Chapter 12
Random Walks

We explore applications of random walks to a variety of systems and the generation of random number sequences.

12.1 Introduction

We introduced the random walk problem in Section 7.2 in the context of the motion of drunken sailors and the one-dimensional motion of particles. Of course, random walks are not restricted to one dimension nor are the applications limited to the wanderings of inebriates. We already know that we can use random walk methods to estimate a definite integral (see Section 11.8). In this chapter we introduce some of the more popular random walk models and discuss several applications. In succeeding chapters we discuss other applications of random walk methods to problems that in many cases have no obvious connection to random walks.

We begin with the most elementary version of a random walk model, a walk on a one-dimensional translationally invariant lattice with steps occurring at uniform time intervals. In Section 7.2 we showed that the variance or dispersion

\[ \langle \Delta x_N^2 \rangle = \langle x_N^2 \rangle - \langle x_N \rangle^2 \]  

(12.1)

depends linearly on \( N \), where \( N \) is the number of steps. Many applications of random walk models make use of asymptotic results for large \( N \). For example, in many cases \( \langle \Delta x_N^2 \rangle \) satisfies a power law for sufficiently large \( N \), that is,

\[ \langle \Delta x_N^2 \rangle \sim N^{2\nu}. \quad (N \gg 1) \]  

(12.2)

In this context the symbol \( \sim \) is interpreted as “asymptotically equal to” and the relation (12.2) is an example of an asymptotic scaling law. For the simple one-dimensional random walk model we know that the relation (12.2) is valid for all \( N \) and that \( \nu = \frac{1}{2} \). In many of the random walk
problems introduced in this chapter, we will determine if such a power law dependence exists for large $N$.

Although we introduced the simple one-dimensional random walk model in Section 7.2, we did not consider all its properties. Some additional properties of this model are considered in Problem 12.1.

**Problem 12.1.** Discrete time random walks in one dimension

a. Suppose that the probability of moving to the right is $p = 0.7$. Compute $\langle x_N \rangle$ and $\langle x_N^2 \rangle$ for $N = 4, 8, 16$, and $32$. What is the interpretation of $\langle x_N \rangle$ in this case? What is the qualitative dependence of $\langle \Delta x_N^2 \rangle$ on $N$? Does $\langle x_N^2 \rangle$ depend simply on $N$?

b. Use the error analysis discussed in Section 11.4 to estimate the number of trials needed to obtain $\langle \Delta x_N^2 \rangle$ to 1% accuracy for $N = 8$ and $N = 32$.

c. An interesting property of random walks is the mean number $\langle D_N \rangle$ of *distinct* lattice sites visited during the course of an $N$ step walk. Do a Monte Carlo simulation of $\langle D_N \rangle$ and determine its $N$ dependence.

We can equally well consider either a large number of successive walks as in Problem 12.1 or a large number of similar (noninteracting) walkers moving at the same time. In Problem 12.2 we consider the motion of many random walkers moving independently of one another on a two-dimensional lattice.

**Problem 12.2.** A random walk in two dimensions

Consider a collection of walkers initially at the origin of a square lattice. At each unit of time, each of the walkers moves at random with equal probability in one of the four possible directions. The following program implements this algorithm and shows the sites that have been visited.

```java
package edu.clarku.sip.chapter12;
import edu.clarku.sip.plot.*;
import edu.clarku.sip.templates.*;
import edu.clarku.sip.graphics.*;
import java.awt.*;
import java.util.Random;

// random walk in two dimensions
public class RandomWalk implements AnimationModel, Runnable, Drawable {
    private int nwalkers; // number of walkers
    private int xpositions[];
    private int ypositions[];
    private Control myControl = new SAnimationControl(this);
    private Thread animationThread;
    private World2D world = new World2D();
    private Random random;
```
public RandomWalk()
{
    world.addDrawable(this);
    random = new Random();
}

public void startCalculation()
{
    nwalkers = (int) myControl.getValue("nwalkers");
    xpositions = new int[nwalkers];
    ypositions = new int[nwalkers];
    random.setSeed(7);
    animationThread = new Thread(this);
    animationThread.start();
}

public void run()
{
    while (animationThread != null)
    {
        step();
        try
        {
            Thread.sleep(100);
        } catch(InterruptedException e){}
    }
}

public void stopCalculation()
{
    animationThread = null;
}

public void step()
{
    move();
    world.render();
}

public void continueCalculation()
{
    animationThread = new Thread(this);
    animationThread.start();
}
public void clear(){}  

public void reset()  
{
    myControl.setValue("nwalkers", 200);
}

void move()  
{
    for (int i = 0; i < nwalkers; i++)
        choice(i);
}

void choice(int i)  
{
    double p = random.nextDouble();
    if (p <= 0.25)
        xpositions[i] = xpositions[i] + 1;
    else if (p <= 0.5)
        xpositions[i] = xpositions[i] - 1;
    else if (p <= 0.75)
        ypositions[i] = ypositions[i] - 1;
    else
        ypositions[i] = ypositions[i] + 1;
}

public void draw(World2D world, Graphics g)  
{
    for (int i = 0; i < nwalkers; i++)
    {
        double x = xpositions[i];
        double y = ypositions[i];
        int size = 2;
        int px = world.xToPix(x - size/2);
        int py = world.yToPix(y - size/2);
        g.setColor(Color.red);
        g.fillRect(px, py, size, size);
    }
}

public static void main(String[] args)  
{
    RandomWalk rw = new RandomWalk();
    rw.reset();
}
}
a. Run Program RandomWalk with the number of walkers \( n_{\text{walkers}} \geq 200 \) and the number of steps taken by each walker \( N \geq 500 \). If each walker represents a bee, describe the qualitative nature of the shape of the swarm of bees. Describe the qualitative nature of the surface of the swarm as a function of \( N \). Is the surface jagged or smooth?

b. Compute the quantities \( \langle x_N \rangle \), \( \langle y_N \rangle \), \( \langle \Delta x_N^2 \rangle \), and \( \langle \Delta y_N^2 \rangle \) as a function of \( N \). The average is over the walkers. Also compute the net mean square displacement \( \langle \Delta R_N^2 \rangle \) given by

\[
\langle \Delta R_N^2 \rangle = \langle x_N^2 \rangle + \langle y_N^2 \rangle - \langle x_N \rangle^2 - \langle y_N \rangle^2.
\] (12.3)

What is the dependence of each quantity on \( N \)?

c. Enumerate all the random walks on a square lattice for \( N = 4 \) and obtain exact results for \( \langle x_N \rangle \), \( \langle y_N \rangle \) and \( \langle \Delta R_N^2 \rangle \). Assume that all four directions are equally probable. Verify your program by comparing the Monte Carlo and exact enumeration results.

d. Estimate \( \langle \Delta R_N^2 \rangle \) for \( N = 8, 16, 32, \) and 64 using a reasonable number of trials for each value of \( N \). Assume that \( \langle \Delta R_N^2 \rangle \) has the asymptotic \( N \) dependence:

\[
\langle \Delta R_N^2 \rangle \sim N^{2\nu}, \quad (N >> 1)
\] (12.4)

and estimate the exponent \( \nu \) from a log-log plot of \( \langle \Delta R_N^2 \rangle \) versus \( N \). If \( \nu \approx \frac{1}{2} \), estimate the magnitude of the self-diffusion coefficient \( D \) given by

\[
\langle R_N^2 \rangle \sim 2dDN.
\] (12.5)

The form (12.5) is similar to (8.39) with the time \( t \) in (8.39) replaced by the number of steps \( N \).

e. Estimate the quantities \( \langle x_N \rangle \), \( \langle y_N \rangle \), \( \langle R_N^2 \rangle = \langle x_N^2 + y_N^2 \rangle \), and \( \langle \Delta R_N^2 \rangle \) for the same values of \( N \) as in part (d), with the probabilities 0.4, 0.2, 0.2, 0.2, corresponding to a step to the right, left, up, and down, respectively. This choice of probabilities corresponds to a biased random walk with a drift to the right. What is the interpretation of \( \langle x_N \rangle \) in this case? What is the dependence of \( \langle \Delta R_N^2 \rangle \) on \( N \)? Does \( \langle R_N^2 \rangle \) depend simply on \( N \)?

f. Consider a random walk that starts at a site that is a distance \( y = h \) above a horizontal line (see Figure 12.1). If the probability of a step down is greater than the probability of a step up, we expect that the walker will eventually reach a site on the horizontal line. This walk is a simple model of the fall of a rain drop in the presence of a random swirling breeze. Do a Monte Carlo simulation to determine the mean time \( \tau \) for the walker to reach any site on the line \( x = 0 \) and find the functional dependence of \( \tau \) on \( h \). Is it possible to define a velocity in the vertical direction? Because the walker does not always move vertically, it suffers a net displacement \( \Delta x \) in the horizontal direction. How does \( \langle \Delta x^2 \rangle \) depend on \( h \) and \( \tau \)? Reasonable values for the step probabilities are 0.1, 0.6, 0.15, 0.15, corresponding to up, down, right, and left, respectively.

g. Do a Monte Carlo simulation of \( \langle \Delta R_N^2 \rangle \) on the triangular lattice (see Figure 8.5) and estimate \( \nu \). Can you conclude that \( \nu \) is independent of the symmetry of the lattice? Does \( D \) depend on the symmetry of the lattice? If so, give a qualitative explanation for this dependence.
12.2 Modified Random Walks

So far we have considered random walks on one- and two-dimensional lattices where the walker has no “memory” of the previous step. What happens if the walkers remember the nature of their previous steps? What happens if there are multiple random walkers, with the condition that no double occupancy is allowed? We explore these and other variations of the simple random walk in this section. All these variations have applications to physical systems.

