Appendix A

The Delta Function

A.1 One-Dimensional Delta Function

A.1.1 Various Definitions of the Delta Function

The delta function can be defined as the limit of $\delta^{(\varepsilon)}(x)$ when $\varepsilon \to 0$ (Figure A.1):

$$\delta(x) = \lim_{\varepsilon \to 0} \delta^{(\varepsilon)}(x), \tag{A.1}$$

where

$$\delta^{(\varepsilon)}(x) = \begin{cases} 1/\varepsilon, & -\varepsilon/2 < x < \varepsilon/2, \\ 0, & |x| > \varepsilon/2. \end{cases} \tag{A.2}$$

The delta function can be defined also by means of the following integral equations:

$$\int_{-\infty}^{+\infty} f(x)\delta(x)\, dx = f(0), \tag{A.3}$$

$$\int_{-\infty}^{+\infty} f(x)\delta(x-a)\, dx = f(a). \tag{A.4}$$

We should mention that the δ-function is not a function in the usual mathematical sense. It can be expressed as the limit of analytical functions such as

$$\delta(x) = \lim_{\varepsilon \to 0} \frac{\sin(x/\varepsilon)}{\pi x}, \qquad \delta(x) = \lim_{a \to \infty} \frac{\sin^2(ax)}{\pi a x^2}, \tag{A.5}$$

or

$$\delta(x) = \lim_{\varepsilon \to 0} \frac{1}{\pi} \frac{\varepsilon}{x^2 + \varepsilon^2}. \tag{A.6}$$

The Fourier transform of $\delta(x)$, which can be obtained from the limit of $\frac{\sin(x/\varepsilon)}{\pi x}$, is

$$\delta(x) = \frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{ikx} dk, \tag{A.7}$$

which in turn is equivalent to

$$\frac{1}{2\pi} \int_{-\infty}^{+\infty} e^{ikx} dk = \frac{1}{2\pi} \lim_{\varepsilon \to 0} \int_{-1/\varepsilon}^{+1/\varepsilon} e^{ikx} dk = \lim_{\varepsilon \to 0} \frac{\sin(x/\varepsilon)}{\pi x} = \delta(x). \tag{A.8}$$

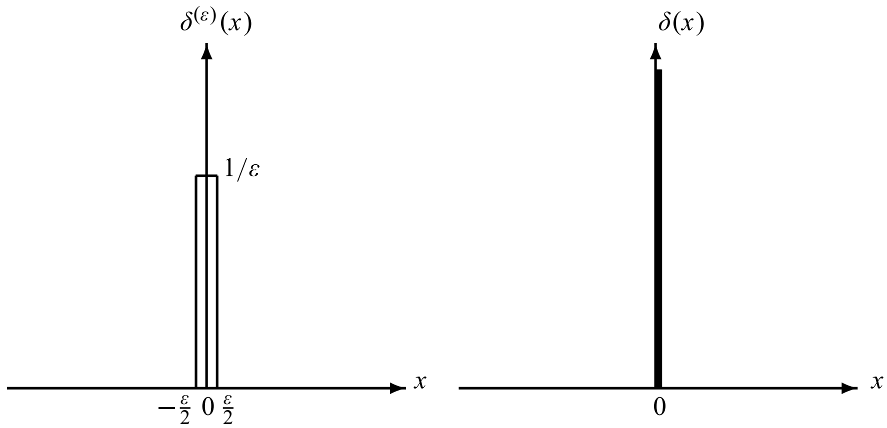

Figure A.1 The delta function $\delta(x)$ as defined by $\delta(x) = \lim_{\varepsilon \to 0} \delta^{(\varepsilon)}(x)$.

