Allowed Energy Eigenvalues in a Custom Potential

Chris Newey

December 8, 2018

Abstract Simulating quantum physics numerically we see how an electron flows through a crystal structure. We begin by building a custom periodic potential and progress to solving for the allowed energy eigenvalues. The code can be found at the end and is C, the data is redirected to .dat files using > | on the command line, and the plots are created using Gnuplot. Analyzing the results, we see the bands of allowed energy states for the periodic potential where each band has the same number of allowed energy states as the number of wells and the highest band resembles a free particle.

1. Potential The first portion of the project involves creating a function, `potential()` in order to create data for a plot of the custom potential. The project requires 25 wells that are -21eV deep centered around $x = 0$. The dimensions require .625nm spacing between the wells and a well width of .6nm. The starting and ending tails need to be wide enough to launch and end the wave function. Using the idea of similar triangles to achieve the correct dimensions:

$$\frac{.625/2}{21+x} = \frac{.025/2}{x} \tag{1}$$

Solving Eqn. (1) for x yields $x = .875$. Next, this result is used to place the bottom of the wells at -21 eV. Cutting the wells off at $y = 0$ with an if statement results in the appropriate well dimensions. The periodic potential is made primarily using:

```
period = .625e-9;
periodhalf = 3.125e-10;
amp = -43.75;
y = amp*fabs(round(((x-periodhalf)/period))
          - ((x-periodhalf)/period))+1.0;
```

The well structure can be seen in Figures 1 and 2.

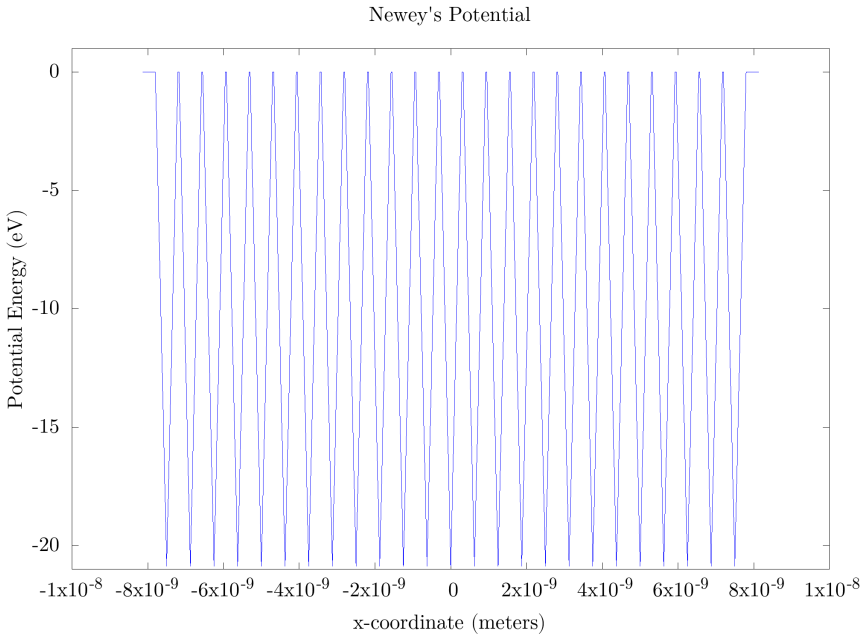

Figure 1: Custom potential of 25 periodic wells.