Problem 12.3. A persistent random walk

a. In a “persistent” random walk, the transition or “jump” probability depends on the previous transition. Consider a walk on a one-dimensional lattice, and suppose that step \( N - 1 \) has been made. Then step \( N \) is made in the same direction with probability \( \alpha \); a step in the opposite direction occurs with probability \( 1 - \alpha \). Write a program to do a Monte Carlo simulation of the persistent random walk in one dimension. Compute \( \langle x_N \rangle \), \( \langle x_N^2 \rangle \), \( \langle \Delta x_N^2 \rangle \), and \( P_N(x) \). Note that it is necessary to specify both the initial position and an initial direction of the walker. What is the \( \alpha = \frac{1}{2} \) limit of the persistent random walk?

b. Consider the cases \( \alpha = 0.25 \) and \( \alpha = 0.75 \) and determine \( \langle \Delta x_N^2 \rangle \) for \( N = 8, 64, 256, \) and \( 512 \). Estimate the value of \( \nu \) from a log-log plot of \( \langle \Delta x_N^2 \rangle \) versus \( N \) for large \( N \). Does \( \nu \) depend on \( \alpha \)? If \( \nu \approx \frac{1}{2} \), determine the self-diffusion coefficient \( D \) for \( \alpha = 0.25 \) and \( 0.75 \). Give a physical argument why \( D(\alpha \neq 0.5) \) is greater (smaller) than \( D(\alpha = 0.5) \).

c. A persistent random walk can be considered as an example of a multistate walk in which the state of the walk is defined by the last transition. In the above example, the walker is in one of two states; at each step the probabilities of remaining in the same state or switching states are \( \alpha \) and \( 1 - \alpha \) respectively. One of the earliest applications of a two state random walk was to the study of diffusion in a chromatographic column. Suppose that a molecule in a chromatographic column can be either in a mobile phase (constant velocity \( v \)) or in a trapped...
phase (zero velocity). Instead of each step changing the position by \( \pm 1 \), the position at each step changes by \( +v \) or 0. A quantity of experimental interest is the probability \( P_N(x) \) that a molecule has traveled a distance \( x \) in \( N \) steps. Choose \( v = 1 \) and \( \alpha = 0.75 \) and compute the qualitative behavior of \( P_N(x) \). Explain why the molecule cannot diffuse in either state, but that it is still possible to define an effective diffusion coefficient for the molecule.

d. You might have expected that the persistent random walk yields a nonzero value for \( \langle x_N \rangle \). Verify that \( \langle x_N \rangle = 0 \), and explain why this result is exact. How does the persistent random walk differ from the biased random walk for which \( p \neq q \)?

The fall of a raindrop considered in Problem 12.2f is an example of a restricted random walk, that is, a walk in the presence of a boundary. (Another example of a restricted random walk was considered in Problem 7.12c.) In the following problem, we discuss in a more general context the effects of various types of restrictions or boundaries on random walks. Another example of a restricted random walk is given in Problem 12.10.

**Problem 12.4.** Restricted random walks

a. Consider a one-dimensional lattice with “trap” sites at \( x = 0 \) and \( x = a \) \( (a > 0) \). A walker begins at site \( x_0 \) \( (0 < x_0 < a) \) and takes unit steps to the left and right with equal probability. When the walker arrives at a trap, it vanishes. Do a Monte Carlo simulation and verify that the mean number of steps \( \tau \) for the particle to be trapped (the first passage time) is given by

\[
\tau = (2D)^{-1} x_0 (a - x_0).
\]  

(12.6)

\( D \) is the self-diffusion coefficient in the absence of the traps, and the average is over all possible walks.

b. Random walk models in the presence of traps have had an important role in condensed matter science. For example, consider the following idealized model of energy transport in solids. The solid is represented as a lattice with two types of sites: hosts and traps. An incident photon is absorbed at a host site and excites the host molecule or atom. The excitation energy or exciton is transferred at random to one of the host’s nearest neighbors and the original excited molecule returns to its ground state. In this way the exciton wanders through the lattice until it reaches a trap site. The exciton is then trapped and a chemical reaction occurs.

A simple version of this energy transport model is given by a one-dimensional lattice with traps placed on a periodic sublattice. Because the traps are placed at regular intervals, we can replace the random walk on an infinite lattice by a random walk on a ring. Consider a ring of \( N \) host or nontrapping sites and one trap site. If a walker has an equal probability of starting from any host site and an equal probability of a step to each nearest neighbor site, what is the \( N \) dependence of the mean survival time \( \tau \) (the mean number of steps taken before a trap site is reached)? Use the results of part (a) rather than doing another simulation.

c. Consider a one-dimensional lattice with reflecting sites at \( x = -a \) and \( x = a \). For example, if a walker reaches the reflecting site at \( x = a \), it is reflected at the next step to \( x = a - 1 \). At \( t = 0 \), the walker starts at \( x = 0 \) and steps with equal probability to nearest neighbor sites. Write a Monte Carlo program to determine \( P_N(x) \), the probability that the walker is at site \( x \) after \( N \) steps. Compare the form of \( P_N(x) \) with and without the presence of the reflecting “walls.” Can
you distinguish the two probability distributions if $N$ is the order of $a$? At what value of $N$ can you first distinguish the two distributions?

Although all of the above problems involved random walks on a lattice, it was not necessary to store the positions of the lattice sites or the path of the walker. In the following problem, we consider a random walk model that requires us to store the positions of a system of random walkers for which double occupancy is excluded, that is, only one walker can occupy a site. The physical motivation of this model arises from solid state physics where the diffusing particles are thermal vacancies whose density depends on the temperature. The main physical quantity of interest is the

*Problem 12.5. Diffusion of interacting particles*

Consider a square lattice with a nonzero density $\rho$ of particles. Each particle moves at random to empty nearest neighbor sites. Double occupancy of sites is excluded; otherwise the particles are noninteracting. The main physical quantity of interest is the self-diffusion coefficient $D$ of an individual particle. The model can be summarized by the following algorithm:

i. Occupy at random the $L \times L$ sites of a square lattice with $N$ particles subject to the conditions that no double occupancy is allowed, and the density $\rho = N/L^2$ has the desired value. (Remember that $\rho < 1$.) Tag each particle, that is, distinguish it from the others, and record its initial position in an array.

ii. At each step choose a particle and one of its nearest neighbor sites at random. If the neighbor site is empty, the particle is moved to this site; otherwise the particle remains in its present position. The measure of “time” in this context is arbitrary. The usual definition is that during one unit of time or one Monte Carlo step per particle, each particle attempts one jump on the average. That is, the time is advanced by $1/N$ each time a particle is chosen even if the particle does not move.

The diffusion coefficient $D$ is obtained as the limit $t \to \infty$ of $D(t)$, where $D(t)$ is given by

$$D(t) = \frac{1}{2dt} \langle \Delta R(t)^2 \rangle,$$

(12.7)

and $\langle \Delta R(t)^2 \rangle$ is the net mean square displacement per tagged particle after $t$ units of time. An example of a program that implements this algorithm is given in the following.

```java
package edu.clarku.sip.chapter12;
import edu.clarku.sip.templates.*;
import edu.clarku.sip.graphics.*;
import edu.clarku.sip.plot.*;
import java.awt.*;
import java.util.Random;

// simulation of particle diffusion in a lattice gas
public class LatticeGas implements AnimationModel, Lattice, Runnable {
```
private int L; // linear dimension of lattice
private double Lhalf;
private int N; // number of particles
private final int OCCUPIED = 1;
private final int EMPTY = 0;
private Thread animationThread;
private Color occupiedColor = Color.red;
private Color emptyColor = Color.blue;
private LatticeWorld latticeWorld = new LatticeWorld();
private Control myControl = new SAnimationControl(this);
private int site[][];
private int x[]; // keeps track of the occupied sites
private int y[]; // keeps track of the occupied sites
private int x0[]; // keeps track of the occupied sites
private int y0[];
private Random random;
private int xposition;
private int yposition;
private Plot plot = new Plot("time", "R2Bar", "Lattice Gas");
private double R2bar;
private double t;

public LatticeGas()
{  plot.setAutoUpdate(true);
}

public int getNumberOfSites()
{  return L;
}

public void startCalculation()
{  L = (int) myControl.getValue("L");
N = (int) myControl.getValue("N");
site = new int[L][L];
x = new int[N + 1];
y = new int[N + 1];
x0 = new int[N + 1];
y0 = new int[N + 1];
latticeWorld.setLattice(this);
Lhalf = 0.5*L;
t = 0;
random = new Random(13907);
fillLattice();
animationThread = new Thread(this);
animationThread.start();
}

public void continueCalculation(){}
public void clear(){}

public void reset()
{
    myControl.setValue("L",40);
    myControl.setValue("N",500);
}

public void stopCalculation()
{
    animationThread = null;
}

public void run()
{
    while(animationThread != null)
    {
        step();
        try
        {
            Thread.sleep(100);
        }
        catch(InterruptedException e){}
    }
}

public void step()
{
    move();
    computeR2Bar();
    t++;
    plot.addPoint(0, t, R2bar);
    latticeWorld.render();
}

// fill the lattice with N particles
void fillLattice()
{
    int i = 0;
    while (i != N)
    {
int xadd = (int)(L*random.nextDouble());
int yadd = (int)(L*random.nextDouble());
if (site[xadd][yadd] == EMPTY)
{
    i = i + 1;  // number of particles added
    site[xadd][yadd] = OCCUPIED;  // site occupied
    x[i] = xadd;  // x[0] is ignored
    y[i] = yadd;
    x0[i] = x[i];  // x-coordinate at t = 0
    y0[i] = y[i];
}
}

void move()
{
    for (int particle = 1; particle <= N; particle++)
    {
        int i = (int)(N*random.nextDouble());
        xposition = x[i];
        yposition = y[i];
        chooseDirection();  // move left, right, up, or down
        if (site[xposition][yposition] == EMPTY)  // if new site is unoccupied
        {
            site[x[i]][y[i]] = EMPTY;  // place where it was is now empty
            x[i] = xposition;  // assigns new positions
            y[i] = yposition;
            site[x[i]][y[i]] = OCCUPIED;  // new site occupied
        }
    }
}

// choose random direction and use periodic boundary conditions
void chooseDirection()
{
    int dir = (int)(4*random.nextDouble()) + 1;
    switch(dir)
    {
    case 1:
        xposition = xposition + 1;
        if (xposition >= L)
            xposition = 0;
        break;
    case 2:
        xposition = xposition - 1;
        if (xposition < 0)
```java
CHAPTER 12. RANDOM WALKS

