence both analytically and numerically (see chapter two of (Giordano and Nakanishi, 2006) for further details).

(b) In the previous part, the velocity grows indefinitely. To circumvent this unphysical problem a source of dissipation should be included. The main dissipation source comes from the air resistance of the bicyclist. As a first approximation to the air drag force, consider the conventional form $-B_1 v$ as the energy loss term and solve the problem numerically by implementing a proper term in the velocity update rule and obtain the velocity as a function of time. Take the generated power $P = 200\ Watts$. What should approximately be the air drag constant B_1 to have a constant velocity of 30 $km/hour$? We remark that the linear drag is known as the Stocks law in hydrodynamics. The coefficient B_1 strongly depends on the object's shape. For a sphere, it turns out that $B_1 = 6\pi\eta R$ where η is the shear viscosity and R is the object radius. For a derivation of this relation, you may consult a standard fluids mechanics textbook.

(c) Noting that the air shear viscosity is $\eta = 2 \times 10^{-5}$ Poise(Pa.s), estimate the rider frontal area (effective radius).

Problem 2.2 In problem one, replace the linear air drag with a quadratic one force $-B_2 v^2$. As stated, the linear drag coefficient B_1 depends on the air viscosity η and the frontal area A of the bicycle rider in contact with air. An estimation of the quadratic drag coefficient B_2 turns out to be $B_2 = \frac{1}{2} C\rho A$ in which ρ is the fluid (here air) density and C a drag coefficient. For a qualitative derivation see chapter two of (Giordano and Nakanishi, 2006).

(a) Write down the Newton equation of motion and obtain the terminal velocity by equalising the input power generated by the rider to the dissipated one into the air.

(b) Try to solve the problem analytically.

(c) Obtain the velocity updating scheme which now includes only the quadratic air drag force (take $C = 1$).

(d) Numerically solve the problem and obtain the velocity-time curve for the rider. Take $\rho_{air} = 1.2kg/m^3$ and $A = 0.33\ m^2$.

Problem 2.3 Add a linear air drag term to the quadratic one and numerically solve the problem. Compare your numerical solution with problem two.

Problem 2.4 In our 2D projectile motion we took the coefficient of quadratic air drag force to be constant c_2. We saw in problem two that this coefficient depends linearly on the air density ρ. However, the air density changes with altitude. Incorporating this fact into account gives rise to considerable changes in the projectile motion. To investigate this effect quantitatively we need to know how air density depends on altitude. In the simplest approach, we take the air to be an isothermal ideal gas. The ideal gas equation of state is $p = \rho\frac{RT}{M}$ where M is the air molar mass. Next, we employ the Euler equation for the airflow which is the simplified version of the Navier-Stocks fluid dynamics equation. The Euler equation is $\rho\frac{\partial\vec{v}(z,t)}{\partial t} = -\nabla p(z,t) + \rho g$. In the steady state we have $\frac{\partial\vec{v}(z,t)}{\partial t} = 0$ and we get $\frac{dp}{dz} = -\rho g$. If we replace p from the equation of state into this differential equation we get a differential equation for pressure ρ:

$$\frac{d\rho}{dz} = -\frac{gM}{RT}\rho$$

. The solution of this first-order differential equation is $\rho(z) = \rho_0 e^{-\frac{gMz}{RT}}$. This is the well-known barometric formula. ρ_0 is the air density at sea level. Now take $c_2 = ce^{-\frac{gMz}{RT}}$ in the quadratic air drag force and numerically solve the projectile problem. Which quantity undergoes a notable change? Obtain the dependence of the maximum height attained by the projectile on the initial velocity.

Problem 2.5 We know that air temperature varies with height therefore the isothermal approximation is not realistic. Another approach is to take the atmosphere to be a poor conductor. Moreover, we assume that the connectivity is poor too. This leads to the so-called *adiabatic* approximation.

(a) By solving the linearised Navier-Stock (Euler) equation show that the density becomes:

$$\rho(z) = \frac{M}{RT} p_0 [1 - \frac{\gamma - 1}{\gamma} \frac{g \rho_0 z}{p_0}]^{\frac{\gamma - 1}{\gamma}}$$

(b) Numerically solve the projectile problem when the atmosphere is taken as an adiabatic ideal gas and compare your results with those in problem four. Take the initial condition the same.

Problem 2.6 Solve the motion of a 2D projectile in the air when the air temperature linearly decreases with altitude i.e.; $T(z) = T_0(1 - \frac{z}{H})$. Compare your findings with previous atmospheric models.

Problem 2.7 Solve the 2D projectile motion in a non-inertial reference frame. In reality, the Earth is a rotating system around the Sun and we have to take into account non-inertial forces such as centrifugal and Coriolis. The true equation of motion turns out to be (Thornton and Marion, 2003):

$$m \frac{d^2 \vec{r}}{dt^2} = \mathbf{F} - m\ddot{\mathbf{R}} - m\dot{\vec{\omega}} \times \vec{r} - m\vec{\omega} \times (\vec{\omega} \times \vec{r}) - 2m\vec{\omega} \times \vec{v}$$

F is the total sum of forces acting on the particle as measured in the inertial reference frame, the second and third terms arise from the translational and rotational accelerations of the moving frame relative to the fixed inertial one (here the Sun system). The third one is called centrifugal and the last term is the so-called *Coriolis* force. Retaining only the Coriolis force in the projectile problem and ignoring the air resistance,

(a) Sketch the projectile range in terms of initial velocity and compare your results to the case where the Coriolis force is absent.

(b) Include the air resistance and Coriolis force into the motion equation and repeat part (a). Take $\omega = 2\pi/year$.

Problem 2.8 **Magnus effect**: An interesting observation in sports is the curve of balls in flight due to their rotation. The curvature in a spinning ball trajectory is a purely hydrodynamical effect and was first investigated in 1850 by G. Magnus and is now known as the Magnus

effect. It can be explained qualitatively by observing that the speed of the ball's surface relative to the air is different on opposite edges of the ball. Suppose a ball with radius r is moving in x direction with a cnetre of mass speed v. The ball is spinning around the $-y$ axis with a constant angular velocity ω. It turns out that the velocity of the ball's top surface is $v - \omega r$ whereas its bottom surface velocity is $v + \omega r$. If we assume that the air drag is quadratic in velocity, it can be shown that there is a net outward force (in $(+z)$ direction) on the bottom surface of the ball which is proportional to the relative velocity $(v + \omega r)^2$ to air. Similarly, the ball experiences a net downward force (in $(-z)$ direction) on its top surface which is proportional to the relative velocity $(v - \omega r)^2$ to air. The net force, the Magnus force, is the algebraic sum of these forces: $F_M \propto (v+\omega r)^2 - (v - \omega r)^2 \sim vrm$. Showing the proportionality coefficient by C_M the Magnus force on a spinning object can be more generally written as $F_M = C_M \vec{\omega} \times \mathbf{v}$. We now wish to solve a sports problem in football. All of us have seen spectacular goals scored from a fixed kick. Let us try to find the ball trajectory when Magnus's force of incorporated. Assume the ball is located 25 m away for the goal perpendicular to the goal line. Assume a player gives an initial velocity $v_0 = 60\ km/h$ in such a way that the ball takes off the ground at angle $\theta_0 = 20$ degrees. what should be the initial angular velocity provided by the kicker so that the ball arrives at the top left corner of the goal? Take the goal post height $H = 2.44\ m$, its width $L = 7.32\ m$ and $\frac{C_M}{m} = 0.02$.

Problem 2.9 **Motion of golf ball**: I invite those of you who are interested in golf play to solve this problem. Hopefully, we have all gradients to attack this problem. Suppose both a drag force F_d and the Magnus force are exerted on a gold ball of radius R and mass m. Let y denote the ball's height to the ground and x the horizontal direction along the initial velocity v_0. The equation motions will be:

$$\frac{dv_x}{dt} = -\frac{F_{d,x}}{m} - \frac{C_M \omega v_y}{m}$$

$$\frac{dv_y}{dt} = -\frac{F_{d,y}}{m} + \frac{C_M \omega v_x}{m} - g$$

Note that the ball is hit with a backspin (spin axis along z) with a constant angular velocity ω. Take the air drag as $\mathbf{F}_d = -C\rho A v^2 \hat{v}$ with a velocity-dependent C. At low speeds up to 14 m/s take $C = \frac{1}{2}$. At

higher velocities take $C(v) = \frac{7}{v}$. You can see (Giordano and Nakanishi, 2006) for a detailed discussion. We take $\frac{C_M \omega}{m} = 0.25\ s^{-1}$.

(a) Numerically solve the equations of motion and plot the ball trajectory in the $y - x$ plane.

(b) Take the drag coefficient C to be constant and re-plot the trajectory. Do you see any significant difference?

(c) Suppose a wind is blowing in the z direction with constant velocity 15 m/s. Plot the trajectory in both the $y - x$ and $z - x$ planes.

Problem 2.10 Motion of Tennis ball: Try to model the motion of a Tennis ball and numerically solve it. You may consult the paper *The aerodynamics of Tennis balls* by *Štěpánek* in American journal of physics (Stepanek, 1988).

Chapter 3

Oscillatory motion

3.1 Simple Harmonic oscillator

Many systems in science and especially in physics undergo regular oscillatory motions. The evolution of planets around the Sun and pendulum motion are among the most ubiquitous ones we observe in our everyday life. Less obvious examples are microscopic phenomena such as the oscillations of atoms in crystalline solids or electron motion in atomic orbitals. In this chapter, we intend to pursue our numerical investigations of these types of motion. Physically speaking, a motion is called oscillatory if an object, or a system of objects, repeats its motion regularly in time along a finite spatial path. You have certainly seen in your undergraduate mechanics' course that every mechanical system in the vicinity of its potential energy minimum performs an oscillatory motion. Let us for simplicity restrict ourselves to one dimension and consider the simplest system i.e.; a point-like particle with mass m which is subjected to a time-independent potential $V(x)$. Suppose the initial conditions are such that the particle is in the vicinity of one of the potential energy minima say x^*. A Taylor expansion of $V(x)$ around x^* gives:

$$V(x) = V(x^*) + (x - x^*)\frac{dV}{dx}(x^*) + \frac{1}{2}(x - x^*)^2\frac{d^2V}{dx^2}(x^*) + \cdots \quad (3.1)$$

With no loss of generality, we can set $V(x^*) = 0$. Since $V(x)$ is minimum at x^* we have $\frac{dV}{dx}(x^*) = 0$ and therefore the Newton equation

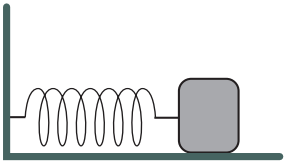

Figure 3.1: *A linear frictionless mass-spring system.*

of motion becomes:

$$m\frac{d^2x}{dt^2} = -\frac{dV}{dx} = -\frac{d^2V}{dx^2}(x^*)(x - x^*) \tag{3.2}$$

Since the potential is minimum at x^* the second derivative of potential is positive. Calling the positive coefficient $\frac{d^2V}{dx^2}(x^*)$ by k and shifting the origin of our new system of coordinates to x^* we find the Newton equation of motion as follows:

$$m\frac{d^2x}{dt^2} = -kx \tag{3.3}$$

where it is understood that now x denotes the deviation of the particle's position from the potential energy minimum. An oscillatory mass-spring system, see figure (3.1), is a prototype example. Equation (3.3) is a linear second-order ordinary differential equation that can simply be solved analytically. In fact, the force on the particle is proportional to its deviation from the equilibrium point: $F(x) = -kx$ which is the simplest form of Hooke's law. Introducing the angular frequency $\omega_0 = \sqrt{\frac{k}{m}}$, the solution of (3.3) turns out to be:

$$x(t) = D\cos(\omega_0 t + \theta_0) \tag{3.4}$$

As you see the particle undergoes a sinusoidal motion which is known as *simple harmonic motion* in the literature. The motion is periodic with the period $T = \frac{2\pi}{\omega_0}$. The amplitude D and the phase θ_0 are given once the initial conditions x_0 and v_0 are specified. According to (3.4) at $t = 0$ we have:

$$x_0 = D\cos\theta_0 \quad ; v_0 = -D\omega_0\sin\theta_0 \tag{3.5}$$

Equation (3.5) simply gives D and θ_0 as follows:

$$\theta_0 = -\tan^{-1}(\frac{v_0}{x_0\omega_0}); \quad D = \sqrt{x_0^2 + \frac{v_0^2}{\omega_0^2}} \tag{3.6}$$

Although the position and velocity of the oscillator are continuously changing, its total mechanical energy E remains constant and is given by:

$$E = \frac{1}{2}mv^2 + \frac{1}{2}kx^2 = \frac{1}{2}kD^2 = \frac{1}{2}k(x_0^2 + \frac{v_0^2}{\omega_0^2}) = \frac{1}{2}mv_0^2 + \frac{1}{2}kx_0^2 \quad (3.7)$$

3.2 Numerical Solution: Euler-Cromer Algorithm

Let us now solve the differential equation (3.3) numerically. As usual, we prefer first to proceed with our simple but favorite Euler algorithm. The recursive equations through which we can iterate the oscillator's velocity and position are:

$$x_{n+1} = x_n + v_n\tau \ ; \ v_{n+1} = v_n + a_n\tau = v_n - \frac{k}{m}x_n\tau \quad (3.8)$$

The programme `Oscillator` (see Appendix 3.A for details) numerically solves the Newton equation of motion for our simple harmonic oscillator. In figure (3.2) we exhibit the time dependence of the oscillator's position for some choices of time step τ. In our system of units, we have set $m = 1$ and $k = 9$ such that ω_0 becomes equal to 3. As you can see our programme fails to reproduce the analytical solution i.e.; simple harmonic sinusoidal motion! The oscillation's amplitude grows over time which is unphysical. This behaviour persists if we go to smaller time steps. As a matter of fact, the failure is not associated with the timestep but is intrinsic to the computational algorithm itself. Up to now, our Euler algorithm has successfully reproduced the analytical solutions but this time it is unable to do so. Unfortunately, the Euler algorithm is not suitable for oscillatory motions. To remedy the problem, we have to resort to more advanced algorithms. The simplest one which can give us a satisfactory answer is the *Euler-Cromer* algorithm (Cromer, 1981). According to this algorithm, we should first update the velocity and then use the updated velocity for position updating. Precisely speaking, the algorithm is implemented for a general force as follows:

$$v_{n+1} = v_n + a_n\tau + o(\tau)^2 \ \ x_{n+1} = x_n + v_{n+1}\tau + o(\tau)^2 \quad (3.9)$$

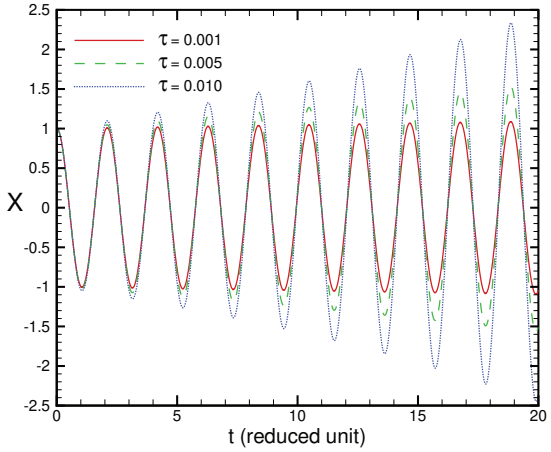

Figure 3.2: *Time evolution of the harmonic oscillator's position for* $T = 20$ *units of time. The Euler algorithm is used.*

Note the only difference to the Euler algorithm is the replacement of v_n by v_{n+1} in the position updating. For our particular case of linear restoring force, we have:

$$v_{n+1} = v_n + a_n\tau = v_n - \frac{k}{m}x_n\tau;\ x_{n+1} = x_n + v_{n+1}\tau \qquad (3.10)$$

Despite this simple change, the result is amazing. Figure (3.3) depicts the position-time curve for the same values of time step τ. You can see that the problem has been nicely cured and we get a reasonable answer. Despite there is no meaningful difference between the three choices of τ in figure (3.3), the difference becomes more prominent if we look at the variation of computed total energy in time. Let us denote the oscillator energy at time step n by E_n and the difference between the computed energy and its exact value by $\Delta E_n = E_n - E$. Figure (3.4) plots the dependence of energy difference ΔE_n as a function of n: The Euler-Cromer algorithm preserves the energy in an oscillatory fashion. By decreasing the timestep τ, we reduce the computational error in total energy. The oscillatory character of ΔE is associated with the restoring force. Let us introduce some more advanced algorithms.

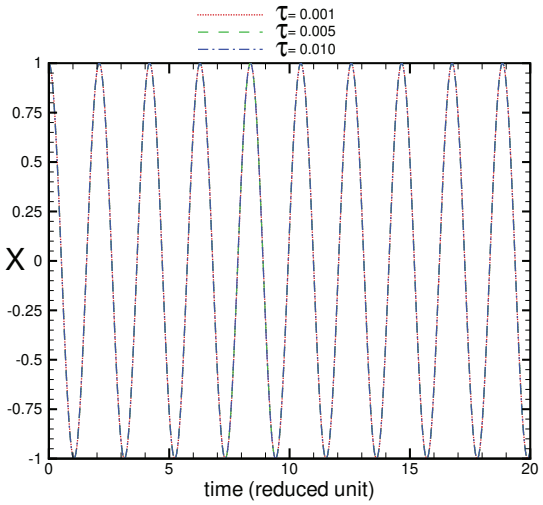

Figure 3.3: *Time evolution of the harmonic oscillator position for $T = 20$ units of time. The Euler-Cromer algorithm is used.*

3.3 Other algorithms

3.3.1 Mid-Point Algorithm

The third algorithm we introduce is the *mid-point* algorithm. Analogous to the Euler-Cromer, in this algorithm the velocity is updated first. However, we use the mean velocity during the timestep interval to obtain the new position. The midpoint algorithm can be written as:

$$v_{n+1} = v_n + a_n\tau + o(\tau)^2;\ x_{n+1} = x_n + \frac{1}{2}(v_n + v_{n+1})\tau + o(\tau)^2 \quad (3.11)$$

Substitution of v_{n+1} in the right-hand side of the position update we find:

$$x_{n+1} = x_n + v_n\tau + \frac{1}{2}a_n\tau^2 + o(\tau)^3 \quad (3.12)$$

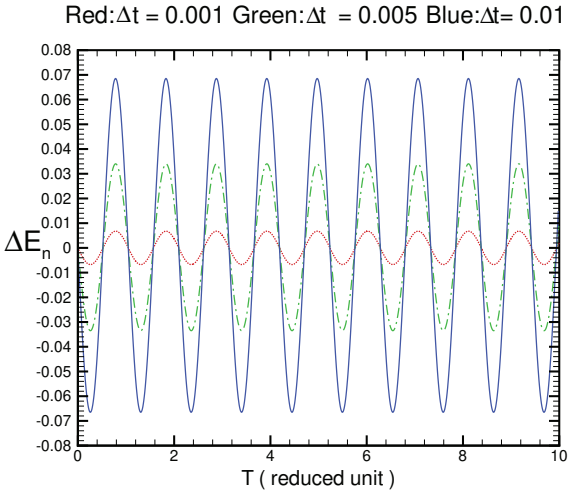

Figure 3.4: *Energy difference ΔE_n as a function of n for various values of time step τ. The Euler-Cromer algorithm is used.*

3.3.2 Euler-Richardson Algorithm

We leave the derivation of this algorithm and only quote the result here. Interested readers can find the derivation in chapter 3 of (H. Gould and Chriastian, 2006). Technically, this algorithm exploits the half-step concept. We first define the velocity and acceleration values at half steps via first-order Taylor expansions as follows:

$$v_{n+\frac{1}{2}} = v(t_n + \frac{\tau}{2}) = v_n + \frac{\tau}{2}a_n + o(\tau)^2 \tag{3.13}$$

$$a_{n+\frac{1}{2}} = a(t_n + \frac{\tau}{2}) = a_n + \frac{\tau}{2}a'_n + o(\tau)^2 \tag{3.14}$$

We now express the update rules:

$$x_{n+1} = x_n + v_{n+\frac{1}{2}}\tau + o(\tau)^3 \tag{3.15}$$

$$v_{n+1} = v_n + a_{n+\frac{1}{2}}\tau + o(\tau)^3 \tag{3.16}$$

3.3.3 Verlet Algorithm

One of the most common and efficient algorithms in computational physics is attributed to Verlet (the letter "t" should not be pronounced). To obtain this algorithm, we write the Taylor expansions of x_{n-1} and x_{n+1} up to the third order in τ as follows:

$$x_{n\pm1} = x_n \pm v_n\tau + \frac{1}{2!}a_n\tau^2 \pm \frac{1}{3!}a'_n\tau^3 + o(\tau)^4 \tag{3.17}$$

Adding the corresponding Taylor expansions gives:

$$x_{n+1} + x_{n-1} = 2x_n + a_n\tau^2 + o(\tau)^4 \tag{3.18}$$

In other words, we get:

$$x_{n+1} = 2x_n - x_{n-1} + a_n\tau^2 + o(\tau)^4 \tag{3.19}$$

Equation (3.19) is known as the Verlet algorithm. It is also known as *Leapfrog* algorithm. Note that its local truncation error is small (fourth order). If the force does not depend on velocity one can update the position solely from (3.19) irrespective of the velocity update because a_n can only depend on t_n and x_n. This makes the algorithm very fast which is very desirable. Nevertheless, it is possible to obtain the updated velocity if it is required. To this end, we Subtract x_{n-1} from x_{n+1} and find:

$$v_n = \frac{x_{n+1} - x_{n-1}}{2\tau} + o(\tau)^2 \tag{3.20}$$

Clearly, the velocity update error is second-order and is not as precise as the position update. The notable point is that the Verlet algorithm is not *self starting*. As you can see from (3.19) to find the new position x_{n+1} we need to know not only the current position x_n but also the previous position x_{n-1}. We call these types of algorithms as *two steps*. Being a two-step algorithm causes a drastic problem in the beginning! According to (3.19) we have $x_1 = 2x_0 - x_{-1} + a_0\tau^2$ but x_{-1} is unknown to us. To circumvent this problem people normally obtain x_1 via another algorithm. For example, we can find x_1 from the Euler algorithm: $x_1 = x_0 + v_0\tau$. Once x_1 is obtained we can iterate (3.19) and update the positions in time.

3.3.4 Velocity Verlet Algorithm

Another version of the Verlet algorithm which is most frequently used in computational physics and in particular in molecular dynamics is the so-called *Velocity Verlet*. To obtain this algorithm we note that according to (3.20) and up to the first order in τ we have $x_{n-1} = x_{n+1} - 2\tau v_n$. If we replace this x_{n-1} into (3.18) and solve for x_{n+1} we simply get:

$$x_{n+1} = x_n + v_n\tau + \frac{1}{2}a_n\tau^2 + o(\tau)^4 \tag{3.21}$$

To obtain the velocity update we note that (3.20) gives:

$$v_{n+1} = \frac{x_{n+2} - x_n}{2\tau} + o(\tau)^2 \tag{3.22}$$

According to (3.19) we have:

$$x_{n+2} = 2x_{n+1} - x_n + a_{n+1}\tau^2 + o(\tau)^4 \tag{3.23}$$

If we replace x_{n+2} from (3.23) into (3.22) we get:

$$v_{n+1} = \frac{x_{n+1} - x_n}{\tau} + \frac{1}{2}a_{n+1}\tau + o(\tau)^2 \tag{3.24}$$

The final stage is to replace x_{n+1} in (3.24) from (3.21) which yields:

$$v_{n+1} = v_n + \frac{1}{2}(a_n + a_{n+1})\tau + o(\tau)^2 \tag{3.25}$$

Equations (3.21) and (3.25) comprise the velocity Verlet algorithm. Notice that between (3.21) and (3.25) you have to take an intermediate step and evaluate the new acceleration a_{n+1}. This is only possible if the force on the particle does not depend on velocity i.e.; $F = F(\mathbf{r}, t)$. If this is the case one can simply obtain a_{n+1} from the updated position x_{n+1}. Now that we have learnt some more algorithms it would be interesting to see how these algorithms are compared to each other when energy conservation is considered. Figure (3.5) plots the time series of the energy difference to the exact value for the Euler-Richardson and Verlet algorithms. Note we have already obtained this quantity for the Euler-Cromer algorithm in figure (3.4). As you see in figure (3.5) the result of ΔE_n in the Euler-Richardson algorithm for $\Delta t = 0.001$ is too tiny (of order 10^{-7}) to be observed in the graph.

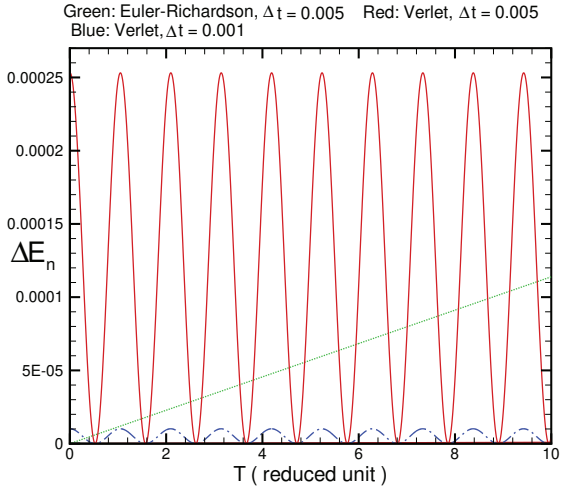

Figure 3.5: *Comparison of the energy difference ΔE_n time series for the Euler-Richardson and Verlet algorithms.*

The Euler-Richardson algorithm gives a better result for a fixed Δt but suffers from the fact that ΔE_n increases with n. On the other hand, the benefit of the Verlet algorithm is that alike Euler-Cromer ΔE_n does not grow monotonously with n but rather oscillates around a fixed value.

3.4 Lissajous figures

Before pursuing our numerical studies on one-dimensional oscillatory motion, let us show you some interesting figures arising from two-dimensional oscillatory motion. These figures are called *Lissajous* figures in honour of French physicist, mathematician, and musician Jules Antoine Lissajous (1822-1880). Among other innovations, Lissajous invented the Lissajous apparatus, an optical-based device that created the figures that bear his name. In 1855 Lissajous invented a set of tuning forks that were part of an apparatus for the visualization and analysis of sound vibrations. This particular set was manufactured by the Parisian scientific instrument maker Rudolph Koenig for

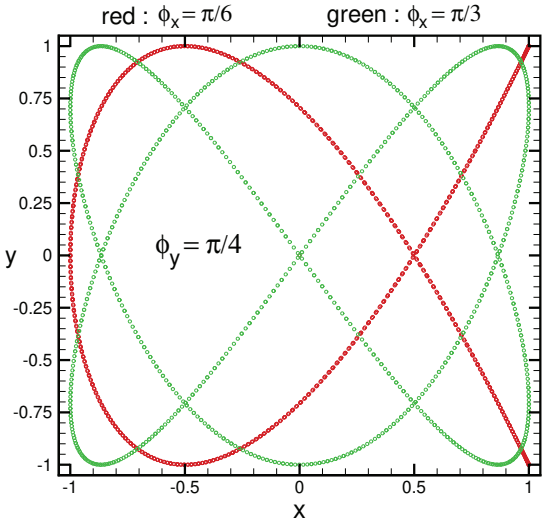

Figure 3.6: *Lissajous figures for two values of ϕ_x.*

educational demonstrations and also for the fine calibration of new tuning forks. You can see the apparatus in the Whipple Museum of Science in Cambridge. Imagine that the vertical and horizontal inputs to an oscilloscope are sinusoidal in time, that is, $x = A_x(\sin\omega_x t + \phi_x)$ and $y = A_y \sin(\omega_y t + \phi_y)$. If the curve that is drawn repeats itself it is called a Lissajous figure. Equivalently x and y can be regarded as the coordinates of a particle restricted to moving in the $x-y$ plane. Figure (3.6) shows the trajectory in the $x-y$ plane for $A_x = A_y = 1, \phi_y = \frac{\pi}{4}$ and two values of $\phi_x = \frac{\pi}{3}, \frac{\pi}{6}$. As you can see, there is a dramatic dependence on the phases. The effect of varying the amplitudes A_x and A_y is only to rescale the figure. Figure (3.7) illustrates this point for $A_x = 2$ and $A_y = 3$. Phases are $\phi_x = \frac{\pi}{6}$ and $\phi_y = \frac{\pi}{4}$ correspondingly. In figure (3.8) we change ω_x to 2.33, the other parameters are : $\phi_x = \frac{\pi}{6}$ and $\phi_y = \frac{\pi}{4}$. You see a notable change in the $x-y$ diagram. Theoretically, we know that the $x-y$ curve is closed when the ratio between ω_x and ω_y is the form of $\frac{a}{b}$ where a and b are integer numbers. In figure (3.8) we see that the curve is more space-filling although it is actually a closed curve.

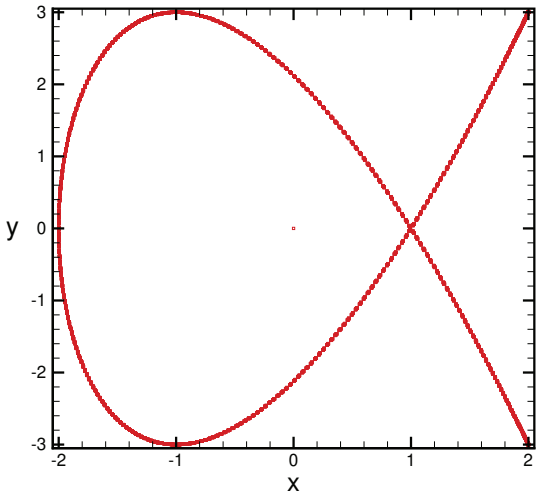

Figure 3.7: *Re scaling of Lissajous figures by changing amplitudes:* $A_x = 2$ *and* $A_y = 3$ *with phases:* $\phi_x = \frac{\pi}{6}$ *and* $\phi_y = \frac{\pi}{4}$.

3.5 Damped harmonic oscillator

From our everyday experience, we know that most of the oscillatory motions in nature gradually attenuate and diminish such that the displacement becomes zero gradually unless energy is injected into the system. Such a kind of motion is said to be damped and the system is said to be dissipative rather than conservative. In mechanical motions such as a mass-spring system or pendulum, the underlying dissipation mechanism is associated with frictional drag force caused by the ground or air. For small velocities, it is a reasonable approximation to assume that the drag force is proportional to the first power of the velocity. Focusing our attention first on a mass-spring system the equation of motion can be written as:

$$m\frac{d^2x}{dt^2} = -kx - \gamma v \tag{3.26}$$

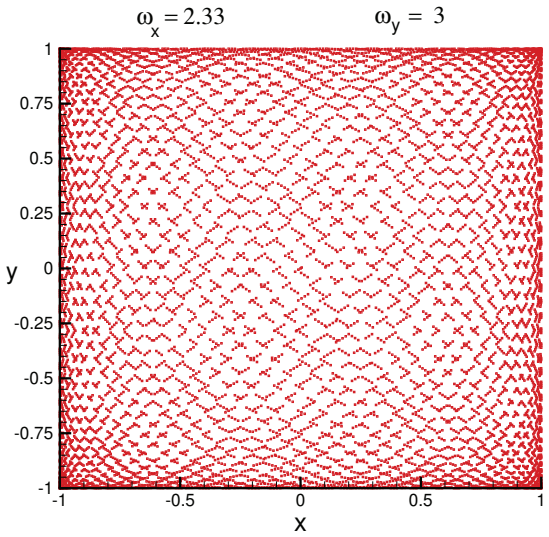

Figure 3.8: *A Lissajous figure for $\phi_x = 2.33$ and $\phi_x = 3$.*

where γ is the damping coefficient. Dividing both sides of (3.26) by m and introducing $\beta = \frac{\gamma}{2m}$ and $\omega_0 = \sqrt{\frac{k}{m}}$ we arrive at the following equation of motion:

$$\frac{d^2x}{dt^2} = -\omega_0^2 x - 2\beta v \tag{3.27}$$

Note ω_0 is the characteristic angular frequency in the absence of damping. Equation (3.27) is a homogeneous second-order linear differential equation. It can be exactly solved once the initial conditions x_0 and v_0 are given. We know from undergraduate mechanics e.g.; in (Thornton and Marion, 2003) that three types of a solution emerge: underdamped when $\omega_0 > \beta$, critically damped when $\omega_0 = \beta$ and overdamped when $\omega_0 < \beta$. In the underdamped case, the solution turns out to be:

$$x(t) = De^{-\beta t}\cos(\omega_1 t + \delta) \tag{3.28}$$

with $\omega_1 = \sqrt{\omega_0^2 - \beta^2}$. Note that $\omega_1 < \omega_0$. The constants D and δ are determined by the initial conditions. From (3.28) we have:

$$x_0 = D\cos\delta; \quad v_0 = -D(\beta\cos\delta + \omega_1\sin\delta) \tag{3.29}$$

dividing v_0 by x_0 we find:

$$\frac{v_0}{x_0} = -\beta - \omega_1\tan\delta \tag{3.30}$$

which gives δ, as follows, in terms of initial conditions:

$$\delta = -\tan^{-1}[\frac{\frac{v_0}{x_0} + \beta}{\omega_1}] \tag{3.31}$$

As expected in the limit of no damping $\beta \to 0$ equation (3.31) becomes identical to (3.6). A bit algebra gives $D = \sqrt{x_0^2 + (\frac{v_0+\beta x_0}{\omega_1})^2}$ which approaches (3.6) in the limit $\beta \to 0$. In the critically damped case, the solution becomes:

$$x(t) = (A + Bt)e^{-\beta t} \tag{3.32}$$

Note there is no oscillation in this case. Constants A and B are determined by the initial conditions. One simply finds:

$$A = x_0; \quad B = v_0 + \beta x_0 \tag{3.33}$$

In the overdamped case ($\omega_0 < \beta$) we have:

$$x(t) = e^{-\beta t}(A_1 e^{\omega_2 t} + A_2 e^{-\omega_2 t}) \tag{3.34}$$

A_1 and A_2 are given by the initial conditions and $\omega_2 = \sqrt{\beta^2 - \omega_0^2}$. At $t = 0$ we have $x_0 = A_1 + A_2$ and $v_0 = -\beta x_0 + \omega_2(A_1 - A_2)$. These equations can simply be solved and we find:

$$A_1 = \frac{x_0(\omega_2 + \beta) + v_0}{2\omega_2}; \quad A_2 = \frac{x_0(\omega_2 - \beta) - v_0}{2\omega_2} \tag{3.35}$$

We now try to simulate the damped motion of a linear one-dimensional oscillator. As we showed, the Euler algorithm fails to reproduce the analytical solution. We incorporate both the Euler-Cromer (EC) and Euler-Richardson (ER) algorithms. Furthermore, we set our parameters as follows: $m = 1, k = 9$ ($\omega_0 = 3$) and $\gamma = 0.5$. The initial

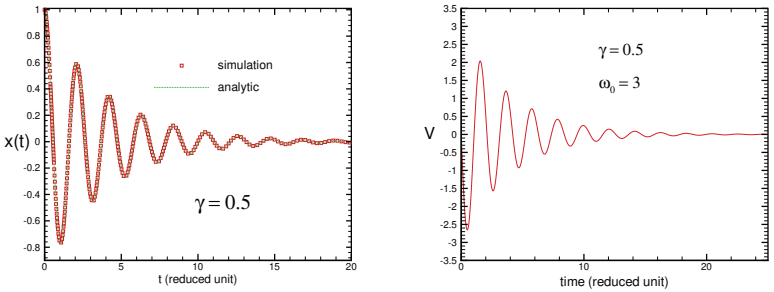

Figure 3.9: *Position (left) and velocity (right) versus time for a damped harmonic oscillator with* $\omega_0 = 3$ *and* $\gamma = 0.5$. *For position, a comparison is made with the analytical solution. The simulation data are obtained by the Euler-Cromer algorithm with a timestep* $\tau = 0.01$.

conditions were chosen such that $x_0 = 1$ and $v_0 = 0$. The programme `DampedOscillator` (see Appendix 3.B for details) numerically solves the Newton equation of motion for the damped harmonic oscillator. Figure (3.9) sketches the time evolution of the position and the velocity of a damped harmonic oscillator. As you see, the agreement between the analytical solution and the numerical answer is excellent. Let us compute the period of the motion. For this purpose, in the code, we evaluate the time steps at which x is maximised. The period is defined as the difference between adjacent maxima. For the given parameters, we find $T = 2.1$ which gives $\omega_{comp} = \frac{2\pi}{T} = 2.992$. On the other hand, the analytical solution gives $\omega_1 = \sqrt{\omega_0^2 - \beta^2} = 2.958$. The error in angular frequency is: $\frac{|\omega_1 - \omega_{comp}|}{\omega_1} = 0.011$. The angular frequency in the underdamped case is $\omega_0 = 3$ therefore damping reduces the angular frequency (increases the period) as expected. Note that $\gamma = 0.5$ is a rather large damping coefficient but it slightly affects the angular frequency. In figure (3.10) we exhibit the position and velocity versus time for three values of $\gamma = 1, 2, 3$. Note that the period increases with increasing γ. Furthermore, the amplitude shows a remarkable decrease upon increasing γ. Let us now come to the issue of energy. Due to the damping mechanism, the total energy is not conserved. In the underdamped case the total energy $E = \frac{1}{2}mv^2 + \frac{1}{2}kx^2$

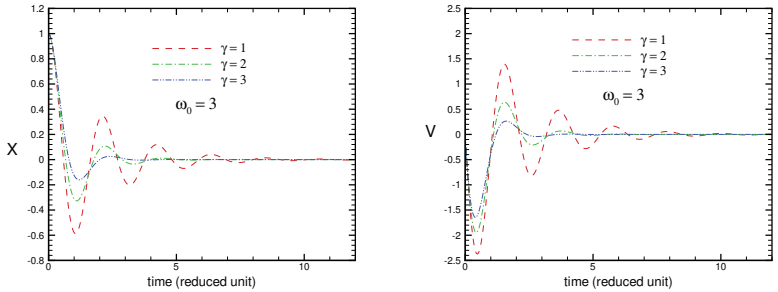

Figure 3.10: *Position (left) and velocity (right) versus time for a damped harmonic oscillator with* $\omega_0 = 3$ *and various values of* $\gamma = 1, 2, 3$. *The simulation data are obtained by the Euler-Cromer algorithm with a timestep* $\tau = 0.01$.

can be obtained after some algebra (using Eq. (3.28)) as follows:

$$E(t) = \frac{D^2}{2} e^{-\frac{\gamma t}{m}} [(k + m\beta^2) cos^2(\omega_1 t + \delta) + m\omega_1^2 sin^2(\omega_1 t + \delta) +$$

$$\gamma\omega_1 sin(\omega_1 t + \delta) cos(\omega_1 t + \delta)] \tag{3.36}$$

As you see it has a complicated trigonometric dependence. Figure (3.11) depicts the dependence of E on time for three values of γ. You see total energy E is an ever-decreasing function of time. Let us now investigate the motion characteristics in the critically damped and overdamped cases. In figure (3.12) we show x and v dependence on time for $\gamma = 5, 6, 7$. Initial conditions are $x_0 = 1, v_0 = 0$. Note that the critical value of the damping coefficient γ_c equals $2m\omega_0$ which in our problem becomes 6. Therefore $\gamma = 5$ is in the underdamped regime whereas $\gamma = 7$ lies in the overdamped regime. We remark that for $\gamma \geq \gamma_c$ there is no oscillation. At a critically damped regime where $\gamma = \gamma_c$ we have the quickest relaxation towards equilibrium as depicted in figure (3.12). For the critically damped case and using (3.32) the total energy E is found to be:

$$E = \frac{1}{2} e^{-\frac{\gamma t}{m}} [(k + m\beta^2)(A + Bt)^2 + mB^2 - 2\beta Bm(A + Bt)] \tag{3.37}$$

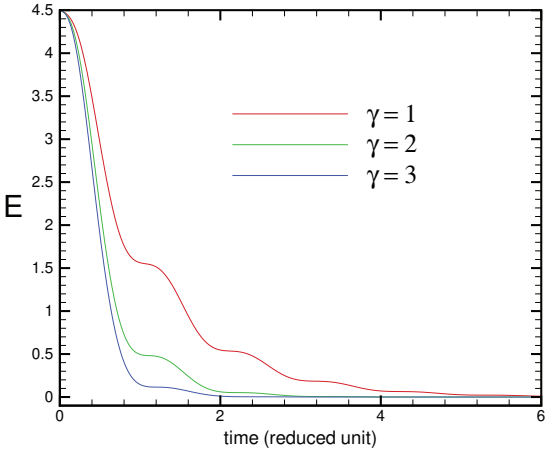

Figure 3.11: *Total energy E versus time for an underdamped harmonic oscillator with $\omega_0 = 3$ and various values of $\gamma = 1, 2, 3$. The simulation data are obtained by the Euler-Cromer algorithm with a timestep $\tau = 0.01$.*

Also for the overdamped case and using (3.34) we find the total energy E, after lengthy but straightforward algebra, as follows:

$$E = \frac{1}{2} e^{\frac{-\gamma t}{m}} \{A_1^2 e^{2\omega_2 t}[k + m(\omega_2^2 + \beta^2 - 2\beta\omega_2)] +$$

$$A_2^2 e^{-2\omega_2 t}[k + m(\omega_2^2 + \beta^2 + 2\beta\omega_2)] + 2A_1 A_2[k - m(\omega_2^2 - \beta^2)]\} \quad (3.38)$$

Figure (3.13) shows the dependence of the damped oscillator's energy E on time for critically damped and overdamped cases. n both cases, the energy decreases over time. As you can see, in the critical damping case, the rate of energy decrease is maximum. The other counterintuitive point is that by increasing the damping coefficient γ the rate of energy decrease does not grow! Of course, this conclusion should be taken with caution. The initial conditions may affect the result. For clarification and a deeper insight onto the problem, we try another initial condition: $x_0 = 0$ but $v_0 = 1$. Figure (3.14) exhibits the energy-time curve for three values of damping coefficient $\gamma = 6, 7, 8$. You see that in this case, the sharpest decrease corresponds now to $\gamma = 8$.

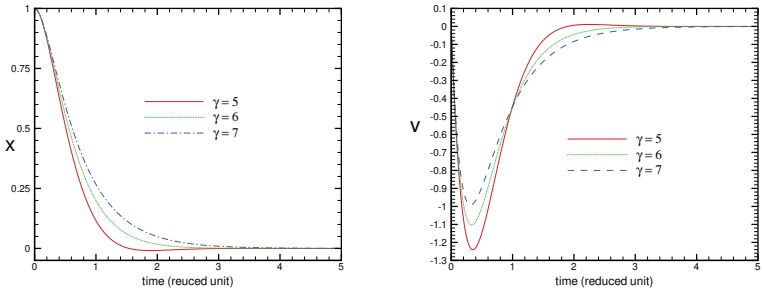

Figure 3.12: *Computed position (left) and velocity (right) versus time for a damped harmonic oscillator with $\omega_0 = 3$ for $\gamma = 5, 6, 7$ corresponding to underdamped, critically damped, and overdamped cases. The simulation data are obtained by the Euler-Cromer algorithm with a timestep $\tau = 0.01$.*

Eventually, we exhibit the two-dimensional phase-space plot for some values of γ in figure (3.15). Note that for $\gamma \geq \gamma_c$ the phase-space curve does not bend around the origin which is due to the non-oscillatory nature of the problem. The origin point $(0, 0)$ in the phase-space is called an *attractor*.

3.6 Driven damped harmonic oscillator

In many practical situations, a physical system is derived by an external force. In this section, we intend to explore the characteristics of this driven motion when the system is a damped harmonic oscillator. Consider a driven damped linear one-dimensional oscillator that is subjected to an external force $F(t)$ in addition to the linear restoring force and a linear damping force. The equation of motion can be written as:

$$m\frac{d^2x}{dt^2} = -kx - \gamma v + F(t) \tag{3.39}$$

For simplicity we assume the driving force $F(t)$ has a sinusoidal dependence on time with frequency ω i.e.; we take $F(t) = F_0 \cos \omega t$. This choice is not too specific of course. In fact, for any type of periodic external force, one can perform the Fourier decomposition of the

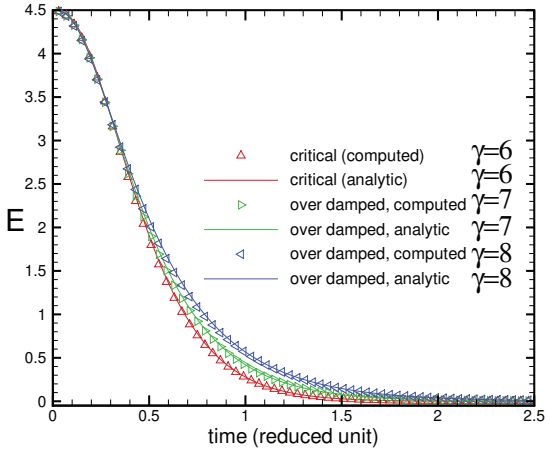

Figure 3.13: *Total energy E versus time for critically damped and overdamped harmonic oscillator with $\omega_0 = 3$. Comparison with the analytical solution is excellent. The simulation data are obtained by the Euler-Cromer algorithm with a timestep $\tau = 0.01$.*

external force. Therefore, if we know the answer to an arbitrary frequency ω we can find the answer by the superposition principle since the differential equation is linear. Let us recall some analytical results from classical mechanics before attempting to solve the problem numerically. The analytical solution of (3.39) is the sum of a particular solution and a complementary one (solution of the corresponding homogeneous equation) (Thornton and Marion, 2003):

$$x(t) = x_c(t) + x_p(t) \tag{3.40}$$

We know the complementary solution from the previous section. The particular solution, the solution of the steady state, turns out to be:

$$x_p(t) = D(\omega)\cos(\omega t - \delta(\omega)) \tag{3.41}$$

$D(\omega)$ and $\delta(\omega)$ are as follows (Thornton and Marion, 2003):

$$D(\omega) = \frac{A}{\sqrt{(\omega^2 - \omega_0^2)^2 + 4\beta^2\omega^2}} \tag{3.42}$$

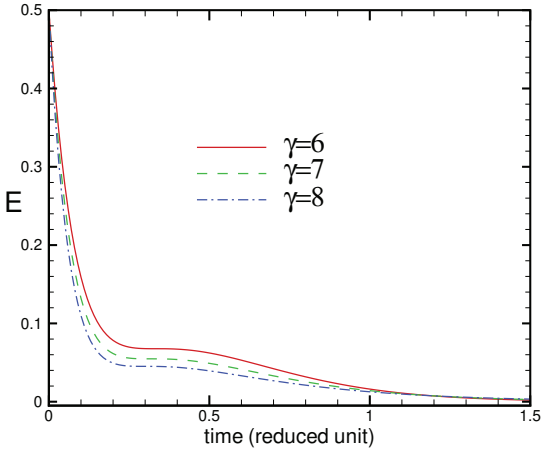

Figure 3.14: *Total energy E versus time for critically damped and overdamped harmonic oscillator with $\omega_0 = 3$. The initial conditions are different from the figure (3.13). Here we have $x_0 = 0$ and $v_0 = 1$. The simulation data are obtained by the Euler-Cromer algorithm with a timestep $\tau = 0.01$.*

$$\delta(\omega) = tan^{-1}(\frac{2\omega\beta}{\omega_0^2 - \omega^2}) \tag{3.43}$$

In which $A = \frac{F_0}{m}$ and $\beta = \frac{\gamma}{2m}$. Note the complementary part of the solution can be in one of the three cases underdamped, overdamped, and critically damped. The constants in the complementary solution $x_c(t)$ are determined by the initial conditions. Let us now solve the problem numerically. The programme `DrivenDampedOscillator` (see Appendix 3.C for details) numerically solves the Newton equation of motion for a driven damped harmonic oscillator. We set the spring constant $k = 9$ and mass $m = 1$ which gives $\omega_0 = \sqrt{\frac{k}{m}} = 3$. Also we take $\gamma = 0.5$ and $\omega = 2$. Figure (3.16) shows the position-time and the velocity-time curves respectively for two different initial conditions. They have been obtained numerically by two methods of Euler-Richardson (ER), and second-order Runge-Kutta (RK2) which will be explained later. Actually, the results of the two algorithms are identical to the eye precision. You see that after a sufficient time

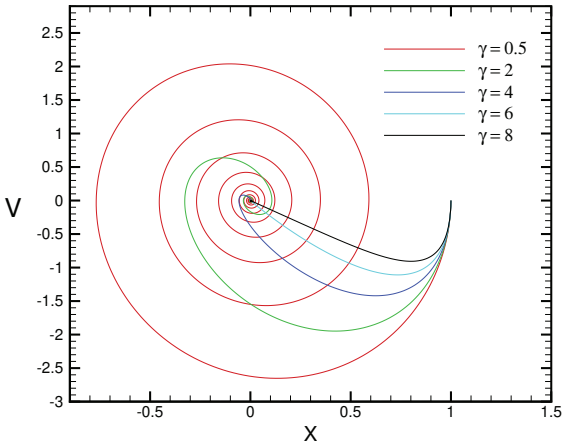

Figure 3.15: *Phase space plot for the damped harmonic oscillator for various values of γ. The simulation data are obtained by the Euler-Cromer algorithm with a timestep $\tau = 0.01$.*

which is 4-5 times the relaxation time $\tau = \frac{1}{\gamma}$, the system reaches a steady state and the effects of initial conditions are smeared out. The bahaviour of the harmonically driven oscillator is entirely different from the non-driven one. Let us explore the effect of varying ω_0 while the driving angular frequency ω is kept fixed. Figure (3.17) depicts the time evolution of the oscillator position for three values of $\omega_0 = 0.2, 0.25, 0.3$ corresponding to three phases of over critical, critical, and under critical. ω is fixed at 2. As you see after elapsing sufficient time, the system executes a harmonic motion with a ω_0 dependent amplitude. It is simply possible to numerically evaluate the motion amplitude in the steady state. For details see appendix 3.C. Figure (3.17) shows the dependence of the computed amplitude on ω. We see that for certain values of ω, there is resonance frequency ω_R at which the steady motion amplitude is maximum. This resonance frequency is simply obtained by minimising the denominator of (3.42). It turns out that:

$$\omega_R = \sqrt{\omega_0^2 - 2\beta^2} \tag{3.44}$$

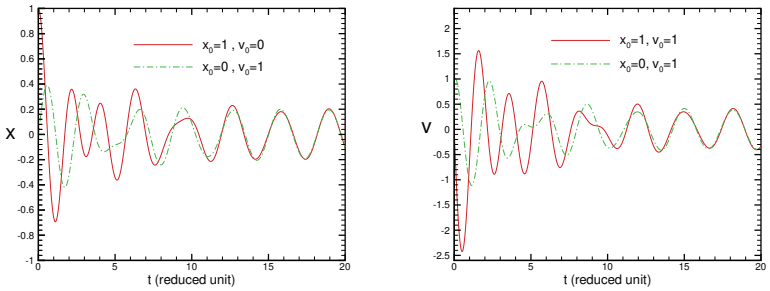

Figure 3.16: *Position-time (left) and velocity-time (right) curves of a driven damped linear oscillator for two initial conditions:* $x_0 = 1, v_0 = 0$ *and* $x_0 = 0, v_0 = 1$. *The parameters values are:* $\omega_0 = 3, \gamma = 0.5$ *and* $\omega = 2$.

It is apparent that for $\beta > \frac{\sqrt{2}}{2}\omega_0$ there will be no resonance.

3.7 Oscillation of a Pendulum

The pendulum motion is a common oscillatory motion in nature. The theoretical formulation of the problem dates back to Galileo's time in the early seventeen century. He conducted a series of experiments and could give physical accounts of this perpetual motion. Galileo's argument was incompatible with Aristotle's physical laws. Huygens obtained the motion period in terms of the pendulum length and gravitational field g. A simple pendulum consists of a particle or bob of mass m attached to the lower end of a rigid string of length L and negligible mass. The upper end of the string pivots without friction on a support. If the bob is pulled to one side from its equilibrium position and released, the pendulum swings in a vertical plane if Coriolis force is neglected. See figure (3.18) for illustration. If θ (in radian) denotes the pendulum deflection angle from the vertical equilibrium position the equation of motion in the absence of friction becomes:

$$\frac{d^2\theta}{dt^2} = -\frac{g}{L}\sin\theta \tag{3.45}$$

Equation (3.45) is a nonlinear second-order differential equation. It cannot be solved exactly and we have to resort to analytical approximations and numerics. If θ is sufficiently small one can replace $\sin\theta$

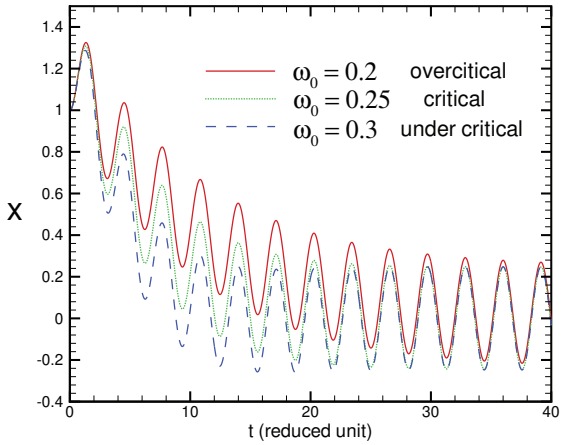

Figure 3.17: *The computed Position-time plots for three values of ω_0. The simulation data are obtained by the Euler-Richardson algorithm with a time step of $\tau = 0.01$.*

with θ and the equation becomes linear analogous to the one we encountered in the linear harmonic oscillator motion. You simply need to replace x by θ and $\frac{k}{m}$ by $\frac{g}{L}$ and obtain the solution:

$$\theta(t) = \theta_0 \cos(\omega_0 t + \phi) \tag{3.46}$$

θ_0 and ϕ are constants that depend on the initial displacement and velocity of the pendulum and $\omega_0 = \sqrt{\frac{g}{L}}$. In particular, the motion period turns out to be $T = 2\pi\sqrt{\frac{L}{g}}$. In the small angle approximation, the period does not depend on the motion angular amplitude θ_0. This is not true when initial angular displacement θ_0 is not small. Let us first evaluate the motion amplitude analytically for large deflections. To this end, we utilise the energy conservation law as follows:

$$E = E_K + E_P = \frac{1}{2}mL^2\dot{\theta}^2 + mgL(1 - \cos\theta) \tag{3.47}$$

Note the zero of the gravitational potential energy is taken as the rest position of the pendulum mass i.e.; at $\theta = 0$. For simplicity,

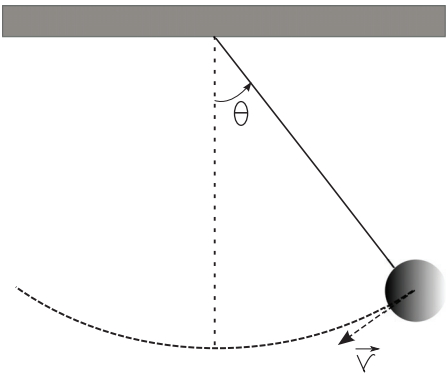

Figure 3.18: *A pendulum performs oscillatory motion in the gravitational field.*

we assume the initial angular velocity is zero and therefore the total energy becomes $E = mgL(1 - \cos\theta_0)$ where θ_0 is the initial angular displacement. From (3.47) we simply find: $\frac{d\theta}{dt} = \sqrt{\frac{2g}{L}(\cos\theta - \cos\theta_0)}$. In terms of half angles, the motion's period becomes:

$$\frac{T}{4} = \int_{\theta_0}^{0} \frac{dt}{d\theta} d\theta = \sqrt{\frac{L}{4g}} \int_{0}^{\theta_0} \frac{d\theta}{\sqrt{\sin^2\frac{\theta_0}{2} - \sin^2\frac{\theta}{2}}} \tag{3.48}$$

By introducing $z = \frac{\sin\frac{\theta}{2}}{\sin\frac{\theta_0}{2}}$ and $k = \sin\frac{\theta_0}{2}$ the integral becomes:

$$T = 4\sqrt{\frac{L}{g}} \int_{0}^{1} \frac{dz}{\sqrt{(1-z^2)(1-k^2z^2)}} \tag{3.49}$$

This integral is the standard elliptical integral of the first kind. Unfortunately, there is no exact solution in terms of known function. Putting the Taylor expansion of the term $(1 - k^2z^2)^{-\frac{1}{2}} = 1 + \frac{k^2z^2}{2} + \frac{3k^4z^4}{8} + \cdots$ in (3.49) gives the following expansion for the period T:

$$T = 2\pi\sqrt{\frac{L}{g}}(1 + \frac{k^2}{4} + \frac{9k^4}{64} + \cdots) \tag{3.50}$$

Substituting in terms of θ_0 gives (Garcia, 1999):

$$T = 2\pi\sqrt{\frac{L}{g}}(1 + \frac{\theta_0^2}{16} + \frac{11\theta_0^4}{3072} + \cdots) \tag{3.51}$$

You see when the initial deflection angle θ_0 is small one recovers the well-known result. Let us now solve the problem numerically. For this purpose, we recast the differential equation (3.45) as two first-order ones as follows:

$$\frac{d\omega}{dt} = -\frac{g}{L}\sin\theta \ ; \ \frac{d\theta}{dt} = \omega \tag{3.52}$$

It turns out that implementing the Euler algorithm leads to numerical instability. The Euler-Cromer algorithm gives:

$$\omega_{n+1} = \omega_n - \frac{g}{L}\sin\theta_n\tau \ ; \quad \theta_{n+1} = \theta_n + \omega_{n+1}\tau \tag{3.53}$$

We noticed in a harmonic oscillator that in problems involving oscillatory motion, the Euler-Cromer method conserves energy over a complete period of motion. We set the pendulum length L in such a way that $\omega_0 = \sqrt{\frac{g}{L}} = 3$. Also the bob mass m is taken to be unity. Let us denote the maximum angle θ by θ_0. Initial angle and angular velocity are set to $\theta(0) = \theta_0 = 0.2\ rad$ and $\dot{\theta}(0) = 0$ respectively. Three numerical algorithms were used: Euler-Cromer, Euler-Richardson, and second-order Runge-Kutta. For $\Delta t = 0.005$ the evaluated periods are $T_{comp} = 2.09, 2.1, 2.1$ for the three methods correspondingly. The analytical period by truncating the series after the fourth order is $T_{analytic} = 2.099$. The corresponding small angle period is $T = 2\pi\sqrt{\frac{L}{g}} = 2.094$. Figure (3.19) shows deflection angle θ versus time for various values of initial displacement computed by the second-order Runge-Kutta algorithm. The values for T are: 2.096, 2.099, 2.115, and 2.178 respectively. In figure (3.20) we have depicted the dependence of period T on the maximum angle obtained from computation and analytical solution. In small θ_0 there is a very good agreement between the analytical and computational solutions. Deviation for values of θ_0 larger than one radian is seen.

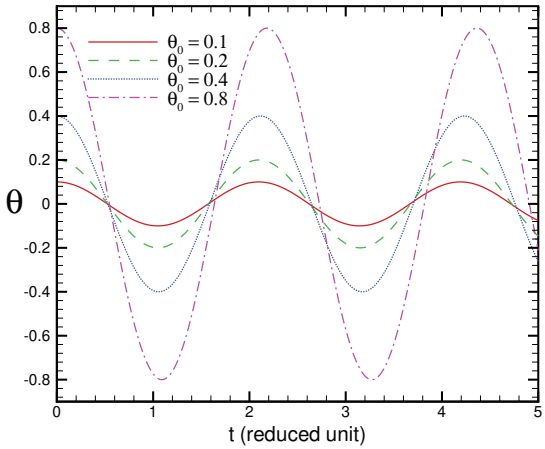

Figure 3.19: *The pendulum deflection angle θ vs time for some values of θ_0 in rads. The second-order Runge-Kutta algorithm has been used with a time step of $\tau = 0.01$.*

3.8 Driven damped nonlinear pendulum

Now we consider more complicated and realistic situations. We wish to add three ingredients to the pendulum problem simultaneously: dissipation, nonlinearity, and drive. Possible sources of friction include the effective bearing where the string of the pendulum connects to the support, air resistance, etc. For simplicity, we assume the damping forces take the simple linear form $-\gamma \frac{d\theta}{dt}$. The minus sign guarantees that the damping force always opposes the pendulum motion. Analogous to the linear harmonic oscillator, we take the driving force to be a sinusoidal one with angular frequency ω. The Newton equation of motion becomes:

$$\frac{d^2\theta}{dt^2} = -\frac{g}{L}\sin\theta - \frac{\gamma}{mL}\frac{d\theta}{dt} + \frac{F_0}{mL}\cos\omega t \tag{3.54}$$

Denoting $\frac{\gamma}{mL}$ by q and $\frac{F_0}{mL}$ by A_0 the equation of motion becomes:

$$\frac{d^2\theta}{dt^2} = -\frac{g}{L}\sin\theta - q\frac{d\theta}{dt} + A_0\cos\omega t \tag{3.55}$$

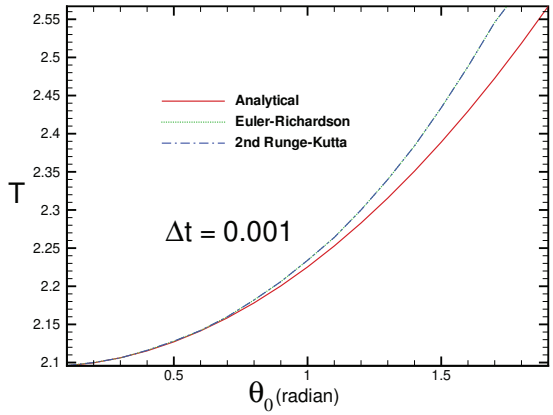

Figure 3.20: *T vs θ_0. Two numerical algorithms Euler-Richardson and 2nd order Runge-Kutta have been used with a time step of $\tau = 0.001$.*

Equation (3.55) is an inhomogeneous nonlinear second-order differential equation. Despite the impossibility of solving the equation exactly, it contains very rich and intriguing features that will be seen shortly. The computer programme `DrivenDampedPendulum` (see Appendix 3.D for details) numerically solves Newton's equation of motion for this driven nonlinear damped pendulum. Note that due to the presence of driving force, the deflection angle θ may become larger than π or less than $-\pi$ therefore we adjust the value of θ after each iteration (by subtraction or adding an appropriate multiple of $\pm\pi$) such that it lies in the interval $[-\pi, \pi]$. Recall that our pendulum can swing around its pivot point which corresponds to $|\theta| > \pi$. Figure (3.21) exhibits the time dependence of θ for various values of driving force amplitudes A_0. Due to nonlinearity in the problem, we do not have a full periodic motion but the overall pattern repeats itself. Another interesting point is that changing the driving force amplitude has a dramatic effect on the time dependence of θ. Let us now investigate the effect of changing the damping coefficient γ while keeping other parameters fixed. Figure (3.22) shows $\theta - t$ for various values of γ. The other parameters' values are :$A_0 = 1.2, \omega = 0.67$. Initial conditions are the same as in figure (3.21). When damping is small the

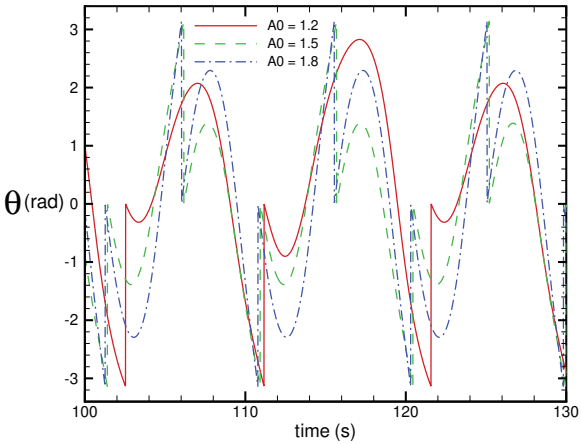

Figure 3.21: *Deflection angle θ vs time. The algorithms Euler-Richardson and 2nd order Runge-Kutta have been used (they give identical results) with a timestep $\tau = 0.01$. Initial conditions are $\theta_0 = 0.2$ rad and $\omega_0 = 0$. Other parameters are $q = 0.5$ and $\omega = 0.66$.*

nonlinearity is strong enough to make the motion non-periodic and quite irregular. By increasing the damping the motion characteristics become more regular and closer to a periodic motion. For sufficiently large damping the system's steady state is a harmonic motion with the same period of the driving force.

3.9 Damped oscillator: nonsinusoidal external force

Lastly in this section, we explore the characteristics of a damped linear oscillator which is driven by a periodic but nonsinusoidal external force. As an exemplification, consider the motion of a swing. It can be modeled by a damped linear oscillator. The external force is impulse type and is non-zero during the push time interval Δt. for simplicity assume the external force is a half sine wave with period T. The associated angular frequency of the external force is $\omega = \frac{2\pi}{T}$. The

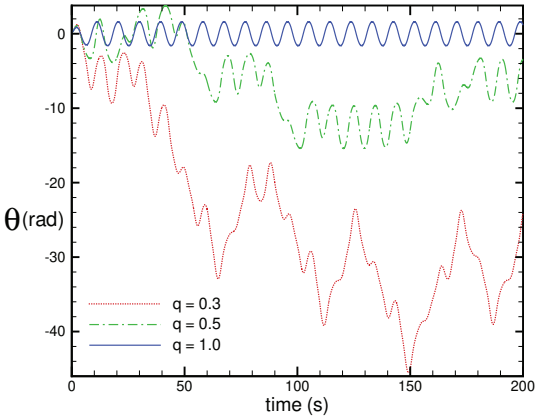

Figure 3.22: *Unrestricted θ vs time for various values of damping coefficient. The other parameters are as follows: $A_0 = 1.2, \omega = 0.66$. Initial conditions are $\theta_0 = 0.2$ rad and $\omega_0 = 0$. The Euler-Richardson algorithm has been used with $\tau = 0.01$.*

external force $F_{ext}(t)$ takes the following form in each period:

$$F_{ext}(t) = F_0 \cos \omega t \quad 0 \leq t \leq \frac{T}{4} \text{ and } \frac{3T}{4} \leq t \leq T \tag{3.56}$$

Figure (3.23) depicts the position versus time for various values of ω in the initial stages of the dynamics (steady-state not reached) and the steady-state behaviour. A comparison with the case of sinusoidal driving force with the same ω is made. You see a substantial difference when the driving force is not sinusoidal but impulsive. Despite the motion being periodic when the force is impulsive-like it is not harmonic and contains two characteristic time scales. In conclusion, we learnt in this chapter to numerically solve one particle oscillatory motion. In the next chapter, we shall learn how to generalise our methodology for systems of interacting particles undergoing oscillatory motion.

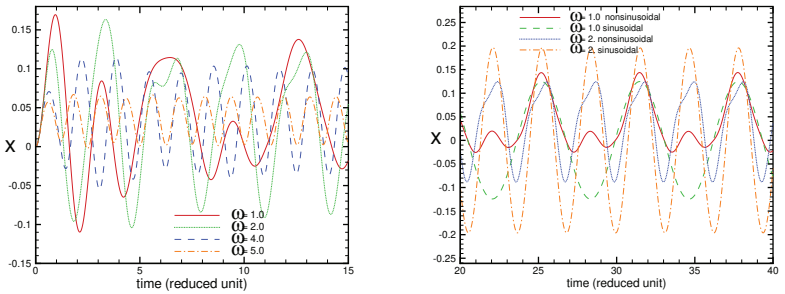

Figure 3.23: *x (Left) vs time for various values of driving force angular frequency ω. The other parameters are: $F_0 = 1., \omega_0 = 3, \gamma = 0.5$. Initial conditions are $x_0 = 0$ and $v_0 = 0$. Euler-Richardson algorithm has been used with $\tau = 0.01$. (Right) x vs time in steady state for various values of driving force angular frequency ω. A comparison to harmonic motion with the same frequency is done.*

3.10 problems

Problem 3.1 Let us adopt a complex number approach and take $x(t)$ as the real part of $X(t)$ which also satisfies the same differential equation.

(a) Show that:

$$X(t) = e^{-\beta t}(A_1 e^{i\omega_1 t} + A_2 e^{-i\omega_1 t}) \tag{3.57}$$

where constants A_1 and A_2 are complex numbers.

(b) Assuming that $X(0)$ is real and equals x_0 obtain A_1 and A_2 in terms of x_0 and v_0.

(c) Substitute A_1 and A_2 in (3.57) and then take the real part to show:

$$x(t) = e^{-\beta t}(x_0 \cos\omega_1 t + \frac{v_0 + \beta x_0}{\omega_1} \cos\omega_1 t) \tag{3.58}$$

(d) By writing (3.58) in the form $x(t) = e^{-\beta t} D\cos(\omega_1 t + \phi)$ show that $D = \sqrt{x_0^2 + (\frac{v_0+\beta x_0}{\omega_1})^2}$ and $\phi = -\tan^{-1}(\frac{v_0+\beta x_0}{\omega_1 x_0})$.

