Phys 410- Computational Physics- Project1

Oct 2016

Ali Izadi Rad

Problem 1- Single Well

Here we want to study the propagation method and re-derive the conditions for the bounds of states is single well.

We want to study the problem of finite potential wells with propagation method.

Generaly we can assume that the wave function has the form of sin and cos inside the well:

where is . We can find A and B coefficient based on and at some point. This is the essence of propagation method to find the wave function and it’s derivative based on it’s value at some point. If propagation be from point x=a to x=b then:

if we define c-b:=a then by finding A and B we have:

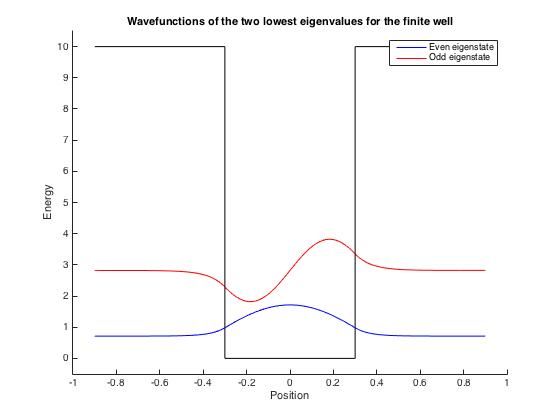
thus:

In general for region that classically is forbidden we expect that wave function decays exponentially thus this can be a hint for finding allowed energy bounds. The transcendal function can be

but there is simpler way to find the even and odd solutions and eigenvalue by looking to behavior of wave function. If we put the center of well at c=0 then for odd solutions, we expect that:

for even solution we expect that:

Which is in agreement by well-known solution for single well potential. Therefore, propagation method works!



We can also look in deferent way and say for even modes, wave function behaves like and then thus :

In the other hand, for odd mode, wae function is like then :

which is again in agreement with previous knowledges.

Problem2- Double Well

Now, we would like to use the propagator method to solve for the energy eigenvalues of the quantum double well potential. The picture here is that we have four walls, call them a,b,c,d from left to right. These four walls construct a finite, double-well each of width W, spacing S between them, and potential depth V0. We set and propagate the wave function along well from a to b. Then we propagate that from spacing (b to c) via forbidden propagator and then from c to d by normal propagator again:

Our function whose solution gives energy state is

general form of solution is not attractive, instead we can work on odd and even solution and apply the constraints on . Thus:

for odd mode wave function heaves like and for even mode .

Now by solving for odd mode we will have:

the following plot shows the location of roots based on intersections of left hand side and right one in above equation. As we can see we have seven eigenvalues.

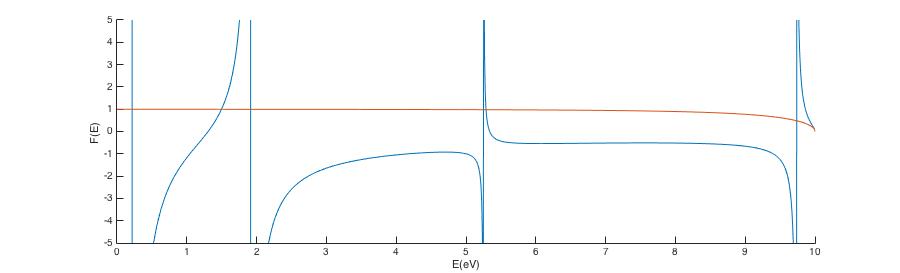


Figure Location of roots for double well problem (odd modes)

Similarly, for even mode we have:

only if

otherwise

the second solution is not valid and there is no root for that function

The following plot shows the locations or roots:

