Chapter 2

Naïve reconstructions and inverse crimes

This book is about developing computational solution methods for real-life inverse problems. The design of reconstruction algorithms is best done by first testing the code extensively with simulated data because every new aspect of the code can be systematically tested. Working directly with measured data may lead to very hard debugging problems, as the source of difficulties can be hard to track.

What happens if proper simulation of errors is neglected? For example, using the same computational grid for the data simulation and reconstruction sometimes results in perfect reconstructions from noise-free data. Such a situation is not realistic and is referred to as an *inverse crime*. Excellent inversion results may be obtained, but these are not representative of any realistic inverse problem, since noise is present in any experimental setting. Such studies are inconclusive at best since robustness against modeling and measurement errors is not tested.

In this chapter, we will introduce these concepts in the context of the three guiding examples in Part I: deconvolution, the backward heat equation, and X-ray tomography.

2.1 Convolution

Linear convolution is a useful process for modeling a variety of practical measurements. *Deconvolution*, the corresponding inverse problem, is related to many engineering problems such as removing unwanted echoes from sound recordings or sharpening a misfocused photograph.

One-dimensional deconvolution will serve as a basic example throughout Part I of the book. Two-dimensional deconvolution is a project topic in Section 10.

2.1.1 Continuum model for one-dimensional convolution

We build a computational model for one-dimensional convolution with periodic boundary conditions. We consider 1-periodic functions $f : \mathbb{R} \to \mathbb{R}$ satisfying $f(x) = f(x+n)$ with any integer $n \in \mathbb{Z}$. Essentially the function f is defined on an interval of length 1 such as $[0,1]$ or $[-\frac{1}{2}, \frac{1}{2}]$ with the endpoints identified; another way of thinking about this is to consider $f(x)$ defined on a circle with radius $(2\pi)^{-1}$ and x being the arc length variable.

The reason for considering periodic functions is that we can avoid some technicalities related to boundary conditions that would obscure the main message about ill-posedness. Also, the Fourier transform and the wavelet transform are easily defined and implemented in the periodic setting.

The continuum measurement model concerns a 1-periodic signal $f : \mathbb{R} \to \mathbb{R}$ blurred by a 1-periodic *point spread function* ψ. Other common names for the point spread function include *device function, impulse response, blurring kernel, convolution kernel*, and *transfer function*.

Let us first construct the point spread function using a building block ψ_0 defined in the interval $[-a,a] \subset \mathbb{R}$ with some constant $0 < a < 1/2$:

$$\psi_0(x) = C_a(x+a)^2(x-a)^2 \quad \text{for } -a \le x \le a, \tag{2.1}$$

where the constant $C_a := (\int_{-a}^{a}(x+a)^2(x-a)^2 dx)^{-1}$ is chosen to enforce the following normalization:

$$\int_{-a}^{a} \psi_0(x)\,dx = 1. \tag{2.2}$$

The periodic point spread function is defined by copying $\psi_0(x)$ to every interval $[n-a,n+a]$ with $n \in \mathbb{Z}$ and setting $\psi(x)$ to zero outside those intervals. The resulting ψ is a non-negative and even function:

$$\psi(x) \ge 0 \quad \text{and} \quad \psi(x) = \psi(-x) \quad \text{for all } x \in \mathbb{R}. \tag{2.3}$$

See Figure 2.1 for a plot of the point spread function with $a = 0.04$.

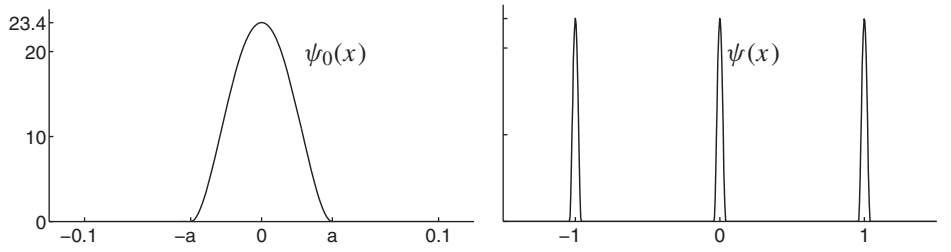

Figure 2.1. Point spread function according to (2.4) with $a = 0.04$ for one-dimensional convolution. Left: the continuously differentiable building block $\psi_0(x)$ used for constructing the periodic point spread function. Right: the periodic point spread function $\psi(x)$.