Problem 3.2 If the restoring force is a nonlinear function of x then the oscillator is called *anharmonic*. Take the restoring force as $-kx^{-\alpha}$ and numerically solve the problem with an appropriate algorithm.

(a) Obtain the position-time curve as well as the period for $\alpha = 2, 3$.

(b) Does the period depend on the amplitude? If yes, draw the period versus amplitude.

(c) Try to obtain the period with the method you learnt in classical physics (in the form of an integral) then try to approximately get the integral value.

(d) Compare your numerical to analytical results.

Problem 3.3 Consider a simple non-harmonic oscillator in one dimension. The force on the oscillator is $-F_0$ if it is on the right of the origin $x = 0$ and $+F_0$ if it is on the left of the origin. In other words, $F(x) = F_0[S(x) - S(-x)]$ where S is the step function. Note that the force is discontinuous at $x = 0$.

(a) Solve the problem analytically and prove that the motion is periodic with period $T = 4\sqrt{\frac{2m}{F_0}}$. See (Masoumi and Foulaadvand, 2008) for details.

(b) Numerically solve the problem and plot the $x - t$ curve as well as the motion period.

Problem 3.4 Suppose a point particle of mass m is moving with a constant velocity $v_0 > 0$ along the x axis in the $x < 0$ region. When it crosses the origin $x = 0$, an impulsive force is exerted on the particle. Numerically solve the problem and plot the $x - t$ curve. Take $t = 0$ as the time of exerting the impulsive force $F(t) = S_0\delta(t)$ on the particle. What is the S_0 dimension? For analytic solution see (Thornton and Marion, 2003).

Problem 3.5 Include a non-linear restoring force $-\lambda x^4$ in a damped oscillator and numerically solve the problem. Consider both underdamped and overdamped cases. Discuss the role of λ on the physics of the problem.

Problem 3.6 Numerically solve the motion of a damped simple oscillator when the dissipation term is quadratic in velocity that is $F_d = -Cv^2$. How many distinctive regimes you can identify?

Problem 3.7 Consider the forced oscillation of an anharmonic oscillator. Take the driving force sinusoidal $F_0 \cos \omega t$. Add a nonlinear term $-\lambda x^4$ to the quadratic potential $\frac{1}{2}kx^2$.

(a) Numerically solve the problem and plot the behaviour of the oscillator in time.

(b) Do you see any steady behaviour? C) Can you observe resonance phenomena (sharp increase in the oscillator amplitude)? Discuss the role of λ on all parts of the problem.

Chapter 4

Coupled Oscillations

4.1 Longitudinal motion

In the previous chapter, we became familiar with the numerical solutions of the oscillatory motion of a point-like particle, mostly in one dimension and, in some special cases, in higher dimensions. In this chapter, we wish to pursue our investigation of the basic features of oscillation in a system comprising many particles. Our ultimate task is to relate the microscopic description of the oscillatory dynamics of an interacting many-body system to the macroscopic wave motion of a continuum medium. You have already seen this in your undergraduate classical mechanics and wave courses, but here, we intend to verify it in a numerical and computational context. Let us start with the well-known problem of N coupled oscillators (in one dimension) connected to each other by a set of springs. We consider the fixed boundary condition where two springs connect the first and the last masses to two rigid walls of infinite mass. For simplicity, we assume the oscillators (masses) are point-like, and springs are massless. Moreover, the equilibrium distance between masses is taken to be a. See figure (4.1) for illustration. Taking the masses to be $m_1, m_2, \cdots, m_N$, the spring constants $k_1, k_2, \cdots, k_{N+1}$ and the deviation from the equilibrium position at time t by $u_1(t), u_2(t), \cdots, u_N(t)$, the Newton equations of

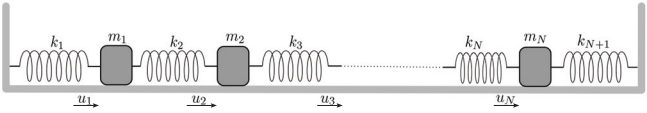

Figure 4.1: *A linear system of coupled mass-spring with fixed boundary condition.*

motion for this chain of oscillators become:

$$\begin{cases} m_1 \frac{d^2 u_1}{dt^2} = -k_1 u_1 - k_2(u_1 - u_2) \\ m_i \frac{d^2 u_i}{dt^2} = -k_i(u_i - u_{i-1}) - k_{i+1}(u_i - u_{i+1}) \quad i = 2, \cdots, N-1 \\ m_N \frac{d^2 u_N}{dt^2} = -k_N(u_N - u_{N-1}) - k_{N+1} u_N. \end{cases} \tag{4.1}$$

Specifying the initial conditions $u_1(0), \cdots, u_N(0)$ and $\dot{u}_1(0), \cdots, \dot{u}_N(0)$ it would be possible to solve the equations at least numerically. For some special cases, analytical solutions exist. The most straightforward solvable case is when all the masses are equal to each other, and all the spring constants are identical. The underlying idea is the concept of *normal mode*. In a normal mode, all the system ingredients (here the masses) perform an oscillatory motion with a common frequency ω but each with a different amplitude and phase. More concisely, in a normal mode, we have:

$$u_j(t) = \Re(a_j e^{i\omega t}) = |a_j| \cos(\omega t + \phi_j) \tag{4.2}$$

where it is understood that $a_j = |a_j| e^{i\phi_j}$ is a complex number. The use of complex numbers considerably facilitates mathematical calculations. We recall that all the results can be equivalently obtained without resorting to complex numbers. Whether a normal mode exists crucially depends on the microscopic detail of the system, boundary conditions, etc. In principle, for a linear oscillatory system possessing N degrees of freedom, we have N normal modes i.e., N normal frequencies $\omega_1, \omega_2, \cdots, \omega_N$ which are associated with their N amplitudes and phases for each degree of freedom. The mode characteristics crucially depend on the boundary conditions. For a one-dimensional coupled system of mass-spring confined between two rigid walls, the normal modes turn out to be:

$$\omega_n = 2\sqrt{\frac{k}{m}} \sin \frac{n\pi}{2(N+1)} \quad n = 1, 2, \cdots, N \tag{4.3}$$

For details, you may see standard classical mechanics textbooks such as (Thornton and Marion, 2003). In mode n, the amplitude of mass j displacement can be written as:

$$a_{j,n} = a_n \sin \frac{jn\pi}{N+1} \tag{4.4}$$

Note a_n is a complex number. The linearity of the equations implies the superposition principle, according to which the general solution will be a linear combination of all modes:

$$u_j(t) = \Re[\sum_{n=1}^{N} c_n a_n \sin \frac{jn\pi}{N+1} e^{i\omega_n t}] \tag{4.5}$$

Showing the unknown $c_n a_n$ by $|c_n||a_n|e^{i\delta_n} = d_n e^{i\delta_n}$ and taking the real part we have:

$$u_j(t) = \sum_{n=1}^{N} \sin \frac{jn\pi}{N+1} (\mu_n \cos \omega_n t - \nu_n \sin \omega_n t) \tag{4.6}$$

In which $\mu_n = d_n \cos \delta_n$ and $\nu_n = d_n \sin \delta_n$. We have $2N$ unknowns $\mu_1, \mu_2, \cdots, \mu_N$ and $\nu_1, \nu_2, \cdots, \nu_N$ which will be determined by the initial conditions. Using some orthogonality relation in trigonometry (Thornton and Marion, 2003) they turn out to be:

$$\mu_n = \frac{2}{N+1} \sum_{j=1}^{N} u_j(0) \sin \frac{jn\pi}{N+1} \tag{4.7}$$

$$\nu_n = -\frac{2}{\omega_n (N+1)} \sum_{j=1}^{N} \dot{u}_j(0) \sin \frac{jn\pi}{N+1} \tag{4.8}$$

Before we discuss the general form where the masses and springs are different, I would like to address other boundary conditions for the present case where the masses and spring constants are identical. In the first case, there is no wall, and N masses are connected by $N-1$ springs. In this case, the motion equations of the first and last mass change, but the other ones' motion equations remain unchanged. We have:

$$\begin{cases} m\frac{d^2 u_1}{dt^2} = -k(u_1 - u_2) \\ m\frac{d^2 u_N}{dt^2} = -k(u_N - u_{N-1}). \end{cases} \tag{4.9}$$

It can be shown that N normal modes will be (Aghamohammadi, 2021):

$$\omega_n = 2\sqrt{\frac{k}{m}} \sin \frac{n\pi}{2N} \quad n = 0, 1, 2, \cdots, N-1 \tag{4.10}$$

Note that the first mode i.e., $n = 0$ has a zero frequency. This arises from the translational symmetry and describes an oscillatory motion with an infinite period i.e., a rigid translation. The other example describes a situation where the last mass is connected to the first one instead of being connected to a wall. We can imagine all the masses moving in a circle. Here the equations of the first and last mass become:

$$\begin{cases} m\frac{d^2u_1}{dt^2} = -k(2u_1 - u_2 - u_N) \\ m\frac{d^2u_N}{dt^2} = -k(2u_N - u_1 - u_{N-1}). \end{cases} \tag{4.11}$$

It can be shown that N normal modes will be (Aghamohammadi, 2021):

$$\omega_n = 2\sqrt{\frac{k}{m}} |\sin \frac{n\pi}{N}| \quad n = 0, 1, 2, \cdots, N-1 \tag{4.12}$$

Similarly, the translational symmetry implies that the first modal frequency becomes zero. We end this issue by noting that in the $N \to \infty$ limit, the density of states does not depend on the type of boundary condition. Now we turn to the most general case where the masses and spring constants are unequal.

4.1.1 unequal masses and spring constants

For this general case, we can write the set of N differential equations in (4.1) in a matrix form as follows:

$$\mathbf{M}\frac{d^2U}{dt^2} = -\mathbf{K}U \tag{4.13}$$

In which $U^\dagger = (u_1, u_2, \cdots, u_N)$, the mass matrix $\mathbf{M}_{ij} = m_i\delta_{i,j}$ and the $\mathbf{K}$ matrix elements are as follows:

$$K_{i,j} = (k_i + k_{i+1})\delta_{i,j} - k_i\delta_{i,j+1} - k_{i+1}\delta_{i,j-1} \tag{4.14}$$

Note that for other types of boundary conditions, the elements of matrix $\mathbf{K}$ will change slightly, however, the formalism we present below will work irrespective of the boundary condition. You can simply verify that the matrix $\mathbf{K}$ is symmetric. According to (4.14) we have:

$$K_{j,i} = (k_j + k_{j+1})\delta_{j,i} - k_j\delta_{j,i+1} - k_{j+1}\delta_{j,i-1} =$$

$$(k_j + k_{j+1})\delta_{i,j} - k_j\delta_{i,j-1} - k_{j+1}\delta_{i,j+1} \tag{4.15}$$

where use has been made from the symmetry of the Kronecker delta function. With interchanging $j \leftrightarrow i$ in the first term of (4.15), replacing j with $i+1$ in the second term, and $j+1$ with i in the third term we conclude:

$$K_{j,i} = (k_i + k_{i+1})\delta_{i,j} - k_{i+1}\delta_{i,j-1} - k_i\delta_{i,j+1} \tag{4.16}$$

Now you see that $K_{j,i} = K_{i,j}$, which proves that matrix $\mathbf{K}$ is symmetric. As you will shortly notice, this symmetry greatly helps us in solving the differential equations (4.1). To solve (4.13) analytically, we perform a linear change of variable from U to V via a real-valued matrix $\mathbf{A}$:

$$V = \mathbf{A}U \tag{4.17}$$

The set of equations (4.13) becomes:

$$\mathbf{M}\frac{d^2V}{dt^2} = -\mathbf{AKA^{-1}}V. \tag{4.18}$$

If the matrix $\mathbf{A}$ is properly chosen such that the combination $\mathbf{D} = \mathbf{AKA^{-1}}$ is diagonal, we can simply solve the equations

$$\mathbf{M}\frac{d^2V}{dt^2} = -\mathbf{D}V \tag{4.19}$$

In terms of initial conditions, we have:

$$V(0) = \mathbf{A}U(0); \quad \dot{V}(0) = \mathbf{A}\dot{U}(0) \tag{4.20}$$

Equations (4.19) are a set of decoupled linear second order differential equations $m_i\frac{d^2v_i}{dt^2} = -D_{ii}v_i \;\; i = 1, \cdots, N$. These uncoupled equations can be simply solved given the initial conditions in (4.20). Having

found $V(t)$, we simply obtain the original solution $U(t)$ by a matrix transformation as follows:

$$U(t) = \mathbf{A^{-1}} V(t) \tag{4.21}$$

It remains to see if it is always possible to obtain $\mathbf{A}$ such that the matrix $\mathbf{D}$ is diagonal. We know from linear algebra that since matrix $\mathbf{K}$ is symmetric, it is always possible to find the required matrix $\mathbf{A}$. A fundamental theorem in linear algebra asserts that for normal matrices (a real matrix is normal whenever it commutes with its transpose $[A, A^t] = 0$) we can always find such a diagonalising transformation (Lipschutz, 2012). You can easily verify that a symmetric matrix is normal. In fact, the columns of $\mathbf{A}$ are the corresponding eigenvectors of $\mathbf{K}$. Moreover, the diagonal elements of $\mathbf{D}$ will be the eigenvalues of $\mathbf{K}$. The problem thus reduces to solve the eigenvalue problem $\mathbf{K}X = \lambda X$. Since the matrix $\mathbf{K}$ is symmetric, its eigenvalues are real, and it would not be difficult, thanks to the band structure of $\mathbf{K}$, to show by the induction that all the eigenvalues are positive. In other words, the matrix $\mathbf{K}$ is a positive definite matrix. For details, you may consult this standard textbook on linear algebra (Lipschutz, 2012). Showing $D_{ii} = d_i$ by ω_i^2 the N frequencies ω_i will be the normal modes of the system, and we have:

$$v_i(t) = \alpha_i \cos(\omega_i t + \theta_i) \quad i = 1, 2, \cdots, N \tag{4.22}$$

$2N$ constant $\alpha_1, \alpha_2, \cdots, \alpha_N$ and $\theta_1, \theta_2, \cdots, \theta_N$ will be determined once the initial conditions are given. According to (4.17) we have:

$$u_i(t) = \sum_{j=1}^{N} A_{ij}^{-1} \alpha_j \cos(\omega_j t + \theta_j) \quad i = 1, 2, \cdots, N \tag{4.23}$$

You see that each mass performs an oscillatory motion which is a linear combination of N normal modes. In some special cases, there exists an analytical solution for eigenvalues. We became familiar with one in which all the masses were equal and all the spring constants identical. Another important example that is amenable to analytic solutions is a different mass among the other but equal ones. The different mass plays the role of impurity, and many exciting phenomena such as localisation arise (A. Aghamohammadi and Mousavi, 2017). We refer interested readers to the literature (A. Kolan and Titus, 1985;

Williams and Maris, 1985). I think we have had enough of analytics. Let us see how we can numerically proceed in the coupled oscillations problem.

4.2 Numerical approach

Two numerical approaches can be devised to solve the linear system of differential equations in (4.13). In the first method, one numerically solves the eigenvalue problem $\mathbf{K}X = \lambda X$ to find the corresponding eigenvalues and eigenvectors. There are numerous packages and canned subroutines that can do the task for you. Nevertheless, I greatly warn you that despite its simplicity, the solution of eigensystems is a fairly complicated business. All these methods and subroutines are subject to serious numerical errors if the matrix $\mathbf{K}$ is mathematically ill-conditioned. Readers are referred to the advanced numerical linear algebra textbooks for further details. However, I strongly recommend that you use canned routines and avoid using your own subroutines. Almost all the canned routines nowadays trace their ancestry back to the routines published in (J. H. Wilkinson and Bauer, 1986). A public-domain implementation of the handbook routines in FORTRAN is the EISPACK set of programmes. Open-source libraries such as LINPACK and LAPACK are among other well-known matrix solvers. One of the best routines I can suggest is `tqli` which you can find in the masterpiece *Numerical Recipe* (W. H. Press and Flannery, 2002). This routine gives you all the eigenvalues and the associated eigenvectors of a real symmetric matrix. Hopefully, for many purposes in physics and engineering, the matrix we wish to find its spectrum is real symmetric, and we will rarely encounter to consider non-symmetric matrices. We will later come back to the above method in more detail, and will solve some problems. Let us now discuss the second approach for solving equations (4.13). This approach is the natural generalization of the previous methods implemented for solving linear differential equations. You remember that the Euler-Cromer did the job for us when we had only one particle. Let us devise this method for our coupled-oscillators problem. For this purpose, we turn the second-order equations (4.13) into $2N$ first-order ones:

$$\frac{du_i}{dt} = v_i \quad i = 1, \cdots, N \tag{4.24}$$

$$\begin{cases} m_1 \frac{dv_1}{dt} = -k_1 u_1 - k_2(u_1 - u_2) \\ m_i \frac{dv_i}{dt} = -k_i(u_i - u_{i-1}) - k_{i+1}(u_i - u_{i+1}) \quad i = 2, \cdots, N-1 \\ m_N \frac{dv_N}{dt} = -k_N(u_N - u_{N-1}) - k_{N+1} u_N. \end{cases} \tag{4.25}$$

Denoting the displacement and velocity of mass i at time step n by u_i^n and v_i^n, the generalisation of the Euler-Cromer algorithm becomes:

$$v_1^{n+1} = v_1^n - \tau[\frac{k_1}{m_1} u_1^n + \frac{k_2}{m_1}(u_1^n - u_2^n)] \tag{4.26}$$

$$v_i^{n+1} = v_i^n - \tau[\frac{k_i}{m_i}(u_i^n - u_{i-1}^n) + \frac{k_{i+1}}{m_i}(u_i^n - u_{i+1}^n)] \; i = 2, \cdots, N-1 \tag{4.27}$$

$$v_N^{n+1} = v_N^n - \tau[\frac{k_N}{m_N}(u_N^n - u_{N-1}^n) + \frac{k_{N+1}}{m_N} u_N^n] \tag{4.28}$$

$$u_i^{n+1} = u_i^n + \tau v_i^{n+1} \quad i = 1, 2, \cdots, N \tag{4.29}$$

Given the initial conditions $U(0)$ and $\dot{U}(0)$, one simply can proceed iteratively in time. Aside from Euler-Cromer, we can employ more advanced algorithms. Let us introduce an important one that is very useful and common for solving ordinary differential equations the so-called *Runge-Kutta* algorithm. We first introduce it in the context of a single first-order differential equation and then will generalize it to the case of a system of second-order differential equations. The advantage of the Runge-Kutta (RK) algorithm is that it applies to the general case of nonlinear differential equations.

4.2.1 Runge-Kutta (RK) algorithm

We begin from the simplest case where we have only one dependent variable x which depends on an independent variable t. Note that x and t are not necessarily position and time. They could denote any variable. The most general first-order nonlinear differential equation can be written as follows:

$$\frac{dx}{dt} = f(x, t) \tag{4.30}$$

where $f(x,t)$ is a given function. The equation is endowed with a given initial condition $x(0)$. There are various orders of Runge-Kutta algorithms, namely second order, fourth order, etc. These are all based on the Taylor expansion of $x(t+\tau)$. As usual, I avoid giving the proof and only state the result here. For details, you may consult books on numerical mathematics and computational physics book such as (H. Gould and Chriastian, 2006; Scherer, 2010; Vesely, 2001). We first introduce k_1^n and k_2^n as follows:

$$k_1^n = f(x_n, t_n)\tau; \quad k_2^n = f(x_n + \frac{k_1^n}{2}, t_n + \frac{\tau}{2})\tau \tag{4.31}$$

where as usual, τ denotes the time step and $x_n = x(t_n)$. The most common form of the second-order Runge-Kutta algorithm, hereafter shown by RK2, is:

$$x_{n+1} = x_n + k_2^n + o(\tau)^3 \tag{4.32}$$

It can be shown that the most general form of the RK2 algorithm has the following form provided $c_1 + c_2 = 1$:

$$x_{n+1} = x_n + c_1 k_1^n + c_2 k_2^n + o(\tau)^3 \tag{4.33}$$

The form (4.32) corresponds to $c_1 = 0,\ c_2 = 1$. Another common choice corresponds to $c_1 = \frac{1}{3},\ c_2 = \frac{2}{3}$ which is known as *Ralston* algorithm. Note that the form of k_2^n depends on the choice for c_1 and c_2. In the Ralston algorithm, we have $k_2^n = f(x_n + \frac{3k_1^n}{4}, t_n + \frac{3\tau}{4})\tau$. In the fourth-order Runge-Kutta (RK4) algorithm, we should define two additional functions k_3^n and k_4^n as follows:

$$k_3^n = f(x_n + \frac{k_2^n}{2}, t_n + \frac{\tau}{2})\tau \tag{4.34}$$

$$k_4^n = f(x_n + k_3^n, t_n + \tau)\tau \tag{4.35}$$

The fourth-order Runge-Kutta algorithm turns out to be:

$$x_{n+1} = x_n + \frac{1}{6}(k_1^n + 2k_2^n + 2k_3^n + k_4^n) + o(\tau)^4 \tag{4.36}$$

Now that we have become familiar with the Runge-Kutta algorithm and its variants, let us generalise it to the case when the number of

dependent variables exceeds one. We do it for the simplest case of two dependent variables. Moreover, we assume the number of independent variables does not increase. Showing the dependent variables by x_1 and x_2, the most general set of first-order differential equations will be:

$$\frac{dx_1}{dt} = f_1(x_1, x_2, t) \tag{4.37}$$

$$\frac{dx_2}{dt} = f_2(x_1, x_2, t) \tag{4.38}$$

where functions f_1 and f_2 are known. For each variable x_1, and x_2, we define the functions $k_{1,x_1}, \cdots, k_{4,x_1}, k_{1,x_2}, \cdots, k_{4,x_2}$ as follows:

$$k^n_{1,x_1} = f_1(x_{1n}, x_{2n}, t_n)\tau; \quad k^n_{1,x_2} = f_2(x_{1n}, x_{2n}, t_n)\tau \tag{4.39}$$

where $x_{1n} = x_1(t_n),\ x_{2n} = x_2(t_n)$.

$$k^n_{2,x_1} = f_1(x_{1n} + \frac{k^n_{1,x_1}}{2}, x_{2n} + \frac{k^n_{1,x_2}}{2}, t_n + \frac{\tau}{2})\tau \tag{4.40}$$

$$k^n_{2,x_2} = f_2(x_{1n} + \frac{k^n_{1,x_1}}{2}, x_{2n} + \frac{k^n_{1,x_2}}{2}, t_n + \frac{\tau}{2})\tau \tag{4.41}$$

$$k^n_{3,x_1} = f_1(x_{1n} + \frac{k^n_{2,x_1}}{2}, x_{2n} + \frac{k^n_{2,x_2}}{2}, t_n + \frac{\tau}{2})\tau \tag{4.42}$$

$$k^n_{3,x_2} = f_2(x_{1n} + \frac{k^n_{2,x_1}}{2}, x_{2n} + \frac{k^n_{2,x_2}}{2}, t_n + \frac{\tau}{2})\tau \tag{4.43}$$

$$k^n_{4,x_1} = f_1(x_{1n} + k^n_{3,x_1}, x_{2n} + k^n_{3,x_2}, t_n + \tau)\tau \tag{4.44}$$

$$k^n_{4,x_2} = f_2(x_{1n} + k^n_{3,x_1}, x_{2n} + k^n_{3,x_2}, t_n + \tau)\tau \tag{4.45}$$

We can now express RK2 and RK4 algorithms. The second-order RK algorithm:

$$x_{1n+1} = x_{1n} + c_1 k^n_{1,x_1} + c_2 k^n_{2,x_1} + o(\tau)^3 \tag{4.46}$$

$$x_{2n+1} = x_{2n} + c_1 k^n_{1,x_2} + c_2 k^n_{2,x_2} + o(\tau)^3 \tag{4.47}$$

where the most natural choice for coefficients are $c_1 = 0$ and $c_2 = 1$. The fourth-order RK algorithm:

$$x_{1n+1} = x_{1n} + \frac{1}{6}(k^n_{1,x_1} + 2k^n_{2,x_1} + 2k^n_{3,x_1} + k^n_{4,x_1}) + o(\tau)^4 \quad (4.48)$$

$$x_{2n+1} = x_{2n} + \frac{1}{6}(k^n_{1,x_2} + 2k^n_{2,x_2} + 2k^n_{3,x_2} + k^n_{4,x_2}) + o(\tau)^4 \quad (4.49)$$

The generalisation to more than two dependent variables is straightforward. I am almost sure that if you have learnt the case of two variables, you can simply write the corresponding lengthy equations for more than two variables. Let us apply the case of two dependent variables to the Newton equation of motion in physics. For simplicity, we consider the one-dimensional motion of a point-like particle that is subjected to a force. In its most general case, the force can depend on time as well as the position and velocity of the particle. Our dependent variables are the particle's position x and its velocity v. As you know, we can recast the second-order Newton equation for a particle of mass m into two first-order equations as follows:

$$\frac{dx}{dt} = v \quad (4.50)$$

$$\frac{dv}{dt} = a(x, v, t) = \frac{F(x, v, t)}{m} \quad (4.51)$$

In our notation we have $f_1(x, v, t) = v$ and $f_2(x, v, t) = a(x, v, t)$. Explicitly the RK4 algorithm for one dimensional Newton equation turns out to be as follows:

$$k^n_{1,x} = v_n\tau; \quad k^n_{1,v} = a(x_n, v_n, t_n)\tau \quad (4.52)$$

$$k^n_{2,x} = (v_n + \frac{k^n_{1,v}}{2})\tau; \quad k^n_{2,v} = a(x_n + \frac{k^n_{1,x}}{2}, v_n + \frac{k^n_{1,v}}{2}, t_n + \frac{\tau}{2})\tau \quad (4.53)$$

$$k^n_{3,x} = (v_n + \frac{k^n_{2,v}}{2})\tau; \quad k^n_{3,v} = a(x_n + \frac{k^n_{2,x}}{2}, v_n + \frac{k^n_{2,v}}{2}, t_n + \frac{\tau}{2})\tau \quad (4.54)$$

$$k^n_{4,x} = (v_n + k^n_{3,v})\tau; \quad k^n_{4,v} = a(x_n + k^n_{3,x}, v_n + k^n_{3,v}, t_n + \tau)\tau \quad (4.55)$$

$$x_{n+1} = x_n + \frac{1}{6}(k^n_{1,x} + 2k^n_{2,x} + 2k^n_{3,x} + k^n_{4,x}) + o(\tau)^4 \quad (4.56)$$

$$v_{n+1} = v_n + \frac{1}{6}(k^n_{1,v} + 2k^n_{2,v} + 2k^n_{3,v} + k^n_{4,v}) + o(\tau)^4 \tag{4.57}$$

We are now able to implement the Runge-Kutta algorithm in our coupled system of mass-spring. Here the $2N$ dependent variables are $u_1, v_1, u_2, v_2, \cdots, u_N, v_N$. The equations of motion will be:

$$\frac{du_i}{dt} = v_i; \quad \frac{dv_i}{dt} = a_i(u_{i-1}, u_i, u_{i+1}) \quad i = 1, 2, \cdots, N \tag{4.58}$$

with appropriate u_0 and u_{N+1} associated with the given boundary conditions. The explicit form of a_i is given by equations (4.13). The programme `CoupledOscillators` (see Appendix 4.A for details) numerically solves the Newton equation of motion for the coupled mass-spring system with the Runge-Kutta algorithm.

4.2.2 Coupled oscillators: numerical results

Let us solve a problem to see how well the numerical solution can reproduce the analytical result. Consider the simplest case of $N = 2$ coupled oscillators. Take the initial condition so that the system is in its first mode. For example take: $u_1(0) = u_2(0) = u_0$ with zero initial velocities. In this mode, we have:

$$u_1(t) = u_2(t) = u_0 \cos(\omega_1 t). \tag{4.59}$$

In which $\omega_1 = 2\sqrt{\frac{k}{m}}\sin(\frac{\pi}{6}) = \sqrt{\frac{k}{m}}$ is the first mode frequency. When $u_1(0) = -u_2(0) = u_0$ the system is initiated in its second mode having frequency $\omega_2 = 2\sqrt{\frac{k}{m}}\sin(\frac{\pi}{3}) = \sqrt{\frac{3k}{m}}$. In this mode, we have:

$$u_1(t) = -u_2(t) = u_0 \cos(\omega_2 t). \tag{4.60}$$

Figure (4.2) sketches the computed position-time plots of both masses. When the initial displacements are random, the system no longer remains in a mode, and its constituents exhibit a motion that contains both mode frequencies. Figure (4.3) shows such a motion with this initial condition: $u_1(0) = +0.1, u_2(0) = -0.4$ and zero initial velocities. When N increases and initial conditions become random, the motion of each particle becomes more complicated, and in general, it contains all the modal frequencies. In figure (4.4), we show the motion of particles $n = 3$ and $n = 8$ in a system having $N = 10$

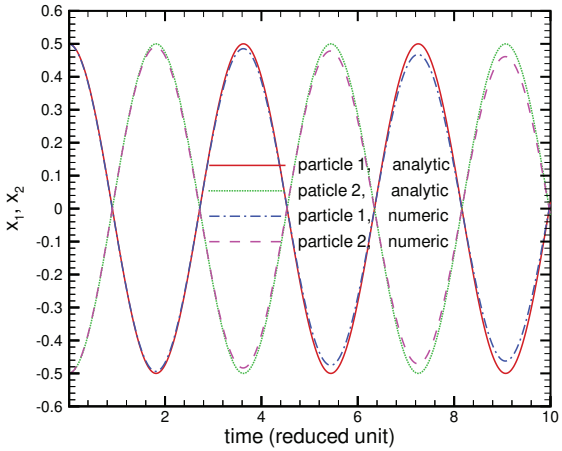

Figure 4.2: *Time evolution of two particles in the first mode. Mass m and stiffness constant k have been set to unity. The numerical algorithm has been velocity Verlet which can be seen to have rather poor performance compared to the exact solution.*

particles with initial displacements and velocities which are randomly and uniformly chosen in the interval $[-1, +1]$. We see in this case of entirely random initial condition particles exhibit a highly complicated behaviour involving contributions from all the modes. In this $N = 10$ particle system, let us consider another initial condition such that all the particles are initially at rest. Particle one is displaced one unit towards the right and released. All the other particles' initial displacements are zero. Figure (4.5) shows the computed position-time plot of particle 5, compared to the analytical solution. Figure (4.6) exhibits the phase space of particle five associated with the described initial condition. From figures (4.5) and (4.6) we conclude that the numerical solution obtained via RK2 deviates, to some extent, from the exact solution. It would be interesting to see how well the numerical simulation preserves energy conservation. We have compared the energy deviation from the constant initial energy for two algorithms: Euler Cromer and RK2. Figure (4.7) shows the time evolution of energy deviation $\Delta E = E_{num} - E$ for each algorithm. We see that ΔE

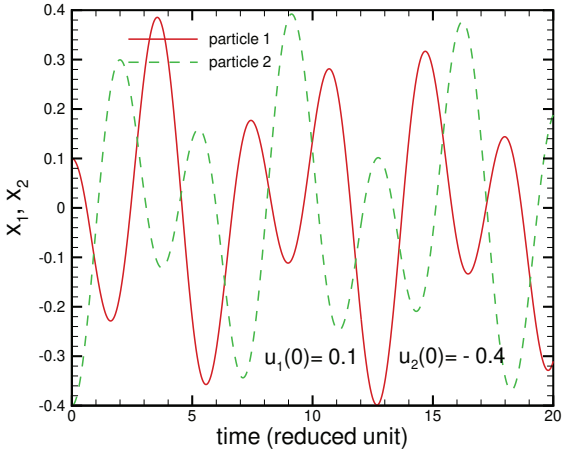

Figure 4.3: *Time evolution of two particles with random initial displacements* $u_1(0) = +0.1, u_2(0) = -0.4$ *and zero initial velocities. RK2 algorithm has been implemented with* $\tau = 0.01$.

increases initially but then begins to decrease, accompanied by fluctuations. By decreasing the time step, we can not avoid such a high energy deviation. It can be concluded that neither of these algorithms is good as far as energy conservation is concerned. For $\tau = 0.01$, we obtain $\Delta E = 0.4997, 0.4961$ for the Euler-Cromer and Runge-Kutta, respectively. These values are increased to $\Delta E = 0.4999, 0.4996$ correspondingly when the timestep is reduced to $\tau = 0.001$. This tells us that contrary to our intuition, we can not decrease the energy deviation by reducing the timestep. Before coming to the next problem, it would be instructive to discuss another type of boundary condition i.e.; the moving wall boundary condition. In this case, the left wall is assumed to perform a period motion with a given frequency Ω. As usual, $N + 1$ springs connect N masses to each other. The equations of motion can be written as follows:

$$m_i \frac{d^2 u_i}{dt^2} = -k_i(u_i - u_{i-1}) - k_{i+1}(u_i - u_{i+1}) \; i = 1, \cdots, N \quad (4.61)$$

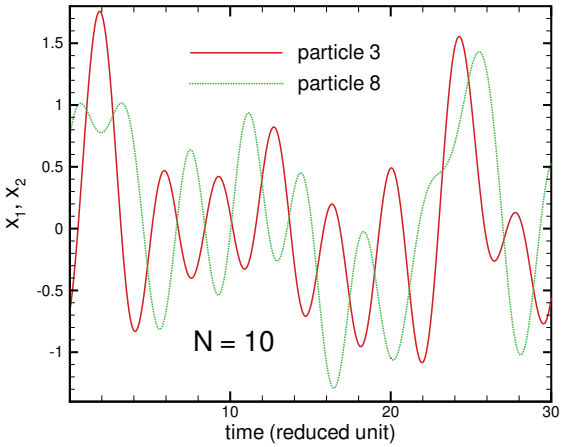

Figure 4.4: *The computed time evolution of particles 3 and 8 with random initial displacements in positions and velocities. RK2 algorithm with* $\tau = 0.01$ *has been implemented.*

Provided we set $u_0(t) = A\cos(\Omega t)$ and $u_{N+1}(t) = 0$. We leave it as an exercise to you to analytically find the modal frequencies for the periodic and free boundary conditions.

4.3 Forced coupled oscillations

We are all familiar with the forced oscillation of a single harmonic oscillator from our classical mechanics' course. However, we may have rarely dealt with this issue when there is more than one particle. For simplicity, we consider a one-dimensional system of identical coupled oscillators with only one of the masses, say mass s, driven by a sinusoidal driving force with frequency ω. The more general case where all the masses are driven can be treated by the superposition principle. In a system of coupled oscillators, we expect to have amplitudes resonance whenever ω equals one of the system modes. To gain a better insight, let us first formulate the problem mathematically. The equations of motion for a driven system with N coupled oscillators can be

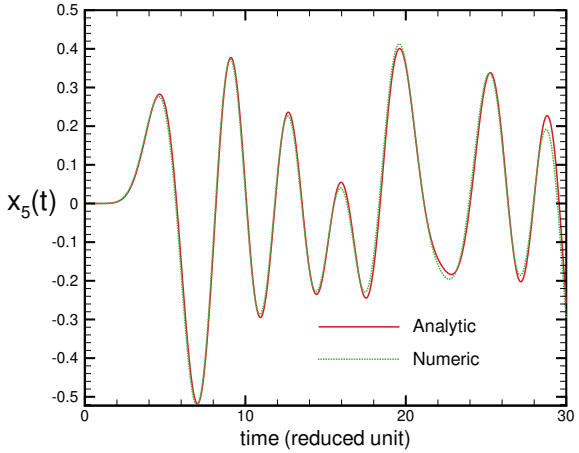

Figure 4.5: *Time evolution of* 5*-th particle. RK2 algorithm has been used with a time step* $\tau = 0.01$*. Comparison to the analytic result is also shown.*

written as follows:

$$m_i \ddot{u}_i = -k_i(u_i - u_{i-1}) - k_{i+1}(u_i - u_{i+1}) + \delta_{s,i} F_0 \cos\omega t \quad (4.62)$$

In a vectorial notation, we can recast these equations into the following compact form by introducing the vector $F^\dagger = (0, \cdots, F_0 \cos\omega t, \cdots, 0)$ with the term $F_0 \cos\omega t$ in the s place:

$$\mathbf{M}\ddot{U} = -\mathbf{K}U + F. \quad (4.63)$$

Note the matrix $\mathbf{K}$ is given in (4.14). By a unitary transformation $V = \mathbf{A}U$ to normal coordinates $\mathbf{V}^\dagger = (v_1, \cdots, v_N)$ we reach the following set of decoupled equations:

$$\mathbf{M}\ddot{V} = -\mathbf{D}V + \mathbf{A}F. \quad (4.64)$$

Each normal oscillator v_i has the following uncoupled equation of motion:

$$\ddot{v}_i = -\omega_i^2 v_i + \sum_j A_{ij}\delta_{s,j}\frac{F_0}{m_i}\cos\omega t \quad (4.65)$$

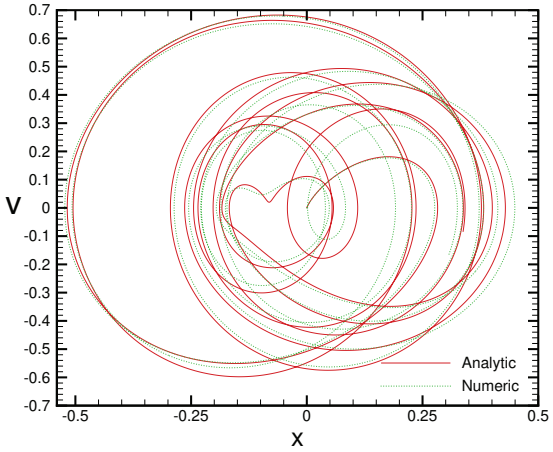

Figure 4.6: *Phase space diagram of* 5*-th particle with* $\tau = 0.01$.

which will be simplified as follows:

$$\ddot{v_i} = -\omega_i^2 v_i + A_{is}\frac{F_0}{m_i}\cos\omega t \quad i = 1, \cdots, N \tag{4.66}$$

Equation (4.66) is a set of forced oscillation equations with a sinusoidal driving force with the same frequency but a different amplitude. In the steady state, v_i has the following time dependence:

$$v_i = D_i(\omega)\cos(\omega t - \delta_i(\omega)). \tag{4.67}$$

In which the amplitude D_i is given below:

$$D_i(\omega) = \frac{A_{is}F_0}{m_i|\omega^2 - \omega_i^2|}. \tag{4.68}$$

When there are drag forces, a term $-\Gamma\dot{\mathbf{U}}$ is added to the right-hand side of (4.63), and we have:

$$\mathbf{M}\ddot{V} = -\mathbf{D}V - \mathbf{A}\Gamma\mathbf{A}^{-1}\dot{V} + \mathbf{A}F. \tag{4.69}$$

We assume the drag force is proportional to the velocity consequently, the matrix Γ is diagonal: $\Gamma_{ij} = \delta_{ij}\gamma_i$. Provided $B = \mathbf{A}\Gamma\mathbf{A}^{-1}$ is diag-

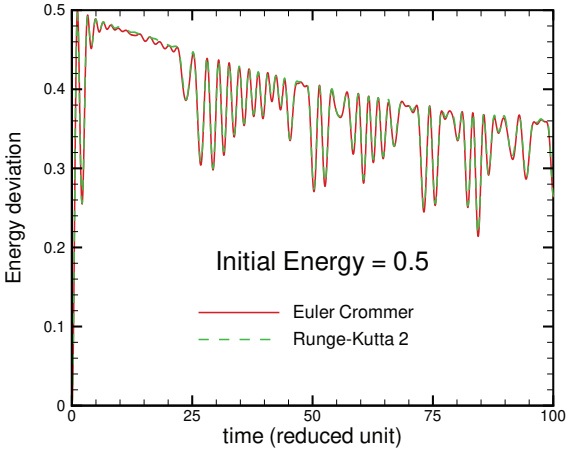

Figure 4.7: *Time evolution of energy deviation ΔE by two algorithms of Eular-Comer and RK2 with a time step $\tau = 0.01$.*

onal we can proceed analytically. The equations of motion become:

$$\ddot{v}_i = -\omega_i^2 v_i - b_i \dot{v}_i + A_{is} \frac{F_0}{m_i} \cos \omega t \quad i = 1, \cdots, N \tag{4.70}$$

The steady-state solution of equations (4.70) turns out to be:

$$v_i = D_i(\omega) \cos(\omega t - \delta_i(\omega)). \tag{4.71}$$

The amplitude $D_i(\omega)$ and the phase $\delta_i(\omega)$ can be simply obtained by replacing F_0 with $A_{is}F_0$, m with m_i and $\frac{\gamma}{m}$ with b_i in equations (3.42) and (3.43) of chapter 3. Now let us return to computation. We can easily modify our programme `CoupledOscillators` such that the driving force is also included. The programme `ForcedCoupledOscillators` (see Appendix 4.B for details) does the job for us. As an example, we have simulated the motion of a system of $N = 3$ forced-oscillators for fixed boundary conditions and have numerically obtained the amplitude dependence on ω for three oscillators as shown in figure (4.8). All the masses and spring constants have been set to unity. We see that each amplitude exhibits three resonances that coincide with three natural mode frequencies of the system.

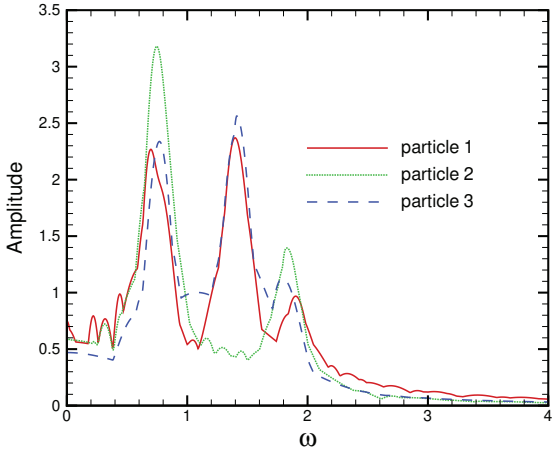

Figure 4.8: *Dependence of amplitudes $D_i(\omega)$, $i = 1, 2.3$ on driving frequency ω. The masses and spring constants have been set to unity. The drag coefficient was taken $b_i = \frac{\gamma_i}{m} = 0.05$. The RK2 algorithm with $\tau = 0.01$ is used.*

4.4 Fourier and spectral analysis

In this section, we intend to gain more insight into the problem of coupled oscillations by investigating the power spectrum of the system's masses displacements time series. Before doing that, let us briefly review the Fourier series. Every real-valued periodic function f with period T can be expanded in a Fourier series as follows:

$$f(t) = \frac{a_0}{2} + \sum_{k=1}^{\infty} (a_k \cos \omega_k t + b_k \sin \omega_k t) \tag{4.72}$$

where $\omega_k = k\omega_0 = k\frac{2\pi}{T}$. The Fourier coefficients are:

$$a_k = \frac{2}{T} \int_0^T f(t) \cos \omega_k t dt \quad k = 0, 1, \cdots \tag{4.73}$$

$$b_k = \frac{2}{T} \int_0^T f(t) \sin \omega_k t dt \quad k = 1, \cdots \tag{4.74}$$

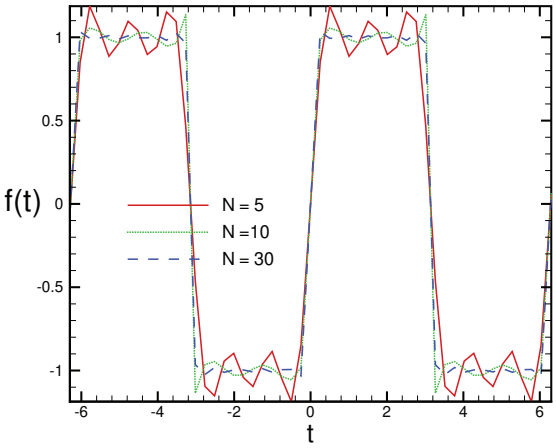

Figure 4.9: *Fourier series of step function sums for* $N = 5, 10, 30$.

Note $a_0/2$ is the average of f over the period T. Let us give an example. Consider the series:

$$f(t) = \frac{4}{\pi}(\sin t + \frac{1}{3}\sin 3t + \cdots) = \frac{4}{\pi}\sum_{k=1}^{N\to\infty}\frac{\sin(2k-1)t}{2k-1} \tag{4.75}$$

In figure (4.9), we have drawn the series in (4.75) for some finite values of N. The series represents the step function in the interval $[-\pi, \pi]$. In other words, the functional form of $f(t)$ is as follows:

$$f(t) = +1 \quad 0 < t < \pi \quad f(t) = -1 \quad -\pi < t < 0. \tag{4.76}$$

You see that the Fourier expansion gives poor results near discontinuity points of $f(t)$ such as $0, \pi$, and $-\pi$. Figure (4.10) shows the sketch of the sum up to 32 terms. You see quite a large error and small oscillations even when 32 terms are considered. These small oscillations that increase in amplitude as a sharp edge is approached are known as the Gibbs phenomenon. For further mathematical insight, you can see an excellent mathematical physics book (Myint-U and Debnath, 2007; Hassani, 2013). As another example, consider the periodic function $f(t) = t \quad -\pi < t < \pi$. Using (4.73) and (4.74) you can simply

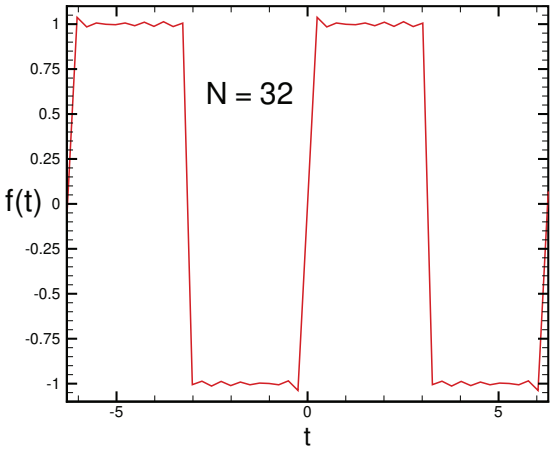

Figure 4.10: *Fourier series of the sum in (4.75) for* $N = 32$.

verify $a_k = 0$ and $b_k = \frac{2}{k}(-1)^{k-1}$. Figure (4.11) shows the Fourier series of the periodic $f(t) = t$ in the interval $[-\pi, \pi]$ for three values of $N = 5, 10, 50$. The partial sum of the Fourier series becomes:

$$f_N(t) = \sum_{k=1}^{N} (-1)^{k-1} \frac{2 \sin kt}{k}. \tag{4.77}$$

Gibbs' phenomenon is clearly seen. Near discontinuities at $\pm\pi$ strong oscillations occur.

4.5 Discrete Fourier transform

In practice, it is rarely possible to give the analytic form of a periodic function. Instead, a set of N discrete data points approximate the continuous function $f(t)$ in the period interval T. These N data points $f_i \;\; i = 0, 1, \cdots, N-1$ are function values sampled at regular sampling interval $t_i = i\Delta$. More concisely, we have $f_i = f(t_i) = f(i\Delta)$. Moreover, we have $T = N\Delta$. In this case, the infinite Fourier series coefficients a_k and b_k are substituted by N discrete values known as discrete

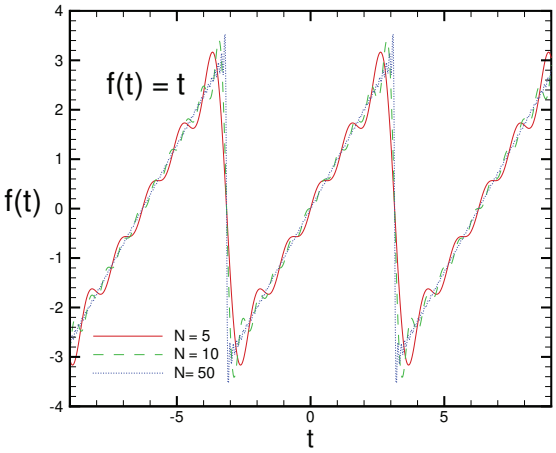

Figure 4.11: *Partial Fourier series for the periodic linear function* $f(t) = t$ *in the interval* $[-\pi, \pi]$.

Fourier transforms (DFT), which are evaluated by turning the integrals (4.73) and (4.74) into a sum as follows (H. Gould and Chriastian, 2006):

$$a_k = \frac{2\Delta}{T} \sum_{i=0}^{N-1} f_i \cos \omega_k t_i \tag{4.78}$$

$$b_k = \frac{2\Delta}{T} \sum_{i=0}^{N-1} f_i \sin \omega_k t_i \tag{4.79}$$

The independent DFT coefficients are $a_0, a_1, \cdots, a_{\frac{N}{2}}, b_1, \cdots, b_{\frac{N}{2}-1}$ comprising N independent DFT coefficients. Sometimes the DFT coefficients are expressed by complex numbers X_k:

$$X_k = \frac{1}{N} \sum_{j=0}^{N-1} e^{-\frac{2\pi i j k}{N}} \tag{4.80}$$

The independent complex DFT coefficients are $k = 0, 1, \cdots, \frac{N}{2} - 1$ (N is assumed to be an even number). Let us now solve a problem.

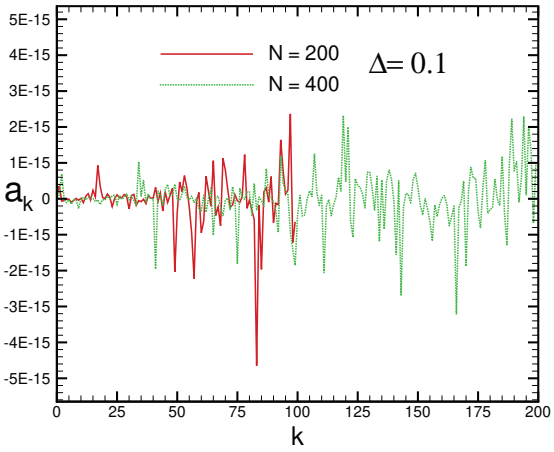

Figure 4.12: *DFT coefficients a_k for the function $f(t) = \sin\frac{\pi t}{10}$ with $N = 200, 400$ sampling points.*

Suppose $f(t) = \sin\frac{\pi t}{10}$. Choose the number of data points to be $N = 200$ and the sampling time $\Delta = 0.1$. Repeat your analysis for $N = 400$ and $\Delta = 0.1$. Explain your results by comparing the period of $f(t)$ with $N\Delta$, the assumed period. If the combination of N and Δ are not properly chosen, do you find any spurious results for the coefficients?

The function $f(t) = \sin(\frac{\pi t}{10})$ is periodic with the period $T = 20$. Figures (4.12) and (4.13) show the DFT coefficients a_k and b_k for $k = 0, 1, \cdots, \frac{N}{2} - 1$: As you can see, all a_k are practically zero. Theoretically, we have $a_k = 0$ due to the orthogonality relation:

$$\int_0^T \sin(\frac{2\pi nt}{T}) \cos(\frac{2\pi mt}{T}) = 0 \tag{4.81}$$

Moreover, in the time-continuous Fourier series, we should have $b_k = \delta_{1,k}$. Discrete Fourier series results are in good agreement with continuous results. In figures (4.14) and (4.15) we have sketched the Fourier coefficients for other combinations of N and Δ. We see spurious behaviour when $N\Delta$ is not an integer multiple of period T. Let us

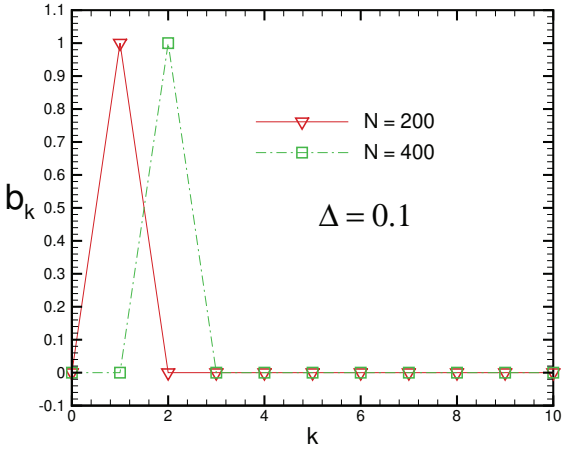

Figure 4.13: *DFT coefficients b_k for the function $f(t) = \sin \frac{\pi t}{10}$ with $N = 200, 400$ sampling points.*

consider the functions $f_1(t) = \sin \frac{\pi t}{10} + \sin \frac{\pi t}{5}$, $f_2(t) = \sin \frac{\pi t}{10} + \cos \frac{\pi t}{5}$, and $f_3(t) = \sin \frac{\pi t}{10} + \frac{1}{2} \cos \frac{\pi t}{5}$. Functions $f_1(t), f_2(t)$ and $f_3(t)$ are periodic with $T = 20$. Figure (4.16) shows a_k and b_k for $f_2(t)$ and $f_3(t)$ respectively with $N = 200$ and $\Delta = 0.2$ where $N\Delta = 20 = T$. The results are in good agreement with time-continuous Fourier coefficients. Figure (4.17) exhibits a_k and b_k for $f_2(t)$ and $f_3(t)$ respectively with $N = 210$ and $\Delta = 0.2$. Note $N\Delta = 21 \neq T$. As you see, the results are spurious. As our last example, consider a non-periodic function that rapidly goes to zero for $|t|$ large. We take $f(t) = t^3 e^{-t^2}$. Figure (4.18) shows a_k and b_k coefficients.

4.6 Power spectrum

The power spectrum is an important concept in data analysis. The power spectrum P_k of a set of data points is defined as:

$$P_k = \sqrt{a_k^2 + b_k^2} \tag{4.82}$$

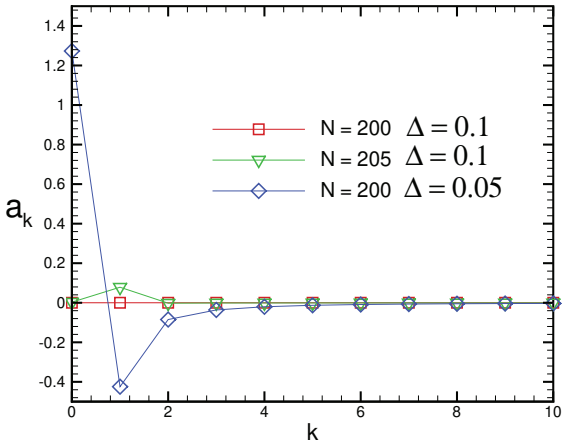

Figure 4.14: *Fourier coefficients* a_k *for* $N = 200, \Delta = 0.1, N = 205, \Delta = 0.1$ *and* $N = 200, \Delta = 0.05$.

Let us give some examples to illustrate the notion of the power spectrum. The examples are chosen from (H. Gould and Chriastian, 2006). In the first example, consider a data set with N points corresponding to $f(t) = 0.3\cos\frac{2\pi t}{T} + r$, where r is a random number uniformly chosen between 0 and 1. We have taken $T = 4$. In figure (4.19) $f(t) = 0.3\cos\frac{2\pi t}{T} + r$ is drawn for $N = 128$ sampling points with spacing $\Delta = \frac{4T}{N} = 0.125$. For each sample point $f_j = f(j\Delta)$ we choose a random number r uniformly distributed in $[0, 1]$. It is possible to identify a period visually. Figure (4.20) shows the power spectrum P_k. As we can see, there is a peak at $k = 4$, which corresponds to the periodicity of $f(t)$. The frequency f_k at which the power spectrum is maximum is $f_k = \frac{k}{\Delta N}$. Here $k = 4$ and we have $f_4 = \frac{1}{4}$. This gives a period $T = \frac{1}{f_4} = 4$ as expected. The power spectrum remains unchanged when $T = 4$ is increased to $T = 16$. As our next example, we consider a one-dimensional random walk. We will investigate the random walk in our future chapters, but for the moment, we borrow somewhat from your knowledge on this subject. Figure (4.21) shows the profile of a random walker, which changes its walk direction at each time step $\Delta t = 1$ with probability $p = 0.5$. The walker has taken 256 steps. No

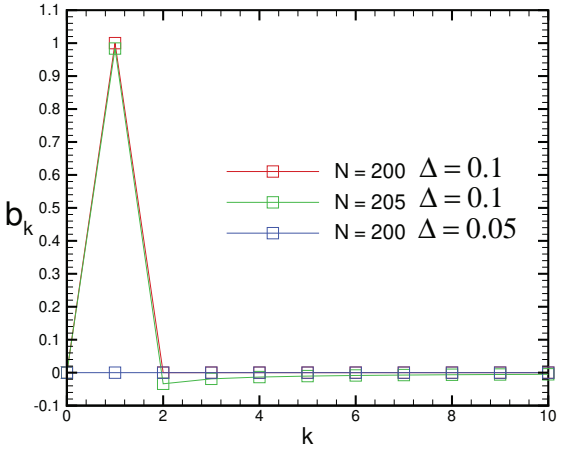

Figure 4.15: *Fourier coefficients* b_k *for* $N = 200, \Delta = 0.1; N = 205, \Delta = 0.1$ *and* $N = 200, \Delta = 0.05$.

particular frequency can be identified. Hence, we expect the power spectrum not to take a large value for large k. The power spectrum of this walk profile is shown in figure (4.22). The sharp peak at $k = 1$ is attributed to the large frequency of changing the walk direction in the next step. Let us now come back to our favourite problem of coupled oscillators. First, we take $N = 2$ oscillators (fixed boundary condition) and prepare the system in the initial condition corresponding to the first mode. More specifically, $u_1(0) = u_2(0) = 1$ with zero initial velocities. As usual, all the masses and spring constants have been set to unity. We numerically solve the system equations by the RK2 algorithm with a timestep $\tau = 0.01$. Figure (4.23) shows the evaluated power spectrum of the second mass both from the numerical solution and the exact one. You see that there is a peak at $\omega_k = 1$, which coincides with the first mode frequency $\omega_1 = \sqrt{\frac{k}{m}} = 1$ of the system. Note that the horizontal axis is versus $\omega_k = k\omega_0 = k\frac{2\pi}{T}$. Next, we take $N = 10$ and initialize the system in the third mode. According to equation (4.4) the initial displacements for the third mode $n = 3$

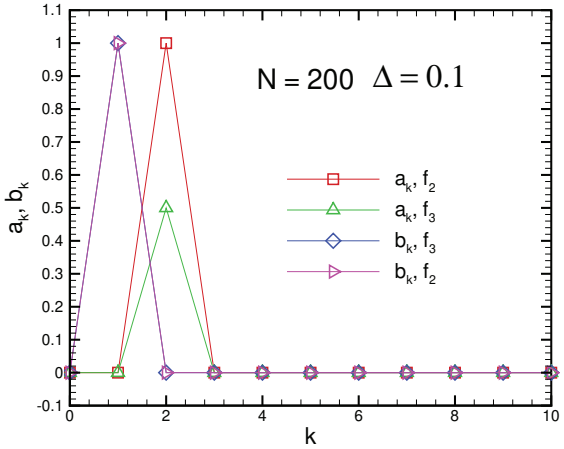

Figure 4.16: *DFT coefficients a_k, b_k for $f_2(t)$ and $f_3(t)$ with $N = 200, \Delta = 0.1$. Note that $N\Delta = 20 = T$.*

become as follows:

$$u_j(0) = \sin \frac{3\pi j}{11} \quad j = 1, \cdots, 10. \tag{4.83}$$

We expect the power spectrum to exhibit a peak at $\omega_3 = 2\sin(\frac{3\pi}{22}) \sim 0.83$. Figure (4.24) shows the corresponding power spectrum for the particle 5: The peak is located at $\omega_k = 0.83$, which coincides with the third modal frequency ω_3. The power spectrum is the same for all other particles because the system is set and remains in a mode. We now consider random initial displacements between -0.5 and $+0.5$ and zero initial velocities. Figure (4.25) shows the power spectrum (both numeric and analytic) for a $N = 2$ particle system sampled at $\Delta = 0.1$. Two peaks associated with the modal frequencies $\omega_1 = \sqrt{\frac{k}{m}} = 1$ and $\omega_2 = \sqrt{\frac{3k}{m}} = 1.7$ are in good agreement with the theoretical values of mode frequencies. Eventually, figure (4.26) exhibits the power spectrum for $N = 10$ particle system prepared with initial condition subjected to a random displacement of particles. Here we have taken $\Delta = 0.01$ and $T = 102.4$, which is four times the sampling time for

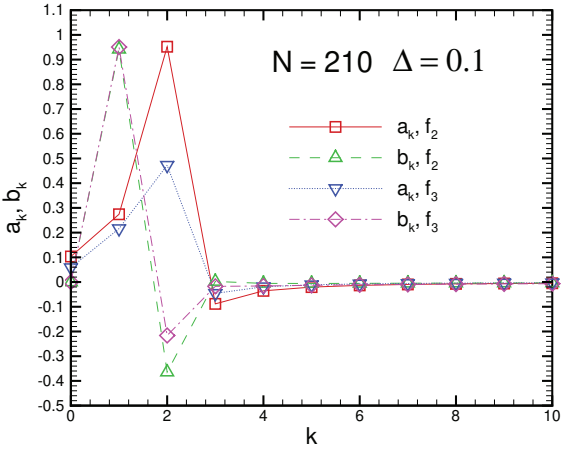

Figure 4.17: *Discrete Fourier coefficients* a_k, b_k *for* $f_2(t)$ *and* $f_3(t)$ *with* $N = 210, \Delta = 0.1$. *Note that* $N\Delta = 21 \neq T$.

$N = 2$ particle system. Note that all the 10 modal frequencies are not detected by looking at the power spectrum peaks. To remedy this problem, a windowing procedure is often prescribed. For details refer to (Garcia, 1999). So far, we have investigated a system where all the masses are identical and all the spring constants are equal. It is worth considering a disordered system. The disorder can be generated by having unequal masses or spring constants (or both). As a consequence of the disorder, the normal modes are no longer simple sinusoidal functions. Instead, some of them become localized such that only some of the particles move significantly while the others remain essentially at rest. This effect is known as the *Anderson localization* (A. Kolan and Titus, 1985). Typically, the modes above a certain frequency are localized, whereas those below this threshold frequency are extended. Let us consider a simple disordered system such that the mass of one oscillator is equal to one-fourth of the others. Let us set $N = 20$ and use fixed boundary conditions. We use random initial displacements between -0.5 and $+0.5$ and zero initial velocities. Data are sampled at intervals of $\Delta = 0.1$. We expect the normal mode frequencies to correspond to the well-defined peaks in the power

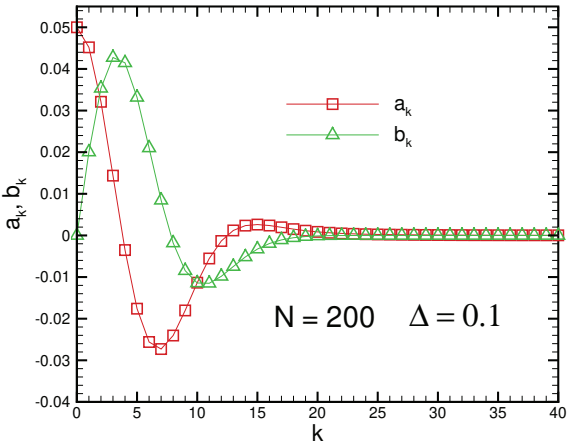

Figure 4.18: *DFT coefficients a_k, b_k for an aperiodic function $f(t) = t^3 e^{-t^2}$ with $N = 200, \Delta = 0.1$.*

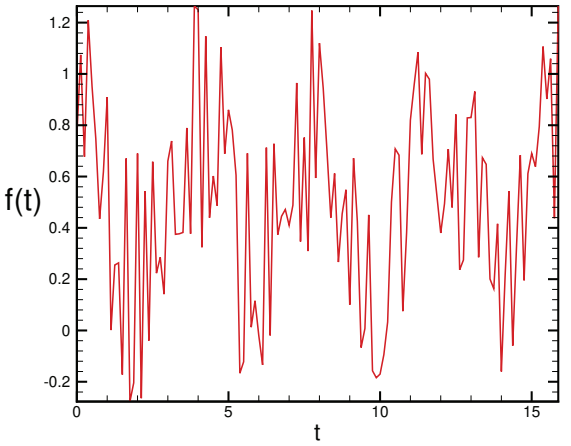

Figure 4.19: *Sampling of the stochastic function $f(t) = 0.3 \cos \frac{2\pi t}{T} + r$ in an interval of $4T$.*

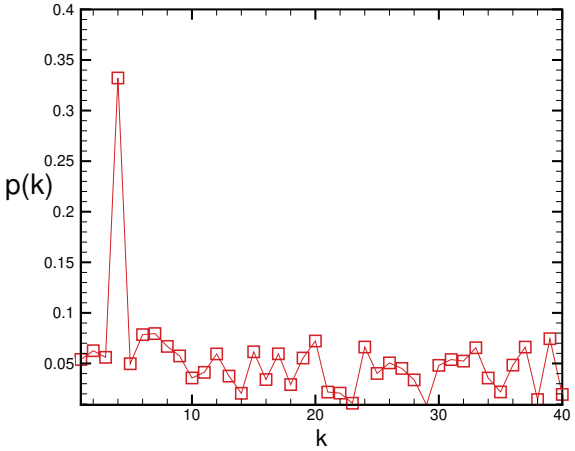

Figure 4.20: *Power spectrum of the function* $f(t) = 0.3\cos\frac{2\pi t}{T} + r$ *with* $T = 4$.

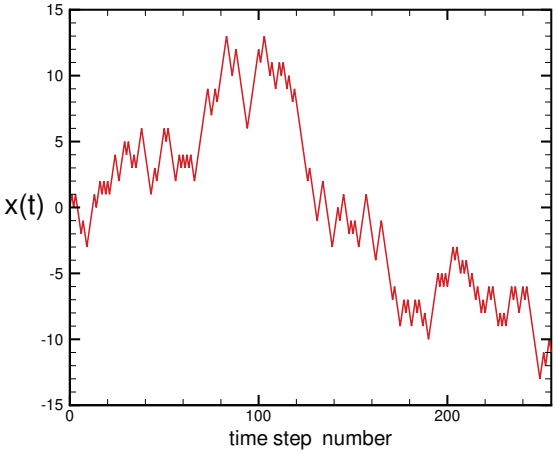

Figure 4.21: *Profile of a random walker with* $N = 256$ *steps.* x *denotes the walker distance from the origin. The walker has been initially placed at* $x = 0$.

