Computational Analysis of the Time Dependent Schroedinger Equation

By: Andrew DeBenedictis, Anna Phillips, and Jen Radoff

Introduction

For this project, we examined solutions to the time dependent schrodinger equation using two schemes: (1) a simple finite difference scheme:

$$\left[\frac{\Psi_{j+1}^{n+1} - 2\Psi_{j}^{n+1} + \Psi_{j-1}^{n+1}}{(\Delta x)^2}\right] + V_j \Psi_j^{n+1} = i\left[\frac{\Psi_j^{n+1} - \Psi_j^{n}}{\Delta t}\right]$$

and (2) the Crank-Nicolson scheme:

$$\left(1 + \tfrac{1}{2} i H \Delta t\right)\psi_j^{n+1} = \left(1 - \tfrac{1}{2} i H \Delta t\right)\psi_j^{n}$$

Since the finite difference scheme failed to preserve normalization in the simple case of a free particle, we only present the Crank-Nicolson scheme for the various potentials. However, our code is capable of running both schemes for any potential.

Importing Python-generated data

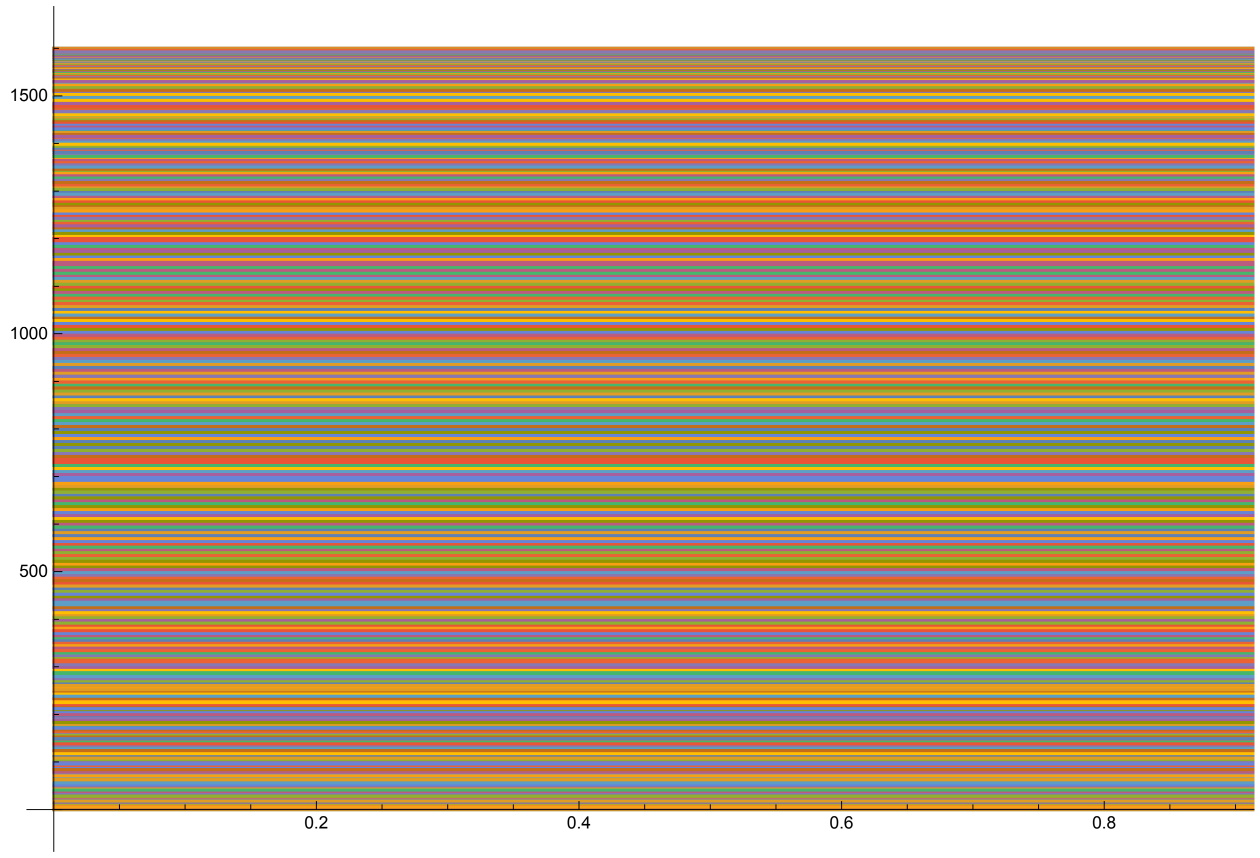
```
In[104]:= NotebookDirectory[]

Out[104]= /Users/Jennifer/Desktop/GradStudentsRock_TDSE/

In[105]:= path = NotebookDirectory[];
          cases = {"freeParticle", "squareWell", "harmonicOscillator", "triangle",
             "kronigPenney", "imagPotential", "complexPotential", "barrier1", "barrier2",
             "barrier3", "squareWellwithEigenState", "harmonicOscillatorEigenState"};
          scheme = {"_SFD", "_CN"};
          numberType = {"Real", "Imag"};
          periodicType = {"periodic", "nonPeriodic"};
          eigenState = {"eValues", "eVectors"};
```

Free Particle

Let’s look at the eigenvalues and eigenvectors to start out since these are scheme-independent:

In[111]:=
```
Clear[freeEVectorsRe, freeEVectorsIm];
Block[{caseChoice, periodicChoice, schemeChoice},
   caseChoice = 1;
   periodicChoice = 1;
    schemeChoice = 1;
   freeEValues = Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <>
       "/" <> periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <>
       "_" <> numberType⟦1⟧ <> "_" <> eigenState⟦1⟧ <> ".csv", "CSV"];
   Do[
     Evaluate[Symbol["freeEVectors" <> {"Re", "Im"}⟦i⟧]] =
       Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <>
         periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <> "_" <>
         numberType⟦i⟧ <> "_" <> eigenState⟦2⟧ <> ".csv", "CSV"];,
     {i, 2}];
   freeEVectors = Sqrt[freeEVectorsRe^2 + freeEVectorsIm^2];
  ];
```

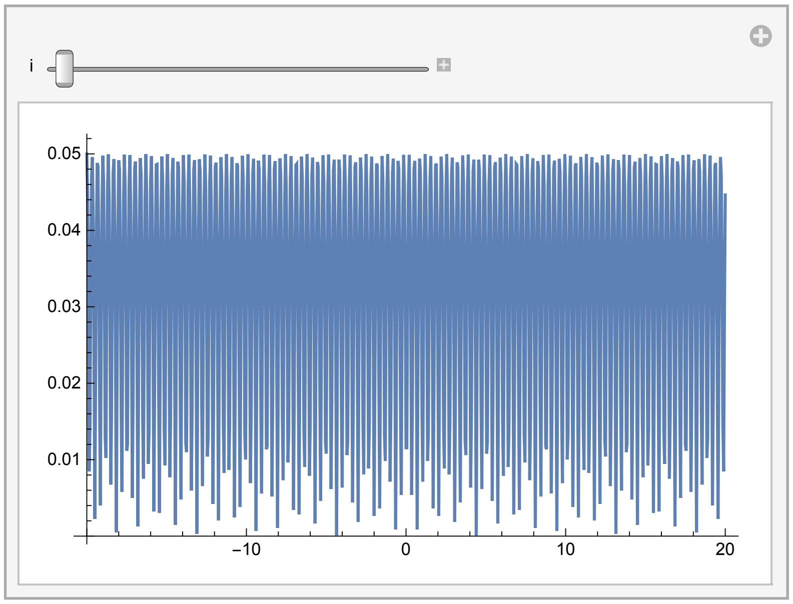
In[113]:=
```
Plot[freeEValues, {x, 0, 1}]
```

Out[113]=


Those are densely packed, as they should be!

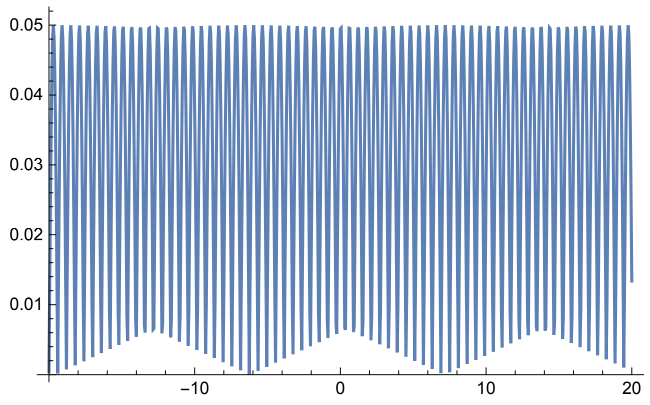
```
In[114]:= Manipulate[ListPlot[freeEVectors[[All, i]], Joined → True, DataRange → {-20, 20},
            AxesOrigin → {-20, 0}], {i, 1, Length[freeEVectors[[All, 1]]], 1}]
```

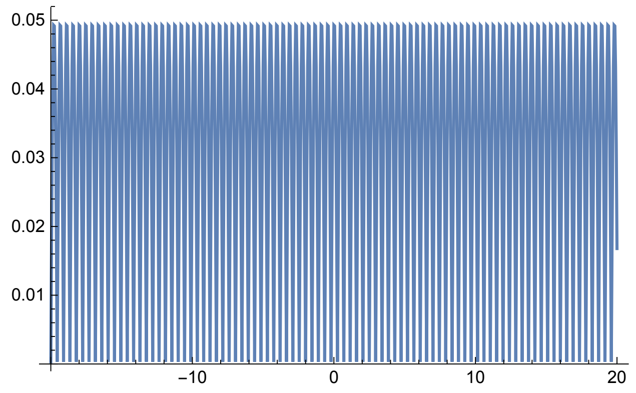
Out[114]=

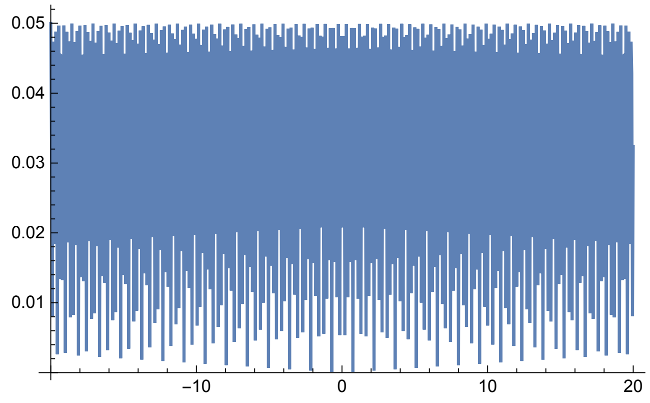


These look wonky, but that seems possible for the free particle--we should be able to create oscillatory waves at nearly any frequency that would be eigenstates. So it’s probably that the system is underdetermined, but nonetheless, our result appears plausible. A couple of representatitve images (100, 115, and 200) are below.

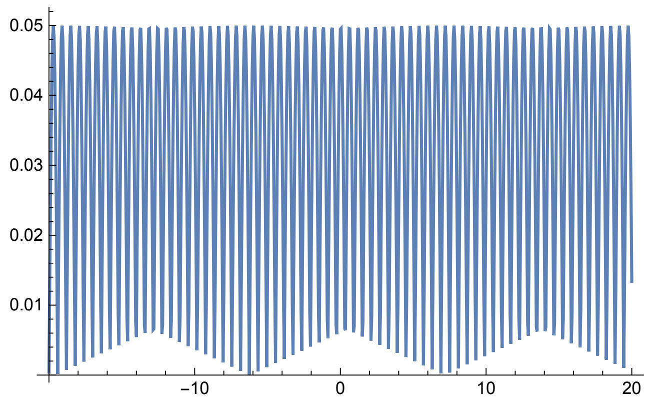
In[115]:=

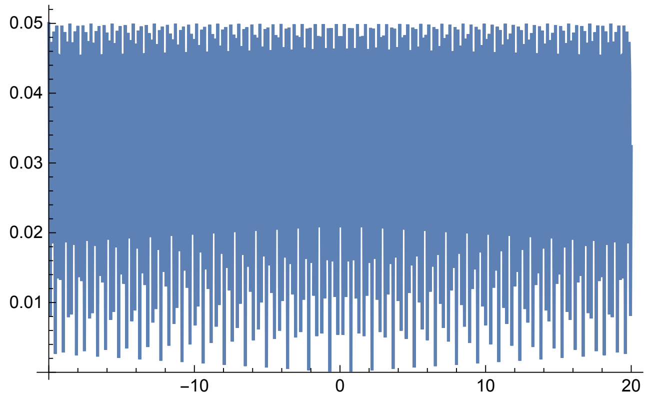

0.05
0.04
0.03
0.02
0.01
−10
0
10
20

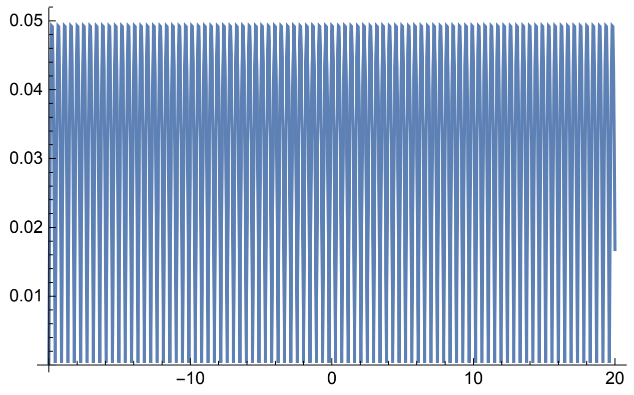

0.05
0.04
0.03
0.02
0.01
−10
0
10
20


0.05
0.04
0.03
0.02
0.01
−10
0
10
20

Out[115]=







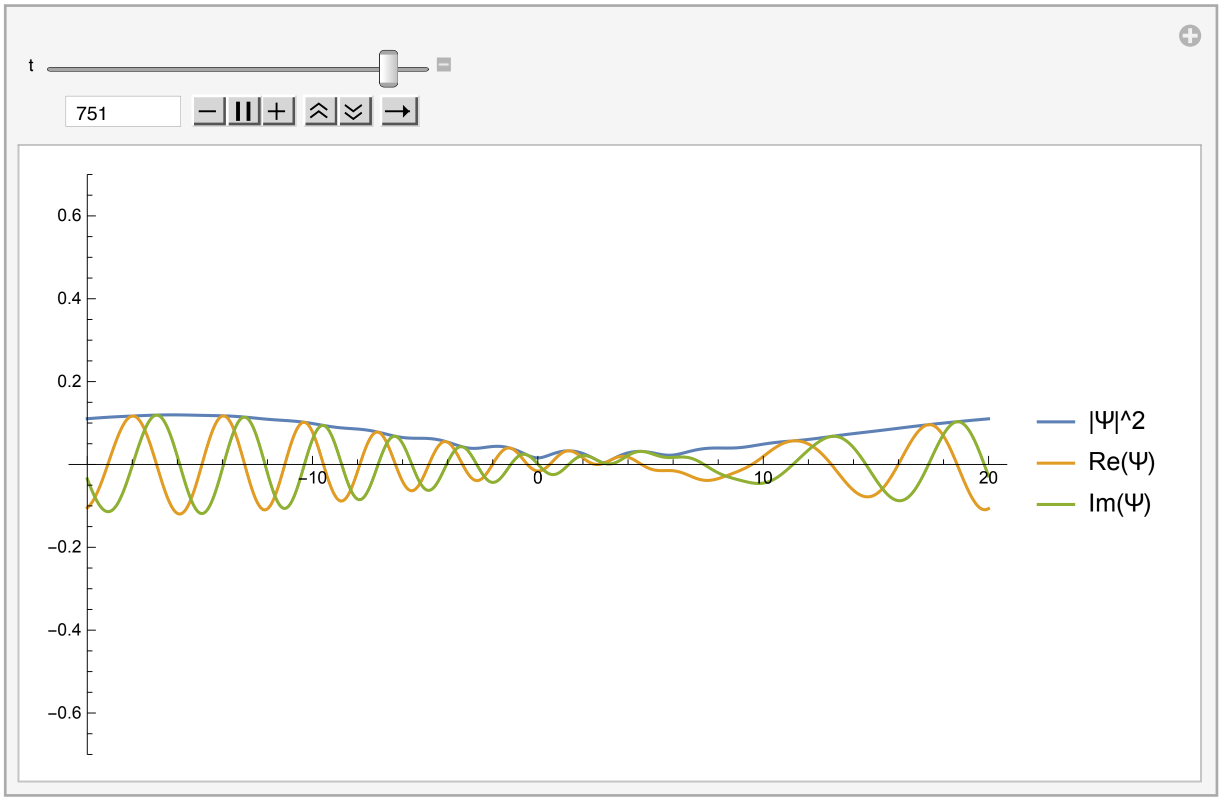
Scheme 1: Simple Finite Difference (SFD)

this imports in form {rows, columns} equal to {time, position}

```
In[116]:= Clear[freeRe, freeIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 1;
            periodicChoice = 1;
             schemeChoice = 1;
            Do[
              Evaluate[Symbol["free" <> {"Re", "Im"}[[i]]]] = Import[
                  path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                   scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"];,
              {i, 2}];
            free = Sqrt[freeRe^2 + freeIm^2];
           ];
```

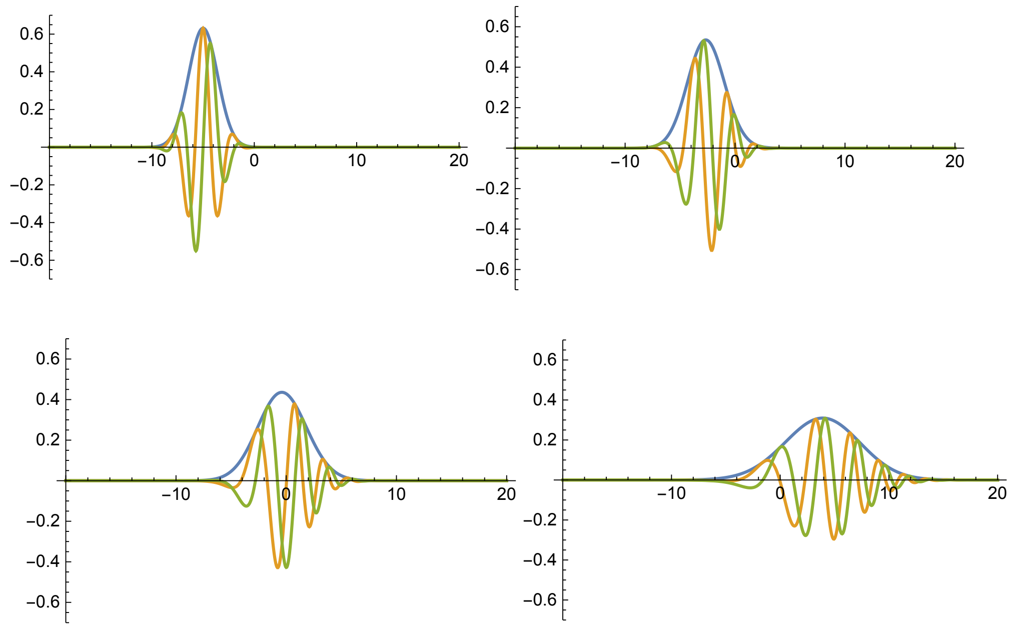
```
In[118]:= Manipulate[ListPlot[{free[[t, All]], freeRe[[t, All]], freeIm[[t, All]]},
            PlotRange → {-.7, .7}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            ImageSize → 500, Joined -> True, DataRange → {-20, 20},
            AxesOrigin → {-20, 0}], {t, 1, Length[freeRe[[All, 1]]], 1}]
```

Out[118]=

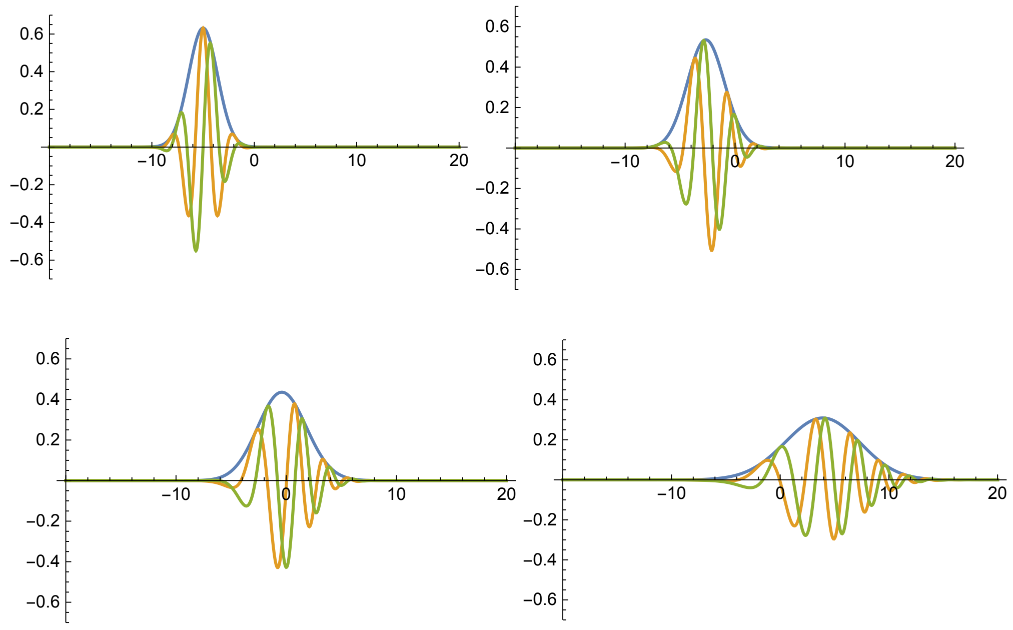


Well, that seems to not work well. The wave decoheres and flattens. Snapshots at timesteps 1, 50, 100, and 200 are shown below.