A.1.2 Properties of the Delta Function

The delta function is even:

$$\delta(-x) = \delta(x) \qquad \text{and} \qquad \delta(x-a) = \delta(a-x). \tag{A.9}$$

Here are some of the most useful properties of the delta function:

$$\int_a^b f(x)\delta(x-x_0)\,dx = \begin{cases} f(x_0), & \text{if} \quad a < x_0 < b, \\ 0, & \text{elsewhere,} \end{cases} \tag{A.10}$$

$$\delta(x) = 0 \qquad \text{for} \quad x \neq 0, \tag{A.11}$$

$$x\delta(x) = 0, \tag{A.12}$$

$$\delta(ax) = \frac{1}{|a|}\delta(x) \qquad (a \neq 0), \tag{A.13}$$

$$f(x)\delta(x-a) = f(a)\delta(x-a), \tag{A.14}$$

$$\int_c^d \delta(a-x)\delta(x-b)\,dx = \delta(a-b) \qquad \text{for} \quad c \le a \le d, \quad c \le b \le d, \tag{A.15}$$

$$\int_a^b \delta(x)\,dx = 1 \qquad \text{for} \quad a \le 0 \le b \tag{A.16}$$

$$\delta[g(x)] = \sum_i \frac{1}{|g'(x_i)|}\delta(x-x_i), \tag{A.17}$$

where x_i is a zero of $g(x)$ and $g'(x_i) \neq 0$. Using (A.17), we can verify that

$$\delta[(x-a)(x-b)] = \frac{1}{|a-b|}[\delta(x-a) + \delta(x-b)] \qquad (a \neq b), \tag{A.18}$$

$$\delta(x^2 - a^2) = \frac{1}{2|a|}[\delta(x-a) + \delta(x+a)] \qquad (a \neq 0). \tag{A.19}$$

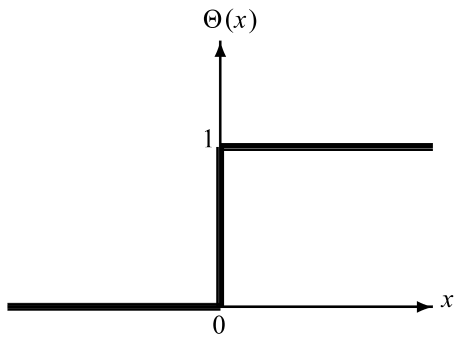

Figure A.2 The Heaviside function $\Theta(x)$.

A.1.3 Derivative of the Delta Function

The Heaviside function, or step function is defined as follows; see Figure A.2:

$$\Theta(x) = \begin{cases} 1, & x > 0, \\ 0, & x < 0. \end{cases} \tag{A.20}$$

The derivative of the Heaviside function gives back the delta function:

$$\frac{d}{dx}\Theta(x) = \delta(x). \tag{A.21}$$

Using the Fourier transform of the delta function, we can write

$$\frac{d\delta(x)}{dx} = \delta'(x) = \frac{i}{2\pi}\int_{-\infty}^{+\infty} ke^{ikx}\,dk. \tag{A.22}$$

Another way of looking at the derivative of the delta function is by means of the following integration by parts of $\delta'(x-a)$:

$$\int_{-\infty}^{\infty} f(x)\delta'(x-a)\,dx = f(x)\delta(x-a)|_{-\infty}^{\infty} - \int_{-\infty}^{\infty} f'(x)\delta(x-a)\,dx = -f'(a), \tag{A.23}$$

or

$$\int_{-\infty}^{\infty} f(x)\delta'(x-a)\,dx = -f'(a), \tag{A.24}$$

where we have used the fact that $f(x)\delta(x-a)$ is zero at $\pm\infty$. Following the same procedure, we can show that

$$\int_{-\infty}^{\infty} f(x)\delta''(x-a)\,dx = (-1)^2 f''(a) = f''(a). \tag{A.25}$$