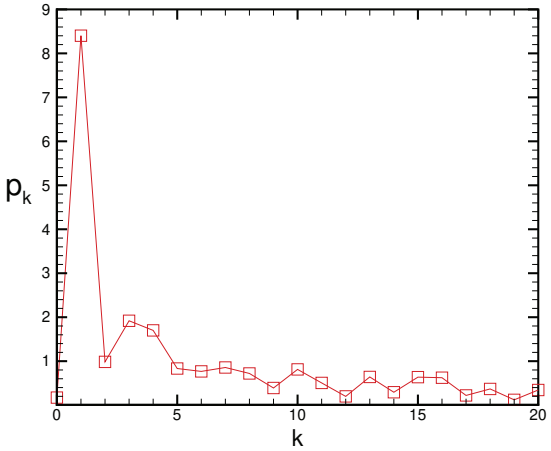

Figure 4.22: *Power spectrum for the random walk profile in figure (4.21).*

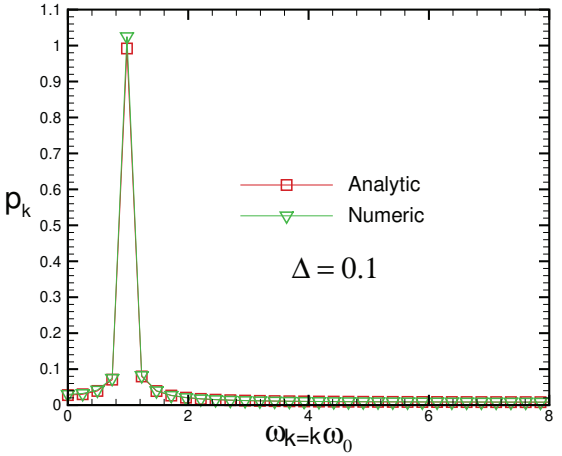

Figure 4.23: *Computed power spectrum of the second oscillator with sampling time $\Delta = 0.1$ and $N_{data} = 256$ points for a $N = 2$ system of coupled oscillators with fixed boundary condition. RK2 algorithm has been used.*

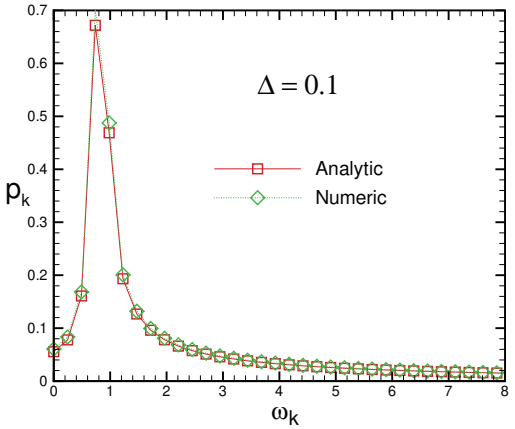

Figure 4.24: *Computed power spectrum of the fifth oscillator with sampling time* $\Delta = 0.1$ *and* $N_{data} = 256$ *points for a* $N = 10$ *system of coupled oscillators with fixed boundary condition prepared in the third mode.*

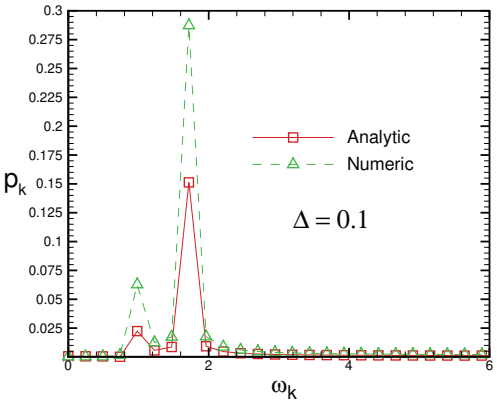

Figure 4.25: *Computed power spectrum of the second oscillator with sampling time* $\Delta = 0.1$ *and* $N_{data} = 256$ *points for a* $N = 2$ *system of coupled oscillators with fixed boundary condition and random initial condition.*

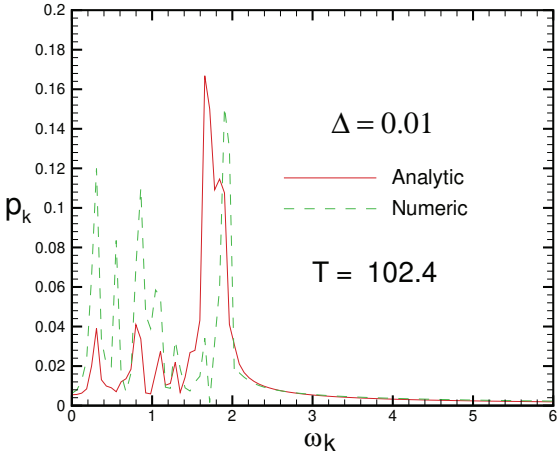

Figure 4.26: *Computed power spectrum of the second oscillator with sampling time* $\Delta = 0.01$ *and* $N_{data} = 2560$ *points for a* $N = 10$ *system of coupled oscillators with fixed boundary condition and random initial condition.*

spectrum $P(\omega)$. In figure (4.27), we depict the power spectrum, each corresponding to a different sample of initial conditions. The peak amplitudes depend on initial conditions. For a sufficient number of initial conditions, we expect to reveal all the peaks (normal modes).

4.7 Continuum wave equation

It is now the appropriate time to obtain the continuum longitudinal wave equation from our discrete chain of oscillators. Suppose we increase the number of oscillators and, at the same time, reduce their mass in such a manner that the mass density remains constant. Remind you that the equilibrium distance between oscillators is a, and hence the average density becomes $\rho = \frac{m}{a}$. Now let us increase the number of oscillators N to infinity, decrease their mass m and their equilibrium distance a toward zero such that ρ and chain length $L = (N + 1)a$ remain constant. In this limit, the distance of oscillator j from the left end of the chain can be shown by the continuum

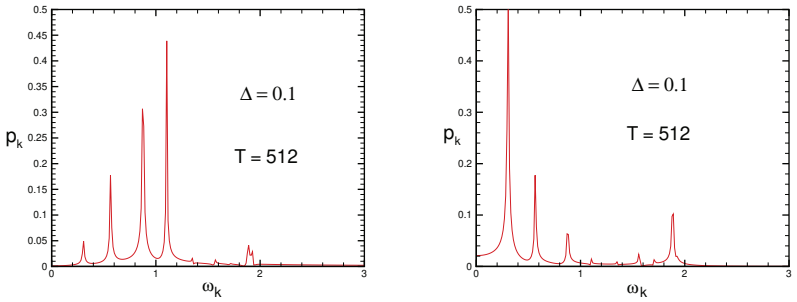

Figure 4.27: *Power spectrum of the second particle in a $N = 10$ particle system for three samples. Sampling time T has been* 512.

variable $x = ja$. We recall the Newton equation for oscillator j:

$$\frac{d^2u_j(t)}{dt^2} = \frac{ka^2}{ma^2}[u_{j+1}(t) + u_{j-1}(t) - 2u_j(t)] \tag{4.84}$$

If we denote $u_j(t)$ by $u(x,t)$, we can recast the Newton equation for the jth oscillator as follows:

$$\frac{\partial^2u(x,t)}{\partial t^2} = \frac{ka^2}{m}\frac{[u(x+a,t) + u(x-at) - 2u(x,t)]}{a^2} \tag{4.85}$$

The Taylor series expansion up to the second term yields:

$$u(x \pm a, t) = u(x,t) \pm a\frac{\partial u(x,t)}{\partial x} + \frac{a^2}{2}\frac{\partial^2 u(x,t)}{\partial x^2} + \cdots \tag{4.86}$$

Inserting expansion (4.86) in (4.85) gives:

$$\frac{\partial^2u(x,t)}{\partial t^2} = \frac{ka^2}{m}\frac{\partial^2u(x,t)}{\partial x^2} \tag{4.87}$$

The coefficient $\frac{ka^2}{m}$ can be written as $\frac{\frac{ka}{a}}{\frac{m}{a}}$. Its denominator is nothing but ρ. In the numerator, we should notice that k goes to infinity such that ka becomes a constant T, which is the force per unit length of the system. The coefficient $\frac{ka^2}{m}$ thus becomes a constant c^2 where c has a velocity dimension. Eventually, in the continuum limit, the equation of motion turns into the d'Alembert wave equation:

$$\frac{\partial^2u(x,t)}{\partial t^2} = c^2\frac{\partial^2u(x,t)}{\partial x^2} \tag{4.88}$$

Let us finish this chapter and invite you to see the numerical schemes for solving the continuum wave equation in the next chapter.

4.8 Problems

Problem 4.1 For the linear mass-spring chain with free boundary conditions, obtain the complex particle amplitudes a_j and compare your results with the fixed boundary conditions discussed in the text.

Problem 4.2 For the linear mass-spring chain with periodic boundary conditions, obtain the complex particle amplitudes a_j and compare your results with the fixed and free boundary conditions results.

Problem 4.3 Obtain the modal frequencies and complex amplitudes for the linear mass-spring chain with one free and one fixed boundary condition. Compare your results with the other boundary conditions. Which boundary condition has the smallest lowest mode? Which boundary condition has the largest lowest mode?

Problem 4.4 We want to simulate the wave reflection in a coupled system of mass-spring.

(a) Choose free boundary conditions and assume the initial condition to be a pulse of the form, $u_1 = 0.2, u_2 = 0.6, u_3 = 1.0, u_4 = 0.6, u_5 = 0.2$, and all other $u_j = 0$. After the pulse reaches the right end, what is the phase of the reflected pulse? are the displacements in the reflected pulse in the same direction as the incoming pulse (a phase shift of zero degrees) or the opposite direction (a phase shift of 180 degrees)? What happens for fixed boundary conditions? Choose N to be as large as possible so that it is easy to distinguish the incident and reflected waves.

(b) Change the boundary condition into periodic. Do you see any reflection? Sketch the system shape at various times.

(c) Change the boundary conditions into fixed and produce a Gaussian-like pulse in the middle of the system. Verify that two travelling waves emerge (one left-moving and the other right-moving).

Problem 4.5 Write a general programme that gives you the N normal modes of a linear chain of mass-spring for arbitrary boundary conditions.

Problem 4.6 Consider the forced mass-spring chain with periodic boundary conditions. Assume a sinusoidal force is exerted on the first mass so that it moves in time as $u_1(t) = \cos \omega t$.

(a) Numerically solve the system equations and show that in the steady-state a travelling wave is established.

(b) verify that all masses oscillate with the same amplitude but different phases.)

(c) What is the phase difference between the first and jth masses?

Problem 4.7 Consider a defective linear mass-spring system. In the simplest case, one of the masses, mass s, has a different mass $m_s = \lambda m$. Note that λ can be greater or smaller than unity.

(a) Write down the equations of motion and try to solve them analytically. Note that you can simply obtain the normal modes via Eq.(4.19). In fact we have: $\omega_i = \sqrt{\frac{D_{ii}}{m_i}}$. However, it would be a difficult task to analytically obtain the mode shape that is the amplitudes and phases of each mass.

(b) Numerically find the normal modes by diagonalisation of matrix K in Eq.(4.14) and discuss their dependence on disorder parameter λ.

(c) Try to see if you observe a *localisation* transition in the system when you increase λ. Localisation means a state in which only a few particles have notable amplitudes, whereas the other ones are essentially at rest. You should be able to find the transition frequency ω_c beyond which localisation occurs.

(d) Another type of disorder is associated with spring constants. Assume all the masses are identical this time, but one of the springs, spring s, has a different stiffness constant. Take its stiffness as $k' = \lambda k$, try to find the normal modes and their shape, and compare your findings with those found in part (b) and the normal mass-spring system. Consider fixed boundary conditions.

Problem 4.8 Suppose there is damping in our mass-spring system (equal masses and equal springs). Take the damping coefficient per mass as $\gamma = 2\beta$ for all the oscillators.

(a) Do we have the concept of mode in this case? In other words, is there a common frequency at which all the N masses perform a damped harmonic oscillation?

(b) What occurs if the masses or spring constants become different? C) Numerically find the normal modes of a damped mass-spring system having N masses.

Problem 4.9 Consider an undamped mass-spring system with N equal masses and fixed boundary conditions (spring constants are equal). A sinusoidal force with frequency ω and amplitude F_0 drives the mass n of the system. Numerically solve the equations of motion and plot the amplitude dependence of mass n versus driving frequency ω. You should observe N peaks at the locations of normal modes. These peaks correspond to resonance phenomena. Besides the peaks, you should observe some frequencies at which the amplitude becomes very small (theoretically zero)! These are anti-resonance frequencies. Give a physical interpretation of them. For more details refer to (Somayyeh Belbasi and Joe, 2014).

Problem 4.10 Consider a disordered mass-spring chain with equal spring constants and one mass impurity say mass n. Take $m_n = \lambda m$. Assume the system is sinusoidally driven on one of its masses with frequency ω and amplitude F_0. The input power given to the system by the driving force at time t is the product of driving force $F(t) = F_0 \cos\omega t$ times the $v_j(t)$ where j is the mass number on which the driving force is acted.

(a) Numerically solve the system equations of motion using the programme `ForcedCoupledOscillators` and obtain the average input power (over the period of the driving force) $\bar{P}_j$.

(b) Sketch $\bar{P}_j$ versus j for a given value of the disorder factor λ. For which j the input power is minimum? For which j the input power is maximum?

(c) Take $j = n$ i.e., the driving force is exerted on the defective particle. For what value of ω is the average input power maximum or minimum? Consider fixed boundary conditions.

Problem 4.11 Consider problem ten with the last mass N connected by a spring to a large mass $M = 10m$. Moreover, assume that $n = \frac{N+1}{2}$ (take N odd) and that the driving force is exerted on the first particle ($j = 1$). Solve the problem numerically and find the value of λ at which the average input power is minimised.

Problem 4.12 Consider a coupled system of mass-spring in two dimensions. Consider a rectangular geometry with fixed boundary conditions at all sides (endmost springs are connected to rigid walls). For simplicity, take all the masses to be identical to each other. Take the spring constants identical as well.

(a) Write the system of equations in the harmonic approximation.

(b) Obtain the modal frequencies and determine the lowest mode in terms of m, k and N where N is the number of masses per row (column). If you cannot proceed analytically, do not worry! Your PC can help you. Simply try to turn the mode problem into a matrix eigenvalue one and then find the modes by the method you learnt in the text.

(c) Discuss two types of polarisation: transverse and longitudinal.

Chapter 5

Partial differential equations: parabolic type

5.1 Foundation and classification

So far, we have only dealt with those parts of physics which are mathematically formulated by ordinary differential equations (ODEs). However, we all know that physics is mainly involved with partial differential equations (PDEs). Maxwell equations in electrodynamics, wave equation in acoustics and optics, Schrödinger equation in quantum mechanics, diffusion equation in statistical physics, heat transfer equation in thermodynamics, and Navier-Stocks equation in fluid mechanics are well-known examples of PDEs being the mathematical milestones. We assume the readers are familiar with the underlying physics of these equations and intend to cover the standard techniques developed for their numerical solutions. For completeness, we first try to briefly review the required mathematics.

5.1.1 classification scheme

We restrict ourselves to linear PDEs, which are ubiquitous in physics and engineering. As a further restriction, we consider only second-

order PDEs in two independent variables x and y. The generalisation to more independent variables is normally straightforward. The most general second-order PDE for the unknown real function $\phi(x, y)$ can be written as follows:

$$a\frac{\partial^2\phi(x,y)}{\partial x^2} + b\frac{\partial^2\phi(x,y)}{\partial x\partial y} + c\frac{\partial^2\phi(x,y)}{\partial y^2} + d\frac{\partial\phi(x,y)}{\partial x} +$$

$$e\frac{\partial\phi(x,y)}{\partial y} + f\phi(x,y) + g = 0 \tag{5.1}$$

Extension to higher orders or more independent variables is straightforward. The real coefficients a, b, c, d and e can depend, in principle, on x and y. The classification scheme is given according to the sign of the discriminant $\Delta = b^2 - 4ac$. The equation is hyperbolic, parabolic, and elliptic if Δ is positive, zero, and negative correspondingly (Myint-U and Debnath, 2007). Note that this classification applies to quasilinear second-order PDEs in two independent variables in which the last four terms in the left-hand side of (5.1) are generally written as $f(x, y, \phi, \frac{\partial\phi}{\partial x}, \frac{\partial\phi}{\partial y})$. Equations with more than two independent variables may not feat neatly into the above classification scheme. Nevertheless, the concepts of parabolic, elliptic, and hyperbolic can be extended to such types of PDEs. For more details, you may consult (Myint-U and Debnath, 2007). Notice the variables x, and y do not necessarily signify spatial coordinates. The nature of these variables depends on the problem under consideration. Some concrete examples from physics include the two-dimensional Poisson equation (elliptical):

$$\frac{\partial^2\phi(x,y)}{\partial x^2} + \frac{\partial^2\phi(x,y)}{\partial y^2} = -\frac{\rho(x,y)}{\epsilon_0} \tag{5.2}$$

which in the special case of zero charge density in the space reduces to the Laplace equation. A prototype example of a parabolic equation is the one- dimensional heat or heat diffusion equation:

$$\frac{\partial T(x,t)}{\partial t} = D\frac{\partial^2 T(x,t)}{\partial x^2} \tag{5.3}$$

where $T(x, t)$ is the temperature of a 1D rod at position x at time t, $D = \frac{k}{\rho c_p}$ the thermal diffusivity, k the thermal conductivity, ρ the mass density and c_p the specific heat capacity at constant pressure. For derivation see (T. L. Bergman and Dewitt, 2011). When dealing

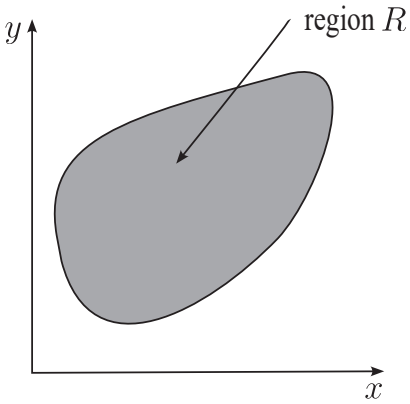

Figure 5.1: *Solution domain for a linear two-dimensional partial differential equation.*

with the diffusion phenomenon, $T(x,t)$ is replaced with $n(x,t)$, which denotes the number density of diffusive particles at position x and time t. Throughout the book, we interchangeably use the words heat and diffusion for this equation. A typical example from hyperbolic PDEs is the ubiquitous one-dimensional d'Alembert wave equation:

$$\frac{\partial^2 \psi(x,t)}{\partial t^2} = c^2 \frac{\partial^2 \psi(x,t)}{\partial x^2} \tag{5.4}$$

The solution of (5.1) is required in a region R in the $x-y$ plane. The value of ϕ or its partial derivative (or a combination of them) should be prescribed on the boundary of R for being able to solve the problem within the solution domain. See figure (5.1) for illustration. Contrary to ODEs, there is no general existence and uniqueness theorem for PDEs. Sometimes there is no solution for a given boundary condition at all. For the hyperbolic equations, the region is semi-infinite, whereas, for the elliptical types, the region is closed and finite. There is a Cauchy problem in the context of PDEs, which specifies the conditions under which boundary conditions there will exist a unique solution. For details, consult (Myint-U and Debnath, 2007) and the reference therein. In this book, we consider those problems for which unique solutions exist.

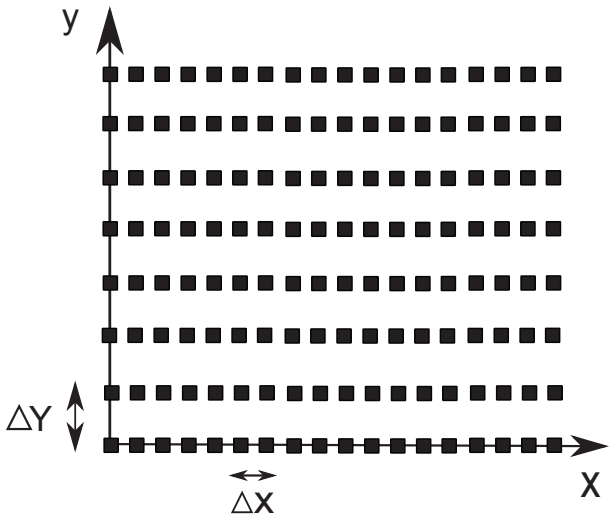

Figure 5.2: *Discretization of a two dimensional space with grids ΔX and ΔY which can be unequal in a general case.*

5.1.2 Numerical solution

To numerically solve a PDE, one should replace the partial derivatives with finite difference relations obtained from Taylor series expansion. To illustrate the objective, consider a two-dimensional rectangular domain in the $x - y$ plane within which we wish to solve a PDE. The rectangular domain is divided into equal increments in the x and y directions. The increments in the x and y directions are denoted by Δx and Δy, respectively. In general, they are not equal i.e.; $\Delta x \neq \Delta y$. Figure (5.2) illustrates the situation. We show the number of grids in the x and y directions by N and M. Counters i and j specify the grid numbers along the x and y directions. It is a matter of choice to start these counters from zero or one.

5.1.3 1st order partial differential equation

Before considering the numerical schemes for solving the above second-order PDEs, let us pause and discuss some of the first-order PDEs in physics and engineering, which occur occasionally but are normally ignored in textbooks. To give an example, consider the nonlinear one-

dimensional Burger's equation in fluid mechanics:

$$\frac{\partial u(x,t)}{\partial t} = -u\frac{\partial u(x,t)}{\partial x} \tag{5.5}$$

or the 1D advection equation in heat transfer (Garcia, 1999):

$$\frac{\partial u(x,t)}{\partial t} = -c\frac{\partial u(x,t)}{\partial x} \tag{5.6}$$

Another important domain where PDEs are first order is fluid mechanics. As an example, consider the 2D motion of an inviscid and incompressible fluid in the absence of external forces. In this case, the Navier-Stocks (NS) equation becomes (E. Guyon and Mitescu, 2001):

$$\rho[\frac{\partial \mathbf{v}}{\partial t} + (\mathbf{v}.\nabla)\mathbf{v}] = -\nabla p(x,y) \tag{5.7}$$

where $\mathbf{v} = (v_x, v_y)$ is the fluid velocity field, $\rho(x,y)$ the local density, $p(x,y,t)$ is the pressure field and η is the viscosity coefficient. When the flow is steady, the NS equation gives the following equations for velocity components v_x and v_y:

$$\rho v_x\frac{\partial v_x}{\partial x} + \rho v_y\frac{\partial v_x}{\partial y} + \frac{\partial p(x,y)}{\partial x} = 0 \tag{5.8}$$

$$\rho v_x\frac{\partial v_y}{\partial x} + \rho v_y\frac{\partial v_y}{\partial y} + \frac{\partial p(x,y)}{\partial y} = 0 \tag{5.9}$$

The incompressibility condition $\nabla.\mathbf{v} = 0$ gives the third equation:

$$\frac{\partial v_x}{\partial x} + \frac{\partial v_y}{\partial y} = 0 \tag{5.10}$$

In fact, equations (5.8) to (5.10) compose a set of first-order nonlinear PDEs. Despite their nonlinearity, numerical schemes are not crucially sensitive to this feature, and many techniques developed for linear PDEs can also be applied to these nonlinear PDEs.

5.2 2nd order prototype parabolic PDE: diffusion equation

Let us begin with parabolic PDEs and apply the general numerical scheme to this type of equation. The simplest, and perhaps the most

important, parabolic equation is the heat (diffusion equation) (5.3) in which $T(x,t)$ is the temperature of a rod at position x and time t. The constant α is the thermal diffusivity coefficient.

$$\frac{\partial T(x,t)}{\partial t} = D\frac{\partial^2 T(x,t)}{\partial x^2} \tag{5.11}$$

Since one of the independent variables is time, the problem lies in the category of *initial value* problems. For simplicity, suppose we have a one- dimensional rod of length L extended from $x = -\frac{L}{2}$ to $x = +\frac{L}{2}$. The solution domain is a semi-infinite region in the $t - x$ plane. To solve the problem, the boundary conditions at $x = \pm\frac{L}{2}$ should be specified. The most general boundary condition is the Robin one:

$$aT(-\frac{L}{2},t) + b\frac{\partial T}{\partial x}(-\frac{L}{2},t) = \alpha \tag{5.12}$$

$$cT(+\frac{L}{2},t) + d\frac{\partial T}{\partial x}(+\frac{L}{2},t) = \beta \tag{5.13}$$

where constants a, b, c, d, α and β are given. If at a boundary only the temperature value is specified, we have *Dirichlet* boundary condition. On the other hand, if only the spatial derivative of T is specified, we have *Neumann* boundary condition. If both of them are involved, we have Robin's (mixed) boundary condition. Besides the mentioned boundary conditions, we also have the so-called *periodic* boundary condition, which is very important in solid-state physics. In this boundary condition, we have:

$$T(-\frac{L}{2},t) = T(\frac{L}{2},t); \quad \frac{\partial T}{\partial x}(-\frac{L}{2},t) = \frac{\partial T}{\partial x}(\frac{L}{2},t) \tag{5.14}$$

I normally avoid solving an equation numerically before attempting to show some analytical calculations. Hopefully, in the present problem, there are some cases, especially in one dimension, that we can proceed analytically. We remind you that the heat (diffusion) equation is divided into two categories. In the first type, the space is unrestricted, and in the second type, it is restricted. Our first example lies in the first category.

Problem: *The initial condition for an infinite 1D heat flow problem is $T(x,0) = \delta(x)$ where $\delta(x)$ is the Dirac function. Find $T(x,t)$.*

We solve the problem by using of Fourier transform. Let us show the Fourier transform of $T(x,t)$ with respect to the spatial coordinate x by $\tilde{T}(q,t)$:

$$\tilde{T}(q,t) = \int_{-\infty}^{+\infty} T(x,t)e^{-iqx}dx \tag{5.15}$$

The inverse Fourier transform is given by:

$$T(x,t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} \tilde{T}(q,t)e^{iqx}dq \tag{5.16}$$

Putting (5.16) in (5.11) we find:

$$\frac{\partial \tilde{T}}{\partial t} = -Dq^2\tilde{T} \tag{5.17}$$

Solving (5.17) gives:

$$\tilde{T}(q,t) = Ce^{-Dq^2t} \tag{5.18}$$

The constant C is found by taking the Fourier transform from the initial condition. On one hand, we have from (5.18) $\tilde{T}(q,0) = C$. On the other, we have:

$$\tilde{T}(q,0) = \int_{-\infty}^{+\infty} T(x,0)e^{-iqx}dx = \int_{-\infty}^{+\infty} \delta(x)e^{-iqx}dx = 1 \tag{5.19}$$

A comparison gives $C = 1$. By the inverse Fourier relation, we have:

$$T(x,t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty} e^{-Dq^2t}e^{iqx}dq \tag{5.20}$$

Using the identity $Dtq^2 - iqx = (\sqrt{Dt}q - \frac{ix}{2\sqrt{Dt}})^2 + \frac{x^2}{4Dt}$ and a complex variable integration we find (Arfken and Weber, 2005):

$$T(x,t) = \frac{1}{\sqrt{4\pi Dt}}e^{-\frac{x^2}{4Dt}} \tag{5.21}$$

Equation (5.21) shows that for a given time t temperature profile $T(x,t)$ is a Gaussian packet in x which spreads in time with a width

proportional to $\sqrt{t}$. When the initial condition $T(x,0)$ is an arbitrary function we have $\tilde{T}(q,0) = \int_{-\infty}^{+\infty} T(x,0)e^{-iqx}dx$ and we find:

$$T(x,t) = \frac{1}{2\pi}\int_{-\infty}^{+\infty}\int_{-\infty}^{+\infty} T(x',0)e^{-iqx'}dx'e^{-Dq^2t}e^{iqx}dq \tag{5.22}$$

We can write the above integral in the form

$$T(x,t) = \int_{-\infty}^{+\infty} T(x',0)T_G(x-x',t)dx' \tag{5.23}$$

where

$$T_G = \frac{1}{2\pi}\int_{-\infty}^{+\infty} dqe^{i(x-x')q-Dq^2t} = \frac{1}{\sqrt{4\pi Dt}}e^{-\frac{(x-x')^2}{4Dt}} \tag{5.24}$$

is the Green function of the problem (Hassani, 2013). Let us consider the three-dimensional case where the heat equation takes the form:

$$\frac{\partial T(\mathbf{r},t)}{\partial t} = D\nabla^2 T(\mathbf{r},t) \tag{5.25}$$

Sometimes the three-dimensional diffusion equation can be effectively mapped into a one-dimensional form when the medium has sufficient symmetry. As an example, consider heat propagation in a medium with spherical symmetry around the origin. We expect $T(\mathbf{r},t)$ to depend on r only. In this case the Laplacian operator ∇^2 becomes $\frac{\partial^2}{\partial r^2} + \frac{2}{r}\frac{\partial}{\partial r}$ and the three-dimensional heat equation (5.25) becomes:

$$\frac{\partial T(r,t)}{\partial t} = D[\frac{\partial^2 T(r,t)}{\partial r^2} + \frac{2}{r}\frac{\partial T(r,t)}{\partial r}] \tag{5.26}$$

We now change variable to $S(r,t) = rT(r,t)$ (Arfken and Weber, 2005). We can simply verify

$$\frac{\partial^2 T(r,t)}{\partial r^2} = \frac{1}{r}\frac{\partial^2 S(r,t)}{\partial r^2} - \frac{2}{r^2}\frac{\partial S(r,t)}{\partial r} + \frac{2S(r,t)}{r^3} \tag{5.27}$$

Replacing (5.27) into (5.26) we find:

$$\frac{\partial S(r,t)}{\partial t} = D\frac{\partial^2 S(r,t)}{\partial r^2} \tag{5.28}$$

which is the desired one-dimensional heat (diffusion) equation. If we have cylindrical symmetry, T will be only a function of $\rho = \sqrt{x^2 + y^2}$ and time. In this case, the Laplacian operator ∇^2 becomes $\frac{\partial^2}{\partial \rho^2} + \frac{1}{\rho}\frac{\partial}{\partial \rho}$ and the heat equation become

$$\frac{\partial T(\rho,t)}{\partial t} = D[\frac{\partial^2 T(\rho,t)}{\partial \rho^2} + \frac{1}{\rho}\frac{\partial T(\rho,t)}{\partial \rho}] \tag{5.29}$$

It seems we have had enough analytical results. Let us try to proceed numerically.

5.3 Numerical solution of 1D heat eq.

We shall now attempt to devise a scheme for the numerical solution of the 1D heat (diffusion) equation. We discretise the one-dimensional space of length L into equally spaced grids with distance Δx. Let $i = 0$ denote the grid point located at the left boundary $x = -\frac{L}{2}$ and $i = N$ the grid point at the right boundary $x = +\frac{L}{2}$. The location of grid point i is at $x_i = i\Delta x - \frac{L}{2}$. Note that we have: $\Delta x = \frac{L}{N}$. Also, let τ denote the temporal grid as usual, and counter n indicates the temporal step $t_n = n\tau$. For notational convenience, we show $T(x_i, t_n)$ by the shorthand T_i^n. The next step is to adopt a finite difference scheme for the partial derivatives. We implement *forward* scheme for time discretisation and a centered scheme for the second-order spatial derivative as follows:

$$\frac{\partial T}{\partial t}(x_i,t_n) \approx \frac{T(x_i,t_n+\tau) - T(x_i,\tau)}{\tau} = \frac{T_i^{n+1} - T_i^n}{\tau} \tag{5.30}$$

$$\frac{\partial^2 T}{\partial x^2}(x_i,t_n) \approx \frac{T(x_i+\Delta x,t_n) + T(x_i-\Delta x,t_n) - 2T(x_i,t_n)}{(\Delta x)^2}$$

$$= \frac{T_{i+1}^n + T_{i-1}^n - 2T_i^n}{(\Delta x)^2} \tag{5.31}$$

The method is known as *forward time centred space* or shortly FTCS in the literature. In this scheme, the 1D heat PDE turns into the following finite difference equation:

$$\frac{T_i^{n+1} - T_i^n}{\tau} = D\frac{T_{i+1}^n + T_{i-1}^n - 2T_i^n}{(\Delta x)^2} \tag{5.32}$$

A simple algebra gives:

$$T_i^{n+1} = T_i^n + \frac{D\tau}{(\Delta x)^2}[T_{i+1}^n + T_{i-1}^n - 2T_i^n] \tag{5.33}$$

Equation (5.33) allows us to find the temperature in the new time step $n+1$ in terms of temperatures at the current step n. Note that FTCS is an explicit method because the next step values are determined solely by the current ones. The method is sometimes called *marching method* because we march forward in time. All we need to know is the initial temperature $T(x,0)$. To numerically solve the problem, one should be able to transform $T(x,0)$ into a discretised form i.e.; to determine T_i^0 for $i = 0, 1, \cdots, N$. A frequently used initial condition is $T(x,0) = \delta(x)$. The simplest way of discretising the Dirac delta function is to take its value zero at all the grid points except the central one $i = \frac{N}{2}$ which takes the value $\frac{1}{\Delta x}$:

$$\delta_i = 0;\ i = 0, \cdots, \frac{N}{2} - 1, \frac{N}{2} + 1, \cdots, N;\quad \delta_{\frac{N}{2}} = \frac{1}{\Delta x} \tag{5.34}$$

With this choice, we have:

$$\int_{-\infty}^{\infty} \delta(x)dx \approx \sum_{i=0}^{N} \delta_i \Delta x = 1 \tag{5.35}$$

Let us now solve a problem:

Consider a rod of length L with its left and right ends kept fixed at temperatures T_L and T_R. Take the initial condition as $\delta(x)$. Find the temperature profile at time t. Before solving the problem numerically, let us try to solve it analytically. The boundary conditions are Dirichlet at both ends:

$$T(-\frac{L}{2}, t) = T_L;\quad T(+\frac{L}{2}, t) = T_R. \tag{5.36}$$

Although we can find the explicit form of $T(x,t)$ by the method of separation of variables or Green's function, we do not intend to do so. You may refer to mathematical physics textbooks such as (Myint-U and Debnath, 2007),(Arfken and Weber, 2005) or (Hassani, 2013) for analytic details. Here we focus our attention on the long-time

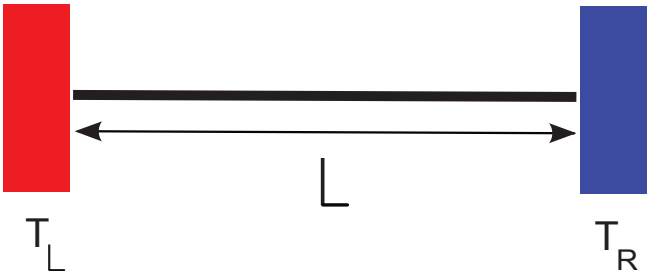

Figure 5.3: *A bar is in contact with two heat reserviours of constant temperature T_L and T_R.*

steady-state limit $T_s(x)$. In the steady-state, limit the heat transfer equation (5.3) becomes:

$$\frac{d^2T_s(x)}{dx^2} = 0 \tag{5.37}$$

It simply turns out $T_s(x) = a + bx$. Implying the boundary conditions in (5.37) we find the constants a and b as follows:

$$a = \frac{T_L + T_R}{2}; \quad b = \frac{T_R - T_L}{L} \tag{5.38}$$

Therefore, we find:

$$T_s(x) = \frac{T_L + T_R}{2} + \frac{T_R - T_L}{L}x \tag{5.39}$$

Now let us see if our computer programme can reproduce this behaviour. The programme `DiffusionFTCS` (see Appendix 5.A for details) numerically solves the diffusion equation (5.3) by the FTCS method. The Dirichlet boundary conditions imply $T_0^0 = T_L$ and $T_N^0 = T_R$. Figure (5.4) shows the temperature profile for some chosen time steps. We have set L and α to unity. The values of temperature at boundaries are $T_L = 10, T_R = 30$. The evolution of the peaked delta function into a smooth line is seen in the graph. In agreement with the analytical results for the steady state, we see that after sufficient time the profile becomes linear between the boundary values. Hopefully, the FTCS has successfully managed to solve the problem. However, care should be taken in choosing the time step τ. Figure (5.5) shows

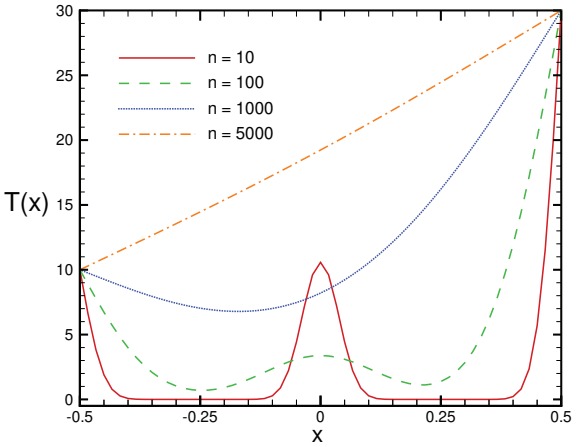

Figure 5.4: *Temperature profiles at some time steps. The number of grid points is $N = 61$ with a time step $\tau = 0.67 \times 10^{-4}$. The values of temperature at boundaries are $T_L = 10, T_R = 30$.*

the temperature profile at $n = 100$ for the same value of N but with a double larger τ. As you see, the solution has become unstable. In fact, the FTCS is not stable for all values of Δx and τ. It can be shown that for τ larger than a characteristic time $\tau^* = \frac{(\Delta x)^2}{2\alpha}$, the FTCS algorithm becomes unstable. We will later discuss the instability conditions in more detail. Let us now consider the same problem, but this time under the Neumann boundary condition. For simplicity, consider:

$$\frac{\partial T}{\partial x}(-\frac{L}{2}, t) = \alpha; \ \frac{\partial T}{\partial x}(\frac{L}{2}, t) = \beta \tag{5.40}$$

Note that we can computed $T_1^{n+1}, \cdots, T_{N-1}^{n+1}$ from (5.33) but we cannot compute T_0^{n+1} and T_N^{n+1} directly from the recursion relation (5.33) because the undefined quantities T_{-1}^n and T_{N+1}^n will appear. To compute these two boundary values, we discretise the space derivatives in the left boundary condition by the forward scheme and at the right

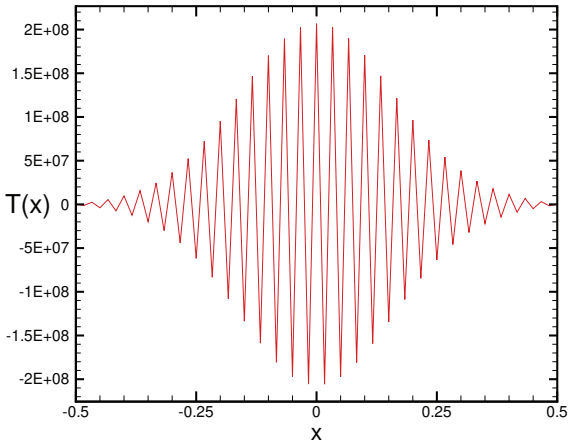

Figure 5.5: *Temperature profiles at $n = 100$ for the same problem. The number of grid points is $N = 61$ with a time step $\tau = 1.47 \times 10^{-4}$.*

boundary condition in the backward scheme:

$$\frac{T_1^n - T_0^n}{\Delta x} = \alpha; \quad \frac{T_N^n - T_{N-1}^n}{\Delta x} = \beta \tag{5.41}$$

which implies:

$$T_0^n = T_1^n - \alpha \Delta x; \quad T_N^n = T_{N-1}^n + \beta \Delta x \quad n = 0, 1, \cdots \tag{5.42}$$

This allows us to compute T_0^{n+1} and T_N^{n+1} once T_1^{n+1} and T_{N-1}^{n+1} are computed. Figure (5.6) exhibits the temperature profile at some instants of time with the Neumann boundary condition. The constant temperature gradient values have been set at $\alpha = 0$ for the left boundary and $\beta = 1$ for the right one. According to the Fourier law, there is a heat influx $J_{in} = -k\frac{\partial T(\frac{L}{2},t)}{\partial x} = -k\beta = -k < 0$ to the system at the right boundary which causes the system temperature becomes higher at right than its left-hand side. Figure (5.7) exhibits the same diagram but this time for $\beta = -2$. In this case, we have a heat outflux $J_{out} = -k\frac{\partial T(\frac{L}{2},t)}{\partial x} = -k\beta = +2k > 0$ at the system right boundary and it is expected that in the long-time regime, the system temperature at the left side becomes higher than its right side.

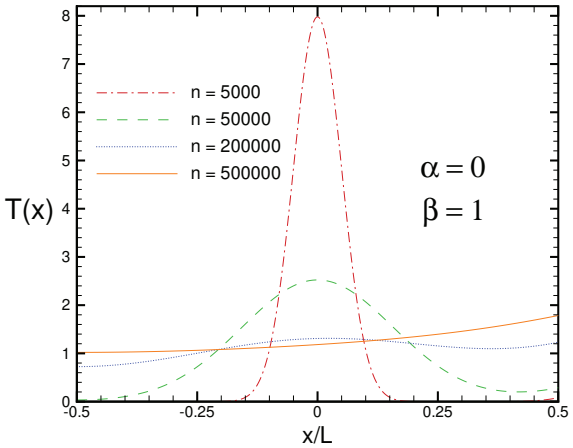

Figure 5.6: *Temperature profiles at some time steps for the Neumann boundary condition. The number of grid points is $N = 1000$ with a time step $\tau = 0.25 \times 10^{-6}$. The constant temperature gradient values have been $\alpha = 0$ for the left boundary and $\beta = 1$ for the right one.*

5.4 Other schemes for solving heat eq.

In previous sections, we saw that the FTCS scheme shows a satisfactory performance in the numerical solution of the heat equation if the temporal and spatial steps are chosen properly. Despite this algorithm being stable and quite simple, it is inefficient for more complicated equations. In the case of parabolic equations, there are more feasible algorithms than there are for hyperbolic ones. One of the best ones being highly efficient is the *Crank-Nicholson* algorithm (Crank and Nicolson, 1947). This algorithm lies in the category of implicit algorithms. Before explaining the Crank-Nicholson (CN) algorithm, let us introduce an implicit scheme for solving the heat equation. If we express the second spatial derivative at time step t_{n+1} rather than t_n we find:

$$\frac{T_i^{n+1} - T_i^n}{\tau} = \alpha \frac{T_{i+1}^{n+1} + T_{i-1}^{n+1} - 2T_i^{n+1}}{(\Delta x)^2} \tag{5.43}$$

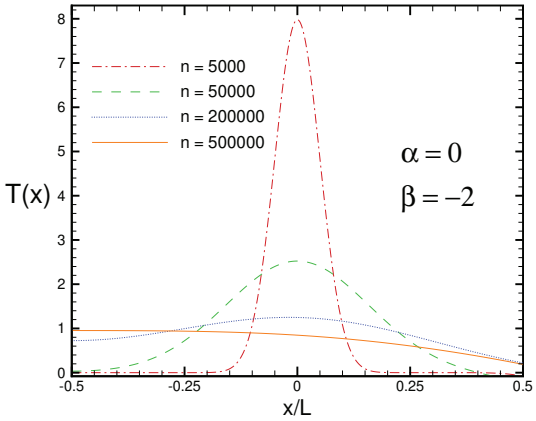

Figure 5.7: *Temperature profiles at some timesteps with Neumann boundary condition. Parameters values: number of grid points* $N = 1000$*, timestep* $\tau = 0.25 \times 10^{-6}$*,* $\alpha = 0$ *and* $\beta = -2$*.*

Defining $a = \frac{\alpha\tau}{(\Delta x)^2}$ we find:

$$(1+2a)T_i^{n+1} - aT_{i-1}^{n+1} - aT_{i+1}^{n+1} = T_i^n \tag{5.44}$$

Note that in (5.44) the range of i is $[1, N-1]$. Introducing the vector $T^{n\dagger} = (T_0^n, T_1^n, \cdots, T_N^n)$ equation (5.44) can be written in compact matrix form as follows:

$$AT^{n+1} = T^n \tag{5.45}$$

Given the boundary values T_0^n and T_N^n the tridiagonal matrix A turns out to be:

$$\begin{cases} a_{00} = a_{NN} = 1;\ a_{ii} = 1 + 2a \quad i = 1, \cdots, N-1, \\ a_{01} = 0;\ a_{ii+1} = -a \quad i = 1, \cdots, N-1, \\ a_{N,N-1} = 0;\ a_{i+1i} = -a \quad i = 0, \cdots, N-2. \end{cases} \tag{5.46}$$

Having T^n, one can exploit standard linear equations solvers to find T^{n+1} numerically. The most important aspect of an implicit algorithm is its high degree of stability. But be careful! Stability does not always

guarantee that the answer is correct. You can verify that the solution obtained by the above implicit method is not very accurate. To increase the accuracy, we replace the second spatial derivative with half explicit and half implicit and arrive at the so-called *Crank-Nicholson* algorithm.

$$\frac{T_i^{n+1}-T_i^n}{\tau}=\alpha\frac{T_{i+1}^{n+1}+T_{i-1}^{n+1}-2T_i^{n+1}+T_{i+1}^n+T_{i-1}^n-2T_i^n}{(2\Delta x)^2} \quad (5.47)$$

This time we define $a=\frac{\alpha\tau}{2(\Delta x)^2}$ and recast (5.47) as follows:

$$(1+2a)T_i^{n+1}-aT_{i-1}^{n+1}-aT_{i+1}^{n+1}=(1-2a)T_i^n+aT_{i-1}^n+aT_{i+1}^n \quad (5.48)$$

In matrix notation (5.48) becomes:

$$AT^{n+1}=BT^n \quad (5.49)$$

where tridiagonal matrix A is the same as in (5.45). Elements of B are obtained from A by substitution $a\rightarrow -a$. Having known temperature at time step t_n, we can simply evaluate the right-hand side vector BT^n and find the unknown temperature T^{n+1} by a standard linear solver subroutine. You may refer to *Numerical Recipe* (W. H. Press and Flannery, 2002) to see various subroutines, especially those designated for tridiagonal matrices. Finally, we discuss the *Dufort-Frankel* algorithm. This algorithm would be unstable when applied without precaution to the heat equation. In this algorithm, the first-order time derivative is discretised in a centred manner. The second-order spatial derivative is also discretised differently than FTCS and CN algorithms. More concisely, we have:

$$\frac{1}{2\tau}[T_i^{n+1}-T_i^{n-1}]=\frac{\alpha}{(\Delta x)^2}[T_{i+1}^n-(T_i^{n+1}+T_i^{n-1})+T_{i-1}^n] \quad (5.50)$$

Notice the term $2T_i^n$ in the FTCS scheme is replaced by $(T_i^{n+1}+T_i^{n-1})$. Defining $a=\frac{2\alpha\tau}{(\Delta x)^2}$ equation (5.50) can be written as follows:

$$T_i^{n+1}=\frac{1-a}{1+a}T_i^{n-1}+\frac{a}{1+a}[T_{i+1}^n+T_{i-1}^n] \quad (5.51)$$

The Dufort-Frankel is a two-step implicit algorithm. In matrix notation, it can be written as:

$$AT^{n+1}=BT^{n-1}+CT^n \quad (5.52)$$

Matrices B and C are tridiagonal. At the current timestep n, we know both T^{n-1} and T^n therefore, the right-hand side of (5.52) is known. You simply need to implement a linear equations solver to find T^{n+1}.

5.5 Diffusion equation with a source

Let us now return to the diffusion equation and consider its extension by adding a source term on the right-hand side of (5.3). Let $n(\mathbf{r}, t)$ specify the density of a diffusive quantity at point $\mathbf{r}$ and at time t. In a prototype example $n(\mathbf{r}, t)$ describes the neutron number density in a nuclear reaction. See Garcia for further details (Garcia, 1999). For simplicity, we assume there is a source term that is linearly proportional to the density function $n(\mathbf{r}, t)$. The inhomogeneous diffusion equation turns out to be:

$$\frac{\partial n(\mathbf{r}, t)}{\partial t} = D\nabla^2 n(\mathbf{r}, t) + Cn(\mathbf{r}, t) \tag{5.53}$$

where the positive constant D is the diffusion constant and the positive constant C is the creation rate for the neutrons (Garcia, 1999). To simplify the analysis consider the 1D version of the problem where x is restricted from $-\frac{L}{2}$ to $\frac{L}{2}$.

$$\frac{\partial n(x, t)}{\partial t} = D\frac{\partial^2 n(x, t)}{\partial x^2} + Cn(x, t) \tag{5.54}$$

For U^{235} we have $D \approx 10^5\ m^2/s$ and $C \approx 10^8\ s^{-1}$. All the neutrons reaching the boundaries will escape from the system therefore, we have the Dirichlet boundary condition:

$$n(-\frac{L}{2}, t) = n(\frac{L}{2}, t) = 0 \tag{5.55}$$

To proceed analytically, we use the method of separation of variables i.e.; we take $n(x, t) = X(x)T(t)$. Putting this assumption into (5.54) gives:

$$\frac{1}{T}\frac{dT}{dt} = \frac{D}{X}\frac{d^2X}{dx^2} + C \tag{5.56}$$

Both sides should be equal to a constant α. Therefore we find:

$$T(t) = T(0)e^{\alpha t};\quad \frac{d^2X}{dx^2} = \frac{\alpha - C}{D}X \tag{5.57}$$

According to the Dirichlet boundary conditions in (5.55) $\alpha - C$ should be negative and consequently:

$$X(x) = E \sin \sqrt{\frac{C-\alpha}{D}} x + F \cos \sqrt{\frac{C-\alpha}{D}} x \tag{5.58}$$

The boundary conditions imply $X(\frac{\pm L}{2}) = 0$ which give:

$$\begin{cases} E \sin \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} + F \cos \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} = 0, \\ -E \sin \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} + F \cos \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} = 0. \end{cases} \tag{5.59}$$

Equation (5.59) is a linear system of homogeneous equations, and to have a non-trivial solution, the determinant of its coefficient matrix should be zero. This gives:

$$2 \sin \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} \cos \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} = \sin \sqrt{\frac{C-\alpha}{D}} L = 0 \tag{5.60}$$

This implies $\sqrt{\frac{C-\alpha}{D}} L = j\pi$ with $j = 0, 1, \cdots$ which gives us the eigenvalues α_j as follows:

$$\alpha_j = C - D(\frac{j\pi}{L})^2; \quad j = 0, 1, \cdots \tag{5.61}$$

Equations (5.59) also imply $E \sin \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} = F \cos \sqrt{\frac{C-\alpha}{D}} \frac{L}{2} = 0$. Having in mind that $\sqrt{\frac{C-\alpha}{D}} L = j\pi$, we find:

$$E_j \sin \frac{j\pi}{2} = 0; \quad F_j \cos \frac{j\pi}{2} = 0. \tag{5.62}$$

From (5.62) we conclude that for even $j = 2k$ coefficient F_{2k} should be zero whereas for odd $j = 2k + 1$ coefficient E_{2k+1} should be zero. In summary, we obtain:

$$X_{2k}(x) = E_{2k} \sin \frac{2k\pi x}{L}; \; X_{2k+1}(x) = F_{2k+1} \cos \frac{(2k+1)\pi x}{L} \tag{5.63}$$

We can recast two equations in (5.63) into one equation as follows:

$$X_j(x) = B_j \sin \frac{j\pi(x + \frac{L}{2})}{L} \tag{5.64}$$

Therefore, the solution becomes:

$$n(x,t) = \sum_{j=1}^{\infty} B_j e^{\alpha_j t} \sin \frac{j\pi(x+\frac{L}{2})}{L} \tag{5.65}$$

In a long-time limit, the contribution of $\alpha_j < 0$ becomes negligible. For those positive $\alpha_j > 0$, the long-time behaviour is dominated by the largest one. According to (5.61) the largest α_j corresponds to $j = 1$. The sufficient condition to have a nonzero density in the long-time is that α_1 is positive. From (5.61) we find $\alpha_1 = C - D(\frac{\pi}{L})^2 > 0$, which gives the following condition for having a nonzero density, actually divergent, in the long-time limit:

$$L > L_c \tag{5.66}$$

where the critical length is $L_c = \pi\sqrt{\frac{D}{C}}$. In other words, if the system's length is smaller than L_c then the flux of neutrons at the boundaries will dampen out the neutron density. However, if $L > L_c$, the density of neutrons, and hence their energy will increase exponentially with dramatic consequences. Let us now solve the problem numerically. Applying FTSC to the problem, equation (5.54) is discretised in the following manner:

$$\frac{n_i^{n+1} - n_i^n}{\tau} = D\frac{n_{i+1}^n + n_{i-1}^n - 2n_i^n}{(\Delta x)^2} + Cn_i^n \tag{5.67}$$

or

$$n_i^{n+1} = (1 + C\tau)n_i^n + \frac{D\tau}{(\Delta x)^2}[n_{i+1}^n + n_{i-1}^n - 2n_i^n] \tag{5.68}$$

The programme `NeutronDiffusion` (see Appendix 5.B for details) numerically solves the diffusion equation (5.54) by the FTCS method. In the programme, we use a reduced system of units in which $C = D = 1$. We desire to obtain the space-averaged density of neutrons at time t that is $\bar{n}(t) = \int_{-\frac{L}{2}}^{+\frac{L}{2}} n(x,t)dx$. In our numerical scheme, the average neutron density becomes:

$$\bar{n}(t) = \int_{-\frac{L}{2}}^{+\frac{L}{2}} n(x,t)dx = \frac{1}{N+1}\sum_{i=0}^{N} n_i^n \tag{5.69}$$

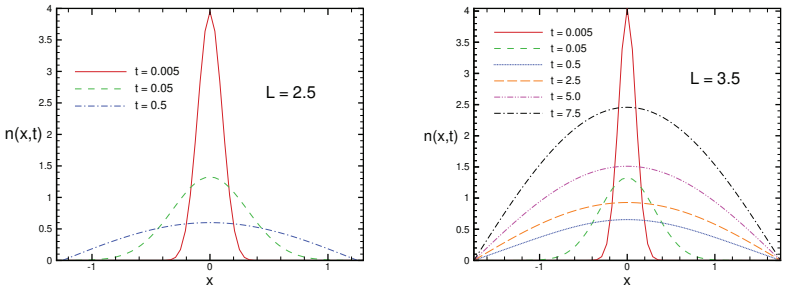

Figure 5.8: *Neutron density profiles at some time steps for a system length* $L = 2.5 < L_c$ *(left) and* $L = 3.5 > L_c$ *(right). The number of grid points is* $N = 61$ *with a timestep* $\tau = 0.5 \times 10^{-4}$.

Notice our choice of spatial grids: $i = 0$ corresponds to $x = -\frac{L}{2}$ and $i = N$ corresponds to $x = \frac{L}{2}$. The initial condition has been a delta function centred at $x = 0$. Figure (5.8) shows the density profile at some instant of time for the system length L smaller and larger than the critical length $L_c = 3.14$. When $L < L_c$ the density profile decreases with time. When $L > L_c$, in the early stages of evolution, the overall density decreases, but as time increases, the density profile overgrows.

5.6 Problems

Problem 5.1 By completing the argument in the exponent of the expression $T_G = \frac{1}{2\pi}\int_{-\infty}^{+\infty} dq e^{i(x-x')q - Kq^2 t}$ show that the Green function of the heat conduction problem becomes:

$$T_G = \frac{1}{\sqrt{4\pi K t}} e^{-\frac{(x-x')^2}{4Kt}}. \tag{5.70}$$

Problem 5.2 Show that if the initial temperature distribution is Gaussian with zero mean and variance σ^2 then the temperature profile $T(x,t)$ remains Gaussian but with a linearly growing variance in time:

$$T(x,t) = \frac{1}{\sqrt{2\pi(\sigma^2 + 2Kt)}} e^{-\frac{x^2}{2(\sigma^2+2Kt)}} \tag{5.71}$$

Problem 5.3 Defining

$$T_G(x,t) = \frac{1}{\sigma(t)\sqrt{2\pi}} exp[\frac{-x^2}{2\sigma^2(t)}] \tag{5.72}$$

with $\sigma(t) = \sqrt{2Kt}$ show that:

$$T(x,t) = \sum_{n=-\infty}^{+\infty} (-1)^n T_G(x+nL,t) \tag{5.73}$$

is the solution of the diffusion equation with the initial condition $T(x,0) = \delta(x)$ and the Dirichlet boundary condition $T(-\frac{L}{2},t) = T(+\frac{L}{2},t) = 0$.

Problem 5.4 Solve the 1D heat equation with a Dirichlet boundary condition at $x = -\frac{L}{2}$ and Neumann boundary condition at $x = \frac{L}{2}$ that is $T(-\frac{L}{2},t) = T_L$; $\frac{\partial T}{\partial x}(\frac{L}{2},t) = \beta$. You may take both T_L and α equal to one.

Problem 5.5 Write a programme that solves the 1D heat equation with the Crank, Crank-Nicholson, and the Dufort-Frankel methods. Compare the results of these methods for the Dirichlet problem we solved in the text.

Problem 5.6 Another scheme for solving parabolic PDEs such as heat equation is *Richardson* scheme. According to this scheme, the 1D heat equation is discretised as follows:

$$\frac{T_i^{n+1} - T_i^{n-1}}{2\tau} = k\frac{T_{i+1}^n + T_{i-1}^n - 2T_i^n}{(\Delta x)^2} \tag{5.74}$$

In other words, the Richardson scheme is centred in both time and space. Write a programme that solves the heat equation with this method and compare your results with those found in problem five.

Problem 5.7 Consider the neutron diffusion problem under a mixed boundary condition. The Dirichlet condition at left: $n(-\frac{L}{2},t) = 0$ and the Neumann at right: $\frac{\partial}{\partial x} n(\frac{L}{2},t) = 0$.

(a) Analytically show that the critical length L_c becomes half of the value when the Dirichlet condition is held at both ends that is$L_c = \frac{\pi}{2}\sqrt{\frac{D}{C}}$.

(b) Numerically solve the problem with one of the algorithms you wish and verify the analytic result.

Problem 5.8 Consider a viscous fluid between two parallel infinite plates a distance d apart in the z direction. The upper plate is kept fixed at $z = d$ whereas the lower one oscillates about its equilibrium position $z = 0$ such that $v_z(0,t) = v_0 \cos \omega t$. Take the fluid kinetic viscosity to be $\nu = \frac{\eta}{\rho}$ where ρ is the density, and η is the fluid shear viscosity. We recall that the governing equation is the linearised Navier-Stocks equation $\frac{\partial}{\partial t} v_z(z,t) = \nu \frac{d^2}{dz^2} v_z$ where we have ignored the pressure variations with z.

(a) Numerically solve the problem and obtain the velocity profile $v_z(z,t)$ at various times with a suitable algorithm (you may proceed with the FTCS scheme).

(b) Add the nonlinear inertial term $(\mathbf{v}.\nabla)\mathbf{v} = v_z \frac{d}{dz} v_z$ to the left-hand side of the linearised Navier-Stocks equation and solve the problem numerically. For numerical values you may take $d = 0.5\ cm, v_0 = 20\ m/s, \omega = 50\pi$ and $\rho = 0.5\ gr/cm^3$.

(c) For what values of η we can ignore the nonlinear term?

Chapter 6

Partial differential equations: hyperbolic type

6.1 Advection equation

The previous chapter was devoted to parabolic-type PDEs. In this chapter, we intend to introduce the numerical solution of another important class of PDEs i.e.; hyperbolic equations. The prototype example is the d'Alembert wave equation (5.4). For pedagogical purposes, we begin with first-order linear hyperbolic PDE. The simplest nontrivial one which can be found in physics is the first order wave equation, the so-called *advection equation*:

$$\frac{\partial u(x,t)}{\partial t} = -c\frac{\partial u(x,t)}{\partial x} \tag{6.1}$$

This equation, also known as the linear convection equation, describes the spatial-temporal evolution of a passive scalar field $u(x,t)$ carried along by a flow having velocity c. The advection equation is the simplest version of the more general flux-conservation equation:

$$\frac{\partial \rho}{\partial t} = -\nabla.\mathbf{J}(\rho) \tag{6.2}$$

where ρ denotes the density of a physical quantity such as energy, mass, or charge, and $\mathbf{J}$ is the corresponding flux. The flux dependence on density normally comes from a phenomenological equation. In the particular case, $J(\rho) = c\rho$, the one-dimensional flux-conservation equation (6.2) reduces to the advection equation (6.1). Equation (6.1) can simply be solved by taking u as a function of $x - ct$. Given the initial condition $u(x, 0) = f(x)$, we find $u(x, t) = f(x - ct)$ as the general solution, which is a right-moving travelling wave. In general, the first-order hyperbolic PDEs can be solved by the method of characteristics. We refer the interested readers to excellent references (Myint-U and Debnath, 2007; Hassani, 2013, 2009) and focus our attention on finite difference schemes devised for the numerical solutions of the hyperbolic type PDEs.

6.2 Numerical solution of the advection equation

6.2.1 Forward time Forward space algorithm

The simplest discretisation scheme seems to be the forward time forward space (FTFS) method which is formulated as follows:

$$\frac{u_i^{n+1} - u_i^n}{\tau} = -c\frac{u_{i+1}^n - u_i^n}{\Delta x} \tag{6.3}$$

Note that we have adopted the same notation as chapter five for space and time grids. In particular, we use the shorthand notation u_i^n for $u(i\Delta x, n\tau)$. Unfortunately, this scheme is unconditionally unstable, and the numerical solution diverges for any choice of time and space grids. Replacing the spatial derivative with a centred scheme does not remedy the problem either. In this FTCS scheme, we have:

$$\frac{u_i^{n+1} - u_i^n}{\tau} = -c\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} \tag{6.4}$$

which gives:

$$u_i^{n+1} = u_i^n - \frac{c\tau}{2\Delta x}(u_{i+1}^n - u_{i-1}^n) \tag{6.5}$$

A conditionally stable algorithm is the forward time backward space (FTBS) method. It is also called *upwind* scheme: $\frac{u_i^{n+1} - u_i^n}{\tau} = -c\frac{u_i^n - u_{i-1}^n}{\Delta x}$.

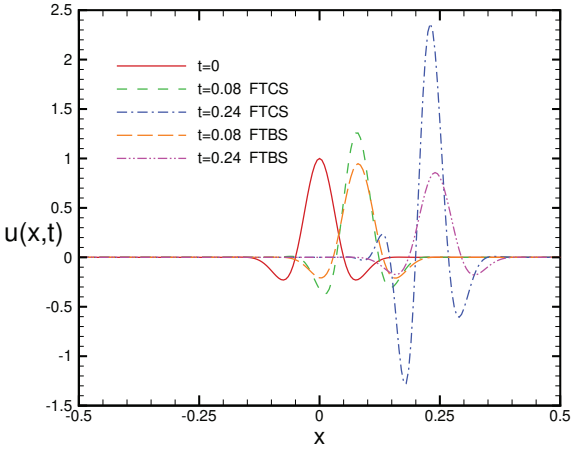

Figure 6.1: *Time evolution of a Gaussian wave pulse in the advection equation using FTCS and FTBS methods. Periodic boundary condition has been implemented. The system length and wave velocity are set to one. The pulse width is taken* $\sigma = 0.05$*, and the wave vector is set to* $k = 0.2$*. The number of grid points is* $N = 200$*. We have taken* $\tau = 0.004$*.*

The upwind scheme gives:

$$u_i^{n+1} = u_i^n - \frac{c\tau}{\Delta x}(u_i^n - u_{i-1}^n) \tag{6.6}$$

Before we turn to more advanced methods, let us numerically solve the advection equation for the Gaussian cosine-modulated initial condition (Garcia, 1999) by FTCS and FTBS methods. We take the initial condition $u(x,0) = f(x)$ as:

$$f(x) = \cos kx \; e^{-\frac{x^2}{2\sigma^2}} \tag{6.7}$$

The programme `Advection` (see Appendix 6.A for details) numerically solves the one-dimensional advection equation by three algorithms FTFS, FTCS, and FTBS. Figure (6.1) shows the results. As you can see, the solution given by the FTCS method is unstable and becomes distorted over time. The FTBS method is stable but does not give the correct solution. Unfortunately, it decays in time. We now discuss the more advanced methods of Lax and Lax-Wendroff.

6.2.2 Lax and Lax-Wendroff algorithms

In the lax scheme, the first term on the right side of (6.5) is replaced by its spatial average over the neighbouring sites i.e.; we replace u_i^n by $\frac{1}{2}(u_{i+1}^n + u_{i-1}^n)$. It can be shown that this substitution prevents wavefunction divergence and makes the algorithm stable, at least for certain choices of time and space grids. More specifically, in the Lax scheme, we have:

$$u_i^{n+1} = \frac{1}{2}(u_{i+1}^n + u_{i-1}^n) - \frac{c\tau}{2\Delta x}(u_{i+1}^n - u_{i-1}^n) \tag{6.8}$$

The lax scheme is stable if the following criterion known as the *Courant-Friedrichs-Lewy* (CFL) condition is satisfied (Garcia, 1999).

$$\frac{c\tau}{\Delta x} \leq 1 \tag{6.9}$$

In other words, τ should be less than or equal to a characteristic time scale $\tau_w = \frac{\Delta x}{c}$. The derivation of the CFL condition is based on the stability analysis, which you can find in many standard textbooks such as (Garcia, 1999) and (Vesely, 2001). Notice that if we use the maximum usable timestep $\tau_{max} = \tau_w$ in (6.8) we find:

$$u_i^{n+1} = u_{i-1}^n \tag{6.10}$$

which is the discretisation of the exact solution of the advection equation. Note that despite the Lax algorithm being stable for $\tau \leq \tau_w$, the solution becomes rapidly flawed when τ becomes smaller. This example dismisses the popular belief that the smaller the time step, the better the solution! To improve the Lax scheme, one can proceed along the following steps, which are based on Taylor's expansion:

$$u(x, t+\tau) = u(x,t) + \frac{\partial u}{\partial t}\tau + \frac{\partial^2 u}{\partial t^2}\frac{\tau^2}{2} + O(\tau^3) \tag{6.11}$$

By taking the time derivative from the advection equation (6.1) we find:

$$\frac{\partial^2 u(x,t)}{\partial t^2} = -c\frac{\partial}{\partial t}(\frac{\partial u}{\partial x}) = -c\frac{\partial}{\partial x}(\frac{\partial u}{\partial t}) = c^2\frac{\partial^2 u(x,t)}{\partial x^2} \tag{6.12}$$

Replacing the time derivatives in (6.11) with space ones (using (6.12) and the advection equation itself), we arrive at:

$$u(x, t+\tau) = u(x,t) - c\frac{\partial u}{\partial x}\tau + c^2\frac{\partial^2 u}{\partial x^2}\frac{\tau^2}{2} + O(\tau^3) \tag{6.13}$$

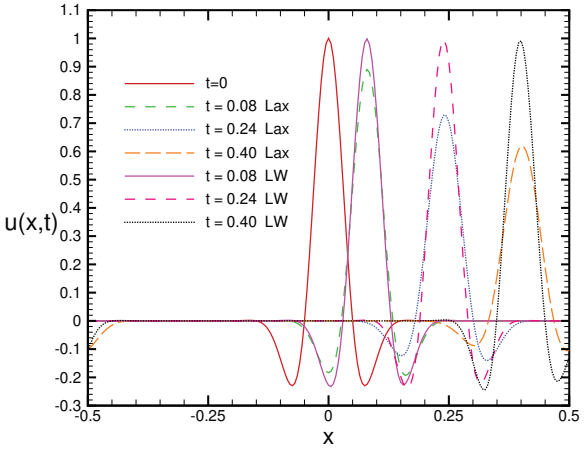

Figure 6.2: *Time evolution of a Gaussian wave pulse in the advection equation using Lax and Lax-Wendroff methods. The parameters are identical to those in figure (6.1).*

The next step is to approximate the spatial derivatives by finite difference. Using a centered scheme for them, we obtain (up to the third order in τ):

$$u_i^{n+1} = u_i^n - c\tau \frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} + \frac{c^2\tau^2}{2} \frac{u_{i+1}^n + u_{i-1}^n - 2u_i^n}{(\Delta x)^2} \qquad (6.14)$$

This formulation is known as the *Lax-Wendroff* method. Notice that the last term is a discretised second derivative and gives an artificial diffusion stabilising the numerical solution. Stability analysis shows that this method is stable when the CFL condition holds. The programme `AdvectionLax` (see Appendix 6.B for details) numerically solves the one-dimensional advection equation by the algorithms Lax and Lax-Wendroff. Figure (6.2) exhibits the results. You observe that both the Lax and the Lax-Wendroff are stable. However, the Lax method does not give the correct answer, and the solution rapidly damps in time like the FTBS method. On the other hand, the Lax-Wendroff algorithm gives the correct solution and seems to be an efficient algorithm for hyperbolic PDEs. Before proceeding toward the

d'Alembert wave equation, let us discuss some implicit algorithms that can be implemented in the advection equation.

6.3 Implicit algorithms

If you substitute the wavefunction values in the right-hand sides of (6.5) and (6.6) by their values in the updated step $n+1$ we will arrive at the implicit formulation. For example, the implicit FTCS scheme appears to be:

$$u_i^{n+1} = u_i^n - \frac{c\tau}{2\Delta x}(u_{i+1}^{n+1} - u_{i-1}^{n+1}) \tag{6.15}$$

Rearranging the terms at $n+1$ on the left and those at n on the right-hand side, we arrive at:

$$\frac{1}{2}a[u_{i+1}^{n+1} - u_{i-1}^{n+1}] + u_i^{n+1} = u_i^n \tag{6.16}$$

where $a = \frac{c\tau}{\Delta x}$. Taking care of the boundary conditions, we can write the set of algebraic equations in (6.16) in a compact matrix form:

$$\mathbf{C}\mathbf{u}^{n+1} = \mathbf{u}^n \tag{6.17}$$

where $\mathbf{u}^\dagger = (u_0, u_1, \cdots, u_N)$ and the elements of the matrix $\mathbf{C}$ are read from (6.16). Given the u_i at the current step n, one can find them at the next step $n+1$ by numerically solving the linear set of equations (6.17). We leave it as an exercise for you to solve the advection problem with this algorithm. Another widely used implicit scheme is the *Crank-Nicolson* algorithm which was introduced in chapter five in the context of parabolic PDEs. This scheme is implemented as follows for the advection equation.

$$\frac{u_i^{n+1} - u_i^n}{\tau} = -\frac{c}{2}[\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x} + \frac{u_{i+1}^{n+1} - u_{i-1}^{n+1}}{2\Delta x}] \tag{6.18}$$

Rearranging the terms gives:

$$\frac{a}{4}[u_{i+1}^{n+1} - u_{i-1}^{n+1}] + u_i^{n+1} = u_i^n - \frac{a}{4}[u_{i+1}^n - u_{i-1}^n] \tag{6.19}$$

where $a = \frac{c\tau}{\Delta x}$. We can recast (6.19) in a compact matrix form as follows:

$$\mathbf{C}\mathbf{u}^{n+1} = \mathbf{B}\mathbf{u}^n \tag{6.20}$$

where matrices $\mathbf{B}$ and $\mathbf{C}$ are tridiagonal except for some elements due to the boundary conditions. The set of linear equations can be simply solved by standard numerical recipes once $\mathbf{u}^n$ is given. Care should be taken into account when implying the boundary conditions at sites $i = 0$ and $i = N$. So far, we have only considered periodic boundary conditions. It is also possible to incorporate other boundary conditions, such as fixed or semi-fixed. We leave the numerical solution of other types of boundary conditions to exercises. In the next section, we will discuss the second-order linear wave equation.