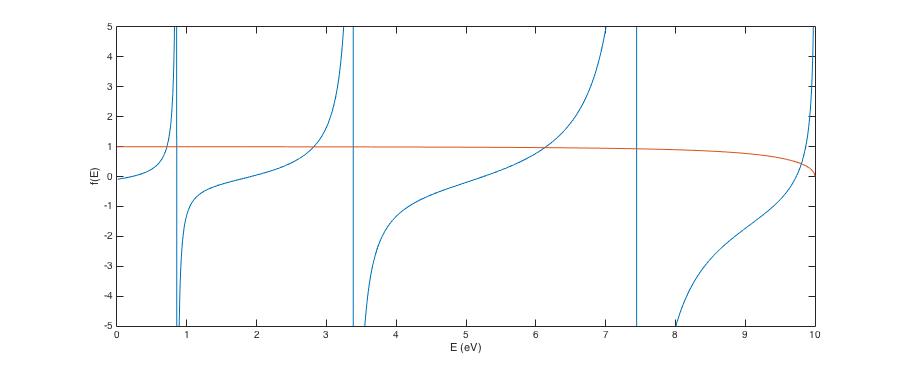
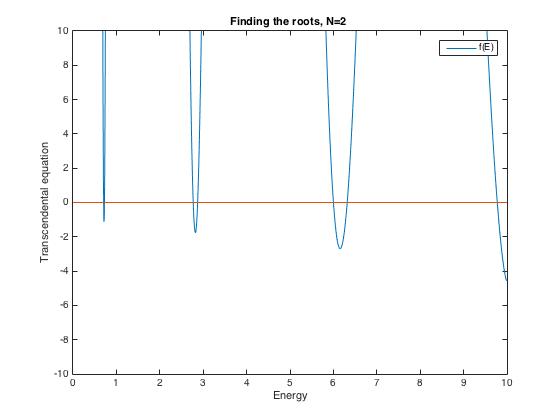


Figure locations of roots for even modes

As we can see again we have seven eigenvalues.

Now we try to find the locations of bound states by MATLAB programing:

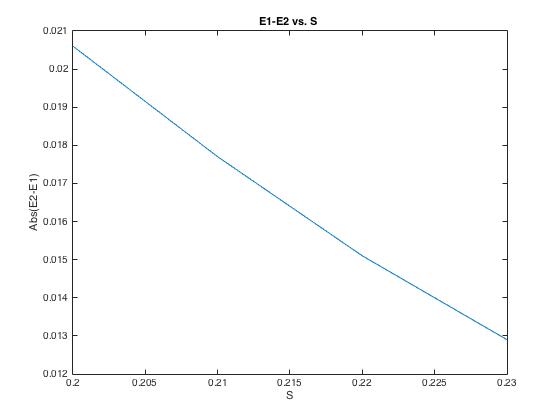


Here is the list of E1-E7 (all in eV):

0.7049, 0.7255, 2.7701, 2.8718 , 6.0008, 6.3181, 9.7760

In the next step, we study behavior of E2-E1 vs separation of two wells.

The observation is that energy deference decreases as separation increases:



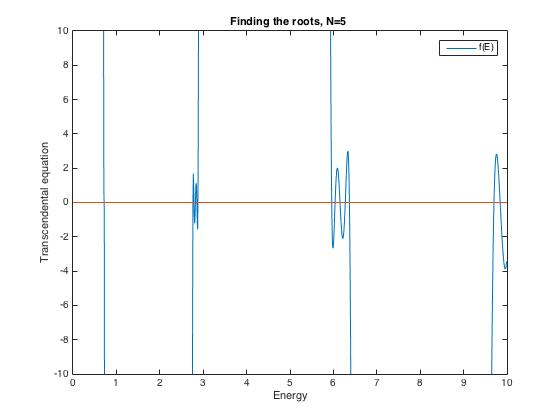
The following codes are for N=2 well:

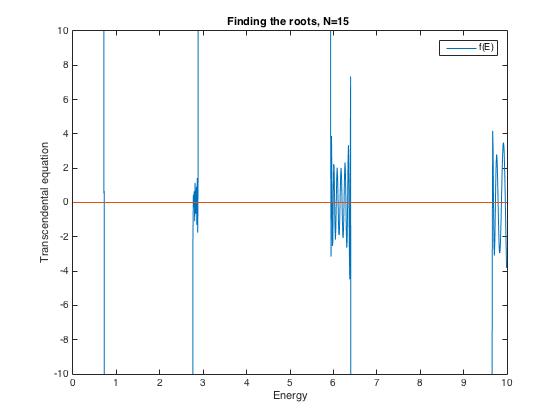
|  |
| --- |
| function root=DoubleWell2(xL,xR)    %%%%%%%%%% Constants %%%%%%%%%%%%%  % configuration of doubell well  % ------ a -b- a ------  % \\\\\| | | |\\\\\\\  % \\\\\| | | |\\\\\\\  % ------ ------  a=0.6; % Well width  m=1; % mass of particle per mass of electron  V0=10; % potential value (eV)  b=0.2; % Sepration of two wells  hbar2 = 0.076199682; % planck constant square    %%%%%%% Alpha and Beta %%%%%%%%%    alpha = @(E) sqrt(2\*m\*E/hbar2);  beta = @(E) sqrt(2\*m\*(V0-E)/hbar2);      %%%%%% Definination of Propagator matrix %%%%%%%      % Normal Propagator  P=@(E)[ cos(alpha(E).\*(a)) , (1./alpha(E)).\*sin(alpha(E).\*(a));  -alpha(E).\*sin(alpha(E).\*(a)), cos(alpha(E).\*(a))];    % wave function of left point of well , \psi(a)  Psi\_ini=@(E)[1;  beta(E)] ;    % Propagator for forbidden area  P\_forbiden=@(E)[ cosh(beta(E).\*(b)) , (1./beta(E)).\*sinh(beta(E).\*(b));  beta(E).\*sinh(beta(E).\*(b)), cosh(beta(E).\*(b))];      %%%%%%% transcendental equation, f(E) %%%%%%%%%%%%%%%%%%%    psicomp\_doub=@(E) P(E)\*P\_forbiden(E)\*P(E)\*Psi\_ini(E); % wave function in right hand side after full propagating ,\psi(d)    subindex = @(AB,r,c) AB(r,c); %# An anonymous function to index a matrix  psi\_doub=@(E) subindex(psicomp\_doub(E),1,1);  psi\_p\_doub=@(E) subindex(psicomp\_doub(E),2,1);  f=@(E) psi\_p\_doub(E)+beta(E).\*psi\_doub(E);      %%%%%%%%%% Plotting %%%%%%%%%%%  E0 = linspace(0,V0,1000);    for ii=1:length(E0)  Y(ii)=f(E0(ii));  end    plot(E0,Y,E0,zeros(size(E0)))    ylim([-10 10])  title('Finding the roots, N=2 '); xlabel('Energy'); ylabel('Transcendental equation');  legend('f(E)')      % Simple bisection for finding roots of functions of a single variable.  % Use as starting point, add comments and elaborations of basic code as needed.  %function root=bisection(xL,xR)  %xL=9.6; xR=10;    if f(xL)\*f(xR)>0; display ('Initial interval contains even number of roots');return;end  tolerance = 1e-12;  accuracy=100\*tolerance;    while accuracy> tolerance  xmiddle=(xL+xR)/2;  if f(xL)\*f(xmiddle)<0; xR=xmiddle;else xL=xmiddle;end;  accuracy=abs(f(xmiddle));  end    root=xmiddle  end |

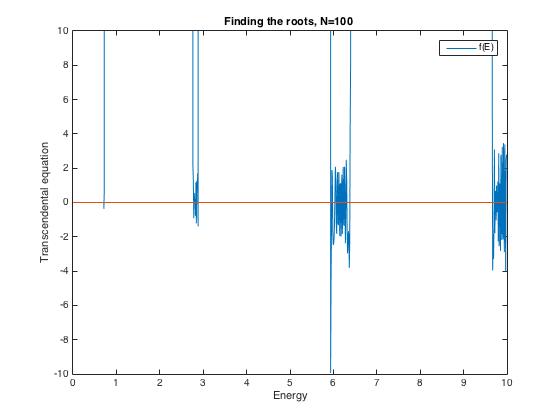
Problem3- General N well

We now want to generalize to the case of the finite N-well. Unlike the above two problems, this general problem is absolutely unbearable analytically, therefore we place the brunt of the work on MATLAB