Similar repeated integrations by parts lead to the following general relation:

$$\int_{-\infty}^{\infty} f(x)\delta^{(n)}(x-a)\,dx = (-1)^n f^{(n)}(a), \tag{A.26}$$

where $\delta^{(n)}(x-a) = d^n[\delta(x-a)]/dx^n$ and $f^{(n)}(a) = d^n f(x)/dx^n|_{x=a}$. In particular, if $f(x)=1$ and $n=1$, we have

$$\int_{-\infty}^{\infty} \delta'(x-a)\, dx = 0. \tag{A.27}$$

Here is a list of useful properties of the derivative of the delta function:

$$\delta'(x) = -\delta'(-x), \tag{A.28}$$
$$x\delta'(x) = -\delta(x), \tag{A.29}$$
$$x^2\delta'(x) = 0, \tag{A.30}$$
$$x^2\delta''(x) = 2\delta(x). \tag{A.31}$$

A.2 Three-Dimensional Delta Function

The three-dimensional form of the delta function is given in Cartesian coordinates by

$$\delta(\vec{r}-\vec{r}\,') = \delta(x-x\,')\delta(y-y\,')\delta(z-z\,') \tag{A.32}$$

and in spherical coordinates by

$$\begin{aligned}\delta(\vec{r}-\vec{r}\,') &= \frac{1}{r^2}\delta(r-r')\delta(\cos\theta-\cos\theta')\delta(\varphi-\varphi') \\ &= \frac{1}{r^2\sin\theta}\delta(r-r')\delta(\theta-\theta')\delta(\varphi-\varphi'),\end{aligned} \tag{A.33}$$

since, according to (A.17), we have $\delta(\cos\theta-\cos\theta') = \delta(\theta-\theta')/\sin\theta$.

The Fourier transform of the three-dimensional delta function is

$$\delta(\vec{r}-\vec{r}\,') = \frac{1}{(2\pi)^3}\int d^3k\, e^{i\vec{k}\cdot(\vec{r}-\vec{r}\,')}, \tag{A.34}$$

and

$$\int d^3r\, f(\vec{r})\delta(\vec{r}) = f(0), \qquad \int d^3r\, f(\vec{r})\delta(\vec{r}-\vec{r}_0) = f(\vec{r}_0). \tag{A.35}$$

The following relations are often encountered:

$$\vec{\nabla}\cdot\left(\frac{\hat{r}}{r^2}\right) = 4\pi\,\delta(\vec{r}), \qquad \nabla^2\left(\frac{1}{r}\right) = -4\pi\,\delta(\vec{r}), \tag{A.36}$$

where $\hat{r}$ the unit vector along $\vec{r}$.

We should mention that the physical dimension of the delta function is one over the dimensions of its argument. Thus, if x is a distance, the physical dimension of $\delta(x)$ is given by $[\delta(x)] = 1/[x] = 1/L$, where L is a length. Similarly, the physical dimensions of $\delta(\vec{r})$ is $1/L^3$, since

$$\left[\delta(\vec{r})\right] = \left[\delta(x)\delta(y)\delta(z)\right] = \frac{1}{[x]}\frac{1}{[y]}\frac{1}{[z]} = \frac{1}{L^3}. \tag{A.37}$$

Appendix B

Angular Momentum in Spherical Coordinates

In this appendix, we will show how to derive the expressions of the gradient $\vec{\nabla}$, the Laplacian ∇^2, and the components of the orbital angular momentum in spherical coordinates.

B.1 Derivation of Some General Relations

The Cartesian coordinates (x, y, z) of a vector $\vec{r}$ are related to its spherical polar coordinates (r, θ, φ) by

$$x = r\sin\theta\cos\varphi, \qquad y = r\sin\theta\sin\varphi, \qquad z = r\cos\theta. \tag{B.1}$$