6.4 d'Alembert Wave equation

One of the oldest PDEs is the ubiquitous classical wave equation which is normally called the d'Alembert wave equation after its mathematical formulation by Jean-Baptiste le Rond d'Alembert a French mathematician, physicist, philosopher, and music theorist in the eighteen century. The applicability range of this classical wave equation includes acoustic, optic, and elasticity, to name a few. In one dimension, this equation takes the form:

$$\frac{\partial^2 u(x,t)}{\partial t^2} = c^2 \frac{\partial^2 u(x,t)}{\partial x^2} \tag{6.21}$$

The constant c is the velocity of wave propagation in a physical medium, and $u(x,t)$ denotes the displacement of the medium differential element at x from its equilibrium at time t. Since the time derivative is second-order, one has to specify two initial conditions to uniquely solve the problem. These initial conditions are normally given by functions $f(x)$ and $g(x)$ as follows:

$$u(x,0) = f(x); \quad \frac{\partial u}{\partial t}(x,0) = g(x) \tag{6.22}$$

The problem lies in the category of initial-boundary value problems. The analytical solution crucially depends on the type of boundary conditions. In general, there are four types of boundary conditions, namely Dirichlet, Neumann, Robin, and mixed. The analytical solution for each type of boundary condition is discussed in many standard mathematical physics textbooks such as (Myint-U and Debnath, 2007; Hassani, 2013, 2009), and we avoid deriving them. Instead, we focus our attention on numerics and only state the analytic solutions when

needed. In the case where the boundaries are at infinity i.e.; at $\pm\infty$ the analytical solution turns out to be (Myint-U and Debnath, 2007):

$$u(x,t) = \frac{1}{2}[f(x-ct) + f(x+ct)] + \frac{1}{2c}\int_{x-ct}^{x+ct} g(t')dt' \tag{6.23}$$

Another analytical solution can be obtained for the semi-infinite string in which $x \in [0, +\infty[$. The most general boundary condition at the left boundary $x = 0$ is:

$$u(0,t) = p(t) \tag{6.24}$$

where $p(t)$ is a given function. The analytical solution for $0 \leq x < ct$ becomes (Myint-U and Debnath, 2007):

$$u(x,t) = p(t-\frac{x}{c}) + \frac{1}{2}[f(x-ct) + f(x+ct)] + \frac{1}{2c}\int_{x-ct}^{x+ct} g(t')dt' \tag{6.25}$$

For $x > ct$, the solution is identical to (6.23). Let us now solve the wave equation with a computer. We use the method of midpoint leapfrog to numerically solve the wave equation (6.21). In this algorithm, both the second-order time and space derivatives are approximated by centred schemes:

$$\frac{u_i^{n+1} + u_i^{n-1} - 2u_i^n}{\tau^2} = c^2 \frac{u_{i+1}^n + u_{i-1}^n - 2u_i^n}{(\Delta x)^2} \tag{6.26}$$

As usual showing $\frac{c\tau}{\Delta x}$ by a we can recast (6.26) as follows:

$$u_i^{n+1} = 2u_i^n - u_i^{n-1} + a^2[u_{i+1}^n + u_{i-1}^n - 2u_i^n] \tag{6.27}$$

This algorithm is two-step in the sense that the wavefunction at time step $n+1$ depends not only on its values at the previous step n but also on the second previous step $n-1$. Therefore, a starter solution is required to find the wavefunction values at $n = 1$ from the initial conditions. A proposition is devised as follows. Backward discretisation of the second initial condition i.e.; $\frac{\partial u}{\partial t}(x,0) = g(x)$ gives:

$$\frac{u_i^0 - u_i^{-1}}{\tau} = g_i \tag{6.28}$$

where g_i denotes $g(x_i)$. This gives:

$$u_i^{-1} = u_i^0 - g_i\tau \tag{6.29}$$

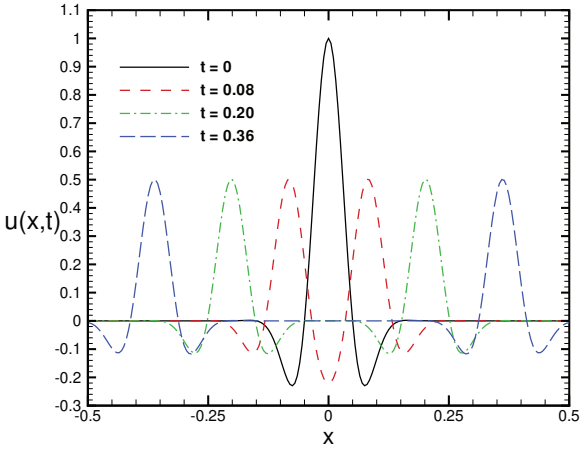

Figure 6.3: *Numerical solution of the d'Alembert wave equation in one spatial dimension. The periodic boundary condition is implemented. CTCS method is used with these parameters:* $c = L = 1, \tau = 0.004$. *The number of spatial grid points is* $N = 200$.

Using (6.29) in (6.27), it would be possible to proceed iteratively in time. Another choice is to use centered discretisation scheme for $\frac{\partial u}{\partial t}(x, 0) = g(x)$ which gives:

$$u_i^{-1} = u_i^0 - 2g_i\tau \tag{6.30}$$

The programme `Dalembert` (see Appendix 6.C for details) numerically solves the one-dimensional classical wave equation by the midpoint leapfrog algorithm. Figure (6.3) exhibits the result for the infinite space. The initial conditions are the same as in the advection problem i.e.; a cosine-modulated Gaussian pulse. Hopefully, we observe that the CTCS method is not only stable but also gives the correct answer. The emergence of two pulses is obvious. One is left moving, whereas the other one goes towards the right. Let us now consider a different boundary condition. Suppose our medium is semi-infinite and extends from $x = 0$ to infinity. Furthermore, we adopt the Dirichlet condition at the left boundary and take $u(0, t) = 0$. In other words, the left boundary is a hard wall if the medium is assumed to be a flex-

ible string. Let the initial wave shape be again a cosine-modulated Gaussian centered at $x_0 > 0$:

$$u(x,0) = \cos k(x - x_0)\ exp(-\frac{(x-x_0)^2}{2\sigma^2}) \tag{6.31}$$

In practice, one cannot take the system length L as infinity. We know that at $t > 0$ two outgoing travelling waves emerge. We are not interested in the right-going wave because we know physically it travels with a constant speed c to the right. The interesting part of the problem lies in the left-moving wave. Physically we know that this wave moves with constant phase velocity c toward the left boundary. There it reflects and propagates towards the right. However, its phase changes by π. Let us see if our programme can reproduce this feature. In the programme we take a finite value L for the medium. With no loss of generality, we put L equal to one and take $x_0 = \frac{L}{2}$. The boundary condition implies $u_0^n = 0$ at all time steps. Imposing this constraint in the programme, we can simply find the solution by the CTCS method. Figure (6.4) sketches the solution profile at various times. The generation of a reflective wave at $x = 0$ is nicely produced by our numerical solution. After reflection, the wave changes phase by π and moves towards the right from below the horizontal axis. It would be interesting to see the application of implicit algorithms to the classical wave equation. We leave it as an exercise to you. Let us now see what the solution looks like if the medium is finite. As a physical example, consider a string of length L, which is fixed at both sides $x = 0$ and $x = L$. In terms of boundary conditions, we have:

$$u(0,t) = 0; \quad u(L,t) = 0 \tag{6.32}$$

This condition can simply be implemented in the programme. We set $u_0^n = u_N^n = 0$ for all time step n. In other words, we do not update the wavefunction at the first and the last grids $i = 0, N$. Figure (6.5) exhibits the solution at various times. The initial condition is a Gaussian pulse centred at $x_0 = \frac{L}{4}$. The wave is now reflected at both boundaries, and you see the superposition principle. Before considering other hyperbolic equations, let us explain a different approach to solving classical wave equations. Introducing auxiliary variables:

$$p(x,t) = \frac{\partial u(x,t)}{\partial t}; \quad q(x,t) = c\frac{\partial u(x,t)}{\partial x} \tag{6.33}$$

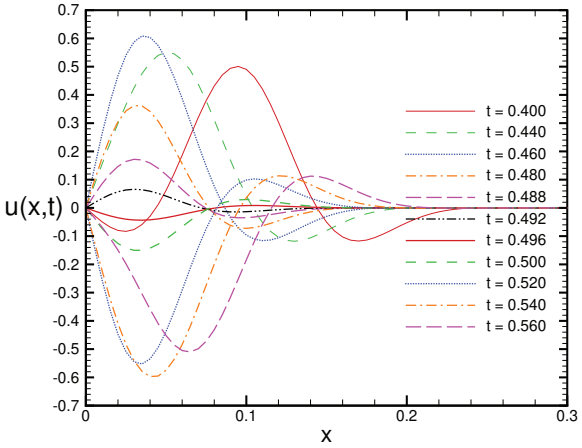

Figure 6.4: *Numerical solution of the d'Alembert wave equation in one spatial dimension. The space is semi-infinite with the left boundary condition* $u(0,t) = 0$. *CTCS method is used with these parameters:* $c = 1, \tau = 0.004$. *The number of grid points is* $N = 200$.

One can write the second-order wave equation as follows:

$$\frac{\partial p(x,t)}{\partial t} = c\frac{\partial q(x,t)}{\partial x} \tag{6.34}$$

Furthermore, the condition $\frac{\partial^2 u(x,t)}{\partial t \partial x} = \frac{\partial^2 u(x,t)}{\partial x \partial t}$ implies:

$$\frac{\partial q(x,t)}{\partial t} = c\frac{\partial p(x,t)}{\partial x} \tag{6.35}$$

In effect, equations (6.34) and (6.35) comprise a set of first-order system of PDSs. The initial conditions in terms of $p(x,t)$ and $q(x,t)$ turn out to be:

$$p(x,0) = g(x); \quad q(x,0) = cf'(x) \tag{6.36}$$

We leave it as an exercise to solve the set of equations (6.34) and (6.35) and find the wave solution $u(x,t)$.

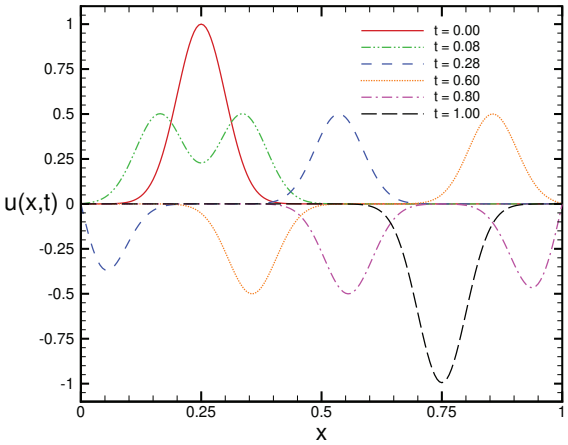

Figure 6.5: *Numerical solution of the d'Alembert wave equation in one spatial dimension. The space is finite with fixed boundary conditions at both boundaries i.e.;* $u(0,t) = u(L,0) = 0$. *All the other parameters are identical to the figure (6.4).*

6.5 Nonlinear hyperbolic PDEs

So far, we have discussed the simplest first-order and second-order hyperbolic PDEs. However, many PDEs in fluid dynamics and heat transfer are nonlinear. The advection and d'Alembert wave equations provided some insight for approaching nonlinear equations. We now intend to discuss a prototype nonlinear PDE the so-called inviscid *Burgers equation*. This model equation is a one-dimensional hyperbolic equation and has found many applications in physics. The Burgers equation is written below:

$$\frac{\partial u(x,t)}{\partial t} = -u\frac{\partial u(x,t)}{\partial x} \tag{6.37}$$

As you can see, by replacing c with u in the advection equation, we arrive at the Burgers equation. Equation (6.37) has the interpretation of wave propagation with a non-constant velocity equal to the wave amplitude. In a flux conservation form, we can recast the Burgers

equation as follows:

$$\frac{\partial u(x,t)}{\partial t} = -\frac{\partial J(x,t)}{\partial x} \tag{6.38}$$

with the flux $J(x,t) = \frac{u^2(x,t)}{2}$. We will later show how Burgers equation is derived from the more general Navier-Stocks equation in fluid dynamics.

6.5.1 Solution of Burgers equation: Lax method

As we saw for the advection equation, in the Lax scheme, the time derivative is discretised forward, whereas the space derivative is discretised in a centred manner. Therefore, we have:

$$\frac{u_i^{n+1} - u_i^n}{\tau} = -[\frac{(u_{i+1}^n)^2 - (u_{i-1}^n)^2}{4\Delta x}] \tag{6.39}$$

Which in turn gives:

$$u_i^{n+1} = u_i^n - \frac{\tau}{4\Delta x}[(u_{i+1}^n)^2 - (u_{i-1}^n)^2] \tag{6.40}$$

Replacing the first term on the right-hand side by the average over its neighbours $i-1$ and $i+1$ gives the Lax scheme:

$$u_i^{n+1} = \frac{1}{2}[u_{i-1}^n + u_{i+1}^n] - \frac{\tau}{4\Delta x}[(u_{i+1}^n)^2 - (u_{i-1}^n)^2] \tag{6.41}$$

It is also possible to improve the Lax method with the more elaborate Lax-Wendroff (LW) method. The details of the LW scheme are thoroughly explained in (Hoffmann and Chiang, 2009), and we only quote the result here.

$$u_i^{n+1} = u_i^n - \frac{\tau}{2\Delta x}[J_{i+1}^n - J_{i-1}^n] +$$

$$\frac{\tau^2}{4(\Delta x)^2}[(u_{i+1}^n + u_i^n)(J_{i+1}^n - J_i^n) - (u_i^n + u_{i-1}^n)(J_i^n - J_{i-1}^n)] \tag{6.42}$$

The programme `Burgers` (see Appendix 6.D for details) numerically solves the one-dimensional Burgers equation by Lax and Lax-Wendroff algorithms. Figure (6.6) exhibits the numerical solution of the Burgers equation obtained by the Lax method. The initial condition is a

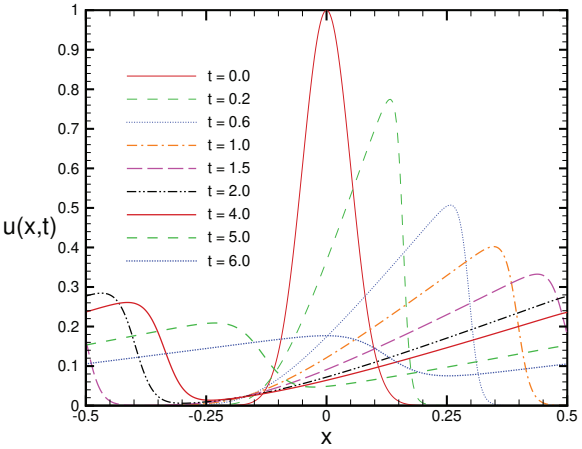

Figure 6.6: *Numerical solution of the Burgers equation in one spatial dimension by Lax scheme with periodic boundary condition. The initial condition is a Gaussian pulse with* $\sigma = 0.05$ *centred at* $x = 0$*. The parameters are: system length* $L = 1$*, number of grid points* $N = 400$*, and timestep* $\tau = 0.001$*.*

Gaussian pulse centred at $x = 0$. The system length L is taken unity, and the periodic boundary condition is imposed. You see that after somewhat evolution, the solution profile becomes sharp. This is reminiscent of the shock wave formation, theoretically predicted in Burgers' equation. Let us see the consequences of applying the Lax-Wendroff method to the problem. In figure (6.7), we show the numerical solution obtained by the Lax-Wendroff method. You see some instabilities in the Lax-Wendroff method, but the overall behaviour is qualitatively the same as in the Lax method. The formation of a shock wave is apparent. Let us see the development of shock waves for another initial condition. Instead of a Gaussian pulse, we assume the initial condition to be a step function that is $u(x, 0) = 1$ for $\frac{-L}{2} < x < 0$ and $u(x, 0) = 0$ for $0 < x < \frac{L}{2}$. Figure (6.8) exhibits the solution profiles at various times obtained by the Lax method. The overall behaviour is qualitatively similar to the Gaussian pulse in the initial condition. As you see, the shock's height depends on the initial condition.

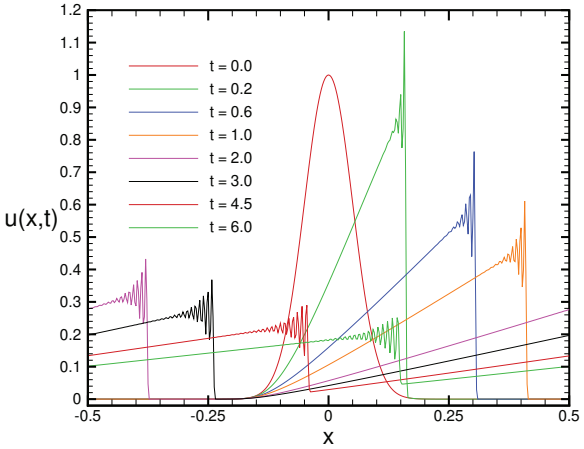

Figure 6.7: *Numerical solution of the Burgers equation in one spatial dimension by Lax-Wendroff scheme with periodic boundary condition. The parameters are identical to fig. (6.6).*

6.5.2 Traffic flow equation

The flow of vehicles can be described by a partial differential equation. You may be astonished to hear it for the first time but let me justify it. It is well-known that the natural modeling approach for vehicular dynamics is to consider each vehicle as a particle and then try to write a dynamical equation of their motion. In a naive approach, the acceleration of each particle depends on the velocity difference as well as the spatial gap to its leader vehicle. This approach is called car-following in the literature. The other popular approach for modelling the motion of vehicles is *cellular automata* in which time, space, and speed of vehicles are taken as discrete variables. For a detailed explanation of these microscopic approaches, see (Andreas Schadschneider and Nishinari, 2010). It is also possible to adopt a macroscopic continuum approach to traffic flow. if you look at a vehicular traffic flow from above and forget the microscopic details, you may equivalently describe its large-scale characteristics via coarse-grained variables such as $\rho(x,t)$ and $v(x,t)$ which are the traffic flow density and velocity at location x of

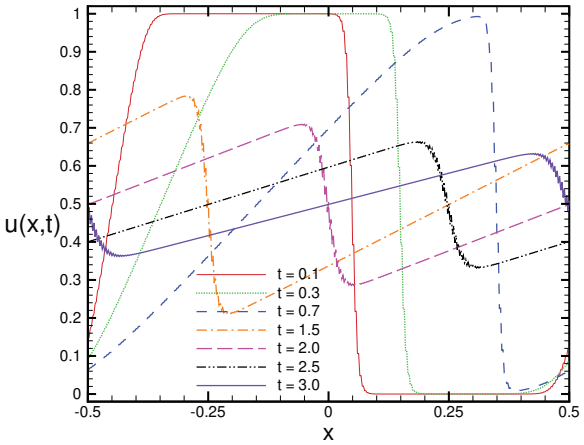

Figure 6.8: *Numerical solution of the Burgers equation in one spatial dimension by Lax scheme with periodic boundary condition. The initial condition is a step function centred at $x = 0$. The parameters are identical to fig. (6.7).*

the road at time t. We shall now discuss the numerical solution of another PDE which can describe the evolution of a density ρ. For the sake of simplicity, we restrict ourselves to one spatial dimension. The time evolution of density $\rho(x,t)$ is governed by the following PDE continuity equation:

$$\frac{\partial \rho}{\partial t} = -\frac{\partial}{\partial x} J(\rho) \tag{6.43}$$

Here $J(x,t)$ is the associated flux to the density $\rho(x,t)$. To solve (6.43), we need to know the dependence of J on ρ. Normally this dependence is given phenomenologically. We wish to apply (6.43) to a traffic flow problem. We all know that the flow of vehicles is a granular flow. Our grains are vehicles, and a correct description should entail the position and the velocity of every vehicle at arbitrary time t. In recent years vast modelling literature has emerged for writing appropriate differential or difference equations that govern the equation of motion of vehicles. You may refer to comprehensive review articles

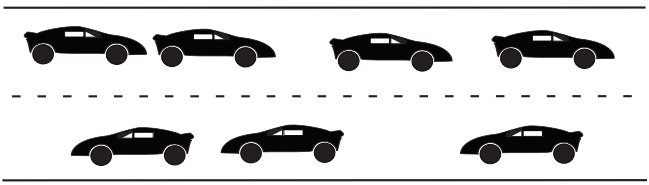

Figure 6.9: *A single-lane bidirectional vehicular traffic flow.*

on traffic flow for further details (L. Santen and Chowdhurry, 2000; Helbing, 2001). In a class of models which are named *car following*, a set of differential equations are written for vehicles. In the simplest formulation, the acceleration of car k can only depend on the motion status of its lead car $k+1$. Denoting the position and velocity of car k at time t by $x_k(t)$ and $v_k(t)$ respectively one can write the motion equation as follows:

$$\frac{d}{dt}v_k(t) = f(x_k(t), v_k(t), x_{k+1}(t), v_{k+1}(t)) \quad k = 1, 2, \cdots \tag{6.44}$$

The function f plays the force role in Newton's equation of motion in classical physics. Solving this set of nonlinear coupled differential equations, even numerically, is a formidable task, and various types of instabilities may emerge. Besides, we may neither always be interested nor it is useful to know such microscopic details of motion for each car. Instead, traffic engineers would like to know the average behaviour of traffic flow quantities at certain locations on the road. Therefore it would be wise to leave the microscopic description in favour of a more practical but macroscopic one. In this continuum description, the function $\rho(x,t)$ gives the car density at location x of the road at time t. For simplicity, we take the road to being single-lane. $J(x,t)$ denotes the number flux of vehicles at location x. The number flux J is defined as the number of vehicles per unit time passing the location x of the road. To complete our description, let $v(x,t)$ show the average velocity of cars at time t and location x. We can use the hydrodynamical approximation and take $J(x,t) = \rho(x,t)v(x,t)$. Notice that we have assumed there is neither a source (on-ramps) nor a sink (off-ramp) in the road. For solving (6.43) we must know the average velocity $v(x,t)$. Here a phenomenological approach is used. We know that in light traffic conditions, when ρ is small, the car flux is proportional to ρ. When the traffic condition is congested, the flux should decrease if

the density increases. This suggests a simple form for J that is:

$$J(x,t) \propto \rho(x,t)[1 - \frac{\rho(x,t)}{\rho_m}]. \tag{6.45}$$

where ρ_m is the maximum density. Showing the proportionality constant by v_m the velocity $v(x,t)$ turns out to:

$$v(x,t) = c(\rho(x,t)) = v_m[1 - \frac{\rho(x,t)}{\rho_m}] \tag{6.46}$$

v_m is the maximum velocity (speed limit) that each car can attain. The maximum density ρ_m is achieved when vehicles are moving very slowly and bumper-to-bumper. By insertion of (6.46) in (6.43), we arrive at the generalised inviscid Burgers equation. Alternatively, this kind of nonlinear equation appears in acoustic and nonlinear wave theory as well. Equation (6.43) with (6.46) for $J(x,t)$ can be analytically solved by the method of characteristics. You can find the details of this nice approach in (Garcia, 1999). We now try to solve the traffic equation for a well-known problem: *traffic at a stoplight*. In this problem, we assume that at $t = 0$ the traffic light goes green. Before that moment, the traffic density is in the form of a step function that is $\rho(x,0) = \rho_m$ for $x < 0$ and $\rho(x,0) = 0$ for $x > 0$. In fact, before the light turns green, the traffic behind the light is at the maximum density ρ_m (bumper-to-bumper). The method of the characteristics gives the analytical solution as follows (Garcia, 1999):

$$\begin{cases} \rho(x,t) = \rho_m & x \leq -v_m t, \\ \rho(x,t) = \frac{1}{2}(1 - \frac{x}{v_m t})\rho_m & \leq -v_m t < x < v_m t, \\ \rho(x,t) = 0 & x \geq v_m t. \end{cases} \tag{6.47}$$

Let us now see how well our numerical scheme can do. The FTCS scheme gives:

$$\rho_i^{n+1} = \rho_i^n - \frac{\tau}{2\Delta x}(J_{i+1}^n - J_{i-1}^n) \tag{6.48}$$

where $J_i^n = J(\rho_i^n)$. On the other hand, the Lax scheme gives:

$$\rho_i^{n+1} = \frac{1}{2}[\rho_{i+1}^n + \rho_{i-1}^n] - \frac{\tau}{2\Delta x}(J_{i+1}^n - J_{i-1}^n) \tag{6.49}$$

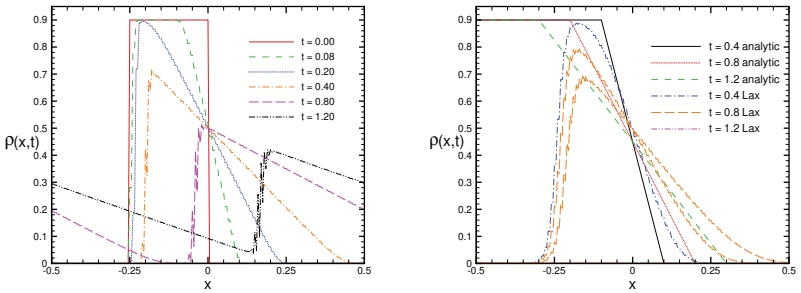

Figure 6.10: *Numerical solution of the traffic equation by the Lax scheme. The periodic boundary condition is imposed. The initial condition is a step function with height* $0.9\rho_{max}$ *centred at* $x = 0$ *and extended back to* $x = -\frac{L}{4}$.

Eventually, the Lax-Wendroff scheme gives:

$$\rho_i^{n+1} = \rho_i^n - \frac{\tau}{2\Delta x}(J_{i+1}^n - J_{i-1}^n) + \frac{\tau^2}{2\Delta x}[c_{i+\frac{1}{2}}^n \frac{J_{i+1}^n - J_i^n}{\Delta x} -$$

$$c_{i-\frac{1}{2}}^n \frac{J_i^n - J_{i-1}^n}{\Delta x}] \tag{6.50}$$

where

$$c_{i\pm\frac{1}{2}}^n \equiv c(\rho_{i\pm\frac{1}{2}}^n); \quad \rho_{i\pm\frac{1}{2}}^n \equiv \frac{\rho_{i\pm1}^n + \rho_i^n}{2} \tag{6.51}$$

For more details see (Hoffmann and Chiang, 2009; Garcia, 1999). The programme `Traffic` (see Appendix 6.E for details) numerically solves the one-dimensional traffic equation by FTCS, Lax, and Lax-Wendroff algorithms. Figure (6.10) shows the density profile's time evolution for the stoplight initial condition. The periodic boundary condition is taken into account. We remark that the FTCS scheme is unstable, and the solution diverges after a few timesteps. The evolution of the shock profile is obvious. This is a consequence of the problem's nonlinearity. For comparison, we show the analytical result (6.47). Despite there are deviations with respect to the analytic result the overall agreement is satisfactory.

6.6 Problems

Problem 6.1 The leapfrog scheme for solving the advection equation uses centred derivatives for both time and space variables:

$$\frac{u_i^{n+1} - u_i^{n-1}}{2\tau} = -c\frac{u_{i+1}^n - u_{i-1}^n}{2\Delta x}$$

. Notice that this algorithm is a two-level scheme. It uses $u(x,t)$ at the current step n and the previous step $n-1$ to find the updated $u(x,t)$. You need a starter sub-step to get it started. Modify the `Advect` to numerically solve the 1D advection problem by the leap-frog method and compare your findings with other algorithms discussed in the chapter. For what value of τ is the algorithm stable?

Problem 6.2 Numerically solve the 1D advection problem by the implicit FTCS method and compare your findings with other explicit algorithms discussed in the chapter.

Problem 6.3 Numerically solve the 1D advection problem by the implicit Crank-Nicolson method and compare your findings with other explicit algorithms discussed in the chapter.

Problem 6.4 Numerically solve the 1D d'Alembert wave equation by the implicit Crank-Nicolson method and compare your findings with the explicit algorithm discussed in the chapter.

Problem 6.5 Numerically solve the 1D d'Alembert wave equation by the method of the decomposition of a second-order PDE into two first-order ones (6.34) and (6.35). Compare your solution with the one obtained in the chapter.

Problem 6.6 Modify the programme `AdvectLax` to numerically solve the advection equation for the fixed $u(x = \frac{-L}{2}, t) = u(\frac{L}{2}, t) = 0$ and the semi-fixed boundary conditions: $u(x = \frac{-L}{2}, t) = \sin\omega t$; $u(x = \frac{L}{2}, t) = 0$. Use $N = 200$ grid points and test various frequencies. Apply both Lax and Lax-Wendroff algorithms.

Problem 6.7 Numerically solve the traffic PDE equation by the Lax-Wendroff method and compare your findings with the Lax method. Use the same parameters of the figure (6.10).

Problem 6.8 After a time t, the total number of vehicles that have passed the intersection traffic light is $N(t) = \int_0^\infty \rho(x,t)dx$. Show that $N(t) = \int_0^t J(0,t)dt$ in which $J(x,t)$ is the traffic flow.

Problem 6.9 Movement of a congested region in a traffic flow is an important issue. As a simple example that exhibits some aspects of the problem, we model the congested region by a Gaussian perturbation pulse as follows: $\rho(x,0) = \rho_0[1 + \alpha e^{\frac{-x^2}{2\sigma^2}}]$. Taking $\sigma = \frac{L}{10}$ and $\alpha = \frac{1}{5}$ numerically solve the traffic PDE for various values of ρ_0 and follow the traffic congestion motion. Does it move forward or backward? Show that for $\rho_0 = \frac{\rho_m}{2}$ the perturbation is almost stationary.

Chapter 7

Partial differential equations: elliptic type

7.1 Laplace equations

The previous two chapters were devoted to the parabolic and hyperbolic types of PDEs. In this chapter, we intend to discuss the numerical techniques for solving the third type of PDEs i.e.; elliptic equations. The elliptic PDE is associated with boundary values. The most important elliptic PDE in physics is undoubtedly the Laplace (Poisson) equation in electrostatic. Besides electrostatic, the governing equations in heat transfer and fluid dynamics reduce to elliptic equations in the steady state. For example in the heat transfer equation (5.3) the left-hand side term $\frac{\partial T}{\partial t}$ becomes zero in the stationary state and the diffusion equation reduces to $\frac{\partial^2 T}{\partial x^2} = 0$ which is an elliptical equation. This chapter explores a completely different numerical methodology for solving an elliptical PDE. In particular, we discuss relaxation and spectral methods. The domain solution of an elliptic PDE is a closed region R in space. On the boundary of R, the function value or its normal derivative or a combination of them should be prescribed if one wants to find the solution within the domain. Here also there is no general existence/uniqueness theorem and there may not exist a solution for a general boundary condition. Let us start with the Laplace equation the paradigm of the elliptic PDEs. For simplicity,

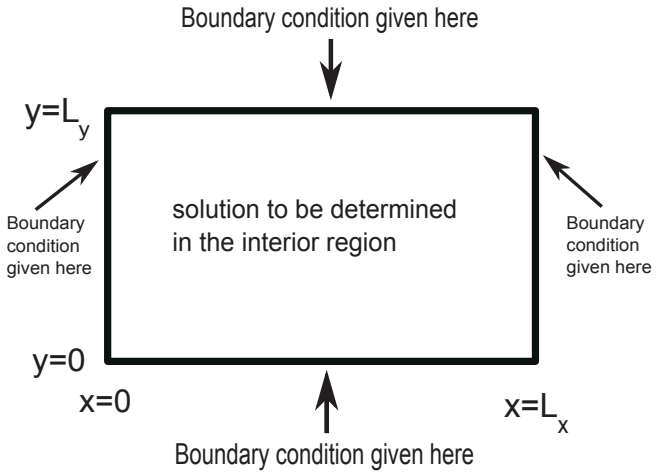

Figure 7.1: *Schematic representation for a boundary value problem in two-dimension with a rectangular boundary.*

we consider a two-dimensional case with a rectangular domain R and the Dirichlet boundary condition. The Laplace equation reads:

$$\frac{\partial^2 \Phi(x,y)}{\partial x^2} + \frac{\partial^2 \Phi(x,y)}{\partial y^2} = 0. \tag{7.1}$$

We take the rectangle's sides lengths as L_x and L_y respectively and adopt the following boundary values:

$$\Phi(x,0) = a_1;\ \Phi(L_x,y) = a_2;\ \Phi(x,L_y) = a_3;\ \Phi(0,y) = a_4 \tag{7.2}$$

The constants $a_1, \cdots, a_4$ are given. See figure (7.1) for illustration.

7.2 Numerical solution of Laplace equation

Replacing the second-order spatial derivatives by centred finite differences we arrive at:

$$\frac{\Phi_{i+1,j} + \Phi_{i-1,j} - 2\Phi_{i,j}}{(\Delta x)^2} + \frac{\Phi_{i,j+1} + \Phi_{i,j-1} - 2\Phi_{i,j}}{(\Delta y)^2} = 0 \tag{7.3}$$

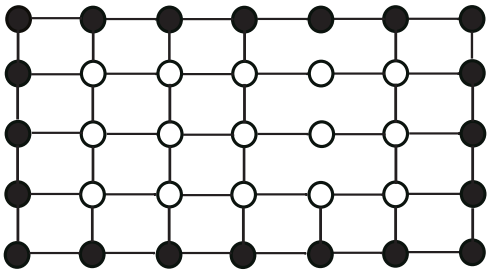

Figure 7.2: *Schematic representation for space discretisation of a rectangular value problem in two dimensions.*

As usual, Δx and Δy denote spatial grid lengths in x and y directions respectively. Notice that in general, they differ from each other. The ranges of the grid counters are $i = 0, \cdots, N_x$ and $i = 0, \cdots, N_y$ correspondingly. Note that $\Delta x = \frac{L_x}{N_x}$ and $\Delta y = \frac{L_y}{N_y}$. See figure (7.2) for an illustration in which $N_x = 6$ and $N_y = 4$ give rise to a total of thirty-five grid points twenty of which lie on the boundary. Be aware that we cannot apply (7.3) to the boundary grids. For example if $i = N_x$ then $\Phi_{N_x+1,j}$ is undefined for any j. In practice, we use (7.3) only for the interior points. When the grid point lies on the boundary, we directly use the given boundary value for the grid value. For clarification let us explicitly write (7.3) for all the interior points of our rectangular region with $N_x = 4$ and $N_y = 3$:

$$\begin{cases} \Phi_{2,1} + a_4 - 2(1+\beta^2)\Phi_{1,1} + \beta^2\Phi_{1,2} + a_1\beta^2 = 0, \\ \Phi_{2,2} + a_4 - 2(1+\beta^2)\Phi_{1,2} + a_3\beta^2 + \beta^2\Phi_{1,1} = 0, \\ \Phi_{3,1} + \Phi_{1,1} - 2(1+\beta^2)\Phi_{2,1} + \beta^2\Phi_{2,2} + a_1\beta^2 = 0, \\ \Phi_{3,2} + \Phi_{1,2} - 2(1+\beta^2)\Phi_{2,2} + \beta^2\Phi_{2,1} + a_3\beta^2 = 0, \\ a_2 + \Phi_{2,1} - 2(1+\beta^2)\Phi_{3,1} + \beta^2\Phi_{3,2} + a_1\beta^2 = 0, \\ a_2 + \Phi_{2,2} - 2(1+\beta^2)\Phi_{3,2} + \beta^2\Phi_{3,1} + a_3\beta^2 = 0. \end{cases} \tag{7.4}$$

where $\beta = \frac{\Delta x}{\Delta y}$. Taking the six Φ values at the interior grids i.e.; $\Phi_{1,1}, \Phi_{2,1}, \Phi_{3,1}, \Phi_{1,2}, \Phi_{2,2}$ and $\Phi_{3,2}$, as the independent unknowns, the six linear equations (7.4) comprise a set of linear equations in a matrix form:

$$A\Phi = b \tag{7.5}$$

Note that matrix A is tridiagonal. We know from linear algebra that provided the determinant of A is non zero we have a unique solution. Of course for a special choice of β the determinant of A becomes zero. In this case, our numerical scheme fails. We leave it as an exercise to find the zero determinant condition. Theoretically, we know that for the Laplace equation, we have a uniqueness theorem that asserts that the solution of the Laplace equation with the Dirichlet boundary condition is unique. Our numerical scheme confirms this theorem in the sense that the solution of the matrix equation is unique. Before proceeding, let us pause a bit and make some additional comments. Suppose on some segments of the rectangular box, the Neumann boundary condition holds. For example, assume that on the upper segment, we have $\frac{\partial \Phi}{\partial n} = a_3$. More concisely, we have $\frac{\partial \Phi}{\partial y} = a_3$ on the upper boundary segment. In this case, we do not have directly the function values at the upper grid points. Implementing a backward finite difference scheme, for $N_x = 4, N_y = 3$ we have:

$$\frac{\Phi_{i,3} - \Phi_{i,2}}{\Delta y} = a_3 \quad i = 1, 2, 3 \tag{7.6}$$

This gives:

$$\Phi_{i,3} = \Phi_{i,2} + a_3 \Delta y \quad i = 1, 2, 3 \tag{7.7}$$

This implies that whenever we encounter a function value in the upper segment of the boundary we can replace it with its southward interior unknown function value. The second point I would like to mention is the structure of matrix A. This structure depends on the finite difference scheme we use for the discretisation of spatial derivatives. Instead of a five-point approximation, we can use higher-order approximations such as nine-point formula (Hoffmann and Chiang, 2009):

$$\frac{-\Phi_{i-2,j} + 16\Phi_{i-1,j} - 30\Phi_{i,j} + 16\Phi_{i+1,j} - \Phi_{i+2,j}}{12(\Delta x)^2} +$$

$$\frac{-\Phi_{i,j-2} + 16\Phi_{i,j-1} - 30\Phi_{i,j} + 16\Phi_{i,j+1} - \Phi_{i,j+2}}{12(\Delta y)^2} = 0 \tag{7.8}$$

Note that we cannot apply (7.8) to the grid points adjacent to boundary grids and should use a five-point formula for these points to prevent encountering undefined quantities such as $\Phi_{-1,j}$, etc! You should be

now convinced that implementing a higher-order formula changes the structure of matrix A. Let us now discuss non-Cartesian boundaries. A prototype example is the Dirichlet problem for the circle. Here the region within which we wish to solve the Laplace equation is the interior of a circle of radius R. The appropriate system of coordinates is the polar system. The potential Φ is a function of polar coordinates ρ and θ with the boundary condition $\Phi(R,\theta) = f(\theta)$. The analytical solution turns out to be (Myint-U and Debnath, 2007):

$$\Phi(\rho,\theta) = \frac{1}{2\pi}\int_0^{2\pi} \frac{1-\rho^2}{1-2\rho\cos(\theta-\eta)+\rho^2} f(\eta)d\eta \tag{7.9}$$

The integral is called Poisson integral formula for a circle. Let us now try to solve the Dirichlet problem for a circle numerically. Showing the circular grids by $\Delta\rho$ and $\Delta\theta$ and using the shorthand notation $\Phi(p\Delta\rho, q\Delta\theta)$ by $\Phi_{p,q}$ and noting that the Laplacian operator in circular coordinate becomes:

$$\nabla^2 = \frac{\partial^2}{\partial\rho^2} + \frac{1}{\rho}\frac{\partial}{\partial\rho} + \frac{1}{\rho^2}\frac{\partial^2}{\partial\theta^2} \tag{7.10}$$

We can turn the Laplace equation into a finite difference form:

$$\frac{\Phi_{p+1,q}+\Phi_{p-1,q}-2\Phi_{p,q}}{(\Delta\rho)^2} + \frac{1}{p\Delta\rho}\frac{\Phi_{p+1,q}-\Phi_{p,q}}{\Delta\rho} +$$

$$\frac{1}{p^2(\Delta\rho)^2}\frac{\Phi_{p,q+1}+\Phi_{p,q-1}-2\Phi_{p,q}}{(\Delta\theta)^2} = 0 \tag{7.11}$$

Multiplying both sides of (7.11) by $(\Delta\rho)^2$ we arrive at the following recursive equation:

$$(1+\frac{1}{p})\Phi_{p+1,q} + \Phi_{p-1,q} - 2[1+\frac{1}{2p}+\frac{1}{p^2(\Delta\theta)^2}]\Phi_{p,q} +$$

$$\frac{1}{p^2(\Delta\theta)^2}[\Phi_{p,q+1}+\Phi_{p,q-1}] = 0 \tag{7.12}$$

Denoting the number of radial and angular grids by N_p+1 and N_q+1 respectively the implementation of boundary condition gives:

$$\Phi_{N_p,q} = f_q := f(q\Delta\theta) \tag{7.13}$$

Note that there are N_q+1 boundary grids. We leave it as an exercise to solve the corresponding linear set of equations and find the solution to the Dirichlet problem for a circle.

7.3 Relaxation methods

Returning now to our matrix method, one can use the numeric machinery of solving the set of linear equations to find the solution of the Laplace equation. Among the various methods, people often prefer to use the relaxation method. The method is based on iteration. We guess a solution $\Phi^{(0)}_{i,j}$ and improve it iteratively. Let us briefly explain the procedure. Suppose we do not know the exact solution of a linear system of equations $AX = b$. Let us show the exact solution by X and assume that we have managed to find an approximate solution X'. The relaxation idea is to improve the approximate solution X' iteratively and approach the exact but unknown solution X step by step. Expressing the difference between the approximate answer X' and the exact answer by δX we have:

$$X' = X + \delta X \tag{7.14}$$

Multiplication of (7.14) by A gives:

$$AX' = A(X + \delta X) = AX + A\delta X = b + A\delta X \tag{7.15}$$

This gives the following equation for the unknown vector δX:

$$A\delta X = AX' - b \tag{7.16}$$

Note that the right-hand side of (7.16) is a known vector and by solving (7.16) one can find δX. Therefore the improved solution becomes:

$$X = X' - \delta X \tag{7.17}$$

In practice we can repeat this procedure and iteratively improve the solution:

$$X^{(k+1)} = X^{(k)} - \delta X^{(k)} \tag{7.18}$$

We iterate (7.18) until a specified convergence criterion is met. In practice, we stop (7.18) if the criterion $|X^{(k+1)} - X^{(k)}| < \epsilon$ is satisfied. The desired precision ϵ is at our will. It should be noted that the convergence of (7.18) crucially depends on the initial choice $X^{(0)}$. If it is not properly chosen the series diverges. Unfortunately, there is not a routine prescription for $X^{(0)}$. People normally choose it on physical grounds and intuition.

7.3.1 Jacobi method

Another frequent way of developing a relaxation scheme is to consider the Laplace equation as the steady-state solution of the heat equation. In two dimensions we have:

$$\frac{\partial T(x,y,t)}{\partial t} = D[\frac{\partial^2 T(x,y,t)}{\partial x^2} + \frac{\partial^2 T(x,y,t)}{\partial y^2}] \tag{7.19}$$

Restricting ourselves to closed geometries with a stationary boundary condition we physically know that the solution relaxes to a steady state $T_s(x,y)$. Setting the time derivative in the steady state to zero $\frac{\partial T_s(x,y)}{\partial t} = 0$ the heat equation reduces to:

$$\frac{\partial^2 T_s(x,y,t)}{\partial x^2} + \frac{\partial^2 T_s(x,y,t)}{\partial y^2} = 0 \tag{7.20}$$

Which is the Laplace equation. In other words, we have

$$lim_{t\to\infty} T(x,y,t) = T_s(x,y) \tag{7.21}$$

The above procedure suggests that the solution of the Laplace equation is the long time limit of an artificial heat equation. For the moment we take this equation as follows:

$$\frac{\partial \Phi(x,y,t)}{\partial t} = \lambda[\frac{\partial^2 \Phi(x,y,t)}{\partial x^2} + \frac{\partial^2 \Phi(x,y,t)}{\partial y^2}] \tag{7.22}$$

The constant λ has no physical interpretation and can be absorbed in the time step τ. We already know how to numerically solve equation the (7.22). The FTCS scheme implies:

$$\Phi_{i,j}^{n+1} = \Phi_{i,j}^n + \frac{\lambda\tau}{(\Delta x)^2}[\Phi_{i+1,j}^n + \Phi_{i-1,j}^n - 2\Phi_{i,j}^n] +$$

$$\frac{\lambda\tau}{(\Delta y)^2}[\Phi_{i,j+1}^n + \Phi_{i,j-1}^n - 2\Phi_{i,j}^n] \tag{7.23}$$

where $\Phi_{i,j}^n$ is a shorthand notation for $\Phi(i\Delta x, j\Delta y, n\tau)$. As a matter of fact, $\Phi_{i,j}^n$ can be interpreted as the $n-$th guess for the potential value at grid (i,j). The stability analysis implies that our FTCS algorithm is stable provided the following condition holds:

$$\frac{\lambda\tau}{(\Delta x)^2} + \frac{\lambda\tau}{(\Delta y)^2} \leq \frac{1}{2} \tag{7.24}$$

We leave it an exercise to you to verify (7.24). Taking $\Delta x = \Delta y = 2\sqrt{\lambda\tau}$ the algorithm (7.23) takes the simple form:

$$\Phi_{i,j}^{n+1} = \frac{1}{4}[\Phi_{i+1,j}^{n} + \Phi_{i-1,j}^{n} + \Phi_{i,j+1}^{n} + \Phi_{i,j-1}^{n}] \tag{7.25}$$

This formula is the so-called *Jacobi* method which is our first practical relaxation scheme in solving elliptical PDEs (Garcia, 1999). The Jacobi method resembles the mean-value theorem in electrostatic which states that the potential value at each point is the mean value of potential for every closed counter that encompasses the point. In the Jacobi method, the updated value of the potential at a grid is the average over its four nearest neighbours. Before programming the Jacobi method, I would like to mention that one can develop the Jacobi method without resorting to the heat equation. Starting with the discrete version of Laplace equation (7.3) and introducing $\beta = \frac{\Delta x}{\Delta y}$ we obtain:

$$2(1+\beta^2)\Phi_{i,j} = \Phi_{i+1,j} + \Phi_{i-1,j} + \beta^2\Phi_{i,j+1} + \beta^2\Phi_{i,j-1} \tag{7.26}$$

This gives us a clue to write the following iterative relation:

$$\Phi_{i,j}^{n+1} = \frac{1}{2(1+\beta^2)}[\Phi_{i+1,j}^{n} + \Phi_{i-1,j}^{n} + \beta^2\Phi_{i,j+1}^{n} + \beta^2\Phi_{i,j-1}^{n}] \tag{7.27}$$

where n denotes the iteration number. The above formulation resembles the implicit formalism in which some variables are replaced by their unknown values at the updated time. It is very easy to make a programme that evaluates the potential according to the Jacobi method. The programme `Jacobi` (see Appendix 7.A for details) numerically solves the two-dimensional Laplace equation by the Jacobi method. Let us now solve some simple examples. As the first one, consider a rectangle of side lengths L_x and L_y within which we wish to obtain the potential. The boundary condition is assumed to be of Dirichlet type:

$$\Phi(x,0) = a_1;\ \Phi(L_x,y) = a_2;\ \Phi(x,L_y) = a_3;\ \Phi(0,y) = a_4 \tag{7.28}$$

For simplicity, we take $a_1 = a_2 = a_4 = 0$ and $a_3 = \Phi_0$. This problem is solved by the method of separation of variables in (Garcia, 1999). We only quote the result:

$$\Phi(x,y) = \Phi_0 \sum_{n=1,3,5,\cdots} \frac{4}{n\pi}\sin(\frac{n\pi x}{L_x})\frac{\sinh(\frac{n\pi y}{L_x})}{\sinh(\frac{n\pi L_y}{L_x})} \tag{7.29}$$

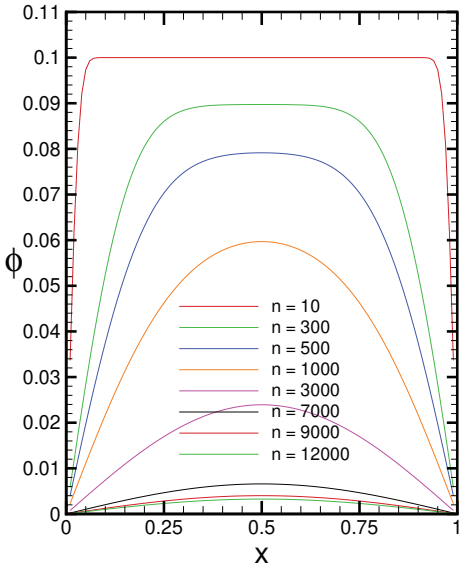

Figure 7.3: *Horizontal profile of potential obtained by the Jacobi method. Dirichlet boundary condition is used. All sides have zero potential except the top one ($y = L_y$) at which the potential is one. Side lengths are $L_x = 1$ and $L_y = 2$. The slow rate of convergence is evident. The number of grid points in x and y directions are $N_x = 100, N_y = 200$ respectively. n denotes the iteration number.*

Figure (7.2) exhibits the numerical result obtained by the Jacobi method. We have taken $L_x = 1, L_y = 2$ and $\Phi_0 = 1$. As for the initial condition, the function value at all the interior points was set to $0.1\Phi_0$. You see the system comes to a steady-state after a large number of iterations around 5000. Remember that you can always speed up the convergence rate by a wiser choice of initial guess of the potential values. See exercises for details. To get a deeper insight, let us now change the boundary conditions and solve the same problem with the only difference that on the top side, the normal derivative of the potential is given. In other words, we consider the Neumann boundary condition in which $\frac{\partial \Phi}{\partial y}(x, L_y)$ is given as the constant a_3.

In this type of boundary condition, we cannot use (7.3) when we are updating the row $j = N_y - 1$ because we do not know a priori the potential at the top side $y = L_y (j = N_y)$. However, we can relate the potential values at the top side row in terms of the potential values at its lower row. This is achieved by using a backward finite difference for the Neumann boundary condition:

$$\frac{\partial \Phi}{\partial y}(x = i\Delta x, L_y) = \frac{\Phi_{i,N_y} - \Phi_{i,N_y-1}}{\Delta y} \tag{7.30}$$

Which gives:

$$\Phi_{i,N_y} = \Phi_{i,N_y-1} + a_3 \Delta y \tag{7.31}$$

Replacing Φ_{i,N_y} by the right-hand side of (7.31) we can now use (7.25) for the grid row corresponding to $j = N_y - 1$. The results turn out to be:

$$\Phi^{n+1}_{i,N_y-1} = \frac{1}{3}[\Phi^n_{i,N_y-2} + \Phi^n_{i-1,N_y-1} + \Phi^n_{i+1,N_y-1} + a_3 \Delta y] \tag{7.32}$$

The algorithm is in our hands now. We iteratively update the grids row by row up to the row $N_y - 1$. After the system becomes stationary the last row (corresponding to $j = N_y$) is updated from its bottom row $j = N_y - 1$ from (7.31). The programme `JacobiNeumann` (see Appendix 7.B for details) implements our explanations to solve the Laplace equation with the Neumann boundary condition. Figure (7.3) exhibits the results of our computations. You see the potential correctly starts from zero at the bottom side to the value with the given normal derivative. I hope that now you can write a programme for a general type of boundary condition. Some exercises are designated for you.

7.3.2 Gauss-Seidel method

We noticed that the convergence rate of the Jacobi method is quite slow. To improve it, we make a little change in the algorithm and use the updated values of potential whenever they become available. This modification, known as the *Gauss-Seidel* method has proven to make notable improvements to the Jacobi method. One way of implementing the Gauss-Seidel (GS) algorithm is the following (Garcia, 1999):

$$\Phi^{n+1}_{i,j} = \frac{1}{4}[\Phi^n_{i+1,j} + \Phi^{n+1}_{i-1,j} + \Phi^n_{i,j+1} + \Phi^{n+1}_{i,j-1}] \tag{7.33}$$

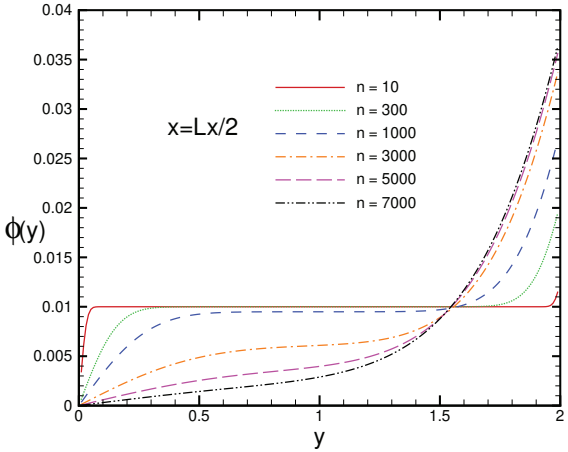

Figure 7.4: *Vertical profile of potential obtained by the Jacobi method with Neumann boundary condition. All sides have zero potential except the top one ($y = L_y$) at which the potential gradient in y direction is set to 0.1. The slow rate of convergence is evident. Number of grid points in x and y directions are $N_x = 100, N_y = 200$ respectively. n denotes the iteration number.*

The implementation of the Gauss-Seidel method has the storage advantage and we no longer need to store both Φ^n and Φ^{n+1} matrices. This gives rise to significant memory saving. When $\Delta x \neq \Delta y$ the GS method takes its general form:

$$\Phi^{n+1}_{i,j} = \frac{1}{2(1+\beta^2)}[\Phi^n_{i+1,j} + \Phi^{n+1}_{i-1,j} + \beta^2\Phi^n_{i,j+1} + \beta^2\Phi^{n+1}_{i,j-1}] \quad (7.34)$$

The programme `GaussSeidel` (see Appendix 7.C for details) numerically solves the Laplace equation by the GS method for the Dirichlet boundary condition. In figure (7.5) we have sketched the same diagram as in figure (7.3) by the GS algorithm. You see that the GS method is faster than the Jacobi method. We can improve the efficiency of the GS method. For this purpose, we start from (7.26) and apply an implicit formulation as follows (Hoffmann and Chiang, 2009):

$$2(1+\beta^2)\Phi^{n+1}_{i,j} - \Phi^{n+1}_{i+1,j} - \Phi^{n+1}_{i-1,j} = \beta^2[\Phi^n_{i,j+1} + \Phi^n_{i,j-1}] \quad (7.35)$$

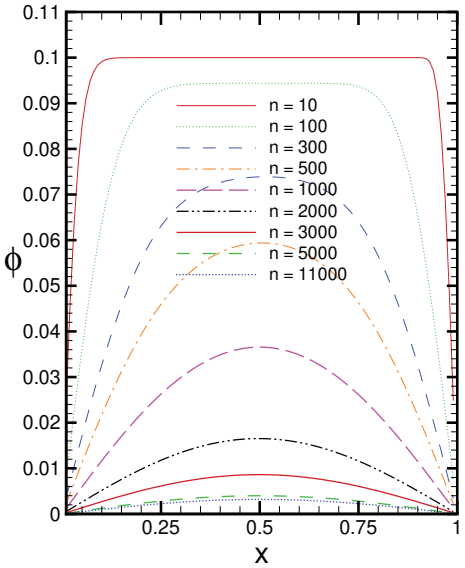

Figure 7.5: *horizontal profile of potential obtained by the Gauss-Seidel method. Dirichlet boundary condition is used. All the sides have zero potential except the top one ($y = L_y$) at which the potential is one. Side lengths are $L_x = 1$ and $L_y = 2$. Number grid points in x and y directions are $N_x = 100, N_y = 50$ respectively. n denotes the iteration number. The steady-state appears after $n = 5000$ iterations.*

This algorithm is known as the *Line Gauss-Seidel* method. As usual we can recast (7.35) in a compact matrix form:

$$A\Phi^{n+1} = \Phi^n \tag{7.36}$$

where A is a tridiagonal matrix. By solving the set of linear algebraic equations (7.36) we can find the updated value of potential Φ^{n+1}. We leave it as an exercise for you to solve the Laplace equation by this implicit GS method. The merit of the implicit GS is two-fold. First, it is more stable, and second, its convergence rate is faster than the explicit GS method. Note that since at each timestep, you solve a set of linear equations, more computation time is needed per iteration.

7.3.3 Simultaneous over relaxation method

It is yet possible to significantly speed up our relaxation algorithm by over-correcting the potential value at each iteration of the GS method. The new method is named *simultaneous over relaxation* or SOR in the literature. Some people use *successive* instead of *simultaneous*. The SOR algorithm is implemented as follows:

$$\Phi_{i,j}^{n+1} = (1-\omega)\Phi_{i,j}^{n} + \frac{\omega}{4}[\Phi_{i+1,j}^{n} + \Phi_{i-1,j}^{n+1} + \Phi_{i,j+1}^{n} + \Phi_{i,j-1}^{n+1}] \quad (7.37)$$

The constant ω is called the over-relaxation parameter. The choice of ω is central to the SOR method. Notice that $\omega = 1$ corresponds to the GS method. It can be shown that for $\omega > 2$ the method becomes unstable (Garcia, 1999). Moreover, for $\omega < 1$ the convergence becomes slow. Thus the acceptable value of ω lies between one and two. The optimal choice depends on the problem geometry. The general formulation of SOR (when $\Delta x \neq \Delta y$) turns out to be (see (Hoffmann and Chiang, 2009) for details):

$$\Phi_{i,j}^{n+1} = (1-\omega)\Phi_{i,j}^{n} + \frac{\omega}{2(1+\beta^2)}[\Phi_{i+1,j}^{n} + \Phi_{i-1,j}^{n+1} +$$

$$\beta^2\Phi_{i,j+1}^{n} + \beta^2\Phi_{i,j-1}^{n+1}] \quad (7.38)$$

you are asked in exercises to solve the Laplace equation with the SOR method. Analogous to the GS method, we can formulate an implicit version of the SOR algorithm. The implicit SOR (line SOR) is formulated as follows:

$$2(1+\beta^2)\Phi_{i,j}^{n+1} - \omega\Phi_{i+1,j}^{n+1} - \omega\Phi_{i-1,j}^{n+1} = \omega\beta^2[\Phi_{i,j+1}^{n} + \Phi_{i,j-1}^{n+1}] +$$

$$2(1-\omega)(1+\beta^2)\Phi_{i,j}^{n} \quad (7.39)$$

7.4 Poisson equation

So far we have looked at the solution of the Laplace equation which is also equivalent to the steady-state equation for heat. Despite restricting ourselves to two dimensions many techniques we introduced can be generalised in a straightforward manner to higher dimensions. The next equation we intend to consider is the inhomogeneous Laplace

equation. In electrostatic we call it *Poisson equation*. In the context of heat transfer, we can regard it as the steady-state equation of a heat transfer problem with a source or sink. For simplicity, we restrict ourselves to two dimensions. The two-dimensional Poisson equation takes the following form in the SI unit:

$$\frac{\partial^2 \Phi(x,y)}{\partial x^2} + \frac{\partial^2 \Phi(x,y)}{\partial y^2} = -\frac{1}{\epsilon_0}\rho(x,y) \tag{7.40}$$

$\rho(x,y)$ is the charged density and ϵ_0 is the free space permittivity. The equation is to be solved within a domain R and must be endowed with appropriate boundary conditions if we want to have a unique solution. Analogous to the Laplace equation there is no existence theorem which means that you may not find a solution for an arbitrary boundary condition. By employing the centred space discretisation scheme for the spatial derivatives the equation takes the following discretised form:

$$\frac{\Phi_{i+1,j} + \Phi_{i-1,j} - 2\Phi_{i,j}}{(\Delta x)^2} + \frac{\Phi_{i,j+1} + \Phi_{i,j-1} - 2\Phi_{i,j}}{(\Delta y)^2} = -\frac{\rho_(i,j)}{\epsilon_0} \tag{7.41}$$

where $\rho_{i,j} = \rho(i\Delta x, j\Delta y)$. We can simply devise a Jacobi relaxation scheme to solve the Poisson equation numerically:

$$\Phi_{i,j}^{n+1} = \frac{1}{4}[\Phi_{i+1,j}^n + \Phi_{i-1,j}^n + \Phi_{i,j+1}^n + \Phi_{i,j-1}^n + \frac{(\Delta x)^2}{\epsilon_0}\rho_{i,j}]. \tag{7.42}$$

Note we have taken $\Delta x = \Delta y$. We can simply generalise the Gauss-Seidel method to the Poisson equation. You should only replace the current iteration number n by $n+1$ in the second and fourth term in the bracket of Eq. (7.42). By a simple change in the programme `Jacobi`, we can numerically solve the Poisson equation. Let us now solve a problem.

Imagine we put a point charge q at the centre of an square of side length L. The potential is zero on three sides located at $x = 0, x = L$ and $y = 0$ and Φ_0 at the top side ($y = L$). Find the potential everywhere inside the square.

For simplicity, we work in a reduced unit in which $\epsilon_0 = 1$. Furthermore, we take $L = \Phi_0 = 1$. According to our notation introduced in (7.28) we have $a_1 = a_2 = a_4 = 0$ and $a_3 = 1$. Moreover,

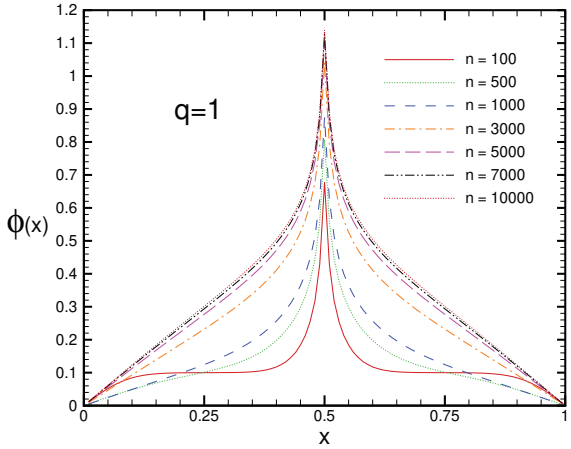

Figure 7.6: *Horizontal profile of potential obtained by the Jacobi method within a square of side length $L = 1$ with a charge $q = 1$ at its centre. Dirichlet boundary condition is used. All the sides have zero potential except the top one ($y = L_y$) at which the potential is one. Grid lengths are $\Delta x = \Delta y = 0.01$. The iteration number is denoted by n. The steady-state appears after $n = 10000$ iterations.*

$\rho_{i,j} = 0$ except for the middle grid $(\frac{N}{2}, \frac{N}{2})$. Notice that N should be an even number. In order to find $\rho_{\frac{N}{2},\frac{N}{2}}$ we make use of the normalisation integral $\int\int_R dxdy\rho(x,y) = q$. Discretising the integral we find: $\rho_{\frac{N}{2},\frac{N}{2}}\Delta x\Delta y = q$ which gives:

$$\rho_{\frac{N}{2},\frac{N}{2}} = \frac{q}{\Delta x\Delta y} \tag{7.43}$$

The programme `Poisson` (see appendix 7.D for details) numerically solves the Poisson equation by the Jacobi relaxation scheme. Figure (7.6) exhibits the results for $q = 1$. You see the convergence rate has decreased in comparison with the Laplace equation. In figure (7.7) we have shown the vertical potential profile in the steady state for various values of q. In exercises, you are asked to solve the Poisson equation with the Neumann boundary condition.

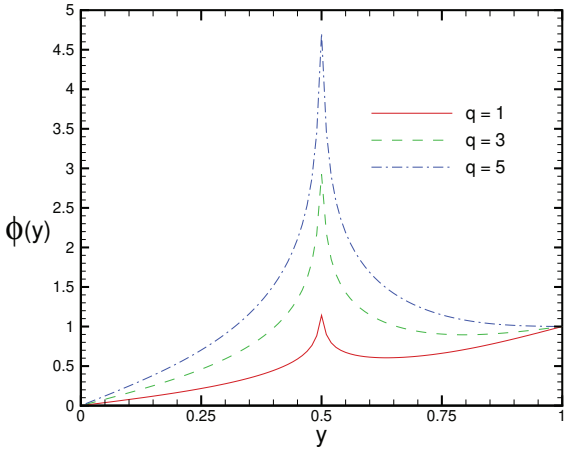

Figure 7.7: *Vertical profile of potential obtained at $x = 0.1L$ by the Jacobi method within a square of side $L = 1$ having a charge q at its centre. All the parameters are the same as in figure (7.6)*

.

7.5 Multiple Fourier transform method

In the last section of this chapter, we discuss a different method that is quite frequently used for solving elliptical PDEs. The method is known as the multiple Fourier transform (MFT) and as the name suggests it employs discrete Fourier transform. You may remember the discrete Fourier transform from chapter four. Here we generalise this concept into higher dimensions. In particular, in two dimensions the discrete Fourier transform of a set of 2D data $x_{j,k} \quad j,k = 1, \cdots, M$ becomes:

$$X_{m,n} = \frac{1}{M} \sum_{j,k=1}^{M} x_{j,k} e^{-\frac{2\pi i}{M}(jm+kn)} \tag{7.44}$$

The inverse Fourier transform becomes:

$$x_{j,k} = \frac{1}{M} \sum_{m,n=1}^{M} X_{m,n} e^{\frac{2\pi i}{M}(jm+kn)}. \tag{7.45}$$

Note that for real-valued $x_{j,k}$ the corresponding $X_{m,n}$ are complex numbers and hence half of them are independent numbers. Having defined the two- dimensional DFT, we now explain the multiple Fourier transform method for a two-dimensional problem. Consider the finite difference form of the Poisson equation (7.41) for a square of side L. For simplicity, we adopt the periodic boundary condition and start our grid numbers from zero as usual. Therefore our unknowns are $\Phi_{j,k}\ \ j,k = 0,1,\cdots,N-1$. According to this boundary condition we have:

$$\Phi_{N,k} = \Phi_{0,k};\quad \Phi_{j,N} = \Phi_{j,0} \tag{7.46}$$

Showing the DFT of the potential by $F_{m,n}$ and that of charge density by $R_{m,n}$ we have:

$$R_{m,n} = \frac{1}{N}\sum_{j,k=0}^{N-1} \rho_{j,k} e^{-\frac{2\pi i}{N}(jm+kn)} \tag{7.47}$$

$$F_{m,n} = \frac{1}{N}\sum_{j,k=0}^{N-1} \Phi_{j,k} e^{-\frac{2\pi i}{N}(jm+kn)} \tag{7.48}$$

We rewrite (7.41) with i replaced by j and j replaced by k (to save i for the imaginary unit number) and then multiply both sides of (7.41) by $e^{-\frac{2\pi i}{N}(jm+kn)}$ and then sum over j,k. We therefore find:

$$\frac{1}{N}\sum_{j,k=0}^{N-1} e^{-\frac{2\pi i}{N}(jm+kn)}[\frac{\Phi_{j+1,k}+\Phi_{j-1,k}-2\Phi_{j,k}}{(\Delta x)^2}+$$

$$\frac{\Phi_{j,k+1}+\Phi_{j,k-1}-2\Phi_{j,k}}{(\Delta y)^2}] =$$

$$-\frac{1}{N}\sum_{j,k=0}^{N-1} e^{-\frac{2\pi i}{N}(jm+kn)}\frac{\rho_{j,k}}{\epsilon_0} \tag{7.49}$$

$$[\frac{e^{\frac{2\pi im}{N}}+e^{\frac{-2\pi im}{N}}-2}{(\Delta x)^2}+\frac{e^{\frac{2\pi in}{N}}+e^{\frac{-2\pi in}{N}}-2}{(\Delta y)^2}]F_{m,n} = -\frac{1}{\epsilon_0}R_{m,n} \tag{7.50}$$

Taking $\Delta x = \Delta y$ gives:

$$[e^{\frac{2\pi im}{N}}+e^{\frac{-2\pi im}{N}}+e^{\frac{2\pi in}{N}}+e^{\frac{-2\pi in}{N}}-4]F_{m,n} = -\frac{1}{\epsilon_0}(\Delta x)^2 R_{m,n} \tag{7.51}$$

Solving for $F_{m,n}$ gives $F_{m,n} = P_{m,n}R_{m,n}$ where:

$$P_{m,n} = \frac{-(\Delta x)^2}{2\epsilon_0[\cos(\frac{2\pi m}{N}) + \cos(\frac{2\pi n}{N}) - 2]} \tag{7.52}$$

Having found the DFT transform of the potential $F_{m,n}$ the potential itself is found via taking the inverse DFT:

$$\Phi_{j,k} = \frac{1}{N}\sum_{m,n=0}^{N-1} F_{m,n}e^{\frac{2\pi i}{N}(jm+kn)} \tag{7.53}$$

In exercises, you are asked to solve the Poisson equation that we solved above by the method of MFT.