We remark that instead of (2.2) one often requires $\int_{-a}^{a} \psi_0(x)^2\,dx = 1$. However, we prefer (2.2) since then constant functions remain unchanged in convolution with ψ; this will be convenient below when we compare plots of reconstructions to the plot of the true signal by showing them in the same figure.

Definition 2.1.1. *The continuum model of convolution, or blurring, is given by the following integral:*

$$(\psi * f)(x) = \int_{-a}^{a} \psi(x')f(x-x')\,dx'. \tag{2.4}$$

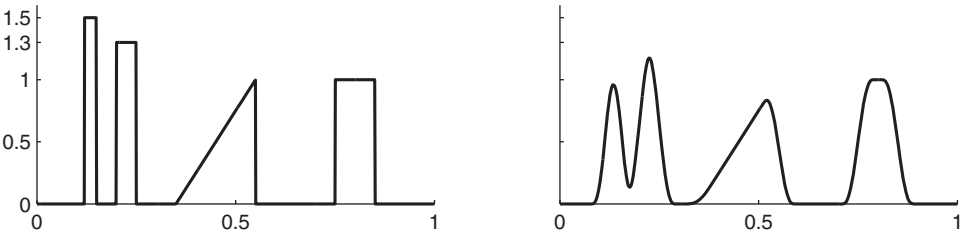

Figure 2.2. Effect of convolution on a piecewise continuous function. Left: target function $f(x)$. Right: the function $(\psi * f)(x)$.

An example of the effect of convolution with the point spread function is found in Figure 2.2. The smoothing effect of the convolution is evident, and motivates the terminology blurring kernel for the point spread function.

Note that formula (2.4) is not of the form (1.1) since the left-hand side is not a k-dimensional vector. However, suppose the function f is defined on an interval $[b, b+1]$, and assume that we have a device that measures the values of the convolution function $(\psi * f)(x)$ at a collection of k equally spaced points $\tilde{x}_1 = b, \tilde{x}_2 = b + \frac{1}{k}, \tilde{x}_3 = b + \frac{2}{k}, \ldots, \tilde{x}_k = b + \frac{k-1}{k}$ and define

$$\mathbf{m} := [(\psi * f)(\tilde{x}_1), (\psi * f)(\tilde{x}_2), \ldots, (\psi * f)(\tilde{x}_k)]^T \in \mathbb{R}^k. \tag{2.5}$$

Then $\mathcal{A}f = \mathbf{m}$ is of the form (1.1).

2.1.2 Discrete convolution model

Next we need to discretize the continuum model to arrive at a finite-dimensional measurement model of the form (1.3). Define

$$x_j = b + \frac{j-1}{n} \quad \text{for } j = 1, 2 \ldots, n; \tag{2.6}$$

then the 1-periodic real-valued function $f(x)$ is represented by a vector $\mathbf{f}$ containing values at the grid points:

$$\mathbf{f} = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n]^T = [f(x_1), f(x_2), \ldots, f(x_n)]^T \in \mathbb{R}^n. \tag{2.7}$$

Furthermore, denote $\Delta x := x_2 - x_1 = 1/n$.

We can approximate the integral appearing in (2.4) by numerical quadrature. For any reasonably well-behaved function $g : [b, b+1] \to \mathbb{R}$ we have

$$\int_b^{b+1} g(x)\, dx \approx \Delta x \sum_{j=1}^{n} g(x_j), \tag{2.8}$$

the approximation becoming better as n increases.

For convenience, let us take $k = n$ and measure the convolution at the same points (2.6) as where the unknown function f is sampled. This is not necessary in general, but

it will lead to a square-shaped matrix A, making it easy to illustrate naïve reconstructions and inverse crimes.

Let us construct an $n \times n$ matrix A so that $A\mathbf{f} \in \mathbb{R}^k$ approximates $\mathcal{A}f$ defined by (2.4). We define a discrete point spread function denoted by

$$\mathbf{p} = [\mathbf{p}_{-\nu}, \mathbf{p}_{-\nu+1}, \ldots, \mathbf{p}_{-1}, \mathbf{p}_0, \mathbf{p}_1, \ldots, \mathbf{p}_{\nu-1}, \mathbf{p}_\nu]^T$$

as follows. Recall that $\psi_0(x) \equiv 0$ for $|x| > a > 0$. Take $\nu > 0$ to be the smallest integer satisfying the inequality $(\nu+1)\Delta x > a$ and set

$$\widetilde{\mathbf{p}}_j = \psi_0(j\Delta x) \quad \text{for } j = -\nu, \ldots, \nu.$$