In[119]:=



Out[119]=



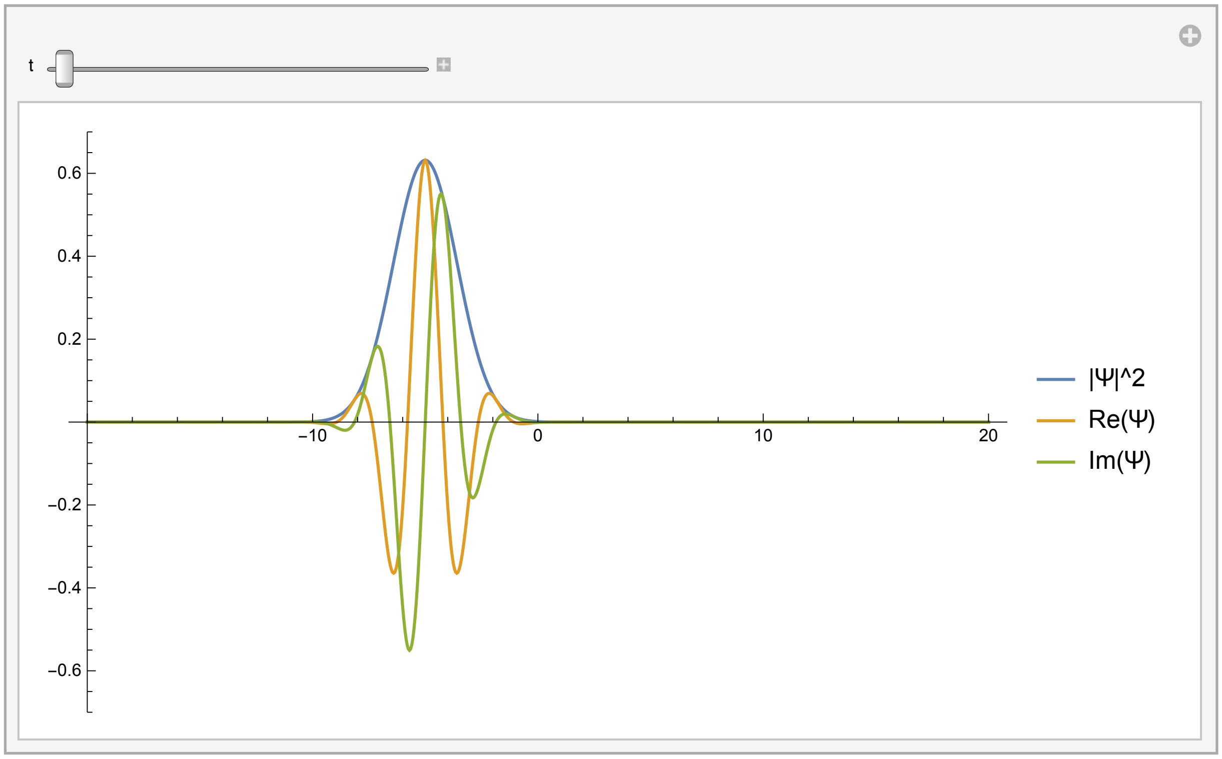
Scheme 2: Crank-Nicolson (CN)

This should be the more stable scheme. For the rest of the cases, we will implement this scheme.

this imports in form {rows, columns} equal to {time, position}

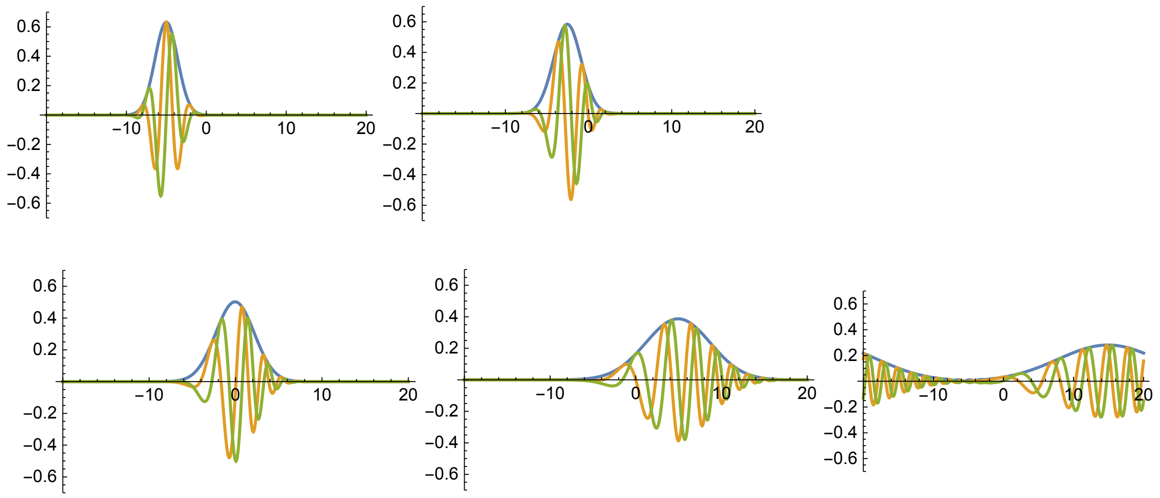
```
In[120]:= Clear[freeCNRe, freeCNIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 1;
            periodicChoice = 1;
             schemeChoice = 2;
            Do[
              Evaluate[Symbol["freeCN" <> {"Re", "Im"}[[i]]]] = Import[
                  path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                   scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"] ;,
              {i, 2}];
            freeCN = Sqrt[freeCNRe^2 + freeCNIm^2];
           ];

In[122]:= Manipulate[ListPlot[{freeCN[[t, All]], freeCNRe[[t, All]], freeCNIm[[t, All]]},
            PlotRange → {-.7, .7}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            ImageSize → 500, Joined → True, DataRange → {-20, 20},
            AxesOrigin → {-20, 0}], {t, 1, Length[freeCNRe[[All, 1]]], 1}]
```

Out[122]=

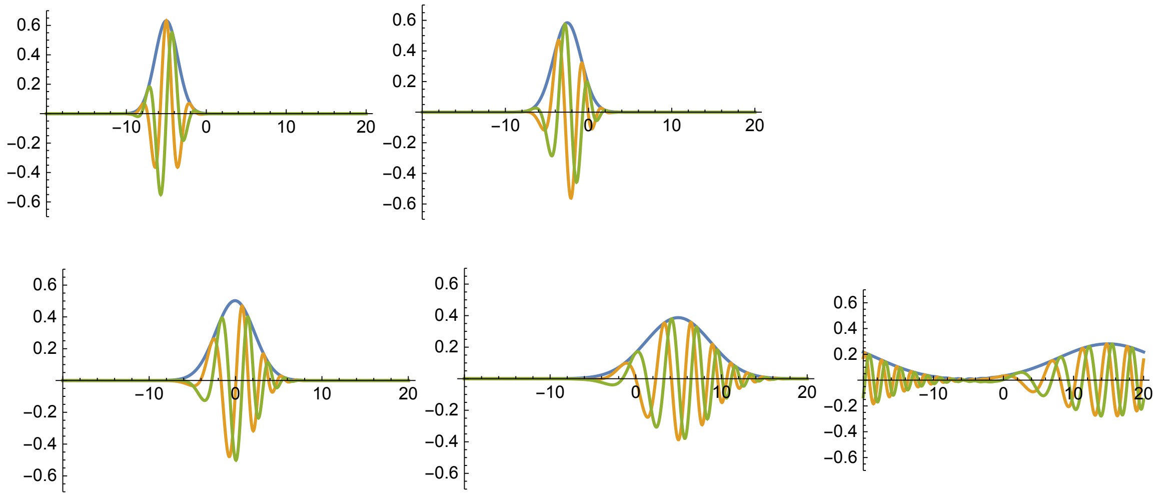


The wave still does decohere, but it takes much longer. We also see it spread out, but doesn’t flatten as before. The same set of timesteps (1, 50, 100, and 200) are shown below along with timestep 400.

In[123]:=



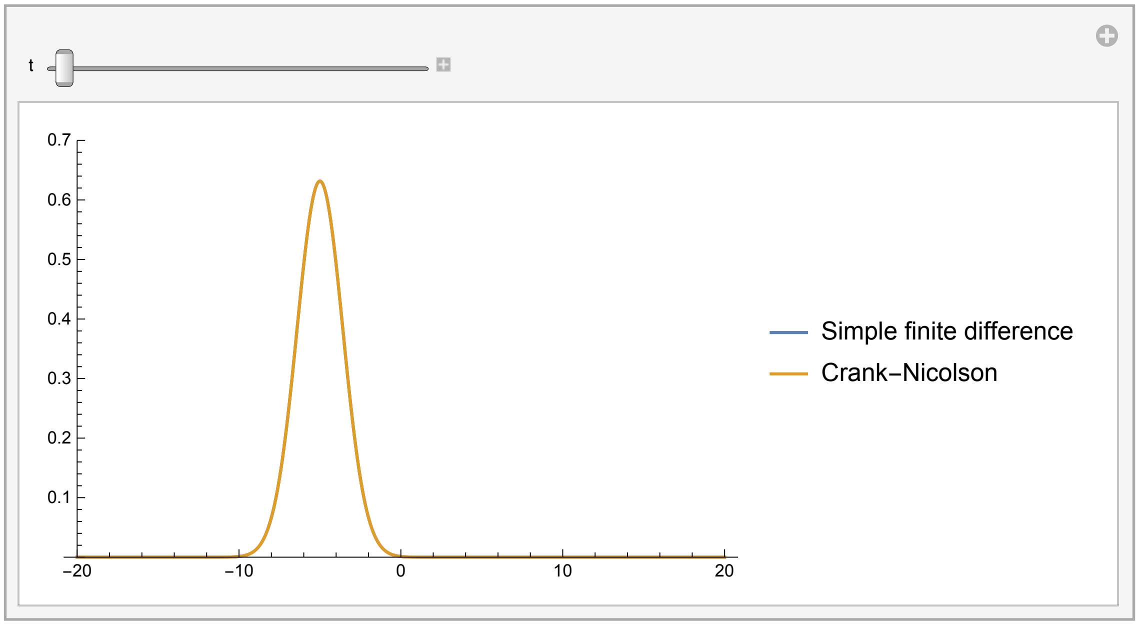
Out[123]=



Let’s compare the two schemes directly

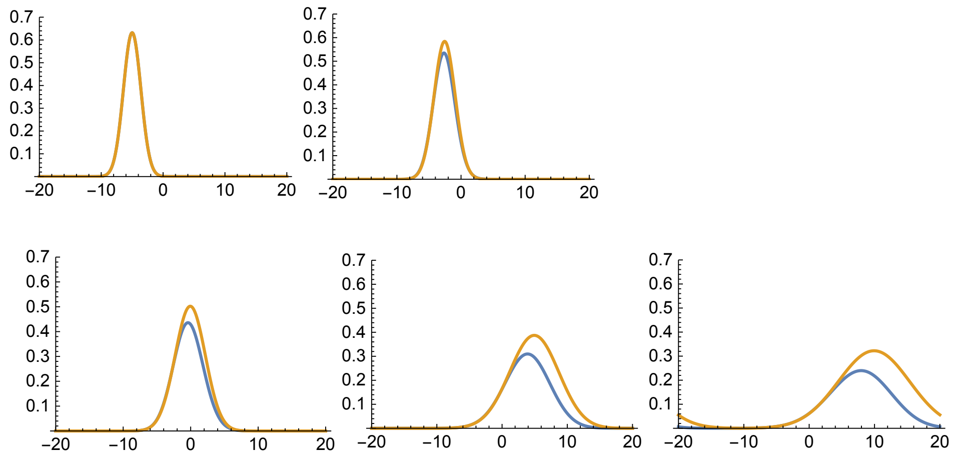
In[124]:=
```
Manipulate[ListPlot[{free⟦t, All⟧, freeCN⟦t, All⟧}, PlotRange → {0, .7},
  Joined → True, PlotLegends → {"Simple finite difference", "Crank-Nicolson"},
  DataRange → {-20, 20}, AxesOrigin → {-20, 0}], {t, 1, Length[freeCNRe⟦All, 1⟧], 1}]
```

Out[124]=

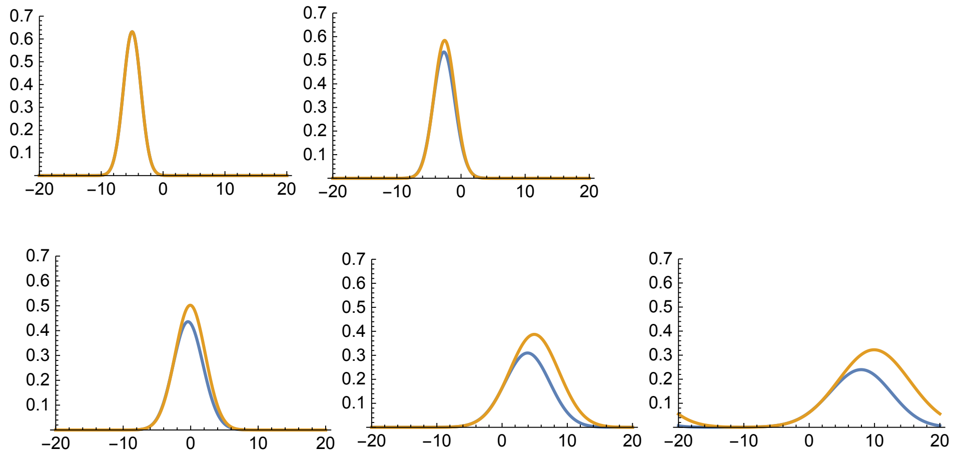


The flattening effect is quite significant. Timesteps 1, 50, 100, 200, and 300 are shown below.

In[125]:=



Out[125]=

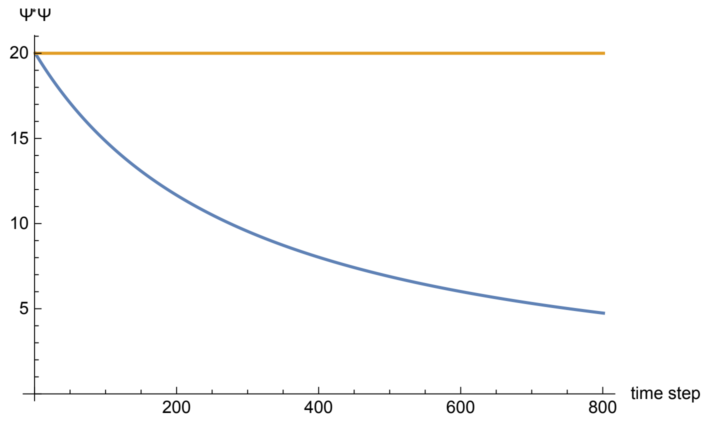


and check normalization

In[126]:=

```
ListPlot[{Total[free^2, {2}], Total[freeCN^2, {2}]},
 Joined → True, AxesLabel → {"time step", "Ψ*Ψ"}]
```

Out[126]=



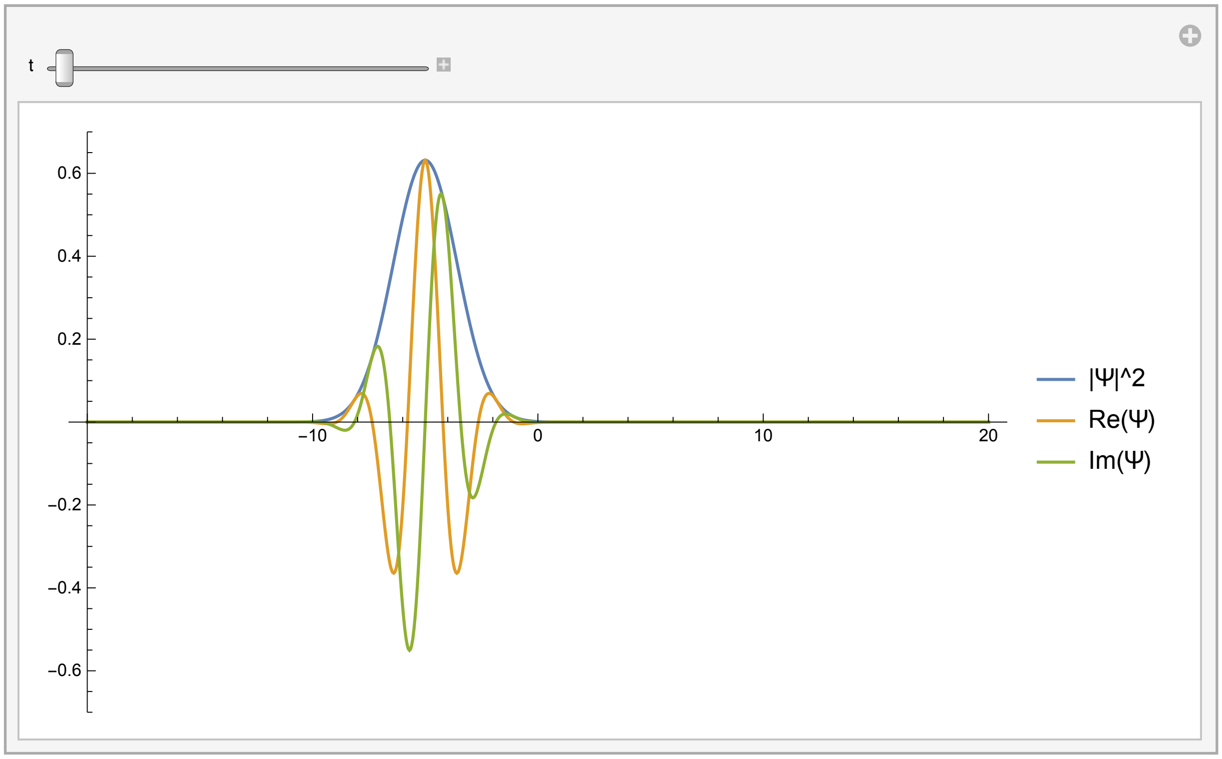
Scheme 2 doesn’t lose normalization!