7.6 Problems

Problem 7.1 a) Explicitly obtain the matrix A corresponding to set of linear equations (7.4). b) Obtain the determinant of matrix A. c) For what value of β do we have $|A| = 0$?.

Problem 7.2 Numerically solve the Laplace equation for the circle using circular griding and compare your findings to the analytic solution (7.9). Take the boundary value $f(\theta) = 1\ for \leq \theta \leq \pi$ and zero otherwise.

Problem 7.3 By using stability analysis verify the condition (7.24).

Problem 7.4 Using the Jacobi method, obtain the solution of the Laplace equation for a rectangle of sides $L_x = 2$ and $L_y = 1$ with the following mixed boundary condition:

$$\Phi(x,0) = 1;\ \Phi(L_x,y) = 0;\ \frac{\partial\Phi}{\partial y}(x,L_y) = 0.1;\ \Phi(0,y) = 0$$

Sketch the vertical and horizontal profiles.

Problem 7.5 Instead of an initial guess $0.1\Phi_0$ for the potential at the interior grids try other values and compare the convergence speed for various choices. Consider the same Dirichlet we solved in the text.

Problem 7.6 Solve the Laplace equations that we solved in the text when $\beta = \frac{\Delta x}{\Delta y} \neq 1$. Test different values of β and find the choice which has the fastest convergence rate

Problem 7.7 Solve the Laplace equation with the Dirichlet boundary condition by the implicit Gauss-Seidel algorithm and compare its convergence rate with the explicit Gauss-Seidel and SOR methods. Use the same parameters we used in the text.

Problem 7.8 a) Solve the Laplace equation with the Dirichlet boundary condition by the SOR algorithm and compare its convergence rate with the Gauss-Seidel method. b) What is the optimal value of the relaxation parameter ω? c) Draw the dependence of the convergence time versus ω. d) Use another algorithm to find the steady-state solution so that you can find the convergence iteration number. Use the same parameters as in the text.

Problem 7.9 Solve the Poisson equation for a square of side length L with the Neumann boundary condition by both Jacobi and Gauss-Seidel methods. Suppose there is a point charge q in the centre of the square. Take the electric field E_0 on the top edge $y = L$. On all the other sides take it $\Phi = 0$.

Problem 7.10 Solve the Poisson equation for a square of sides L with the periodic boundary condition by the MFT method. Suppose there is a point charge q in the centre of the square. Take the potential $\phi_0 = 1$ on the top side $y = L$. On all the other sides take it $\Phi = 0$. Compare your finding for the same problem with the Dirichlet and Neumann boundary conditions.

Chapter 8

Quantum mechanics

8.1 Introduction

Up to now, we have dealt with classical systems either in discrete or in continuum nature. In this chapter, we turn our attention to quantum systems and will learn how to explore these systems on numerical grounds. For simplicity, we restrict ourselves to one dimension but many of the techniques that will be discussed can be generalised to higher dimensions in a straightforward manner. I assume you have a theoretical quantum mechanics background at the graduate level. We begin our discussion with the milestone of quantum mechanics the so-called Schrödinger equation (C. Cohen-Tannoudji and Laloe, 1992).

$$i\hbar\frac{\partial}{\partial t}\Psi(x,t) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\Psi(x,t) + V(x,t)\Psi(x,t) \tag{8.1}$$

Equation (8.1) seems to be a partial differential equation because it involves partial time and space derivatives. Indeed, it is but take care! The wavefunction Ψ is a complex number. This is the first time in this book we encounter a PDE which involves complex numbers. In fact, the Schrödinger equation does not lie in our classification scheme. We express the complex wavefunction $\Psi(x,t)$ in terms of its real and imaginary part

$$\Psi(x,t) = \Psi_R(x,t) + i\Psi_I(x,t) \tag{8.2}$$

and replace (8.2) in the Schrödinger equation (8.1). After setting the real and imaginary parts of both sides equal to each other we find:

$$\hbar\frac{\partial}{\partial t}\Psi_R(x,t) = -\frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\Psi_I(x,t) + V(x,t)\Psi_I(x,t) \tag{8.3}$$

$$\hbar\frac{\partial}{\partial t}\Psi_I(x,t) = \frac{\hbar^2}{2m}\frac{\partial^2}{\partial x^2}\Psi_R(x,t) - V(x,t)\Psi_R(x,t) \tag{8.4}$$

As you see the Schrödinger equation for the complex function $\Psi(x,t)$ is equivalent to a set of two coupled second-order real-valued PDSs. In equations (8.3) and (8.4) everything is real and we should be able to solve them at least numerically. Technically speaking, the coupled set of equations are initial-boundary value problems and we should specify the initial condition $\Psi(x,0)$ together with the boundary values of the wavefunction Ψ if we want to uniquely solve the problem. To solve the problem numerically, we must replace the partial derivatives with finite difference forms in spatial and temporal grids. We may be tempted to use the explicit FTCS scheme. Let us see if it works.

8.2 Numerical algorithms for Schrödinger equation

8.2.1 FTCS method

We shall begin with the FTCS algorithm. Throughout the chapter, we work in atomic units in which $\hbar = m = 1$. FTCS scheme gives:

$$\frac{\Psi_{R,i}^{n+1} - \Psi_{R,i}^{n}}{\tau} = -\frac{\Psi_{I,i+1}^{n} + \Psi_{I,i-1}^{n} - 2\Psi_{I,i}^{n}}{2(\Delta x)^2} + V_i^n\Psi_{I,i}^{n} \tag{8.5}$$

$$\frac{\Psi_{I,i}^{n+1} - \Psi_{I,i}^{n}}{\tau} = \frac{\Psi_{R,i+1}^{n} + \Psi_{R,i-1}^{n} - 2\Psi_{R,i}^{n}}{2(\Delta x)^2} - V_i^n\Psi_{R,i}^{n} \tag{8.6}$$

where as usual $\Psi_i^n = \Psi(i\Delta x, n\tau)$ and $V_i^n = V(i\Delta x, n\tau)$. It turns out that:

$$\Psi_{R,i}^{n+1} = \Psi_{R,i}^{n} - \frac{\tau}{2(\Delta x)^2}[\Psi_{I,i+1}^{n} + \Psi_{I,i-1}^{n} - 2\Psi_{I,i}^{n}] + \tau V_i^n\Psi_{I,i}^{n} \tag{8.7}$$

$$\Psi_{I,i}^{n+1} = \Psi_{I,i}^{n} + \frac{\tau}{2(\Delta x)^2}[\Psi_{R,i+1}^{n} + \Psi_{R,i-1}^{n} - 2\Psi_{R,i}^{n}] - \tau V_i^n\Psi_{R,i}^{n} \tag{8.8}$$

Let us see if the FTCS works for the simplest case of a free particle for which $V(x,t) = 0$. Setting $V_i^n = 0$ in (8.7) and (8.8) and specifying the initial condition $\Psi_{I,i}^0$ and $\Psi_{R,i}^0$ we can find the wavefunction at the updated timestep $n+1$ in terms of its value at the current timestep n. For the initial condition, we take a cosine modulated Gaussian wave packet centred at the origin (Liboff, 2002):

$$\Psi(x,0) = \frac{1}{\sqrt{\sigma_0\sqrt{2\pi}}} e^{ik_0 x} e^{-x^2/4\sigma_0^2} \tag{8.9}$$

where σ_0 denotes the packet width. Note that this wavefunction gives a normalised probability density function i.e.; $\int_{-\infty}^{\infty} \Psi^*(x,0)\Psi(x,0)dx = 1$. Moreover, you can verify that the expectation value of the particle's velocity becomes $\langle v\rangle = \langle \frac{p}{m}\rangle = \frac{\hbar k_0}{m}$. This nonzero initial expectation value of the momentum is sometimes called *momentum boost* (Liboff, 2002). Due to the momentum boost, it can be shown that even in free space the wave packet spreads in time and widens. At time t the evolved wavefunction becomes (C. Cohen-Tannoudji and Laloe, 1992; Liboff, 2002):

$$\Psi(x,t) = \frac{1}{\sqrt{\sigma_0\sqrt{2\pi}}(1+i\frac{t}{\tau})^{1/2}} e^{i\frac{t}{\tau}(\frac{x}{2\sigma_0})^2} exp[\frac{-i\tau(x-p_0t/m)^2}{4\sigma_0^2 t(1+i\frac{t}{\tau})}] \tag{8.10}$$

in which $\tau = \frac{2m\sigma_0^2}{\hbar}$ and $p_0 = \hbar k_0$. You can verify that $\langle x\rangle = \frac{\hbar k_0 t}{m}$. Since Ψ itself is a complex number and cannot be shown on a graph, we draw the probability density $P(x,t) = |\Psi(x,t)|^2$. From (8.10) we find (Liboff, 2002; Garcia, 1999):

$$P(x,t) = |\Psi(x,t)|^2 = \frac{1}{\sigma_0\sqrt{2\pi}(1+\frac{t^2}{\tau^2})^{1/2}} exp[-\frac{(x-p_0t/m)^2}{2\sigma_0^2(1+\frac{t^2}{\tau^2})}] \tag{8.11}$$

The packet spreads in time with a time-dependent standard deviation $\sigma(t)$:

$$\sigma(t) = \sigma_0\sqrt{1+\frac{t^2}{\tau^2}} \tag{8.12}$$

For large time $t \gg \tau$ we have $\sigma(t) \to \frac{\sigma_0 t}{\tau}$. As you see the wavepacket variance increases linearly with time at large times. The programme `SchroFTCS` (see Appendix 8.A for details) numerically solves the time-dependent Schrödinger equation with the explicit FTCS method. The

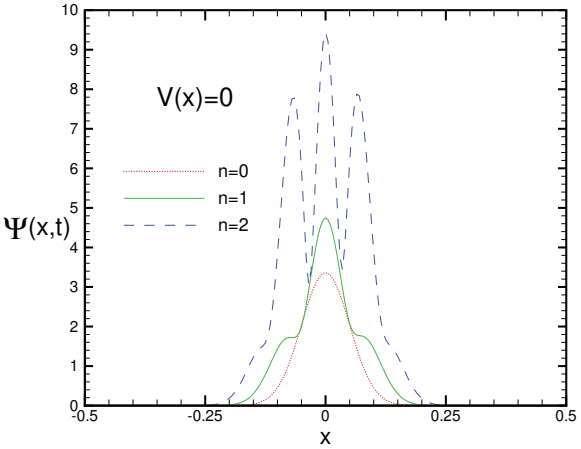

Figure 8.1: *Time evolution of a Gaussian wave packet. FTCS method is used with* $\Delta x = 0.005$ *and* $\tau = 0.005$.

results are shown in figure (8.1) for a Gaussian wave packet initially at $x = 0$. We have taken $\sigma_0 = 0.05$ and $k_0 = 0.2$ which gives rise to a momentum $p_0 = \hbar k_0 = 0.2$. As you can see the wavefunction dramatically grows and distorts in time. You can imagine the catastrophe for larger time steps. In fact, the explicit FTCS scheme is highly unstable. We will see in short that if we use the implicit FTCS algorithm it becomes stable but before that, we would like to discuss an alternative approach to solve the time-dependent Schrödinger equation. The method is called *Visscher*.

8.2.2 Visscher method

In this method, the real part of the wavefunction updates at integer timesteps $n = 1, 2, \cdots$ whereas its imaginary part gets updated at half steps $n = \frac{1}{2}, \frac{3}{2}, \cdots$. The algorithm turns out to be (H. Gould and Chriastian, 2006):

$$\Psi_{R,i}^{n+1} = \Psi_{R,i}^{n} - \frac{\tau}{2(\Delta x)^2}[\Psi_{I,i+1}^{n+\frac{1}{2}} + \Psi_{I,i-1}^{n+\frac{1}{2}} - 2\Psi_{I,i}^{n+\frac{1}{2}}] + \tau V_i^n \Psi_{I,i}^{n+\frac{1}{2}} \quad (8.13)$$

$$\Psi_{I,i}^{n+\frac{3}{2}} = \Psi_{I,i}^{n+\frac{1}{2}} + \frac{\tau}{2(\Delta x)^2}[\Psi_{R,i+1}^{n} + \Psi_{R,i-1}^{n} - 2\Psi_{R,i}^{n}] - \tau V_i^n \Psi_{R,i}^{n} \quad (8.14)$$

In order to run the iteration we need to know $\Psi_{R,i}^{0}$ and $\Psi_{I,i}^{\frac{1}{2}}$. Visscher has shown that the stability condition for his algorithm is:

$$-\frac{2\hbar}{\tau} \leq V(x,t) \leq \frac{2\hbar}{\tau} - \frac{2\hbar^2}{(m\Delta x)^2} \quad (8.15)$$

For the particular case of a free particle $V(x,t) = 0$ the stability condition reduces to:

$$\tau \leq \frac{(m\Delta x)^2}{\hbar} \quad (8.16)$$

To obtain $\Psi_{I,i}^{\frac{1}{2}}$ we use Taylor expansion:

$$\Psi_{I,i}^{\frac{1}{2}} = \Psi_{I,i}(\frac{\tau}{2}) = \Psi_{I,i}(0) + \frac{\tau}{2}\frac{\partial \Psi_{I,i}^{0}}{\partial t} = \Psi_{I,i}(0) +$$

$$\frac{\tau\hbar}{4m(\Delta x)^2}[\Psi_{R,i+1}^{0} + \Psi_{R,i-1}^{0} - 2\Psi_{R,i}^{0}] - \frac{\tau}{2\hbar} V_i^0 \Psi_{R,i}^{0} \quad (8.17)$$

Now we can march forward in time according to equations (8.13) and (8.14). Figure (8.2) shows the time evolution of the real and imaginary parts of the wavefunction. The parameters are $k_0 = 2, \sigma_0 = 1, \Delta x = 0.04, \tau = 0.1$. The wavefunction is initially at $x_0 = -15$. As you see the algorithm is not stable. Let us implement a stable scheme. The scheme we wish to use is the *Crank* algorithm that you are well familiar with it from previous chapters.

8.2.3 Crank method

Now we discuss the application of Crank's method to Schrödinger equation. According to Crank's algorithm we have:

$$i\hbar\frac{[\Psi_i^{n+1} - \Psi_i^n]}{\tau} = -\frac{\hbar^2}{2m(\Delta x)^2}[\Psi_{i+1}^{n+1} + \Psi_{i-1}^{n+1} - 2\Psi_i^{n+1}] + V_i^n \Psi_i^{n+1} \quad (8.18)$$

Note that the right-hand side is evaluated in timestep $n+1$. Let us $\Psi^{(n)}$ store the value of the wavefunction Ψ at spatial grids in timestep

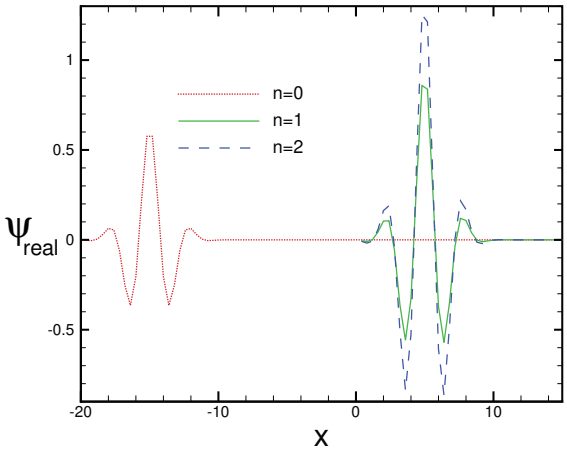

Figure 8.2: *The real part of a Gaussian packet in free space after the first and the second timesteps. Visscher's method has been used.*

n i.e.; $\Psi^{\dagger(n)} = (\Psi_0^n, \Psi_1^n, \cdots, \Psi_N^n)$ where $N+1$ is the the number of spatial grids. The Schrödinger equation in Crank algorithm becomes: $i\hbar\frac{[\Psi^{(n+1)}-\Psi^{(n)}]}{\tau} = H\Psi^{(n+1)}$. This equation can be recast in the following form:

$$(I + \frac{i\tau H}{\hbar})\Psi^{(n+1)} = \Psi^{(n)} \tag{8.19}$$

Given the initial condition at $n = 0$, we can march forward in time to obtain the wavefunction. Notice that (8.19) is formal because it is not a real set of linear equations. In fact (8.19) is a complex set of linear equations or in other words, a complex linear system of equations (see (W. H. Press and Flannery, 2002) chapter two for more details). To solve it we should separate the real and imaginary parts. After the separation we obtain:

$$\Psi_R^{(n+1)} - \frac{\tau H}{\hbar}\Psi_I^{(n+1)} = \Psi_R^{(n)} \tag{8.20}$$

$$\Psi_I^{(n+1)} + \frac{\tau H}{\hbar}\Psi_R^{(n+1)} = \Psi_I^{(n)} \tag{8.21}$$

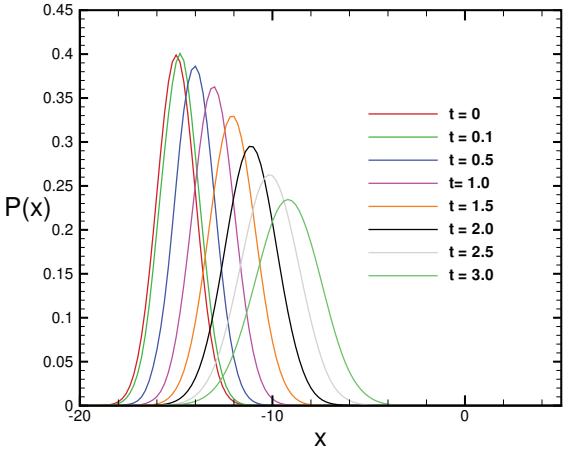

Figure 8.3: *The time evolution of a cosine-modulated Gaussian wave packet by the implicit Crank algorithm. The parameters are:* $N = 200, \sigma_0 = 1, k_0 = 2, \tau = 0.01, L = 40$. *The algorithm is stable and we see the spreading phenomenon.*

To solve this 2 by 2 set of linear equations we substitute $\Psi_I^{(n+1)}$ from (8.21) into (8.20) and arrive at this equation:

$$(I + \frac{\tau^2 H^2}{\hbar^2})\Psi_R^{(n+1)} = \Psi_R^{(n)} + \frac{\tau H}{\hbar}\Psi_I^{(n)} \tag{8.22}$$

Once we know the wavefunction at timestep n we can solve the linear set of equations (8.22) to find the real part of the waverfunction at timestep $n+1$. After finding $\Psi_R^{(n+1)}$ from (8.22) we replace it in (8.21) and find $\Psi_I^{(n+1)}$ by solving the equation $\Psi_I^{(n+1)} = \Psi_I^{(n)} - \frac{\tau H}{\hbar}\Psi_R^{(n+1)}$. The programme `SchroCrank` (see Appendix 8.B for details) numerically solves the time-dependent Schrödinger equation with the Crank method for a system of length L which has $N+1$ grid points. We have set $\Psi(x_{-1})$ and $\Psi(x_{N+1})$ to zero (whenever encountered) to be able to run the iterations. Figure (8.3) shows the evolution of a cosine-modulated Gaussian wave packet in a potential-free space by the Crank method. As you can see the algorithm is stable. The wave packet spreads during the time evolution. Let us look into the problem

from a different viewpoint. We try to solve the initial value problem of Schrödinger equation $i\hbar\frac{\partial\Psi}{\partial t} = H\Psi$ for a free particle ($V(x,t) = 0$). When the potential is zero we can proceed analytically to solve the coupled partial differential equations (8.3) and (8.4). Taking another time derivative from (8.3) and replacing $\frac{\partial\Psi_I(x,t)}{\partial t}$ from (8.4) we arrive at:

$$\frac{\partial^2\Psi_R(x,t)}{\partial t^2} = -\frac{\hbar^2}{4m^2}\frac{\partial^4\Psi_R(x,t)}{\partial x^4} \tag{8.23}$$

analogously we find:

$$\frac{\partial^2\Psi_I(x,t)}{\partial t^2} = -\frac{\hbar^2}{4m^2}\frac{\partial^4\Psi_I(x,t)}{\partial x^4} \tag{8.24}$$

Let $\Phi_R(k,t) =$ denote the Fourier integral of $\Psi_R(x,t)$ that is $\Psi_R(x,t) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} dk\Phi_R(k,t)e^{ikx}$. With this Fourier transformation (8.23) becomes:

$$\frac{\partial^2\Phi_R(k,t)}{\partial t^2} = -\frac{\hbar^2k^4}{4m^2}\Phi_R(k,t) \tag{8.25}$$

The same equation holds for $\Phi_I(k,t)$. Introducing $\Phi(k,t) = \Phi_R(k,t) + i\Phi_I(k,t)$ gives:

$$\frac{\partial^2\Phi(k,t)}{\partial t^2} = -\frac{\hbar^2k^4}{4m^2}\Phi(k,t) \tag{8.26}$$

An integration gives:

$$\Phi(k,t) = \Phi(k,0)e^{i\hbar k^2t/2m} \tag{8.27}$$

The constant $\Phi(k,0)$ turns out to be:

$$\Phi(k,0) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} dx\Psi(x,0)e^{-ikx} \tag{8.28}$$

Replacing $\Psi(x,0)$ from (8.9) into (8.28) gives:

$$\Phi(k,0) = \frac{1}{\sqrt{2\pi}}\int_{-\infty}^{+\infty} dx\frac{1}{\sqrt{\sigma_0\sqrt{2\pi}}}e^{ik_0x}e^{-x^2/4\sigma_0^2}e^{-ikx} \tag{8.29}$$

If you remember we had a similar integral in chapter (5). The integration gives:

$$\Phi(k,0) = \frac{\sqrt{\sigma_0}}{\sqrt{\sqrt{2\pi}}}e^{-\sigma_0^2(k-k_0)^2} \tag{8.30}$$

Putting (8.30) into (8.27) gives:

$$\Phi(k,t) = \frac{\sqrt{2\sigma_0}}{\sqrt{\sqrt{2\pi}}} e^{-\sigma_0^2(k-k_0)^2} e^{i\hbar k^2 t/2m} \tag{8.31}$$

The inverse Fourier integral transform gives:

$$\Psi(x,t) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} dk \Phi(k,t) e^{ikx} \tag{8.32}$$

By replacing (8.31) into (8.32) we find:

$$\Psi(x,t) = \frac{\sqrt{2\sigma_0}}{\sqrt{\sqrt{2\pi}}} \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} dk e^{-\sigma_0^2(k-k_0)^2} e^{i\hbar k^2 t/2m} e^{ikx} \tag{8.33}$$

A few more changes of variables and a Gaussian integration should reproduce the analytic result in (8.10). We leave the details as an exercise to you.

8.2.4 Crank-Nicolson Algorithm

The Crank algorithm can be improved to give more accurate results. We implement the Crank-Nicolson for this purpose. One simply replaces the term $H\Psi^{(n+1)}$ in the Crank algorithm by $\frac{1}{2}[H\Psi^{(n+1)} + H\Psi^{(n)}]$. In other words, we find:

$$i\hbar(\frac{\Psi^{(n+1)} - \Psi^{(n)}}{\tau}) = \frac{1}{2}(H\Psi^{(n+1)} + H\Psi^{(n)}) \tag{8.34}$$

A simplification gives the Crank-Nicolson algorithm in the following form:

$$(I + \frac{i\tau H}{2\hbar})\Psi^{(n+1)} = (I - \frac{i\tau H}{2\hbar})\Psi^{(n)} \tag{8.35}$$

The separation of real and imaginary parts gives the following set of linear equations:

$$\Psi_R^{(n+1)} - \frac{\tau H}{2\hbar}\Psi_I^{(n+1)} = \Psi_R^{(n)} + \frac{\tau H}{2\hbar}\Psi_I^{(n)} \tag{8.36}$$

$$\Psi_I^{(n+1)} + \frac{\tau H}{2\hbar}\Psi_R^{(n+1)} = \Psi_I^{(n)} - \frac{\tau H}{2\hbar}\Psi_R^{(n)} \tag{8.37}$$

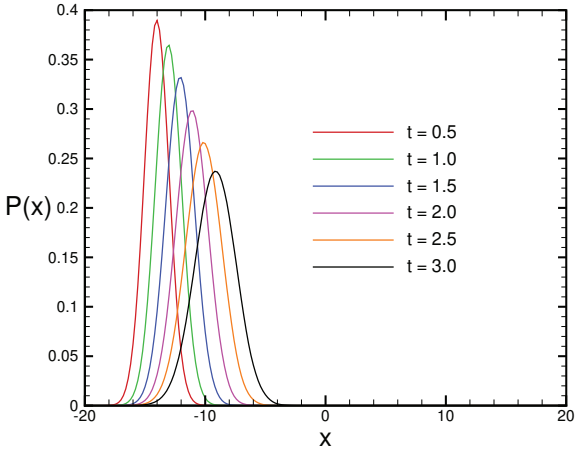

Figure 8.4: *The time evolution of a cosine-modulated Gaussian wave packet in the Crank-Nicolson algorithm. The algorithm is stable and we see the spreading phenomenon. The parameters are identical to those in figure (8.3).*

To solve this 2 by 2 system of linear equations we substitute $\Psi_I^{(n+1)}$ from (8.37) into (8.36) and arrive at this equation:

$$(I + \frac{\tau^2 H^2}{4\hbar^2})\Psi_R^{(n+1)} = (I - \frac{\tau^2 H^2}{4\hbar^2})\Psi_R^{(n)} + \frac{\tau H}{\hbar}\Psi_I^{(n)} \tag{8.38}$$

Once we know the wave function at timestep n we can solve the linear set of equations (8.38) to find the real part of the wave function at timestep $n+1$. After finding $\Psi_R^{(n+1)}$ from (8.38) we replace it in (8.37) and find $\Psi_I^{(n+1)}$ by solving $\Psi_I^{(n+1)} = \Psi_I^{(n)} - \frac{\tau H}{2\hbar}\Psi_R^{(n+1)} - \frac{\tau H}{2\hbar}\Psi_R^{(n)}$. The programme `SchroCrankNicol` (see Appendix 8.C for details) numerically solves the time-dependent Schrödinger equation with the Crank-Nicolson method for a system of length L with $N+1$ grid points. We set $\Psi(x_{-1})$ and $\Psi(x_{N+1})$ (whenever encountered) to zero to be able to run the iteration. Figure (8.4) shows the evolution of a cosine-modulated Gaussian wave packet in a potential-free space by the Crank-Nicolson algorithm. We end this section by noting that the Crank-Nicolson algorithm resembles the *Pàde* approximation for the

exponential e^{-z}:

$$e^{-z} \approx \frac{1-z}{1+z} \tag{8.39}$$

The Crank-Nicolson algorithm can be alternatively written as follows (see Eq. (8.35)):

$$\Psi^{(n+1)} = (I + \frac{i\tau H}{2\hbar})^{-1}(I - \frac{i\tau H}{2\hbar})\Psi^{(n)} \tag{8.40}$$

On the other hand, we know that in quantum mechanics the time evolution operator is $U(t,0) = e^{\frac{-iHt}{\hbar}}$. Applying the *Pàde* approximation for short times formally gives:

$$U(t,0) = \frac{I - \frac{iHt}{\hbar}}{I + \frac{iHt}{\hbar}} \tag{8.41}$$

You see the correspondence. Moreover, only the Crank-Nicolson preserves the unitary property of the evolution operator. Neither the FTCS nor the Crank algorithms preserve this property.

8.3 Expectation values

So far we have managed to compute the evolution of wavefunction provided that its value is given at $t = 0$. However, the main task of quantum mechanics i.e.; the evaluation of operators' expectation values is not fulfilled yet. Here we wish to discuss this point from a numerical viewpoint. For a general operator $\hat{A}$ that is a function of coordinate x and momentum p the expectation value is given:

$$\langle \hat{A}(x,p) \rangle = \langle \hat{A}(x, \frac{\hbar}{i}\frac{\partial}{\partial x}) \rangle = \frac{\int \Psi^*(x,t) A(x, \frac{\hbar}{i}\frac{\partial}{\partial x}) \Psi(x,t) dx}{\int \Psi^*(x,t)\Psi(x,t)dx} \tag{8.42}$$

Once we compute the wave function at time and space grids we can numerically compute the integrals in (8.42). Let us do it for a simple case when $\hat{A} = \hat{p}$. The denominator integral is simply approximated by:

$$\int \Psi^*(x,t)\Psi(x,t)dx = \sum_{j=0}^{N} \Delta x[(\Psi^n_{R,j})^2 + (\Psi^n_{I,j})^2] \tag{8.43}$$

in which $t = n\tau$ and $x = j\Delta x$ as usual. For $\hat{A} = \hat{p}$ the numerator integral becomes:

$$\langle \hat{p} \rangle = \int \Psi^*(x,t) \frac{\hbar}{i} \frac{\partial}{\partial x} \Psi(x,t) dx \tag{8.44}$$

After expressing Ψ by its real and imaginary parts, (8.44) becomes:

$$\langle \hat{p} \rangle = \frac{\hbar}{i} \int [\Psi_R \frac{\partial \Psi_R}{\partial x} + \Psi_I \frac{\partial \Psi_I}{\partial x} + i(\Psi_R \frac{\partial \Psi_I}{\partial x} - \Psi_I \frac{\partial \Psi_R}{\partial x})] dx \tag{8.45}$$

The integrals of the first two terms vanish because it equals $[\Psi_R^2(x,t) + \Psi_I^2(x,t)]_{-\infty}^{+\infty}$ and all the wave functions vanish at $x = \pm\infty$. The remaining integral becomes:

$$\langle \hat{p} \rangle = \hbar \int [\Psi_R \frac{\partial \Psi_I}{\partial x} - \Psi_I \frac{\partial \Psi_R}{\partial x}] dx \tag{8.46}$$

We approximate this integral by a rectangular method:

$$\langle \hat{p} \rangle = \hbar \sum_{j=0}^{N} \Delta x [\Psi_{R,j}^n (\frac{\Psi_{I,j+1}^n - \Psi_{I,j}^n}{\Delta x}) - \Psi_{I,j}^n (\frac{\Psi_{R,j+1}^n - \Psi_{R,j}^n}{\Delta x})] \tag{8.47}$$

It is simplified as follows:

$$\langle \hat{p} \rangle = \hbar \sum_{j=0}^{N} [\Psi_{R,j}^n \Psi_{I,j+1}^n - \Psi_{I,j}^n \Psi_{R,j+1}^n] \tag{8.48}$$

Do not forget to replace $\Psi_{R,N+1}^n$ with $\Psi_{R,0}^n$ and $\Psi_{I,N+1}^n$ with $\Psi_{I,0}^n$that if you have periodic boundary condition. Otherwise, for nonperiodic boundary conditions, you can use a backward finite difference scheme (when necessary) to avoid dealing with variables at non-existing grids $i = -1$ and $i = N + 1$.

8.4 Wavepacket evolution in a potential

In this section, we shall consider the solution of wave packet propagation in a nonzero potential region. As our first example let us solve the incidence of a wavepacket on a potential step. Consider a step potential of height V_0 which begins at $x = 0$. See figure (8.5) for illustration. We take $\Psi(x,0)$ a Gaussian wavepacket centred at $x_0 = -10$.

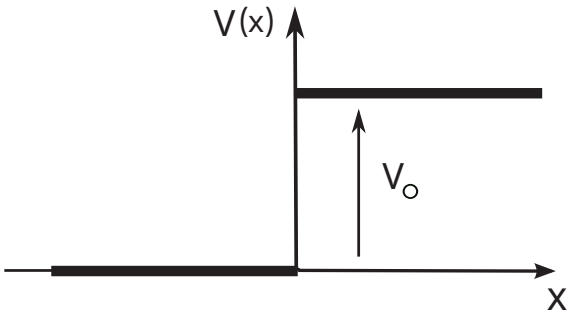

Figure 8.5: *A cosine-modulated Gaussian wave packet is incident from the left onto a step potential.*

The other parameters are $L = 40, k_0 = 2, \sigma_0 = 1$. Moreover, we have taken $N = 200$ and $\tau = 0.01$ and used *Crank's* algorithm. Figure (8.6) shows the evolution of a Gaussian cosine modulated wave packet incident on a step potential of height $V_0 = 2$. As you see the packet shape drastically deviates from Gaussian when it reaches the vicinity of the step i.e.; $x = 0$. At $x = 0$ the packet becomes distorted and undergoes fluctuations. Let us see the packet evolution for longer times. Figure (8.7) shows this evolution. The separation of the wave packet on both sides of the step indicates the passage possibility of the particle from the step potential. The time the peak of the reflected wave reaches the initial position of the incident packet is roughly $t_r \approx 19$ (in reduced units). As can be seen from the figure (8.6) it almost takes $t_i \approx 7$ units of time for the incident wave packet to reach the barrier. Our simulation shows that $t_i \neq t_r$. The reason is the absence of symmetry in potential. We recall that when the initial wavefunction is a cosine travelling wave with energy $E > V_0$ the transmission and reflectance coefficients T and R become (Eisberg and Resnick, 1974):

$$R = 1 - T = (\frac{1 - \sqrt{1 - V_0/E}}{1 + \sqrt{1 - V_0/E}})^2 \tag{8.49}$$

Note the wavepacket energy is roughly $\langle E \rangle = \frac{\hbar^2 k_0^2}{2m}$. In our atomic unit, we have $\langle E \rangle \approx 2$. This value is of the order of step energy V_0 therefore we have a notable reflection probability. We have repeated the analysis for a higher step potential. Figure (8.8) shows the evolution of wave packet for a step of height $V_0 = 10$. The parameters of

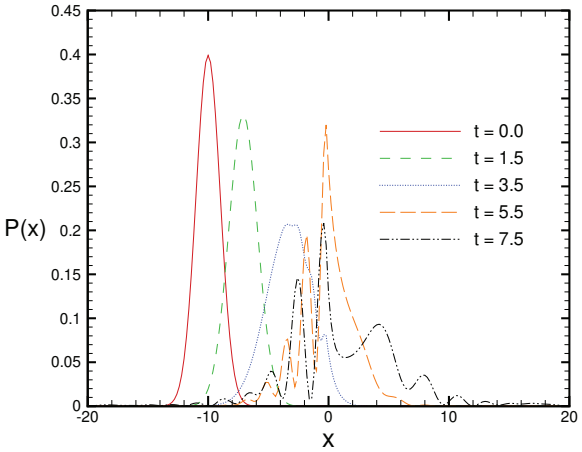

Figure 8.6: *The time evolution of a cosine-modulated Gaussian wave packet incident on a step potential of height* $V_0 = 2$.

the incident Gaussian packet remain the same. As you see almost all of the wave packet is reflected. The transmission probability is very small. The reason is due to the large height of the step potential. The correspondence to classical mechanics is more evident. Note the wavepacket energy is roughly $\langle E \rangle = \frac{\hbar^2 k_0^2}{2m} = 2$. This value is less than the step height and therefore the transmission probability becomes small. In exercises, you are asked to obtain the energy expectation value of the initial wavepacket.

8.5 Time independent Schrödinger eq.

So far we have been considering time-dependent Schrödinger equation. You know well that notable information can be obtained by solving the time-independent Schrödinger equation. It should not be forgotten that this equation can only be considered if the potential does not depend on time. In this case, we can use the method of separation of variables and write the wave function as $\Psi(x,t) = \phi(x)\psi(t)$. Inserting this in the Schrödinger equation (8.1) implies $\psi(t) = Ce^{-\frac{iEt}{\hbar}}$ where

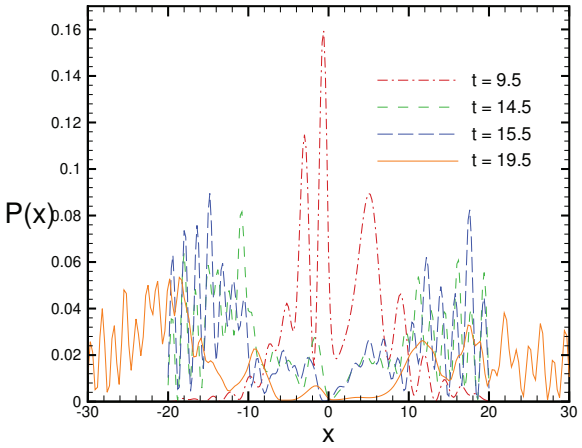

Figure 8.7: *The time evolution of a cosine-modulated Gaussian wave packet incident on a step potential of height $V_0 = 2$ for a longer time. Reflected and transmitted waves are apparent.*

the energy E is the separation constant. Constant C will be determined by the normalisation condition and can be set to one at the moment. Furthermore, the differential equation governing the wavefunction $\phi(x)$ becomes:

$$-\frac{\hbar^2}{2m}\frac{d^2\phi(x)}{dx^2} + V(x)\phi(x) = E\phi(x) \tag{8.50}$$

The function $\phi(x)$ and energy E are called eigenfunction, or interchangeably eigenstate, and eigenvalue respectively. Note that both the eigenfunction and eigenvalue are unknown. We should be able to find them by solving (8.50) provided appropriate boundary conditions are given. Equation (8.50) with the given boundary conditions on $\phi(x)$ is in effect an eigenvalue-boundary problem and is mathematically classified as the *Sturm-Liouville* problem. For the existence and uniqueness of the problem consult mathematical physics textbooks. An excellent one that I can suggest is (Myint-U and Debnath, 2007). In principle, there are many (actually infinite) eigenstates/eigenvalues for a Sturm-Liouville problem. Once you find all the eigenvalues E_n

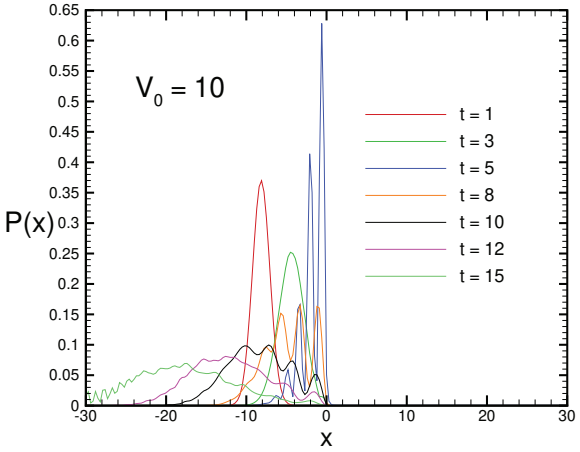

Figure 8.8: *The time evolution of a cosine-modulated Gaussian wave packet incident on a step potential of height $V_0 = 10$. The transmitted wave has a tiny amplitude.*

and their associated eigenfunctions $\phi_n(x)$, the wavefunction $\Psi(x,t)$ can be written as follows (thanks to the superposition principle).

$$\Psi(x,t) = \sum_n c_n \phi_n(x) e^{-\frac{iE_n t}{\hbar}} \tag{8.51}$$

The sum sign is formal and may take an integral form if the eigenfunctions/values are continuous in nature. Coefficients c_n are given by the orthogonality property of the Sturm-Liouville eigenfunctions.

$$c_n = \int \phi_n^*(x)\Psi(x,0)dx \tag{8.52}$$

8.5.1 Step function potential

In this section, we discuss the numerical solution of the Schrödinger equation for some well-known potentials. Let us begin with the step function potential. Assume the step is located at $x = 0$ with height $V_0 > 0$. More concisely, we have:

$$V(x) = 0 \quad x < 0; \quad V(x) = V_0 \quad x \geq 0 \tag{8.53}$$

First, we prove there exist no negative eigenvalues $E < 0$. To see this, suppose there is an eigenstate with a negative eigenvalue. In the left region $x < L$ we have: $\phi_L(x) = Ae^{\kappa x} + Be^{-\kappa x}$ in which $\kappa = \frac{\sqrt{2m|E|}}{\hbar}$. In the right region we have: $\phi_R(x) = Ce^{\xi x} + De^{-\xi x}$ where $\xi = \frac{\sqrt{2m(V_0+|E|)}}{\hbar}$. The finiteness of the wavefunction at $\pm\infty$ implies $B = C = 0$. Continuity of the wavefunction and its derivative at $x = 0$ gives:

$$A = D; \quad \kappa A = -\xi D \tag{8.54}$$

Equation (8.54) gives $\kappa = -\xi$. This implies that a positive quantity κ is equal to a negative quantity $-\xi$ which is an inconsistency. Therefore we conclude there is no negative energy eigenvalue for this problem. For positive values of E, three cases can be identified: $E < V_0, E > V_0$, and $E = V_0$. For the first case i.e.; when the energy eigenvalue is less than the step height the analytical solution becomes (see reference (Eisberg and Resnick, 1974) for details):

$$\phi(x) = D\cos kx - D\frac{K}{k}\sin kx \quad x \leq 0 \tag{8.55}$$

$$\phi(x) = De^{-Kx} \quad x \geq 0 \tag{8.56}$$

in which $k = \frac{\sqrt{2mE}}{\hbar}$ and $K = \frac{\sqrt{2m(V_0-E)}}{\hbar}$. Note that the wave function is real. In the second case where $E > V_0$ the wavefunction turns out to be:

$$\phi(x) = Ae^{ikx} + A\frac{k-K}{k+K}e^{-ikx} \quad x < 0 \tag{8.57}$$

$$\phi(x) = A\frac{2k}{k+K}e^{iKx} \quad x \geq 0 \tag{8.58}$$

Here $K = \frac{\sqrt{2m(E-V_0)}}{\hbar}$. Notice the wave function does not seem to be real and includes an imaginary part. This is true but do not forget that we have put the amplitude of the left coming wave e^{-iKx} equal to zero. If we had not done this then it would be possible to choose the unimportant complex constant A in such a way that the whole wavefunction becomes real. Finally, we consider the third case which is $E = V_0$. In this case, the Schrödinger equation in the right region

reduces to $\phi''(x) = 0$. The solution is linear in $x : \phi_R(x) = Cx + D$. The finiteness of the wavefunction at $x \to +\infty$ implies $C = 0$. In the left region we have: $\phi_L(x) = Ae^{ikx} + Be^{-ikx}$. Continuity of ϕ and its derivative at $x = 0$ gives:

$$A + B = D;\ ik(A - B) = 0 \tag{8.59}$$

Equation (8.59) gives $A = B$. Consequently, the wave function becomes:

$$\phi_L(x) = 2A\cos kx;\ \phi_R(x) = 2A \tag{8.60}$$

We have managed to find all the eigenfunctions and their associated eigenvalues. In some books, you see that the unspecified constants are obtained from the normalisation of the wave function. This task cannot be fulfilled if the space is infinite. Hopefully, the unspecified constants do not appear in the expectation values because they cancel out from the numerator and the denominator of any quantity expectation values. As you see, since the potential is extended in the entire space we have a continuum of energy eigenvalues. Let us now see if we can reproduce these analytical results by computer or not. The answer is yes and the algorithm is very simple: Select a value for E and then turn the time-independent equation (8.50) into a discretised form and then march towards left and right to find the eigenfunctions. More precisely, we recast (8.50) in a finite difference form:

$$-\frac{\hbar^2}{2m}\frac{\phi_{i+1} + \phi_{i-1} - 2\phi_i}{(\Delta x)^2} + V_i\phi_i = E\phi_i \tag{8.61}$$

where as usual $\phi_i = \phi(i\Delta x)$ and $V_i = V(i\Delta x)$. Equation (8.61) gives:

$$\phi_{i+1} = 2\phi_i[1 - \frac{m(\Delta x)^2}{\hbar^2}(E - V_i)] - \phi_{i-1} \quad i = 1, 2, \cdots \tag{8.62}$$

Equation (8.62) is a two-step algorithm and we should specify both ϕ_0 and ϕ_1 to iterate. Let $i = 0$ correspond to the origin $x = 0$. The grid numbers $i = 1, 2, \cdots$ give the right region $x > 0$ whereas $i = -1, -2, \cdots$ gives the left region $x < 0$. Actually (8.62) is suitable for obtaining the eigenvalues in the right region. For the left region we change i into $i - 1$ in (8.61). Expressing ϕ_{i-2} in terms of ϕ_{i-1} and ϕ_i gives:

$$\phi_{i-2} = 2[1 - \frac{m(\Delta x)^2}{\hbar^2}(E - V_{i-1})]\phi_{i-1} - \phi_i \quad i = 0, -1, \cdots \tag{8.63}$$

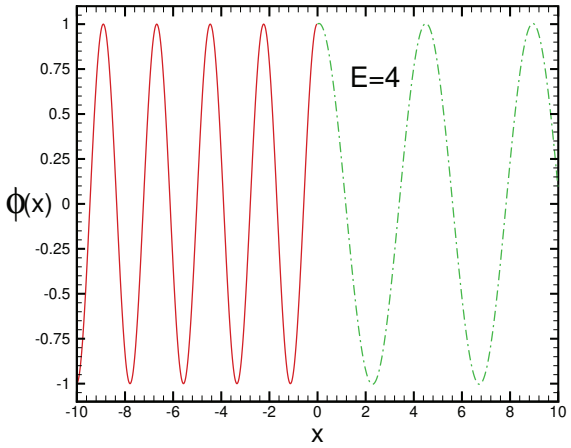

Figure 8.9: *Wave function for the step function potential of height $V_0 = 3$. The energy is set to $E = 4$. The wavefunction is sinusoidal in both left and right regions. The wavelength is larger in the right region $x > 0$. The space grid is set to $\Delta = 0.04$.*

You see that if we want to iterate backward, we need to know ϕ_0 and ϕ_{-1}. We can assign arbitrary values to ϕ_0 and ϕ_1. Continuity of wave function derivative at $x = 0$ gives ϕ_{-1} in terms of ϕ_{+1}. Setting the forward finite difference of $\frac{d\phi(0^+)}{dx} = \frac{\phi_1 - \phi_0}{\Delta x}$ equal to the backward finite difference of $\frac{d\phi(0^-)}{dx} = \frac{\phi_0 - \phi_{-1}}{\Delta x}$ gives:

$$\phi_{-1} = 2\phi_0 - \phi_1 \tag{8.64}$$

The numerical result is shown in figure (8.9). The step height is $V_0 = 3$ and the energy has been set at $E = 4$. We have chosen $\phi_0 = 1$ and $\frac{d\phi(0)}{dx} = 0.1$. The wave numbers k and K are 2.83 and 1.41 respectively. They give the wavelengths $\lambda_L = \frac{2\pi}{k} = 2.22$ and $\lambda_R = \frac{2\pi}{K} = 4.45$ respectively. The agreement between theoretical results and computational ones is very appealing. In fact, $\frac{d\phi(0)}{dx}$ determines the wavefunction amplitude. Figure (8.10) shows the wavefunction for some different choices of the wavefunction's derivative $\frac{d\phi(0)}{dx}$. Now let us see what happens to the wavefunction if the energy eignvalue E is

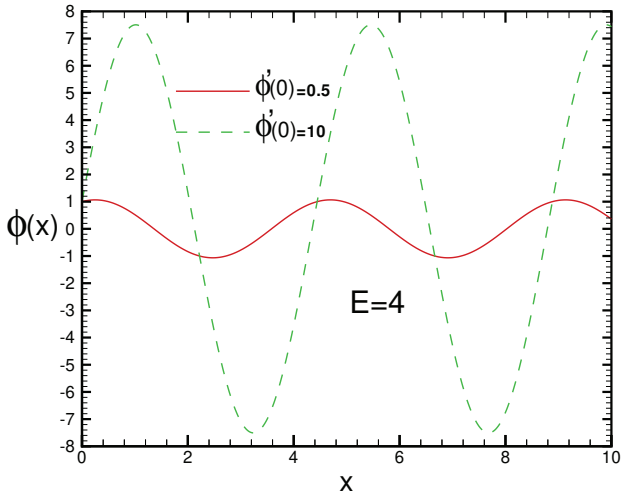

Figure 8.10: *Computed wave function for two different values of $\phi'(0)$. The energy is set to $E = 4$ and other parameters are identical to those in figure (8.9). As you see the wave number is the same for the two values.*

smaller than V_0. Figure (8.11) shows the computed wavefunctions for eigenenergies $E = 2$ and $E = 2.5$ respectively. You can see the substantial difference with the case $E > V_0$. In fact, in the classically forbidden region ($x > 0$) we have seemingly exponential growing wavefunctions! Our expectation from physics is to have exponentially decreasing wavefunctions but do not be disappointed! The reason is that we have theoretically learnt, in our quantum mechanic course, to set, by hand, the coefficient of the growing term $e^{\kappa x}$ to zero on physical grounds. However, a computer does not understand physics and simply computes both $e^{\kappa x}$ and $e^{-\kappa x}$ terms. The computed wavefunction contains both exponentially increasing and decreasing solutions. As you see from the figure (8.11) when E approaches V_0 from below, the wavefunction in the classically forbidden region decreases, and in the limit $E \to V_0$ the solution becomes linear in x which is in good agreement with the analytical result. Figure (8.12) illustrates this behaviour.

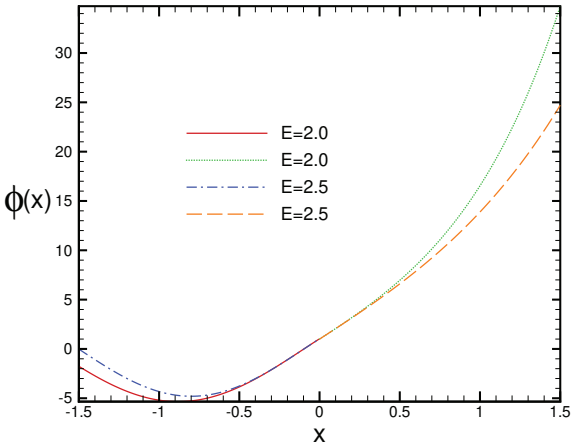

Figure 8.11: *Computed wavefunctions in a step potential of height* $V_0 = 3$ *for* $E = 2$ *and* $E = 2.5$.

8.5.2 Infinite square well

The potentials we have considered so far cannot confine the particle. In other words, the particle wavefunction is an extended function, and the particle can be found at any location of the space with any positive continuum energy eigenvalue E. We now consider the confining potentials under which the particle can only be found in a finite region of space. These states are called bound states in quantum mechanics. The typical example is an infinite square well with the following potential:

$$\begin{cases} V(x) = 0 \quad for\ |x| < a, \\ V(x) = +\infty \quad for\ |x| \geq a. \end{cases} \tag{8.65}$$

The well's width is $2a$. See the schematic in figure (8.13). The wavefunction should be zero outside the well's region $|x| \geq a$. Inside the well the potential is zero and we have:

$$\phi(x) = Ae^{ikx} + Be^{-ikx} \tag{8.66}$$

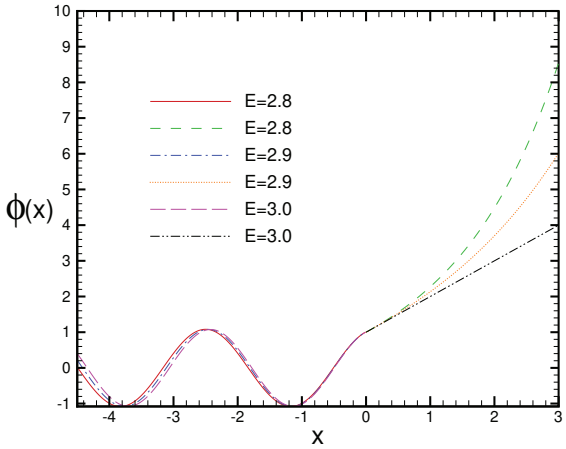

Figure 8.12: *Computed wave functions for a step potential of height $V_0 = 3$ for some eigenvalues E close to V_0.*

where $k = \frac{\sqrt{2mE}}{\hbar}$. The continuity of wavefunction at the well's boundaries $x = \pm a$ gives $\phi(-a) = \phi(a) = 0$ therefore we have:

$$Ae^{-ika} + Be^{ika} = 0; \quad Ae^{ika} + Be^{-ika} = 0 \tag{8.67}$$

Equations (8.67) comprise a set of homogeneous linear equations in terms of unknowns A and B. The determinant of the 2×2 matrix of coefficients turns out to be $e^{-2ika} - e^{2ika}$. To have a nontrivial solution, we must set the determinant equal to zero. This gives $e^{-2ika} = e^{2ika}$ or in other words $e^{ika} = \pm e^{-ika}$. Two distinctive cases arise. For the positive sign, we have $e^{2ika} = 1$ which implies that $2ka$ should be an integer multiple of 2π i.e.; $k_n = n\frac{\pi}{a} \quad n = 0, 1, 2, \cdots$. The quantised energy eigenvalues become $E_n = \frac{k_n^2 \hbar^2}{2m} = n^2 \frac{\hbar^2 \pi^2}{2ma^2} = (2n)^2 \frac{\hbar^2 \pi^2}{8ma^2}$. Moreover, $e^{-ika} = e^{ika}$ gives $A + B = 0$ (see Eq. (8.67)). Equation (8.66) in turn gives $\phi_n(x) = A \sin k_n x = A \sin \frac{2n\pi x}{2a}$. You see that the eigenfunction has odd parity. Remind you from quantum mechanics that when the potential is symmetric (like the present case), its eigenfunctions have a definite parity. The normalisation of this odd parity eigenstate gives the normalisation coefficient $A = \frac{1}{\sqrt{a}}$. In the second

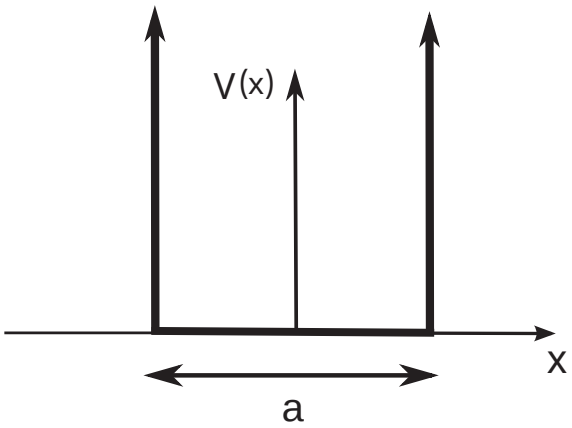

Figure 8.13: *Schematic representation of a symmetric infinite well of width* $2a$.

case we have $e^{ika} = -e^{-ika}$ or equivalently $e^{2ika} = -1$. This implies $2ka$ must be an odd integer multiple of π i.e.; $2ka = (2n+1)\pi$ or $k_n = (2n+1)\frac{\pi}{2a}$ $n = 0, 1, 2, \cdots$. The quantised energy eigenvalues become $E_n = \frac{k_n^2\hbar^2}{2m} = (2n+1)^2\frac{\hbar^2\pi^2}{8ma^2}$. Furthermore, $e^{ika} = -e^{-ika}$ gives $A = B$ (see Eq. (8.67)). Equation (8.66) gives the even eigenstate $\phi_n(x) = A\cos k_n x = A\cos\frac{(2n+1)\pi x}{2a}$. This eigenfunction has even parity. Normalisation gives the coefficient $A = \frac{1}{\sqrt{a}}$. To summarize, the eigenstates are either odd or even. They have well-defined parity because the potential is symmetric. The odd parity eigenstates (normalised) are:

$$\phi_n(x) = \frac{1}{\sqrt{a}}\sin\frac{n\pi x}{2a} \quad n = 2, 4, \cdots \tag{8.68}$$

The even parity eigenstates (normalised) are:

$$\phi_n(x) = \frac{1}{\sqrt{a}}\cos\frac{n\pi x}{2a} \quad n = 1, 3, \cdots \tag{8.69}$$

Remind you that we can write the eigenvalue as:

$$E_k = \frac{k^2\hbar^2\pi^2}{8ma^2} \tag{8.70}$$

where $k = 2n$ for the even states and $k = 2n + 1$ for the odd ones. Eventually, we draw your attention to the intricate point that the wavefunction derivatives are not continuous at the wall positions i.e.; at $x = \pm a$. Despite we see a contradiction to the quantum mechanics postulate but we should not worry. In fact, this discrepancy roots in our unphysical assumption of allowing infinities in the potential. Let us now solve the problem numerically. We will develop three different numerical methods. First, we introduce the *shooting method*.

Method one: shooting method

In this method, the first thing that attracts your attention is probably its name. The name roots in the projectile motion. If you require a cannonball to hit a target on the ground, you have to properly adjust the firing velocity vector. In a mathematical sense, the time-independent Schrödinger equation is a linear second-order differential equation and one needs two initial conditions $\phi(0)$ and $\phi'(0)$ to find the solution. These two quantities play the role of cannonball initial position and velocity. The unknown energy E must be found (by try and error) with the requirement that the wavefunction becomes zero at the right boundary i.e.; $\phi(+a) = 0$. We provisionally take $\phi(0) = \phi_0 = 1$ for the even eigenstates. Of course, after obtaining the eigenstate (unnormalised) we will normalise it by multiplication at a constant factor. Even parity allows us to determine ϕ_1. To see this note that the iteration rule (8.62) gives:

$$\phi_1 = 2\phi_0 - \phi_{-1} - \frac{2m}{\hbar^2}(\Delta x)^2(E - V_0)\phi_0 \tag{8.71}$$

Even parity implies $\phi_{-1} = \phi(-\Delta x) = \phi(\Delta x) = \phi_1$. Putting this into (8.71) we obtain $\phi_1 = \phi(0)[1 - \frac{m(\Delta x)^2}{\hbar^2}(E - V_0)]$ and consequently we can iterate. The essence of the shooting method is to make an initial guess for E and construct the wavefunction up to the last grid $i = N$. If ϕ_N is zero then our choice is the energy. Otherwise, which is normally the case, we have to increase or decrease E and repeat the procedure until the condition $\phi_N = 0$ is satisfied within a given tolerance. The main issue is how to adjust E. Here the best strategy is to define a trend. In our problem, E should be varied in such a direction to decrease the wavefunction deviation from its boundary value. Suppose $|\phi_N^{(k)} - \phi(a)|$ denotes the absolute value of differ-

ence between the computed wavefunction at the last grid $\phi_N^{(k)}$ with its desired one $\phi(a)$ in the kth step of energy adjustment. The new energy guess $E^{(k+1)}$ should decrease this difference. Having defined an energy increment δE we obtain $|\phi_N^{(k+1)} - \phi(a)|$ for both new energy trials $E^{(k+1)} = E^{(k)} \pm \delta E$. The one which give a smaller value $|\phi_N^{(k+1)} - \phi(a)|$ with respect to $|\phi_N^{(k)} - \phi(a)|$ is chosen as or new energy guess. Two more comments are in order. First, to verify the condition $\phi_N^{computed} = 0$ and hence to find a criterion for stopping the energy variation procedure you should define a tolerance parameter ϵ. We stop the procedure at kth step whenever the condition $0 < |\phi_N^{(k)} - \phi(a)| < \epsilon$ is fulfilled. The second point governs the divergence of wave function before the wavefunction evaluation reaches the last grid $i = N$. If the energy E is not sufficiently close to an eigenvalue then the wavefunction ϕ_i may become too large at intermediate grids $i < N$. To avoid such divergence, we define a cutoff quantity *maxsize*. If we encounter a grid i for which $|\phi_i| > maxsize$ then we terminate the wavefunction evaluation and store ϕ_{i-1} as our ϕ_N in the energy variation trend. The programme `Shooting` (see appendix 8.D for details) numerically solves the time-dependent Schrödinger equation for given boundary values in a finite region of space. In particular, the programme solves the infinite square well. The first nine energy eigenvalues in the reduced units obtained by the Shooting method are $E_1 = 1.233, E_2 = 4.932, E_3 = 11.097, E_4 = 19.728, E_5 = 30.825, E_6 = 44.388, E_7 = 60.413, E_8 = 78.912$, and $E_9 = 99.873$. The theoretical values are: $E_1 = 1.233, E_2 = 4.934, E_3 = 11.097, E_4 = 19.726, E_5 = 30.825, E_6 = 44.388, E_7 = 60.417, E_8 = 78.912$, and $E_9 = 99.873$. You see the excellent agreement between theoretical and computational results. Figure (8.14) shows the computed wave functions for some values of E. We have taken our initial guess as $E^{(0)} = 1.4$. The calculated ground state is $E_G = 1.236$. The exact value is $E_G = 1.233$. The difference is related to the tolerance parameter ϵ which is set to 0.01. If we consider a smaller ϵ an improved answer should be obtained. You will be asked in exercises to investigate the role of ϵ on the problem.

Method two: Matrix method

In the second method, the time-independent Schrödinger equation is turned into a matrix eigenvalue problem. To this end, we utilise the potential symmetry and consider first the even parity solutions

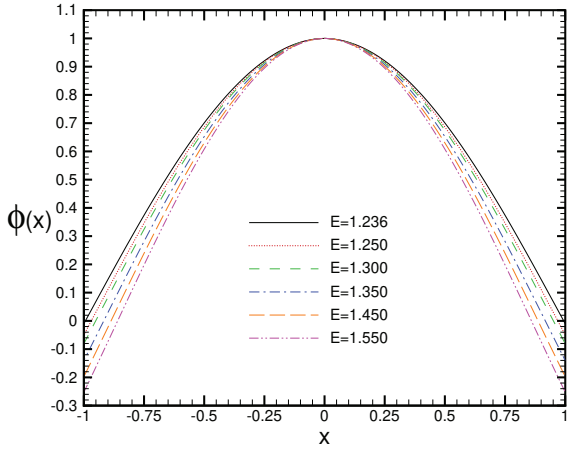

Figure 8.14: *Computed ground state wave function for a symmetric infinite square well potential calculated by shooting method for several values of energyE. The spatial grid size was* $\Delta x = 0.02$. *We sat* $\phi(0) = 1$ *and* $\phi'(0) = 0$.

which have the property $\phi(-x) = \phi(x)$. This property implies $\phi'(0) = 0$. We discretise our half space $[0, a]$ with $N + 1$ grid points such that $x_0 = 0$ and $x_N = a$. In this region the potential is zero and hence the Schrödinger equation becomes $\frac{-\hbar^2}{2m}\phi''(x) = E\phi(x)$. The boundary condition $\phi(a) = 0$ gives $\phi_N = 0$ so the unknowns are $\phi_0, \phi_1, \phi_2, \cdots, \phi_{N-1}$ and of course the energy E respectively. Note there are $N + 1$ unknowns. For the middle points $i = 1, 2, \cdots, N - 2$ the finite-difference version of the Schrödinger equation in the reduced units will be:

$$\frac{2\phi_i - \phi_{i-1} - \phi_{i+1}}{2(\Delta x)^2} = E\phi_i \tag{8.72}$$

The points x_0 and x_{N-1} need special attention. At $x = x_0$ we have: $\frac{2\phi_0 - \phi_{-1} - \phi_1}{2(\Delta x)^2} = E\phi_0$. To proceed further, we exploit the evenness of the wave function and replace ϕ_{-1} by ϕ_1. This gives:

$$\frac{\phi_0 - \phi_1}{(\Delta x)^2} = E\phi_0 \tag{8.73}$$

Analogously at x_{N-1} we have: $\frac{2\phi_{N-1}-\phi_{N-2}-\phi_N}{2(\Delta x)^2} = E\phi_{N-1}$. Putting $\phi_N = 0$ we obtain:

$$\frac{2\phi_{N-1} - \phi_{N-2}}{2(\Delta x)^2} = E\phi_{N-1} \tag{8.74}$$

The Schrödinger equation at grid points gives N equations. Now everything is ready for writing the finite-difference Schrödinger equation in the form of a matrix eigenvalue problem: $A\Phi = E\Phi$ in which $\Phi^\dagger = (\phi_0, \phi_1, \cdots, \phi_{N-1})$. Unfortunately, the matrix A is not symmetric! It differs from a diagonal matrix by the a_{01} entry. In fact, (8.73) gives $a_{01} = -\frac{1}{(\Delta x)^2}$. If it was $-\frac{1}{2(\Delta x)^2}$ the matrix of coefficients would be symmetric and we could analytically find the eigenvalues/eigenvectors (see (Thornton and Marion, 2003) or (Goldstein, 2001)). Incidentally, we have to resort to numerical computations. Although we may use a special subroutine for obtaining the spectrum of non-symmetric matrices (such as the routine `hqr` in Numerical Recipe (W. H. Press and Flannery, 2002)) we prefer to work with symmetric matrices because it is much more reliable to computationally obtain their spectra. To this end, we separate ϕ_0 from our unknowns, and after finding the rest of unknowns i.e.; $\phi_1, \phi_2, \cdots, \phi_{N-1}$ and E we find ϕ_0 from the finite difference form of $\phi'(0) = 0$. The Schrödinger equation at $x = x_1$ gives: $\frac{2\phi_1-\phi_0-\phi_2}{2(\Delta x)^2} = E\phi_1$. The easiest discretised form of $\phi'(0) = 0$ is a forward difference: $\frac{\phi_1-\phi_0}{\Delta x} = 0$ which implies $\phi_1 = \phi_0$. Therefore the Schrödinger gives:

$$\frac{\phi_1 - \phi_2}{2(\Delta x)^2} = E\phi_1 \tag{8.75}$$

The other equations remain unchanged. You see that we have managed to make matrix A symmetric. The price we paid is that a_{11} is no longer equal to $\frac{1}{(\Delta x)^2}$. In fact it is $\frac{1}{2(\Delta x)^2}$ now. The merit of working numerically with symmetric matrices is priceless. We implement the routine `tqli` (see chapter 11 of Numerical Recipe) to find the eigenvalues/eigenvectors. The first five eigenvalues for $N = 200$ grid points are as follows:$E_1 = 1.225, E_2 = 11.05, E_3 = 30.70, E_4 = 60.14, E_5 = 99.40$. Interestingly you see the even eigenvalues (corresponding to even eigenvectors) are nicely computed (thanks to `tqli` algorithm). Note the eigenvalues depend drastically on the number of grid points N. The same computed eigenvalues for $N = 1000$ are as follows:

$E_1 = 1.15, E_2 = 10.97, E_3 = 30.62, E_4 = 60.26, E_5 = 99.58$. You see that by increasing N some eigenvalues have improved but others have not. Now we consider the odd parity solutions. In this case, we have $\phi(-x) = -\phi(x)$ which implies $\phi(0) = \phi(x_0) = \phi_0 = 0$. Here we have one less unknown. All the equations of the even parity solutions remain unchanged except this one: $H\phi(x_1) = E\phi(x_1)$. Centred space discretisation gives: $\frac{2\phi_1-\phi_0-\phi_2}{2(\Delta x)^2} = E\phi_1$. Setting $\phi_0 = 0$ gives $\frac{2\phi_1-\phi_2}{2(\Delta x)^2} = E\phi_1$. Hopefully, the matrix A is not only symmetric but also all the diagonal elements are equal to two. As you know from undergraduate physics the eigenvalues of this matrix are given by $\gamma_r = 4\sin^2\frac{r\pi}{2N}$ (see (Thornton and Marion, 2003) chapter 12). Note that in our finite-difference form, the diagonal elements of matrix A is $\frac{2}{(\Delta x)^2}$ whereas its off-diagonal ones are $-\frac{1}{(\Delta x)^2}$. The matrix eigenvalue equations turn out to be $A\Phi = 2(\Delta x)^2 E\Phi$. This gives $E_r = \frac{2}{(\Delta x)^2}\sin^2(\frac{r\pi}{N})$. Replacing the spatial grid $\Delta x = \frac{a}{N}$ we have:

$$E_r = 2(\frac{N}{a})^2 \sin^2(\frac{r\pi}{N}) \tag{8.76}$$

The first four analytical eigenvalues for $N = 200$ are $E_1 = 4.93, E_2 = 19.74, E_3 = 44.40$, and $E_4 = 78.93$. The computed ones by the `tqli` subroutine are $E_1 = 4.88, E_2 = 19.54, E_3 = 43.96, E_4 = 78.15$ respectively. As you see the analytical eigenvalues given by the exact diagonalisation of matrix A coincide, with two decimal point approximation, with the exact solution $E_n = \frac{n^2\pi^2\hbar^2}{8ma^2}$. I would like to mention that contrary to the public belief which overlooks the matrix eigenvalue method, our results show, at least in the context of a simple quantum mechanical problem, that this method is worth further attention.

Method three: Root finding

We now discuss the third numerical method for finding the spectrum of the time-independent Schrödinger equation. The desired E is the one for which the wave function ϕ becomes zero at the right wall boundary. In the finite-difference form, we have $\phi_N = 0$. Actually, ϕ_N is a function of energy E. So what we should do is simply find the roots of the algebraic equation $\phi_N(E) = 0$. There are numerous root-finding schemes in the literature. We refer the readers to chapter nine of *Numerical Recipe* for more details.