The orthonormal Cartesian basis $(\hat{x}, \hat{y}, \hat{z})$ is related to its spherical counterpart $(\hat{r}, \hat{\theta}, \hat{\varphi})$ by

$$\hat{x} = \hat{r}\sin\theta\cos\varphi + \hat{\theta}\cos\theta\cos\varphi - \hat{\varphi}\sin\varphi, \tag{B.2}$$

$$\hat{y} = \hat{r}\sin\theta\sin\varphi + \hat{\theta}\cos\theta\sin\varphi + \hat{\varphi}\cos\varphi, \tag{B.3}$$

$$\hat{z} = \hat{r}\cos\theta - \hat{\theta}\sin\theta. \tag{B.4}$$

Differentiating (B.1), we obtain

$$dx = \sin\theta\cos\varphi\, dr + r\cos\theta\cos\varphi\, d\theta - r\sin\theta\sin\varphi\, d\varphi, \tag{B.5}$$

$$dy = \sin\theta\sin\varphi\, dr + r\cos\theta\sin\varphi\, d\theta + r\sin\theta\cos\varphi\, d\varphi, \tag{B.6}$$

$$dz = \cos\theta\, dr - r\sin\theta\, d\theta. \tag{B.7}$$

Solving these equations for dr, $d\theta$, and $d\varphi$, we obtain

$$dr = \sin\theta\cos\varphi\, dx + \sin\theta\sin\varphi\, dy + \cos\theta\, dz, \tag{B.8}$$

$$d\theta = \frac{1}{r}\cos\theta\cos\varphi\, dx + \frac{1}{r}\cos\theta\sin\varphi\, dy - \frac{1}{r}\sin\theta\, dz, \tag{B.9}$$

$$d\varphi = -\frac{\sin\varphi}{r\sin\theta}\, dx + \frac{\cos\varphi}{r\sin\theta}\, dy. \tag{B.10}$$

We can verify that (B.5) to (B.10) lead to

$$\frac{\partial r}{\partial x} = \sin\theta\cos\varphi, \qquad \frac{\partial\theta}{\partial x} = \frac{1}{r}\cos\varphi\cos\theta, \qquad \frac{\partial\varphi}{\partial x} = -\frac{\sin\varphi}{r\sin\theta}, \tag{B.11}$$

$$\frac{\partial r}{\partial y} = \sin\theta\sin\varphi, \qquad \frac{\partial\theta}{\partial y} = \frac{1}{r}\sin\varphi\cos\theta, \qquad \frac{\partial\varphi}{\partial y} = \frac{\cos\varphi}{r\sin\theta}, \tag{B.12}$$

$$\frac{\partial r}{\partial z} = \cos\theta, \qquad \frac{\partial\theta}{\partial z} = -\frac{1}{r}\sin\theta, \qquad \frac{\partial\varphi}{\partial z} = 0, \tag{B.13}$$

which, in turn, yield

$$\begin{aligned}\frac{\partial}{\partial x} &= \frac{\partial}{\partial r}\frac{\partial r}{\partial x} + \frac{\partial}{\partial\theta}\frac{\partial\theta}{\partial x} + \frac{\partial}{\partial\varphi}\frac{\partial\varphi}{\partial x} \\ &= \sin\theta\cos\varphi\frac{\partial}{\partial r} + \frac{1}{r}\cos\theta\cos\varphi\frac{\partial}{\partial\theta} - \frac{\sin\varphi}{r\sin\theta}\frac{\partial}{\partial\varphi},\end{aligned} \tag{B.14}$$

$$\begin{aligned}\frac{\partial}{\partial y} &= \frac{\partial}{\partial r}\frac{\partial r}{\partial y} + \frac{\partial}{\partial\theta}\frac{\partial\theta}{\partial y} + \frac{\partial}{\partial\varphi}\frac{\partial\varphi}{\partial y} \\ &= \sin\theta\sin\varphi\frac{\partial}{\partial r} + \frac{1}{r}\cos\theta\sin\varphi\frac{\partial}{\partial\theta} + \frac{\cos\varphi}{r\sin\theta}\frac{\partial}{\partial\varphi},\end{aligned} \tag{B.15}$$

$$\frac{\partial}{\partial z} = \frac{\partial}{\partial r}\frac{\partial r}{\partial z} + \frac{\partial}{\partial\theta}\frac{\partial\theta}{\partial z} + \frac{\partial}{\partial\varphi}\frac{\partial\varphi}{\partial z} = \cos\theta\frac{\partial}{\partial r} - \frac{\sin\theta}{r}\frac{\partial}{\partial\theta}. \tag{B.16}$$

