Appendix CC

More Accurate Solutions of the Eigenvalue Problem

We consider the finite well again using a more accurate 5-point formula for the derivatives. As before, our basic strategy will be to use a finite-difference approximation of the second derivative in Eqs. (2.24) and (2.27) to convert these differential equations into a set of linear equations, which can be easily solved with MATLAB.

A 5-POINT FINITE DIFFERENCE FORMULA

To get started, we first introduce dimensionless variables that give the position of the particle in nanometers and the energy and potential energy in electron volts.

$$x = \chi \cdot \text{nm}, \quad E = \epsilon \cdot \text{eV}, \quad V_0 = \mathcal{V}_0 \cdot \text{eV}. \tag{CC.1}$$

By substituting these expressions for x, E, and V_0 into Eqs. (2.24) and (2.27) we obtain the following equations

$$-\frac{\mathrm{d}^2 u}{\mathrm{d}\chi^2} = E_0\, \epsilon u, \quad \text{for } -5 \le \chi \le 5, \tag{CC.2}$$

and

$$-\frac{\mathrm{d}^2 u}{\mathrm{d}\chi^2} + \mathcal{V}_0 E_0 u = \epsilon\, E_0 u, \quad \text{for } |\chi| \ge 5\ , \tag{CC.3}$$

where $u(x)$ is the wave function and E_0 is a dimensionless number given by the equation

$$E_0 = \frac{2m(\text{nm})^2\text{eV}}{\hbar^2}. \tag{CC.4}$$

For the finite well described in Section 2.3, the well extends from $\chi = -5$ to $\chi = +5$ and $\mathcal{V}_0 = 0.3$.

We introduce the uniform grid,

$$\chi_i = (i-1) * \delta \text{ with } i = 0, 1, \ldots, n+1,$$

where δ is the grid spacing. As we shall see, only the points, $\chi_1, \ldots, \chi_n$ will play a role in the actual computation with $\chi_0 = -\delta$ and $\chi_{n+1} = n * \delta$ serving as auxiliary points. The second derivative $u''(\chi)$ may be approximated by the following 5-point finite difference formula

$$u''(\chi) = \frac{-u(\chi + 2*\delta) + 16 * u(\chi + \delta) - 30 * u(\chi) + 16 * u(\chi - \delta) - u(\chi - 2 * \delta}{12 * \delta^2}.$$

The value of $u(\chi)$ corresponding to the grid point χ_i will again be denoted by u_i. With this notation, the value of the second derivative at the grid point χ_i is

$$u_i'' = \frac{-u_{i+2} + 16 * u_{i+1} - 30 * u_i + 16 * u_{i-1} - u_{i-2}}{12 * \delta^2}. \tag{CC.5}$$

Special care must be taken at the end points to ensure that the boundary conditions are satisfied. For the even solutions, the wave function is non-zero and has a zero derivative at the origin. According to the finite difference formula, the value of the second derivative at the origin is

$$u_1'' = \frac{-u_3 + 16 * u_2 - 30 * u_1 + 16 * u_0 - u_{-1}}{12 * \delta^2},$$

and the value at the second grid point is

$$u_2'' = \frac{-u_4 + 16 * u_3 - 30 * u_2 + 16 * u_1 - u_0}{12 * \delta^2}.$$

We note, however, that for an even function, $u_0 = u(-\delta) = u(+\delta) = u_2$ and $u_{-1} = u(-2 * \delta) = (2 * \delta) = u_3$. The above equations can thus be written as

$$u_1'' = \frac{2 * (-u_3 + 16 * u_2) - 30 * u_1}{12 * \delta^2}, \tag{CC.6}$$

and the value at the second grid point is

$$u_2'' = \frac{-u_4 + 16 * u_3 - 31 * u_2 + 16 * u_1}{12 * \delta^2}. \tag{CC.7}$$