```
CHAPTER 12. RANDOM WALKS

```java
drawSite(lw, g, i, j);
}

public void drawSite(LatticeWorld lw, Graphics g, int x, int y)
{
    if (site[x][y] == OCCUPIED)
        g.setColor(occupiedColor);
    else
        g.setColor(emptyColor);
    int px = lw.xToPix(x);
    int py = lw.yToPix(y);
    int size = lw.getCellSize();
    g.fillRect(px, py, size, size);
}

public static void main(String[] args)
{
    LatticeGas lg = new LatticeGas();
    lg.reset();
}
```

a. Do a Monte Carlo simulation to determine $D$ on a square lattice for $\rho = 0, 0.1, 0.2, 0.3, 0.5, 0.7$. Choose $L \geq 40$. Although $D$ is defined as the limit $t \to \infty$ of (12.7), $D(t)$ for this model fluctuates after a short equilibration time and no improvement in accuracy is achieved by increasing $t$. Better statistics for $D$ can be obtained by averaging $D$ over as many particles as possible and hence by considering a lattice with $L$ as large as possible. The accuracy of $D$ also can be increased by averaging $\langle R(t)^2 \rangle$ over different initial starting times. Why is it necessary to limit the number of Monte Carlo steps so that $\langle R(t)^2 \rangle$ is less than $(L/2)^2$? Verify that deviations of $D(t)$ from its mean value are proportional to the inverse square root of the total number of particles that enter into the average in (12.7).

b. Why is $D$ a monotonically decreasing function of the density $\rho$? To gain some insight into this dependence, determine the dependence on $\rho$ of the probability that if a particle jumps to a vacancy at time $t$, it returns to its previous position at time $t+1$. Is there a qualitative relation between the density dependence of $D$ and this probability?

c. Consider a one-dimensional lattice model for which particles move at random, but double occupancy of sites is excluded. This restriction implies that particles cannot pass by each other. Compute $\langle \Delta x^2 \rangle$ as a function of $t$. Do the particles diffuse, that is, is $\langle \Delta x^2 \rangle$ proportional to $t$? If not, what is the $t$ dependence of $\langle \Delta x^2 \rangle$?

Problem 12.6. Random walk on a continuum

One of the first continuum models of a random walk was proposed by Rayleigh in 1919. The model is known as the freely jointed chain in polymer physics. In this model the length $a$ of each step is a random variable with probability density $p(a)$, and the direction of each step is uniformly random.
CHAPTER 12. RANDOM WALKS

For simplicity, we first consider a walker in two dimensions with \( p(a) \) chosen so that each step has unit length. At each step the walker takes a step of unit length at a random angle. Write a Monte Carlo program to compute \( P_N(r) \Delta r \), the probability that the displacement of the walker is in the range \( r \) to \( r + \Delta r \) after \( N \) steps, where \( r \) is the distance from the origin. Verify that for sufficiently large \( N \), the probability density \( P_N(r) \) can be approximated by a Gaussian. Is a Gaussian a good approximation for small \( N \)? Is it necessary to do a Monte Carlo simulation to confirm that \( \langle R^2_N \rangle \sim N \), or can you give a simple argument for this dependence based on the form of \( P_N(r) \)?

**Problem 12.7.** Random walks with steps of variable length

a. Consider a random walk in one dimension with jumps of all lengths allowed. The probability density that the length of a single step is \( a \) is denoted by \( p(a) \). If the form of \( p(a) \) is given by \( p(a) = e^{-a} \), what is the form of \( P_N(x) \)? Suggestions: Use the inverse transform method discussed in Section 11.5 to generate step lengths according to the probability density \( p(a) \). Consider a walk of \( N \) steps and determine the net displacement \( x \). Generate many such walks and determine \( P_N(x) \). Plot \( P_N(x) \) versus \( x \) and confirm that the form of \( P_N(x) \) is consistent with a Gaussian distribution. Is this random walk equivalent to a diffusion process for sufficiently large \( N \)?

b. Assume that the probability density \( p(a) \) is given by \( p(a) = C/a^2 \) for \( a \geq 1 \). Determine the normalization constant \( C \) using the condition \( C \int_1^\infty a^{-2} da = 1 \). Does the second moment of \( p(a) \) exist? Do a Monte Carlo simulation as in part (a) and verify that the form of \( P_N(x) \) is given by

\[
P_N(x) \sim \frac{bN}{x^2 + b^2 N^2};
\]

(12.8)

What is the magnitude of the constant \( b \)? Does the variance \( \langle x^2 \rangle - \langle x \rangle^2 \) of \( P_N(x) \) exist? Is this random walk equivalent to a diffusion process?

**Problem 12.8.** The central limit theorem

Consider a continuous random variable \( x \) with probability density \( f(x) \). That is, \( f(x) \Delta x \) is the probability that \( x \) has a value between \( x \) and \( x + \Delta x \). The \( m \)th moment of \( f(x) \) is defined as

\[
\langle x^m \rangle = \int x^m f(x) \, dx.
\]

(12.9)

The mean value \( \langle x \rangle \) is given by (12.9) with \( m = 1 \). The variance \( \sigma_x^2 \) of \( f(x) \) is defined as

\[
\sigma_x^2 = \langle x^2 \rangle - \langle x \rangle^2.
\]

(12.10)

Consider the sum \( y_n \) corresponding to the average of \( n \) values of \( x \):

\[
y_n = \frac{1}{n} (x_1 + x_2 + \ldots + x_n).
\]

(12.11)

We adopt the notation \( y = y_n \). Suppose that we make many measurements of \( y \). We know that the values of \( y \) are not identical, but are distributed according to a probability density \( P(y) \), where \( P(y) \Delta y \) is the probability that the measured value of \( y \) is in the range \( y \) to \( y + \Delta y \). The main quantities of interest are the mean \( \langle y \rangle \), the variance \( \sigma_y^2 = \langle y^2 \rangle - \langle y \rangle^2 \), and \( P(y) \) itself.
a. Suppose that \( f(x) \) is uniform in the interval \([-1, 1]\). Calculate \( \langle x \rangle \) and \( \sigma_x \) analytically. Use a Monte Carlo method to make a sufficient number of measurements of \( y \) to determine \( P(y), \langle y \rangle \), and \( \sigma_y \) with reasonable accuracy. For example, choose \( n = 1000 \) and make 100 measurements of \( y \). Verify that \( \sigma_y \) is approximately equal to \( \sigma_x / \sqrt{n} \). Plot \( P(y) \) versus \( y \) and discuss its qualitative form. Does the form of \( P(y) \) change significantly if \( n \) is increased? Does the form of \( P(y) \) change if the number of measurements of \( y \) is increased?

b. To test the generality of the results of part (a), consider the exponential probability density

\[
f(x) = \begin{cases} 
e^{-x}, & \text{if } x \geq 0 \\ 0, & \text{if } x < 0 \end{cases}
\]

Calculate \( \langle x \rangle \) and \( \sigma_x \) analytically. Modify your Monte Carlo program and estimate \( \langle y \rangle, \sigma_y \), and \( P(y) \). Is \( \sigma_y \) related to \( \sigma_x \) as in part (a)? Plot \( P(y) \) and discuss its qualitative form and its dependence on \( n \) and on the number of measurements of \( y \).

c. Let \( y \) be the Monte Carlo estimate of the integral (see Problem 11.3a)

\[
4 \int_0^1 dx \sqrt{1 - x^2}.
\]

In this case \( y \) is found by sampling the integrand \( f(x) = 4 \sqrt{1 - x^2} \) \( n \) times. Choose \( n \geq 1000 \) and make at least 100 measurements of \( y \). Show that the values of \( y \) are distributed according to a Gaussian distribution. How is the variance of \( P(y) \) related to the variance of \( f(x) \)?

d. Consider the Lorentzian probability density

\[
f(x) = \frac{1}{\pi} \frac{1}{x^2 + 1}.
\]

Calculate the mean value \( \langle x \rangle \). Does the second moment and hence the variance of \( f(x) \) exist? Do a Monte Carlo calculation of \( \langle y \rangle, \sigma_y \), and \( P(y) \). Plot \( P(y) \) as a function of \( y \) and discuss its qualitative form. What is the dependence of \( P(y) \) on the number of trials?

Problem 12.8 illustrates the central limit theorem which states that the probability distribution of a variable \( y \) is a Gaussian centered at \( \langle y \rangle \) with a standard deviation \( 1/\sqrt{n} \) times the standard deviation of \( f(x) \). The requirements are that \( f(x) \) has finite first and second moments, that the measurements of \( y \) are statistically independent, and that \( n \) is large. Use the central limit theorem to explain your results in Problem 12.8 and in Problem 12.7a. What is the relation of the central limit theorem to the calculations of the probability distribution in the random walk models that we already have considered?

Problem 12.9. Generation of the Gaussian distribution

Consider the sum

\[
y = \sum_{i=1}^{12} r_i,
\]

(12.15)
where \( r_i \) is a uniform random number in the unit interval. Make many “measurements” of \( y \) and show that the probability distribution of \( y \) approximates the Gaussian distribution with mean value 6 and variance 1. Discuss how to use this result to generate a Gaussian distribution with arbitrary mean and variance. This way of generating a Gaussian distribution is particularly useful when a “quick and dirty” approximation is appropriate.

Many of the problems we have considered have revealed the slow convergence of Monte Carlo simulations and the difficulty of obtaining quantitative results for asymptotic quantities. We conclude this section with a cautionary note and consider a “simple” problem for which straightforward Monte Carlo methods give misleading asymptotic results.

Problem 12.10. Random walk on lattices containing random traps

a. We have considered the mean survival time of a one-dimensional random walker in the presence of a periodic distribution of traps (see Problem 12.4b). Now suppose that the trap sites are distributed at random on a one-dimensional lattice with density \( \rho \). If a walker is placed at random at any nontrapping site, determine its mean survival time \( \tau \), the mean number of steps before a trap site is reached. Assume that the walker has an equal probability of moving to nearest neighbor sites at each step and use periodic boundary conditions.