B.2 Gradient and Laplacian in Spherical Coordinates

We can show that a combination of (B.14) to (B.16) allows us to express the operator $\vec{\nabla}$ in spherical coordinates:

$$\vec{\nabla} = \hat{x}\frac{\partial}{\partial x} + \hat{y}\frac{\partial}{\partial y} + \hat{z}\frac{\partial}{\partial z} = \hat{r}\frac{\partial}{\partial r} + \hat{\theta}\frac{1}{r}\frac{\partial}{\partial\theta} + \hat{\varphi}\frac{1}{r\sin\theta}\frac{\partial}{\partial\varphi}, \tag{B.17}$$

and also the Laplacian operator ∇^2:

$$\nabla^2 = \vec{\nabla}.\vec{\nabla} = \left(\hat{r}\frac{\partial}{\partial r} + \frac{\hat{\theta}}{r}\frac{\partial}{\partial\theta} + \frac{\hat{\varphi}}{r\sin\varphi}\frac{\partial}{\partial\varphi}\right)\cdot\left(\hat{r}\frac{\partial}{\partial r} + \frac{\hat{\theta}}{r}\frac{\partial}{\partial\theta} + \frac{\hat{\varphi}}{r\sin\theta}\frac{\partial}{\partial\varphi}\right). \tag{B.18}$$

Now, using the relations

$$\frac{\partial\hat{r}}{\partial r} = 0, \qquad \frac{\partial\hat{\theta}}{\partial r} = 0, \qquad \frac{\partial\hat{\varphi}}{\partial r} = 0, \tag{B.19}$$

$$\frac{\partial\hat{r}}{\partial\theta} = \hat{\theta}, \qquad \frac{\partial\hat{\theta}}{\partial\theta} = -\hat{r}, \qquad \frac{\partial\hat{\varphi}}{\partial\theta} = 0, \tag{B.20}$$

$$\frac{\partial\hat{r}}{\partial\varphi} = \hat{\varphi}\sin\theta, \qquad \frac{\partial\hat{\theta}}{\partial\varphi} = \hat{\varphi}\cos\theta, \qquad \frac{\partial\hat{\varphi}}{\partial\varphi} = -\hat{r}\sin\theta - \hat{\theta}\cos\theta, \tag{B.21}$$

we can show that the Laplacian operator reduces to

$$\nabla^2 = \frac{1}{r^2}\left[\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right) + \frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{\sin^2\theta}\frac{\partial^2}{\partial\varphi^2}\right]. \tag{B.22}$$

B.3 Angular Momentum in Spherical Coordinates

The orbital angular momentum operator $\vec{L}$ can be expressed in spherical coordinates as

$$\hat{\vec{L}} = \hat{\vec{R}} \times \hat{\vec{P}} = (-i\hbar r)\hat{r} \times \vec{\nabla} = (-i\hbar r)\hat{r} \times \left[\hat{r}\frac{\partial}{\partial r} + \frac{\hat{\theta}}{r}\frac{\partial}{\partial \theta} + \frac{\hat{\varphi}}{r\sin\theta}\frac{\partial}{\partial \varphi}\right], \tag{B.23}$$

or as

$$\hat{\vec{L}} = -i\hbar\left(\hat{\varphi}\frac{\partial}{\partial\theta} - \frac{\hat{\theta}}{\sin\theta}\frac{\partial}{\partial\varphi}\right). \tag{B.24}$$