We have plotted bounds states for N=5,15 and 100 to show the behavior of eigenvalue of energies as N increases:







We can see exactly what the effect of increasing N has on the energy levels. It contributes a sort of “fine structure” to the single well potential. In other words, as N increases, the roots of F(E) stay clustered more or less in the same region, but there exists more and more roots within this region as N rises. So what will the behavior be as We can infer that in this limit, the roots will begin to span a compact continuum, more or less centered on where the isolated root for the single well was in that region. This is reminiscent of the “band structure” of a crystal. This should come as no surprise because an infinite chain of finite wells is a familiar model of a crystal. To sum up we observes that splitting of energy decreases as N increases and number of nearby energy levels is roughly proportional to where is energy of band structures that we mentioned above.

**Fast Algorithm:**

**In the N-well problem, we need to multiplies propagation matrixes together. The number of multiplication is order of magnitude of N. but this is not the case. We can save the calculation up to Log(N). to see that, imagine:**

**We can rewrite this one as a following:**

**we can calculate just once and then save it. After that we can multiplies together instead of . In general if then we need multipication.**

The code of this part is at the following:

|  |
| --- |
| function root=DoublewellN(N,xL,xR)    %%%%%%%%%% Constants %%%%%%%%%%%%%    a=0.6; % Well width  m=1; % mass of particle per mass of electron  V0=10; % potential value (eV)  b=0.23; % Sepration of two wells  hbar2 = 0.076199682; % planck constant square    %%%%%%% Alpha and Beta %%%%%%%%%    alpha = @(E) sqrt(2\*m\*E/hbar2);  beta = @(E) sqrt(2\*m\*(V0-E)/hbar2);      %%%%%% Definination of Propagator matrix %%%%%%%      % Normal Propagator  P=@(E)[ cos(alpha(E).\*(a)) , (1./alpha(E)).\*sin(alpha(E).\*(a));  -alpha(E).\*sin(alpha(E).\*(a)), cos(alpha(E).\*(a))];    % wave function of left point of well , \psi(a)  Psi\_ini=@(E)[1;  beta(E)] ;    % Propagator for forbidden area  P\_forbiden=@(E)[ cosh(beta(E).\*(b)) , (1./beta(E)).\*sinh(beta(E).\*(b));  beta(E).\*sinh(beta(E).\*(b)), cosh(beta(E).\*(b))];        %%%%%%%%% Building \psi and \psi prim and transcendental equation, f(E%%%%%%    psicomp=@(x) Psi\_ini(x);        a=0.6;  b=0.3;    for ii=1:(2\*N-1)  if mod(ii,2) % if ii is odd  psicomp=@(E) P(E)\*psicomp(E);  else  psicomp=@(E) P\_forbiden(E)\*psicomp(E) ;  end  end  subindex = @(AB,r,c) AB(r,c); %# An anonymous function to index a matrix  psi\_Nwell=@(E) subindex(psicomp(E),1,1);  psi\_p\_Nwell=@(E) subindex(psicomp(E),2,1);    f=@(E) psi\_p\_Nwell(E)+beta(E).\*psi\_Nwell(E);    %%%%%%%%%% Plotting %%%%%%%%%%%  E0 = linspace(0,V0,1000);    for ii=1:length(E0)  Y(ii)=f(E0(ii));  end    plot(E0,Y,E0,zeros(size(E0)))    ylim([-10 10])  title('Finding the roots, N=100 '); xlabel('Energy'); ylabel('Transcendental equation');  legend('f(E)')      % Simple bisection for finding roots of functions of a single variable.  % Use as starting point, add comments and elaborations of basic code as needed.  %function root=bisection(xL,xR)  %xL=0.715; xR=0.75;    if f(xL)\*f(xR)>0; display ('Initial interval contains even number of roots');return;end  tolerance = 1e-12;  accuracy=100\*tolerance;    while accuracy> tolerance  xmiddle=(xL+xR)/2;  if f(xL)\*f(xmiddle)<0; xR=xmiddle;else xL=xmiddle;end;  accuracy=abs(f(xmiddle));  end    root=xmiddle          end |