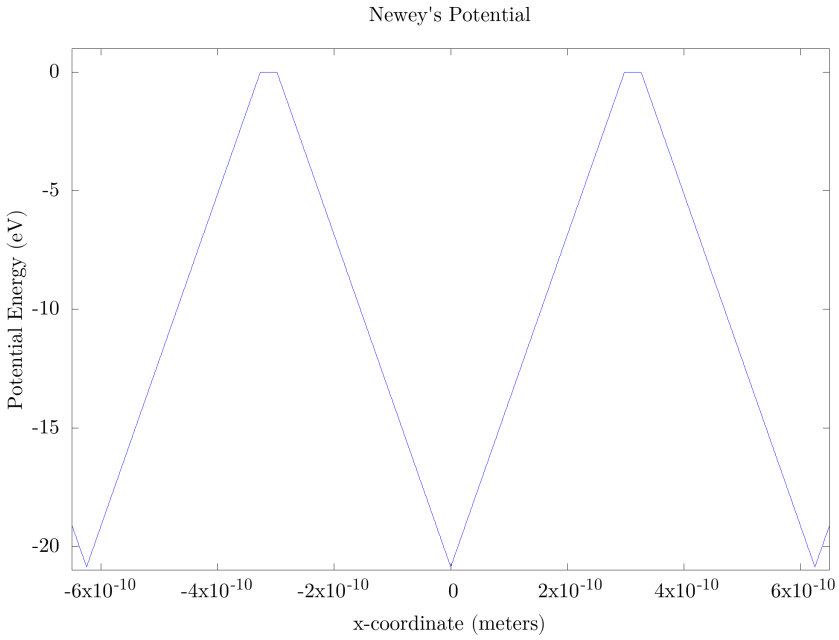

Figure 2: Close up of Custom potential.

2. Schrodinger Now we create `schroedinger()`, which holds the system of first order derivatives representing Schroedinger's equation to be used by `rk4()`. The system is:

```
yp[0] = y[1];
yp[1] = constant*m*(potential(x)-energy)*y[0];
```

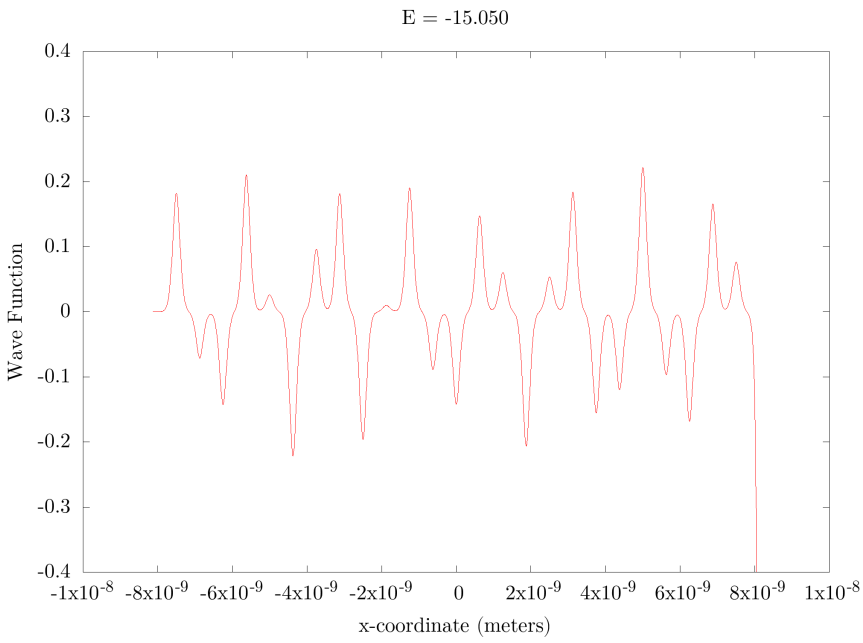

Figure 3: Guessed Wave Function E = -15.050

The main sets up the initial conditions:

$$\psi(x_{init}) = e^{-ka} \tag{2}$$

and its derivative:

$$\frac{d\psi(x_{init})}{dx} = ke^{-ka} \tag{3}$$

where a is the well width and the corresponding wave number,

$$k = \sqrt{\frac{-2mE}{\hbar^2}} \tag{4}$$

The initial conditions are for a given energy value, E, which is input by the user on the command line. The increments to be used for part 2 and the remainder of the project by `potential()` are xstart = -8.125e-9, xstop = 8.125e-9, and xinc = 6.0e-12.

Finally, the main uses `rk4(2,schroedinger,xstart,xstop,xinc,y);` So, we use `rk4()` to compute a wave function using `schroedinger()` for the `potential()` given the initial conditions. Checking the work at this stage involves a binary search for two allowed energy levels to an accuracy of 10^{-3}. The plots of the wave functions can be seen in Fig. 3 and Fig. 4.