Using (B.24) along with (B.2) to (B.4), we express the components $\hat{L}_x, \hat{L}_y, \hat{L}_z$ within the context of the spherical coordinates. For instance, the expression for $\hat{L}_x$ can be written as follows:

$$\begin{aligned}\hat{L}_x &= \hat{x}.\vec{L} = -i\hbar\left(\hat{r}\sin\theta\cos\varphi + \hat{\theta}\cos\theta\cos\varphi - \hat{\varphi}\sin\varphi\right)\cdot\left(\hat{\varphi}\frac{\partial}{\partial\theta} - \frac{\hat{\theta}}{\sin\theta}\frac{\partial}{\partial\varphi}\right)\\ &= i\hbar\left(\sin\varphi\frac{\partial}{\partial\theta} + \cot\theta\cos\varphi\frac{\partial}{\partial\varphi}\right). \end{aligned} \tag{B.25}$$

Similarly, we can easily obtain

$$\hat{L}_y = i\hbar\left(-\cos\varphi\frac{\partial}{\partial\theta} + \cot\theta\sin\varphi\frac{\partial}{\partial\varphi}\right), \tag{B.26}$$

$$\hat{L}_z = -i\hbar\frac{\partial}{\partial\varphi}. \tag{B.27}$$

From the expressions (B.25) and (B.26) for $\hat{L}_x$ and $\hat{L}_y$, we infer that

$$\hat{L}_\pm = \hat{L}_x \pm i\hat{L}_y = \pm\hbar e^{\pm i\varphi}\left(\frac{\partial}{\partial\theta} \pm i\cot\theta\frac{\partial}{\partial\varphi}\right). \tag{B.28}$$

The expression for $\vec{L}^2$ is

$$\vec{L}^2 = -\hbar^2 r^2(\hat{r}\times\vec{\nabla})\cdot(\hat{r}\times\vec{\nabla}) = -\hbar^2 r^2\left[\nabla^2 - \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right)\right]; \tag{B.29}$$

it can be easily written in terms of the spherical coordinates as

$$\vec{L}^2 = -\hbar^2\left[\frac{1}{\sin\theta}\frac{\partial}{\partial\theta}\left(\sin\theta\frac{\partial}{\partial\theta}\right) + \frac{1}{\sin^2\theta}\frac{\partial^2}{\partial\varphi^2}\right]. \tag{B.30}$$

This expression was derived by substituting (B.22) into (B.29).

Note that, using the expression (B.29) for $\vec{L}^2$, we can rewrite ∇^2 as

$$\nabla^2 = \frac{1}{r^2}\frac{\partial}{\partial r}\left(r^2\frac{\partial}{\partial r}\right) - \frac{1}{\hbar^2 r^2}\vec{L}^2 = \frac{1}{r}\frac{\partial^2}{\partial r^2}r - \frac{1}{\hbar^2 r^2}\vec{L}^2. \tag{B.31}$$

Appendix C

C++ Code for Solving the Schrödinger Equation

This C++ code is designed to solve the one-dimensional Schrödinger equation for a harmonic oscillator (HO) potential as well as for an infinite square well (ISW) potential as outlined in Chapter 4. My special thanks are due to Dr. M. Bulut and to Prof. Dr. H. Mueller-Krumbhaar and his Ph.D. student C. Gugenberger who have worked selflessly hard to write and test the code listed below. Dr. Mevlut wrote an early code for the ISW, while Prof. Mueller-Krumbhaar and Gugenberger not only wrote a new code (see the version listed below) for the HO but also designed it in a way that it applies to the ISW potential as well (they have also added effective didactic comments so that our readers can effortlessly understand the code and make use of it).

Note: to shift from the harmonic oscillator code to the infinite square well code, one needs simply to erase the first double forward-slash (i.e., "//") from the oscillator's program line below:

```
E_pot[i] = 0.5*dist*dist; // E_pot[i]=0;//E_pot=0:Infinite Well!
```

Of course, one still needs to rescale the energy and the value of 'xRange' in order to agree with the algorithm outlined at the end of Chapter 4.