This problem is more difficult than it might first appear, and there are a number of pitfalls. The major complication is that it is necessary to perform three averages: the distribution of traps, the origin of the walker, and the different walks for a given trap distribution and origin. Choose reasonable values for the number of trials associated with each average and do a Monte Carlo simulation to estimate the mean survival time \( \tau \). If \( \tau \) exhibits a power law dependence on \( \rho \), for example, \( \tau \approx \tau_0 \rho^{-z} \), estimate the exponent \( z \).

b. A seemingly straightforward extension of part (a) is to estimate the probability of survival \( S_N \) of an \( N \) step random walk. Choose \( \rho = 0.5 \) and do a Monte Carlo simulation of \( S_N \) for \( N \) as large as possible. (Published results are for \( N = 2 \times 10^3 \) on lattices with \( L = 50000 \) sites and 50000 trials.) Assume that the asymptotic form of \( S_N \) is given by

\[
S_N \sim e^{-bN^\alpha},
\]  
(12.16)

where \( b \) is a constant that depends on \( \rho \). Are your results consistent with this form? Is it possible to make a meaningful estimate of the exponent \( \alpha \)?

c. The object of part (b) is to convince you that it is not possible to use Monte Carlo methods directly to obtain the correct asymptotic behavior of \( S_N \). The difficulty is that we are trying to estimate \( S_N \) in the asymptotic region where \( S_N \) is very small, and the small number of samples in this region prevent us from obtaining meaningful results. It has been proved using analytical methods, that the asymptotic \( N \) dependence of \( S_N \) has the form (12.16), but with \( \alpha = 1/3 \). Are your Monte Carlo results consistent with this value of \( \alpha \)?

d. A method that reduces the number of required averages and hence reduces the fluctuations is to determine exactly, for a given distribution of trap sites, the probability that the walker is at site \( i \) after \( N \) steps. The method is illustrated in Figure 12.2. The first line represents a given configuration of traps distributed randomly on a one-dimensional lattice. One walker is placed at each regular site; trap sites are assigned the value 0. Because each walker moves with
CHAPTER 12. RANDOM WALKS

12.3 Applications to Polymers

Random walk models play an important role in polymer physics (cf. de Gennes). A polymer consists of \( N \) repeat units (monomers) with \( N \) very large (\( N \sim 10^3 - 10^5 \)). For example, polyethylene can be represented as \( \cdots - \text{CH}_2-\text{CH}_2-\text{CH}_2-\cdots \). The detailed structure of the polymer is important for many practical applications. For example, if we wish to improve the fabrication of rubber, a good understanding of the local motions of the monomers in the rubber chain is essential. However, if we are interested in the \( \textit{global} \) properties of the polymer, the details of the chain structure can be ignored.
Let us consider a familiar example of a polymer chain in a good solvent: a noodle in warm water. A short time after we place a noodle in warm water, the noodle becomes flexible, and it neither collapses into a little ball or becomes fully stretched. Instead, it adopts a random structure as shown schematically in Figure 12.3. If we do not add too many noodles, we can say that the noodles behave as a dilute solution of polymer chains in a good solvent. The dilute nature of the solution implies that we can ignore entanglement effects of the noodles and consider each noodle individually. The presence of a good solvent implies that the polymers can move freely and adopt many different configurations.

A fundamental geometrical property that characterizes a polymer in a good solvent is the mean square end-to-end distance $\langle R^2_N \rangle$, where $N$ is the number of monomers. It is known that for a dilute solution of polymer chains in a good solvent, the asymptotic dependence of $\langle R^2_N \rangle$ is given by (12.4) with the exponent $\nu \approx 0.592$ in three dimensions. The result for $\nu$ in two dimensions is known to be exactly $\nu = 3/4$ for the model of polymers that we will discussed. The proportionality constant in (12.4) depends on the structure of the monomers and on the solvent. In contrast, the exponent $\nu$ is independent of these details.

We now discuss a random walk model that incorporates the global features of linear polymers in solution. We already have introduced a model of a polymer chain consisting of straight line segments of the same size joined together at random angles (see Problem 12.6). A further idealization is to place the polymer chain on a lattice (see Figure 12.3b). If we ignore the interactions of the monomers, this simple random walk model would yield $\nu = 1/2$, independent of the dimension and symmetry of the lattice. Because this result for $\nu$ does not agree with experiment, we know that we are overlooking an important physical feature of polymers.

Figure 12.3: (a) Schematic illustration of a linear polymer in a good solvent. (b) Example of the corresponding self-avoiding walk on a square lattice.
A more realistic model of linear polymers accounts for its most important physical feature, that is, two monomers cannot occupy the same spatial position. This constraint, known as the excluded volume condition, implies that the walk cannot be adequately described by a purely random walk. A well known lattice model for a flexible polymer chain that incorporates this constraint is known as the self-avoiding walk (SAW). Consider the set of all $N$ step walks starting from the origin subject to the global constraint that no lattice site can be visited more than once in each walk; this constraint accounts for the excluded volume condition. Self-avoiding walks have many applications in the sciences, such as the physics of magnetic materials and the study of phase transitions, and they are of interest as purely mathematical objects. Many of the obvious questions about them have resisted rigorous analysis, and exact enumeration and Monte Carlo simulation have played an important role in our current understanding. We consider Monte Carlo simulations of the self-avoiding walk in two dimensions in Problems 12.11 and 12.12. Another algorithm for the self-avoiding walk is considered in Project 12.20.

**Problem 12.11.** The two-dimensional self-avoiding walk

a. Consider the self-avoiding walk on the square lattice. Choose an arbitrary site as the origin and assume that the first step is “up.” The walks generated by the three other possible initial steps only differ by a rotation of the whole lattice and do not have to be considered explicitly. The second step can be in three possible directions because of the constraint that the walk cannot return to the origin. To obtain unbiased results, we generate a random number to choose one of the three directions. Successive steps are generated in the same way. Unfortunately, the walk will not usually continue indefinitely. The difficulty is that to obtain unbiased results, we must generate a random number (for example, 1, 2, or 3) as usual, even though one or more of the steps might lead to a self-intersection. If the next step does lead to a self-intersection, the walk must be terminated to keep the statistics unbiased. An example of a self-intersection for is shown in Figure 12.4a. The next step leads to a self-intersection and violates the constraint for $N = 3$. In this case a new walk is started again at the origin.

Write a program that implements this algorithm and record the fraction $f(N)$ of successful attempts of constructing polymer chains with $N$ total monomers. It is convenient to represent
the lattice as a two-dimensional array so that you can record the sites that already have been
visited. What is the qualitative dependence of \( f(N) \) on \( N \)? What is the maximum value of \( N \)
that you can reasonably consider? Determine the mean square end-to-end distance \( \langle R_N^2 \rangle \) for
these values of \( N \).

b. The disadvantage of the straightforward sampling method in part (a) is that it becomes very
inefficient for long chains, that is, the fraction of successful attempts decreases exponentially
fast. To overcome this attrition, several “enrichment” techniques have been developed. We first
discuss a relatively simple procedure proposed by Rosenbluth and Rosenbluth in which each
walk of \( N \) steps is associated with a weighting function \( W(N) \). Because the first step to the
north is always possible, we have \( W(1) = 1 \). In order that all allowed configurations of a given
\( N \) are counted equally, the weights \( W(N) \) for \( N > 1 \) are determined according to the following
possibilities:

(a) All three possible steps violate the self-intersection constraint (see Figure 12.4b). The walk
is terminated with a weight \( W(N) = 0 \), and a new walk is generated at the origin.

(b) All three steps are possible and \( W(N) = W(N-1) \).

(c) Only \( m \) steps are possible with \( 1 \leq m < 3 \) (see Figure 12.4c). In this case \( W(N) =
\left(\frac{m}{3}\right)W(N-1) \), and a random number is generated to choose one of the \( m \) possible steps.

The correct (unbiased) value of \( \langle R_N^2 \rangle \) is obtained by weighting \( R_{N,i}^2 \), the value of \( R_N^2 \)
found in the \( i \)th trial, by the value of \( W_i(N) \), the weight found for the particular walk. Hence we write

\[
\langle R_N^2 \rangle = \frac{\sum_i W_i(N) R_{N,i}^2}{\sum_i W_i(N)},
\]

where the sum is over all trials. Incorporate the Rosenbluth method into your Monte Carlo
program, and calculate \( \langle R_N^2 \rangle \) for \( N = 4, 8, 16, \) and 32. Estimate the exponent \( \nu \) from a log-log
plot of \( \langle R_N^2 \rangle \) versus \( N \). Can you distinguish your estimate for \( \nu \) from its random walk value
\( \nu = \frac{1}{2} \) ?

*Problem 12.12. The reptation method

One of the more efficient enrichment algorithms is the “reptation” method (see Wall and Mandel).
For simplicity, consider a model polymer chain in which all bond angles are \( \pm 90^\circ \). As an example
of this model, the five independent \( N = 5 \) polymer chains are shown in Figure 12.5. (Other chains
differ only by a rotation or a reflection.) The reptation method can be stated as follows:

i. Choose a chain at random and remove the tail link.

ii. Attempt to add a link to the head of the chain. There is a maximum of two directions in
which the new head link can be added.

iii. If the attempt violates the self-intersection constraint, return to the original chain and inter-
change the head and tail. Include the chain in the statistical sample.
CHAPTER 12. RANDOM WALKS

The above steps are repeated many times to obtain a statistical average of \( R_N^2 \).

As an example of the reptation method, suppose we choose chain \( a \) of Figure 12.5. A new link can be added in two directions (see Figure 12.6a), so that on the average we find, \( a \rightarrow \frac{1}{2}c + \frac{1}{2}d \). In contrast, a link can be added to chain \( b \) in only one direction, and we obtain \( b \rightarrow \frac{1}{2}e + \frac{1}{2}b \), where the tail and head of chain \( b \) have been interchanged (see Figure 12.6b). Confirm that \( c \rightarrow \frac{1}{2}e + \frac{1}{2}a \), \( d \rightarrow \frac{1}{2}c + \frac{1}{2}d \), and \( e \rightarrow \frac{1}{2}a + \frac{1}{2}b \), and that all five chains are equally probable. That is, the transformations in the reptation method preserve the proper statistical weights of the chains without attrition. There is just one problem: unless we begin with a double ended “cul-de-sac” configuration such as shown in Figure 12.7, we will never obtain such a configuration using the above transformation. Hence, the reptation method introduces a small statistical bias, and the calculated mean end-to-end distance will be slightly larger than if all configurations were considered. However, the probability of such trapped configurations is very small, and the bias can be neglected for most purposes.