Scheme 2 with non-Periodic boundary conditions

We can see what happens when we effectively put our free particle in a box by fixing the edges.

```
In[127]:= Clear[freeNonPeriodicCNRe, freeNonPeriodicCNIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 1;
            periodicChoice = 2;
             schemeChoice = 2;
            Do[
             Evaluate[Symbol["freeNonPeriodicCN" <> {"Re", "Im"}[[i]]]] = Import[
                 path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                  scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"];,
              {i, 2}];
            freeNonPeriodicCN = Sqrt[freeNonPeriodicCNRe^2 + freeNonPeriodicCNIm^2];
           ];

In[129]:= Manipulate[ListPlot[{freeNonPeriodicCN[[t, All]],
               freeNonPeriodicCNRe[[t, All]], freeNonPeriodicCNIm[[t, All]]},
            PlotRange → {-.7, .7}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            ImageSize → 500, Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}],
           {t, 1, Length[freeNonPeriodicCNRe[[All, 1]]], 1}]
```

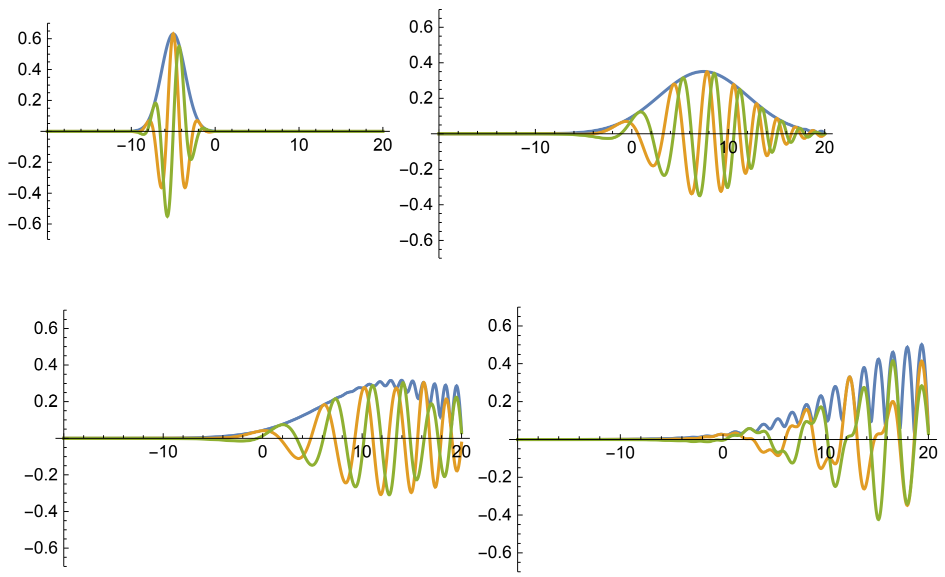
Out[129]=



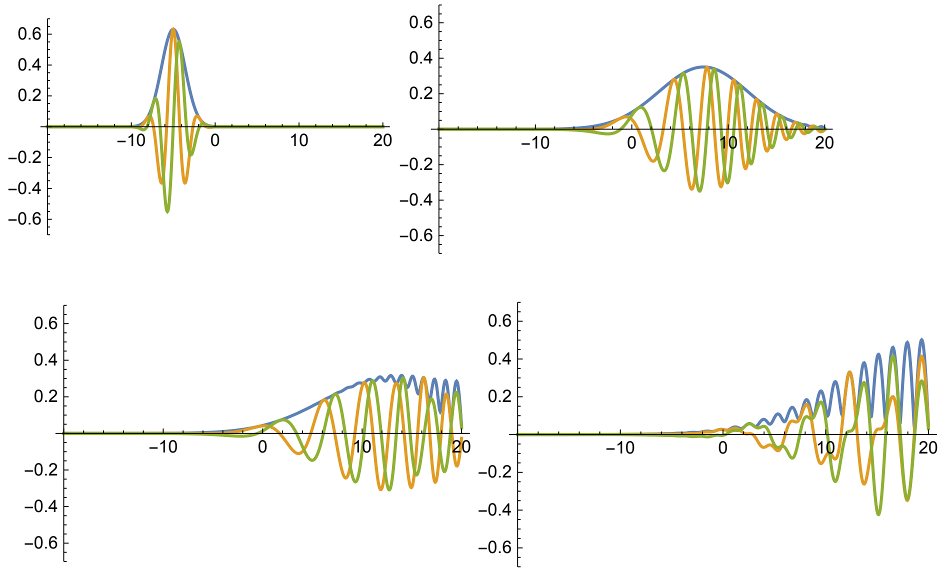
SPLAT!

Using the joined option seems to hide the fact that the very end data point does connect to zero. Timesteps 1, 250, 350, and 450 are shown below.

In[130]:=



Out[130]=



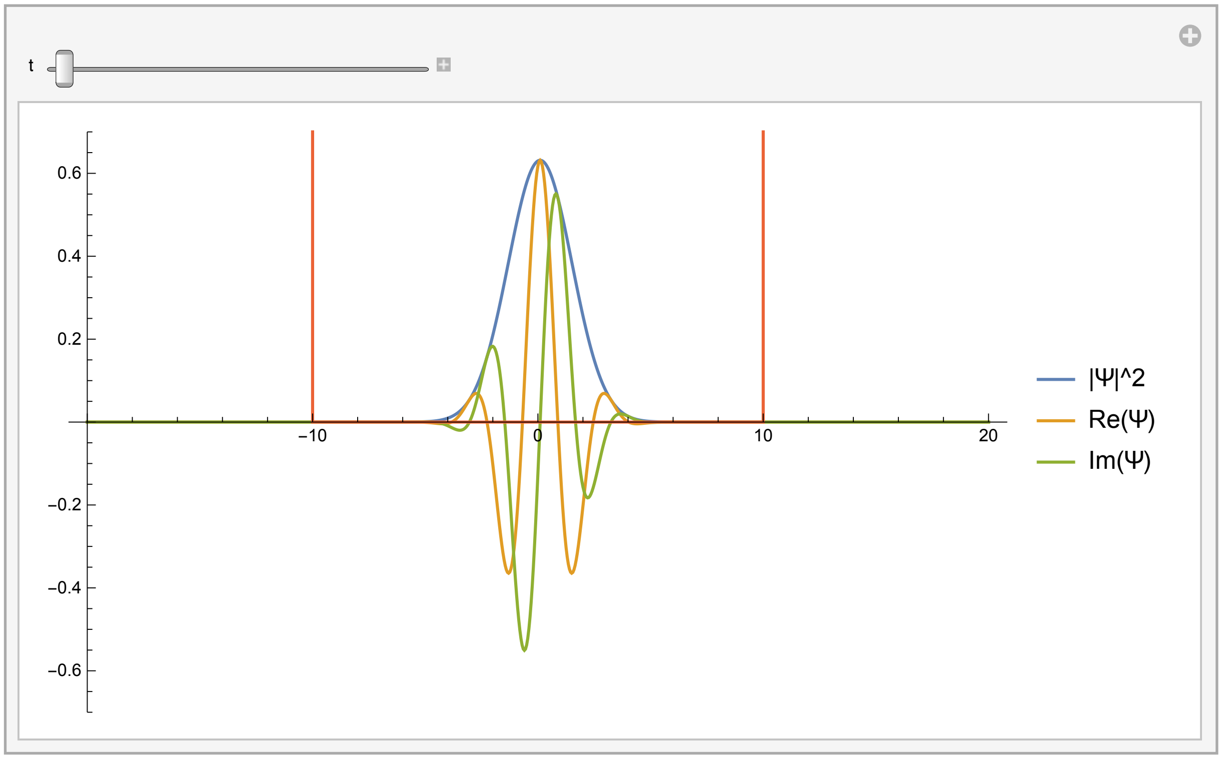
Square Well

We’ll look at two versions of the square well: one with our gaussian wave packet, and another with an actual eigenstate (cos(nx/L)).

With a wave packet.

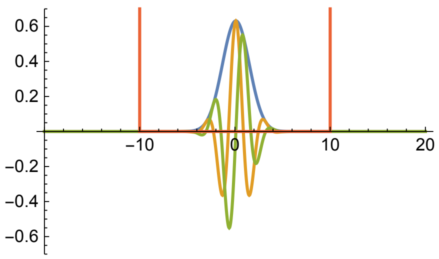
```
In[131]:= Clear[squareCNRe, squareCNIm, squareCN, squarePotential];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 2;
            periodicChoice = 2;
             schemeChoice = 2;
            Do[
              Evaluate[Symbol["squareCN" <> {"Re", "Im"}[[i]]]] = Import[
                  path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                   scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"] ;,
              {i, 2}];
            squarePotential = Flatten[Import[
                path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                 scheme[[schemeChoice]] <> "_" <> "Potential" <> ".csv", "CSV"]];
            squareCN = Sqrt[squareCNRe^2 + squareCNIm^2];
           ];

In[133]:= Manipulate[ListPlot[
            {squareCN[[t, All]], squareCNRe[[t, All]], squareCNIm[[t, All]], squarePotential},
            PlotRange → {-.7, .7}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500, Joined → True],
           {t, 1, Length[squareCNRe[[All, 1]]], 1}]
```

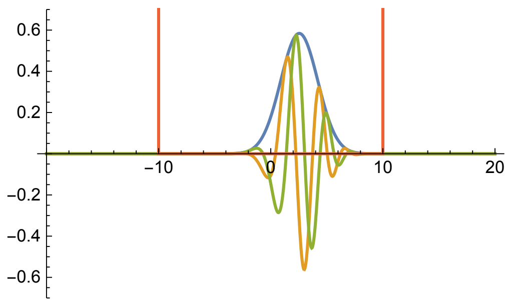
Out[133]=

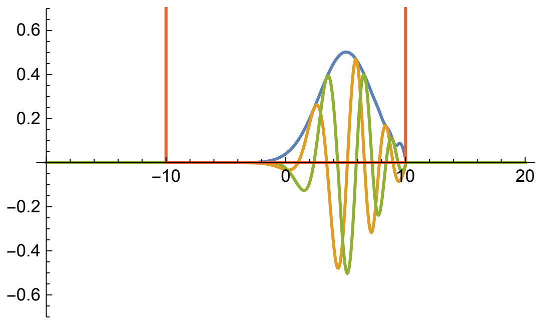


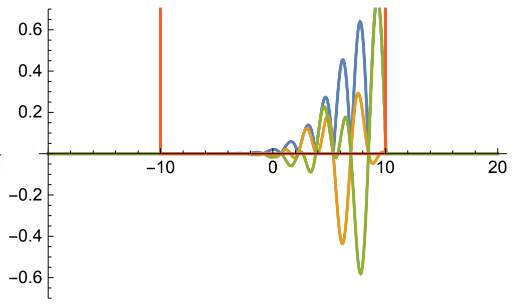
That looks exactly like what we'd expect! Timesteps 1, 50, 100, and 150 are shown below.

In[134]:=

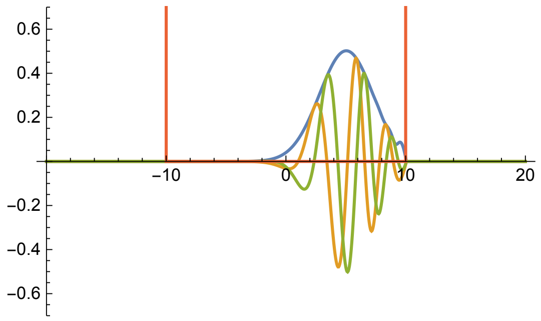


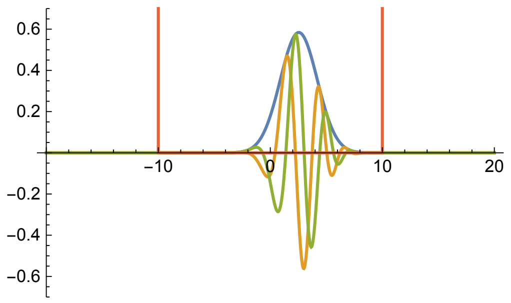


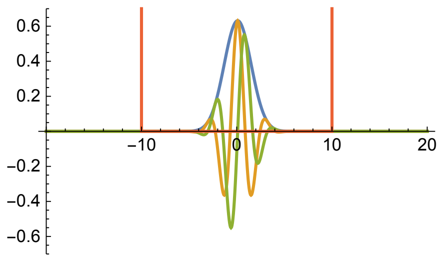


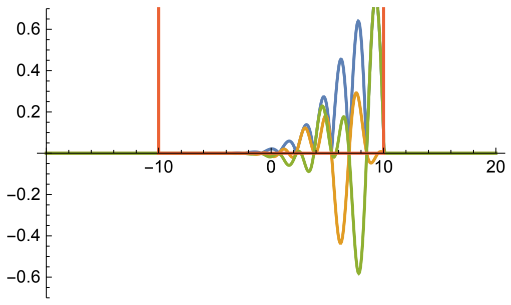


Out[134]=





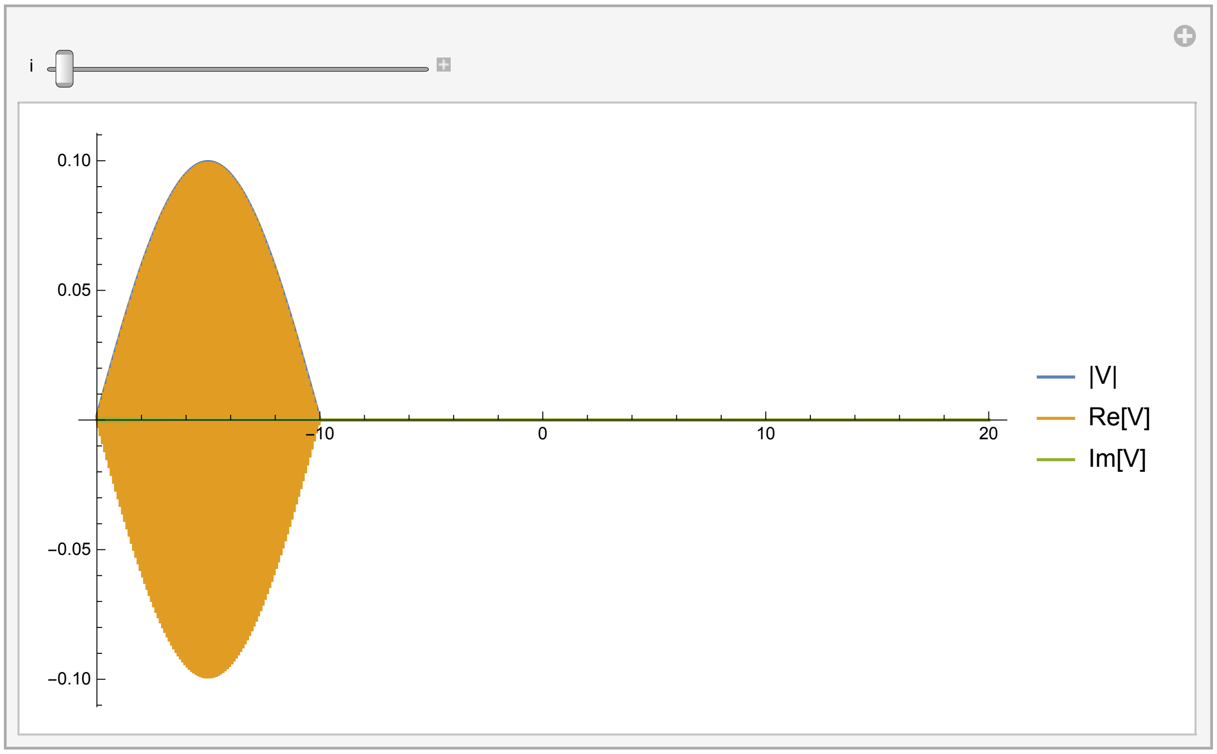




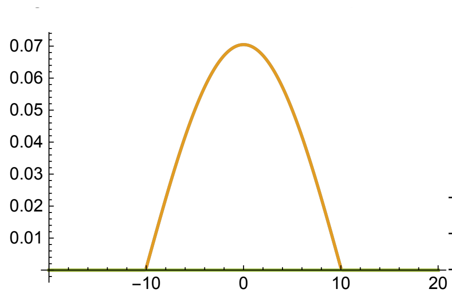
Let' s look at our eigenvectors.