The second derivatives at χ_{n-1} and χ_n are given by the formulas

$$u_{n-1}'' = \frac{-u_{n+1} + 16 * u_n - 30 * u_{n-1} + 16 * u_{n-2} - u_{n-3}}{12 * \delta^2}$$

and

$$u_n'' = \frac{-u_{n+2} + 16 * u_{n+1} - 30 * u_n + 16 * u_{n-1} - u_{n-2}}{12 * \delta^2}.$$

However, even- and odd-functions are both zero at the last grid points, $\chi_{n+1} = n\delta$ and $\chi_{n+2} = (n+1) * \delta$, and both these equations may be written as

$$u_{n-1}'' = \frac{16 * u_n - 30 * u_{n-1} + 16 * u_{n-2} - u_{n-3}}{12 * \delta^2} \tag{CC.8}$$

and

$$u_n'' = \frac{-30 * u_n + 16 * u_{n-1} - u_{n-2}}{12 * \delta^2}. \tag{CC.9}$$

Using Eqs. (CC.5)-(CC.9), the Schrödinger equations for a finite well can be transformed into a set of linear equations. We note that Eq. (CC.3), which applies outside the well, has a second derivative and another term depending on the potential $\mathcal{V}_0$, while Eq. (CC.2), which applies inside the well, has only a second derivative. If we chose a very coarse 5-point grid with the points, $\chi = 0, 4, 8, 12, 16$, the conditions that Eqs. (CC.2) and (CC.3) are satisfied at the grid points are

$$\begin{aligned}
-u_1'' &= E_0 \epsilon u_1, \\
-u_2'' &= E_0 \epsilon u_2, \\
-u_3'' + \mathcal{V}_0 E_0 u_3 &= E_0 \epsilon u_3, \\
-u_4'' + \mathcal{V}_0 E_0 u_4 &= E_0 \epsilon u_4, \\
-u_5'' + \mathcal{V}_0 E_0 u_5 &= E_0 \epsilon u_5.
\end{aligned}$$

We now use Eqs. (CC.5)-(CC.9) to evaluate the second derivatives in the above equations, and we multiply each of the resulting equations by $12 * \delta^2$ to obtain

$$\begin{array}{ccccccc}
30 * u_1 & -32 * u_2 & 2 * u_3 & & & = & 12 * \delta^2 E_0 \epsilon u_1 \\
-16 * u_1 & +31 * u_2 & -16 * u_3 & +u_4 & & = & 12 * \delta^2 E_0 \epsilon u_2 \\
u_1 & -16 * u_2 & (30 + 12 * \delta^2 \mathcal{V}_0 E_0) u_3 & -16 * u_4 & u_5 & = & 12 * \delta^2 E_0 \epsilon u_3 \\
 & u_2 & -16 * u_3 & (30 + 12 * \delta^2 \mathcal{V}_0 E_0) u_4 & -16 * u_5 & = & 12 * \delta^2 E_0 \epsilon u_4 \\
 & & u_3 & -16 * u_4 & (30 + 12 * \delta^2 \mathcal{V}_0 E_0) u_5 & = & 12 * \delta^2 E_0 \epsilon u_5
\end{array}$$

These last equations can be written in matrix form. We define the matrix **A** by the equation

$$\mathbf{A} = \begin{bmatrix} 30 & -32 & +2 & 0 & 0 \\ -16 & 31 & -16 & +1 & 0 \\ +1 & -16 & (30 + 12 * \delta^2 \mathcal{V}_0 E_0) & -16 & +1 \\ 0 & +1 & -16 & (30 + 12 * \delta^2 \mathcal{V}_0 E_0) & -16 \\ 0 & 0 & +1 & -16 & (30 + 12 * \delta^2 \mathcal{V}_0 E_0) \end{bmatrix}$$

and the vector **u** by the equation

$$\mathbf{u} = \begin{bmatrix} u_1 \\ u_2 \\ u_3 \\ u_4 \\ u_5 \end{bmatrix}.$$

With this notation, the above equations for u_1, u_2, u_3, u_4, and u_5 can be written simply

$$\mathbf{Au} = 12 * \delta^2 E_0 \epsilon \mathbf{u} \tag{CC.10}$$

We have thus converted the eigenvalue problem for the finite well into a matrix eigenvalue problem. MATLAB Program CC.1 given below may be used to find the eigenvalues and eigenfunctions of the matrix A.