Adopt the \( \pm 90^\circ \) bond angle restriction and calculate by hand the exact value of \( \langle R_N^2 \rangle \) for \( N = 5 \). Then write a Monte Carlo program that implements the reptation method. Generate one walk of \( N = 5 \) and use the reptation method to generate a statistical sample of chains. As a check on your Monte Carlo program, compute \( \langle R_N^2 \rangle \) for \( N = 5 \) and compare your result with the exact result. Then extend your Monte Carlo computations of \( \langle R_N^2 \rangle \) to larger \( N \). Modify the reptation model so that the bond angle also can be \( 180^\circ \). This modification leads to a maximum of three directions for a new bond. Compare the results of the two models.

Problem 12.13. The dynamics of polymers in a dilute solution

In principle, the dynamics of a polymer chain undergoing collisions with solvent molecules can
be simulated by using a molecular dynamics method. However, in practice only relatively small chains can be simulated in this way. An alternative approach is to use a Monte Carlo model that simplifies the effect of the random collisions of the solvent molecules with the atoms of the chain. Most of these models (cf. Verdier and Stockmayer) consider the chain to be composed of beads connected by bonds and restrict the motion of the beads to a lattice. For simplicity, we assume that the bond angles can be either $\pm 90^\circ$ or $180^\circ$. Begin with an allowed configuration of $N$ beads ($N + 1$ bonds). A possible starting configuration can be generated by taking successive steps in the positive $y$ direction and positive $x$ directions. The dynamics of the model is summarized by the following algorithm.

1. Select at random a bead (occupied site) on the polymer chain. If the bead is not an end site, then the bead can move to a nearest neighbor site of another bead if this site is empty and if the new angle between adjacent bonds is either $\pm 90^\circ$ or $180^\circ$. For example, bead 4 in Figure 12.8 can move to position 4' while bead 3 cannot move if selected. That is, a selected bead can move to a diagonally opposite unoccupied site only if the two bonds to which it is attached are mutually perpendicular.

2. If the selected bead is an end site, move it to one of two (maximum) possible unoccupied sites so that the bond to which it is connected changes its orientation by $\pm 90^\circ$ (see Figure 12.8).

3. If the selected bead cannot move, retain the previous configuration.

The physical quantities of interest include the mean square end-to-end distance $\langle R_N^2 \rangle$ and the mean square displacement of the center of mass of the chain $\langle \Delta R_{cm}^2 (N) \rangle$. The unit of time is the number of Monte Carlo steps per bead during which all beads have one chance on the average to move to a different site.
a. Consider a two-dimensional lattice and compute $\langle R^2_N \rangle$ and $\langle \Delta R^2_{\text{cm}}(N) \rangle$ for various values of $N$. How do these quantities depend on $N$? (The first published results for three dimensions were limited to 32 Monte Carlo steps per bead for $N = 8, 16$, and 32 and only 8 Monte Carlo steps per bead for $N = 64$.) Estimate the accuracy of your calculation by calculating the standard deviations of the means. Also compute the probability $P_N(N \Delta R)$ that the end-to-end distance is $R$. How does this probability compare to a Gaussian distribution?

b. We know that two configurations are strongly correlated if they differ by only the position of one bead. Hence, it would be a waste of computer time to measure the end-to-end distance and the position of the center of mass after every single move. Ideally, we wish to compute these quantities for configurations that are approximately statistically independent. Because we do not know a priori the mean number of Monte Carlo steps per bead needed to obtain configurations that are statistically independent, we need to estimate this time in our preliminary calculations. A simple way to estimate this time is to plot the time average of $R^2$ as a function of the time $t$. If you start with a configuration that is not typical, then you will notice that the
time average of $R^2$ eventually approaches a value that no longer changes with $t$ except for small fluctuations. Plot the difference between the time average of $R^2$ as a function of $t$ and determine if this difference can be roughly approximated by an exponential of the form $e^{-t/\tau_r}$? The time $\tau_r$ is known as the relaxation time. If $\tau_r$ depends on $N$, try to quantify this relationship.

(c) The relaxation time $\tau_r$ is usually the same order of magnitude as the correlation time $\tau_c$, where $\tau_c$ is the time needed to obtain statistically independent configurations. This time can be obtained by computing the equilibrium averaged autocorrelation function for a chain of fixed $N$:

$$C_N(t) = \frac{\langle R^2_N(t') R^2_N(t) \rangle - \langle R^2_N \rangle^2}{\langle R^4_N \rangle - 2\langle R^2_N \rangle^2}. \quad (12.20)$$

Note that $C_N(t)$ has the same form as the velocity correlation function (8.40) and the autocorrelation function that is introduced in (12.40). $C_N(t)$ is defined so that $C_N(t=0)=1$ and $C_N(t)=0$ if the configurations are not correlated. Because the configurations will become uncorrelated if the time $t$ between the configurations is sufficiently long, we have that $C_N(t) \to 0$ for $t \gg 1$. In general, we expect that $C(t) \sim e^{-t/\tau_c}$, that is, $C(t)$ decays exponentially with a decay or correlation time $\tau_c$. Estimate $\tau_c$ from a plot of $\ln C(t)$ versus $t$. Another way of estimating $\tau_c$ is from the integral $\int_0^\infty dt C(t)$. (Because we determine $C(t)$ at discrete values of $t$, this integral is actually a sum.) How do your two estimates of $\tau_c$ compare? A more detailed discussion of the estimation of correlation times can be found in Section ??.

Another type of random walk that is less constrained than the self-avoiding random walk is the “true” self-avoiding walk (TSAW). The TSAW describes the path of a random walker that avoids visiting a lattice site with a probability that is a function of the number of times the site has been visited already. This constraint leads to a reduced excluded volume interaction in comparison to the usual self-avoiding walk.

**Problem 12.14.** The true self-avoiding walk in one dimension

In one dimension the true self-avoiding walk corresponds to a walker who can jump to one of two nearest neighbors with a probability that depends on the number of times these neighbors already have been visited. Suppose that the walker is at site $i$ at step $t$. The probability that at time $t+1$, the walker will jump to site $i+1$ is given by

$$p_{i+1} = \frac{e^{-g n_{i+1}}}{e^{-g n_{i+1}} + e^{-g n_{i-1}}}, \quad (12.21)$$

where $n_{i\pm 1}$ is the number of times that the walker has already visited site $i \pm 1$. The probability of a jump to site $i-1$ is $p_{i-1} = 1 - p_{i+1}$. The parameter $g$ ($g > 0$) is a measure of the “desire” of the path to avoid itself. The first few steps of a typical true self-avoiding walk are shown in Figure 12.9. The main quantity of interest is the exponent $\nu$. We know that $g = 0$ corresponds to the usual random walk with $\nu = \frac{1}{2}$ and that the limit $g \to \infty$ corresponds to the self-avoiding walk. What is the value of $\nu$ for a self-avoiding walk in one dimension? Is the value of $\nu$ for any finite value of $g$ different than these two limiting cases?

Write a program to do a Monte Carlo simulation of the true self-avoiding walk in one dimension. Use an array to record the number of visits to every site. At each step calculate the probability $p$ of a jump to the right. Generate a random number $r$ and compare it to $p$. If $r \leq p$,
move the walker to the right; otherwise move the walker to the left. Compute $\langle \Delta x_N^2 \rangle$, where $x$ is the distance of the walker from the origin, as a function of the number of steps $N$. Make a log-log plot of $\langle \Delta x_N^2 \rangle$ versus $N$ and estimate $\nu$. Can you distinguish $\nu$ from its random walk and self-avoiding walk values? Reasonable choices of parameters are $g = 0.1$ and $N \sim 10^3$. Averages over $10^3$ trials give qualitative results. For comparison, published results (see Bernasconi and Pietronero) are for $N = 10^4$ and for $10^3$ trials; extended results for $g = 2$ are given for $N = 2 \times 10^5$ and $10^4$ trials.

### 12.4 Diffusion Controlled Chemical Reactions

Imagine a system containing particles of a single species $A$. The particles diffuse, and when two particles collide, a “reaction” occurs such that either one particle is annihilated or the two combine to form an inert species which is no longer involved in the reaction. In the latter case we can represent the chemical reaction as

$$A + A \rightarrow \text{(inert)}$$  \hspace{1cm} (12.22)

If we ignore the spatial fluctuations of the density of species $A$, we can describe the kinetics by a simple rate equation:

$$\frac{dA(t)}{dt} = -kA^2(t),$$  \hspace{1cm} (12.23)

where $A$ is the concentration of $A$ particles at time $t$ and $k$ is the rate constant. (In the chemical kinetics literature it is traditional to use the term concentration rather than the number density.) For simplicity, we assume that all reactants are entered into the system at $t = 0$ and that no
reactants are added later (a closed system). It is easy to show that the solution of the first-order differential equation (12.23) is

\[ A(t) = \frac{1}{kt + 1/A(0)}, \]

and \( A(t) \sim t^{-1} \) in the limit of long times.

Another interesting case is the bimolecular reaction

\[ A + B \rightarrow 0. \]

If we neglect fluctuations in the concentration as before (this neglect yields what is known as a mean-field approximation), we can write the corresponding rate equation as

\[ \frac{dA(t)}{dt} = \frac{dB(t)}{dt} = -kA(t)B(t). \]

We also have that

\[ A(t) - B(t) = \text{constant}, \]

because each reaction leaves the difference between the concentration of \( A \) and \( B \) particles unchanged. For the special case of equal initial concentrations, the solution of (12.26) with (12.27) is the same as (12.24). What is the solution for the case \( A(0) \neq B(0) \)?

The above derivation of the time dependence of \( A \) for the kinetics of the one and two species annihilation process is straightforward, but is based on the neglect of spatial fluctuations. In the following two problems, we simulate the kinetics of these processes and test this assumption.

**Problem 12.15.** Diffusion controlled chemical reactions in one dimension

a. Assume that \( N \) particles do a random walk on a one-dimensional lattice of length \( L \) with periodic boundary conditions. Every particle moves once in one unit of time. It is convenient to associate an array \( \text{site}(j) \) which records the label of the particle, if any, at site \( j \). Because we are interested in the long time behavior of the system when the concentration \( A = N/L \) of particles is small, it is efficient to also maintain an array of particle positions, \( \text{posx} \) such that \( \text{site}(\text{posx}(i)) = i \). For example, if particle 5 is located at site 12, then \( \text{posx}(5) = 12 \) and \( \text{site}(12) = 5 \). We also need an array, \( \text{newSite} \), to maintain the new positions of the walkers as they are moved one at a time. If two walkers land on the same position \( k \), then \( \text{newSite}(k) = 0 \), and the value of \( \text{posx} \) for these two walkers to 0. After all the walkers have moved, we let \( \text{site} = \text{ewSite} \) for all sites, and remove all the reacting particles in \( \text{posx} \) that have values equal to 0. This operation can be accomplished by replacing any reacting particle in \( \text{posx} \) by the last particle in the array. Begin with all sites occupied, \( A(t = 0) = 1 \).