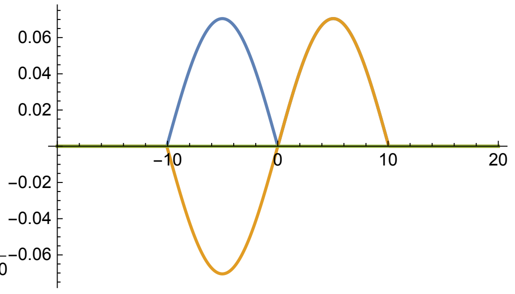
```
In[135]:= Clear[squareEVectorsRe, squareEVectorsIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 2;
            periodicChoice = 2;
             schemeChoice = 2;
            squareEValues = Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <>
                "/" <> periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <>
                "_" <> numberType⟦1⟧ <> "_" <> eigenState⟦1⟧ <> ".csv", "CSV"];
            Do[
             Evaluate[Symbol["squareEVectors" <> {"Re", "Im"}⟦i⟧]] =
                Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <>
                  periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <> "_" <>
                  numberType⟦i⟧ <> "_" <> eigenState⟦2⟧ <> ".csv", "CSV"];,
              {i, 2}];
            squareEVectors = Sqrt[squareEVectorsRe^2 + squareEVectorsIm^2];
           ];

In[137]:= Manipulate[ListPlot[
            {squareEVectors⟦All, i⟧, squareEVectorsRe⟦All, i⟧, squareEVectorsIm⟦All, i⟧},
            Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
            PlotRange → Full, PlotLegends → {"|V|", "Re[V]", "Im[V]"}],
           {i, 1, Length[squareEVectors⟦All, 1⟧], 1}]
```

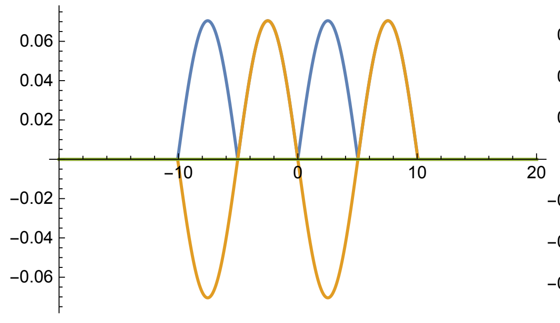
Out[137]=

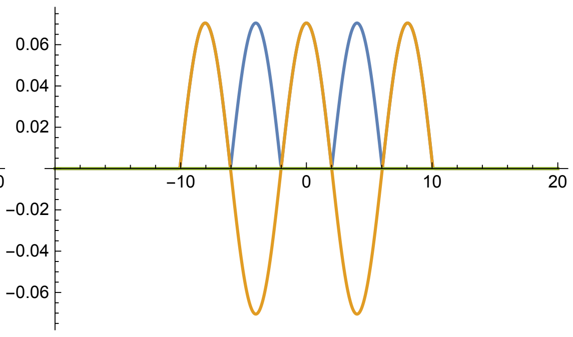


153 is the ground state, 154 is the first excited state (153-160 are shown below). Note that these eigenvectors are real--we expect that.

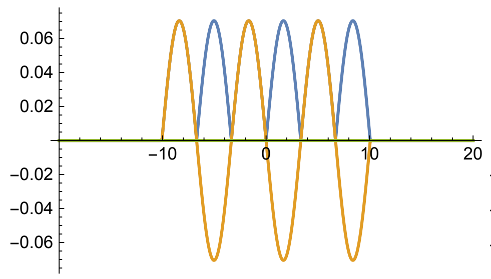

0.07
0.06
0.05
0.04
0.03
0.02
0.01
−10
0
10
20

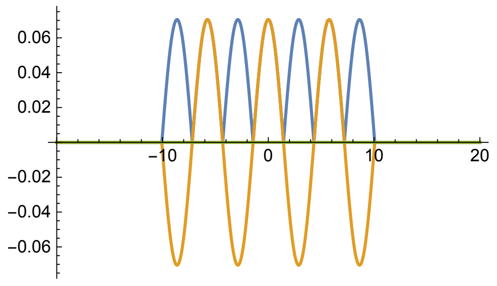

0.06
0.04
0.02
−0.02
−0.04
−0.06
−10
0
10
20

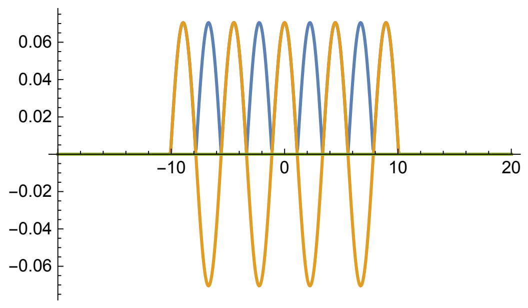

0.06
0.04
0.02
−0.02
−0.04
−0.06
−10
0
10
20

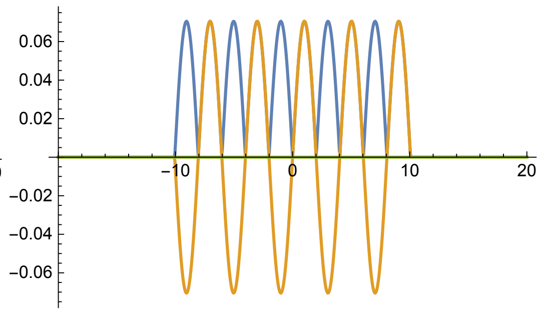

0.06
0.04
0.02
−0.02
−0.04
−0.06
−10
0
10
20

In[138]:=

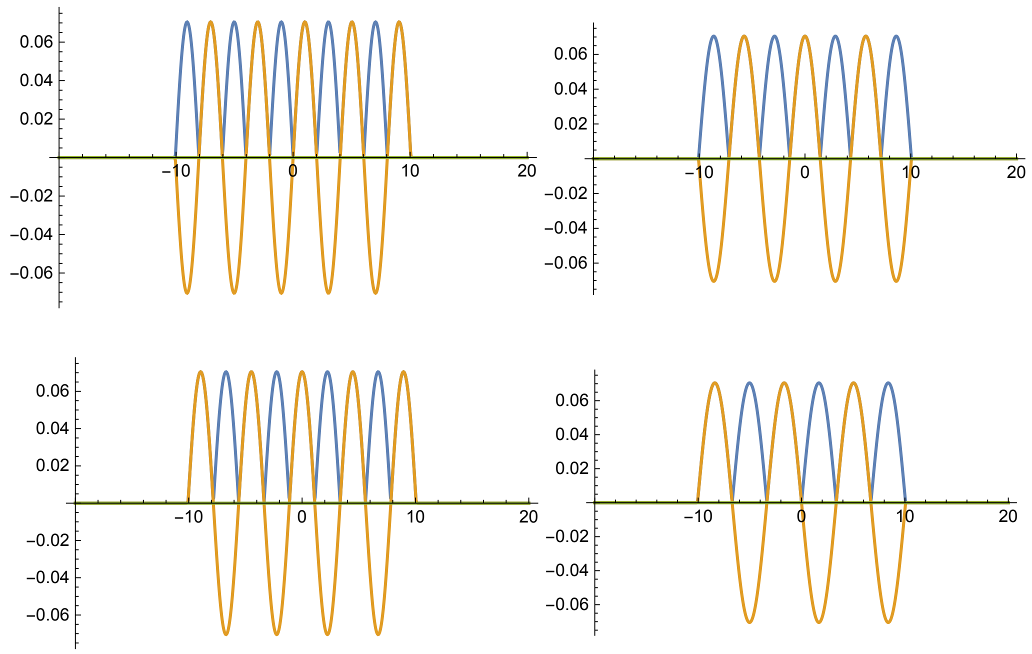

0.06
0.04
0.02
−0.02
−0.04
−0.06
−10
0
10
20


0.06
0.04
0.02
−0.02
−0.04
−0.06
−10
0
10
20


0.06
0.04
0.02
−0.02
−0.04
−0.06
−10
0
10
20

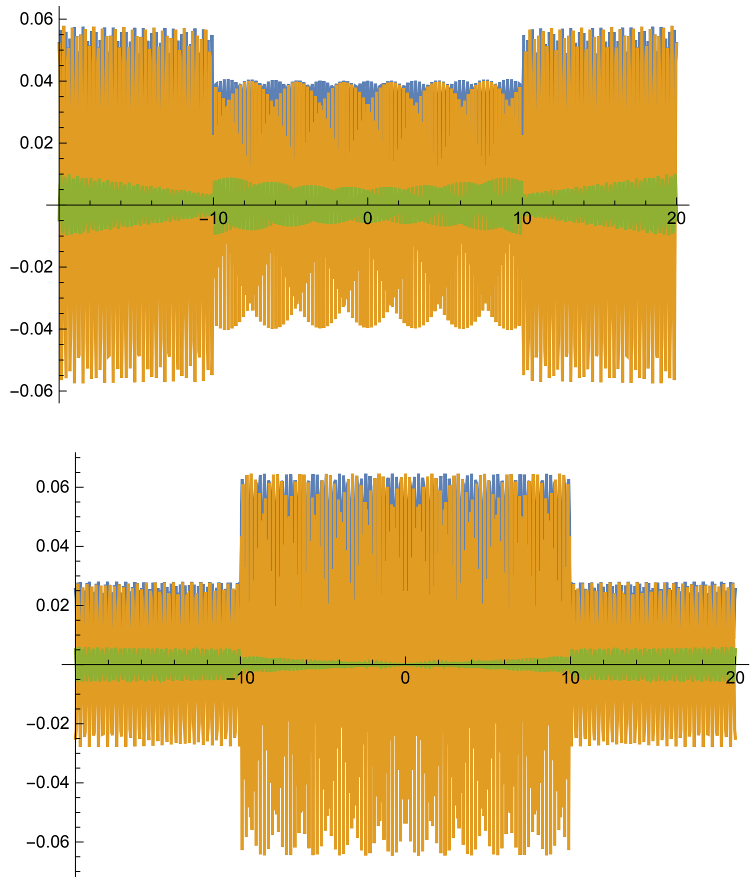

0.06
0.04
0.02
−0.02
−0.04
−0.06
−10
0
10
20

Out[138]=

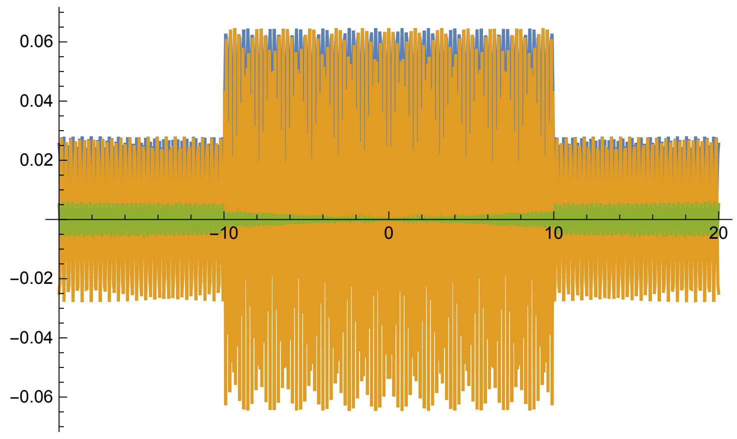


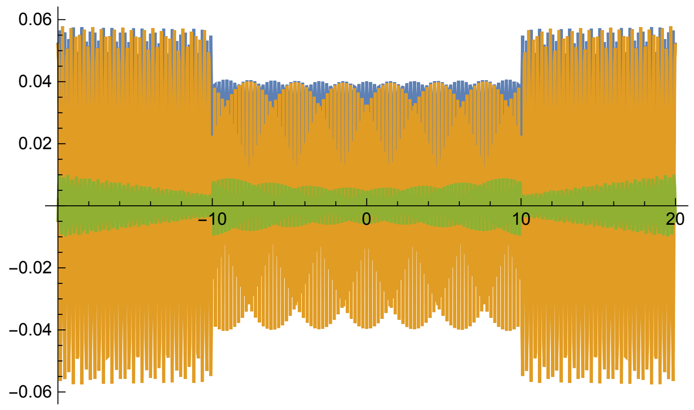
So there seem to be some very odd states in here (likely coming from the fact that our particle in a “box” is only set to be a semi-infinite well (1000 units high). Once the energy gets large, it makes sense that the particle could be behind the walls. Take a look at 562 and 563 below. Note that these also have significant non-real components.

In[139]:=



Out[139]=

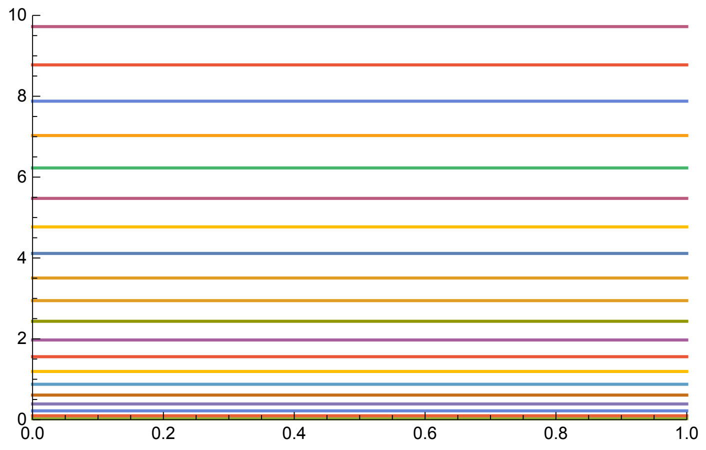




Let’s look at the first few energy eigenstates.

In[140]:=
```
Plot[squareEValues, {x, 0, 1}, PlotRange → {{0, 1}, {0, 10}}]
```

Out[140]=



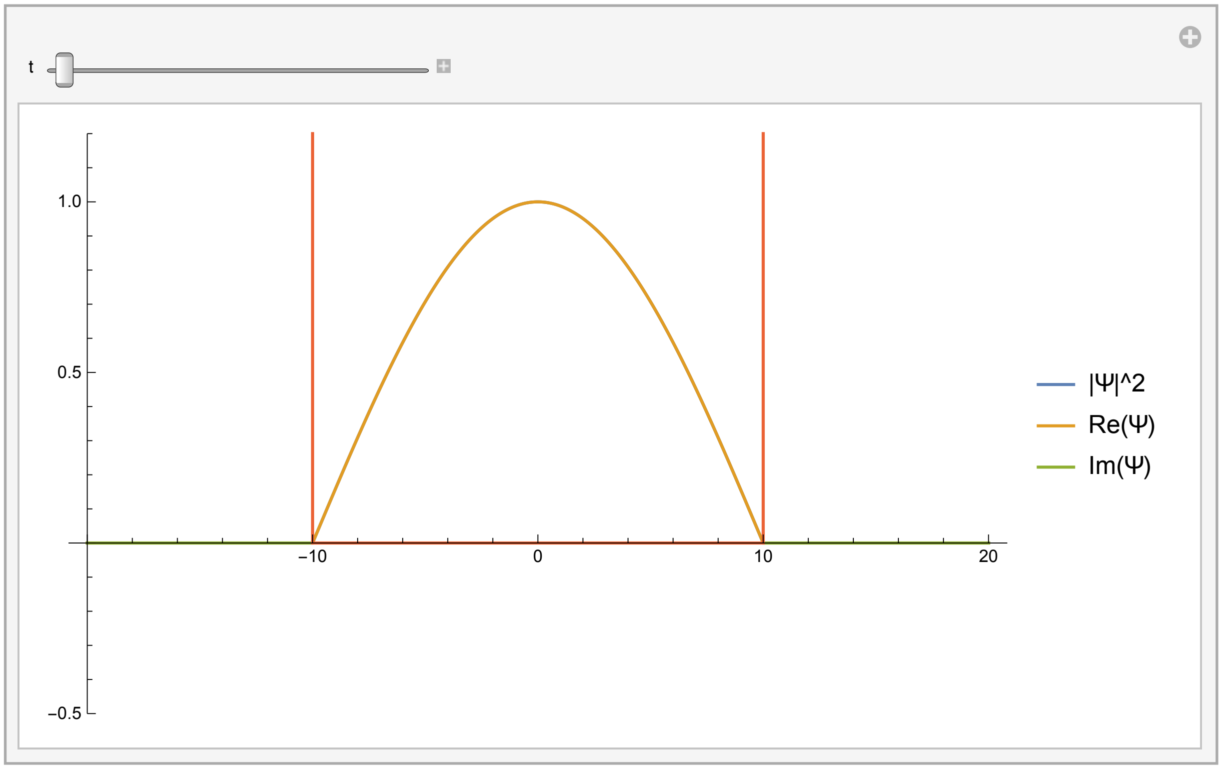
As we’d expect, we see the separation between the eigenstates increasing.

With an initial cosine wave.

Since we can’t seem to calculate an eigenvector, let’s input it manually.

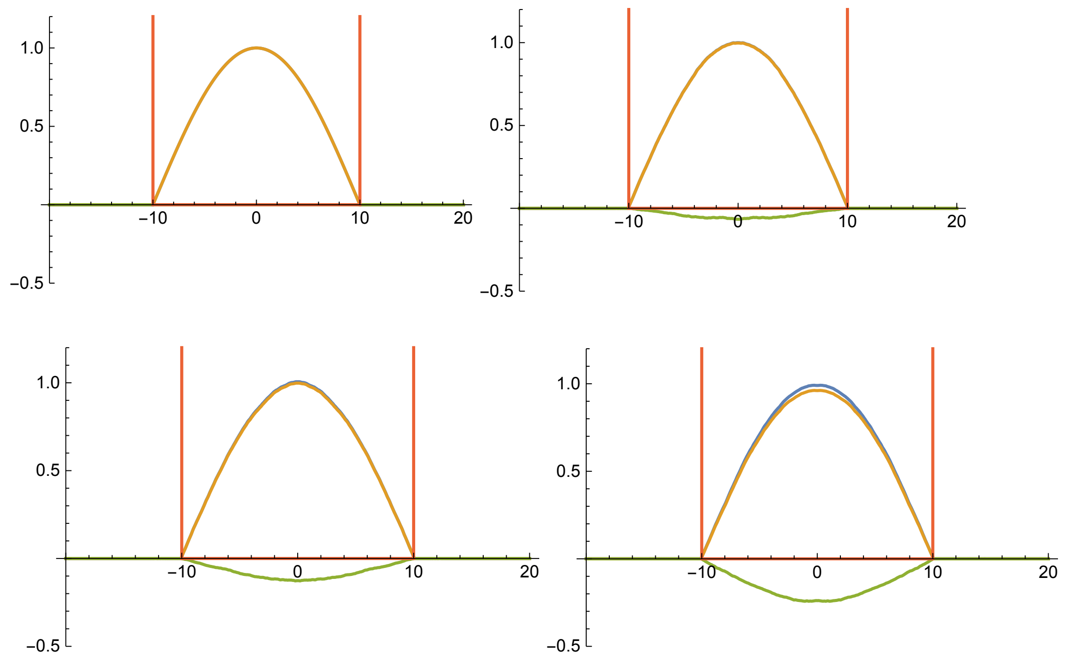
```
In[141]:= Clear[squareCosCNRe, squareCosCNIm, squareCosCN, squareCosPotential];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 11;
            periodicChoice = 2;
             schemeChoice = 2;
            Do[
              Evaluate[Symbol["squareCosCN" <> {"Re", "Im"}[[i]]]] = Import[
                  path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                   scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"];,
              {i, 2}];
            squareCosPotential = Flatten[Import[
                path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                  scheme[[schemeChoice]] <> "_" <> "Potential" <> ".csv", "CSV"]];
            squareCosCN = Sqrt[squareCosCNRe^2 + squareCosCNIm^2];
           ];

In[143]:= Manipulate[ListPlot[{squareCosCN[[t, All]], squareCosCNRe[[t, All]],
              squareCosCNIm[[t, All]], squareCosPotential}, PlotRange → {-.5, 1.2},
            PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"}, DataRange → {-20, 20},
            AxesOrigin → {-20, 0}, ImageSize → 500, Joined → True],
           {t, 1, Length[squareCosCNRe[[All, 1]]], 1}]
```

Out[143]=

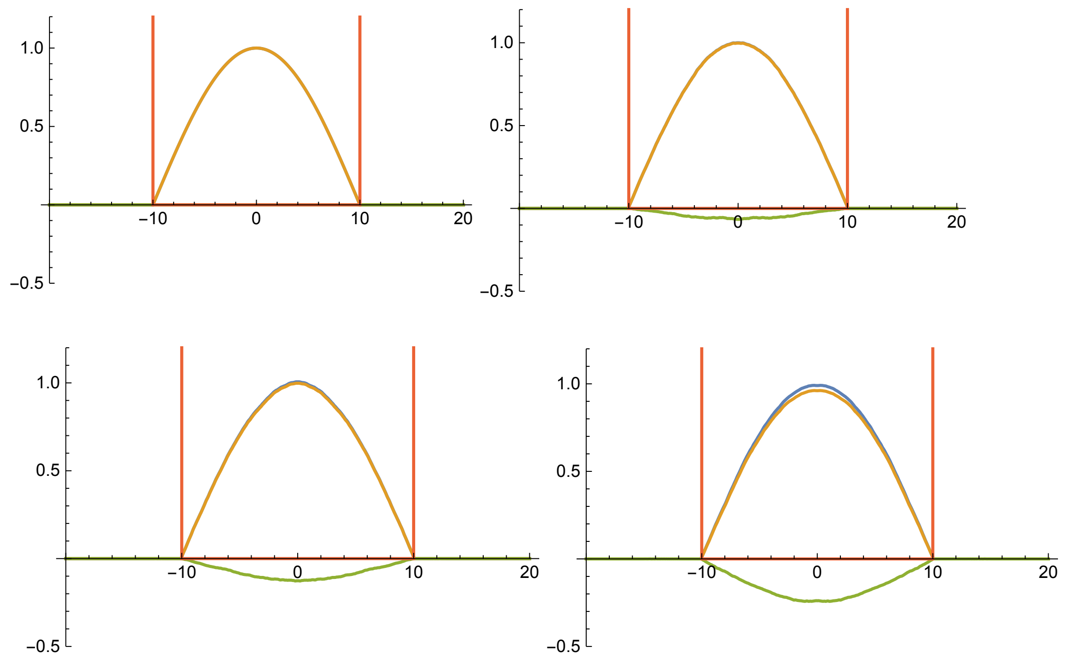


We can see that this is pretty stable, though it does slowly destabalize over time (which is what we'd expect). See timesteps 1, 200, 400, and 800 below.