MATLAB Program CC.1

A MATLAB program for finding the eigenvalues and eigenvectors of an electron moving in a finite well.

```
xmax=20;
L=5;
n=5;
delta=xmax/n;
n1=fix(L/delta)+1;
n2=n-n1;
e0=1.759
v1=ones(n-1,1);
v2=ones(n-2,1)
d=[30*ones(n1,1);(30+3.6*e0*delta^2)*ones(n2,1)];
A=-16*diag(v1,-1)-16*diag(v1,1)+diag(v2,-2)+diag(v2,2)+diag(d);
A(1,2)=-32;
A(1,3)=2;
A(2,2)=31;
A
[E V]=eig(A);
GroundState = V(1,1)/(12.0*e0*delta^2)
```

As before, the first four lines of MATLAB Program CC.1 define the length of the physical region (xmax), the χ coordinate of the edge of the well (L), the number of grid points (n), and the step size (delta). The MATLAB function "fix" in the next line of the program rounds the ratio "L/delta" to the integer toward zero. The integer $n1$, which is the number of grid points within the well, is then obtained by adding the point at the origin. The integer $n2$ is the number of grid points outside the well. After defining the constant E0 the program then defines vectors $\mathbf{v}_1$ and $\mathbf{v}_2$. The vector $\mathbf{v}_1$ is used to include the elements just below and above the diagonal of the matrix, and $\mathbf{v}_2$ is used to include the elements twice removed from the diagonal elements of the matrix **A**. While the **A** matrix has n diagonal elements, it has $n - 1$ elements just below and above the diagonal and $n - 2$ elements twice removed from the diagonal. The vector **d** consists of the elements along the diagonal of the **A** matrix with the semicolon separating the elements of the vector corresponding to points inside the well from the elements corresponding to points outside the well. The function diag used to define the **A** matrix has a number of functions in MATLAB. If the argument of diag is a matrix, diag gives the diagonal elements of the matrix. When diag has a single argument that is a vector with n elements, the function diag returns an $n \times n$ matrix with those elements along the diagonal. Matrices with the element below or above the diagonal can be produced by giving an additional integer which gives the position of the vector below or above the diagonal. Thus, diag($\mathbf{v}_1$,-1) and diag($\mathbf{v}_1$,1) return matrices with the elements of $\mathbf{v}_1$ along the locations one step below and above the diagonal, while diag($\mathbf{v}_2$,-2) and diag($\mathbf{v}_2$, 2) returns a matrix with the elements of $\mathbf{v}_2$ along the second locations below and above the diagonal, and diag(d) returns an $n \times n$ matrix with the

elements **d** along the diagonal. The **A** matrix is the sum of these five matrices. One can readily confirm that MATLAB Program CC.1 produces the same matrix as the matrix shown in the text.

This program produces the output

```
GroundState =

    0.0207
    0.1517
    0.3253
    0.3900
    0.4594
```

With a five point grid ($n = 5$), the two lowest eigenvalues are 0.0207 eV and 0.1517 eV, while the lowest eigenvalues for $n = 20$, are 0.290 eV and 0.2407 eV. The lowest eigenvalue thus approaches the very accurate eigenvalue 0.0325 eV given in the third chapter.

Thus far, we have shown how differential equations can be converted into sets of linear equations or equivalently into matrix equations. The process of converting continuous differential equations into sets of linear equations or equivalently into equations with matrices is called *discretization*. In addition to making finite difference approximations to the derivatives, differential equations can be discretized using the *spline collocation* or the *finite element* methods. We shall now show how to solve differential equations and eigenvalue problems using the spline collocation methods.