Make a log-log plot of the quantity \( 1/A(t) - 1 \) versus the time \( t \). The times should be separated by exponential intervals so that your data is equally spaced on a logarithmic plot. For example, you might include data with times equal to \( 2^p \), with \( p = 1, 2, 3, \ldots \). Does your log-log plot yield a straight line in the limit of long times? If so, calculate its slope. Is the mean-field approximation for \( A(t) \) valid in one dimension? You can obtain crude results for small lattices of order \( L = 100 \) and times of order \( t = 10^2 \). To obtain results to within ten percent, you need lattices of order \( L = 10^4 \) and times of order \( t = 2^{13} \).
b. More insight into the origin of the time dependence of \( A(t) \) can be gained from the behavior of the quantity \( P(r, t) \), the probability distribution of the nearest neighbor distances at time \( t \). The nearest neighbor distance of a particle is defined as the minimum distance between the particle and all other particles. \( (P(r) \) is not the same as the radial distribution function \( g(r) \) considered in Section 8.8.) The distribution of these distances changes dramatically as the reaction proceeds, and this change can give information about the reaction mechanism. Place particles at random on a one-dimensional lattice and verify that the most probable nearest neighbor distance is \( r = 1 \) (one lattice constant) for all concentrations. (This result is true in any dimension.) Then verify that the distribution of nearest neighbor distances on a \( d = 1 \) lattice is given by

\[
P(r, t = 0) = 2A e^{-2A(r-1)}. \tag{12.28}
\]

Is the form (12.28) properly normalized? Start with \( A(t = 0) = 0.1 \) and find \( P(r, t) \) for \( t = 10, 100, \) and 1000. Average over all particles. How does \( P(r, t) \) change as the reaction proceeds? Does it retain the same form as the concentration decreases?

c. Compute the quantity \( D(t) \), the number of distinct sites visited by an individual walker. How does the time dependence of \( D(t) \) compare to the computed time dependence of \( 1/A(t) - 1 \)?

d. Write a program to simulate the \( A + B = 0 \) reaction. For simplicity, assume that multiple occupancy of the same site is not allowed, for example, an \( A \) particle cannot jump to a site already occupied by an \( A \) particle. The easiest procedure is to allow a walker to choose one of its nearest neighbor sites at random, but to not move the walker if the chosen site is already occupied by a particle of the same type. If the site is occupied by a walker of another type, then the pair of reacting particles is annihilated. Keep separate arrays for the \( A \) and \( B \) particles, with the value of the array denoting the label of the particle as before. An easy way to distinguish \( A \) and \( B \) walkers is to make the array element \( \text{site}(k) \) positive if the site is occupied by an \( A \) particle and negative if the site is occupied by a \( B \) particle. Start with equal concentrations of \( A \) and \( B \) particles and occupy the sites at random. Some of the interesting questions are similar to those that we posed in parts (a)–(c). Color code the particles and observe what happens to the relative positions of the particles.

*Problem 12.16. Reaction diffusion in two dimensions

a. Do a similar simulation as in Problem 12.15 on a two-dimensional lattice for the reaction \( A + A \rightarrow 0 \). In this case it is convenient to have one array for each dimension, for example, \( \text{posx} \) and \( \text{posy} \). Set \( A(t = 0) = 1 \), and choose \( L = 50 \). Show the configuration of walkers after each Monte Carlo step per walker. Describe the geometry of the clusters of particles as the diffusion process proceeds. Are the particles uniformly distributed throughout the lattice for all times? Calculate \( A(t) \) and compare your results for \( 1/A(t) - 1/A(0) \) to the \( t \)-dependence of \( D(t) \), the number of distinct lattice sites that are visited in time \( t \). \( (D(t) \sim t/\log t \) for two dimensions.) How well do the slopes compare? Do a similar simulation with \( A(t = 0) = 0.01 \). What slope do you obtain in this case? What can you conclude about the initial density dependence? Is the mean-field approximation valid in this case? If time permits, do a similar simulation in three dimensions.
b. Begin with $A$ and $B$ type random walkers initially segregated on the left and right halves (in the $x$ direction) of a square lattice. The process $A + B \rightarrow C$ exhibits a reaction front where the production of particles of type $C$ is nonzero. Some of the quantities of interest are the time dependence of the mean position $x(t)$ and the width $w(t)$ of the reaction front. The rules of this process are the same as in part (a) except that a particle of type $C$ is added to a site when a reaction occurs. A particular site can be occupied by one particle of type $A$ or type $B$ as well as any number of particles of type $C$. If $n(x,t)$ is the number of particles of type $C$ at a distance $x$ from the initial boundary of the reactants, then $x(t)$ and $w(t)$ can be written as
\begin{align}
x(t) &= \frac{\sum_x x n(x,t)}{\sum_x n(x,t)} \quad (12.29) \\
w^2(t) &= \frac{\sum_x [x - x(t)]^2 n(x,t)}{\sum_x n(x,t)}. \quad (12.30)
\end{align}
Choose lattice sizes of order $100 \times 100$, and average over at least 10 trials. The fluctuations in $x(t)$ and $w(t)$ can be reduced by averaging $n(x,t)$ over the order of 100 time units centered about $t$. More details can be found in Jiang and Ebner.

12.5 The Continuum Limit

In Chapter 7 we showed that the solution of the diffusion equation is equivalent to the long time behavior of a simple random walk on a lattice. In the following, we show directly that the continuum limit of the one-dimensional random walk model is a diffusion equation.

If there is an equal probability of taking a step to the right or left, the random walk can be written in terms of the simple “master” equation
\begin{equation}
P(i,N) = \frac{1}{2}[P(i+1,N-1) + P(i-1,N-1)], \quad (12.31)
\end{equation}
where $P(i,N)$ is the probability that the walker is at site $i$ after $N$ steps. To obtain a differential equation for the probability density $P(x,t)$, we identify $t = N\tau$, $x = ia$, and $P(i,N) = aP(x,t)$, where $\tau$ is the time between steps and $a$ is the lattice spacing. This association allows us to rewrite (12.31) in the equivalent form
\begin{equation}
P(x,t) = \frac{1}{2}[P(x+a,t-\tau) + P(x-a,t-\tau)]. \quad (12.32)
\end{equation}
We rewrite (12.32) by subtracting $P(x,t-\tau)$ from both sides of (12.32) and dividing by $\tau$:
\begin{equation}
\frac{1}{\tau} [P(x,t) - P(x,t-\tau)] = \frac{a^2}{2\tau} \left[ P(x+a,t-\tau) - 2P(x,t-\tau) + P(x-a,t-\tau) \right] a^{-2}.
\end{equation}
If we expand $P(x,t-\tau)$ and $P(x \pm a,t-\tau)$ in a Taylor series and take the limit $a \to 0$ and $\tau \to 0$ with the ratio $D \equiv a^2/2\tau$ finite, we obtain the diffusion equation
\begin{equation}
\frac{\partial P(x,t)}{\partial t} = D \frac{\partial^2 P(x,t)}{\partial x^2}.
\end{equation}
(12.34a)
The generalization of (12.34a) to three dimensions is
\[ \frac{\partial P(x, y, z, t)}{\partial t} = D \nabla^2 P(x, y, z, t). \]  

(12.34b)

where \( \nabla^2 = \partial^2 / \partial x^2 + \partial^2 / \partial y^2 + \partial^2 / \partial z^2 \) is the Laplacian operator. Equation (12.34) is known as the \textit{diffusion} equation and is frequently used to describe the dynamics of fluid molecules.

The numerical solution of the prototypical \textit{parabolic} partial differential equation (12.34) is a nontrivial problem in numerical analysis (cf. Press et al. or Koonin and Meredith.) An indirect method of solving (12.34) numerically is to use a Monte Carlo method, that is, replace (12.34) by a corresponding random walk on a lattice with discrete time steps. Because the asymptotic behavior of the partial differential equation and the random walk model are equivalent, this approach uses the Monte Carlo technique as a method of \textit{numerical analysis}. In contrast, if our goal is to understand a random walk lattice model directly, the Monte Carlo technique is a \textit{simulation} method. The difference between simulation and numerical analysis is sometimes in the eyes of the beholder.

\textit{Problem 12.17.} Biased random walk

Show that the form of the differential equation satisfied by \( P(x, t) \) corresponding to a random walk with a drift, that is, a walk for \( p \neq q \), is

\[ \frac{\partial P(x, t)}{\partial t} = D \nabla^2 P(x, y, z, t) - v \frac{\partial P(x, t)}{\partial x}. \]  

(12.35)

How is \( v \) related to \( p \) and \( q \)?

\section{12.6 Random Number Sequences}

So far we have used the random number generator supplied with Java to generate the desired random numbers for our Monte Carlo applications. In principle, we might have generated these numbers from a random physical process, such as the decay of radioactive nuclei or the thermal noise from a semiconductor device. In practice, random number sequences are generated from a physical process only for specific purposes such as a lottery. Although we can store the outcome of a random physical process so that the random number sequence would be both truly random and reproducible, such a method would be inconvenient and inefficient in part because we often require very long sequences. In fact, there are companies that market CD-ROMs and DVDs that contain many millions of random numbers. In practice we use a digital computer, a deterministic machine, to generate sequences of random numbers. These sequences cannot be truly random and are sometimes referred to as \textit{pseudorandom}. However, such a distinction is not important if the sequence satisfies all our criteria for randomness.

Most random number generators yield a sequence in which each number is used to find the succeeding one according to a well defined algorithm. The most important features of a desirable random number generator are that its sequence satisfies the known statistical tests for randomness, the probability distribution is uniform, the sequence has a long period, the method is efficient, the sequence is reproducible, and the algorithm is machine independent.