3. Target Error Function Moving forward, the project calculates the target error function which allows us to plot a function whose roots are the allowed energy states of the custom potential well. The target error function is defined as:

$$f(E) = \psi(x_{final}) + \left(\frac{1}{k_1}\right)\frac{\partial\psi(x_{init})}{\partial x} \tag{5}$$

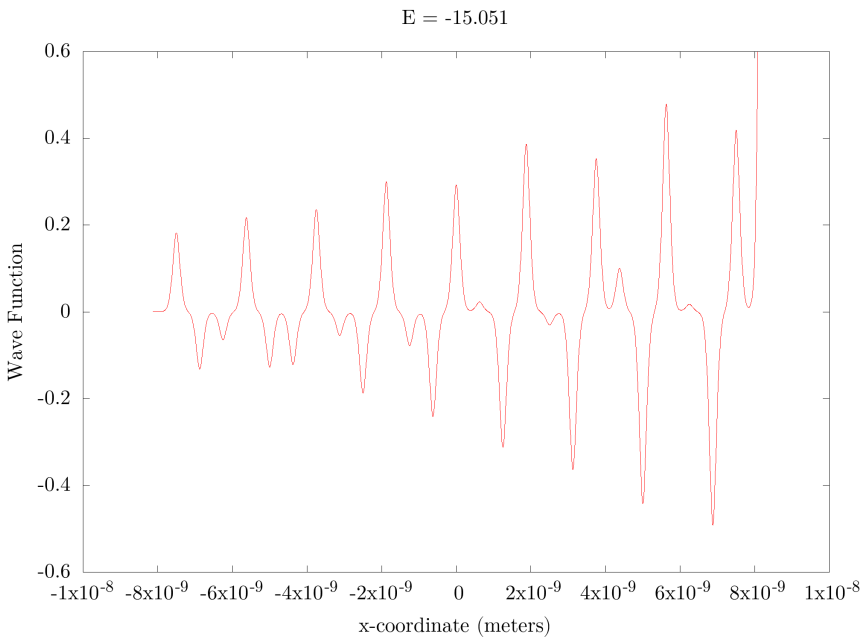

Figure 4: Guessed Wave Function E = -15.051

Now, the main program uses: `sweep(wave_shoot,-21.0,0.0,1.0e-4);` which accepts an argument of a trial energy from -21eV to 0eV in 1.0e-4eV increments, calculates the wave number k, and initializes the wave function and its derivative for the current energy. Then, `wave_shoot()` calls `rk4_quiet()` to solve `schroedinger()` through `potential()`. The final value of the solution for each energy is used to compute $f(E)$ which is returned to `sweep`. Plotting $f(E)$, Fig. 5 shows three energy bands. Interestingly, the last band resembles a free particle, showing how an electron flows through a crystal structure.

4. Energy Eigenvalues The fourth part of the project combines the previous sections but, instead of sweeping and plotting the error function we search for the roots which are the allowed energy states of the potential. The root finding method is regula falsi called as `sweep_regula_falsi(wave shoot,-21.0,0.0,1.0e-5,500,1.0e-9);` So, we use a sweep increment of 1.0e-5 eV, a root convergence tolerance of 1.0e-9eV, and 500 maximum iterations. Finally, the results are printed to standard output taking about 8 minutes to run on my new computer and about 12 minutes on my old laptop running Linux. The eigenvalues are in Table 1.