For solving the Schrödinger equation using spline collocation, we shall here use a continuous differentiable basis of piecewise Hermite cubic splines. To define the spline functions for a single variable χ, we introduce the grid

$$\chi_i = i * \delta, \text{with } \; i = 0, 1, \ldots, n,$$

where δ is again the grid spacing. The points χ_i are called *nodes*.

The basis functions $v_i(\chi)$ and $s_i(\chi)$ for $1 \leq i \leq N-1$ are defined by the equations

$$v_i(\chi) = \begin{cases} \dfrac{1}{\delta^3}(\chi - \chi_{i-1})^2[\delta + 2(\chi_i - \chi)], & \chi_{i-1} \leq \chi \leq \chi_i \\ \dfrac{1}{\delta^3}(\chi_{i+1} - \chi)^2[\delta + 2(\chi - \chi_i)], & \chi_i \leq \chi \leq \chi_{i+1} \\ 0, & \text{otherwise} \end{cases} \tag{CC.11}$$

$$s_i(\chi) = \begin{cases} \dfrac{1}{\delta^3}(\chi - \chi_{i-1})^2(\chi - x_i), & \chi_{i-1} \leq \chi \leq \chi_i \\ \dfrac{1}{\delta^3}(\chi_{i+1} - \chi)^2(\chi - x_i), & \chi_i \leq \chi \leq \chi_{i+1} \\ 0, & \text{otherwise,} \end{cases} \tag{CC.12}$$

The spline functions $v_i(\chi)$ and $s_i(\chi)$ together with the special functions at the ends of the entire region are shown in Fig. CC.1.

The values of these functions and their first derivatives at the nodal points follow immediately from eqs. (CC.11) and (CC.12)

$$v_i(\chi_j) = \delta_{ij}, \;\; v_i'(\chi_j) = 0, \;\; s_i(\chi_j) = 0, \;\; s_i'(\chi_j) = \delta_{ij}\frac{1}{h}, \tag{CC.13}$$

where δ_{ij} is the Kronecker delta function. These conditions are sufficient to determine the polynomials within each interval.

An approximate solution of an ordinary differential equation can be expressed as a linear combination of Hermite splines

$$u(\chi) = \sum_{i=0}^{N}[\alpha_i v_i(\chi) + \beta_i s_i(\chi)], \tag{CC.14}$$

For the Gauss quadrature points, ξ_{i1} and ξ_{i2}, within the i-th interval, four functions of the basis, v_{i-1}, s_{i-1}, v_i and s_i, have nonzero values. These functions are illustrated in Figure CC.2.

The Gauss points for cubic polynomials are given by the formulas

$$\xi_{i1} = x_{i-1} + \frac{3 - \sqrt{3}}{6}\delta, \qquad \xi_{i2} = x_{i-1} + \frac{3 + \sqrt{3}}{6}\delta \tag{CC.15}$$

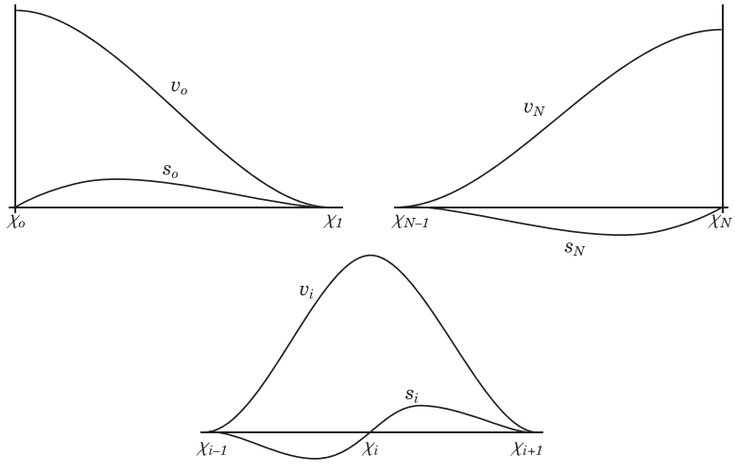

FIGURE CC.1 These three graphs show and v_i and s_i for $1 \leq i \leq N-1$ (bottom) together with the special functions v_0, s_0, v_N, s_N.