The most widely used random number generator is based on the \textit{linear congruential} method. That is, given the seed \( x_0 \), each number in the sequence is determined by the one-dimensional map

\[ x_n = (ax_{n-1} + c) \mod m, \]  

(12.36)
where $a$, $c$, and $m$ as well as $x_n$ are integers. The notation $y = z \mod m$ means that $m$ is subtracted from $z$ until $0 \leq y < m$. The map (12.36) is characterized by three parameters, the multiplier $a$, the increment $c$, and the modulus $m$. Because $m$ is the largest integer generated by (12.36), the maximum possible period is $m$.

In general, the period depends on all three parameters. For example, if $a = 3$, $c = 4$, $m = 32$, and $x_0 = 1$, the sequence generated by (12.36) is $1, 7, 25, 15, 17, 23, 9, 31, 1, 7, 25, \ldots$ and the period is 8 rather than the maximum possible value of 32. If we are careful to choose $a$, $c$, and $m$ such that the maximum period is obtained, then all possible integers between 0 and $m-1$ will occur in the sequence. Because we usually wish to have random numbers $r$ in the unit interval $0 \leq r < 1$ rather than random integers, random number generators usually return the ratio $x_n/m$ which is always less than unity. One advantage of the linear congruential method is that it is very fast. As the above example illustrates, $m$, $a$, and $c$ must be chosen carefully to achieve optimum results. Several rules have been developed (see Knuth) to obtain the longest period. Some of the properties of the linear congruential method are explored in Problem 12.18.

Another popular random number generator is the generalized feedback shift register method which uses bit manipulation. Every integer is represented as a series of 1’s and 0’s called bits. These bits can be shuffled by using the bitwise exclusive or operator $\oplus$ defined by $a \oplus b = 1$ if the bits $a \neq b$; $a \oplus b = 0$ if $a = b$. The $n$th member of the sequence is given by

$$x_n = x_{n-p} \oplus x_{n-q},$$

(12.37)

where $p > q$, and $p$, $q$ and $x_n$ are integers. The first $p$ random numbers must be supplied by another random number generator. As an example of how the exclusive or operator works, suppose that $n = 6, p = 5, q = 3, x_3 = 11$, and $x_1 = 6$. Then $x_6 = x_1 \oplus x_3 = 0110 \oplus 1011 = 1101 = 2^3 + 2^2 + 2^0 = 8 + 4 + 1 = 13$. Not all values of $p$ and $q$ lead to good results. Some common pairs are $(p, q) = (31, 3), (250, 103), \text{and} (521, 168)$. In Fortran the exclusive or operation on the integers $m$ and $n$ is given by the intrinsic function $\text{ieor}(m, n)$; the equivalent operation in C and Java is given by $m \land n$.

The above two examples of random number generators illustrate their general nature. That is, numbers in the sequence are used to find the succeeding ones according to a well defined algorithm, and the sequence is determined by the seed, the first number of the sequence (or the first $p$ members of the sequence for the generalized feedback shift register method. In general, the maximum possible period is related to the size of the computer word, for example, 16, 32, or 64 bits, that is used. We also note that the choice of the constants and the proper initialization of the sequence is very important and that the algorithm must be implemented with care.

There is no necessary and sufficient test for the randomness of a finite sequence of numbers; the most that can be said about any finite sequence of numbers is that it is apparently random. Because no single statistical test is a reliable indicator, we need to consider several tests. Some of the best known tests are discussed in Problem 12.18. Many of these tests can be stated in terms of random walks.

**Problem 12.18.** Statistical tests of randomness

**a. Period.** The most obvious requirement for a random number generator is that its period be much greater than the number of random numbers needed in a specific calculation. One way to visualize the period of the random number generator is to use it to generate a plot of
CHAPTER 12. RANDOM WALKS

the displacement \( x \) of a random walker as a function of the number of steps \( N \). When the period of the random number is reached, the plot will begin to repeat itself. Generate such a plot using (12.36) for the two choices of parameters \( a = 899, c = 0, m = 32768, \) and \( a = 16807, c = 0, m = 2^{31} - 1 \) with \( x_0 = 12 \). What are the periods of the corresponding random number generators? Obtain similar plots using different values for the parameters \( a, c, \) and \( m \). Why is the seed value \( x_0 = 0 \) forbidden for this choice of \( c \)? Do some combinations of these parameters give longer periods than others? What is the period of the random number generator supplied with the random number generator that you are using?

b. Uniformity. A random number sequence should contain numbers distributed in the unit interval with equal probability. The simplest test of uniformity is to divide this interval into \( M \) equal size subintervals or bins and place each member of the sequence into one of the bins. For example, consider the first \( N = 10^4 \) numbers generated by (12.36) with \( a = 106, c = 1283, \) and \( m = 6075 \) (see Press et al.). Place each number into one of \( M = 100 \) bins. Is the number of entries in each bin approximately equal? What happens if you increase \( N \)?

c. Chi-square test. Is the distribution of numbers in the bins of part (b) consistent with the laws of statistics? The most common test of this consistency is the chi-square or \( \chi^2 \) test. Let \( y_i \) be the observed number in bin \( i \) and \( E_i \) be the expected value. For the example in part (b) with \( N = 10^4 \) and \( M = 100 \), we have \( E_i = 100 \). The chi-square statistic is

\[
\chi^2 = \sum_{i=1}^{M} \frac{(y_i - E_i)^2}{E_i}. \tag{12.38}
\]

The magnitude of the number \( \chi^2 \) is a measure of the agreement between the observed and expected distributions. In general, the individual terms in the sum (12.38) are expected to be order 1, and because there are \( M \) terms in the sum, we expect \( \chi^2 \leq M \). As an example, we did five independent runs of a random number generator with \( N = 10^4 \) and \( M = 100 \), and found \( \chi^2 \approx 92, 124, 85, 91, \) and 99. These values of \( \chi^2 \) are consistent with this expectation. Although we usually want \( \chi^2 \) to be as small as possible, we would be suspicious if \( \chi^2 \approx 0 \), because such a small value suggests that \( N \) is a multiple of the period of the generator and that each value in the sequence appears an equal number of times.

A more quantitative measure of our confidence that the discrepancy \( (y_i - E_i) \) is distributed according to the Gaussian distribution is given by the chi-square probability function \( P(x, \nu) \) defined as:

\[
P(x, \nu) = \frac{1}{2^{\nu/2}\Gamma(\nu/2)} \int_0^x t^{(\nu-2)/2} e^{-t/2} dt. \tag{12.39}
\]

The Gamma function \( \Gamma(z) \) in (12.39) is given by \( \Gamma(z) = \int_0^\infty t^{z-1} e^{-t} dt \); the familiar relation \( \Gamma(z+1) = z! \) holds if \( z \) is a positive integer. The quantity \( \nu \) in (12.39) is the number of degrees of freedom. In our case \( \nu = M - 1 \), because we have imposed the constraint that \( \sum_{i=1}^{M} E_i = N \). The function \( Q(x, \nu) = 1 - P(x, \nu) \) is the probability that the measured value of \( \chi^2 \) is greater than \( x \). For our example we can solve for \( x \) in the equation \( Q(x, \nu) = q \) with \( \nu = 99 \) for various values of \( q \) or find the solution in a statistical table. (A quick search of the Web will yield several sites that will compute \( x \) for particular values of \( \nu \) and probability \( q \).) For \( \nu = 99 \), we find that \( x \approx 139 \) for \( q = 0.005 \), \( x \approx 123 \) for \( q = 0.05 \), \( x \approx 111 \) for \( q = 0.2 \), and \( x \approx 98 \) for \( q = 0.5 \).
CHAPTER 12. RANDOM WALKS

Our above results for $\chi^2$ show that $\chi^2 > 123$ for one run out of five (20%). Because 123 is the value of $x$ at the 5% level, we expect to see $\chi^2 \geq 123$ in only one out of twenty runs. Hence, our confidence level is less than 95%. Instead, we can assume an approximately 80% confidence level in our random number generator because the value of $x$ for this confidence level is 111. We might be able to increase our confidence level by doing more runs. Suppose that we make twenty runs and we still find only one measurement of $\chi^2$ greater than 123. In this case our confidence level would rise to 95%. Determine $\chi^2$ for twenty independent runs using the values of $a$, $c$, and $m$ given in parts (12.18) and (b). Estimate your level of confidence in these random number generators.

d. **Filling sites.** Although a random number sequence might contain numbers that are distributed in the unit interval with equal probability, consecutive numbers might not appear in a perfectly uniform way, but have a tendency to be clumped or correlated in some way. One test of this correlation is to fill a square lattice of $L^2$ sites at random. Consider an array $n(x, y)$ that is initially empty, where $1 \leq x_i, y_i \leq L$. A point is selected randomly by choosing its two coordinates $x_i$ and $y_i$ from two consecutive numbers in the sequence. If the site is empty, it is filled and $n(x_i, y_i) = 1$; otherwise it is not changed. This procedure is repeated $tL^2$ times, where $t$ is the number of Monte Carlo steps per site. Because this process is analogous to the decay of radioactive nuclei, we expect that the fraction of empty lattice sites should decay as $e^{-t}$. Determine the fraction of unfilled sites using the random number generator that you have been using for $L = 10$, $15$, and $20$. Are your results consistent with the expected fraction? Repeat the same test using (12.36) with $a = 65549$, $c = 0$, and $m = 231$. The existence of triplet correlations can be determined by a similar test on a simple cubic lattice by choosing the three coordinates $x_i, y_i,$ and $z_i$ from three consecutive random numbers.

e. **Hidden correlations.** Sometimes a picture is worth a thousand numbers. Another way of checking for correlations is to plot $x_{i+k}$ versus $x_i$. If there are any obvious patterns in the plot, then there is something wrong with the generator. Use the generator (12.36) with $a = 16807$, $c = 0$, and $m = 2^{31} - 1$. Can you detect any structure in the plotted points for $k = 1$ to $k = 5$? Also test the random number generator that you have been using. Do you see any evidence of lattice structure, for example, equidistant parallel lines? Is the logistic map $x_{n+1} = 4x_n(1 - x_n)$ for $r = 1$ a suitable random number generator?

f. **Short-term correlations.** Another measure of short term correlations is the autocorrelation function

$$C(k) = \frac{\langle x_{i+k}x_i \rangle - \langle x_i \rangle^2}{\langle x_i^2 \rangle - \langle x_i \rangle^2},$$

(12.40)

where $x_i$ is the $i$th term in the sequence. We have used the fact that $\langle x_{i+k} \rangle = \langle x_i \rangle$, that is, the choice of the origin of the sequence is irrelevant. The quantity $\langle x_{i+k}x_i \rangle$ is found for a particular choice of $k$ by forming all the possible products of $x_{i+k}x_i$ and dividing by the number of products. If $x_{i+k}$ and $x_i$ are not correlated, then $\langle x_{i+k}x_i \rangle = \langle x_{i+k} \rangle \langle x_i \rangle$ and $C(k) = 0$. Is $C(k)$ identically zero for any finite sequence? Compute $C(k)$ for $a = 106$, $c = 1283$, and $m = 6075$.

g. **Random walk.** Another test based on the properties of random walks has been proposed (see Vattulainen et al.). Assume that a walker begins at the origin of the $x$-$y$ plane and generate $n_w$ walks of $N$ steps. Count the number of walks in each quadrant $q_i$ of the $x$-$y$ plane, and use the
\( \chi^2 \) test (12.38) with \( y_i \rightarrow q_i \), \( M = 4 \), and \( E_i = n_w/4 \). If \( \chi^2 > 7.815 \) (a 5\% probability if the random number generator is perfect), we say that the run fails. The random number generator fails if two out of three independent runs fail. The probability of a perfect generator failing two out of three runs would be approximately \( 3 \times 0.95 \times (0.05)^2 \approx 0.007 \). Test several random number generators.