For example, with $a = 0.04$ as in Figure 2.1 and $n = 64$, we get $\nu = 2$. By (2.8) the normalization condition (2.2) almost holds: $\Delta x \sum_{j=-\nu}^{\nu} \widetilde{p}_j \approx 1$. However, in practice it is a good idea to normalize the discrete point spread function explicitly by the formula

$$\mathbf{p} = \Big(\Delta x \sum_{j=-\nu}^{\nu} \widetilde{\mathbf{p}}_j\Big)^{-1} \widetilde{\mathbf{p}}; \tag{2.9}$$

then it follows that

$$\Delta x \sum_{j=-\nu}^{\nu} \mathbf{p}_j = 1. \tag{2.10}$$

Now

$$\begin{aligned}\int_{-a}^{a} \psi(x')f(x_j - x')dx' &\approx \Delta x \sum_{\ell=-\nu}^{\nu} \psi(x_\ell) f(x_j - x_\ell) \\ &\approx \Delta x \sum_{\ell=-\nu}^{\nu} \mathbf{p}_\ell \mathbf{f}_{j-\ell}.\end{aligned}$$

Hence discrete convolution is defined by the formula

$$(\mathbf{p} * \mathbf{f})_j = \sum_{\ell=-\nu}^{\nu} \mathbf{p}_\ell \mathbf{f}_{j-\ell}, \tag{2.11}$$

where $\mathbf{f}_{j-\ell}$ is defined using periodic boundary conditions for the cases $j - \ell < 1$ and $j - \ell > n$. Then

$$\Delta x(\mathbf{p} * \mathbf{f}) \approx \mathcal{A}f, \tag{2.12}$$

and we define the measurement vector $\mathbf{m} = [\mathbf{m}_1, \ldots, \mathbf{m}_k]^T$ by

$$\mathbf{m}_j = \Delta x(\mathbf{p} * \mathbf{f})_j + \varepsilon_j. \tag{2.13}$$

We would like to write formula (2.13) using a matrix A so that we would arrive at the desired model (1.3). To this end, set

$$\begin{bmatrix} \mathbf{m}_1 \\ \vdots \\ \mathbf{m}_k \end{bmatrix} = \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{k1} & \cdots & a_{kn} \end{bmatrix} \begin{bmatrix} \mathbf{f}_1 \\ \vdots \\ \mathbf{f}_n \end{bmatrix} + \begin{bmatrix} \varepsilon_1 \\ \vdots \\ \varepsilon_k \end{bmatrix}.$$

The answer is to build a circulant matrix having the elements of $\mathbf{p}$ appearing systematically on every row of A.

Let us illustrate the structure of the convolution matrix A by an example in the case $n = 64$. As observed above, if $a = 0.04$, then $\nu = 2$, and the point spread function takes the form $p = [p_{-2} \quad p_{-1} \quad p_0 \quad p_1 \quad p_2]^T$. According to (2.11) we have

$$\begin{aligned}
(\mathbf{p} * \mathbf{f})_1 &= \mathbf{p}_0\mathbf{f}_1 + \mathbf{p}_{-1}\mathbf{f}_2 + \mathbf{p}_{-2}\mathbf{f}_3 + \mathbf{p}_2\mathbf{f}_{n-1} + \mathbf{p}_1\mathbf{f}_n,\\
(\mathbf{p} * \mathbf{f})_2 &= \mathbf{p}_1\mathbf{f}_1 + \mathbf{p}_0\mathbf{f}_2 + \mathbf{p}_{-1}\mathbf{f}_3 + \mathbf{p}_{-2}\mathbf{f}_4 + \mathbf{p}_2\mathbf{f}_n,\\
(\mathbf{p} * \mathbf{f})_3 &= \mathbf{p}_2\mathbf{f}_1 + \mathbf{p}_1\mathbf{f}_2 + \mathbf{p}_0\mathbf{f}_3 + \mathbf{p}_{-1}\mathbf{f}_4 + \mathbf{p}_{-2}\mathbf{f}_5,\\
&\vdots\\
(\mathbf{p} * \mathbf{f})_n &= \mathbf{p}_{-1}\mathbf{f}_1 + \mathbf{p}_{-2}\mathbf{f}_2 + \mathbf{p}_2\mathbf{f}_{n-2} + \mathbf{p}_1\mathbf{f}_{n-1} + \mathbf{p}_0\mathbf{f}_n.
\end{aligned}$$