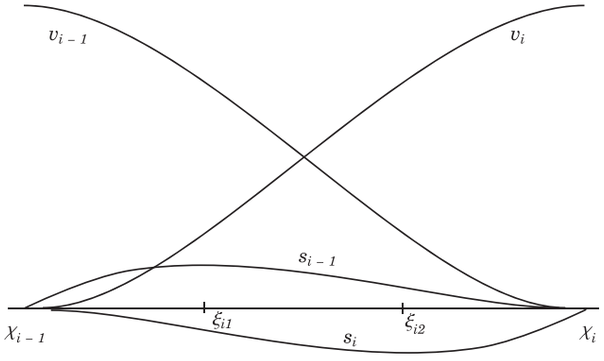

FIGURE CC.2 Four of the splines are nonzero at the Gauss points within the interval $[x_{i-1}, x_i]$.

Using (CC.14), the solution can be evaluated at the Gauss points. We have

$$u(\xi_{i1}) = b_{11}\alpha_{i-1} + b_{12}\beta_{i-1} + b_{13}\alpha_i + b_{14}\beta_i, \tag{CC.16}$$

$$u(\xi_{i2}) = b_{21}\alpha_{i-1} + b_{22}\beta_{i-1} + b_{23}\alpha_i + b_{24}\beta_i. \tag{CC.17}$$

The coefficient matrix b_{ij}, which correspond to the values of the spline basis functions at the collocations points, can be evaluated using Eqs. (CC.11) and (CC.12). Equations (CC.16) and (CC.17) can be written in matrix form as

$$\begin{bmatrix} u(\xi_{i1}) \\ u(\xi_{i2}) \end{bmatrix} = \begin{bmatrix} b_{11} & b_{12} & b_{13} & b_{14} \\ b_{21} & b_{22} & b_{23} & b_{24} \end{bmatrix} \begin{bmatrix} \alpha_{i-1} \\ \beta_{i-1} \\ \alpha_i \\ \beta_i \end{bmatrix}. \tag{CC.18}$$

Treating each of the subintervals in a similar manner, the vector $\mathbf{u_G}$ consisting of the values of the approximate solution at the Gauss points can be written as the product of a matrix times a vector

$$\mathbf{u_G} = \mathbf{Bu}, \tag{CC.19}$$

where

$$\mathbf{u} = [\alpha_0, \beta_0, \alpha_1, \beta_1, \ldots, \alpha_N, \beta_N]^{\mathrm{T}}, \tag{CC.20}$$

$$\mathbf{u_G} = [u(\xi_{11}), u(\xi_{12}), u(\xi_{21}), u(\xi_{22}), \ldots, u(\xi_{N1}), u(\xi_{N2})]^{\mathrm{T}} \tag{CC.21}$$

and **B** is a rectangular matrix having $2N + 2$ columns and $2N$ rows. **B** has the structure

$$\mathbf{B} = \begin{bmatrix} \boxed{\mathbf{B}_1} & & & \\ & \boxed{\mathbf{B}_2} & & \\ & & \ddots & \\ & & & \boxed{\mathbf{B}_{N-1}} \\ & & & & \boxed{\mathbf{B}_N} \end{bmatrix}. \tag{CC.22}$$

Two adjacent blocks $\mathbf{B}_i$ and $\mathbf{B}_{i+1}$ overlap in two columns.