The C++ Code: osci.cpp

```
/* osci.cpp: Solution of the one-dimensional Schrodinger equation for
  a particle in a harmonic potential, using the shooting method.
  To compile and link with gnu compiler, type: g++ -o osci osci.cpp
  To run the current C++ program, simply type: osci
  Plot by gnuplot: /GNUPLOT> set terminal windows
                   /GNUPLOT> plot "psi-osc.dat" with lines  */
#include <cstdio>
#include <cstdlib>
#include <cmath>
#define MAX(a, b) (((a) > (b)) ? (a) : (b))
int main(int argc, char*argv[])
{// Runtime constants
const static double Epsilon = 1e-10; // Defines the precision of
                 //...  energy calculations
```

```
const static int N_of_Divisions = 1000;
const static int N_max = 5; //Number of calculated Eigenstates

FILE *Wavefunction_file, *Energy_file, *Potential_file;
Wavefunction_file = fopen("psi-osc.dat","w");
Energy_file = fopen("E_n_Oszillator.dat","w");
Potential_file = fopen("HarmonicPotentialNoDim.dat", "w");
if (!(Wavefunction_file && Energy_file && Potential_file))
{ printf("Problems to create files output.\n"); exit(2); }

/* Physical parameters using dimensionless quantities.
 ATTENTION:  We set initially:  hbar = m = omega = a = 1, and
 reintroduce physical values at the end.  According to Eq.(4.117),
 the ground state energy then is  E_n = 0.5. Since the wave function
 vanishes only at -infinity and  +infinity, we have to cut off the
 calculation somewhere, as given by  'xRange'. If xRange is chosen
 too large, the open (positive) end of  the  wave function can
 diverge numerically in this simple shooting  approach. */

const static double xRange = 12; // xRange=11.834 corresponds to a
  //... physical range of -20fm < x < +20fm, see after Eq.(4.199).
const static double h_0 = xRange / N_of_Divisions;
double* E_pot = new double[N_of_Divisions+1];
double dist;

for (int i = 0; i <= N_of_Divisions; ++i)
{ // Harmonic potential, as given in Eq. (4.115), but dimensionless
dist = i*h_0 - 0.5*xRange;
E_pot[i] = 0.5*dist*dist; // E_pot[i]=0;//E_pot=0:Infinite Well!
fprintf(Potential_file, "%16.12e \t\t %16.12e\n", dist, E_pot[i]);
}
fclose(Potential_file);

/* Since the Schrodinger equation is linear, the amplitude of the
 wavefunction will be fixed by normalization.
At left we set it small but nonzero. */
const static double Psi_left = 1.0e-3; // left boundary condition
const static double Psi_right = 0.0; // right boundary condition

double *Psi, *EigenEnergies;// Arrays to hold the results
Psi = new double[N_of_Divisions+1]; //N_of_Points = N_of_Divisions+1
EigenEnergies = new double[N_max+1];
Psi[0] = Psi_left;
Psi[1] = Psi_left + 1.0e-3; // Add arbitrary small value

int N_quantum;//N_quantum is Energy Quantum Number
int Nodes_plus; // Number of nodes (+1) in wavefunction
```

```
double K_square;// Square of wave vector
// Initial Eigen-energy search limits
double E_lowerLimit = 0.0;// Eigen-energy must be positive
double E_upperLimit = 10.0;
int End_sign = -1;
bool Limits_are_defined = false;
double Normalization_coefficient;
double E_trial;

// MAIN LOOP begins:-----------------------------------
for(N_quantum=1; N_quantum <= N_max; ++N_quantum)
{
// Find the eigen-values for energy. See theorems (4.1) and (4.2).