Consequently the matrix A looks like this:

$$A = \Delta x \begin{bmatrix}
\mathbf{p}_0 & \mathbf{p}_{-1} & \mathbf{p}_{-2} & 0 & 0 & 0 & \cdots & \mathbf{p}_2 & \mathbf{p}_1\\
\mathbf{p}_1 & \mathbf{p}_0 & \mathbf{p}_{-1} & \mathbf{p}_{-2} & 0 & 0 & \cdots & 0 & \mathbf{p}_2\\
\mathbf{p}_2 & \mathbf{p}_1 & \mathbf{p}_0 & \mathbf{p}_{-1} & \mathbf{p}_{-2} & 0 & \cdots & 0 & 0\\
0 & \mathbf{p}_2 & \mathbf{p}_1 & \mathbf{p}_0 & \mathbf{p}_{-1} & \mathbf{p}_{-2} & \cdots & 0 & 0\\
\vdots & & & & \ddots & & & & \\
\vdots & & & & & \ddots & & & \\
0 & 0 & \cdots & & \mathbf{p}_2 & \mathbf{p}_1 & \mathbf{p}_0 & \mathbf{p}_{-1} & \mathbf{p}_{-2}\\
\mathbf{p}_{-2} & 0 & \cdots & & 0 & \mathbf{p}_2 & \mathbf{p}_1 & \mathbf{p}_0 & \mathbf{p}_{-1}\\
\mathbf{p}_{-1} & \mathbf{p}_{-2} & \cdots & & 0 & 0 & \mathbf{p}_2 & \mathbf{p}_1 & \mathbf{p}_0
\end{bmatrix}; \tag{2.14}$$

note the systematic band-diagonal structure, which characterizes A as a circulant matrix. Linear systems involving circulant matrices can be quickly solved using fast Fourier transforms, a topic we will return to later.

Returning to the general case of $\mathbf{p}$ defined by (2.9), the approximation formula (2.12) can be written in the form

$$A\mathbf{f} \approx \mathcal{A}f. \tag{2.15}$$

Figure 2.3 shows data computed by the discrete model $A\mathbf{f}$ and compares the result to the continuous data $(\psi * f)(x)$ defined by (2.4).

Now let's add a little noise to the data. For example, we might take $k = 64 = n$ and construct the measurement noise in a probabilistic manner by taking a realization of a random vector with 64 independently distributed Gaussian elements having standard deviation $\sigma = 0.01 \cdot \max|f(x)|$. This corresponds to a relative noise level of 1%.

2.1.3 Naïve deconvolution and inverse crimes

We illustrate numerically the failure of the following naïve reconstruction attempt:

$$\mathbf{f} \approx A^{-1}\mathbf{m} \approx A^{-1}(A\mathbf{f} + \varepsilon) = \mathbf{f} + A^{-1}(\varepsilon). \tag{2.16}$$

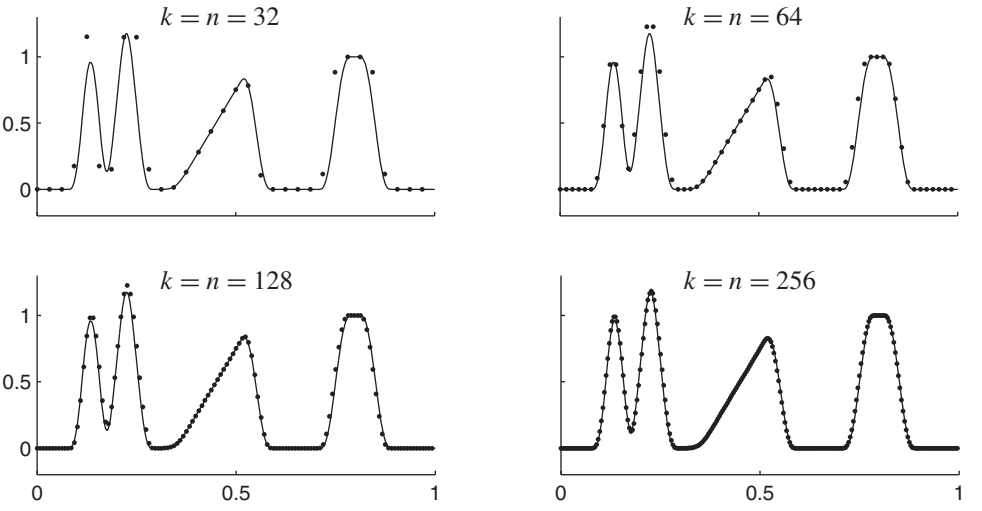

Figure 2.3. Illustration of the approximation $A\mathbf{f} \approx \mathcal{A}f$ of formula (2.15) for different choices of $k = n$. The actual function $(\psi * f)(x)$ defined by (2.4) is shown with a thin solid line, and the data points are indicated as dots. Note how the discrete approximation becomes better as the discretization is refined.