The first and second derivatives of the approximate solution can be represented by matrices with the same block structure. We express the vector $\mathbf{u}'_{\mathbf{G}}$ consisting of the values of the first derivative of the solution at the Gauss points as

$$\mathbf{u}'_{\mathbf{G}} = \mathbf{C}\mathbf{u}, \tag{CC.23}$$

where **u** is given by Eq. (CC.20) and

$$\mathbf{u}'_{\mathbf{G}} = [u'(\xi_{11}), u'(\xi_{12}), u'(\xi_{21}), u'(\xi_{22}), \ldots, u'(\xi_{N1}), u'(\xi_{N2})]^{\mathrm{T}}. \tag{CC.24}$$

The negative of the second derivative of the function at the Gauss points can be written as

$$-\mathbf{u}''_{\mathbf{G}} = \mathbf{A}\mathbf{u}. \tag{CC.25}$$

As for the matrix **B**, matrices **C** and **A** have $N + 2$ columns and $2N$ rows, and the Hamiltonian of the system will have this property. The matrices may be converted into square matrices by adding a single row to the top and bottom of these matrices. These additional rows may be chosen to impose the boundary conditions. In the case of homogeneous Dirichlet or Neumann boundary conditions, the boundary conditions can also be imposed by removing from the matrices **B**, **C**, and **A** the columns corresponding to the zero value of the functions or its derivatives. For the even solutions, the derivative of the wave function at the center of the well is zero and the wave function will be zero at the edge of the physical region. The boundary conditions can then be imposed by deleting the second column and the second to last column of the matrices. The matrices then have the same number of rows and columns.

The collocation matrix, which represents the operators on the left-hand side of Eqs. (CC.2) and (CC.3), is defined differently for points inside and outside the potential well. For points inside the well, the collocation matrix depends only upon the **A** matrix representing the negative of the second derivative of the wave function

$$\mathbf{C}_1 = \mathbf{A}.$$

Outside the well, the collocation matrix depends upon both the **A** and **B** matrices according to the equation

$$\mathbf{C}_2 = \mathbf{A} + \mathcal{V}_0 * E_0 B.$$

In the MATLAB program to be described shortly, the $\mathbf{C}_1$ and $\mathbf{C}_2$ matrices are brought together to form the collocation matrix with the command

$$\mathbf{C}_{mat} = [\mathbf{C}_1; \mathbf{C}_2]. \tag{CC.26}$$

According to eq. (CC.19) a vector consisting of the values of the wave function at the collocation points is related to a vectors with the spine coefficient by an equation of the form

$$\mathbf{v} = \mathbf{B}_{mat}\mathbf{u}. \tag{CC.27}$$

Using eqs(CC.26) and (CC.27), the eigenvalue equation for an electron in a finite well may be represented by the following matrix equation

$$\mathbf{C}_{mat}\mathbf{u} = E_0 \epsilon \mathbf{B}_{mat}\mathbf{u}. \tag{CC.28}$$

This last equation with matrices on both the left- and right-hand sides defines a generalized eigenvalue problem.

To obtain a standard eigenvalue problem, we first recall that the boundary conditions may be imposed by deleting two columns from the matrices. After removing the two appropriate columns, $\mathbf{B}_{mat}$ then has the same number of rows and columns and may be inverted. We may then use eq. (CC.27) to write the generalized eigenvalue equation (CC.28) in the form of the following standard eigenvalue equation

$$\mathbf{L}_{mat}\mathbf{v} = E_0 \epsilon \mathbf{v}, \tag{CC.29}$$

where

$$\mathbf{L}_{mat} = \mathbf{C}_{mat}\mathbf{B}_{mat}^{-1}. \tag{CC.30}$$

The matrix eigenvalue equation may thus be written in two equivalent forms. Eq.(CC.28) is a generalized eigenvalue equation, and eq. (CC.29) is a standard eigenvalue equation. MATLAB has routines that can be used to solve either of these two kinds of equations. The more stable approach is to solve eq. (CC.28) rather than to solve eq. (CC.29). The collocation matrix, $\mathbf{C}_{mat}$, which appears in eq. (CC.28), is a banded matrix for which MATLAB has very efficient subroutines, while the matrix, $\mathbf{L}_{mat}$, is in general a dense matrix. The MATLAB Program CC.2 finds the lowest eigenvalues for a finite well by solving eq. (CC.28).