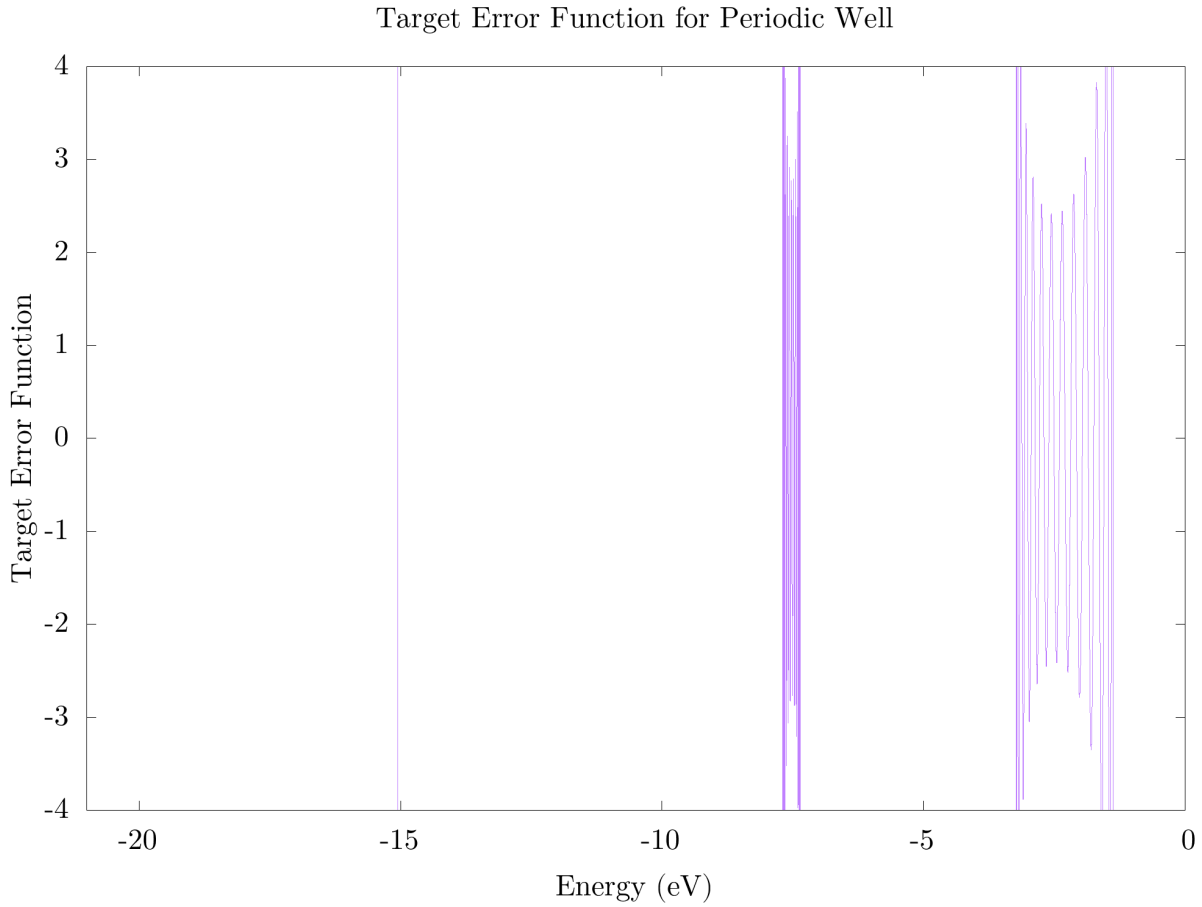

Figure 5: Target Error Function

Table 1: Energy Eigenvalues

Band 1	Band 2	Band 3
-15.05872129	-7.696615843	-3.23244419
-15.05860817	-7.693363235	-3.218120011
-15.0584365	-7.687977709	-3.194373163
-15.05826702	-7.680514667	-3.161416198
-15.05746601	-7.671037952	-3.1193454
-15.05711356	-7.659659005	-3.068683451
-15.05665662	-7.646489294	-3.009683633
-15.05617628	-7.631670571	-2.94276912
-15.05530357	-7.615367658	-2.868334425
-15.05473942	-7.597766004	-2.786987689
-15.05407737	-7.57907633	-2.699177705
-15.05338033	-7.559530666	-2.605500184
-15.05273765	-7.53938233	-2.506607764
-15.0519589	-7.518907143	-2.403153325
-15.05125556	-7.498396063	-2.295989215
-15.05057708	-7.478157281	-2.186023783
-15.04982688	-7.458509164	-2.074335626
-15.04925121	-7.43978011	-1.962392487
-15.04871396	-7.422290936	-1.851685693
-15.04822175	-7.406361169	-1.744400998
-15.04782838	-7.392292325	-1.643173599
-15.04734349	-7.380359651	-1.551373323
-15.04704093	-7.370808071	-1.472835554
-15.04683944	-7.363835268	-1.412088961
-15.04671401	-7.359590261	-1.373435151