In[144]:=



Out[144]=

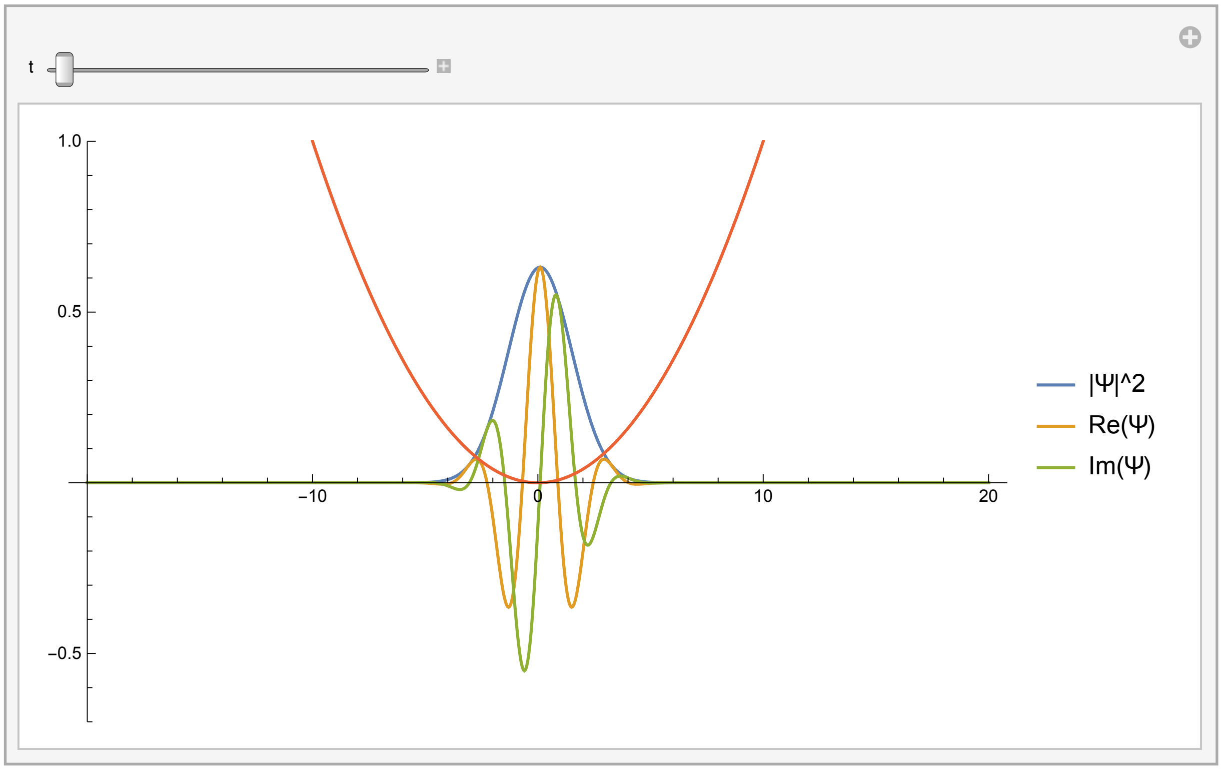


Harmonic Oscillator

With a wave packet

```
In[145]:= Clear[hoCNRe, hoCNIm, hoCN, hoPotential];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 3;
            periodicChoice = 2;
             schemeChoice = 2;
            Do[
              Evaluate[Symbol["hoCN" <> {"Re", "Im"}⟦i⟧]] = Import[
                  path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
                   scheme⟦schemeChoice⟧ <> "_" <> numberType⟦i⟧ <> ".csv", "CSV"];,
              {i, 2}];
            hoPotential = Flatten[Import[
                path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
                 scheme⟦schemeChoice⟧ <> "_" <> "Potential" <> ".csv", "CSV"]];
            hoCN = Sqrt[hoCNRe^2 + hoCNIm^2];
           ];

In[147]:= Manipulate[ListPlot[{hoCN⟦t, All⟧, hoCNRe⟦t, All⟧, hoCNIm⟦t, All⟧, hoPotential / 200},
            PlotRange → {-.7, 1}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            ImageSize → 500, DataRange → {-20, 20}, AxesOrigin → {-20, 0},
            Joined → True], {t, 1, Length[hoCNRe⟦All, 1⟧], 1}]
```

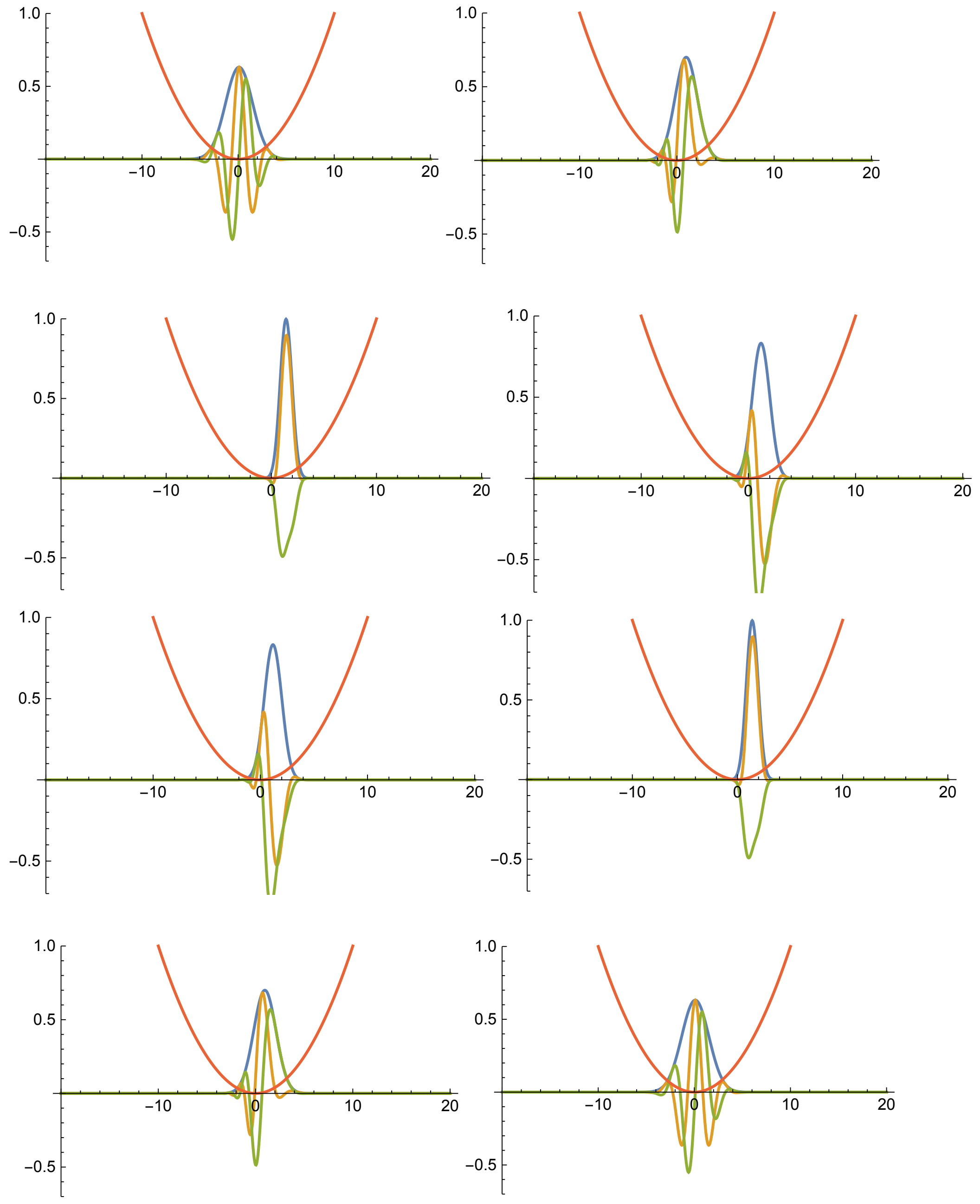
Out[147]=



That looks reasonable. Some snapshots (timesteps 1, 20, 40, and 60) are below.

In[148]:=

Out[148]=

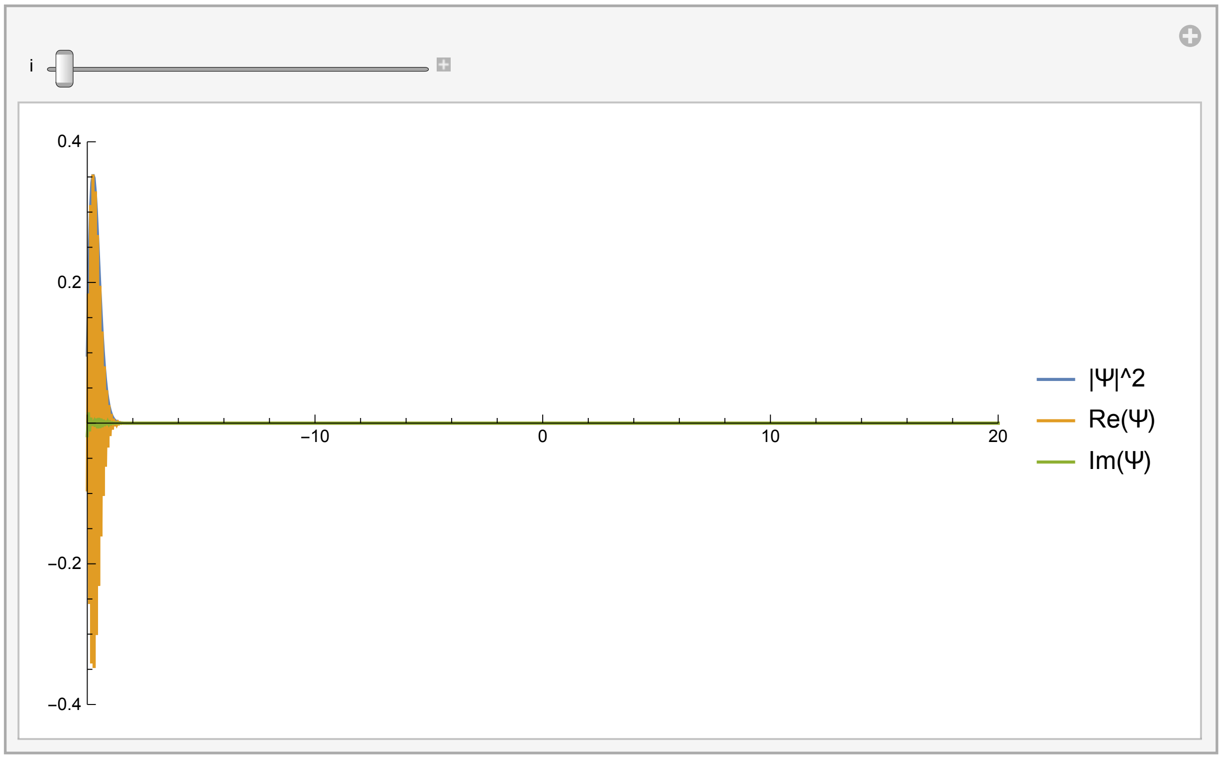


In[149]:=
```
Clear[hoEVectorsRe, hoEVectorsIm];
Block[{caseChoice, periodicChoice, schemeChoice},
   caseChoice = 3;
   periodicChoice = 2;
    schemeChoice = 2;
   hoEValues = Import[path <> "/Runs/" <> cases[[caseChoice]] <>
       "/" <> periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <>
       "_" <> numberType[[1]] <> "_" <> eigenState[[1]] <> ".csv", "CSV"];
   Do[
    Evaluate[Symbol["hoEVectors" <> {"Re", "Im"}[[i]]]] =
       Import[path <> "/Runs/" <> cases[[caseChoice]] <> "/" <>
         periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <> "_" <>
         numberType[[i]] <> "_" <> eigenState[[2]] <> ".csv", "CSV"];,
    {i, 2}];
   hoEVectors = Sqrt[hoEVectorsRe^2 + hoEVectorsIm^2];
  ];
```

Eigenvectors

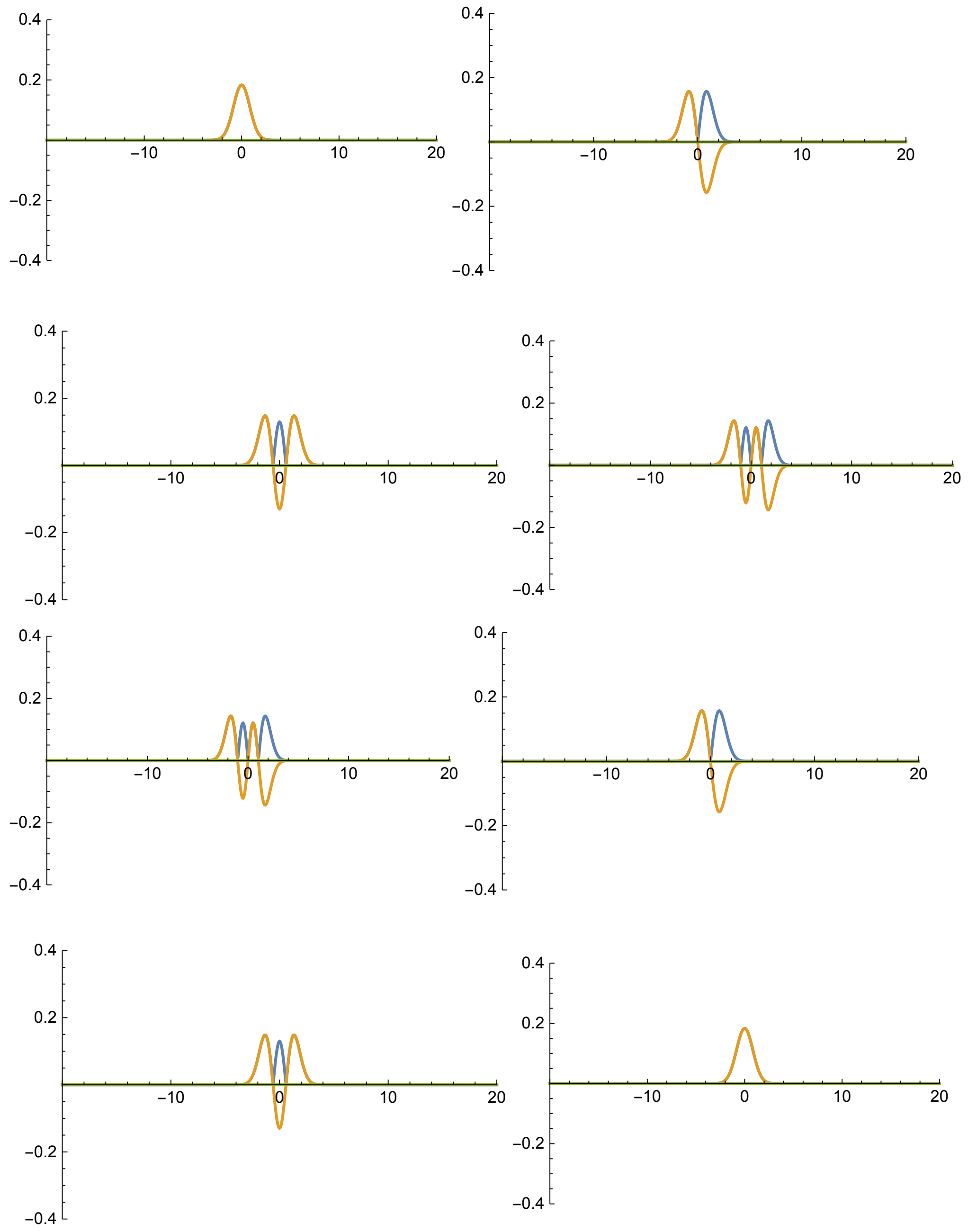
In[151]:=
```
Manipulate[
  ListPlot[{hoEVectors[[All, i]], hoEVectorsRe[[All, i]], hoEVectorsIm[[All, i]]},
   Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
   PlotRange → {{-20, 20}, {-.4, .4}}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"}],
  {i, 1, Length[hoEVectors[[All, 1]]], 1}]
```

Out[151]=



53 is the ground state, then they are in a sensible order for a while. 53-56 are shown below.