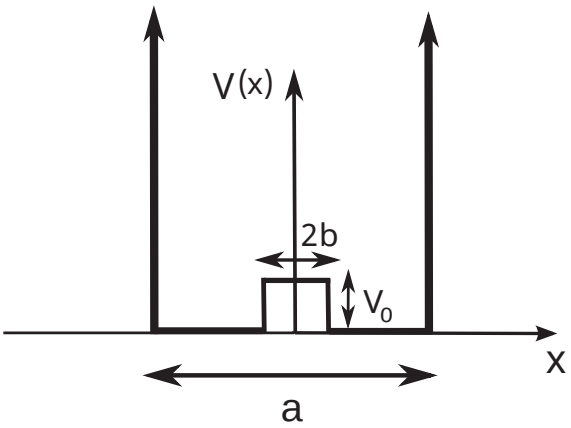

Figure 8.15: *Schematic representation of a symmetric perturbed infinite well of width a.*

8.5.3 Perturbation of the infinite square well

Let us consider the effect of a small perturbation on the eigenstates and eigenvalues of the symmetric infinite square well. Suppose we place a small rectangular bump of half-width $b < a$ and height V_b symmetrically about $x = 0$. See figure (8.15) for a schematic representation. We wish to see how the ground state energy and eigenstate change with V_b and b. As in the infinite square well take $a = 1$. We first proceed analytically. Because the potential is symmetric the eigenfunctions are even or odd. We first consider the even parity solutions. It can be proved that there exists no negative eigenvalue. Suppose the contrary is true i.e.; there is a negative eigenvalue $E = -|E|$. In the region $b < x < a$ we have $\phi(x) = Ae^{\kappa x} + Be^{-\kappa x}$ with $\kappa = \frac{\sqrt{2m|E|}}{\hbar}$. Within the bump region $0 < x < b$ we have: $\phi_{bump}(x) = Ce^{\xi x} + De^{-\xi x}$ with $\xi = \frac{\sqrt{2m(V_b+|E|)}}{\hbar}$. Boundary condition $\phi(a) = 0$ implies:

$$Ae^{\kappa a} + Be^{-\kappa a} = 0 \tag{8.77}$$

Continuity of wavefunction at $x = b$ implies:

$$Ae^{\kappa b} + Be^{-\kappa b} = Ce^{\xi b} + De^{-\xi b} \tag{8.78}$$

Continuity of wavefunction derivative at $x = b$ implies:

$$\kappa(Ae^{\kappa b} - Be^{-\kappa b}) = \xi(Ce^{\xi b} - De^{-\xi b}) \tag{8.79}$$

The even parity implies $\phi'(0) = 0$ which gives $\xi(C - D) = 0$. Accordingly, we have $C = D$. Putting this into (8.78) and (8.79) we arrive at:

$$Ae^{\kappa b} + Be^{-\kappa b} - C(e^{\xi b} + e^{-\xi b}) = 0 \tag{8.80}$$

$$\kappa(Ae^{\kappa b} - Be^{-\kappa b}) - C\xi(e^{\xi b} - e^{-\xi b}) = 0 \tag{8.81}$$

Equations (8.77), (8.80) and (8.81) comprise a linear set of equations for unknown coefficients A, B and C. To have a nontrivial solution, the determinant of the coefficient matrix should be set to zero. The determinant, after some straightforward algebra, turns out to be $-2\kappa(e^{\xi b} + e^{-\xi b})$. As you see it is always negative and cannot be zero. This means that our assumption of having a negative eigenvalue has been incorrect. As in the infinite square well problem, we consider three distinct cases: $0 < E < V_b$, $E > V_b$, and $E = V_b$. When $E > V_b$ we have sinusoidal forms in the bump and outside regions:

$$\phi(x) = Ae^{ikx} + Be^{-ikx} \qquad b \leq x \leq a \tag{8.82}$$

$$\phi(x) = Ce^{iKx} + De^{-iKx} \qquad 0 \leq x \leq b \tag{8.83}$$

with $k = \frac{\sqrt{2mE}}{\hbar}$ and $K = \frac{\sqrt{2m(E-V_b)}}{\hbar}$. Note that K is smaller than k. Because the potential is symmetric its eigenstates have definite parity. Consider the even eigenstates first. For the even eigenstates their derivative $\phi'(x)$ is an odd function that is $\phi'(-x) = -\phi'(x)$. Therefore at $x = 0$ we should have $\phi'(0) = 0$. Substituting $\phi(x)$ from (8.83) gives:

$$\phi'(0) = iK(Ce^{iKx} - De^{-iKx})|_{x=0} = iK(C - D) = 0 \tag{8.84}$$

Therefore we have $C = D$. Continuity of the wavefunction at $x = b$ gives:

$$C(e^{iKb} + e^{-iKb}) = Ae^{ikb} + Be^{-ikb} \tag{8.85}$$

Continuity of the wavefunction derivative at $x = b$ gives:

$$CK(e^{iKb} - e^{-iKb}) = k(Ae^{ikb} - Be^{-ikb}) \tag{8.86}$$

Moreover, the wavefunction should be zero at $x = a$ which gives:

$$Ae^{ika} + Be^{-ika} = 0 \tag{8.87}$$

Equations (8.85)-(8.87) form a homogeneous system of linear equations for the unknowns A, B, and C. The necessary condition for having a nontrivial solution is that the determinant of the coefficient matrix should be zero. Some algebra gives us the determinant as follows:

$$K \sin(Kb) \sin k(a-b) - k \cos(Kb) \cos k(a-b) \tag{8.88}$$

Setting (8.88) to zero gives us the required equation for the unknown even eigenvalues. Note k and K are functions of E. You can verify that in the limit of $b \to 0$ equation (8.88) reduces to $k \cos ka = 0$ which is exactly the equation we encountered for the even parity solutions of the symmetric infinite square well. We remark that (8.88) is a transcendental equation. Before considering the odd eigenstates let us argue that (8.88) gives a set of discrete eigenvalues. To see this it would be more suitable to recast (8.88) as follows:

$$\frac{k}{K} = \sqrt{\frac{E}{E - V_b}} = \tan(Kb) \tan k(a-b) \tag{8.89}$$

The intersection of the left-hand side with the periodic function on the right-hand side gives an infinite discrete number of eigenvalues. In the exercises, you are asked to find the corresponding eigenvalues numerically. Let us now consider odd eigenstates. Here we have $\phi(0) = 0$. This time (8.83) gives $C = -D$. Similar calculations give the determinant as follows:

$$K \cos(Kb) \sin k(a-b) + k \sin(Kb) \cos k(a-b) \tag{8.90}$$

In the limit, $b \to 0$ equation (8.90) reduces to $K \sin ka = 0$ which is the same equation we obtained for the odd parity solutions of the symmetric infinite square well. We can recast (8.90) into a more appropriate form as follows:

$$\frac{k}{K} = \sqrt{\frac{E}{E - V_b}} = -\tan k(a-b) \cot Kb \tag{8.91}$$

You can qualitatively see that there is an infinite number of odd eigenvalues. I leave it as an exercise for you to investigate the solution of (8.91). So far we have discussed the case $E > V_b$. Similarly, you can find the eigenvalue equation for the other two cases $0 < E < V_b$ and $E = V_b$. when $E < V_b$ equation (8.83) becomes:

$$\phi(x) = Ce^{\kappa x} + De^{-\kappa x} \tag{8.92}$$

where $\kappa = \frac{\sqrt{2m(V_b - E)}}{\hbar}$. This means that all the results obtained for the case $E > V_b$ are the same provided K is substituted by $-i\kappa$. Using $\sin ix = i \sinh x$ and $\cos ix = \cosh x$ equations (8.88) and (8.90) turn into the following equations:

$$\kappa \sinh(\kappa b) \sin k(a-b) - k \cosh(\kappa b) \cos k(a-b) \tag{8.93}$$

$$-\kappa \cosh(\kappa b) \sin k(a-b) + k \sinh(\kappa b) \cos k(a-b) \tag{8.94}$$

We leave it as an exercise to find the eigenvalues form (8.93) and (8.94). Also the third case i.e.; $E = V_b$ is left for the exercises. Now we try to solve the problem numerically. The method we choose is the shooting method. The only difference is the potential form. We take the total number of grid points N and assume M grids of them specify the bump region. In particular, we have $b = M\Delta x$. More technically, we take $M = int(\frac{b}{a})N$. Figure (8.16) shows the dependence of the ground state energy on bump region half width b. As you see the ground state energy increases with increasing b. Next, we investigate the dependence of E_0 on the bump region height V_b for some values of b. Figure (8.17) depicts such behaviour. We have also computed the ground state energy for very large values of V_b. The results for $V_b = 10$ and $V_b = 20$ are $E_0 = 2.76$ and $E_0 = 3.61$ respectively. The computed ground state and the lowest excited state for $V_b = 10$ and $b = 0.1a$ are shown in figure (8.18).

8.6 finite square well

The next potential we wish to consider is the symmetric finite square well. See figure (8.19) for a schematic representation. The finite square well potential is given by: $V(x) = 0 \quad |x| \leq a$ and $V(x) = V_0 \quad |x| > a$.

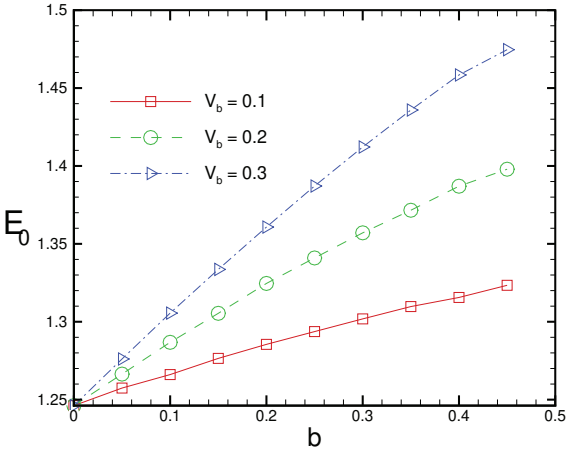

Figure 8.16: *Dependence of E_0 on b for various values of V_b. The shooting method has been used with $\Delta x = 0.01$.. The well half width a is set to one.*

The input parameters are the well's depth, V_0, and a the well's half-width. Because the potential is symmetric the eigenfunctions have definite parity. Consider the even parity first. We prove there is no negative eigenvalue. Suppose the contrary and assume there is a negative eigenvalue $E = -|E|$. In the region $0 \leq x \leq a$ we have $\phi_I(x) = Ae^{\kappa x} + Be^{-\kappa x}$ with $\kappa = \frac{\sqrt{2m|E|}}{\hbar}$. In the region $x > a$ we have: $\phi_{II}(x) = Ce^{\xi x} + De^{-\xi x}$ with $\xi = \frac{\sqrt{2m(V_0+|E|)}}{\hbar}$. The wavefuction's finiteness at infinity implies $C = 0$. Continuity of the wavefunction at $x = a$ implies:

$$Ae^{\kappa a} + Be^{-\kappa a} = De^{-\xi a} \tag{8.95}$$

Continuity of the wavefunction derivative at $x = a$ implies:

$$\kappa(Ae^{\kappa a} - Be^{-\kappa a}) = -\xi De^{-\xi a} \tag{8.96}$$

The even parity implies $\phi'(0) = 0$ which gives $\kappa(A - B) = 0$. Accordingly we have $A = B$ and therefore equations (8.95) and (8.96) simplify as follows:

$$A(e^{\kappa a} + e^{-\kappa a}) = De^{-\xi a} \tag{8.97}$$

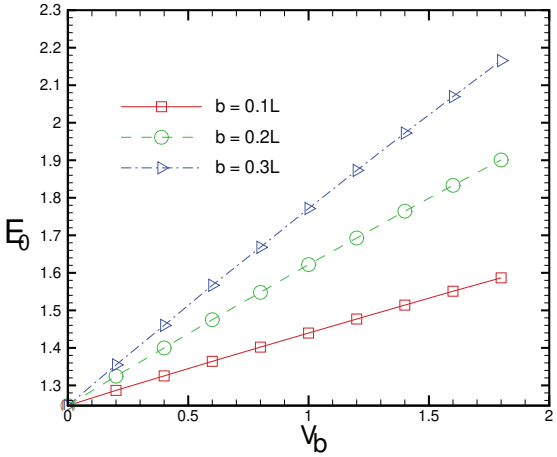

Figure 8.17: *Dependence of E_0 on V_b for various values of b.*

$$\kappa A(e^{\kappa a} - e^{-\kappa a}) = -\xi D e^{-\xi a} \tag{8.98}$$

Dividing (8.98) by (8.97) gives: $\tanh(\kappa a) = -\frac{\xi}{\kappa}$. This is an inconsistent equation because $|\tanh(\kappa a)|$ is less than one whereas $\frac{\xi}{\kappa}$ is greater than one. This inconsistency implies no negative eigenvalue is allowed. In exercises, you are asked to show the inconsistency condition holds for odd states if there are negative eigenvalues. As in the preceding problems we consider three distinct cases of $0 \leq E < V_0$, $E = V_0$ and $E > V_0$. Let us first consider the case $E > V_0$. In the region $0 \leq x \leq a$ we have:

$$\phi_I(x) = Ae^{ikx} + Be^{-ikx} \tag{8.99}$$

In the region $x > a$ we have:

$$\phi_{II}(x) = Ce^{iKx} + De^{-iKx} \tag{8.100}$$

With $k = \frac{\sqrt{2mE}}{\hbar}$ and $K = \frac{\sqrt{2m(E-V_0)}}{\hbar}$. Note that K is smaller than k. Continuity of wavefunction at $x = a$ implies:

$$Ae^{ika} + Be^{-ika} = Ce^{iKa} + De^{-iKa} \tag{8.101}$$

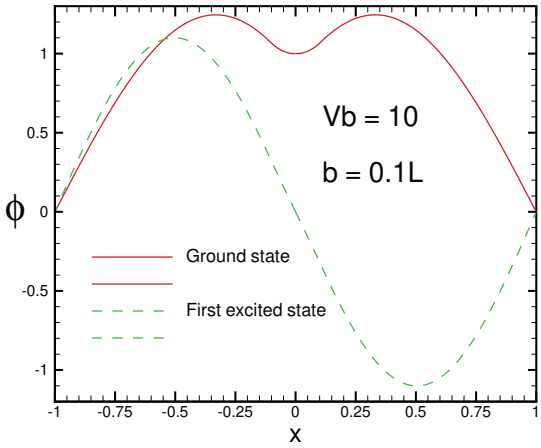

Figure 8.18: *The ground and the first excited state for a perturbed symmetric infinite potential well. The bump region specifications are* $V_b = 10$ *and* $b = 0.1a$*. The grid spacing was taken as* $\Delta x = 0.01$*.*

Continuity of wavefunction derivative at $x = a$ implies:

$$k(Ae^{ika} - Be^{-ika}) = K(Ce^{iKa} - De^{-iKa}) \tag{8.102}$$

We consider the even wavefunction first. For these functions we have $\phi'(0) = 0$ which gives $ik(A - B) = 0$. Accordingly, we have $A = B$. Setting $A = B$ in (8.101) and (8.102) gives two equations for four unknowns A, C, D and energy E. We can add another equation by requiring the normalisation condition but still, we have one more unknown. This means that we can freely choose E. In other words, there is a continuum of energy states for $E > V_0$. Once you specify E the constants A, C, and D will be determined. For odd eigenfunctions we have $\phi(0) = 0$ which gives $A = -B$. Putting this into (8.101) and (8.102) gives two equations for four unknowns A, C, D and energy E. Again we conclude there is a continuum of energy E. Next, we consider the second case i.e.; $E < V_0$. In this situation $\phi_I(x)$ remains unchanged but $\phi_{II}(x)$ modifies as:

$$\phi_{II}(x) = De^{-\kappa x} \tag{8.103}$$

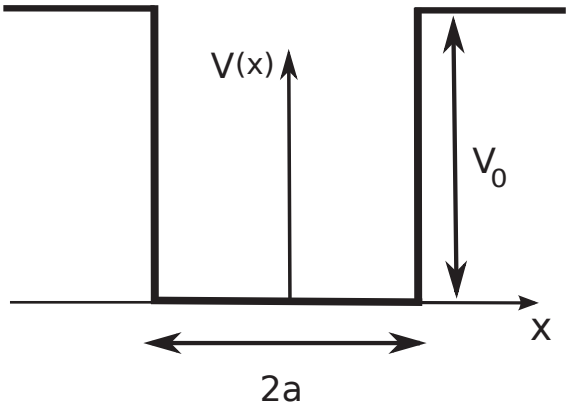

Figure 8.19: *A finite symmetric quantum square well with width* $2a$ *and depth* V_0*. In figure* $a = 2$ *and* $V_0 = 1$*.*

Where $\kappa = \frac{\sqrt{2m(V_0-E)}}{\hbar}$. Note that we have set the coefficient of the term $e^{\kappa x}$ equal to zero to avoid divergence at large x. Continuity of wave function and its derivative at $x = a$ give:

$$Ae^{ika} + Be^{-ika} = De^{-\kappa a} \tag{8.104}$$

$$ik(Ae^{ika} - Be^{-ika}) = -\kappa De^{-\kappa a} \tag{8.105}$$

Consider even eigenfunctions first. The even parity implies $\phi'(0) = 0$ which gives $ik(A - B) = 0$. Accordingly, we have $A = B$. Dividing (8.105) by (8.104) gives:

$$\frac{\kappa}{k} = \tan ka \tag{8.106}$$

This is a nonlinear equation for E. We can solve it graphically. Two curves $\frac{\kappa}{k} = \sqrt{\frac{V_0-E}{E}}$ and $\tan \frac{a\sqrt{2mE}}{\hbar}$ should be intersected. The number of times these two curves intersect each other (number of eigenvalues) crucially depends on the well's depth V_0. For $V_0 < \frac{\pi^2\hbar^2}{2ma^2}$ we have one solution, for $\frac{\pi^2\hbar^2}{2ma^2} < V_0 < \frac{2\pi^2\hbar^2}{ma^2}$ we have two solutions etc. Figure (8.20) shows the intersection of two curves $f(E) = \frac{\kappa(E)}{k(E)}$ and $f(E) = \tan k(E)a$ for $V_0 = 1$ and $V_0 = 13$. As you see for $V_0 = 1$ there is one

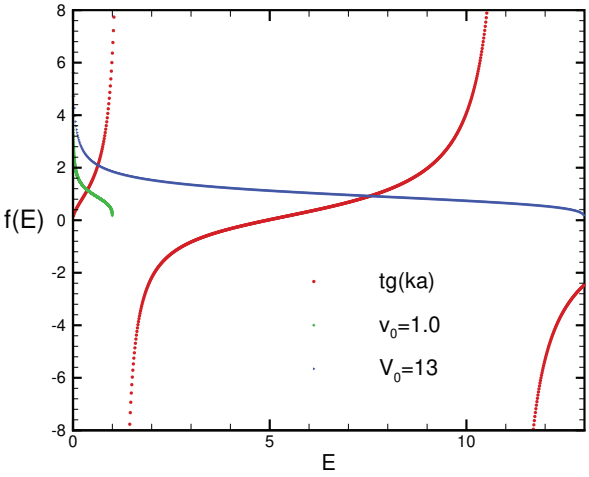

Figure 8.20: *Intersection of two curves* $f(E) = \frac{\kappa(E)}{k(E)}$ *and* $f(E) = \tan k(E)a$. *For the well's depth* $V_0 = 1$, *we have one intersection whereas for* $V_0 = 13$ *we have two intersections.*

solution which means there is a single bound state. From the figure, we find its eigenvalue $E_G = 0.365$. If we increase the well's depth to $V_0 = 13$ two solutions appear. The energy eigenvalues turn out to be $E_1 = 0.637$ and $E_2 = 7.553$. You see that by increasing V_0 the ground state energy increase. We leave it as an exercise for you to check in the limit $V_0 \to \infty$ the ground state E_G approaches to that of an infinite well that is 1.233. Before considering odd states, let us consider the special case $E = V_0$. In this case, the Schrödinger equation in region $x > a$ becomes $-\frac{\hbar^2 \phi_{II}''}{2m} = 0$. The solution is: $\phi_{II}(x) = Cx + D$. The finiteness of the wavefunction at infinity implies $C = 0$ therefore the wavefunction is constant for $x > a$. As usual, the even parity implies $A = B$. Continuity of wave function and its derivative at $x = a$ give: $Ae(^{ika}+e^{-ika}) = D$ and $ikA(e^{ika} - e^{-ika}) = 0$. The latter equation implies $\sin ka = 0$ or $A = 0$ or $k = 0$. But none of these equations are consistent. A cannot be zero because the wavefunction would be zero in region $|x| \leq a$. k cannot be zero because now we

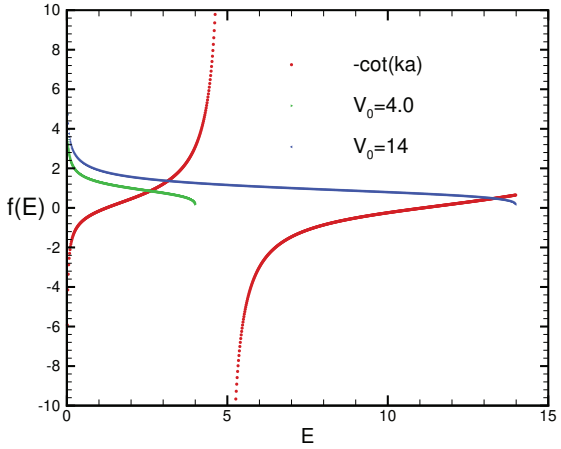

Figure 8.21: *Intersection of two curves* $f(E) = \frac{\kappa(E)}{k(E)}$ *and* $f(E) = -\cot k(E)a$. *For the well's depth* $V_0 = 4$, *we have one intersection whereas for* $V_0 = 14$ *we have two intersections.*

have $k = \frac{\sqrt{2mV_0}}{\hbar} \neq 0$. Eventually, $\sin ka$ cannot be zero because ka cannot be an integer multiple of π for otherwise we have $\frac{a\sqrt{2mV_0}}{\hbar} = n\pi$ and this violates the independency of a and V_0. In conclusion, there is no eigenvalue $E = V_0$. Now let us consider the odd states. In the case $E < V_0$. Hopefully, many of the steps done for the even states remain unchanged especially, (8.103)-(8.105). For odd states $\phi(0) = 0$ implies $A = -B$. Substitution of B by $-A$ and then dividing (8.105) by (8.104) gives:

$$\frac{\kappa}{k} = -\cot ka \tag{8.107}$$

Figure (8.21) shows the intersection of two curves $f(E) = \frac{\kappa(E)}{k(E)}$ and $f(E) = -\cot k(E)a$ for $V_0 = 4$ and $V_0 = 14$. For $V_0 = 4$ the single energy solution becomes $E_1 = 2.596$. For $V_0 = 14$ there are two solutions $E_1 = 3.15$ and $E_2 = 13.3$.

8.6.1 Numerical solution

Now we try to solve the problem numerically. We sketch the wavefunction up to x_{max} (an approximation to infinity) which is taken to be $x_{max} = 5a$. The interval $[0, x_{max}]$ is discretise by N grids and we take M as the grid number at $x = a$ ($M < N$). Let us consider even states first. Without loss of generality, we can assign an arbitrary value say one to ϕ_0. Later after we obtain the solution we will multiply all ϕ_i by a normalising factor. To find ϕ_1 we set $i = 0$ in (8.62) which gives:

$$\phi_1 = 2\phi_0[1 - \frac{m(\Delta x)^2}{\hbar^2}(E - V_0)] - \phi_{-1} \tag{8.108}$$

For even parity solutions we have $\phi_{-1} = \phi(-\Delta x) = \phi(\Delta x) = \phi_1$. Replacing ϕ_{-1} by ϕ_1 in (8.108) gives ϕ_1 as follows:

$$\phi_1 = \phi_0[1 - \frac{m(\Delta x)^2}{\hbar^2}(E - V_0)] \tag{8.109}$$

We can now iterate (8.62) once E is specified. For $E > V_0$ we have analytically shown that every value of E can be an eigenvalue. Figure (8.22) shows two even wavefunctions corresponding to $E = 12$ and $E = 25$. The well's depth is $V_0 = 10$. As you see we have a sinusoidal form for both regions. Of course, the wave vectors differ in regions $|x| \leq a$ and $|x| > a$. To evaluate odd eigenfunctions we process differently. First, we note that $\phi(0) = 0$ for odd functions which gives $\phi_0 = 0$. To find ϕ_1 we arbitrary set $\phi'(0) = 1$ which gives: $\phi_1 = \Delta x + \phi_0 = \Delta x$. Note that replacing ϕ_{-1} by $-\phi_1$ in (8.108) gives you a trivial result $0 = 0$. Having specified ϕ_1 we can iterate (8.62) to find the eigenstates once E is given. Figure (8.23) shows two odd eigenstates corresponding to $E = 12$ and $E = 25$. It is worth mentioning that any eigenvalue $E > V_0$ is doubly degenerate (one even eigenstate and one odd eigenstate). To numerically find the eigenvalue for the case $E < V_0$ we proceed as follows: we set E to a small number slightly above zero and numerically evaluate ϕ from (8.62) and check whether it decreases at large distances. If the wavefunction's magnitude increases then we increase E by the amount ΔE and the procedure is repeated until ϕ converges at a large distance say x_{max}. We record E as the bound state eigenvalue. Remind you that if we are looking for an even (odd) state we set $\phi_0 = 0$ ($\phi_1 = \Delta x$). Figure (8.24) shows the ground state wave function obtained by this method. The eigenvalue turns out to

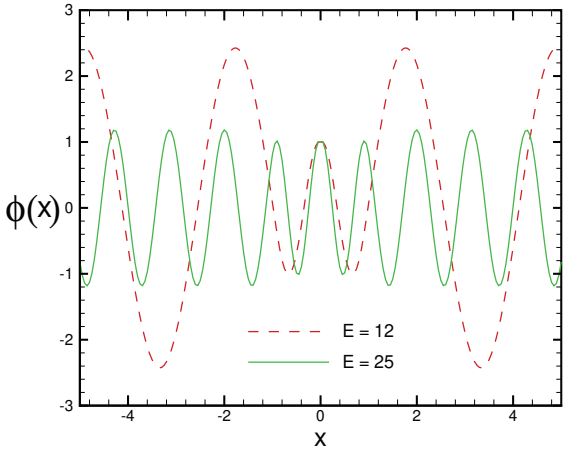

Figure 8.22: *Two even eigenstates (unnormalised) corresponding to* $E = 12$ *and* $E = 25$ *which are larger than* $V_0 = 10$ *in a finite symmetric quantum well of half-width* $a = 1$.

be $E_0 = 0.528$ which is quite in good agreement with the analytical solution found graphically in figure (8.20).

8.7 Harmonic oscillator potential

Our next example is devoted to the harmonic oscillator. Consider the symmetric potential $V(x) = \frac{1}{2}kx^2$ for a harmonic oscillator. In this problem, the space is unlimited and the boundary condition on the wavefunction is $lim_{|x|\to\infty}\phi(x) = 0$. We solve the problem by turning the Schrödinger equation into a matrix eigenvalue problem. Noting the potential symmetry $V(-x) = V(x)$ we consider first the even parity solutions: $\phi(-x) = \phi(x)$. From the physics of the problem, we know $\phi(x) \to 0$ when $|x| \to \infty$. We approximate the mathematical infinity by a large but finite number x_{max}. Let us take this large number $x_{max} = 5$ in the atomic unit in which our length scale $\sqrt{\frac{\hbar}{m\omega_0}}$ is set to one. Moreover, we have taken the spring constant $k = 1$ which

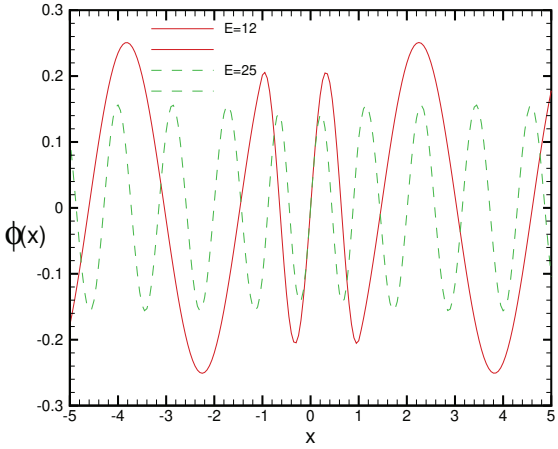

Figure 8.23: *Two odd eigenstates (unnormalised) for two values of E=12,25 which are larger than $V_0 = 10$ in a finite symmetric quantum well of half-width $a = 1$.*

gives $\omega_0 = \sqrt{\frac{k}{m}} = 1$. We discretise the interval $[0, x_{max}]$ with $N+1$ grids $x_0, x_1, \cdots, x_N$ such that $x_0 = 0$ and $x_N = x_{max}$. As usual $\phi(x_i)$ and $V(x_i)$ are denoted by ϕ_i and V_i respectively. Furthermore, we set $\phi_N = 0$ as an approximation to the condition $\phi(x) \to 0$ at $x \to \infty$. We separate ϕ_0 from our unknowns and after finding the rest of them i.e.; $\phi_1, \phi_2, \cdots, \phi_{N-1}$ and E we find ϕ_0 from the finite difference form of $\phi'(0) = 0$. You will shortly see the reason why we separate ϕ_0 from the rest of the unknowns. The Schrödinger equation at $x = x_1$ becomes: $\frac{2\phi_1 - \phi_0 - \phi_2}{2(\Delta x)^2} + V_1\phi_1 = E\phi_1$. The easiest discretised form of $\phi'(0) = 0$ is forward difference: $\frac{\phi_1 - \phi_0}{\Delta x} = 0$ which implies $\phi_1 = \phi_0$. Therefore the Schrödinger equation at $x = x_1$ becomes:

$$\frac{\phi_1 - \phi_2}{2(\Delta x)^2} + V_1\phi_1 = E\phi_1 \tag{8.110}$$

The other equations for grids $i = 2, \cdots, N-2$ become:

$$\frac{2\phi_i - \phi_{i-1} - \phi_{i+1}}{2(\Delta x)^2} + V_i\phi_i = E\phi_i \tag{8.111}$$

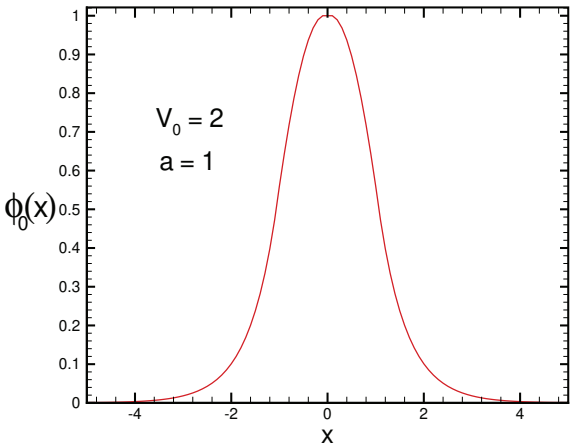

Figure 8.24: *Ground state wave function $\phi(x)$ for a finite symmetric quantum square well. The well's depth is $V_0 = 2$. The eigenvalue is $E_0 = 0.528$. The number of grids is $N = 100$ and energy increment ΔE is taken* 0.0000001.

The Schrödinger equation at the grid $i = N - 1$ turn out to be:

$$\frac{2\phi_{N-1} - \phi_{N-2}}{2(\Delta x)^2} + V_{N-1}\phi_{N-1} = E\phi_{N-1} \tag{8.112}$$

We can write the above linear set of $N - 1$ equations in the matrix form : $A\Phi = 2(\Delta x)^2 E\Phi$ where $\Phi^\dagger = (\phi_1, \cdots, \phi_{N-1})$. The diagonal elements are read as:

$$a_{11} = 1 + 2(\Delta x)^2 V_1;\ a_{ii} = 2[1 + (\Delta x)^2 V_i]\ \ i = 2, \cdots, N-1 \tag{8.113}$$

The off-diagonal elements are -1. For the particular case of harmonic oscillator we have $V(x) = \frac{1}{2}kx^2$ which gives $V_i = \frac{1}{2}kx_i^2 = \frac{1}{2}ki^2(\Delta x)^2$. You see that we have managed to make matrix A symmetric and can benefit from the matrix-solver canned routines. I use my favourite one which is `tqli` (see chapter 11 of (W. H. Press and Flannery, 2002)) to find the eigenvalues. Before showing our numerical results we draw your attention to the fact why we separated ϕ_0 from other unknowns.

If we had treated ϕ_0 on an equal footing with other unknowns then the matrix A would not be symmetric. You may argue why not assign ϕ_0 an arbitrary value as we did for the even eigenstates in the shooting method. Well, we could do that but a constant term $\frac{-\phi_0}{2(\Delta x)^2}$ would appear in the Schrödinger equation at $x = x_1$ which would not allow us to write the equations as $A\Phi = 2(\Delta x)^2\Phi$. Employing the eigenvalue solver routine `tqli` in the programme `HarmonicOscillator` (for details see Appendix 8.E) gives you the even eigenvalues. The first five eigenvalues (for $N = 200$) are as follows: $E_1 = 0.507, E_2 = 2.517, E_3 = 4.523, E_4 = 6.528, E_5 = 8.542$. They become improved if we increase N to 700: $E_1 = 0.500, E_2 = 2.503, E_3 = 4.506, E_4 = 6.511, E_5 = 8.520$. Higher values of N do not lead to better results. In fact, increasing the dimension of matrix A increases the numerical errors as well. Let us see what happens if we increase x_{max} to 10. The results for $N = 400$ and $N = 1400$ grid points are as follows (Δx is the same as for the case $x_{max} = 5$): $E_1 = 0.507, E_2 = 2.517, E_3 = 4.523, E_4 = 6.526, E_5 = 8.530$ and $E_1 = 0.503, E_2 = 2.506, E_3 = 4.507, E_4 = 6.512, E_5 = 8.509$ respectively. You see the larger eigenvalues are improved when x_{max} is increased. We know from quantum mechanics that the exact eigenvalues are $E_n = (n + \frac{1}{2})\hbar\omega_0$ ($\omega_0 = \sqrt{\frac{k}{m}}$) which in our reduced units become $= n + \frac{1}{2}$. The first five exact eigenvalues corresponding to even eigenfunctions are :$E_0 = 0.5, E_2 = 2.5, E_4 = 4.5, E_6 = 6.5$, and $E_8 = 8.5$. You see that the computed eigenvalues (corresponding to even eigenfunctions) are quite close to exact values. Unfortunately, the computed eigenfunctions are not as good as the computed eigenvalues. It would be better to compute the eigenfunctions by replacing the computed eigenvalue E in the Schrödinger equation. Then we assign an arbitrary value to ϕ_0 and set $\phi_1 = \phi_0$. It would then be possible to compute ϕ_i for $i = 2, 3, \cdots, N-1$ by marching equation (8.62). The analytical eigenfunctions of the harmonic oscillator are: $\phi_n(x) = (2^n n!\sqrt{\pi})^{-\frac{1}{2}} H_n(\xi) e^{-\frac{\xi^2}{2}}$ with dimensionless $\xi = \sqrt{\frac{m\omega_0}{\hbar}}x$. In reduced units we have:

$$\phi_n(x) = (2^n n!\sqrt{\pi})^{-\frac{1}{2}} H_n(x) e^{-\frac{x^2}{2}} \tag{8.114}$$

H_n are Hermite polynomials. The first three polynomials are: $H_0(x) = 1, H_1(x) = 2x$ and $H_2(x) = 4x^2 - 2$. Some computed eigenfunctions are shown in figure (8.25). As you can see the agreement is very good

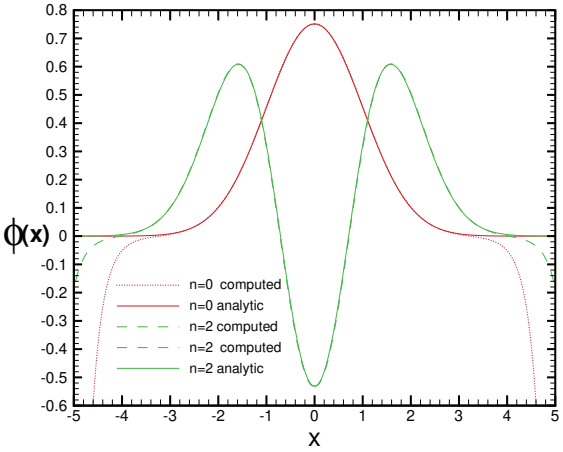

Figure 8.25: *The first two even eigenfunctions of a harmonic oscillator computed by the matrix method for $N = 200$ grid points. Comparison to exact analytical wavefunctions is done.*

except at large distances. Now we consider the odd-parity solutions for eigenfuctions. In this case we have $\phi(0) = \phi(x_0) = 0$ which implies $\phi_0 = 0$. This time, we have one less unknown. All the equations of the even parity solutions remain unchanged except this one: $H\phi(x_1) = E\phi(x_1)$. Centred space discretisation gives: $\frac{2\phi_1 - \phi_0 - \phi_2}{2(\Delta x)^2} + V_1\phi_1 = E\phi_1$ which becomes $\frac{2\phi_1 - \phi_2}{2\Delta x^2} + V_1\phi_1 = E\phi_1$. Hopefully, the matrix A is not only symmetric but also all the diagonal elements are equal to each other. Note that in our finite-difference form, the diagonal elements of matrix A are $2 + 2(\Delta x)^2 V_i$ whereas its off-diagonals are minus one. The matrix eigenvalue equation becomes $A\Phi = 2(\Delta x)^2 E\Phi$. The first three computed eigenvalues by the `tqli` subroutine for $N = 200$ grid points are $E_1 = 1.499, E_2 = 3.499, E_3 = 5.498$. You see the agreement with exact results $1.5, 3.5$ and 5.5 are much better than the even-parity eigenvalues. The reason is that here the condition $\phi_0 = 0$ is exact but in even-parity the condition $\phi_0 = \phi_1$ is only an approximation to $\phi'(0) = 0$. Figure (8.26) shows the computed first odd-parity eigenfunction: As you see in both even and dd parity eigenfunctions, there are deviations from the exact analytical solution near x_{max}. This de-

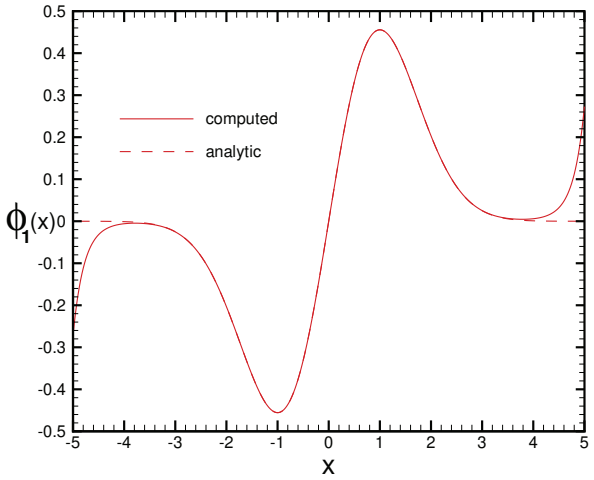

Figure 8.26: *The first odd parity eigenfunctions of a harmonic oscillator computed by the matrix method. Comparison to analytical eigenfunction is done.*

viation is more enhanced for the even-parity eigenfunctions. In the problems, you are asked to test other values of N and x_{max} to see if the numerical solution is improved.

8.8 Variational method

The ground state is the most important eigenstate in any quantum mechanical system. Some special approaches have been developed for obtaining ground state energy. Among them perhaps the *variational method* is the most practical one. You are familiar with this fascinating method in your quantum mechanics course. This approach has numerous applications in chemistry, atomic and molecular physics, nuclear physics, and condensed matter physics. The variational scheme is based on the variation principle in quantum mechanics. According to this principle, the expectation value of the Hamiltonian for an arbitrary trial wave function $\Phi(x)$ is greater than or equal to the ground

state energy E_0. More concisely we have (in one dimension for simplicity):

$$E_T[\Phi] = \langle H \rangle = \frac{\int \Psi^*(x) H \Psi(x) dx}{\int \Psi^*(x) \Psi(x) dx} \geq E_0 \tag{8.115}$$

In fact, the trial energy functional $E_T[\Phi]$ can be regarded as an upper limit for the ground state energy E_0. By minimisation of $E_T[\Phi]$ we approach the unknown true ground state. If you want to proceed analytically then you should include some parameters $a_1, a_2, \cdots$ in the trial wave function Ψ and then analytically find the dependence of trial energy E_T on them. The next step is to find the absolute minimum of $E_T(a_1, a_2, \cdots)$ in the parameter space $a_1, a_2, \cdots$. For simple Hamiltonians, you may follow this procedure but it would be a formidable task if the system becomes more complicated. Another difficulty of this approach is that even if you manage to analytically evaluate the integrals in (8.115) its minimisation remains a sophisticated mathematical problem because in general $E_T(a_1, a_2, \cdots)$ is a nonlinear function of parameters and you should solve a set of nonlinear equations $\frac{\partial E_T}{\partial a_j} = 0$ for $j = 1, 2, \cdots$ which are normally not amenable to exact solutions. We will not follow this scheme and turn our attention to a numerical approach. This time we employ some elements from stochasticity. We search the space of functions stochastically! First, we restrict the space between x_{min} and x_{max}. The values of these two limits depend on the problem. For example, if we are considering a symmetric infinite quantum well of width $2a$ then $x_{min} = -a$ and $x_{max} = +a$. Second, we discretise the space between x_{min} and x_{max} with grid points separated from each other by Δx. Any trial wavefunction Ψ_T is specified by its values on grid points: $\Psi_0, \Psi_1, \cdots, \Psi_N$. Next, we look for the minimising trial wavefunction. For simplicity, we assume the trial wave function is real. As a matter of fact, we only search a subspace of the space of functions. The trial energy (8.115) will be approximated by:

$$E_T(\Psi_1, \cdots, \Psi_N) = \frac{\sum_j \Delta x \Psi_j [\frac{-\hbar^2}{2m} \frac{\Psi_{j+1} + \Psi_{j-1} - 2\Psi_j}{(\Delta x)^2} + V_j \Psi_j]}{\sum_j \Delta x (\Psi_j)^2} \tag{8.116}$$

The algorithms is simple: we first make a guess $\Psi_0^{(0)}, \Psi_1^{(0)}, \cdots, \Psi_N^{(0)}$ for the minimised trial function. Next, we randomly select one of the grids say grid k, and change its wavefunction value by a small amount

$\delta\Psi$. More concisely, we evaluate (8.116) for $\Psi_k^{(0)} \to \Psi_k^{(0)} + \delta\Psi$. If the energy is reduced we accept this change otherwise we reject our change and choose another grid. After repeating this procedure for a sufficient number of updates, we will arrive in the vicinity of a minimised wavefunction. We recall that this stochastic search method is quite similar to the Monte Carlo method in statistical mechanics. You may even accept those changes in which the energy is increased locally. But remember that you should accept those changes probabilistically. The larger the energy change the less probable the change acceptance. The programme `Variational` (for details see appendix 8.F) finds the ground state energy based on the variational principle for a given potential.

8.9 Problems

Problem 8.1 Obtain equations (8.10) and (8.11).

Problem 8.2 Do the remainder calculations to reach from (PsiFourierInt) to (8.10).

Problem 8.3 Verify that the average position of a particle at arbitrary time t (associated with the initial Gaussian wave packet) is $\langle x\rangle = \int_{-\infty}^{+\infty} xP(x,t)dx = \frac{\hbar k_0 t}{m}$.

Problem 8.4

(a) Modify the programme `SchroCrankNicol` such that it can evaluate the particle's energy expectation value. Note that theoretically we have $\langle E\rangle = \frac{\int \Psi^*(x,t)H\Psi(x,t)dx}{\int \Psi^*(x,t)\Psi(x,t)dx}$. Repeat the steps we did and obtain $\langle \hat{p}\rangle$.

(b) Plot the time series of $\langle E\rangle$ for various values of space and time grids Δx and τ. Is the computed $\langle E\rangle$ conserved?

(c) Which choice gives the best result?

Problem 8.5

(a) Numerically obtain the momentum expectation value of a particle moving in a free space.

(b) Obtain the time series of $\langle E \rangle$. Is it conserved? Discuss your results for various values of the system's length L.

Problem 8.6 Analytically obtain the energy expectation value of the initial Gaussian wave packet. Does it remain constant in time? Explain why.

Problem 8.7 In the numerical solution of the symmetric infinite quantum well investigate the effect of varying ϵ on the solution (the ground state in particular). Which choice of ϵ gives the best ground state energy?

Problem 8.8 In the problem of perturbation in the infinite square well, obtain the wavefunction form in the well's region for the particular case when $E = V_b$.

Problem 8.9 Try other values of grid number N and x_{max} in the first odd eigenstate of the harmonic oscillator. Which choice gives the best eigenstate solution?

Problem 8.10

(a) Numerically find the first three eigenvalues of the perturbed infinite square well from equation (8.88) for various values of b and V_b.

(b) Verify that in the limit $b \to 0$ the energies approach the infinite well values.

(c) send $b \to 0$ and $V_b \to \infty$ such that $bV_b = 1$. Find the first few eigenvalues. Note that this limit corresponds to the Dirac delta function potential at $x = 0$. d) Solve the problem analytically in this limit and compare your numerical energies with analytical ones.

Problem 8.11 In the finite square well suppose we have a negative energy eigenvalue for an odd state. Prove that an inconsistency will arise. Obtain the inconsistent condition.

Chapter 9

Molecular dynamics

9.1 Introduction

After practicing numerical techniques for solving quantum problems, we return again to classical systems. This time, we intend to deal with systems that include many particles. Newton's equation of motion governs the multi-particle system dynamics. For each system's particle, we can write a differential equation of motion once the interaction among particles is specified. The numerical method of solving these equations of motion is known as *molecular dynamics*. The name *molecule* interchangeably applies to system particles. Our molecules need not necessarily be atomic-sized particles but can be micron-sized colloids or even larger up to stars. In fact, by adopting a force field among particles, we simulate the system motion, which makes the *molecular dynamics simulation* an appropriate name for this approach. The basic idea of molecular dynamics is to integrate the equations of motion in discrete steps of time τ. We have already done this for simple systems comprising $N = 1$ or $N = 2$ particles in previous chapters. In this chapter, we shall do this job systematically for a system having a large number of particles, say $N = 10^4$. For simplicity, we assume our particles are point-like. Knowing the trajectory of every particle in the system is the most detailed information we can gain from a classical system. It enables us to perform averages, extracts macroscopic quantities, and provides a deep insight into its physical characteristics.

9.2 Inter-particle forces

The basic ingredient in molecular dynamics is intermolecular forces. For simplicity, we assume our particles are point-like, and their internal structure can be ignored. The particles positions and velocities are shown by $\mathbf{r}_1(t), \mathbf{r}_2(t), \cdots, \mathbf{r}_N(t)$ and $\mathbf{v}_1(t), \mathbf{v}_2(t), \cdots, \mathbf{r}_N(t)$ respectively. For notational convenience, we drop the explicit time dependence from these variables afterward. Furthermore, we assume the inter-particle forces do not depend on particles' velocities, hence a potential function $U(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_N)$ describes the interaction among particles. The force $\mathbf{F}_i$ on particle i is given by the gradient of U with respect to the coordinates of particle i:

$$\mathbf{F}_i = -\nabla_i U(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_N) \tag{9.1}$$

In many systems, the force between any two particles i and i solely depends on their distance $r_{ij} = |\mathbf{r}_i - \mathbf{r}_j|$. In this case, the total potential energy U is the sum of pairwise potentials:

$$U(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_N) = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} u(r_{ij}) \tag{9.2}$$

You can verify that this way of writing the limits includes all the pairs. Let us verify it for a $N = 4$ particle system. We have six independent pairs $(1,2), (1,3), (1,4), (2,3), (2,4), (3,4)$. Performing first the sum over i in (9.2) gives:

$$U(\mathbf{r}_1, \mathbf{r}_2, \mathbf{r}_3, \mathbf{r}_4) = \sum_{j=2}^{4} u(r_{1j}) + \sum_{j=3}^{4} u(r_{2j}) + \sum_{j=4}^{4} u(r_{3j}) \tag{9.3}$$

Performing the sums over j gives:

$$U = u(r_{12}) + u(r_{13}) + u(r_{14}) + u(r_{23}) + u(r_{24}) + u(r_{34}) \tag{9.4}$$

The pairwise approximation to the total potential energy is appropriate for simple liquids such as liquid Argon. The quantitative form of $u(r)$ for electrically neutral molecules can be constructed by quantum mechanical calculations. Such calculations are complicated, and it is usually sufficient to choose a simple phenomenological form for $u(r)$. Two essential features of $u(r)$ are a strong repulsion for small r, and a

weak attraction at large r (Allen and Tildesley, 1986; Frenkel and Smit, 2002). The repulsion at small r is a consequence of the Pauli exclusion principle, which implies that the electron wave functions of two molecules must distort to avoid overlap, causing some of the electrons to be in different quantum states. The net effect is an increase in kinetic energy, and the effective repulsive interaction between the electrons, known as *core repulsion*. The dominant weak attraction at larger r is due to the transient mutual polarization of each molecule; the resultant attractive potential is called the van der Waals potential. For quantum mechanical details, you may see the textbook (Schiff, 1968). The simplest model potential which captures these basic features is the one introduced by J. E. Lennard-Jones (Jones, 1924).

$$u(r) = 4\epsilon[(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}] \tag{9.5}$$

The first (second) term models the repulsion at small r (weak attraction at large r). Note that both terms are short-range. The weak attractive potential is sometimes called *van der Waals* potential. The parameters ϵ and σ have energy and length dimensions. Their values depend on the type of atoms. A plot of the Lennard-Jones potential is given in figure (9.1). Moreover, for convenience, we use the abbreviation LJ for Lennard-Jones. To find the inter-molecular distance at which the potential is minimum, we take the derivative of the LJ potential with respect to variable r:

$$\frac{du(r)}{dr} = \frac{24\epsilon}{r}[(\frac{\sigma}{r})^{6} - 2(\frac{\sigma}{r})^{12}] \tag{9.6}$$

Putting $u'(r) = 0$ gives us $r = 2^{1/6}\sigma$. We can achieve considerable savings in computer time by neglecting pair interactions beyond some distance r_c. A common choice for r_c is 2.5σ. We can effectively take $u(r) = 0$ for $r > r_c\sigma$. In other words, we have truncated the LJ potential beyond r_c. Suppose a particle is located at the origin and another particle is located at $\mathbf{r}$. The force exerted on the particle in the origin by the other particle located at the $\mathbf{r}$ is given by the potential gradient $\mathbf{F}(r) = -\nabla u(r) = -u'(r)\hat{r}$. Using (9.6) we have:

$$\mathbf{F}(r) = \frac{24\epsilon}{r}[2(\frac{\sigma}{r})^{12} - (\frac{\sigma}{r})^{6}]\hat{r} \tag{9.7}$$

In a general situation, let us evaluate the force on an arbitrary particle i by an arbitrary particle $j \neq i$. These two particles interact, in a

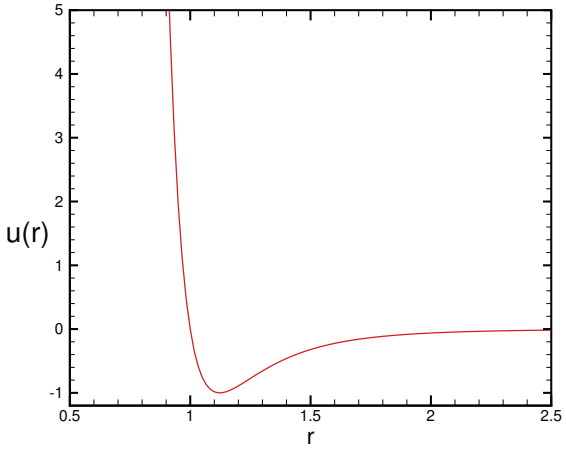

Figure 9.1: *Plot of the Lennard-Jones potential in reduced units.*

pairwise manner, via the term $u(r_{ij})$ in the total potential. Showing this force by $\mathbf{F}_{j\to i}$ we have:

$$\mathbf{F}_{j\to i} = -\nabla_i u(r_{ij}) \tag{9.8}$$

By ∇_i we mean the gradient is taken with respect to the coordinates of particle i i.e.; $\nabla_i = (\frac{\partial}{\partial x_i}, \frac{\partial}{\partial y_i}, \frac{\partial}{\partial z_i})$. Let us evaluate the first component that is $\frac{\partial u(r_{ij})}{\partial x_i}$. Noting that $r_{ij} = \sqrt{(x_i - x_j)^2 + (y_i - y_j)^2 + (z_i - z_j)^2}$ and applying the chain rule gives:

$$\frac{\partial u(r_{ij})}{\partial x_i} = \frac{du(r_{ij})}{dr_{ij}} \frac{\partial r_{ij}}{\partial x_i} = u'(r_{ij}) \frac{x_i - x_j}{r_{ij}} \tag{9.9}$$

Changing x into y and z gives the other two components. Collecting everything together, we find:

$$\mathbf{F}_{j\to i} = \frac{u'(r_{ij})}{r_{ij}}(x_j - x_i, y_j - y_i, z_j - z_i) = \frac{u'(r_{ij})}{r_{ij}}(\mathbf{r}_j - \mathbf{r}_i) \tag{9.10}$$

It is customary to introduce the vector $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ and write (9.10) as follows:

$$\mathbf{F}_{j\to i} = -\frac{u'(r_{ij})}{r_{ij}}\mathbf{r}_{ij} = -u'(r_{ij})\hat{\mathbf{r}}_{ij} \tag{9.11}$$

Vector $\mathbf{r}_{ij}$ connects particle j to particle i. Note that the force exerted on particle i is along the direction joining this particle to particle j. Depending on the value of $u'(r_{ij})$, this force can be attractive or repulsive. According to (9.10) the x component of the force exerted on particle i by particle j turns out to be:

$$F_{x,j\to i} = \frac{u'(r_{ij})}{r_{ij}}(x_j - x_i) \tag{9.12}$$

Similar expressions apply to y and z components. We remark that in some cases, people use a shifted-force potential (Haile, 1992). For inferring the philosophy of using this potential, you may refer to advanced books such as (Haile, 1992). Here we only quote the form of the shifted-force LJ potential $u_s(r)$:

$$u_s(r) = u(r) - u(r_c) - (r - r_c)\frac{du}{dr}|r_c \tag{9.13}$$

Note that $u_s(r)$ and its derivative are continuous at the truncation point r_c.

9.3 Integration from equations of motion

If the inter-particle potential is specified, we can write down the Newton equations of motion for each particle:

$$m_i\frac{d^2\mathbf{r}_i}{dt^2} = \mathbf{F}_i \quad i = 1, 2, \cdots, N \tag{9.14}$$

In what follows, we assume the particles are identical and take their mass to be m. We all know from classical mechanics that it is impossible to find the analytic solution of differential equations (9.14) for a large value of N. Even for $N = 3$, the exact solution is limited to a few special potentials. However, you might be tempted to solve the set of equations numerically by one of the well-known algorithms, such as Runge-Kutta or Verlet. Despite their simple form, the numerical solution does not give a satisfying answer. The inclusion of a repulsive term makes the system disintegrate (particles escape from each other if time is elapsed sufficiently). Even if the inter-particle forces are entirely attractive, but not strong enough, the numerical solution may lead to system disintegration. Of course, if N is as large as the

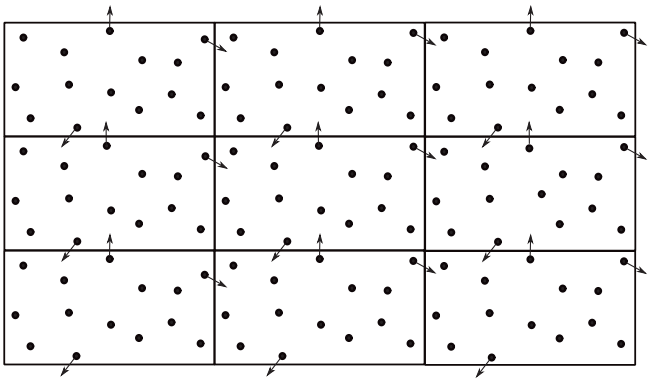

Figure 9.2: *Periodic boundary condition imposed on a two-dimensional system of particles.*

Avogadro number, we can be hopeful that numerical solutions reproduce the correct behaviour of matter in bulk. Unfortunately, even by supercomputers, we can typically simulate 10^{10} particles which are yet too small to model a bulk piece of matter. Actually, we need a surface potential to sustain the system, but it is pretty challenging to model such forces properly. Physicists have adopted an alternative method. According to this method, fictitious particles act on the system particles especially, those located near the boundaries. Their role is to prevent the system from disintegrating. They provide confinement for the system particles to remain in a limited region of space. We call this region the simulation box and normally take it to be a square. A conventional choice is to take the imaginary particles as the periodic images of the system particles in neigbouring image boxes. Each image box contains the original particles in the same relative positions as the central box. In figure (9.2) we have shown some of these image boxes and their image particles for a two-dimensional system of particles confined in a rectangular box of edge lengths L_x and L_y. The simulation box is surrounded by similar space-filling boxes, which are duplicates of the main box (central box). Each duplicated box is specified by the coordinates of its left bottom corner $(n_x L_x, n_y L_y)$ in which n_x and n_y are integer numbers. Note that the case $n_x = n_y = 0$ corresponds to the simulation box itself. For simplicity, we use the shorthand notation (n_x, n_y) for the neighbouring box coordinates. Each system particle

not only interacts with the remaining $N-1$ particles but also interacts with their imaginary particles located in the neighbouring boxes if we want to prevent the system from being disintegrated. Note that each system particle possesses eight images. The coordination of particle i images turns out to be $(x_i + n_x L_x, y_i + n_y L_y)$ with the exclusion of $n_x = n_y = 0$. Evidently, we cannot consider the interaction with all the boxes' particles (there are infinite boxes actually!). At this stage, an approximative scheme should be devised. This approximation is called *minimal image approximation*. According to this approximation, Since the length of the simulation box edge is normally much larger than the interaction range between particles, at each instant of time, every system particle i can only interact with particle $j \neq i$ or one of its images. Consider a pair of particles i and j. Clearly, among particle j and its neighbours, one of them is nearest to particle i. We call this particle the minimal j neighbour to particle i and show the corresponding box number by (n_x^*, n_y^*). Note the x-distance between particle i and the its minimal j image is less than $\frac{L_x}{2}$. Similarly, the y-distance between particle i and its minimal j image is less than $\frac{L_y}{2}$. We can now devise a method for evaluating the instantaneous force exerted on every particle i in the simulation box at timestep t. For each particle $j \neq i$ determine its minimal image with respect to particle i. Calculate the force on particle i by this image particle and then sum over all particles j ($N-1$ term). Once the total force is computed, we can use a numerical algorithm to update the positions and velocities of system particles. Note that after updating the positions and velocities of the system particles, the positions of each particle's images in other boxes will be correspondingly updated. As a particle moves in the simulation box, its periodic images move in their own image boxes. Hence only the motion of the particles in the central box needs to be followed. When a particle enters or leaves the central box, the move is accompanied by an image of that particle leaving or entering a neighboring box through the opposite face. In order to simulate this effect, we bring back the exited particle inside the box by adding/subtracting an appropriate integer multiple of L_x or L_y to/from its relevant coordinate. For example, if the updated y coordinate of a particle is $1.0001L_y$ we reupdate its y coordinate to value $0.0001L_y$. Sometimes people call this procedure the *periodic boundary condition* (PBC). Implementing PBC allows you to deal with only N particles inside the central box i.e., the simulation box. We would like to emphasize that

the validity of the *minimal image approximation* crucially depends on the interaction range among particles. For short-range potentials such as Lennard-Jonnes, this approximation remains valid. However, if the potential includes long-range terms such as a Coulomb interaction, the interaction with other images particle beyond the minimal one should be considered. This is performed by a special method and includes a sum over all the image particles. The technique is known as the Ewald sum in the literature (Allen and Tildesley, 1986).

9.3.1 The numerical algorithm

Now that we can compute the total force on every system particle $i = 1, 2, \cdots, N$, we need a numerical method for computing the trajectory of each particle. This algorithm should conserve the phase-space volume and be consistent with the known conservation laws. One of the most adopted and commonly used algorithms in molecular dynamics simulations is *velocity Verlet* algorithm (Verlet, 1967), which was explained in chapter three. We recall that the positions and velocities of particles are updated according to the following scheme:

$$x_i^{n+1} = x_i^n + v_{xi}^n \tau + \frac{1}{2} a_{xi}^n \tau^2 \quad i = 1, 2, \cdots, N \tag{9.15}$$

$$v_{xi}^{n+1} = v_{xi}^n + \frac{1}{2}(a_{xi}^n + a_{xi}^{n+1})\tau \quad i = 1, 2, \cdots, N \tag{9.16}$$

Analogous expressions apply to y and z components. Note the implementation order of the above commands in programming. First, the positions should be updated, and then from the updated positions, you should compute the updated acceleration a_i^{n+1} in order to compute the updated velocities.

9.3.2 Reduced units

Before discussing the structure of a molecular dynamics programme, we should give our final remark. Since our particles have atomic scale, their mass, length, and other potential parameters have very tiny values compared to the macroscopic values we are familiar with. For instance, the parameter values of the LJ potential for argon are as follows: $\sigma = 3.4 \times 10^{-10}\ m$, $\epsilon = 1.65 \times 10^{-21}\ J$ and $m = 6.69 \times 10^{-26}\ kg$. If we adopt the conventional SI unit, we have to deal with very small

or very huge quantities. This causes dramatic systematic and round-off errors which lead to incorrect answers. In order to circumvent the problem, it is convenient to choose an appropriate unit so that the computed quantities are neither too small nor too large. People frequently use a unit where mass, energy, and length are the basic quantities. The natural choice of unit is to take the particle mass m as the mass unit, σ as the length unit, and ϵ as the energy unit. We can thus express the other quantities in terms of these units. Let us express four important quantities that are *time, velocity, force* and *pressure*. In terms of m, σ and ϵ we have: $\sigma(\frac{m}{\epsilon})^{\frac{1}{2}} = 2.17 \times 10^{-12}\ s, (\frac{\epsilon}{m})^{\frac{1}{2}} = 1.57 \times 10^{2}\ m/s$, $\frac{\epsilon}{\sigma} = 4.85 \times 10^{-12}\ N$ and $\frac{\epsilon}{\sigma^2} = 1.43 \times 10^{-2}\ Pa$ as the time, velocity, force and pressure units respectively (H. Gould and Chriastian, 2006). As a clarification, if we take $\tau = 0.01$ in our MD programme, it means that in SI units, the timestep is $\tau = 0.01 \times 2.17 \times 10^{-12} = 2.17 \times 10^{-14}\ s$.

9.4 A molecular dynamics programme

We now develop a molecular dynamics (MD) programme to simulate a system of particles interacting via a short-range potential so that we can use minimal image approximation. For simplicity, we work in two dimensions because the calculations are not so time-consuming, and all the techniques we shall discuss can be applied to three dimensions straightforwardly. Moreover, we assume our point-like particles interact via Lennard-Jones' potential. To update the system status in time, we should determine the initial conditions specified by $6N$ constants $\mathbf{r}_1(0), \mathbf{r}_2(0), \cdots, \mathbf{r}_N(0), \mathbf{v}_1(0), \mathbf{v}_2(0), \cdots, \mathbf{r}_N(0)$ respectively. An appropriate choice of the initial conditions is more challenging than might first appear. If you naively give initial positions and velocities to particles, the programme fails to produce reasonable trajectories, and normally, after some timesteps, the particles' velocities become extremely large. In an MD programme, the initial positions are normally determined according to the system's state we wish to simulate. For example, if the system is a dilute gas, we can choose the initial positions of the particles by placing them at random, making sure that no two particles are too close to one another. If two particles were too close, they would exert a very large repulsive force F on each other. Any simple finite difference integration method would break down because the condition $(F/m)(\tau)^2 \ll \sigma$ would not be fulfilled. In a ran-

dom setting of particles, you can simply verify that if the separation between two particles is greater than $2^{1/6}\sigma$, this condition is satisfied. Unfortunately, there is no general method for the initial placement of particles, and every problem should be treated particularly. If you are investigating the properties of a solid phase, then you can place the particles initially on a regular lattice that corresponds to the system's solid phase structure. Contrary to the initial placement of particles, for the initial velocity assignment, there is a general procedure. Usually, the temperature of the system under consideration is constant, and the velocity components distribution would be Gaussian. Therefore, a plausible scheme for initial velocities assignments would be to extract them from a Gaussian distribution function having a width proportional to k_BT. Later we will discuss this method in more detail. I have explained all the necessary ingredients for constructing a molecular dynamics programme. From the programming viewpoint, we need some arrays to store the values of particles' positions and velocities. As far as elementary quantities need to be computed, we do not need to store the system configurations at all timesteps. Many macroscopic quantities such as temperature and pressure, can be obtained by time averaging over their instantaneous values. Hence, we only need to store the system configuration (particles' positions and velocities) at the current timestep. From a computational point of view, we need six arrays X, YZ, V_x, V_y, and V_z for this storage. Of course, three more arrays A_x, A_y, and A_z are needed to store the acceleration components. All these arrays have a dimension equal to the number of particles N. For example, $Y[i]$ denotes the y component of particle i at the current time step t. In appendix 9.A we will explain the detailed structure of a molecular dynamics programme. In the next section, we will discuss the physical properties that can be investigated via molecular dynamics simulations.

9.5 Macroscopic quantities

Relating the information at the microscopic level to macroscopic quantities such as temperature and pressure is one of the fundamental goals of physics. For almost more than a century, statistical mechanics provided such bridging for us, thanks mainly to the efforts of Ludwig Boltzmann. With the advent of computers, molecular dynamics opened a novel stride. The main goal of a molecular dynamics pro-

gramme is to extract macroscopic quantities by averaging over simulated microscopic trajectories of system particles. For centuries this has been a great dream for statistical physicists. Computers made this long-lived dream fulfilled in the late 1950th through the pioneering work of Alder and Wainwright (Alder and Wainwright, 1959). From statistical physics, we know that macroscopic quantities of interest are time averaged over certain collective microscopic behaviour of system particles. Molecular dynamics enables us to obtain the phase space trajectory of each system particle and hence, provides us a framework to describe the system from a macroscopic point of view. One practical question is whether our time intervals are sufficiently long to allow the system to explore the whole accessible phase space and give meaningful averages. Calculations in statistical mechanics are done by replacing time averages with ensemble averages over all possible configurations. The quasi-ergodic hypothesis asserts that these two types of averages give equivalent results if the same quantities are held fixed. In statistical mechanics, we work in ensembles. Each ensemble is specified with some constraints. For example, if E, V, and N are held fixed we have a microcanonical ensemble. At first, you may infer that there will be no ensemble in molecular dynamics simulations because N, V, and E are constant. However, we will see that to simulate realistic situations, we will have to deal with cases in which other variables remain constant. For example, if we want to simulate a melting phenomenon, we have to keep the pressure constant instead of the volume. Therefore, the ensemble concept will appear in molecular dynamics simulation in a similar sense as in statistical mechanics. For the sake of simplicity, it is natural to start with the NVE ensemble. We shall now try to see how we can obtain macroscopic quantities from microscopic details provided by MD simulations in a microcanonical ensemble.

9.5.1 Temperature

Temperature is the system's average kinetic energy. According to the equipartition theorem, the system temperature at time t is given by the following relation:

$$T(t) = \frac{2K(t)}{dNk_B} = \frac{1}{dNk_B} \sum_{i=1}^{N} m_i \mathbf{v}_i(t).\mathbf{v}_i(t) \tag{9.17}$$

where k_B is the Boltzmann constant, d is the spatial dimension, and $K(t)$ is the system instantaneous total kinetic energy. Note that the particles' masses can be different in principle. The temperature that we measure in a laboratory experiment is the mean temperature, which corresponds to the time average of $T(t)$ over many configurations of the particles:

$$T = \frac{1}{dNk_B} \sum_{i=1}^{N} m_i \overline{\mathbf{v}_i(t).\mathbf{v}_i(t)} \tag{9.18}$$

Overline denotes time averaging.