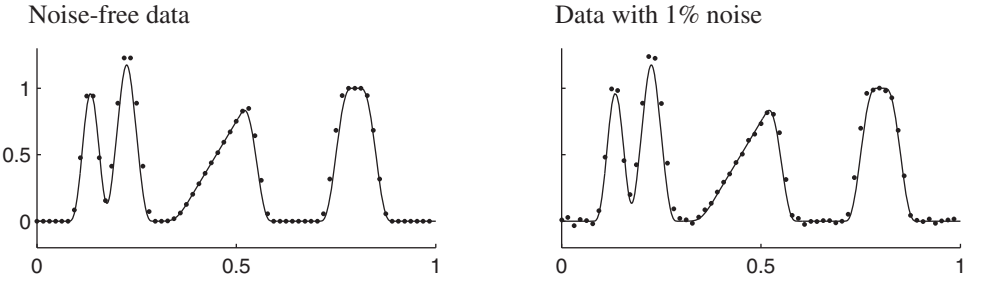

Figure 2.4. Illustration of simulated measurement noise. The actual function $(\psi * f)(x)$ defined by (2.4) is shown with a thin solid line, and the data points are indicated as dots. Left: noise-free discrete data $A\mathbf{f}$ with $n = 64 = k$. Right: the same data corrupted with 1% white noise.

In the case of no added noise ($\varepsilon = 0$) we use the data shown in the left plot of Figure 2.4 and get the left plot in Figure 2.5. The naïve reconstruction seems perfect! However, there is a catch. This apparently accurate reconstruction is not to be trusted; it is an example of an *inverse crime*. We will show how to avoid inverse crimes in Section 2.1.4.

If we apply naïve reconstruction (2.16) to the slightly noisy data shown in the right plot of Figure 2.4, we get the result shown in the right plot in Figure 2.5. It is completely useless. This example shows how sensitive inverse problems are to the smallest errors in measurement. We need to introduce *regularization* to overcome extreme sensitivity to measurement errors.

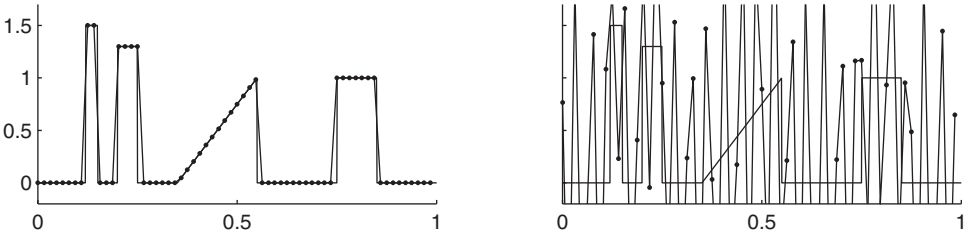

Figure 2.5. Two naïve deconvolutions by applying the inverse matrix A^{-1} to data. The original target function $f(x)$ is shown with a thin solid line, and the reconstruction is shown as dots. Left: naïve reconstruction (involving inverse crime) from the noise-free discrete data $A\mathbf{f}$ with $n = k = 64$ shown in the left plot in Figure 2.4. Right: naïve reconstruction from the noisy data shown in the right plot of Figure 2.4.

2.1.4 Naïve reconstruction without inverse crime

In the case of the deconvolution problem, we first simulate the measurements by convolving our known function f with a known discretized point spread function. In reality, when a blurred signal or image is encountered, the point spread function that "caused" the blurring is both unknown and can unlikely be expressed in simple terms. Thus, using the same point spread function for simulating a blurred signal and deconvolving the signal constitutes a serious inverse crime. Using the same point spread function *and* the same discretization mesh is an inverse felony!