Limits_are_defined = false;
while (Limits_are_defined == false)
{ /* First, determine an upper limit for energy, so that the wave-
 function Psi[i] has one node more than physically needed.   */
Nodes_plus = 0;
E_trial = E_upperLimit;
for (int i=2; i <= N_of_Divisions; ++i)
{ K_square = 2.0*(E_trial - E_pot[i]);
// Now use the NUMEROV-equation (4.197) to calculate wavefunction
Psi[i] = 2.0*Psi[i-1]*(1.0 - (5.0*h_0*h_0*K_square / 12.0))
/(1.0 + (h_0*h_0*K_square/12.0))-Psi[i-2];
if (Psi[i]*Psi[i-1] < 0) ++Nodes_plus;
}
/* If one runs into the following condition, the modification
 of the upper limit was too aggressive. */
if (E_upperLimit < E_lowerLimit)
E_upperLimit = MAX(2*E_upperLimit, -2*E_upperLimit);
if (Nodes_plus > N_quantum) E_upperLimit *= 0.7;
else if (Nodes_plus < N_quantum) E_upperLimit *= 2.0;
else Limits_are_defined = true; // At least one node should appear.
} // End of the loop: while (Limits_are_defined == false)
// Refine the energy by satisfying the right boundary condition.
End_sign = -End_sign;
while ((E_upperLimit - E_lowerLimit) > Epsilon)
{ E_trial = (E_upperLimit + E_lowerLimit) / 2.0;
for (int i=2; i <= N_of_Divisions; ++i)
{ // Again eq.(4.197) of the Numerov-algorithm:
K_square = 2.0*(E_trial - E_pot[i]);
Psi[i] = 2.0*Psi[i-1] * (1.0 - (5.0*h_0*h_0*K_square / 12.0))
 /(1.0 + (h_0*h_0*K_square/12.0))-Psi[i-2];
}
if (End_sign*Psi[N_of_Divisions] > Psi_right) E_lowerLimit = E_trial;
else E_upperLimit = E_trial;
} // End of loop: while ((E_upperLimit - E_lowerLimit) > Epsilon)
```

```
// Initialization for the next iteration in main loop
E_trial = (E_upperLimit+E_lowerLimit)/2;
EigenEnergies[N_quantum] = E_trial;
E_upperLimit = E_trial;
E_lowerLimit = E_trial;

// Now find the normalization coefficient
double Integral = 0.0;
for (int i=1; i <= N_of_Divisions; ++i)
{ // Simple integration
Integral += 0.5*h_0*(Psi[i-1]*Psi[i-1]+Psi[i]*Psi[i]);
}
Normalization_coefficient = sqrt(1.0/Integral);
// Output of normalized dimensionless wave function
for (int i=0; i <=N_of_Divisions; ++i)
{ fprintf(Wavefunction_file, "%16.12e \t\t %16.12e\n",
 i*h_0 - 0.5*xRange, Normalization_coefficient*Psi[i]);
}
fprintf(Wavefunction_file,"\n");
} // End of MAIN LOOP. ---------------------------------
fclose(Wavefunction_file);

/*Finally convert dimensionless units in real units. Note that
  energy does not depend explicitly on the particle's mass anymore:
    hbar = 1.05457e-34;// Planck constant/2pi
    omega = 5.34e21; // Frequency in 1/s
    MeV = 1.602176487e-13; // in J
    The correct normalization would be hbar*omega/MeV = 3.5148461144,
    but we use the approximation 3.5 for energy-scale as in chap. 4.9 */
const static double Energyscale = 3.5;// in MeV
// Output with rescaled dimensions; assign Energy_file
printf("Quantum Harmonic Oscillator, program osci.cpp\n");
printf("Energies in MeV:\n");
printf("n \t\t E_n\n");
for (N_quantum=1; N_quantum <= N_max; ++N_quantum)
{ fprintf(Energy_file,"%d \t\t %16.12e\n", N_quantum-1,
  Energyscale*EigenEnergies[N_quantum]);
  printf("%d \t\t %16.12e\n", N_quantum-1,
  Energyscale*EigenEnergies[N_quantum]);
}
fprintf(Energy_file,"\n");
fclose(Energy_file);
printf("Wave-Functions in File: psi_osc.dat \n");
printf("\n");
return 0;
}
```