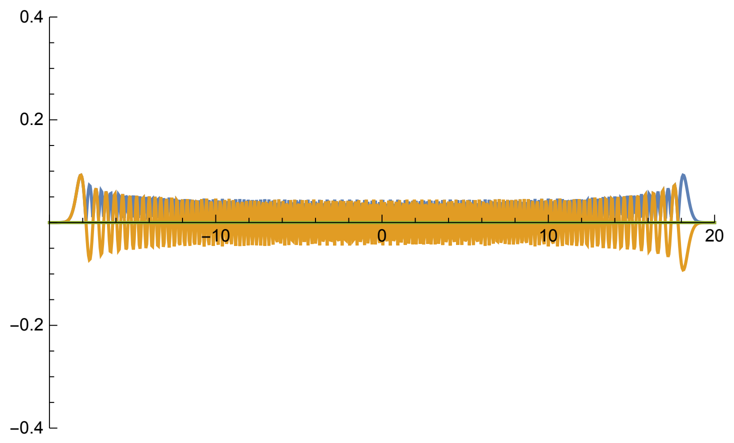
In[152]:=

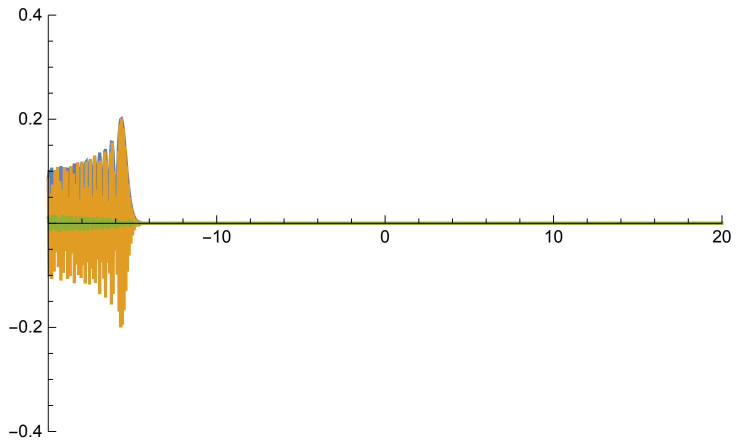


Out[152]=

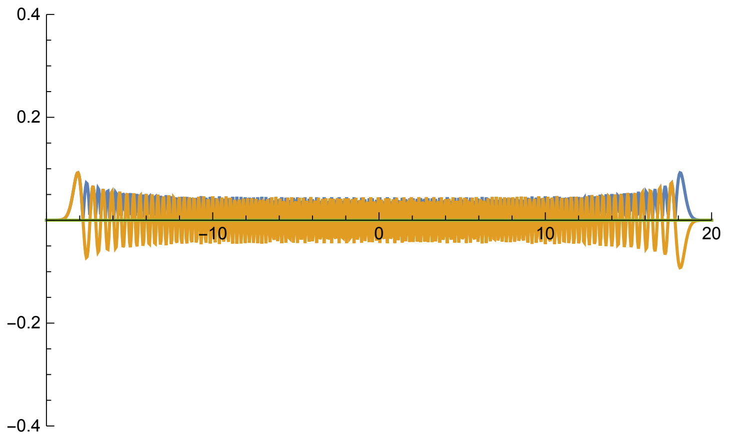
We also seem some sensible high energy states (eg 659, left), and some odd, assymetrical results (18, right)

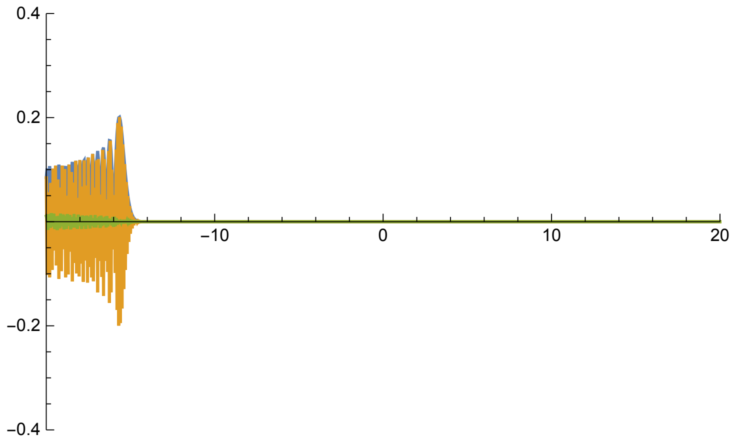
In[153]:=





Out[153]=





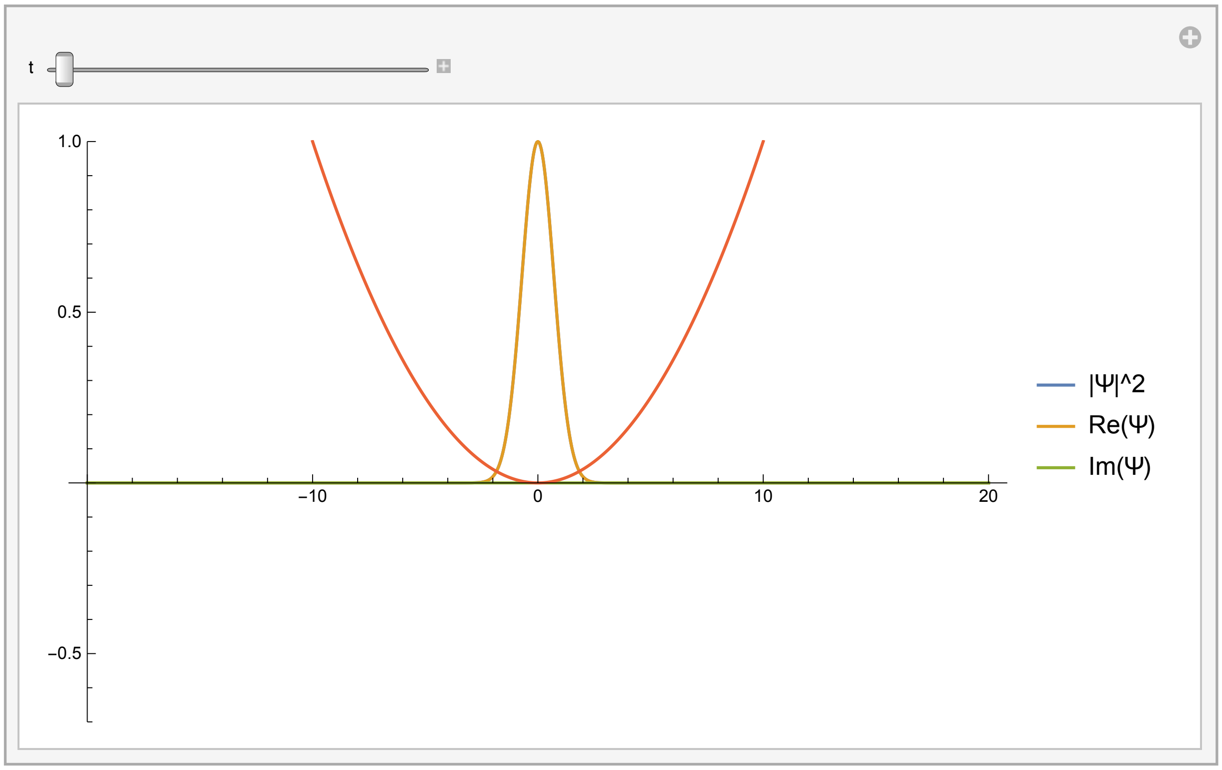
With an initial eigenstate

We can start our particle off in an eigenstate and see how it evolves.

In[154]:=
```
Clear[hoEigCNRe, hoEigCNIm, hoEigCN, hoEigPotential];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 12;
  periodicChoice = 2;
   schemeChoice = 2;
  Do[
    Evaluate[Symbol["hoEigCN" <> {"Re", "Im"}[[i]]]] = Import[
        path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
          scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"] ;,
    {i, 2}];
  hoEigPotential = Flatten[Import[
      path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
        scheme[[schemeChoice]] <> "_" <> "Potential" <> ".csv", "CSV"]];
  hoEigCN = Sqrt[hoEigCNRe^2 + hoEigCNIm^2];
 ];
```

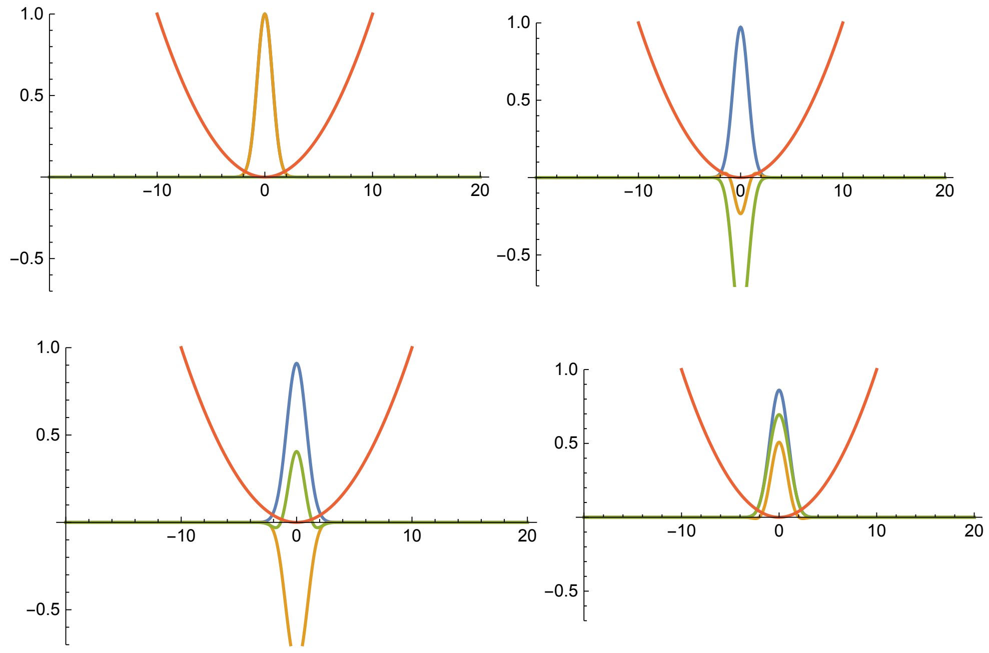
```
In[156]:= Manipulate[ListPlot[
    {hoEigCN[[t, All]], hoEigCNRe[[t, All]], hoEigCNIm[[t, All]], hoEigPotential / 200},
    PlotRange → {-.7, 1}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
    ImageSize → 500, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, Joined → True],
  {t, 1, Length[hoEigCNRe[[All, 1]]], 1}]
```

Out[156]=

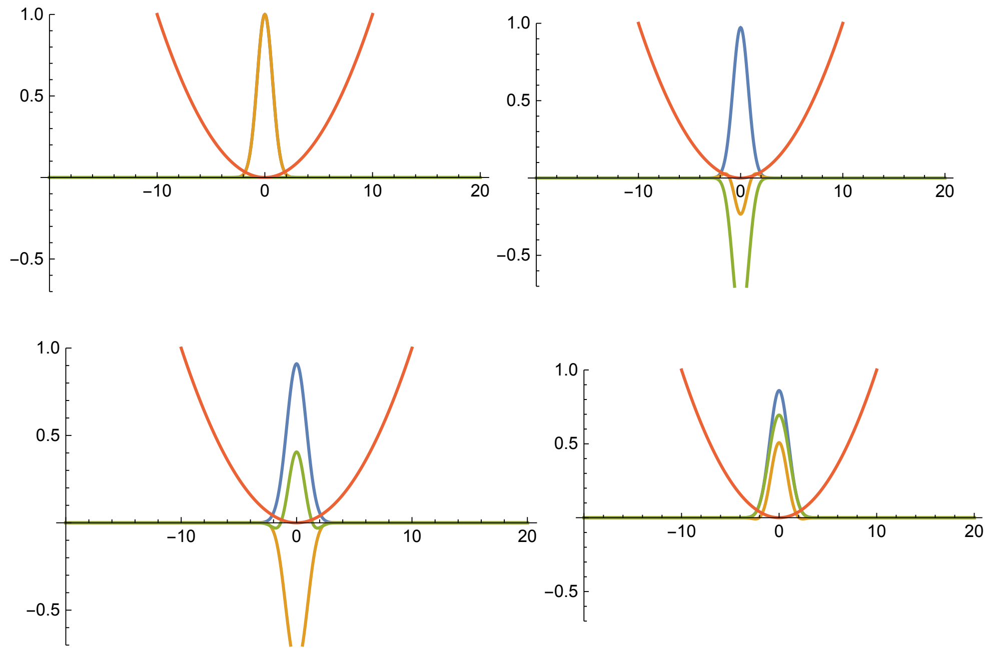


This seems to alternate real vs imaginary parts, but |Ψ|^2 stays about constant. Timesteps 1, 100, 200, and 300 are shown.

In[157]:=



Out[157]=



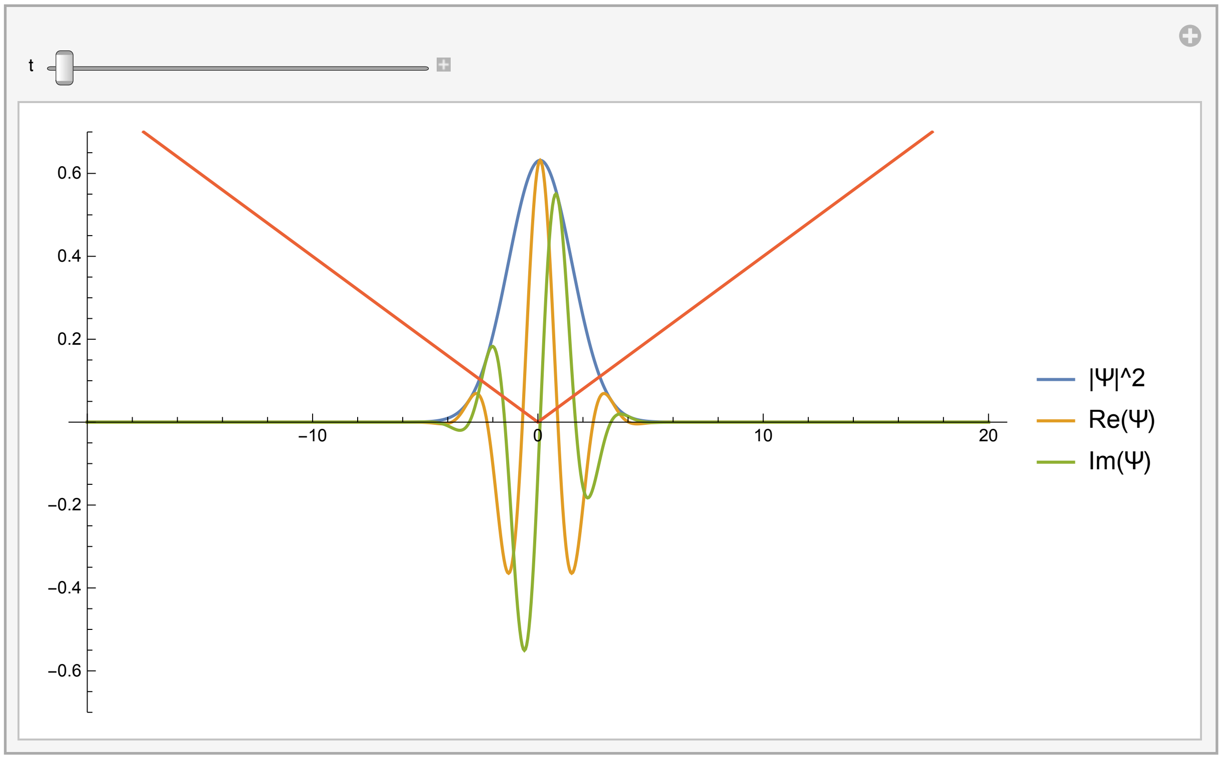
Triangle

Here is our triangle potential.

Let’s start with our wave packet.

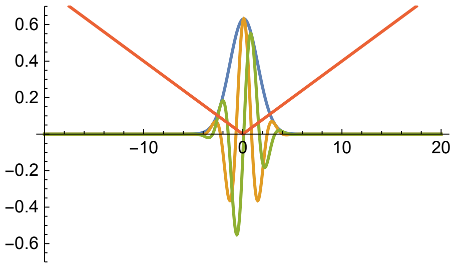
```
In[158]:= Clear[triCNRe, triCNIm, triCN, triPotential];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 4;
            periodicChoice = 2;
             schemeChoice = 2;
            Do[
             Evaluate[Symbol["triCN" <> {"Re", "Im"}[[i]]]] = Import[
                 path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                  scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"] ;,
              {i, 2}];
            triPotential = Flatten[Import[
               path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
                scheme[[schemeChoice]] <> "_" <> "Potential" <> ".csv", "CSV"]];
            triCN = Sqrt[triCNRe^2 + triCNIm^2];
           ];

In[160]:= Manipulate[
           ListPlot[{triCN[[t, All]], triCNRe[[t, All]], triCNIm[[t, All]], triPotential / 50},
            PlotRange → {-.7, .7}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            ImageSize → 500, Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0},
            PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"}], {t, 1, Length[triCNRe[[All, 1]]], 1}]
```

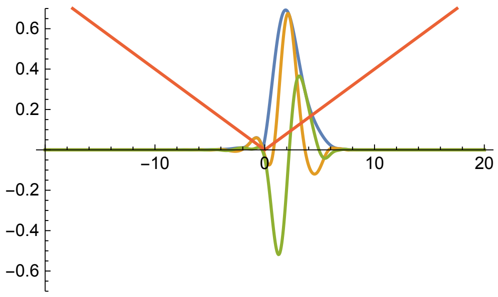
Out[160]=

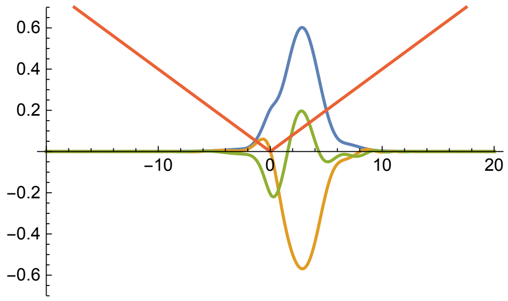


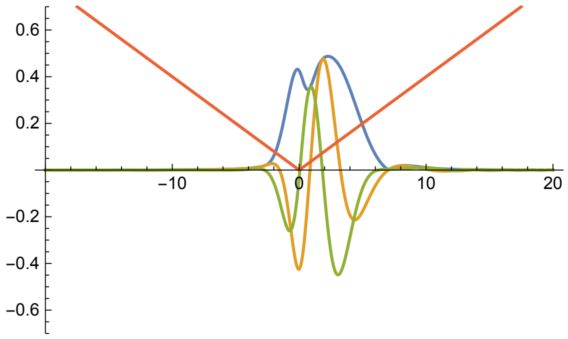
Timesteps 1, 50, 100, and 150 are shown.