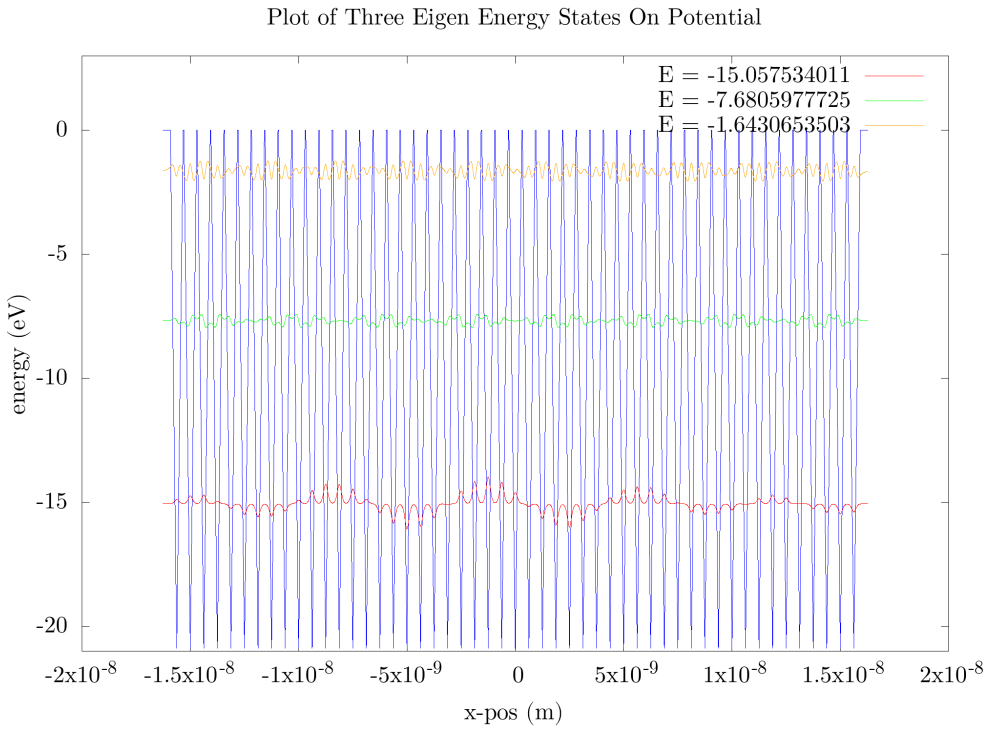

Figure 6: Plot of three allowed wave functions on top of potential.

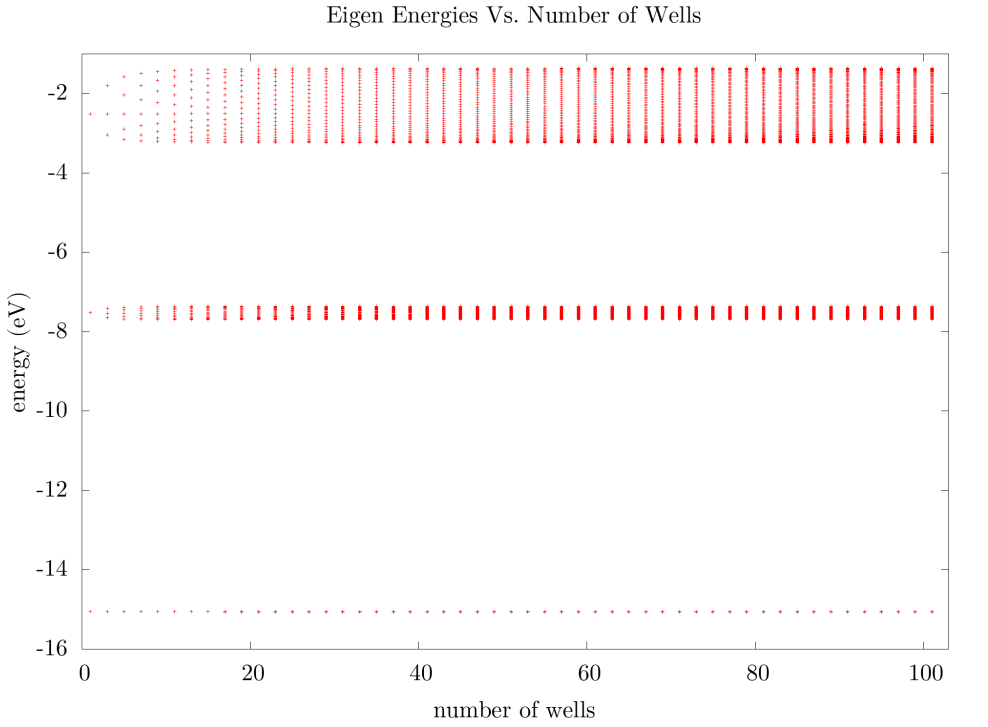

Figure 7: Plot of eigen energies for different numbers of potentials. Number of allowed states is equal to the number of wells using the same well dimensions and algorithms