9.5.2 Pressure

Another macroscopic quantity of interest is the mean pressure. The pressure is related to the force per unit area normal to an imaginary surface of an isotropic fluid system. In a solid system, the pressure concept is replaced with the stress tensor. Coming back to a fluid system and according to Newton's second law, this force is related to the momentum exchange per unit time that crosses the surface. For simple gases, with no interaction among particles, we can use kinetic energy arguments to obtain the pressure. In a general situation, when there are interactions among particles, use should be made of other relations. This relation involves the concept of *virial*. It can be shown that virial is related to pressure. In general, the momentum flux across a surface has two contributions. The kinetic contribution, NkT/V, where V is the system volume, is due to the motion of particles and is derived in many texts using simple kinetic theory arguments. The other contribution arises from the momentum transferred across the surface due to the forces between particles on different surface sides. It can be shown that the instantaneous pressure at time t, including both contributions to the momentum flux, is given by:

$$p(t)V = Nk_BT(t) + \frac{1}{d} \sum_{i<j} \mathbf{r}_{ij}.\mathbf{F}_{ij} \tag{9.19}$$

Recall that $\mathbf{r}_{ij} = \mathbf{r}_i - \mathbf{r}_j$ and $\mathbf{F}_{ij}$ is the force exerted on particle i due to particle j. The mean pressure, $p = \overline{p(t)}$, is found by computing the time average of the instantaneous pressure $p(t)$. We introduce other macroscopic quantities later. Let us see the results of a molecular dynamics simulation.

9.6 A molecular dynamics simulation

In this section, the results of a molecular dynamics simulation programme are shown for a system of interacting particles. The interactions among particles are assumed to be short-range. For simplicity, we work in two dimensions and consider a system consisting of $N = 64$ particles interacting via Lennard-Jones (LJ) potential in a square simulation box with length $L = 10$ in reduced units. The reduced density turns out to be $\rho^* = \frac{64}{100} = 0.64$. The particles are initially placed on a regular square lattice points. For initial velocities, we choose the velocity modulus of all particles identical i.e.; $v_i = v = \sqrt{\frac{2k_B T_{init}}{m}}$ in which T_{init} is the initial system temperature. In our problem, we take $T_{init} = 1$. The velocities directions are randomly chosen from the interval $[0, 2\pi]$. As mentioned, we rescale the velocities so that the system's total linear momentum becomes zero. The programme `2DLJverlet` (see Appendix 9.A for details) numerically solves the Newton equations of motion for a two-dimensional system of particles interacting via LJ inter-molecular potential under periodic boundary conditions with the velocity Verlet algorithm. In figure (9.3), the evolution of the potential energy per particle is depicted. The velocity Verlet algorithm with $\tau = 0.01$ has been implemented to integrate Newton's equations of motion. Figure (9.4) exhibits the trajectory of particle 9 during $T = 10000$ timesteps which, corresponds to $t^* = 10$ time units. Note the complicated structure of the trajectory. The initial position of particle 9 is $(1.11, 2.22)$ (bottom left corner). You see that this particle considerably deviates from its initial position, which means that the system cannot be in a solid phase despite initially it had a regular solid-like structure. Figure (9.5) shows the time series of potential, kinetic and total energies per particle. After some transient timesteps, the system comes to equilibrium in which macroscopic collective quantities are in a steady-state and fluctuate about a mean value. The mean values are $-2.17, 0.38$, and -1.78 for potential, kinetic, and total energy correspondingly. Note that the equilibrium temperature is 0.38 in our reduced unit. In fact, the system temperature has fallen from the initial value $k_B T = 1.$ to $k_B T = 0.38$. The interaction among particles has caused this effect. Moreover, note that despite the total energy per particle E/N should be constant, it undergoes fluctuations which should be a computational error. Try to see if fluctuations are reduced if you decrease τ.

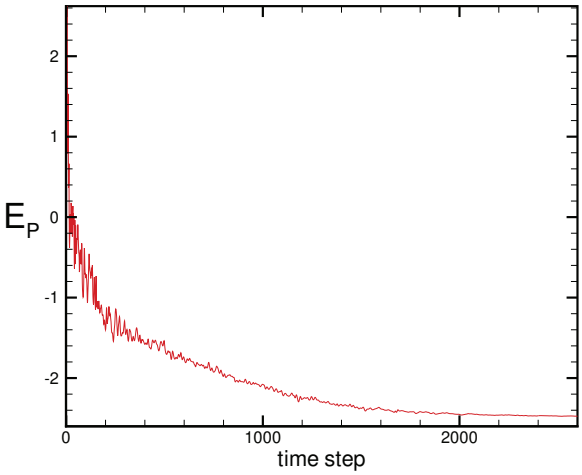

Figure 9.3: *Time evolution of the potential energy per particle E_p/N for a $N = 64$ particle system in two dimensions with the reduced density $\rho^* = 0.64$. The particles interact with each other via the Lennard-Jones potential. The system has been prepared with an initial temperature of $T_{init} = 1$.*

9.6.1 Velocity distribution

One of the tests for the accuracy and validity of our MD programme is its capability of reproducing the correct velocity distribution of particles. From statistical physics, we know that in equilibrium, the velocity distribution of each velocity component is Gaussian with zero mean and a standard deviation proportional to system temperature. Let us see the predictions of our MD programme for a three-dimensional system. The programme `3DLJverlet` (see Appendix 9.B for details) numerically solves Newton's equations of motion for a three-dimensional system of particles interacting via LJ inter-molecular potential under periodic boundary conditions with the velocity Verlet algorithm. The number of particles is taken $N = 108$, and the system box (cube)

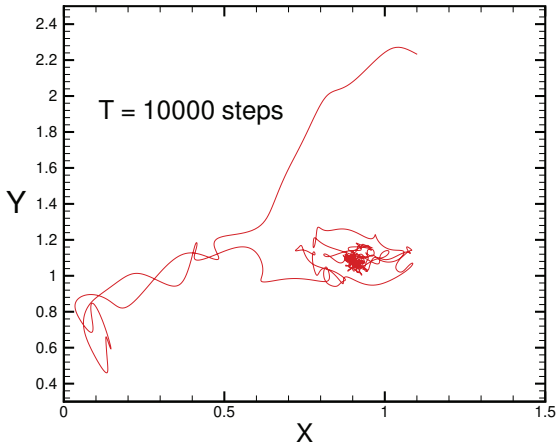

Figure 9.4: *Trajectory of 9-th particle in a 2D system with* $\rho^* = 0.64$ *during* $T = 10^4$ *timesteps.*

size is $L_x = L_y = L_z = 8.94$. The system evolves for $T_{eq} = 10^4$ timesteps to ensure it has reached its equilibrium. Then we evaluate the average system temperature $k_B\bar{T}$. Particles are initially set on a 3D FCC lattice with an initial Gaussian velocity distribution corresponding to initial temperature $kT_{init} = 2.$. After equilibration, the system temperature becomes $T_{eq} = 2.6$. Next, we let the system evolve for further timesteps for averaging the velocities distributions. In principle, each component of velocities can take very large values in magnitude. However, in practice, each component lies with a very high probability in the interval $[-4\sqrt{\frac{k_BT}{m}}, 4\sqrt{\frac{k_BT}{m}}]$ in which T is the average system temperature. Remind you that after equilibrium, each component of particles velocity should have the following normalised Gaussian distribution function (Haile, 1992):

$$f(v_x) = \sqrt{\frac{m}{2\pi k_B T}} exp(-\frac{mv_x^2}{2k_BT}) \tag{9.20}$$

As you see, the standard deviation is $\sigma = \sqrt{\frac{k_BT}{m}}$. To compute the velocity component distribution function, we divide each velocity compo-

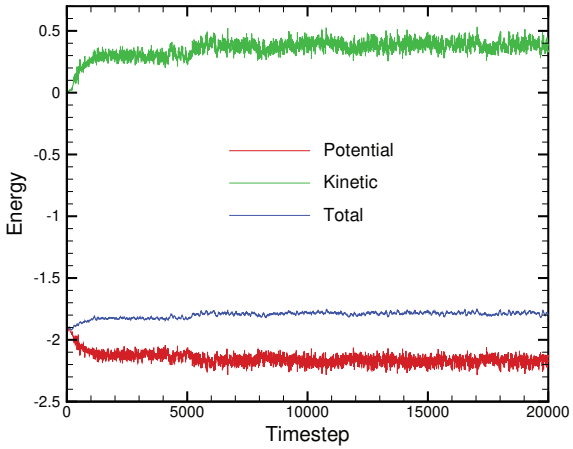

Figure 9.5: *Time evolution of potential, kinetic and total energies per particle for $t^* = 200$ time units for a $N = 64$ particle LJ system in two dimensions with $\rho^* = 0.64$.*

nent interval into $N_{bin} = 50$ bins. Then for each particle, we evaluate the bin number of its velocities components and increase the associated bin counter. In other words, we build up velocity components histograms. Figure (9.6) shows $f(v_x)$ for various average temperatures. You see, there is a good agreement between numerical and theoretical results. From the graph, the most probable and the mean value of the velocity component is zero.

9.6.2 Equation of state

Molecular dynamics simulation allows us to obtain the equation of state i.e.; the relationship between temperature, pressure, and density. Let us find this relation for a two-dimensional LJ system. The number of particles is taken as $N = 64$ initially set on a regular lattice square. The square box size is $L = 12$, and the initial temperature is $k_B T = 2$ in reduced units. We let the system evolve for $T_{eq} = 10^4$ timesteps to ensure it has reached equilibrium. Figure (9.7) shows the temporal

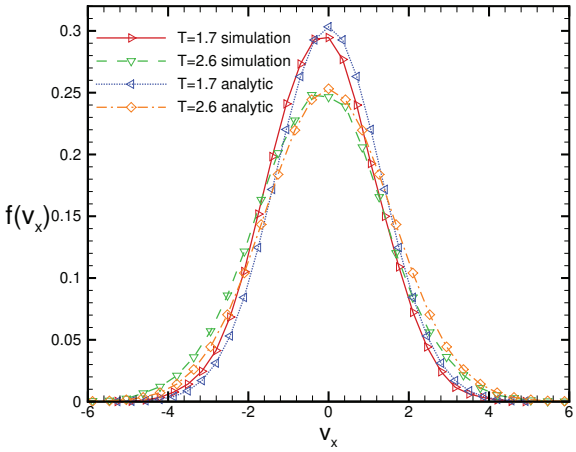

Figure 9.6: *Velocity component distribution function $f(v_x)$ for a 3D Lennard-Jones system having $N = 108$ particles. A comparison to the theoretical result is shown.*

evolution of pressure.

$$p(t) = \frac{Nk_BT(t)}{2A} + \frac{1}{2A}\sum_{i,j}\mathbf{r}_{ij}.\mathbf{F}_{ij} \tag{9.21}$$

Mass density in reduced units is $\rho^* = \frac{Nm^*}{L^2} = 0.444$. As you see, the pressure time series soon reaches a steady state. The fluctuation remains large, which is due to the smallness of particle number N. The time-average pressure is $\bar{p} = 1.05$, and the system average temperature turns out to be $k_B\bar{T} = 1.78$ in reduced units which is less than the initial temperature $k_BT_{init} = 2$. The ideal gas pressure associated with these values of density and temperature is $p_{id} = \frac{Nk_BT}{A} = 0.79$. The relative excess pressure is $p_{ex,rel} = \frac{\bar{p}-p_{id}}{p_{id}} = 0.32$ which is quite large. The difference comes from the nature of interaction among particles. Let us look at the variation of compressibility factor $Z = \frac{pA}{Nk_BT}$ versus reduced density ρ^*. For the ideal gas, one has $Z = 1$. Any difference from unity comes from the interaction among particles. Figure (9.8) shows the dependence of Z on ρ^*. We initially set the

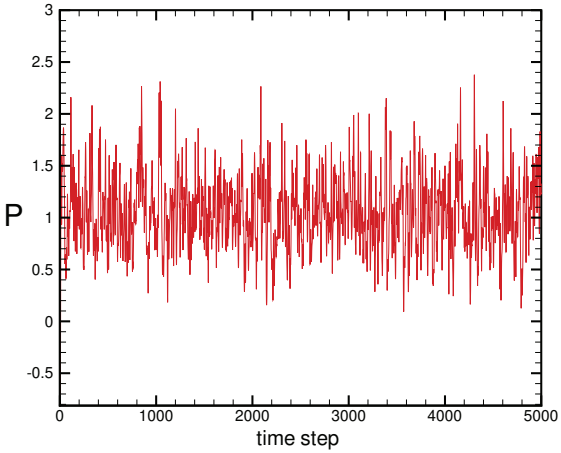

Figure 9.7: *Instantaneous pressure for a 2D Lennard-Jones system of reduced mass density* $\rho^* = 0.444$ *initially prepared at* $k_B T_{init} = 2$.

box size at $L = 20$ and gradually reduced it by 2 percent in each size reduction step. There is no singularity or cusp in the $Z - \rho$ diagram. This suggests that no phase transition can occur in this 2D system. The well-known Mermin-Wagner theorem confirms this statement.

9.6.3 Heat capacity

Another quantity of interest is the thermal capacity. This notion, together with latent heat, was first introduced by Scottish physician, physicist, and chemist Joseph black in the nineteen century. Experimentally it is much easier to measure the heat capacity in constant pressure, because, in most of the situations where heat is transferred, the pressure is constant or can be quite easily held constant. On the contrary, it is not easy to keep volume constant when we heat an object. In simulations, the converse is true, because it is not straightforward to keep the pressure constant in an MD programme. Heat capacity at constant volume $C_V = (\frac{\partial E}{\partial T})_V$ is an example of a linear response function, that is, the response of the system energy to a change in its temperature. It can be quite easily computed in an MD simu-

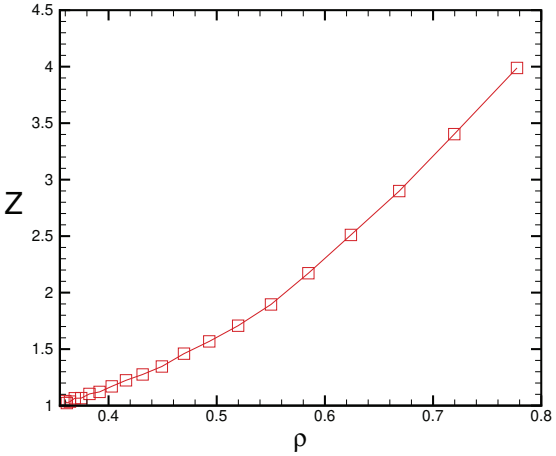

Figure 9.8: *Compressibility factor Z vs density for a 2D Lennard-Jones system of $N = 144$ particles.*

lation. For this purpose, we should know the dependence of system energy E on temperature $E(T)$. Although, a molecular dynamics simulation gives T as a function of E, but once we obtain $E(T)$ we can simply converse it and find $E(T)$. We learnt that the total energy E is determined by the initial conditions, and the temperature is a derived quantity that is found only after the system has reached thermal equilibrium. The temperature can be changed to a desired value by adjusting the energy E. This is normally achieved by rescaling the particle velocities. To obtain the curve $T(E)$ we rescale the velocities by an appropriate factor λ that is $v_\alpha[j] = \lambda v_\alpha[j]$ $j = 1, \cdots, N$ $\alpha = x, y, z$. This changes the system's energy. Then we let the system evolve until it reaches its new equilibrium, with a new temperature. We repeat this procedure, and for each rescaling, we find a new point in the $E - T$ plane.The collection of these points gives us the desired $E(T)$ curve. Once this curve is obtained, we can implement a numerical differentiation scheme to numerically evaluate $C_V = (\frac{\partial E}{\partial T})_V$. Let us see the result for a 3D system of particles with Lennard-Jones' potential. After each velocity rescaling, the system evolves for $T_{eq} = 10^4$ timesteps to ensure it has reached its new equilibrium then we evaluate the av-

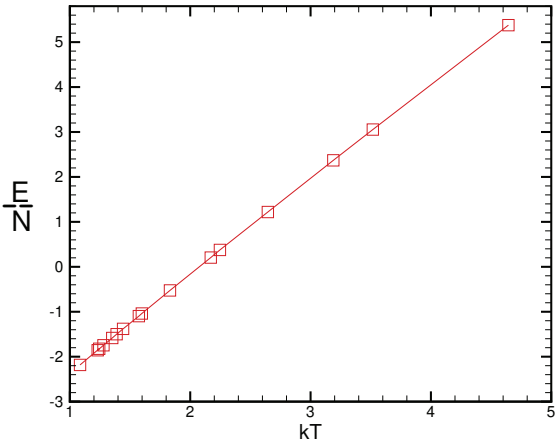

Figure 9.9: *Energy per particle vs temperature for a 3D Lennard-Jones system with $\rho = 0.74$. The simulation has been performed with $N = 256$ particles.*

erage system temperature $k_B\bar{T}$. Figure (9.9) exhibits the energy per particle curve versus T. Figure (9.10) shows a similar diagram for a different density $\rho = 1.45$. As you can see, for these densities, the specific heat capacity in constant volume equals the curve slope. For $\rho^* = 0.74$ and $\rho^* = 0.1.45$ the specific heat capacities turn out to be 2.11 and 3.19, respectively. You see that there is a strong density dependence. Remind you that the ideal gas specific heat capacity is 1.5 in our reduced units.

9.7 Triangular lattice in two dimensions

In this section, we show how to put particles on a two-dimensional triangular lattice. Triangular lattice is important in solid-state physics because it is the 2D analogue of the ubiquitous fcc structure in 3D. Generally speaking, to simulate a solid, we need to choose the shape of the simulation box to be commensurate with the symmetry of the solid phase of the system. This choice is necessary even though we

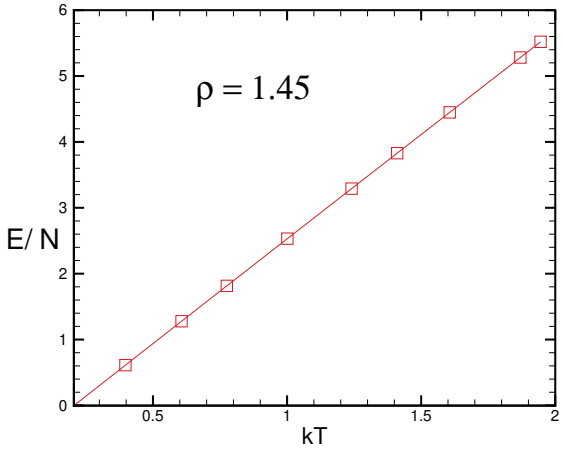

Figure 9.10: *Energy per particle vs temperature for a 3D Lennard-Jones system with the reduced $\rho^* = 1.45$. The simulation has been performed with $N = 500$ particles. Random initial conditions have been used.*

have used periodic boundary conditions to minimize surface effects. If the box does not match the correct crystalline structure, the particles cannot form a perfect crystal, and some of them will wander around in an endless effort for their *correct* positions. Consequently, the simulation of a small system would lead to spurious results. In a triangular lattice, each particle has six neighbours. The commensurate box is a rectangle having side lengths L_x and $L_y = \frac{\sqrt{3}}{2} L_x$. Showing the nearest neighbour distance by a, first, we relate the number density $\rho = \frac{N}{A}$ to lattice constant a. First note that $\rho = \frac{N}{L_x L_y} = \frac{2N}{\sqrt{3} L_x^2}$. In order to construct a triangular lattice with a given density ρ and N points, you should first determine L_x. This is simply achieved: $L_x = 3^{-1/4} \sqrt{\frac{2N}{\rho}}$. The other length L_y is determined from $L_y = \frac{\sqrt{3}}{2} L_x$. On the other hand, we know that each lattice point is connected to six triangles. Each triangle has three vertices therefore, every two triangles contain one lattice point. Triangle area is $s = \frac{\sqrt{3} a^2}{4}$ hence we have:

$\rho = \frac{1}{2s} = \frac{2}{\sqrt{3}a^2}$. This gives a in terms of ρ as follows:

$$a = 3^{-1/4}\sqrt{\frac{2}{\rho}} \tag{9.22}$$

Once the lattice constant a is found, it would be an easy task to put N points on a triangular structure. Let us now evaluate the potential energy per particle for a 2D gas with LJ interaction initially placed statically on points of a triangular lattice with periodic boundary conditions. Performing the sums with a computer, we find $\langle E_P \rangle = -0.1418$ in the LJ energy unit. For $\rho = 1.1547$ we obtain $\langle E_P \rangle = -2.905$ in the LJ energy unit. Note the high sensitivity of E_P to density ρ. For LJ interaction among particles, a triangular lattice has lower energy. In exercises, you will compute the potential energy (per particle) for a square lattice and see that it is higher than in a triangular one. In other words, the square lattice is unstable, and if you give a slight velocity to particles, they will evolve into a stable triangular lattice. Figure (9.11) shows the LJ system's snapshot initially prepared on a square lattice. As you see, the unstable square structure is substituted by the stable triangular one. The vacancies correspond to the incommensurate state of the square box.

9.8 Structural and static properties

Specification and quantification of structural properties of a many-body system are of prime importance in physics. The ordering degree in a system can be expressed in terms of its structural characteristics. They provide useful statistical information about how the system particles are spatially and orientationally organised. Insight into the structure of a many-body system can be gained by looking at how the positions of the particles are correlated with one another due to their interactions. Suppose there are N point-like particles in a volume V. We denote the global particle number density with $n = \frac{N}{V}$. Some useful definitions are in order. The n-particle probability density function $\rho_n(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_n)$ gives the probability $\rho_n(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_n) d\mathbf{r}_1 d\mathbf{r}_2 \cdots d\mathbf{r}_n$ that the volumes $d\mathbf{r}_1$ around $\mathbf{r}_1$, $d\mathbf{r}_2$ around $\mathbf{r}_2$, etc. contain precisely one atom each. The n particle probability distribution function is defined to be:

$$g_n(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_n) = \frac{\rho_n(\mathbf{r}_1, \mathbf{r}_2, \cdots, \mathbf{r}_n)}{\rho_1(\mathbf{r}_1)\rho_1(\mathbf{r}_2)\cdots\rho_1(\mathbf{r}_n)} \tag{9.23}$$

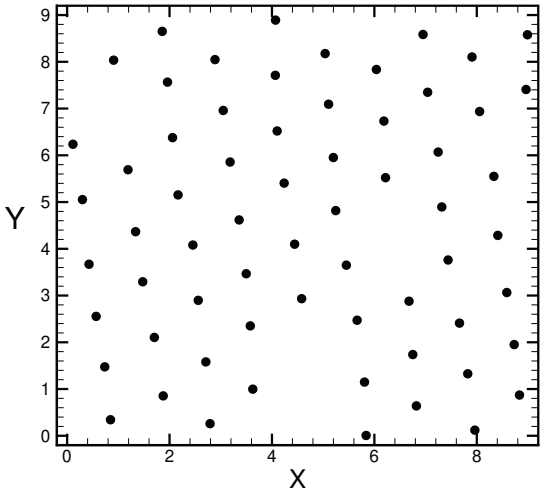

Figure 9.11: *A simulated snapshot with* $\tau = 0.01$ *at* $t = 1$ *for a* $N = 64$ *LJ system initially placed on a square lattice of side* $L = 9$ *with assigned velocity components randomly chosen from the interval* $[-0.01, +0.01]$.

It can be shown (Sólyom, 2007) that it is sufficient to know the one and two-particle distribution functions for the description of structural properties. These distribution functions can be determined from experiments. Denoting the position of particle i by $\mathbf{R}_i$ It turns out:

$$\rho_1(\mathbf{r}) = \langle \sum_{i=1}^{N} \delta(\mathbf{r} - \mathbf{R}_i) \rangle \tag{9.24}$$

in which $\langle\ \rangle$ denotes configurational or thermal averaging. Similarly the two-particle density $\rho_2(\mathbf{r}_1, \mathbf{r}_2)$ is given by:

$$\rho_2(\mathbf{r}_1, \mathbf{r}_2) = \langle \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \delta(\mathbf{r}_1 - \mathbf{R}_i)\delta(\mathbf{r}_2 - \mathbf{R}_j) \rangle \tag{9.25}$$

From these definitions, we find:

$$\int_V \rho_1(\mathbf{r}) d\mathbf{r} = N \tag{9.26}$$

$$\int_V \rho_2(\mathbf{r}_1, \mathbf{r}_2) d\mathbf{r}_2 = (N-1)\rho_1(\mathbf{r}_1) \tag{9.27}$$

If long-range order is present, the positions of the two atoms are correlated even if the separation between $\mathbf{r}_1$ and $\mathbf{r}_2$ is very large. If, however, there is no long-range order, then the correlation between atomic positions is washed out at large separations, and we have:

$$\rho_2(\mathbf{r}_1, \mathbf{r}_2) \to \rho_1(\mathbf{r}_1)\rho_2(\mathbf{r}_2), \quad if\ |\mathbf{r}_1 - \mathbf{r}_2| \to \infty \tag{9.28}$$

Pair correlation function $C(\mathbf{r}_1, \mathbf{r}_2)$ measures the degree of spatial correlation and is defined as follows:

$$C(\mathbf{r}_1, \mathbf{r}_2) = \rho_2(\mathbf{r}_1, \mathbf{r}_2) - \rho_1(\mathbf{r}_1)\rho_2(\mathbf{r}_2) \tag{9.29}$$

This correlation function indicates whether the presence of an atom at $\mathbf{r}_1$ affects the probability of finding another atom at $\mathbf{r}_2$. For perfectly random atomic arrangements, the correlation function is identically zero. For amorphous systems with short-range order, the function takes finite values at small separations and drops off exponentially at large distances. On the other hand, for crystalline samples, the function shows the same periodicity of the underlying structure even at large separations. If the system is homogeneous then $\rho_1(\mathbf{r}_1)$ does not depend on $\mathbf{r}_1$. Moreover, the two-point particle density $\rho_2(\mathbf{r}_1, \mathbf{r}_2)$ depends only on the difference $\mathbf{r}_1 - \mathbf{r}_2$. It can be shown that (Sólyom, 2007)

$$\rho_2(\mathbf{r}_1 - \mathbf{r}_2) = \frac{1}{V}\langle \sum_{i=1}^{N} \sum_{j=1, j\neq i}^{N} \delta(\mathbf{r}_1 - \mathbf{r}_2 - \mathbf{R}_i + \mathbf{R}_j) \rangle \tag{9.30}$$

Denoting $\mathbf{r}_1 - \mathbf{r}_2$ by $\mathbf{r}$ for homogeneous systems we have $g_2(\mathbf{r}) = \frac{1}{n^2}\rho_2(\mathbf{r})$. In isotropic systems $g_2(\mathbf{r})$ is only a function of $r = |\mathbf{r}|$: $g_2(\mathbf{r}) = g_2(r)$. Normally the subscript "2" is dropped. The quantity $g(r)$ is called *radial distribution function* and provides valuable structural information for us. It measures the degree of spatial correlation among particles. A meaningful question regarding the distribution of particles in an isotropic system is how many particles are within the differential volume element $d\mathbf{r}$ about the point $\mathbf{r}$ from a typical particle on average? This number is proportional to n and $d\mathbf{r}$. The proportionality constant is $g(\mathbf{r})$. Integration gives:

$$\int n g(\mathbf{r}) d\mathbf{r} = N - 1 \approx N \tag{9.31}$$

For an ideal gas, there are no correlations between the particles, and the normalization condition implies that $g(r) = 1$ for all r. For the Lennard-Jones interaction, we expect that $g(r) \to 0$ as $r \to 0$, because the repulsive force between particles increases rapidly as $r \to \infty$. We also expect that $g(r) \to 1$ as $r \to \infty$, because the correlation of a given particle with the other particles decreases as their separation increases. Lastly, we note that for an isotropic system, we have $C(r) = g(r) - 1$. For a general system, the computational algorithm for obtaining $g(\mathbf{r})$ is that at each timestep, you perform a loop on all the particles, and for each particle $j = 1, \cdots N$ count the number of particles inside a small volume element ΔV about $\mathbf{r}_j$. Let us denote this number by $n_j[t]$. The radial distribution function $g(\mathbf{r})$ turns out to be:

$$g(\mathbf{r}) = lim_{\Delta V \to 0} \frac{1}{N(T - T_{eq})} \sum_{j=1, t=T_{eq}}^{N,T} n_j[t] \tag{9.32}$$

In fact (9.32) is a double averaging over particles and time where time averaging has been computed after the system has reached equilibrium after a transient T_{eq} timesteps. If the system is homogeneous and isotropic, you need not change the volume element orientation ΔV. Any direction gives the same results. Let us compute the radial distribution function for a Lennard-Jones system. The subroutine `RDF` (see Appendix 9.C for details) computes the radial distribution function for a 2D LJ system. For simplicity, we consider a system of $N = 64$ particles that are fixed on the nodes of a triangular lattice with $Lx = 8$ and $Ly = \frac{\sqrt{3}}{2} Lx$. From the structure of a 2D triangular lattice, we know that the next nearest neighbour (NN) distance is $\sqrt{3}a = 1.73a$, the third nearest neighbour distance is $2a$ etc where a is the lattice constant (nearest neighbour distance). Figure (9.12) shows the radial distribution function $g(r)$. Note that $a = 1$, and $\rho = 1.156$ in this problem. As you see, the first peak is located at $r = 1$. Others are located at integer-valued r and the distance of the second NN, third NN, etc. Let us increase the density from $\rho = 1.1556$ to $\rho = 1.314$ by decreasing L_x to 7.5 while N is kept fixed at 64. Moreover, the system temperature is raised from zero by giving initial random velocities leading to the initial temperature $k_B T = 2$. After equilibration, the system temperature becomes $k_B T = 1.003$. Figure (9.13) shows the corresponding $g(r)$. As you see, the sharp peaks have survived, which means the solid structure has been preserved. The peak loca-

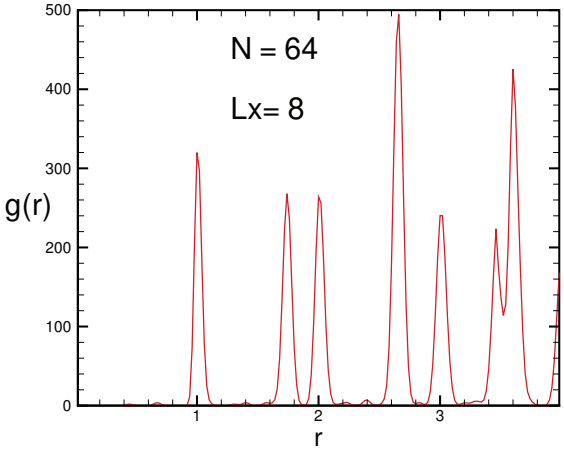

Figure 9.12: *Radial distribution function (RDF) for a zero temperature LJ fluid at* $\rho = 1.156$.

tions are displaced a bit as a consequence of density increment. In figure (9.14) we show $g(r)$ for three different densities. Note that the higher the density, the sharper the peaks. When the density is decreased, the sharpness of peaks diminishes. This signifies that the system is approaching a liquid phase. The first computational result for the structure factor of a three-dimensional Lennard-Jones system was carried out by Rahman (Rahman, 1964) in 1964.

9.9 Dynamical properties

9.9.1 Mean-squared displacement

In this section, we focus on the dynamical aspects of an interacting system in the framework of molecular dynamics. We also discuss how the transport of particles in a system near equilibrium is related to its equilibrium properties. One important dynamical quantity which is related to the transport of particles is the average distance $\langle r(t) \rangle$ of a particle from itself after a given time t. Alternatively, the mean-squared displacement $\langle (\Delta \mathbf{r})^2 \rangle = \langle |\mathbf{r}(t) - \mathbf{r}(0)|^2 \rangle$ gives the average square of

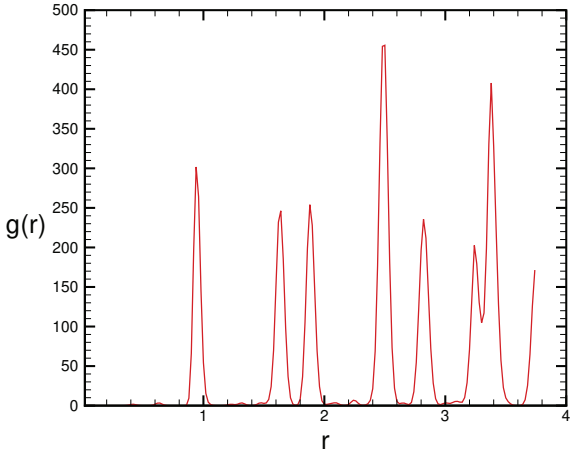

Figure 9.13: *Radial distribution function for a denser LJ fluid at* $\rho = 1.314$ *and* $k_B T = 1$.

this distance. Here 0 symbolizes the time origin and t refers to the time amount elapsed after the time origin. In fluid phases, the mean-squared displacement (MSD) shows a scaling behaviour $\langle(\Delta \mathbf{r})^2\rangle = Ct^\alpha$ at sufficiently large times. If $\alpha = \frac{1}{2}$ we say that the particles exhibit a diffusive behaviour. For $\alpha > \frac{1}{2}$ we have a super-diffusive behaviour whereas $\alpha < \frac{1}{2}$ is termed sub-diffusive behaviour. The value of the scaling exponent depends on dimensionality, density, nature of the interactions, etc. When we have normal diffusion ($\alpha = \frac{1}{2}$) the MSD will be:

$$\langle(\Delta \mathbf{r})^2\rangle = 2dDt \tag{9.33}$$

where d is the space dimension. The coefficient D is known as diffusion or self-diffusion constant. In a solid phase, the MSD asymptotically approaches a constant value at large times and there will be no diffusion. Instead, the particles perform a random fluctuation about their equilibrium position. To compute the MSD, we save the position of a particle at regular time intervals in a file after equilibrium is established. The mean-squared displacement is computed by the following

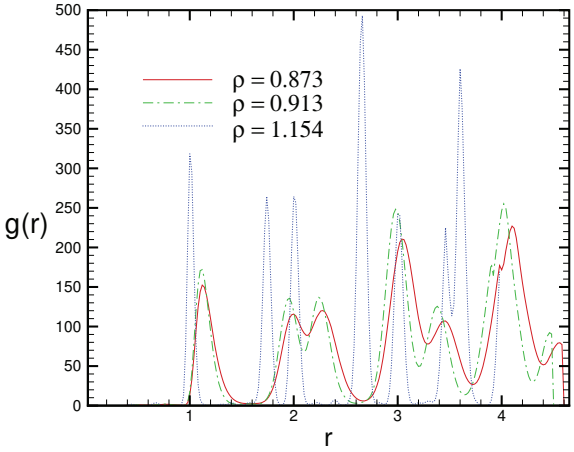

Figure 9.14: *Radial distribution function for a two-dimensional Lennard-Jones system for various densities.*

double summation over particles and their trajectories:

$$MSD[t] = \frac{1}{(T - T_{eq})N} \sum_{j=1,n=T_{eq}}^{N,T-t} [(X_j[t+n] - X_j[n])^2 +$$

$$(Y_j[t+n] - Y_j[n])^2 + (Z_j[t+n] - Z_j[n])^2] \tag{9.34}$$

where T is the number of timesteps, and T_{eq} is the number of timesteps until the system reaches equilibrium. $X_j[m]$ shows the x component of particle j at timestep m etc. The subroutine `MSD` (see Appendix 9.D for details) evaluates the MSD of particles in an MD programme. Figure (9.15) exhibits the trajectory of a tagged particle (here 7th particle) in a two-dimensional LJ system, consisting of 256 particles, at $\rho = 0.819$ and $k_BT = 1.06$. The trajectory qualitatively looks like a Brownian motion. Note the trajectory is not restricted to lying in the simulation box. It belongs to the main particle's motion and its periodic images in other cells. The length of simulation box is $L_x = 20$ and $L_y = \frac{\sqrt{3}}{2} L_x$. Figure (9.16) shows $\langle(\Delta \mathbf{r})^2\rangle$ versus time for a 2D LJ system having $N = 256$ atoms. In liquid or gaseous phases,

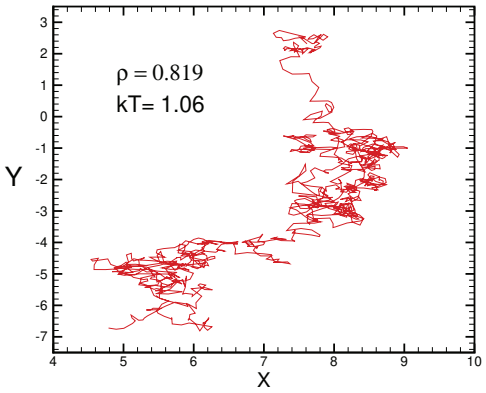

Figure 9.15: *Trajectory of particle 7 in a LJ system of* $\rho = 0.819$ *and* $k_BT = 1.06$ *from the time interval* $[8000, 20000]$.

$\langle(\Delta\mathbf{r})^2\rangle$ is unbounded. On the other hand, in a solid phase, it reaches an asymptotic value. As you see, in the solid phase ($\rho = 1.15$ and $k_BT = 0.50$), the MSD becomes saturated in time, whereas in a fluid-like structure corresponding to $\rho = 0.91$ and $k_BT = 0.52$ the MSD is notably increased with time. In the gaseous phase corresponding to $\rho = 0.73$, and $k_BT = 0.63$ the MSD increases linearly with time for large times. A linear curve was fitted to the linear part of the MSD curve to find the diffusion coefficient D. Figure (9.17) shows the dependence of D on ρ for two values of initial temperature. D is larger for a higher initial temperature.

9.9.2 velocity autocorrelation

Velocity autocorrelation $C(t)$ is another important dynamical quantity. Suppose a particle has velocity $\mathbf{v}$ at an instant which we choose it to be as the time origin $t = 0$. Without a net force on this particle, its velocity would remain constant. However, its interactions with other particles in the system will change its velocity, and we expect that after a sufficient time t, its velocity will not be strongly correlated with its velocity at the initial time $t = 0$. To measure the degree of correlation, we define the so-called *velocity auto correlation* $C(t)$

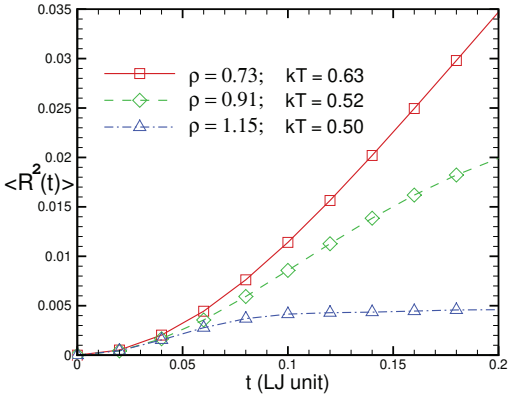

Figure 9.16: *Mean-squared displacement versus time for a 2D LJ gas at three distinctive densities.*

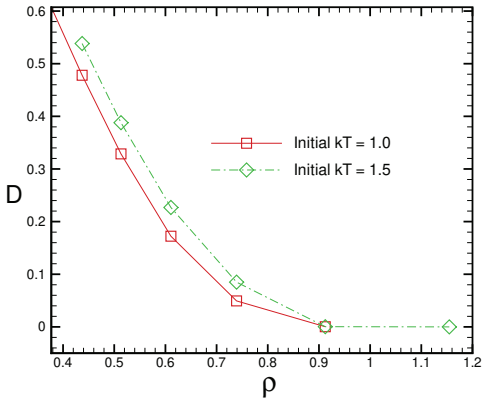

Figure 9.17: *Dependence of the diffusion constant D on the density ρ for a 2D LJ system.*

as the degree of correlation between velocities of a given particle at the initial time and after time t. This dimensionless quantity can be defined in principle for each particle of the system as follows:

$$C_j(t) = \frac{1}{v_j^2(0)} \langle \mathbf{v}_j(t).\mathbf{v}_j(0) \rangle \tag{9.35}$$

where $v_j^2(0) = \langle \mathbf{v}_j(0).\mathbf{v}_j(0) \rangle$. The average is performed over all time origins $t = 0$ after the system becomes equilibrated in the same manner as we did for mean-squared displacement. From the equipartition theorem, we know that if the fluid temperature is T then we have $v_j^2(0) = kTd/m$ where d is the space dimension. Due to normalisation condition in (9.35) we have $C_j(0) = 1$. Despite $C_j(t)$ should theoretically be the same for all particles, numerically, we would obtain more reliable results if we define $C(t)$ as a particle average over $C_j(t)$:

$$C(t) = \frac{1}{N} \sum_{j=1}^{N} C_j(t) \tag{9.36}$$

Note that $C(0) = 1$. A more concise definition of velocity autocorrelation is the following:

$$C_j(t) \propto \langle \mathbf{v}_j(t).\mathbf{v}_j(0) \rangle - \langle \mathbf{v}_j(t) \rangle . \langle \mathbf{v}_j(0) \rangle \tag{9.37}$$

When there is no drive in the system we have $\langle \mathbf{v}_j(t) \rangle = \langle \mathbf{v}_j(0) \rangle = 0$ therefore (9.37) reduces to (9.35). For large time t, we physically expect $\mathbf{v}_j(t)$ to be uncorrelated with $\mathbf{v}_j(0)$, and hence $C(t) \to 0$ for $t \to \infty$. It can be shown (Haile, 1992) that the self-diffusion coefficient defined by (9.33) can be related to the integral of $C(t)$:

$$D = v_0^2 \int_0^\infty C(t) dt. \tag{9.38}$$

Diffusion constant D is an important transport coefficient. Later, we will show that other transport coefficients, such as the shear viscosity and the thermal conductivity, can also be expressed as an integral over appropriate autocorrelation functions. This kind of approach to transport coefficients is generally discussed in the framework of *Green-Kubo* formalism (Allen and Tildesley, 1986; Frenkel and Smit, 2002; Rapaport, 1995; Reif, 1965; Reichel, 1998). The subroutine `VAC` (see Appendix 9.E for details) computes the velocity autocorrelation for an

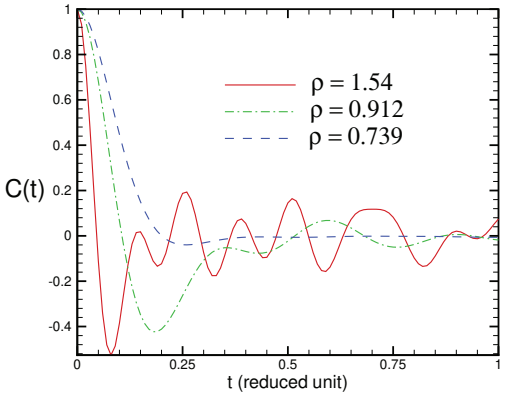

Figure 9.18: *Velocity autocorrelation function for various densities of a two-dimensional LJ system. The initial temperature was* $k_BT = 1$.

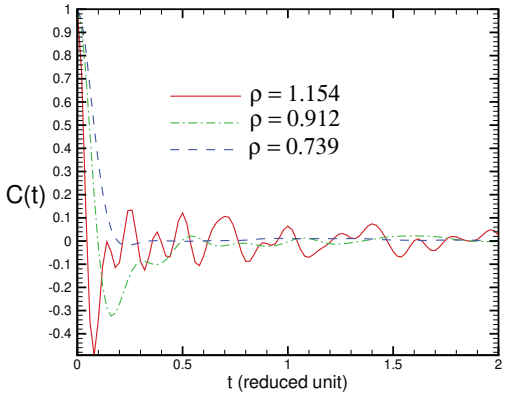

Figure 9.19: *Velocity autocorrelation function for the same parameters in figure (9.18). The initial temperature has been* $k_BT = 2$ *in all three cases.*

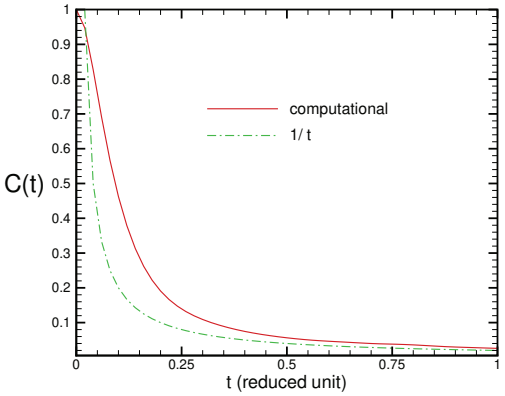

Figure 9.20: *Velocity autocorrelation function at $\rho = 0.56$, for a two dimensional LJ system. The particle number is $N = 265$, and the initial temperature was $k_BT = 5$.*

LJ system of particles. Figure (9.18) shows $C(t)$ versus t for a two-dimensional LJ system of particles at three different densities. In each case, the initial temperature has been set to $k_BT = 1$ in LJ reduced units. For large densities, $C(t)$ becomes negative and passes through a minimum, for short time scales. Note that we have performed averaging over all particles. Next, we increase the initial temperature up to $k_BT = 2$ to ensure the particles deviate large enough from their solid equilibrium positions. Figure (9.19) shows the same diagram as in figure (9.18). Clearly, $C(t)$ reaches zero sooner than in the previous case. The reason is that for fixed density, the system temperature has risen, and the atoms possess higher kinetic energy, which results in a more irregular behaviour. This leads to a sooner forgetting of the velocity memory. Let us now increase the particles number to $N = 256$. Moreover, we set $L_x = 23$ which gives $\rho = 0.56 \sim 0.5\rho_{max}$. Figure (9.20) compares the computed $C(t)$ with $C(t) \sim \frac{1}{t}$. At long time, $C(t)$ decreases like t^{-1}. Figure (9.21) shows $C(t)$ for a 2D solid state of a LJ gas, both with the initial temperature $k_BT = 0.1$. The number of particles is $N = 256$. We observe that in this solid-like system, the velocity correlation persists over a relatively long time. The fluctuations in autocorrelation are enhanced compared to fluid-like structures.

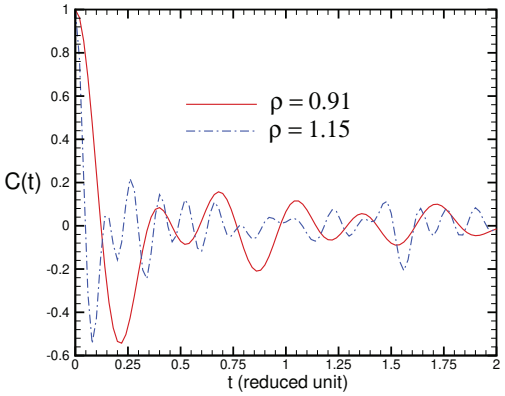

Figure 9.21: *Velocity autocorrelation functions at* $\rho = 1.15, k_BT = 0.051$ *and* $\rho = 0.91, k_BT = 0.049$. $N = 256$ *with initial temperature* $k_BT = 0.1$.

9.10 Problems

Problem 9.1 Verify the energy conservation for the simulated two-dimensional Lennard-Jones system in the text with $N = 64$ particles. Theoretically, the total energy should be conserved. In the simulation, it may not be due to round-off and systematic errors such as truncation.

(a) Plot the time series of the total energy per particle for the interval $[0, t^* = 10]$ and fit a linear curve to it. Take $\tau = 0.01$.

(b) Draw the curve slope as a function of timestep τ to see how well the Verlet algorithm preserves the energy as a function of τ.

(c) Test the accuracy of the Verlet algorithm in preserving the total linear momentum $P(t)$. Draw $|P(t)|$ versus time. Theoretically, the total linear momentum should remain zero if you have initially prepared the system with zero total momentum.

(d) Obtain the angular momentum time series. Prepare the system such that initially, its total angular momentum is zero.

Problem 9.2 Solve problem one by substituting the Verlet algorithm with the fourth-order Runge-Kutta. Which algorithm shows a better performance in energy conservation? Verify your claim for larger N.

Problem 9.3 In the velocity distribution discussed in the text, keep the number of particles constant but vary the simulation box size and obtain the velocity component distribution for each size. Do you see any dependence of the distribution on the system density? Is your observation in agreement with the theory?

Problem 9.4 Obtain the velocity magnitude distribution for the three-dimensional LJ system discussed in the text in figure (9.6) and compare it to the theoretical Maxwell-Boltzmann distribution. Discuss various initial conditions with the same initial kinetic energy.

Problem 9.5 Compute the potential energy per particle for a 2D LJ solid of density ρ in two cases of triangular and square lattices. Remind you that the square lattice side length L should be $L = \sqrt{L_x L_y} = (\frac{\sqrt{3}}{2})^{1/2} L_x$ if both lattices have identical densities. Which lattice has lower energy?

Problem 9.6 Verify that triangular structure is the energy minimising structure for a two-dimensional Lennard-Jones system below melting temperature by initially placing immobile particles on ordered arrays such as honeycomb lattice or in a disordered arrangement.

Problem 9.7 Find the radial distribution function for a three-dimensional Lennard-Jones system at various densities.

Problem 9.8 Find the temporal dependence of mean-squared displacement for a three-dimensional Lennard-Jones system at various densities.

Problem 9.9 Find the temporal dependence of velocity autocorrelation for a three-dimensional LJ system for various densities at $k_B T = 1$. To have a system at the desired temperature $k_B T = 1$ you should appropriately adjust the initial temperature.

Problem 9.10 Verify the Green-Kubo relation for the self-diffusion coefficient D for a three-dimensional LJ system. Obtain D directly by numerical evaluation of the integral and compare it to the long-time slope of the mean squared displacement.

Problem 9.11 Obtain the density dependence of diffusion constant D for a three-dimensional LJ system at $T = 1$.

Problem 9.12 **Melting transition**: If we sufficiently heat a solid body, it will eventually melt. Although we are familiar with this ubiquitous phase transition, its theoretical explanation turns out to be a difficult task in physics. Molecular dynamics provides a framework to tackle this problem computationally. However, as you will see, computer simulation of melting transition is not straightforward. We can naively simulate the process of heating by increasing the particles' kinetic energies over time. The problem with this method is that the system pressure will vary, which is not consistent with what we normally observe in reality. In most melting transitions, the pressure remains constant. One way that you can simulate the melting transition is the following. Initially, prepare a system in a solid phase at the desired pressure. At certain regular times, scale the velocities of all particles such that the system's kinetic energy is slightly increased by a factor $\lambda > 1$. Then perform a simulation until the system reaches an equilibrium state. Measure the temperature and pressure and then adjust the system size appropriately such that the pressure returns to its desired value. This part should be done in a loop. Note that the temperature has changed. In the next stage, examine if the system has met one of the melting criteria. If these criteria are not met, then repeat the procedure. Apply the above method to find the melting transition of a three-dimensional Argon gas and compare your results with the experimental ones.

Problem 9.13 Another way of simulating melting transition is to decrease the system density. Start the system with a solid status. Gradually decrease the density by increasing the simulation box size: $L_x = \lambda L_x$ etc. The scaling parameter λ should be slightly above one. We suggest $\lambda = 1.01$. After each step of rescaling, scale the particle coordinates as well $X[i] \to \lambda X[i]$, and let the system reach equilibrium. Next, to keep the system pressure constant, you should pump energy into the system by appropriately changing the particles' energies. This part should be done in a loop. Follow the rest of the problem in a similar manner to the previous problem.

Chapter 10

Stochastic processes

10.1 Randomness and determinism

The problems we have considered up to now have been deterministic in nature except quantum mechanics which was discussed in chapter eight. However, quantum mechanics is out of the classical physics realm. Here we intend to introduce some problems in classical physics in which randomness is of prominent importance. Most deterministic problems we have encountered so far are associated with an equation, mostly differential, that has to be solved given the prescribed initial condition. These problems are classified as deterministic because the system status is predictable in any future time at least in principle. A distinguished class of problems involves some degrees of randomness which play a significant role in their character. These stochastic systems normally consist of a very large number of constituents and therefore have a huge degree of freedom. Randomness can thus arise in various aspects for example the inability to solve the full equations of motion, lack of identifying the initial or boundary conditions, determination of the system number of particles, etc. Despite, the underlying rules which govern the physics of the system being deterministic, our insufficient knowledge of the system's constituents forces us to replace *statistical* approach for the description of the system's macroscopic properties. Another reason that we have to resort to a statistical description is that even if we manage to solve the equations of motion for every system constituent, meaningful information would be possi-

ble only if we average over the constituents' behaviour. In practice, we model a *real* but non-treatable deterministic process by a stochastic one that is close, if not identical, to the real system in average properties. To find a deeper insight into the philosophy of the statistical description I can recommend to you the valuable textbook (Reif, 1965). Let us leave this qualitative discussion and prepare ourselves to deal with some stochastic problems in physics. We shall start with simple problems which illustrate the success of a statistical approach. To each stochastic process, we associate a random variable X. A typical realisation of X is denoted by x. The range of x depends on the character of the problem. Principally, we divide the random processes into two classes discrete and continuous. A prototype example of a discrete random process is throwing a die. If we could correctly write down the Newton equation of motion for our rigid body (the dice) we could deterministically predict the outcome. This task seems improbable therefore and we should inevitably grasp a stochastic description. The outcome of this seemingly random process can be either one of the six faces which are realisations of the random processes. We can assign a real-valued number to each realisation to quantify the average properties of the process. The natural assigning numbers are those shown by the faces that are x_i $i = 1, \cdots, 6$. It is now possible to evaluate the mean value of the process. For this purpose, we multiply x_i by the occurrence probability of ith outcome p_i and sum over the total number of outcomes: $\langle X \rangle = \sum_{i=1}^{6} x_i p_i$. For a normal dice, all the outcomes are equally probable therefore $p_i = 1/6$ and we have: $\langle X \rangle = 1/6 \sum_{i=1}^{6} x_i = 21/6 = 3.5$. For a general discrete stochastic process with M outcomes, the ith outcome occurs with probability p_i. Showing the realisation of the ith outcome by x_i, the average of the random process X will be:

$$\langle X \rangle = \sum_{i=1}^{M} x_i p_i \tag{10.1}$$

A prototype of a random process of continuous nature could be the time difference (time headway) between the passing of two consecutive cars at a fixed location on a road. Another example is the velocity magnitude of an emitted electron from a metal surface in the photoelectric experiment.

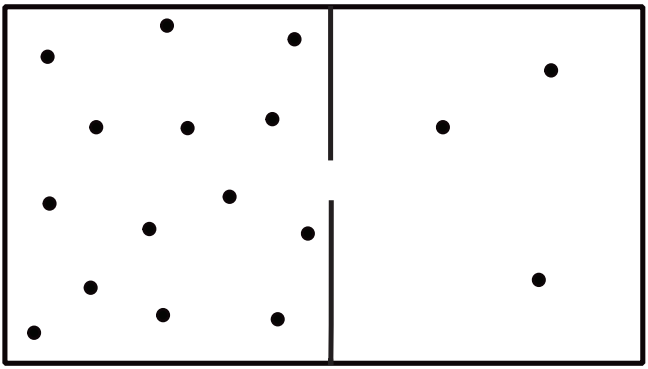

Figure 10.1: *A closed box is divided into two equal parts by a fixed partition. A small hole is opened in the partition such that one particle can pass through the hole per time unit.*

10.2 Particles in box

A remarkable point in dealing with systems having many particles is that in the long-time limit, many of them evolve to an equilibrium state. We shall illustrate this point by considering an interesting toy model. The model was put forward in (H. Gould and Chriastian, 2006) and many of the concepts introduced in this model can simply be generalised to other problems. Consider a closed box that is divided into two segments of equal volumes (see Fig. (10.1)). The left segment contains a gas comprising of N identical particles and the right segment is initially empty. Imagine we make a small hole in the partition between the two halves such that one particle can pass through the hole per time unit. What happens from a statistical point of view? Our experience or intuition tells us that after some time, the average number of particles in each half of the box will become $N/2$, and we say that the system has reached equilibrium. Let us simulate this process. For this purpose, we shall model the problem by an equivalent model. We make a simplifying assumption: the probability per time unit that a particle goes through the hole in the partition is the same for all N particles regardless of the particle number in either half. Furthermore, we assume that the hole size is such that only one particle can pass through at a time. This time is considered to be the timestep of our problem. We show the

number of particles in the left segment at timestep t by $n(t)$. Hence, the number of particles in the right segment will be $N - n(t)$ correspondingly. To model the motion of particles let us propose a simple dynamical rule for updating $n(t)$. At each timestep, one segment is chosen at random and one of its particles is removed to the other segment. To randomly select the halves and a particle, we need a computational tool. This tool is *random number generator*. Almost all the computational softwares possess such a tool and can give you a random number $0 < r < 1$, upon calling, which is uniformly distributed in the unit interval $[0, 1]$. You can consult other references (H. Gould and Chriastian, 2006; W. H. Press and Flannery, 2002) to see how *random number generator* algorithms work in detail. In this book, we only make use of this valuable tool. Each time you call this generator, a random number r is returned. Once you are equipped with this valuable tool you can generate any type of random variable with an arbitrary probability distribution function. We will later come back to this point. For the present, we notify that using a random number generator we can generate discrete random variables. For example in our present problem, we can choose the segments by an `if` command as follows: first, generate a random number r between zero and one. If r is less than 0.5 then choose the left segment otherwise if it is equal to or greater than 0.5 then choose the right segment. If you want to randomly choose one of the particles in the left segment, proceed as follows: suppose there are $M > 0$ particles in the left segment. Generate a random number $0 < r < 1$. We know that $0 < rM < M$. This means that the integer part of rM i.e. $[rM]$ lies between zero and $M - 1$. By adding one unit to $[rM]$ it turns out that $i = 1 + [rM]$ (number of the chosen particle) would be an integer number between one and M that is $i = 1, \cdots, M$. We are now ready to simulate this toy model. Initially set an arbitrary number of particles $N_0 > 0$ in the left segment. Let us take the extreme value $N_0 = N$. Make a loop over time steps. At each timestep t choose a segment at random. Then randomly choose a particle from the present ones in the selected segment. Next, reduce (add) the number of particles in the selected (other) segment by one unit. The programme `Box` (see appendix 10.A for details) simulates our particle in a box toy model. Figure (10.2) shows the time evolution of n (number of particles in the left side box) for $N = 8, 16$ and 64 respectively. In figure (10.3) we show the time evolution of n for larger values $N = 400, 800, 3600$. You see that if

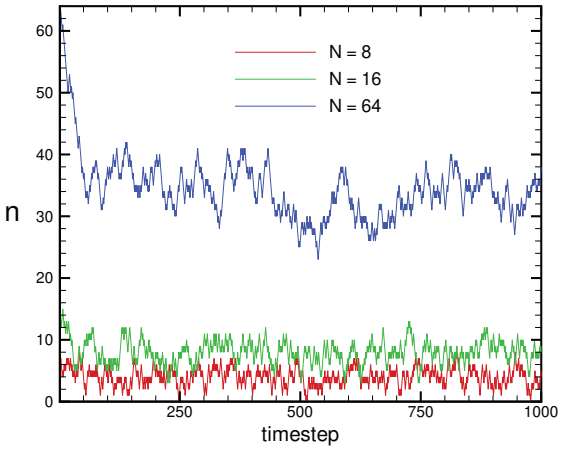

Figure 10.2: *Time evolution of particle number n in the left box for total number of particles $N = 8, 16, 64$.*

we wait for enough the system reaches an equilibrium state. A simple criterion for reaching equilibrium is that the time series $n(t)$ does not show a trend and fluctuates around a constant mean value. In general, the equilibration time increases with N. A better way of quantifying equilibrium is to measure the fluctuations around the mean value of $\langle n \rangle$. The mean-squared of fluctuations Δn is defined as:

$$(\Delta n)^2 = \langle [n - \langle n \rangle]^2 \rangle = \langle n^2 \rangle - \langle n \rangle^2 \tag{10.2}$$

The bracket denotes the average over simulation time. It would be more meaningful to consider relative fluctuation $\Delta_{rel} = \frac{\Delta n}{\langle n \rangle}$. Notice that Δn is the standard deviation from the mean number of particles in the left side box. Note that $\langle n \rangle$ equals $\frac{N}{2}$. The evaluated fractional deviations from the mean are $0.088, 0.051, 0.033, 0.008$, and 0.005 for $N = 8, 16, 64, 400, 800$ and 3600 respectively. You see that the relative fluctuation of n appears to be a decreasing function of the total number of particles N. This behaviour is generic in statistical physics. It would be instructive to derive a time dependence for $\langle n \rangle$ to show explicitly how chance can generate deterministic behavior. During a time step Δt the average number of particles in the left segment,$\langle n \rangle$,

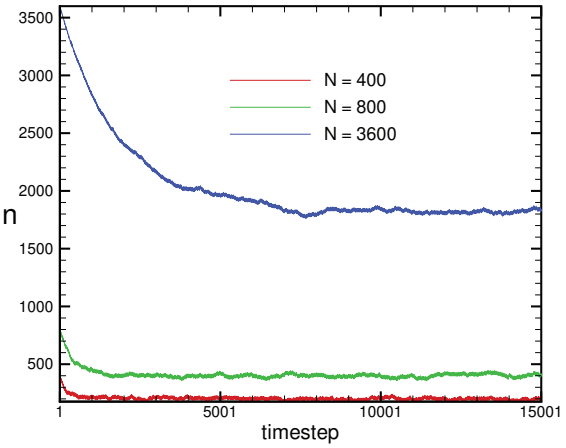

Figure 10.3: *Time evolution of n for $N = 400, 800, 3600$ particles.*

undergoes a change $\delta\langle n\rangle$. This change is proportional to Δt. A negative contribution comes from moving a particle from the left half into the right half. This term is proportional to the probability of finding a particle in the left half i.e. $\frac{\langle n\rangle(t)}{N}$. The other positive contribution is associated with the movement of a particle from the right segment into the left one. This term is proportional to the probability of finding a particle in the left segment which is $\frac{N-\langle n\rangle(t)}{N}$. Therefore we can write the following equation:

$$\delta\langle n\rangle = [-\frac{\langle n\rangle(t)}{N} + \frac{N - \langle n\rangle(t)}{N}]\Delta t \tag{10.3}$$

Dividing by Δt and taking the limit $\Delta t \to 0$ we arrive at the following differential equation for $\langle n\rangle$:

$$\frac{d\langle n\rangle}{dt} = 1 - 2\frac{\langle n\rangle(t)}{N} \tag{10.4}$$

In figure (10.4) we exhibit the dependence of $\langle n\rangle$ on t from simulation and the following analytical solution of (10.4) for the initial condition $\langle n\rangle(0) = N$:

$$\langle n\rangle(t) = \frac{N}{2}(1 + e^{-2t/N}) \tag{10.5}$$

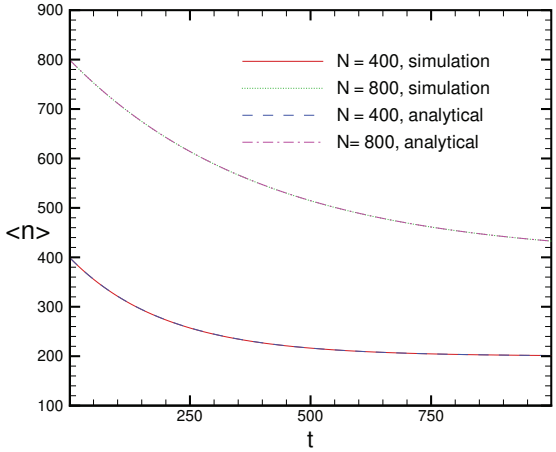

Figure 10.4: *Time dependence of* $\langle n \rangle$ *for various* N.

Note that $\langle n \rangle$ decays exponentially to its equilibrium value $\frac{N}{2}$. There is a very good agreement between simulation and analytical formula. According to (10.5) the relaxation time is $\tau = \frac{N}{2}$ which linearly grows with the particle number N.

10.3 Random walk

We proceed by considering a well-known stochastic problem the so-called *random walk* which is widely used in statistical physics. A random walk is a mathematical formalization of a path that consists of a succession of random steps. For example, the path traced by a molecule as it travels in a liquid or a gas, the search path of a foraging animal, and the price of a fluctuating stock can all be modeled as random walks. The term random walk was first introduced by Karl Pearson in 1905 (Pearson, 1905). Since then random walk and its variants have been extensively used in physics, chemistry, ecology, economics, psychology, computer science, and biology. The random walk can explain the observed behaviors of many processes in these fields, and thus serves as a fundamental model for the recorded stochastic

activity. To have a better insight, we shall pose the problem from a physical point of view. Consider the Brownian motion. Historically this transport phenomenon was observed in 1827 by Robert Brown, a botanist, who looked at the erratic and irregular motions of micron-sized particles found in pollen grains immersed in water through a microscope. The mathematical explanation of the Brownian motion came less than a century later when Albert Einstein published a paper in 1905 that explained, in precise detail, how the motion that Brown had observed was a result of the pollen being moved by individual kicks from water molecules. This explanation of the Brownian motion served as definitive confirmation that atoms and molecules actually exist, and was further verified experimentally by Jean Perrin in 1908 who was awarded the Nobel Prize in Physics in 1926 "*for his work on the discontinuous structure of matter*". The directions of the atomic bombardment forces are constantly changing, and at different times the particle is hit more on one side than another, leading to the seemingly random nature of the motion. A year after the pioneering explanation of Einstein, which was more qualitative, a rigorous mathematical theory was put forward by Polish mathematician Smoluchowski (Smoluchowski, 1906). In a Brownian motion, a large particle (Brownian particle) is immersed in a solvent liquid comprised of much smaller molecules. The Brownian particle incessantly and repeatedly receives random kicks from collisions by its adjacent solvent molecules. As a result, it performs an irregular and random motion and follows a complicated zigzag trajectory. This trajectory can be best described by a set of discrete jumps in the Brownian particle positions or what is known as a *random walk*. The jumps' directions are random in nature and there is no preferred direction. The jump length is not constant and can vary from small to large values. For a detailed see (Reif, 1965). To model this random walk, it is easier to assume that time is evolved in discrete steps. Later we will consider other types of a random walk in continuous time. Let us now simulate this discrete random walk. For the sake of simplicity, we consider the motion to be one-dimensional but many of the concepts that we shall introduce and develop can be genralised to higher dimensions straightforwardly.