MATLAB Program CC.2

A MATLAB program for finding the eigenvalues and eigenvectors for an electron moving in a finite well using the spline collocation method.

```
xmax=20;
L=5;
n=4;
delta=xmax/n;
deltas=delta*delta;
n1=fix(L/delta);
n2=n-n1;
e0=1.759;
v0=0.3;

% Construct B matrix
p1=(9 -4*sqrt(3))/18;
p2=(9+4*sqrt(3))/18;
p3=(3-sqrt(3))/36;
p4=(3+sqrt(3))/36;

B=[p2 p4 p1 -p3; p1 p3 p2 -p4]

% Construct C matrix
p5=2*sqrt(3);

C=[1 -1/p5 -1 1/p5; 1 1/p5 -1 -1/p5]/delta

% Construct A matrix
p6=sqrt(3)-1;
p7=sqrt(3)+1;

A=[p5 p7 -p5 p6;-p5 -p6 p5 -p7]/deltas

% Construct Collocation matrix
C1=zeros(2*n1, 2*n+2);
for row=1:2:2*n1-1
  C1(row:row+1, row:row+3)=A;
end
C2=zeros(2*n2,2*n+2);
for row=1:2:2*n2
  C2(row:row+1,row+2*n1:row+2*n1+3)=A+v0*e0*B;
end
Colmat=[C1;C2];
Colmat(:,2)=[];
Colmat(:,2*n)=[];
% Construct B matrix
Bmat=zeros(2*n, 2*n+2);
for row=1:2:2*n-1
```

```
  Bmat(row:row+1, row:row+3)=B;
end
Bmat(:,2)=[];
Bmat(:,2*n)=[];
% Construct eigenvalue matrix
V =eig(Colmat,Bmat);
GroundState = sort(V)/e0
```

As before, the first four lines of MATLAB Program CC.2 define the length of the physical region (xmax), the χ coordinate of the edge of the well (L), the number of intervals (n), and the step size (delta). The MATLAB function "fix" in the next line of the program rounds the ratio "L/delta" to the integer toward zero. The integer $n1$ is the number of intervals within the well, and the integer $n2$ is the number of intervals outside the well. Each interval contains two collocation points. After defining the constant E_0, the program then defines the **A**, **B**, and **C** matrices, which are used to construct the collocation matrix $\mathbf{C}_{\text{mat}}$. MATLAB Program CC.2 produces the following output

```
GroundState =

    0.0334
    0.2547
    0.3292
    0.4069
    0.5724
    0.7212
    0.9315
    1.0941
```

With only four intervals, the program returns a lowest eigenvalue of 0.0334 eV. Since the potential well described in Section 2.3 is 0.3 eV deep, the second eigenvalue 0.2547 eV also corresponds to a bound state. Program C.2 converges very rapidly. For a grid with 20 intervals the program returns the lowest eigenvalues, 0.0342 and 0.2715 eV.

While MATLAB Program CC.2 gives the eigenvalues of an electron moving in a finite well, it does not give the eigenvectors. To obtain the eigenvectors, one can replace the last two statements of the program with the statements

```
[E V] =eig(Colmat,Bmat);
GroundState = diag(V)/e0
```

The program then produces the output

```
GroundState =

    1.0941
    0.0334
    0.9315
    0.7212
    0.5724
    0.2547
    0.4069
    0.3292
```

We obtain the same eigenvalues in a different order. The lowest eigenvalue is the second eigenvalue produced, while the next lowest eigenvalue is the sixth eigenvalue produced. To get the eigenvectors for the two lowest states, we first note that the eigenvector **v**, which is the solution of the simple eigenvalue equation (CC.29), is related to the eigenvector **u**, which is a solution of the general eigenvalue equation (CC.28) by Eq. (CC.27). One may thus get the wave function for the lowest state by multiplying $\mathbf{B}_{\text{mat}}$ times the second column of the matrix E produced by the last MALAB program, and one may get the wave function corresponding to the next lowest eigenvalue by multiplying $\mathbf{B}_{\text{mat}}$ times the sixth column of the matrix E. This whole process can, of course, be made more automatic.