In[161]:=

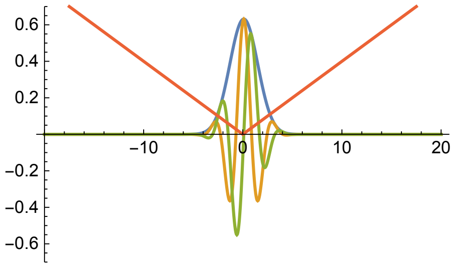


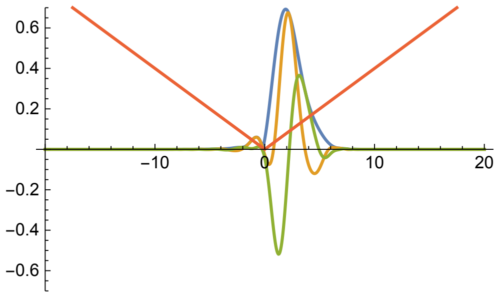


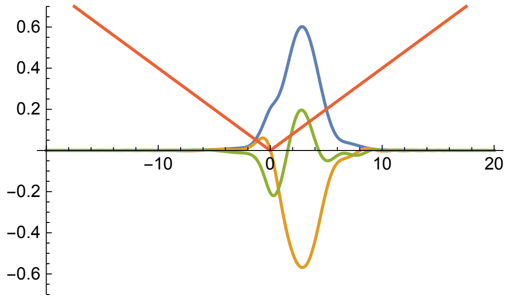


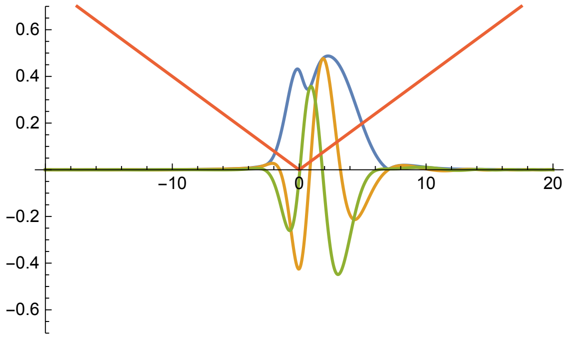


Out[161]=





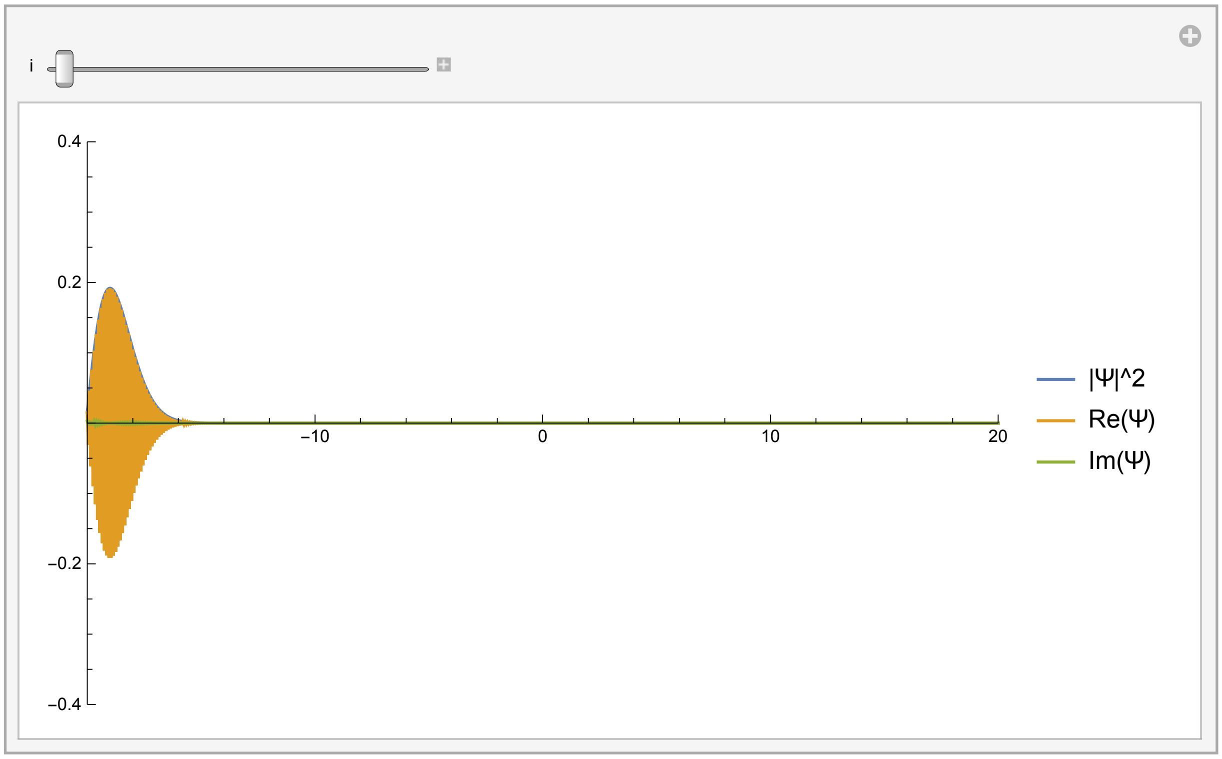




Let’s look at the eigenvectors.

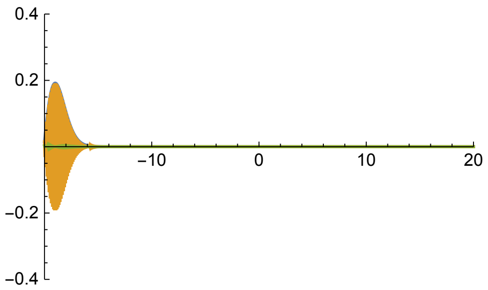
In[162]:=
```
Clear[triEVectorsRe, triEVectorsIm];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 4;
  periodicChoice = 2;
   schemeChoice = 2;
  triEValues = Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <>
      "/" <> periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <>
      "_" <> numberType⟦1⟧ <> "_" <> eigenState⟦1⟧ <> ".csv", "CSV"];
  Do[
   Evaluate[Symbol["triEVectors" <> {"Re", "Im"}⟦i⟧]] =
      Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <>
        periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <> "_" <>
        numberType⟦i⟧ <> "_" <> eigenState⟦2⟧ <> ".csv", "CSV"];,
    {i, 2}];
  triEVectors = Sqrt[triEVectorsRe^2 + triEVectorsIm^2];
 ];
```

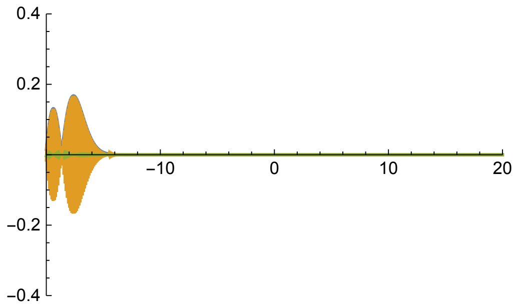
In[164]:=
```
Manipulate[
 ListPlot[{triEVectors⟦All, i⟧, triEVectorsRe⟦All, i⟧, triEVectorsIm⟦All, i⟧},
  Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
  PlotRange → {{-20, 20}, {-.4, .4}}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"}],
 {i, 1, Length[triEVectors⟦All, 1⟧], 1}]
```

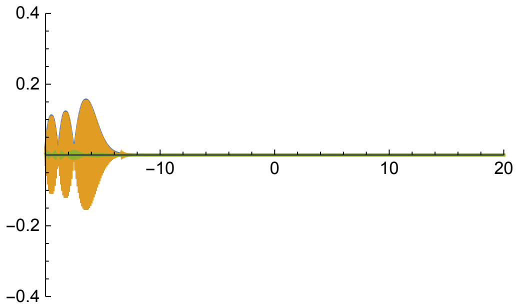
Out[164]=


Many of these do not look sensible (eg 1, 2, and 3, shown in the first row below. 528 looks like a ground state, and then there are a few sensible states after that, (see second row) though we do seem to skip over the first excited state. There are two more good looking batches starting at 555 and 626 (not shown below)

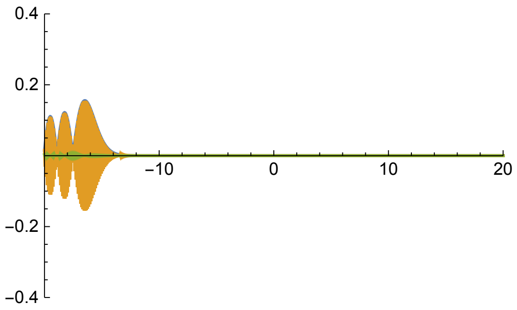
In[165]:=

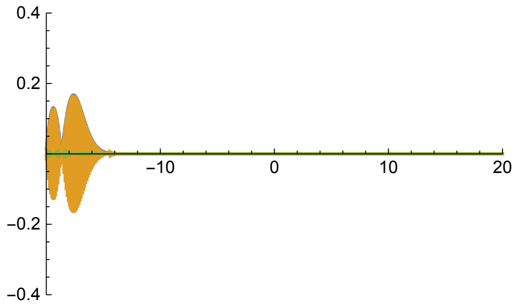


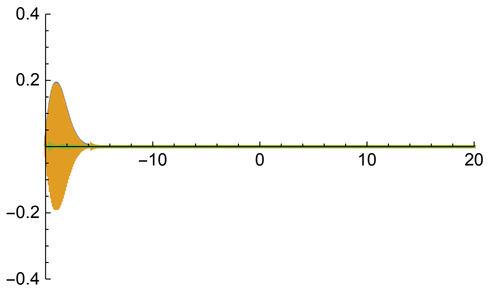




Out[165]=

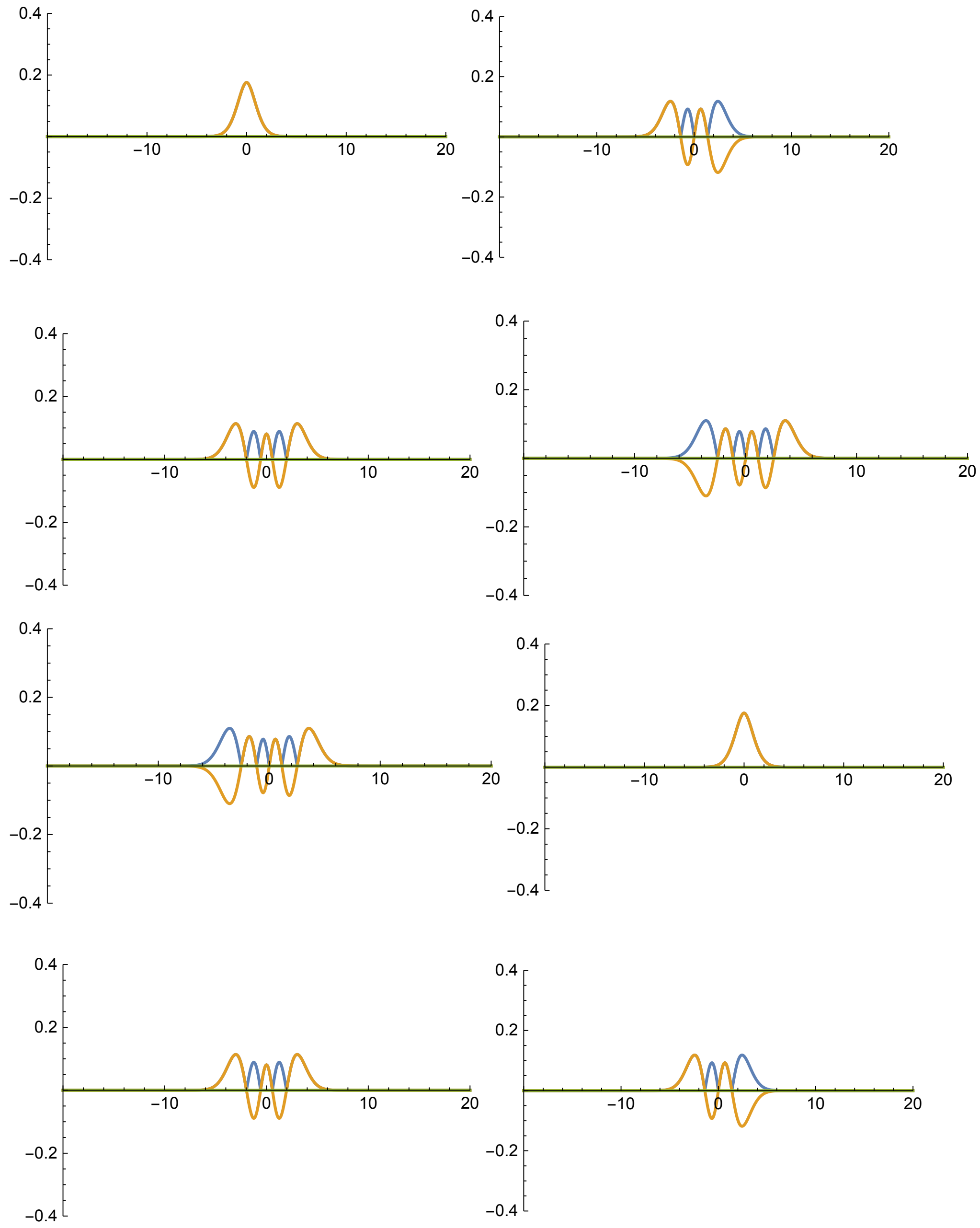






In[166]:=

Out[166]=



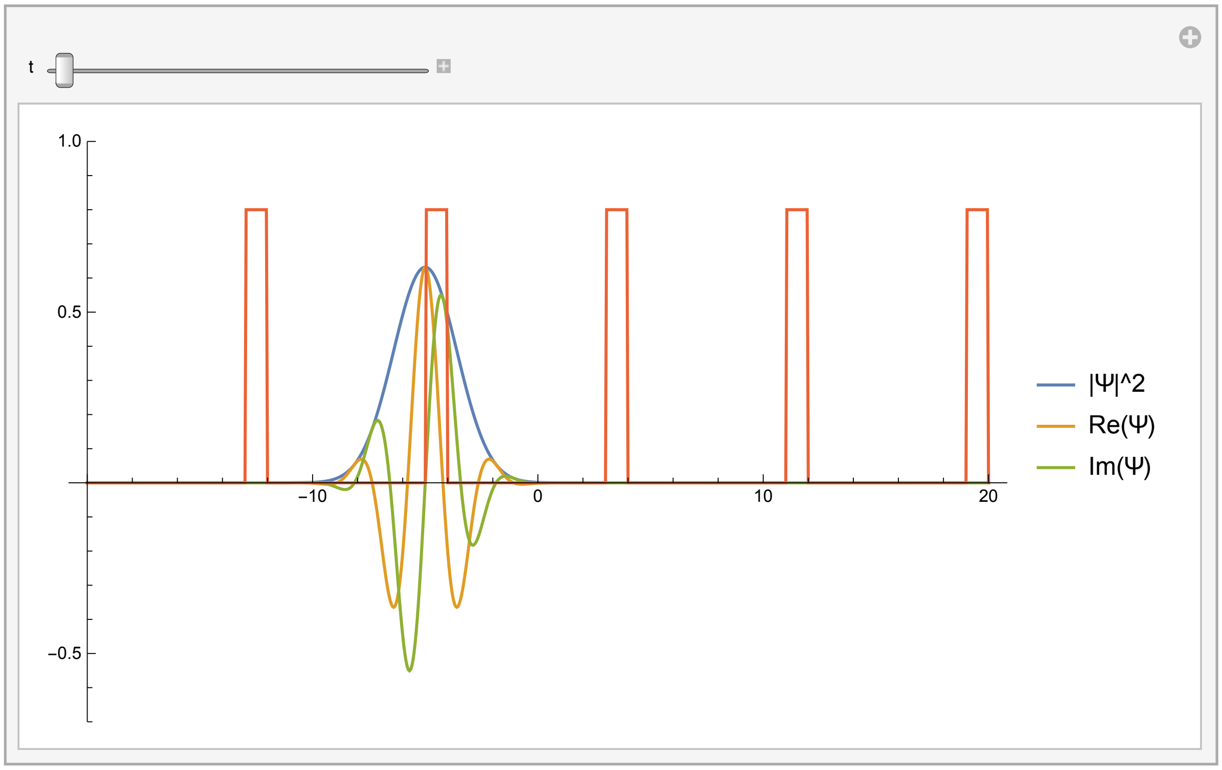
Kronig-Penney

Now let’s do another periodic potential.

With a wave packet

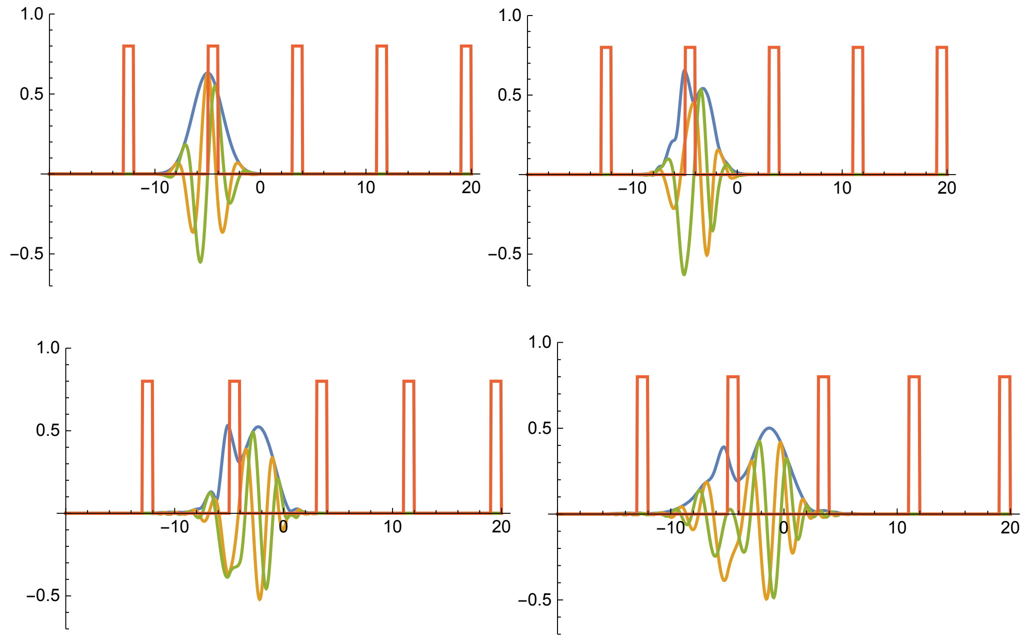
```
In[167]:= Clear[kpCNRe, kpCNIm, kpCN, kpPotential];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 5;
  periodicChoice = 1;
   schemeChoice = 2;
  Do[
   Evaluate[Symbol["kpCN" <> {"Re", "Im"}[[i]]]] = Import[
      path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
       scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"] ;,
   {i, 2}];
  kpPotential = Flatten[Import[
     path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
      scheme[[schemeChoice]] <> "_" <> "Potential" <> ".csv", "CSV"]];
  kpCN = Sqrt[kpCNRe^2 + kpCNIm^2];
 ];
Manipulate[ListPlot[{kpCN[[t, All]], kpCNRe[[t, All]], kpCNIm[[t, All]], kpPotential / 5},
  PlotRange → {-.7, 1}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
  ImageSize → 500, Joined → True, DataRange → {-20, 20},
  AxesOrigin → {-20, 0}], {t, 1, Length[kpCNRe[[All, 1]]], 1}]
```

Out[169]=

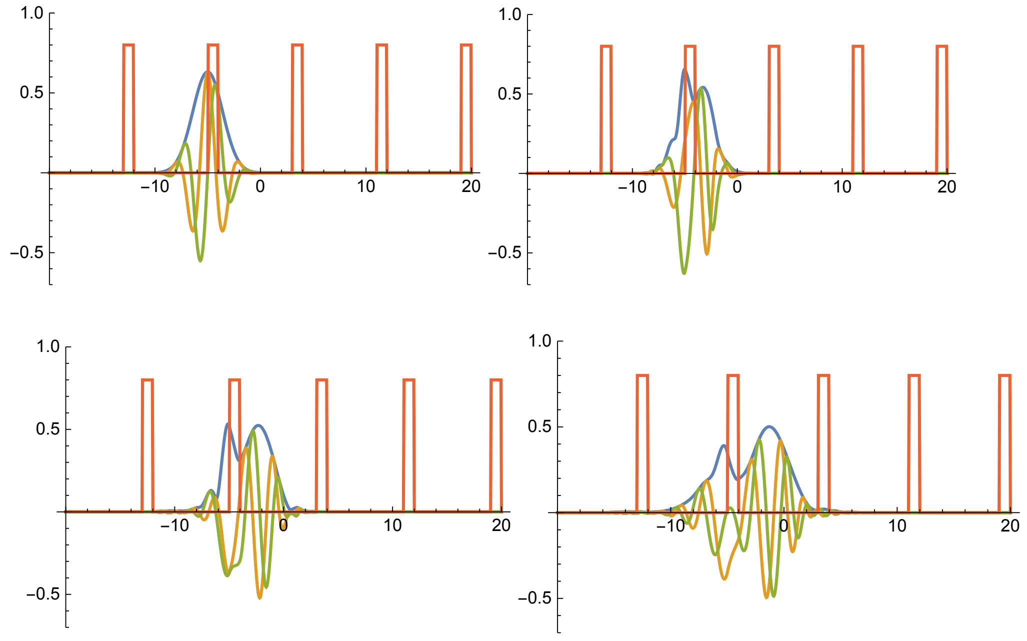


The movie looks best if you slow it down--we can see the that the part of the wave packet trapped on one side of one wall goes in one direction, and the part trapped on the other goes in the other direction. Nifty. Snapshots at t=1, 20, 40, 60, 80, 100, 120, and 140 shown.