Fast Algorithm:

|  |
| --- |
| function root=FastDoublewellN(m,xL,xR)  % N=2^m  %%%%%%%%%% Constants %%%%%%%%%%%%%    a=0.6; % Well width  m=1; % mass of particle per mass of electron  V0=10; % potential value (eV)  b=0.23; % Sepration of two wells  hbar2 = 0.076199682; % planck constant square    %%%%%%% Alpha and Beta %%%%%%%%%    alpha = @(E) sqrt(2\*m\*E/hbar2);  beta = @(E) sqrt(2\*m\*(V0-E)/hbar2);      %%%%%% Definination of Propagator matrix %%%%%%%      % Normal Propagator  P=@(E)[ cos(alpha(E).\*(a)) , (1./alpha(E)).\*sin(alpha(E).\*(a));  -alpha(E).\*sin(alpha(E).\*(a)), cos(alpha(E).\*(a))];  % wave function of left point of well , \psi(a)  Psi\_ini=@(E)[1;  beta(E)] ;    % Propagator for forbidden area  P\_forbiden=@(E)[ cosh(beta(E).\*(b)) , (1./beta(E)).\*sinh(beta(E).\*(b));  beta(E).\*sinh(beta(E).\*(b)), cosh(beta(E).\*(b))];  % PP is the production of P\*P\_forbiden and it is like A^2 is our example: B=A\*A\*...A --> B=A^2\*A^2\*...A^2  PP=@(E) [cos(alpha(E).\*(a)).\*cosh(beta(E).\*(b))+ (1./alpha(E)).\*sin(alpha(E).\*(a)).\* beta(E).\*sinh(beta(E).\*(b)), cos(alpha(E).\*(a)).\*(1./beta(E)).\*sinh(beta(E).\*(b))+ (1./alpha(E)).\*sin(alpha(E).\*(a)).\*cosh(beta(E).\*(b)); -alpha(E).\*sin(alpha(E).\*(a)).\*cosh(beta(E).\*(b))+cos(alpha(E).\*(a)).\*beta(E).\*sinh(beta(E).\*(b)), -alpha(E).\*sin(alpha(E).\*(a)).\*(1./beta(E)).\*sinh(beta(E).\*(b))+cos(alpha(E).\*(a)).\* cosh(beta(E).\*(b))];      %%%%%%%%% Building \psi and \psi prim and transcendental equation, f(E%%%%%%    psicomp=@(x) P(x)\*Psi\_ini(x);      % N=2^m      for ii=1:m    psicomp=@(E) PP(E)\*psicomp(E) ;    end  subindex = @(AB,r,c) AB(r,c); %# An anonymous function to index a matrix  psi\_Nwell=@(E) subindex(psicomp(E),1,1);  psi\_p\_Nwell=@(E) subindex(psicomp(E),2,1);    f=@(E) psi\_p\_Nwell(E)+beta(E).\*psi\_Nwell(E);    %%%%%%%%%% Plotting %%%%%%%%%%%  E0 = linspace(0,V0,1000);    for ii=1:length(E0)  Y(ii)=f(E0(ii));  end    plot(E0,Y,E0,zeros(size(E0)))    ylim([-10 10])  title('Finding the roots, N=100 '); xlabel('Energy'); ylabel('Transcendental equation');  legend('f(E)')      % Simple bisection for finding roots of functions of a single variable.  % Use as starting point, add comments and elaborations of basic code as needed.  %function root=bisection(xL,xR)  %xL=0.715; xR=0.75;    if f(xL)\*f(xR)>0; display ('Initial interval contains even number of roots');return;end  tolerance = 1e-12;  accuracy=100\*tolerance;    while accuracy> tolerance  xmiddle=(xL+xR)/2;  if f(xL)\*f(xmiddle)<0; xR=xmiddle;else xL=xmiddle;end;  accuracy=abs(f(xmiddle));  end    root=xmiddle          end |