We show one simple way to avoid inverse crime. We use a modified point spread function by taking $a = 0.041$ in (2.1) when simulating data. We compute the function $(\psi * f)(x)$ defined in (2.4) approximately at 1000 uniformly spaced points in the interval $[0,1]$ using trapezoidal rule with 400 quadrature points for the evaluation of the integral. Finally, we interpolate the values of $\psi * f$ at the 64 grid points using splines.

Now the data has been simulated completely differently than using the 64×64 model matrix A as was (criminally) done in Section 2.1.3.

We apply naïve inversion (2.16) to the crime-free data and show the results in Figure 2.6. Compare the left plots in Figures 2.5 and 2.6. Proper simulation of crime-free data

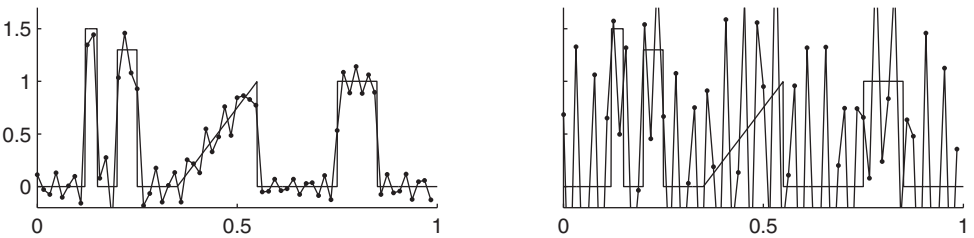

Figure 2.6. Two naïve deconvolutions by applying the inverse matrix A^{-1} to data generated avoiding inverse crime. The original target function $f(x)$ is shown with a thin solid line, and the reconstruction is shown as dots. Left: naïve reconstruction from noise-free discrete data with $n = k = 64$. Right: naïve reconstruction from noisy data. Compare to Figure 2.5.

reveals the ill-posedness of the deconvolution problem: The slightest perturbations in the data are amplified in naïve reconstruction using (2.16).

Exercise 2.1.1. *Determine whether the point spread function ψ is a $C^\infty(\mathbb{R})$ function.*

Exercise 2.1.2. *What is the effect of increasing the support of ψ_0 on v? Use the MATLAB programs* `DCcontdatacomp.m` *and* `DCcontdataplot.m` *to study the effect of increasing a on the convolved function. What do you observe?*

Exercise 2.1.3. *Plot a constant function of height 2 on* $[0,1]$ *before and after convolution with ψ. Use the MATLAB program* `DC2discretedatacomp.m` *to add noise to the convolved function and* `DC2naiveplot.m` *to compute a naïve reconstruction. Plot your results.*

2.2 Heat propagation

A classic ill-posed problem is that of determining the temperature distribution in a region from knowledge of the temperature distribution at the present time. This problem is known as the backward heat equation. We will begin with a discussion of the governing PDEs and their origins and then move to a simple discrete model.

2.2.1 Diffusion processes

The heat equation is the prototypical equation for modeling processes governed by pure diffusion. Following a probabilistic description as in, for example, [181], it can be derived by modeling the Brownian motion of the individual molecules in what we will assume to be a homogeneous material.

Suppose we have a material, such as depicted in Figure 2.7, containing n molecules, each of mass m, and suppose each molecule in this small volume is continually in motion. We will derive a model for one-dimensional spatial motion for simplicity, and so assume each molecule can only move to the left or to the right a distance Δx, representing an average displacement in time period Δt. To extend to higher dimensions, discrete motion in each of the three Cartesian coordinates would be permissible. Let p be the probability that the molecule moves to the right, and let q be the probability that the molecule moves to the left. Note that $p+q=1$. Let $u(x,t)$ be the probability per unit length that a molecule

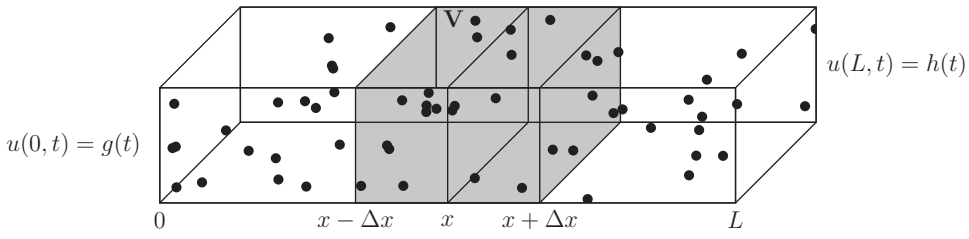

Figure 2.7. An illustration of the molecules in Brownian motion.