*Problem 12.19. An “improved” random number generator

One way to reduce sequential correlation and to lengthen the period is to mix or shuffle two different random number generators. The following procedure illustrates the approach for two random number generators that we denote as \texttt{ran1} and \texttt{ran2}.

1. Make a list of 256 random numbers using \texttt{ran1}. (The number 256 is arbitrary, but should be less than the period of \texttt{ran1}.)

2. Choose a random number \( x \) from this list by using \texttt{ran2} to generate a random index between 1 and 256. The integer \( x \) is the desired number.

3. Replace the number chosen in step 2 by a new random number generated by \texttt{ran1}.

Consider two random number generators with relatively short periods and strong sequential correlation and show that the above shuffling scheme improves the quality of the random numbers.

At least some of the statistical tests given in Problem 12.18 should be done whenever serious calculations are contemplated. However, even if a random number generator passes all these tests, there still can be problems in rare cases. Typically, these problems arise when a small number of events have a large weight. In these cases a very small bias in the random number generator might lead to a systematic error in the final results, and two generators, which appear equally good as determined by various statistical tests, might give statistically different results when used in a specific application. For this reason, it is important that the random number generator that is used be reported along with the actual results. Confidence in the results also can be increased by repeating the calculation with another random number generator.

Because all random number generators are based on a deterministic algorithm, it always is possible to construct a test generator for which a particular algorithm will fail. The success of a random number generator in passing various statistical tests is necessary and improves our overall confidence in its statistical properties, but it is not a sufficient condition for their use in all applications. In Project ?? we discuss an application of Monte Carlo methods to the Ising model for which some commonly used random number generators give incorrect results.

### 12.7 Projects

Almost all of the problems in this chapter can be done using more efficient programs, greater number of trials, and larger systems. More applications of random walks and random number sequences are discussed in subsequent chapters. Many more ideas for projects can be gained from the references.
Figure 12.10: Examples of the first several changes generated by the pivot algorithm for a self-avoiding walk of $N = 10$ steps (11 sites). The open circle denotes the pivot point. This figure is adopted from the article by MacDonald et al.

Project 12.20. Application of the pivot algorithm

The algorithms that we have discussed for generating self-avoiding random walks are all based on making local deformations of the walk (polymer chain) for a given value of $N$, the number of bonds. As discussed in Problem 12.13, the time $\tau_c$ between statistically independent configurations is nonzero. The problem is that $\tau_c$ increases with $N$ as some power, for example, $\tau_c \sim N^3$. This power law dependence of $\tau_c$ on $N$ is called critical slowing down and implies that it becomes increasingly more time consuming to generate long walks. We now discuss an example of a global algorithm that reduces the dependence of $\tau_c$ on $N$. Another example of a global algorithm that reduces critical slowing down is discussed in Project ??.

a. Consider the walk shown in Figure 12.10a. Select a site at random and one of the four possible directions. The shorter portion of the walk is rotated (pivoted) to this new direction by treating the walk as a rigid structure. The new walk is accepted only if the new walk is self-avoiding; otherwise the old walk is retained. (The shorter portion of the walk is chosen to save computer time.) Some typical moves are shown in Figure 12.10. Note that if an end point is chosen, the previous walk is retained. Write a program to implement this algorithm and compute the dependence of the mean square end-to-end distance $\langle R^2_N \rangle$ on $N$. Consider values of $N$ in the range $10 \leq N \leq 80$. A discussion of the results and the implementation of the algorithm can be found in MacDonald et al. and Madras and Sokal, respectively.

b. Compute the correlation time $\tau_c$ for different values of $N$ using the approach discussed in Problem 12.13c.

Project 12.21. A simple reaction diffusion model
In Problem 12.16 we saw that simple patterns can develop as a result of random behavior. The phenomenon of pattern formation is of much interest in a variety of contexts ranging from the large scale structure of the universe to the roll patterns seen in convection (for example, smoke rings). In the following, we explore the patterns that can develop in a simple reaction diffusion model based on the reactions, $A + 2B \rightarrow 3B$, and $B \rightarrow C$, where $C$ is inert. Such a reaction is called autocatalytic. In Problem 12.16 we considered chemical reactions in a closed system where the reactions can proceed to equilibrium. In contrast, open systems allow a continuous supply of fresh reactants and a removal of products. These two processes allow steady states to be realized and oscillatory conditions to be maintained indefinitely. The model of interest also assumes that $A$ is added at a constant rate and that both $A$ and $B$ are removed by the feed process. Pearson (see references) models these processes by two coupled reaction diffusion equations:

\[
\begin{align*}
\frac{\partial A}{\partial t} &= D_A \nabla^2 A - AB^2 + f(1 - A) \quad (12.41a) \\
\frac{\partial B}{\partial t} &= D_B \nabla^2 B + AB^2 - (f + k)B. \quad (12.41b)
\end{align*}
\]

The $AB^2$ term represents the reaction $A + 2B \rightarrow 3B$. This term is negative in (12.41a) because the reactant $A$ decreases, and is positive in (12.41b) because the reactant $B$ increases. The term $+f$ represents the constant addition of $A$, and the terms $-fA$ and $-fB$ represent the removal process; the term $-kB$ represents the reaction $B \rightarrow C$. All the quantities in (12.41) are dimensionless. We assume that the diffusion coefficients are $D_A = 2 \times 10^{-5}$ and $D_B = 10^{-5}$, and the behavior of the system is determined by the values of the rate constant $k$ and the feed rate $f$.

a. We first consider the behavior of the reaction kinetics that results when the diffusion terms

Figure 12.11: Evolution of the pattern starting from the initial conditions suggested in Project c.
in (12.41) are neglected. It is clear from (12.41) that there is a trivial steady state solution \( A = 1, B = 0 \). Are there other solutions, and if so, are they stable? The steady state solutions can easily be found by solving (12.41) with \( \partial A / \partial t = \partial B / \partial t = 0 \). To determine the stability, we can add a perturbation and determine whether the perturbation grows or not. However, without the diffusion terms, it is more straightforward to solve (12.41) numerically using a simple Euler algorithm. Choose a time step equal to unity, and let \( A = 0.1 \) and \( B = 0.5 \) at \( t = 0 \). Determine the steady state values for \( 0 < f \leq 0.3 \) and \( 0 < k \leq 0.07 \) in increments of \( \Delta f = 0.02 \) and \( \Delta k = 0.005 \). Record the steady state values of \( A \) and \( B \). Then repeat this exercise for the initial values \( A = 0.5 \) and \( B = 0.1 \). You should find that for some values of \( f \) and \( k \), only one steady state solution can be obtained for the two initial conditions, and for other initial values of \( A \) and \( B \) there are two steady state solutions. Try other initial conditions. If you obtain a new solution, change the initial \( A \) or \( B \) slightly to see if your new solution is stable. On an \( f \) versus \( k \) plot indicate where there are two solutions and where there are one. In this way you can determine the approximate phase diagram for this process.

b. There is a small region in \( f \)-\( k \) space where one of the steady state solutions becomes unstable and periodic solutions occur (the mechanism is known as a Hopf bifurcation). Try \( f = 0.009 \), \( k = 0.03 \), and set \( A = 0.1 \) and \( B = 0.5 \) at \( t = 0 \). Plot the values of \( A \) and \( B \) versus the time \( t \). Are they periodic? Try other values of \( f \) and \( k \) and estimate where the periodic solutions occur.

c. Numerical solutions of the full equation with diffusion (12.41) can be found by making a finite difference approximation to the spatial derivatives as in (3.19) and using a simple Euler algorithm for the time integration. Adopt periodic boundary conditions. Although it is straightforward to write a program to do the numerical integration, an exploration of the dynamics of this system requires a supercomputer. However, we can find some preliminary results with a small system and a coarse grid. Consider a \( 0.5 \times 0.5 \) system with a spatial mesh of \( 128 \times 128 \) grid points on a square lattice. Choose \( f = 0.18 \), \( k = 0.057 \), and \( \Delta t = 0.1 \). Let the entire system be in the initial trivial state \((A = 1, B = 0)\) except for a \( 20 \times 20 \) grid located at the center of the system where the sites are \( A = 1/2, B = 1/4 \) with a \( \pm 1\% \) random noise. The effect of the noise is to break the square symmetry. Let the system evolve for approximately 80,000 time steps and look at the patterns that develop. Color code the grid according to the concentration of \( A \), with red representing \( A = 1 \) and blue representing \( A \approx 0.2 \) and with several intermediate colors. Very interesting patterns have been found by Pearson.

References and Suggestions for Further Reading


J. Bernasconi and L. Pietronero, “True self-avoiding walk in one dimension,” Phys. Rev. B 29, 5196 (1984). The authors present results for the exponent \( \nu \) accurate to 1%.


CHAPTER 12. RANDOM WALKS


F. Reif, Fundamentals of Statistical and Thermal Physics, McGraw-Hill (1965). This well known text on statistical physics has a good discussion on random walks (Chapter 1) and diffusion (Chapter 12).