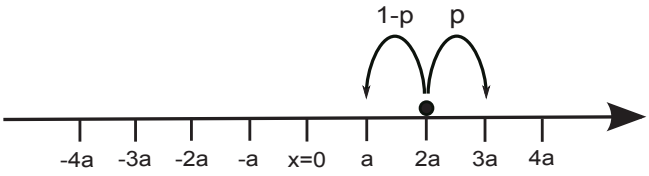

Figure 10.5: *A one-dimensional discrete time random walk on a discrete lattice.*

10.3.1 One-dimensional random walk

Let us now simulate a one-dimensional random walk. We make another simplification and take the step lengths to be a constant shown by a. Due to one-dimensionality of the problem, the ith step can either be towards the right $(+a)$ or left $(-a)$. Suppose the Brownian particle or the walker starts its walk from the origin. The walker position (distance from the origin) after taking N steps is denoted by x_N. The random variable x_N can take $2N+1$ states: $x_N = -Na, (-N+1)a, \cdots, -a, 0, +a, +2a, \cdots, Na$. See figure (10.5) for a schematic illustration. One of the fundamental questions that we wish to address is to find the probability $P(m, N)$ that after N steps the walker arrives at site m. To answer this question, we introduce a random variable s_n which is the n-th step that the walker takes. Random variable s_n can be $\pm a$. We can write $x_n = s_1 + s_2 + \cdots + s_n$ where x_n is the walker distance to the origin after taking n steps. The statistical properties of the random variable x_n are determined by the corresponding properties of random variables s_i. Let us take the probability of making a right step by p and a left step by q. Evidently, we have $p+q=1$. The discrete random variable s_i has the probability distribution function $p_i(x) = p\delta(x-a)+q\delta(x+a)$. The average of the step variable s_i turns out to be $\langle s_i \rangle = \int x p_i(x) dx = p(a) + q(-a) = (p-q)a$ $\sqrt{i} > 0$ which does not depend on i. Note that the direction of the walker step is independent of the preceding steps. This is an example of a memoryless or Markov random process. we can simulate a one-dimensional walk of N steps by flipping a coin N times and increasing the walker position x by step length a each time the coin is head and decreasing it by a each time the coin is tail. On a computer, we replace the coin-flipping with a random number generator. Tossing a coin is modeled by calling the random number generator. If the generated number r is

less than half then we adopt a head otherwise the coin is assumed to be a tail. There is no worry for the particular case $r = 0.5$. Actually, the occurrence probability of $r = 0.5$ is zero in practice! The average of walker position after taking N steps will be $\langle x_N\rangle = \sum_{i=1}^{N}\langle s_i\rangle$. Because the step variables are identical their averages are independent of their numbers thus we have $\langle x_N\rangle = N\langle s\rangle = N(p-q)a$. This type of averaging is called *ensemble* averaging. In practice, you perform the walk a large number of times say M times. The average position $\langle x_N\rangle$ is obtained as follows:

$$\langle x_N\rangle = \frac{1}{M}\sum_{s=1}^{M} x_N^{(s)} \tag{10.6}$$

where $x_N^{(s)}$ is the walker position after N steps in the trial s. In an unbiased random walk where p equals q, we have $\langle x_N\rangle = 0$. Let us proceed a bit analytically before calling a computer to help us. A useful quantity is the mean squared displacement of the walker $\langle(\Delta x_N)^2\rangle = \langle[x_N - \langle x_N\rangle]^2\rangle$ which is defined as the average of the displacement square with respect to the mean position. It turns out that:

$$\langle[x_N - \langle x_N\rangle]^2\rangle = \langle[\sum_{i=1}^{N} s_i - \sum_{i=1}^{N}\langle s_i\rangle]^2\rangle = \langle[\sum_{i=1}^{N}(s_i - \langle s_i\rangle)]^2\rangle \tag{10.7}$$

Denoting $s_i - \langle s_i\rangle$ by Δ_i we have:

$$\langle(\Delta x_N)^2\rangle = \langle[\sum_{i=1}^{N}\Delta_i]^2\rangle = \langle\sum_{i,j=1}^{N}\Delta_i\Delta_j\rangle \tag{10.8}$$

Separating the $i = j$ term we find

$$\langle(\Delta x_N)^2\rangle = \langle\sum_{i=1}^{N}(\Delta_i)^2\rangle + \langle\sum_{i\neq j}^{N}\Delta_i\Delta_j\rangle \tag{10.9}$$

The first term on the right hand side of (10.9) gives $N\langle\Delta^2\rangle = N\langle[s - \langle s\rangle]^2\rangle = N[\langle s^2\rangle - \langle s\rangle^2]$. The mean square of the step variable s is simply found to be $\langle s^2\rangle = p(a^2)+q(a^2) = (p+q)a^2 = a^2$ thus we have: $\langle\Delta^2\rangle = a^2-(p-q)^2a^2 = 4pqa^2$ where use has been made of $p+q = 1$. Therefore the first term on the right-hand side of (10.9) becomes $4pqNa^2$. The

second term of the right-hand side of (10.9) is zero. To verify this you should first note that $\langle \Delta_i \rangle = \langle [s_i - \langle s_i \rangle] \rangle = \langle s_i \rangle - \langle s_i \rangle = 0$ and that due to independence of steps $\langle \Delta_i \Delta_j \rangle = \langle \Delta_i \rangle \langle \Delta_j \rangle$. In conclusion, we arrive at the ultimate result:

$$\langle (\Delta x_N)^2 \rangle = 4pqNa^2 \tag{10.10}$$

Note that for a given N the mean squared displacement (MSD) is maximum for $p = q = 0.5$. The basic feature is that MSD is proportional to the number of steps N. The root mean-squared (rms) of the displacement i.e.; $\sqrt{\langle (\Delta x_N)^2 \rangle}$ is thus proportional to $\sqrt{N}$. If the timestep between steps is shown by Δt, after N steps the time has elapsed for $t = N\Delta t$. Recalling the diffusion equation from chapter 5, in terms of the time we have:

$$\langle (\Delta x_N)^2 \rangle = 2Dt \tag{10.11}$$

where the diffusion constant D turns out to be $D = \frac{2pq}{\Delta t} a^2$. Before simulating the fascinating problem of random walk let us complete our discussion by writing $P(m, N)$, which is the probability of finding the walker at site m after N steps. Clearly $P(m, N) = 0$ for $|m| > N$. If $|m| \leq N$ we have:

$$P(m, N) = \frac{N!}{[\frac{N+m}{2}]![\frac{N-m}{2}]!} p^{\frac{N+m}{2}} q^{\frac{N-m}{2}} \tag{10.12}$$

For its derivation see chapter one of the excellent monograph (Reif, 1965). Let us now simulate our one-dimensional random walk and see to what extent the simulation results are in agreement with analytical ones. To obtain any average numerically we have to sample an ensemble of N step walks and average over the ensemble members (trials) to obtain meaningful averages. Each N-step walk is called a *trial* or *sample*. The more the number of trials the less the systematic error and the more reliable our results. The programme `RandomWalk` (see Appendix 10.A for details) simulates the motion of a one-dimensional random walker. Let us explore some general characteristics of a normal random walk. In figure (10.6) we show a space-time plot of a random walker with $N = 500$ steps and $p = 0.5$. This plot is called random walk *profile*. Profiles of trials are different from each other. The random walk of each trial begins from the origin. The step length and timestep are both set to unity. The computed

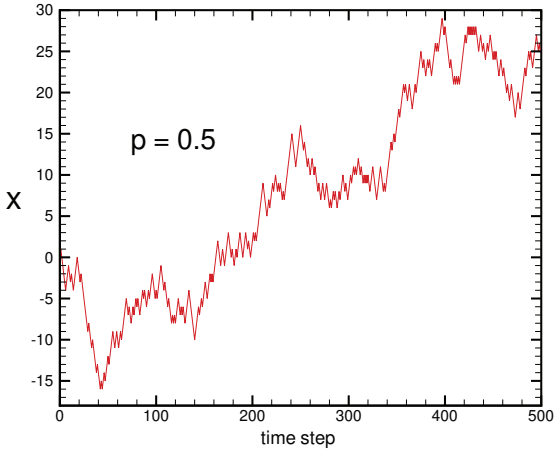

Figure 10.6: *Space-time plot for a typical random walk trial with $p = 0.5$ and $N = 500$ steps. In this trial, the walker has arrived at the site $m = 25$ after $N = 500$ steps.*

value of the ensemble average $\langle x_N \rangle$ approaches $(p - q)N$ if the number of trials, M, goes to infinity. For example for $N = 100$ we obtain $\langle x_N \rangle = -0.42, -0.06, 0.0006$ for runs with $M = 100, 10000$ and 10^6 trials respectively. In figure (10.7), the mean-squared displacement $\langle (\Delta x)^2 \rangle$ is depicted versus the number of timestep number both computationally and analytically from relation (10.10) where N is the number of steps and $q = 1-p$ denotes the probability to step backward. The agreement between the simulation and the analytical formula is excellent.

10.3.2 Higher dimensional random walk

Many of the features that were discussed in the context of a one-dimensional random walk can be generalised into higher dimensions straightforwardly. Let us discuss a two-dimensional random walk on a two-dimensional grid of points which constitutes a square lattice. See figure (10.8) for illustration. The probabilities of taking a step to right, left, up, and down are shown by p_r, p_l, p_u, and p_d respectively.

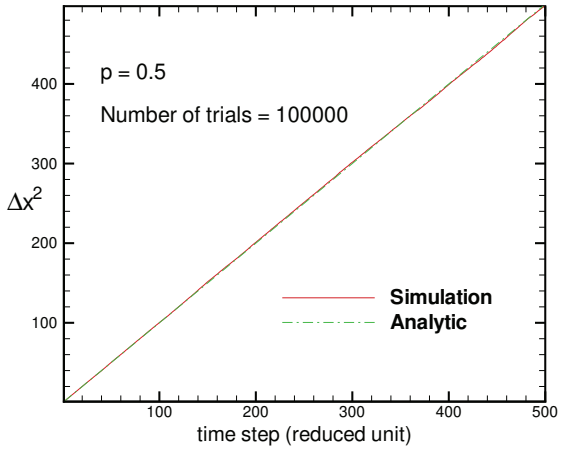

Figure 10.7: *Mean-squared displacement of a random walker for $p = 0.5$. The number of trials has been $M = 10^5$.*

In the symmetric case, we have $p_r = p_l = p_u = p_d = 0.25$. To simulate a trial walk we generate a random number $0 < r < 1$. If $0 < r < 0.25$ we step to right, if $0.25 < r < 0.50$ we step to left, if $0.50 < r < 0.75$ we step upwards, and finally if $0.75 < r < 1.0$ we step downwards. In figure (10.9) we have depicted the trajectory of a trial walk of $N = 1000$ steps starting from the origin. As we explained earlier each trial represents the trajectory of a walker. Suppose we plot the trajectories of M trials in the same graph. Figures (10.10) and (10.11) exhibit the trajectories of a collection of $M = 10000$ walkers each taking $N = 500$ steps (figure (10.10)) and $N = 1000$ (figure (10.11)) respectively. If each walker represents a bee, the qualitative nature of the trajectories' shape resembles the swarm of bees. As you can see, the surface of the swarm looks like a circle with a jagged surface. The values of the mean-squared displacement written inside figures have been obtained by computation over an ensemble of $M = 10000$ walkers. In two dimensions we can define $\langle y_N \rangle$ and $\langle (\Delta y_N)^2 \rangle$ analogous to $\langle x_N \rangle$ and $\langle (\Delta x_N)^2 \rangle$. It turns out that $\langle x_N \rangle = N(p_r - p_l), \langle y_N \rangle = N(p_u - p_d)$. Similarly, we have $x_N = \sum_{i=1}^{N} s_{x,i}$ and $y_N = \sum_{i=1}^{N} s_{y,i}$. Each component of the two dimensional step vector $\mathbf{s} = (s_x, s_y)$ is a random

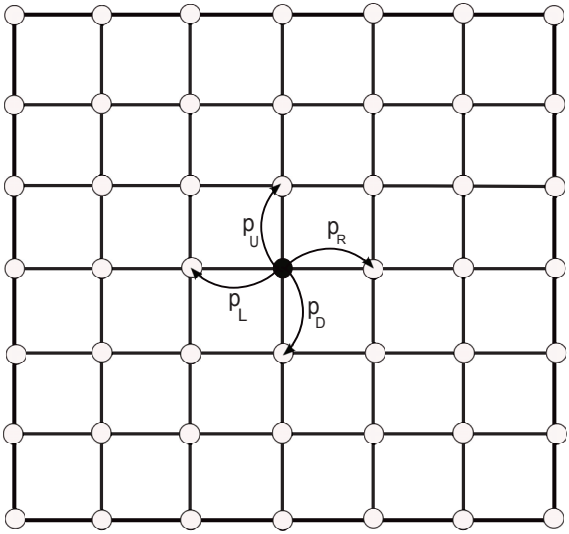

Figure 10.8: *Two-dimensional random walk on a square lattice.*

variable with a binary distribution: $P(s_x) = p_r\delta(s_x - a) + p_l\delta(s_x + a)$; $P(s_y) = p_u\delta(s_y - b) + p_d\delta(s_y + b)$ where a and b are lattice constants in x and y directions. We simply find that $\langle(\Delta_x)^2\rangle = \langle[s_x - \langle s_x\rangle]^2\rangle = \langle s_x^2\rangle - \langle s_x\rangle^2$. Noting that $\langle s_x^2\rangle = (p_r + p_l)a^2$. We find:

$$\langle(\Delta_x)^2\rangle = [(p_r + p_l) - (p_r - p_l)^2]a^2 \tag{10.13}$$

Similarly, we find the mean-squared displacement in y direction:

$$\langle(\Delta_y)^2\rangle = [(p_u + p_d) - (p_u - p_d)^2]b^2 \tag{10.14}$$

For $a = b$ the mean-squared displacement $\langle(\Delta\mathbf{r})^2\rangle = \langle(\Delta x_N)^2\rangle + \langle(\Delta y_N)^2\rangle$ becomes:

$$\langle(\Delta\mathbf{r})^2\rangle = N[1 - (p_r - p_l)^2 - (p_u - p_d)^2]a^2 \tag{10.15}$$

Figure (10.12) shows ensemble averages $\langle x\rangle, \langle y\rangle, \langle\Delta x^2\rangle, \langle\Delta y^2\rangle$ as well as $\langle\Delta\mathbf{r}^2\rangle = \langle\Delta x^2\rangle + \langle\Delta y^2\rangle$ dependence on step number N. Due to symmetry in x and y directions we have $\langle x\rangle = \langle y\rangle$ and $\langle(\Delta x^2)\rangle = \langle(\Delta y^2)\rangle$. Figure (10.12) exhibits this symmetry on a computational

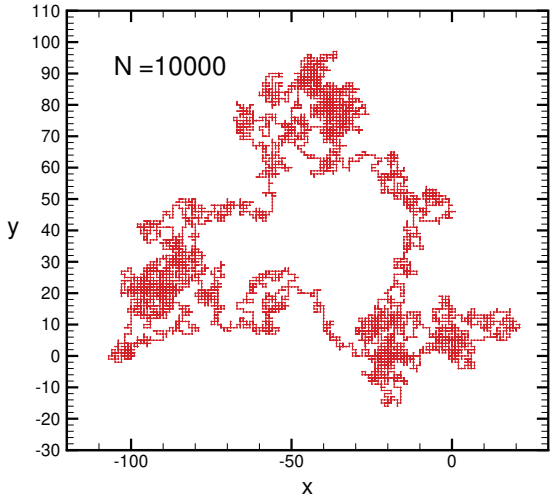

Figure 10.9: *Trajectory of a 2D symmetric random walker which has started its walk from the origin.*

ground. In general, the square root of the mean squared displacement (rms) attains a scaling form for large N:

$$rms(N) = \sqrt{\langle(\Delta \mathbf{r})^2\rangle} \sim N^{\nu} \quad (N >> 1) \tag{10.16}$$

By computing the averages over an ensemble of $M = 100000$ trials we obtain the values $\langle(\Delta \mathbf{r})^2\rangle = 0.42, 0.59, 074, 0.90$ for $N = 8, 16, 32$ and 64 respectively. n a logarithmic scale, we have:

$$rms(N) = N^{\nu} \rightarrow log\ rms = \nu log N \tag{10.17}$$

A linear fitting to the above equation gives: $\nu = 0.58$. Note that the expected value $\nu = 0.5$ for a normal random walk is obtained for large N. In our cases, the discrepancy between the large N value $\nu = 0.5$ and the computed value $\nu = 0.58$ is associated with the smallness of N.

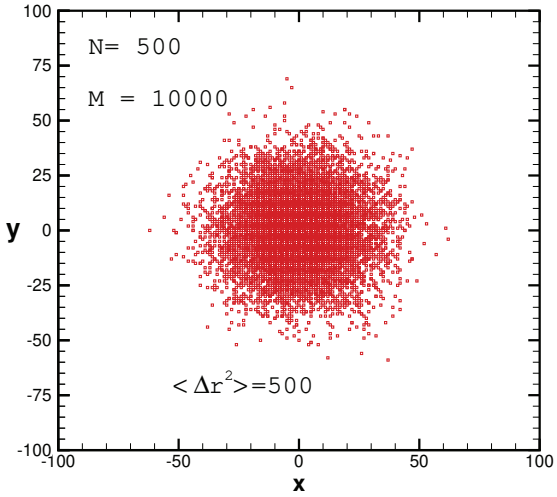

Figure 10.10: *Trajectories of $M = 10000$ random walkers each taking $N = 500$ steps. The walks are symmetric and have started from the origin.*

10.4 Variants of random walk

In this section, we investigate the statistical properties of various kinds of random walk. The problems that we shall consider include persistent random walk, random walk with variable step length, random walk with trap, and self-avoiding random walk. Each of these variants can model a physical process which will be explained. Let us start with a persistent random walk.

10.4.1 Persistent random walk

In a persistent random walk, the walker has a memory of its preceding steps. In other words, the transition or jump probability depends on the previous step. Consider a persistent walk with N steps on a one-dimensional lattice. Each step is made in the same direction as its preceding step with probability α. Correspondingly a step in the opposite direction of the previous step occurs with probability $1 - \alpha$.

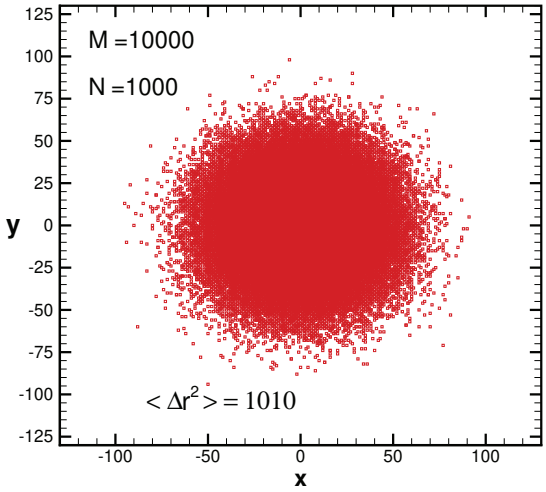

Figure 10.11: *Trajectories of $M = 10000$ random walkers each taking $N = 1000$ steps. The walks are symmetric and have started from the origin.*

When $\alpha = \frac{1}{2}$ the persistent random walk becomes a normal random walk. It would be very easy to write a programme to simulate this type of random walk. Note that it is necessary to specify both the initial position and an initial direction for the walker. The programme `PersistWalk` (see Appendix 10.B for details) simulates the motion of a one- dimensional persistent random walk. Let us first see a typical profile of a persistent walk of $N = 500$ steps in figure (10.13). You observe the distinctive characteristics when α changes. When $\alpha \neq 0.5$ we see that the walker changes its direction quite often. In figure (10.14) we have shown the dependence of $\langle(\Delta x)^2\rangle$ on step number N for $\alpha = 0.5$ and $\alpha = 0.3$. Ensemble averaging has been performed for $M = 100000$ trials. As you can see, when $\alpha = 0.5$, the persistent walk is identical to a normal random walk where the walker loses its short-time memory. However, when α deviates from 0.5 the walk characteristics become substantially different from a normal walk. In particular, the diffusion constant (slope of the MSD curve vs N) reduces in comparison to the normal walk. To gain more insight, let us

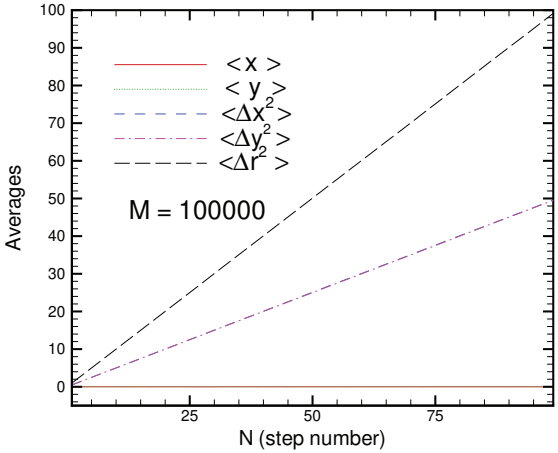

Figure 10.12: *Mean positional components and mean-squared displacements vs step numbers N for a symmetric two-dimensional random walk. Ensemble averages have been performed over $M = 10^4$ trials.*

see how the distribution function $P_N(x)$ behaves for a normal and persistent walks in figure (10.15). In a persistent walk, when α becomes less than 0.5, the distribution function becomes more localized compared to a symmetric normal walk ($\alpha = 0.5$). This is in accordance with the reduction of the diffusion constant.

10.4.2 Random walk with steps of variable lengths

So far we have restricted our random walk to the case where the step-length is constant. Now we want to release this assumption and allow the step-length to vary. For the sake of simplicity, we consider a discrete-time random walk in one dimension. Each walking step is made at timesteps of duration Δt. The probability that the length of a single step is between a and $a + \Delta a$ is $f(a)\Delta a$, where $f(a)$ is the step-length probability density. Consider $f(a)$ in the exponential form $f(a) = Ce^{-a/\lambda}$ for $a > 0$. Normalization condition $\int_0^\infty p(a)da = 1$ implies $C = 1/\lambda$. To simulate this walk of variable step-length we need to generate a step-length at each timestep. These step-length numbers

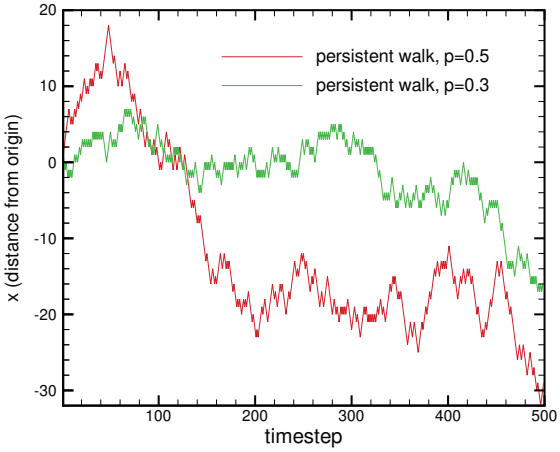

Figure 10.13: *Profile of a persistent random walk with* $N = 500$ *steps for two values of* $\alpha = 0.3$ *and* 0.5.

should have an exponentially decaying distribution. For this purpose, you first need to obtain the accumulative probability distribution function $F(x)$ associated with $f(x)$. For our exponential distribution, we have $F(x) = 1/\lambda \int_{-\infty}^{x} e^{-x/\lambda} dx = 1 - e^{-x/\lambda}$. Next, we generate a random number r uniformly distributed between zero and one. Equating $r = F(x)$ gives us the random number x with the desired exponentially decaying distribution function. Note that we have to obtain the inverse function $F^{-1}(x)$. For the exponential function this turns out to be $F^{-1}(x) = -\lambda ln(1-x)$ and we conclude that the variable $x = F^{-1}(r) = -\lambda ln(1-r)$ has the required exponential distribution. Now we can express the algorithm. At each timestep i, generate a random variable a_i from the exponential distribution $f(a) = e^{-a}$ (λ is set to one). Then randomly choose a direction sign (minus for left and plus for right) and change the current walker position by $\pm a_i$ depending on the chosen walk direction. The main quantity of interest is $P_N(x)\Delta x$, the probability that the walker displacement is between x and $x + \Delta x$ after N steps. We assume the walk is symmetric i.e., the probability of walking rightward equals the probability of walking leftward. Let us see in figure (10.16) the profile of a typical walk of $N = 100$ steps.

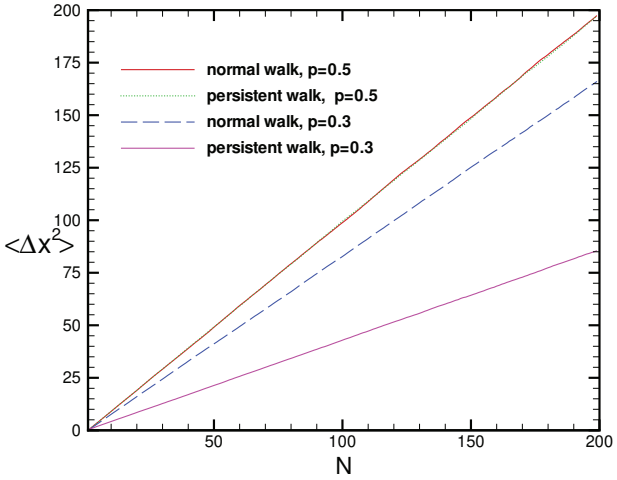

Figure 10.14: *Dependence of the mean squared displacement* $\langle(\Delta x)^2\rangle$ *on* N *for a one dimensional random walk with* $\alpha = 0.3$ *and* 0.5.

You see the profile characteristics differ substantially from each other. In the variable step walk, one encounters long jumps which are absent in the normal random walk. Let us see how $P_N(x)$ behaves in variable and fixed step length walks. Figure (10.17) depicts this behaviour: A drastic difference between distribution functions is observed. In the step variable case, the probability of finding the walker near the origin is notably reduced compared to a normal random walk. It would be interesting to see how the variance of displacement $\langle(\Delta x)^2\rangle$ behaves in the case of step variable random walk. In figure (10.18) we compare this quantity to its normal walk counterpart. Another frequent choice for the step magnitude probability density function $f(a)$ is power law form $f(a) = Ca^{-1-\alpha}$ $a > 1$. This type of random walk is known as a Levy flight for $\alpha \leq 2$. As an example take $\alpha = 1$ which gives $f(a) = \frac{C}{a^2}$. Constant C is determined by the normalisation condition: $C\int_1^\infty a^{-2}da = 1$. It turns out that $C = 1$. Let us simulate this random walk. We use the general inverse method for generating a random variable with power law distribution $f(a) = a^{-2}$. The corresponding cumulative distribution function $F(a) = \int_1^a \frac{1}{a^2}da$ turns out

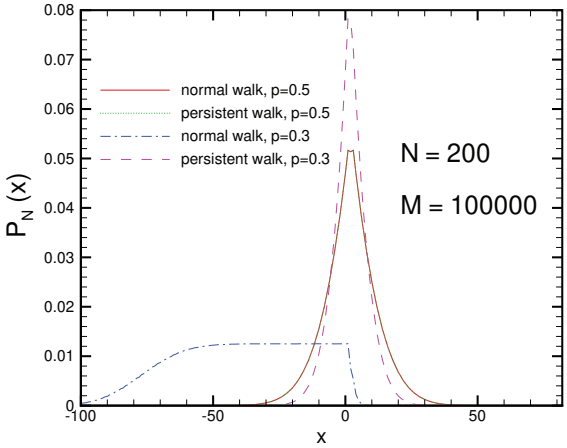

Figure 10.15: *Distribution function $P_N(x)$ for $N = 200$ steps. The persistent walk is compared to a normal walk.*

to be $F(a) = 1 - \frac{1}{a}$. The step length variable a is thus generated as $a = P^{-1}(r) = \frac{1}{1-r}$. Figure (10.19) exhibits $P_N(x)$ for a $N = 50$ step walk with variable step length generated from a long-tail distribution function $p(a) = \frac{C}{a^2}$.

10.4.3 Random walk on a continuum

In one dimension each step can have only two directions: right or left. In higher dimensions the situation is different. For simplicity suppose we are in dimension two and assume that the random walk occurs in a continuum i.e.; the walker can be found at any point of the 2D space. Suppose the magnitude of the step length a can take any value between zero and infinity. Moreover, the step direction can be at any angle θ uniformly distributed between zero and 2π. This random walk was proposed by Rayleigh in 1919 (Weiss, 1994). Here the variable of most interest is R_N, the walker distance from the origin after N steps. Let us first consider a walker in two dimensions with fixed step size a and random angle direction. The algorithm for a programme that simulates this walk is very simple. Suppose the walker is at the location

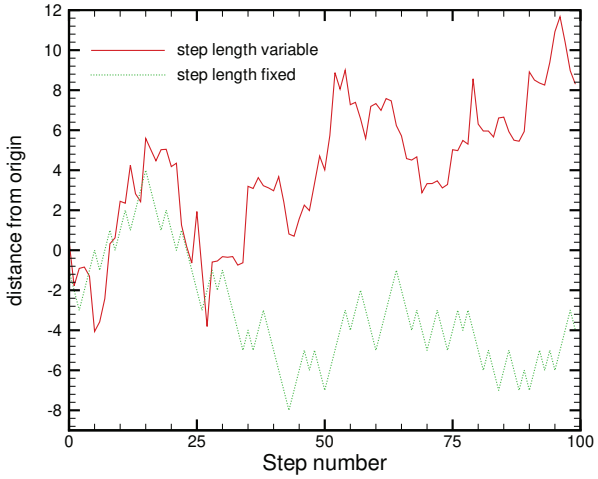

Figure 10.16: *Profiles of a walker with $N = 100$ steps of variable length in comparison to a normal random walk.*

(x_i, y_i) at the end of ith step. Generate a random number $0 < r < 1$ and multiply it by 2π. The quantity $\theta_{i+1} = 2\pi r$, which lies between zero and 2π, will be the direction of the next step $i + 1$ with respect to the direction of the preceding step θ_i. Then update the walker position to $(x_{i+1}, y_{i+1}) = (x_i + a\cos\theta_{i+1}, y_i + a\sin\theta_{i+1})$. The programme `2DContinuumWalk` (see Appendix 10.C for details) simulates the motion of a two-dimensional Rayleigh random walk. Let us see the profile of a typical Rayleigh walk with $N = 10000$ steps of fixed length in figure (10.20) which has started from the origin. In figures (10.21) and (10.22) we show the N dependence of $\langle R\rangle$ and $\langle(\Delta R)^2\rangle$ on N respectively. Ensemble averaging has been performed over $M = 100000$ samples. In contrast to one dimension, the dependence of $\langle R\rangle$ on N is not linear. It turns out that this dependence takes a power law form. To see this explicitly, let us find the fitting exponent ν from the power law behaviour $\langle R\rangle \sim N^{\nu}$. By linear fitting to the equation $log\langle R\rangle = \nu log N$ we obtain $\nu = 0.78$. Another interesting quantity is $P_N(R)\Delta R$, the probability that R is between R and $R + \Delta R$ after N steps. In figure (10.23) we exhibits $P(R)$ for $N = 200$ steps and com-

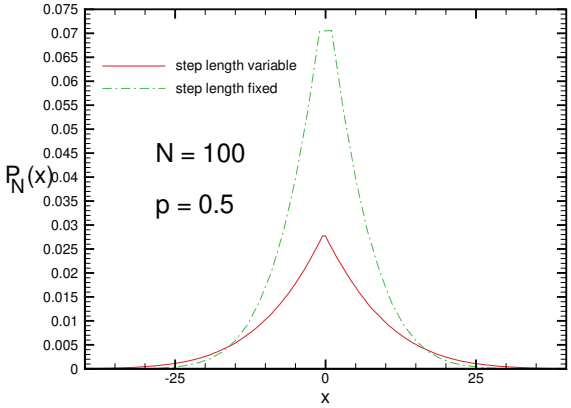

Figure 10.17: *Probability distribution $P_N(x)$ for walkers with variable and fixed step lengths. We have taken $M = 100000$ for ensemble averaging.*

pare it to the theoretical form $P_{th}(R) = 2\pi R e^{-\frac{(R-\langle R\rangle)^2}{2(\Delta R)^2}}$. Note that in the theoretical formula, we have used the computed values for $\langle R\rangle$ and $\langle \Delta R^2\rangle$ (Weiss, 1994).

10.4.4 Random walk with a trap

Up to now, our one-dimensional random walk has had no spatial restriction and the walker can be found at any position even indefinitely far from its starting point. Now we wish to consider a restricted random walk. Imagine a one-dimensional lattice with trapping sites at $x = 0$ and $x = L$. A walker begins at site x_0 $(0 < x_0 < L)$ and takes unit steps to the left and right with equal probability. When the walker arrives at a trapping site, the walk terminates. We want to find the mean first passage time τ to a tapping site (Redner, 2008). See figure (10.24) for illustration. It can be verified that the mean number of steps for the particle to be trapped (the mean first passage time) is given by $\tau = (2D)^{-1}x_0(L-x_0)$ where D is the self-diffusion coefficient in the absence of the traps. We note from (10.11) that in one dimension $D = 2pq = 2p(1-p)$. Figure (10.25) shows the computed mean first passage time τ and its theoretical formula $\tau = \frac{1}{2D}x_0(L-x_0)$ in which L

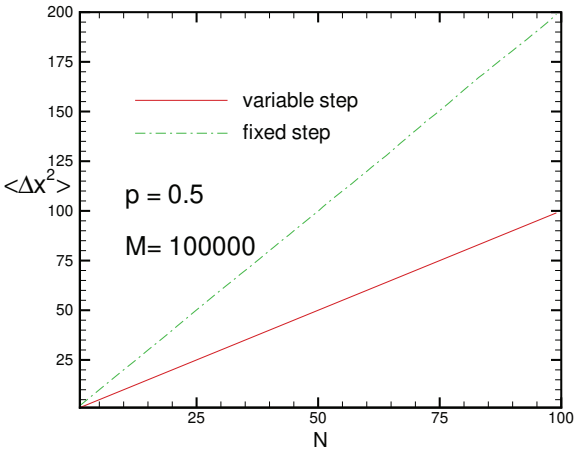

Figure 10.18: *Mean squared displacement* $\langle(\Delta x)^2\rangle(N)$ *for walkers with variable and fixed step lengths. Ensemble averaging has been performed over* $M = 10^5$ *members.*

is the system length. We have taken $L = 100$ in our code and averaged over an ensemble of $M = 10^4$ trials. The agreement between theory and simulation is very good. Let us consider another type of restricted random walk. Imagine a 1D random walk with reflecting boundary condition at the left (right) boundary $x = -L$ $(+L)$. When the walker reaches $x = -L$ $(+L)$ it is reflected toward $-L + 1$ $(L - 1)$. At $t = 0$, the walker starts at $x = 0$ and steps with equal probabilities to left or right. We have simulated this random walk and obtained $P_N(x)$ where N denotes the number of steps. On average, after N steps the walker has reached a distance $R(N) \sim \sqrt{\langle(\Delta x)^2\rangle}$ from its starting point. We expect those N for which $R(N)$ is less than the distance to the boundaries, the probability distribution function $P_N(x)$ is not affected by the presence of the reflecting boundaries. For $R(N) \geq L$, we expect deviations from the free random walk. Figure (10.26) compares $P_N(x)$ in the presence of reflecting boundaries to the free case for $N = 2000$. At this N we have $R(N) = \sqrt{2000} \sim 44$ which is less than $L = 100$. The two distribution functions are almost identical to each other because the walker cannot feel the boundary's existence.

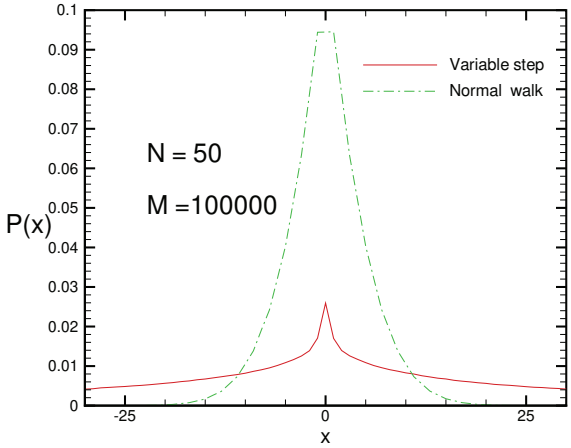

Figure 10.19: *$P_N(x)$ for walkers with variable and fixed lengths.*

Now let us increase N. Figure (10.27) shows the same quantity $P_N(x)$ for $N = 20000$. In this case, we have $R(N) = \sqrt{20000} \sim 141$ which is larger than $L = 100$. You see notable deviations from the unrestricted random walk. The walker feels the presence of the reflecting walls and its characteristics have undergone substantial changes.

10.4.5 Self-avoiding random walk

In most of the random walk problems we have considered so far, the walker has had no memory of its past. Of course, an exception was the persistence walk where the walker had a short-time local memory of its prior step. Here we want to address the question of global memory. Actually, in some physical processes, the assumption of having a sort of memory is appropriate. Let us introduce a prototype of such a kind of random walk the so-called *self-avoiding* random walk. In a self-avoiding walk (SAW) the walker is not allowed to pass a site that has been passed in the preceding steps. In other words, each site can at most be visited once by the walker. From the geometrical viewpoint, the profile of a SAW is not self-intersecting and its path does not cut itself. In programming a SAW, you should keep track

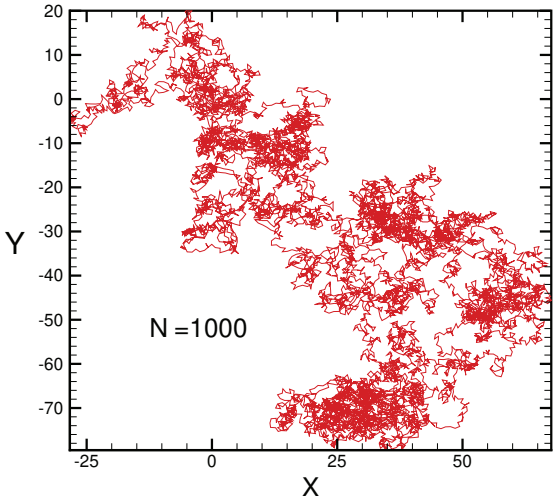

Figure 10.20: *Trajectory of a 2D Rayleigh walk with fixed (unit) step length and* $N = 1000$ *steps.*

of site occupation. For this purpose assign a binary value $s = 0, 1$ to each site. At the beginning of the simulation set the array s equal to one for all sites i.e.; $s[i] = 1 \quad i = 1, 2, \cdots$. Whenever a site is visited turn its status number into $s = 0$. At each step n, randomly choose one of the adjacent sites of the walker which have not been visited and move the walker to this new site with the prescribed probability. Remember that you may reach a site that has no non-visited adjacent sites. In this case, the walk is terminated. The basic difference of a SAW ensemble is that each trial m has its own duration N_m. Suppose we want to compute the mean-squared displacement after N walks. We should count the number of trials whose durations are larger or equal to N and make an average over them. More precisely, suppose the trial m lasts for N_m steps. This trial can contribute to those $\langle(\Delta \mathbf{r}_N)^2\rangle$ in which $N \leq N_m$. There is an elementary algorithm called *growing* in the context of SAWs (Giordano and Nakanishi, 2006). As compared to a normal random walk, a SAW will, on average, go farther distances for the same number of steps. It can be

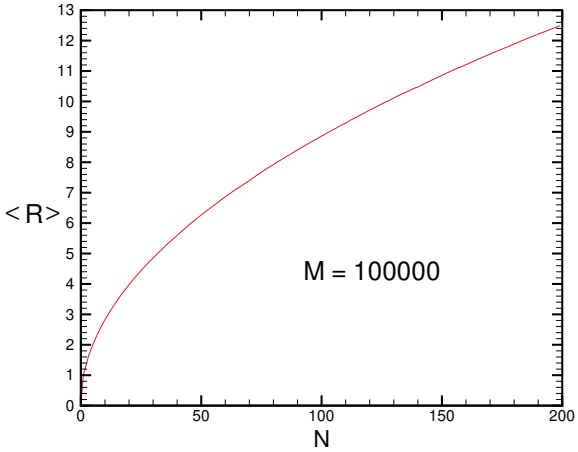

Figure 10.21: *$\langle R \rangle$ versus N of a 2D Rayleigh walk with fixed (unit) step length.*

shown that for large times the mean-squared displacement of a SAW has a power law behaviour $\langle (\Delta \mathbf{r})^2 \rangle \sim t^{2\nu}$ where ν is the Flory exponent (Nakanishi, 1994). The Flory exponent has a dimensionality dependence. For $d = 2, 3$ we have $\nu = 0.75$ and 0.6 respectively which are both higher than the normal random walk exponent 0.5. As a matter of fact, the simulation method that we explained is subjected to flaws. We refer interested readers to consult other books such as (Giordano and Nakanishi, 2006) for further details where you can see nice graphs for SAWs. Self-avoiding walk can mimic the basic ingredients of a physical process in polymer physics (de Gennes, 1979; Dio and Edvards, 2001). A polymer is comprised of N repeated units (monomers) with $N >> 1$. Although, the fine structure of the polymer is important for many practical applications, such details can be ignored if we are interested in its global properties. Let us now introduce a random walk model that incorporates the global aspects of dilute linear polymer chains in a solution. As an idealisation, we assume the monomers can only occupy the sites of a regular lattice. An important physical feature of a polymer chain is that no two monomers can occupy the same spatial position. This constraint is known as the

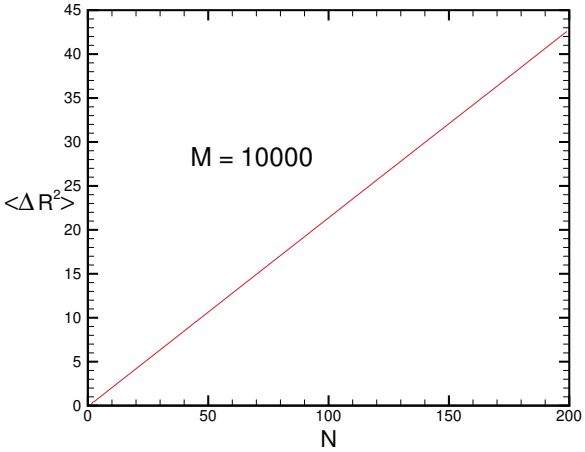

Figure 10.22: $\langle (\Delta R)^2 \rangle$ *of a 2D Rayleigh walk with fixed (unit) step length.*

excluded volume condition, which is ignored in a simple random walk. This constraint is nevertheless, satisfied by the self-avoiding random walk. A correspondence between a SAW of N steps and a polymeric configuration of a N -monomer polymer is now clear. The random walk's root mean-squared displacement $\sqrt{\langle (\Delta \mathbf{r})^2 \rangle}$ gives its place to *end-to-end* mean distance $\langle R_N \rangle$ between the first and the last sites of a polymer consisting of N monomers. We leave the further discussion on SAWs to other references such as (Rosenbluth and Rosenbluth, 1955; Wall and Mandel, 1975; Prellberg and Krawczyk, 2004).

10.5 Random walk and diffusion

You may have read in several places that random walk is connected to diffusion. This is indeed true. Here we want to make this analogy on quantitative grounds. Consider an ensemble of random walkers on a 2D square lattice. They all start their walk from the origin. To make the problem more realistic you may assume that these particles are physical entities such as tiny coffee powder particles in water. The question that we wish to address is how the particles' density

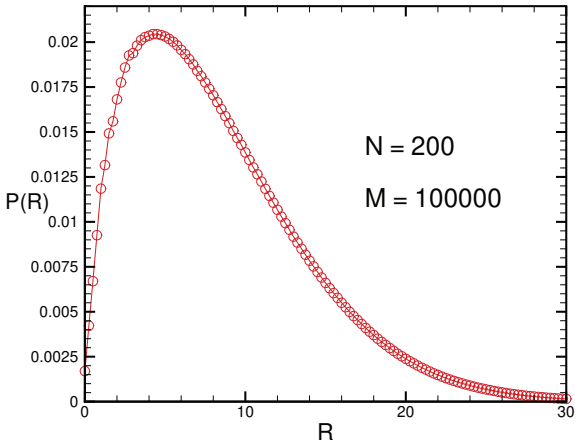

Figure 10.23: *$P_N(R)$ of a 2D Rayleigh walk with a fixed (unit) step length for $N = 200$ steps.*

$\rho(x, y, t)$ evolves in time. Up to now, we have focused on the individual properties of a random walk. However, the concept of ensemble can have a physical interpretation as will shortly see. To obtain $\rho(x, y, t)$ let us again come back to our single random walker who is performing a random walk. Let $P(i, j, n)$ denote the probability that the walker is at site (i, j) of a discrete 2D lattice in n-th step. The following master equation can be written for $P(i, j, n)$:

$$P(i,j,n) = p_r P(i-1,j,n-1) + p_l P(i+1,j,n-1) +$$

$$p_u P(i,j-1,n-1) + p_d P(i,j+1,n-1) \tag{10.18}$$

A rearrangement of (10.18) gives:

$$P(i,j,n) - P(i,j,n-1) = p_r[P(i-1,j,n-1) - P(i,j,n-1)] +$$

$$p_l[P(i+1,j,n-1) - P(i,j,n-1)] + p_u[P(i,j-1,n-1) - P(i,j,n-1)]$$

$$+ p_d[P(i,j+1,n-1) - P(i,j,n-1)] \tag{10.19}$$

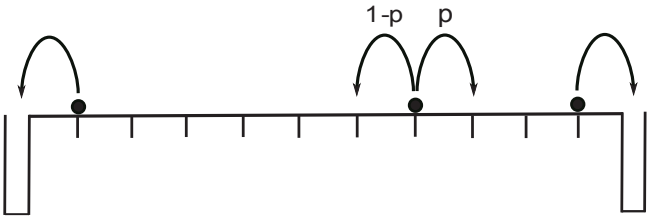

Figure 10.24: *A trapping random walk in one dimension.*

Now we coarse-grain the space and time by letting both the timestep Δt and the step length a go to zero. We can thus speak of the continuous time and space variables $t = n\Delta t, x = ia$, and $y = ja$. The function $P(i, j, t)$ will be substituted by $P(x, y, t)$. Dividing both sides of (10.19) by Δt and taking the continuum limit we find:

$$\frac{\partial P(x,y,t)}{\partial t} = \frac{p_r}{\Delta t}[P(x-a,y,t-\Delta t) - P(x,y,t-\Delta t)] +$$

$$\frac{p_l}{\Delta t}[P(x+a,y,t-\Delta t) - P(x,y,t-\Delta t)] + \frac{p_u}{\Delta t}[P(x,y-a,t-\Delta t) -$$

$$P(x,y,t-\Delta t)] + \frac{p_d}{\Delta t}[P(x,y+a,t-\Delta t) - P(x,y,t-\Delta t)] \quad (10.20)$$

Taylor Expanding the expressions inside brackets up to the term a^2 gives (we neglect the terms linear in Δt):

$$\frac{\partial P(x,y,t)}{\partial t} = \frac{(p_l - p_r)a}{\Delta t}\frac{\partial P(x,y,t)}{\partial x} + \frac{(p_d - p_u)a}{\Delta t}\frac{\partial P(x,y,t)}{\partial y}$$

$$+(p_l + p_r)\frac{a^2}{2\Delta t}\frac{\partial^2 P(x,y,t)}{\partial x^2} + (p_u + p_d)\frac{a^2}{2\Delta t}\frac{\partial^2 P(x,y,t)}{\partial y^2} \quad (10.21)$$

To have a diffusive behaviour we assume that the $\frac{a^2}{\Delta t}$ approaches $4D$ in the continuum limit where $a, \Delta t \to 0$ and D is the self-diffusion constant. In a symmetric random walk, we have $p_l = p_r = p_d = p_u = 0.25$ therefore the first two terms in the right-hand side of (10.21) vanish and we arrive at the diffusion equation:

$$\frac{\partial P(x,y,t)}{\partial t} = D[\frac{\partial^2 P(x,y,t)}{\partial x^2} + \frac{\partial^2 P(x,y,t)}{\partial y^2}] = D\nabla^2 P(x,y,t) \quad (10.22)$$

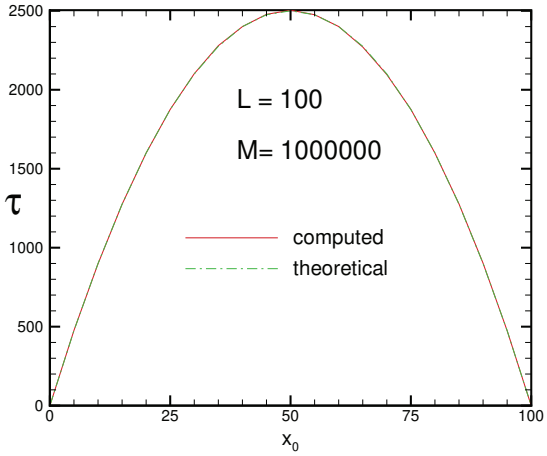

Figure 10.25: *Average trapping time τ vs x_0 for a one dimensional restricted random walk. The system length is $L = 100$.*

For a non-symmetric random walk, the diffusion equation becomes biased. Let us define the continuum limits of $\frac{(p_l - p_r)a}{\Delta t}$ and $\frac{(p_d - p_u)a}{\Delta t}$ by C_x and C_y respectively. The two-dimensional biased diffusion equation becomes:

$$\frac{\partial P(x,y,t)}{\partial t} = 2D[(p_l + p_r)\frac{\partial^2 P(x,y,t)}{\partial x^2} + (p_u + p_d)\frac{\partial^2 P(x,y,t)}{\partial y^2}]+$$

$$C_x \frac{\partial P(x,y,t)}{\partial x} + C_y \frac{\partial P(x,y,t)}{\partial y} \qquad (10.23)$$

To see the analytical solution of the biased diffusion equation consult (Weiss, 1994). In exercises, you are asked to solve (10.23) by a linear transformation of coordinates. Generalization of (10.22) to higher dimensions is straightforward. We should not forget that a differential equation is not completed unless the initial and boundary values are prescribed. In our random walk problem, we took the starting point of the walker at the origin therefore the initial condition turns out to be: $P(x,y,0) = \delta(x)\delta(y)$. As for the boundary condition we can take $P(x,y,t) = 0$ when $r = \sqrt{x^2 + y^2} \to \infty$. Coming back to our density

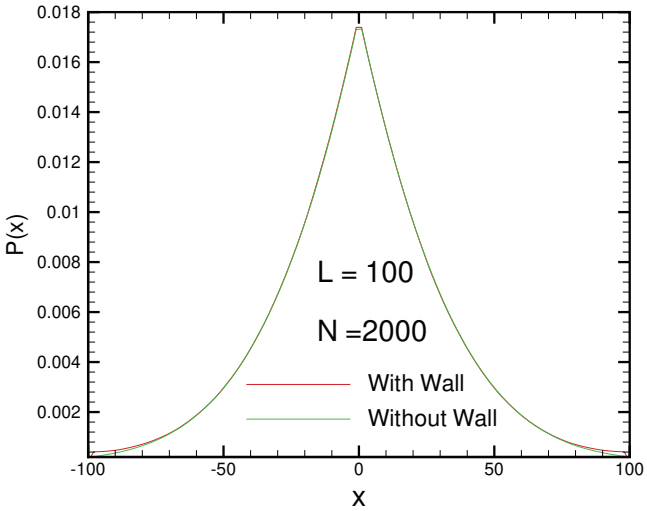

Figure 10.26: *The probability distribution function $P_N(x)$ for a one-dimensional reflecting symmetric random walk with $N = 2000$ steps. The result is compared with the non-reflecting random walk. The system length has been taken $L = 100$.*

evolution problem, the grain density $\rho(x, y, t)$ is simply obtained by multiplying $P(x, y, t)$ by the ensemble number M. We remark that the diffusion equation (10.22) is mathematically identical to the heat equation we encountered in chapter (5). We interchangeably use heat and diffusion. This is not accidental. The spread of phonon wave packets which are responsible for heat transport is a random phenomenon and can be modeled by a diffusion equation. To reinforce the connection between the diffusion equation and random walk, we draw your attention to $P(m, N)$ i.e.; the probability that the walker is at site m after taking N steps. You will prove (I hope) in exercises that in large N limit $P(m, N)$ becomes Gaussian with a width equal to MSD $\langle(\Delta x)^2\rangle$. Interestingly the solution of the diffusion equation also became a Gaussian with a width equal to $2Dt$ which is identical to the MSD of a random walker when N is large enough (see Eq. (5.21) of chapter (5).

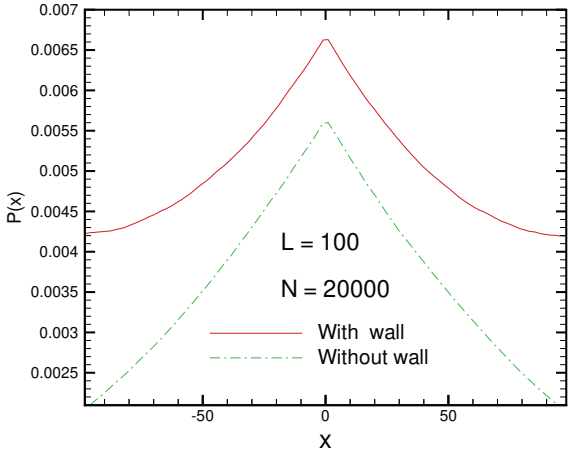

Figure 10.27: *Distribution function $P_N(x)$ for $N = 20000$ steps.*

10.6 Random walk and entropy

Entropy is a concept that entered physics in the mid-19th via the experimental works of French engineer Nicholas Saadi Carnot and later more rigorously by German physicist and mathematician Rudolf Julius Emanuel Clausius who proved the Clausius inequality named after him. Austrian physicist Ludwig Boltzmann reconsidered entropy from the statistical physics viewpoint in the late nineteen century and established a firm statistical basis for the second law of thermodynamics with his outstanding H-theorem. Now that we can simulate some random processes it would illustrative and fascinating to verify one important consequence of the second law of thermodynamics which is the principle of entropy increase. According to this corollary of the second law, the entropy of an isolated system increases until it reaches an equilibrium. Entropy is a very useful quantity for non-equilibrium systems and can illustrate how a far-from-equilibrium system approaches equilibrium. Physically speaking, entropy is an extensive thermodynamic state variable. For simple systems such as ideal and van der Waals gases, entropy can be obtained as a function of independent thermodynamic variables namely T and p (W. Greiner and Stöcker,

2001). Here we look into entropy from a statistical physics point of view. We recall the statistical definition of entropy S (Reif, 1965):

$$S = -\sum_i P_i ln P_i \tag{10.24}$$

where P_i is the probability that a system is in its i configurational state. As a simple example (perhaps the simplest) let our system be a random walker executing a discrete-time one-dimensional random walk on a lattice. If the lattice is infinite the system never reaches an equilibrium state. However, if we restrict the walker between two reflecting walls the situation will be entirely different. Let the lattice contain $2L+1$ sites $i = -L, -L+1, \cdots, -1, 0, 1, \cdots, L-1, L$. Suppose a walker starts its symmetric random walk from the origin at $t = 0$ and let $P(i, n)$ denote the probability that the walker is at site i after taking n steps. Our initial condition implies $P(i, 0) = \delta_{i,0}$. We want to obtain the system entropy $S(n)$ at the $n-$th step. According to (10.24) we have:

$$S(n) = -\sum_{i=-L}^{L} P(i, n) ln P(i, n) \tag{10.25}$$

To obtain $P(i, n)$ we proceed recursively and write down a master equation for it as follows:

$$P(i, n) = \frac{1}{2} P(i-1, n-1) + \frac{1}{2} P(i+1, n-1) \tag{10.26}$$

The restrictive geometry implies the boundary conditions $P(-L, n) = P(+L, n) = 0$. Let us now simulate the problem. Let an ensemble of M independent walkers execute a random walk. Each walker starts its walk from the origin $i = 0$. The probability that the site i is occupied at step n can simply be obtained as $P(i, n) = \frac{1}{M} \sum_{m=1}^{M} s^{(m)}(i, n)$ where $s^{(m)}(i, n)$ is zero (one) if the $m-$th walker is (is not) at site i after step n. In figure (10.28) we have shown the site occupation profile at various times for a symmetric reflecting random walk on a 101-site chain ($L = 50$). As you can see for sufficiently large times the occupation probability becomes almost uniform throughout the lattice. In the problems, you are asked to obtain the temporal evolution of the system entropy.

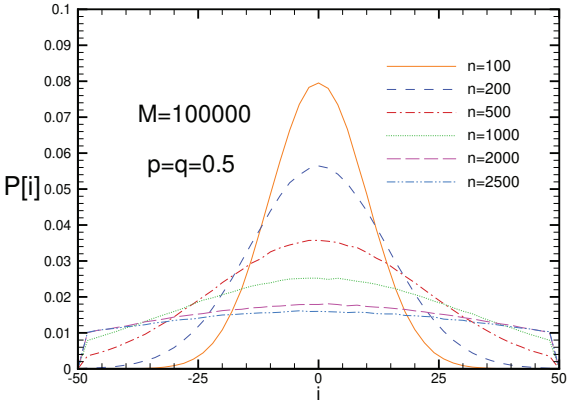

Figure 10.28: *Site occupation probability profile for a restricted 1D symmetric random walk on a 1D chain of* 101 *sites. Average has been performed over* $M = 10^6$ *trials.*

10.7 Problems

Problem 10.1 Modify the particle in a box problem in a manner that the probability of moving a particle to the other segment depends on the present number of particles of the chosen segment. Write down the corresponding differential equation for the average number of particles in the left segment.

Problem 10.2

(a) Derive the formula (10.12) for $P(m, N)$.

(b) Choose $N = 10$ and draw $P(m, N)$ as a function of m.

(c) What would be the shape of $P(m, N)$ when N goes to a large limit?

Problem 10.3

(a) Using the analytical form of $P(m, N)$, obtain the mean position of the walker after taking N steps.

(b) Let n_R be the random variable that shows the number of right steps in the sample walk. Analytically show that its mean value is $\langle n_R \rangle = pN$.

(c) Show its dispersion $\langle (\Delta n_R)^2 \rangle$ is equal to Npq.

Problem 10.4

(a) Numerically compute $P(m, N)$ for $N = 10, 100$ and 1000 and draw it versus m.

(b) At which m the probability distribution function is maximised?

(c) According to the central limit theorem in statistics, for large N the probability function $P(m, N)$ approaches a Gaussian function of the form:

$$P_G(x) = \frac{1}{\sqrt{2\pi \langle (\Delta x)^2 \rangle}} e^{-\frac{(x - \langle x \rangle)^2}{2\langle (\Delta x)^2 \rangle}} \tag{10.27}$$

(d) For each $N = 10, 100, 1000$ compute $\langle x_N \rangle, \langle (\Delta x)^2 \rangle$ and $P(m, N)$ numerically and plot them on the same graph. Verify that for large $N, P(m, N)$ approaches the Gaussian function.

Problem 10.5 An interesting quantity in a random walk is the mean number of lattice sites $\langle D_N \rangle$ visited during a N step walk. Compute and draw $\langle D_N \rangle$ in terms of N for various values of p.

Problem 10.6 Another interesting quantity in a random walk is the mean first passage time $\langle \tau_m \rangle$ which is the meantime the site m is first visited by the walker during a course of a N step walk (Redner, 2008). Compute and draw $\langle \tau_m \rangle$ versus m for various values of N. Fix m and sketch $\langle \tau_m \rangle$ as a function of p.

Problem 10.7 Assume that $\langle (\Delta x)^2 \rangle \sim N^{2\nu}$ for large N in a persistent random walk. Estimate the value of the exponent ν from a log-log plot of MSD versus N for large N. Plot the dependence of ν on α.

Problem 10.8 Consider a one-dimensional random walk with variable step length. Take the probability density function of the step length to be uniform in the interval $[-a, +a]$ that is $f(a) = \frac{1}{2a}$. Note the

step direction is automatically included in $f(a)$ so you do not need to choose the walk direction. Simulate the random walk and obtain the MSD $\langle(\Delta x)^2\rangle$ as a function of N. Does the MSD resembles a power law function $\langle(\Delta x)^2\rangle \sim N^\beta$ for large N If yes, then estimate the value of β from a log-log plot of MSD versus N for large N.

Problem 10.9

(a) By a suitable change of variables (x, y) into (ξ, η) change the biased diffusion equation (10.23) into an unbiased equation in the new coordinates (ξ, η).

(b) Solve the unbiased equation and then by a reverse transformation of coordinates obtain the desired solution in (x, y). For simplicity take $p_l + p_r = p_u + p_d$.

Problem 10.10 **Random walk in a disordered medium**: Consider a one-dimensional random walk in a disordered medium in which the step probabilities depend on the walker's location. Assume the medium is a lattice and let a rightward walk probability be $p = 0.5$ in all sites except at site $i = d > 0$. At this site, we have $p_d = f$.

(a) simulate the problem and find the $\langle x_N\rangle$ and $\langle(\Delta x_N)^2\rangle$ as a function of f for various values of d.

(b) take the right step probability at site i to be a random variable p_i from the probability distribution $p(x)$. Obtain the mean and mean squared displacement for various choices of $p(x)$ and compare them to their corresponding values in a normal 1D random walk.

Problem 10.11 **Random walk on triangular lattice**: Consider a random walk on a two-dimensional triangular lattice with lattice constant a (distance between nearest neighbours). Here each site has six nearest neighbours. Take the moving probability to each of the six nearest neighbours equal to each other.

(a) Simulate the walk and obtain the mean squared displacement as a function of N (number of steps). Does the MSD behave linearly in time (step numbers)?

(b) Obtain the diffusion coefficient D of this triangular lattice random walk and compare it to a two-dimensional random walk on a square lattice. Are they the same? which lattice has a larger D?

(c) Try to analytically obtain the solution. Hint: show the step variable $\mathbf{s}_n$ at $n-$th step by a phasor $\mathbf{s}_n = ae^{i\theta_n}$ where $\theta_n = \frac{2\pi s}{6}$ $s = 1, \cdots, 6$. The walker distance after N step becomes $\mathbf{r}_n = \mathbf{s}_1 + \mathbf{s}_2 + \cdots + \mathbf{s}_N$.

Problem 10.12 Obtain the entropy evolution of a restrictive 1D random walk between reflecting boundaries discussed in the text. Discuss the system size on the steady-state value of the entropy.

Problem 10.13 Obtain the entropy evolution of other types of random walk on a 1D lattice. Discuss persistent, and step-length variable random walks. Which type of random walk reaches equilibrium sooner?

Problem 10.14 Numerically compute the entropy of a disordered random walk in which one of the sites is defective. Take the right hopping probability as $p = \alpha \neq 1/2$ for the defective site which is located at site $0 < d$ of a chain of length $L + 1$.

Bibliography

J. Tobochnik H. Gould and W. Chriastian. *An Introduction to Computer Simulation Methods: Applications to Physical Systems*. Addison Wesely, third edition, 2006.

Tao Pang. *An introduction to computational physics*. Cambridge university press, 1st edition, 2006.

Philipp O.J. Scherer. *Computational physics*. Springer, 1st edition, 2010.

Manuel José Páez Rubin H. Landau and Christian C. Bordeianu. *A Survey of Computational Physics*. Princeton university press, 1st edition, 2008.

Nicholas J. Giordano and Hisao Nakanishi. *Computational physics*. Pearson Prentice Hall, 1st edition, 2006.

N. G. van Kampen. *Stochastic Processes in Physics and Chemistry*. North-Holland, Amsterdam, 1st edition, 1992.

L. E. Reichel. *A modern course in statistical physics*. John Wiley and sons Inc, 1st edition, 1998.

I. N. Sneddon. *Elements of partial differential equations*. McGraw Hill, New York, 1st edition, 1957.

Tyn Myint-U and Lukenath Debnath. *Linear partial differential equations for scientists and engineers*. Birkhäuser, 1st edition, 2007.

Sadri Hassani. *Mathematical physics: A modern introduction to its foundation*. Springer, 2nd edition, 2013.

G. Nicolis and I. Prigogine. *Self-Organisation in Nonequilibrium Systems*. John Wiley and sons, New York, 1st edition, 1977.

C. J. Jachimowski D. A. Mcquarrie and M. E. Russel. Kinetics of small systems. ii. *J. Chem. Phys.*, 40:2914, 1994.

M. E. Foulaadvand. Solution of radio active multiple decay equations. *Gamma (Quartely Persian journal in educational physics: www.gammajournal.ir)*, 10:40, 2006.

R. Lefever and G. Nicolis. Chemical instabilities and sustained oscillations. *J. Theor. Biol.*, 30:267, 1971.

Keith. R. Symon. *Mechanics*. Addison Wesley, 3rd edition, 1971.

Stephen T. Thornton and Jerry. B. Marion. *Classical dynamics of particles and systems*. Brooks Cole, 5th edition, 2003.

Grant. R. Fowles and George L. Cassiday. *Analytical Mechanics*. Brooks Cole, 7th edition, 2004.

Atam P. Arya. *Introduction to classical mechanics*. Benjamin Cummings, 2nd edition, 1997.

Murray Spiegel. *Schaums Mathematical Handbook of Formulas and Tables*. McGraw-Hill, 2nd edition, 1998.

A. Stepanek. The aerodynamics of tennis balls, the topspin lob. *Am .J. Physics*, 56 (5):138, 1988.

A. Cromer. Stable solutions using the euler approximation. *Am. J. Phys.*, 49:455, 1981.

Alejandro Garcia. *Numerical Methods for Physics*. Benjamin Cummings, 2nd edition, 1999.

D. Masoumi and M. E. Foulaadvand. Constant force simple non harmonic oscillator. *Gamma (Persian educational physics journal: www.gammajournal.ir)*, 21:15, 2008.

Amir Aghamohammadi. *Waves*. Alzahra University Press, Tehran, 1st edition, 2021.

Symour Lipschutz. *Schaum's Outline of Linear Algebra*. McGraw-Hill, 5th edition, 2012.

M. H. Yaghoubi A. Aghamohammadi, M. E. Foulaadvand and A. H. Mousavi. Normal modes of a defected linear system of beaded springs. *Am .J. Physics*, 85 (3):193, 2017.

B. Cipra A. Kolan and B. Titus. unknown. *Computer in physics*, 9 (4):387, 1985.

M. L. Williams and H. J. Maris. Numerical study of phonon localization in disordered systems. *Phys. Rev. B*, 31:4508, 1985.

A. S. Householder J. H. Wilkinson, C. Reinsch and F. L. Bauer. *Handbook for automatic computation: Volume 2: Linear Algebra*. Springer, 1st edition, 1986.

W. T. Vetterling W. H. Press, S. A. Teukolsky and B. P. Flannery. *Numerical Recipes in C*. Oxford university press, 2nd edition, 2002.

F. J. Vesely. *Computational physics*. Kluwer Academic/Plenum Publishers, 2nd edition, 2001.

M. Ebrahim Foulaadvand Somayyeh Belbasi and Y. S. Joe. Anti-resonance in a one-dimensional chain of driven coupled oscillators. *American Journal of Physics*, 82(1):4508, 2014.

F. P. Incropera T. L. Bergman, A. S. Lavine and D. P. Dewitt. *Fundamentals of Heat and Mass Transfer*. 7th edition, 2011.

L. Petit E. Guyon, J-P Hulin and C. D. Mitescu. *Physical hydrodynamics*. Oxford university press, 1st edition, 2001.

G. B. Arfken and H. L. Weber. *Mathematical methods for Physicists*. Elsevier, 6th edition, 2005.

J. Crank and P. Nicolson. A practical method for numerical evaluation of solutions of partial differential equations of the heat-conduction type. *Proceedings of the Cambridge Philosophical Society*, 43, No. 50:50–67, 1947.

Sadri Hassani. *Mathematical methods For students of physics and related*. Springer, 2nd edition, 2009.

Klaus A. Hoffmann and Steve T. Chiang. *Computational fluid dynamics*. Engineering Education System, 4th edition, 2009.

Debashish Chowdhury Andreas Schadschneider and Katsohiro Nishinari. *Stochastic Transport in Complex Systems: From Molecules to Vehicles*. Elsevier Science, 1st edition, 2010.

A. Schadschneider L. Santen and D. Chowdhurry. Statistical physics of vehicular traffic and some related systems. *Physics Report*, 329: 199–329, 2000.

D. Helbing. Traffic and related self-driven many-particle systems. *Rev. Mod. Physics*, 73:1067, 2001.

B. Diu C. Cohen-Tannoudji and F. Laloe. *Quantum mechanics*. Wiley-VCH, 1st edition, 1992.

Richard Liboff. *Introductory quantum mechanics*. Pearson, 4th edition, 2002.

Robert. M. Eisberg and Robert Resnick. *Quantum physics of atoms, molecules, solids, nuclei and particles*. John-wiley and sons Inc., 1st edition, 1974.

Herbert Goldstein. *Classical mechannics*. Addison-Wesley, 3rd edition, 2001.

M. P. Allen and D. J. Tildesley. *Computer simulation of liquids*. Oxford Science Publications, 1st edition, 1986.

Daan Frenkel and Berend Smit. *Understanding Molecular Simulation: From Algorithms to Applications*. Academic Press, 2nd edition, 2002.

L. I. Schiff. *Quantum Mechanics*. McGraw-Hill, New York, 3rd edition, 1968.

J. E. Jones. On the determination of molecular fields. ii. from the equation of state of a gas. *Proc. Roy. Soc. (Lond.)*, 106A:463, 1924.

J. M. Haile. *Molecular dynamics simulation*. John Wiley and Sons, Inc., 1st edition, 1992.

L. Verlet. Computer experiments on classical fluids i. *Phys. Rev.*, 159: 98, 1967.

B.J. Alder and T. E. Wainwright. On the determination of molecular fields. ii. from the equation of state of a gas. *J. Chem. Phys.*, 31: 459, 1959.

Jenő Sólyom. *Fundamentals of the physics of solid*. Springer, 1st edition, 2007.

A. Rahman. On the determination of molecular fields. ii. from the equation of state of a gas. *Phys. Rev.*, 136:A405, 1964.

Dennis C. Rapaport. *The art of molecular dynamics simulations*. Cambridge university press, 2nd edition, 1995.

F. Reif. *Fundamentals of statistical and thermal physics*. McGraw Hill Inc., 1st edition, 1965.

Karl Pearson. The problem of the random walk. *Nature*, 72:294, 1905.

M. Smoluchowski. Zur kinetischen theorie der brownschen molekularbewegung und der suspensionen. *Annalen der Physik*, 326 (14): 756–780, 1906.

George H. Weiss. *Aspects and the applications of random walk*. Northholand, 1st edition, 1994.

Sidney Redner. *A guide to first-passage processes*. Cambridge University Press, 2nd edition, 2008.

H. Nakanishi. *Random and self-avoiding walks in a disordered medium, Annual reviews of Computational physics, Vol.1, D. Stauffer, Ed.* World Scientific, 1994.

Pierre-Gilles de Gennes. *Scaling concepts in polymers*. Cornell University, 1st edition, 1979.

M. Dio and S. F. Edvards. *The theory of polymer dynamics*. Clarendon Press Oxford, 8th edition, 2001.

Marshall N. Rosenbluth and Arianna W. Rosenbluth. Monte carlo calculation of the average extension of molecular chains. *J. Chem. Phys.*, 23:356, 1955.

Frederick T. Wall and Frederic Mandel. Macromolecular dimensions obtained by an efficient monte carlo method without sample attrition. *J. Chem. Phys.*, 63:4592, 1975.

Thomas Prellberg and Jaroslaw Krawczyk. Flat histogram version of the pruned and enriched rosenbluth method. *Phys. Rev. Lett.*, 92: 120602, 2004.

Ludwig Neise W. Greiner and Horst Stöcker. *Thermodynamics and Statistical Mechanics*. Springer-Verlag, 1st edition, 2001.