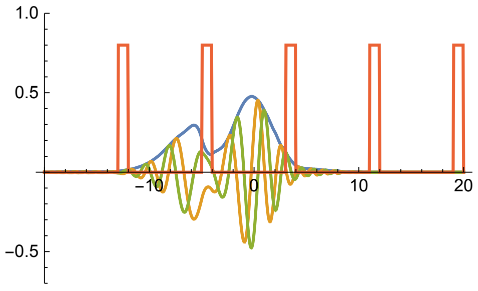
In[170]:=

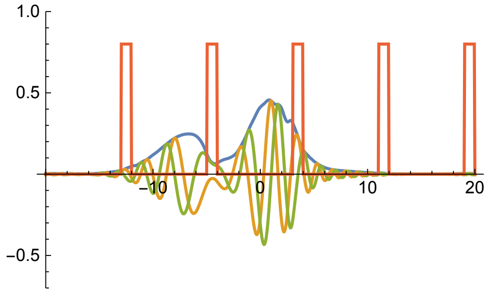


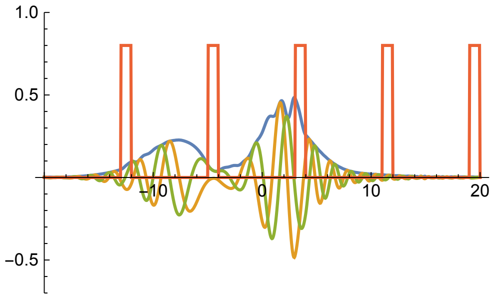
Out[170]=

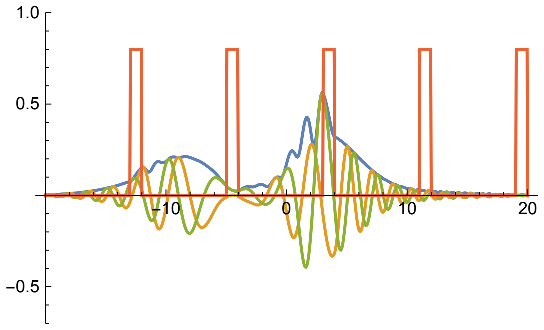


In[171]:=

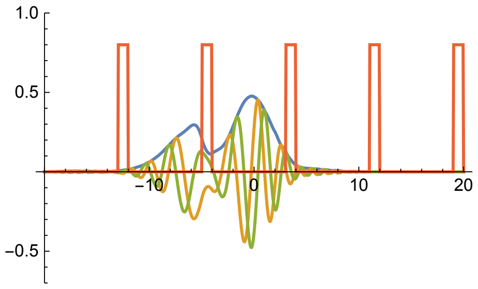


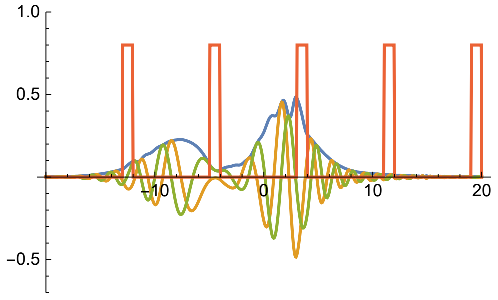


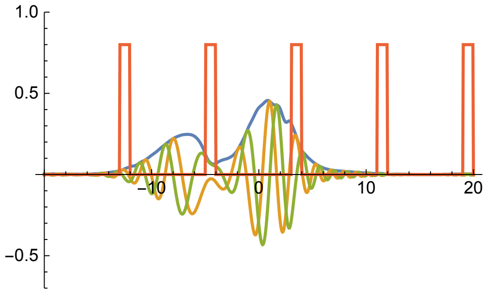


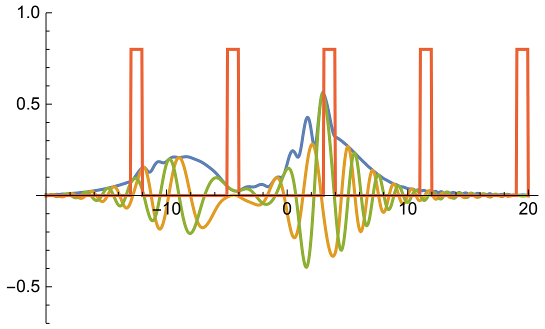


Out[171]=







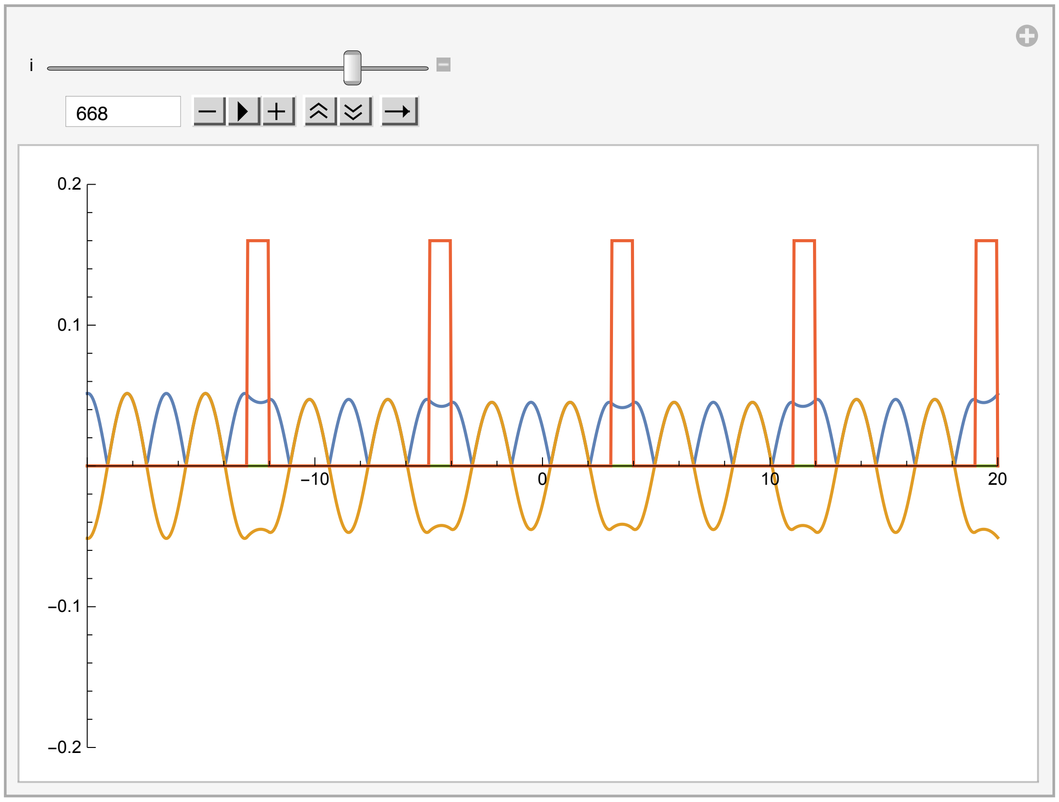


The eigenvectors

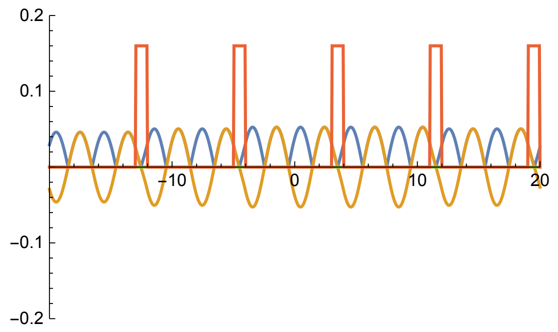
```
In[172]:= Clear[kpEVectorsRe, kpEVectorsIm];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 5;
  periodicChoice = 1;
   schemeChoice = 2;
  kpEValues = Import[path <> "/Runs/" <> cases[[caseChoice]] <>
      "/" <> periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <>
      "_" <> numberType[[1]] <> "_" <> eigenState[[1]] <> ".csv", "CSV"];
  Do[
   Evaluate[Symbol["kpEVectors" <> {"Re", "Im"}[[i]]]] =
     Import[path <> "/Runs/" <> cases[[caseChoice]] <> "/" <>
       periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <> "_" <>
       numberType[[i]] <> "_" <> eigenState[[2]] <> ".csv", "CSV"];,
    {i, 2}];
  kpEVectors = Sqrt[kpEVectorsRe^2 + kpEVectorsIm^2];
 ];
```

```
In[174]:= Manipulate[ListPlot[
    {kpEVectors[[All, i]], kpEVectorsRe[[All, i]], kpEVectorsIm[[All, i]], kpPotential / 25},
    Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
    PlotRange → {{-20, 20}, {-.2, .2}}], {i, 1, Length[kpEVectors[[All, 1]]], 1}]
```

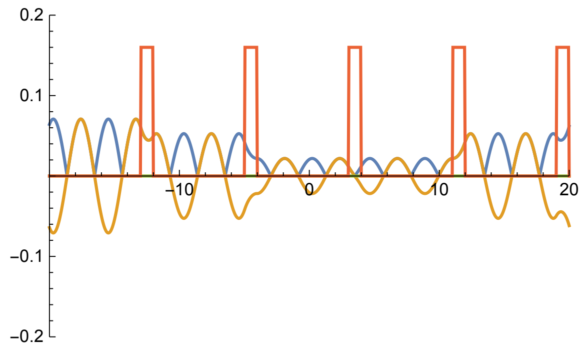
Out[174]=



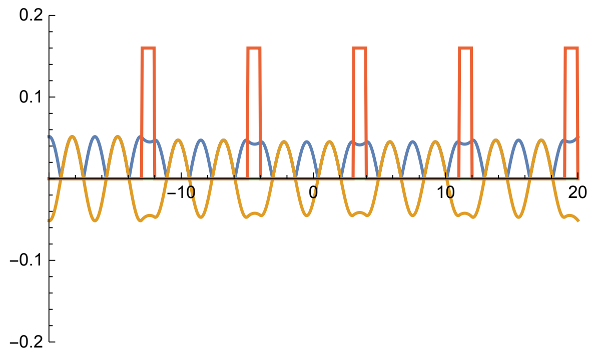
For this one, we have some interesting eigenvectors starting at 638 as shown:



That’s not the ground state, but we can see that the notes are on the barriers. Some of these solutions seem to offer equal aplitudes within each well (such as 638), but others (such as 639) show larger amplitudes in some wells than others (shown below).



One particlarly interesting one is 668, which shows the decay of the amplitude within each well:



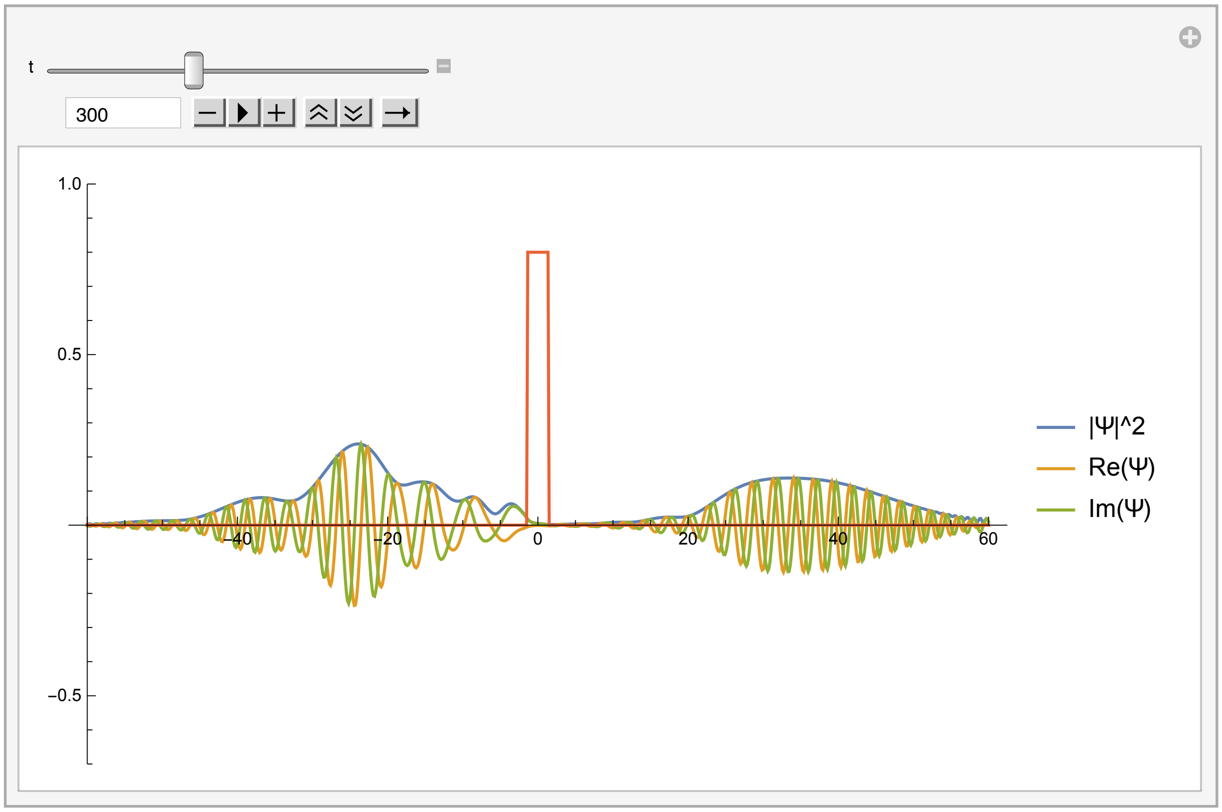
Barrier

Using the energy output from our python script, we can see that the total enery is about 3.95. Roudning a bit, we’ll say that the energy of the wave is 4 and look at barrier heights of 4, 2, and 6. The range is now -60 to +60 units, which allows us to see the long range dynamics. There are still 801 spacial grid points and 800 time steps. The barriers are all one unit wide.

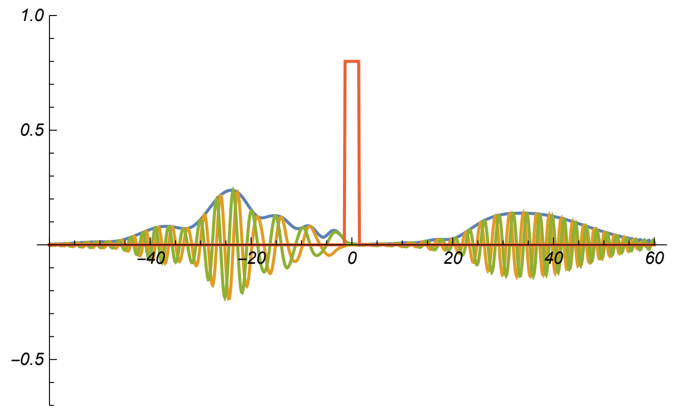
Height = E

In[175]:=
```
Clear[b1CNRe, b1CNIm, b1CN, b1Potential];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 8;
  periodicChoice = 2;
   schemeChoice = 2;
  Do[
   Evaluate[Symbol["b1CN" <> {"Re", "Im"}⟦i⟧]] = Import[
      path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
       scheme⟦schemeChoice⟧ <> "_" <> numberType⟦i⟧ <> ".csv", "CSV"];,
   {i, 2}];
  b1Potential = Flatten[Import[
     path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
      scheme⟦schemeChoice⟧ <> "_" <> "Potential" <> ".csv", "CSV"]];
  b1CN = Sqrt[b1CNRe^2 + b1CNIm^2];
 ];
Manipulate[ListPlot[{b1CN⟦t, All⟧, b1CNRe⟦t, All⟧, b1CNIm⟦t, All⟧, b1Potential / 5},
  PlotRange → {-.7, 1}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
  ImageSize → 500, Joined → True, DataRange → {-60, 60},
  AxesOrigin → {-60, 0}], {t, 1, Length[b1CNRe⟦All, 1⟧], 1}]
```

Out[177]=



If you slow this down, it looks like about half of the wave gets transmitted and the other half gets reflected. Looking at time step 300 (right before the wave starts interacting with the edges), most of the wave has been transmitted:



The total normalization is

```
In[178]:= Total[b1CN[[300, All]]]

Out[178]= 55.7697885756006123
```

Our current run has 801 grid points for total grid of 120 units (meaning each unit is about 7 grid points), and the barrier is 1 unit wide. So we want the first 397 grid points for the reflected wave and the last 397 for the transmitted wave.

```
In[179]:= 800 / 120 // N

Out[179]= 6.66667

In[180]:= Dimensions[b1CN[[300]]]

Out[180]= {801}
```

The fraction reflected is

```
In[181]:= Total[Table[b1CN[[300, i]], {i, 1, 397}]] / Total[b1CN[[300, All]]]

Out[181]= 0.5659875872637265 49
```

And the fraction transmitted is

```
In[182]:= Total[Table[b1CN[[300, i]], {i, 404, 801}]] / Total[b1CN[[300, All]]]

Out[182]= 0.4334509251968957 55
```

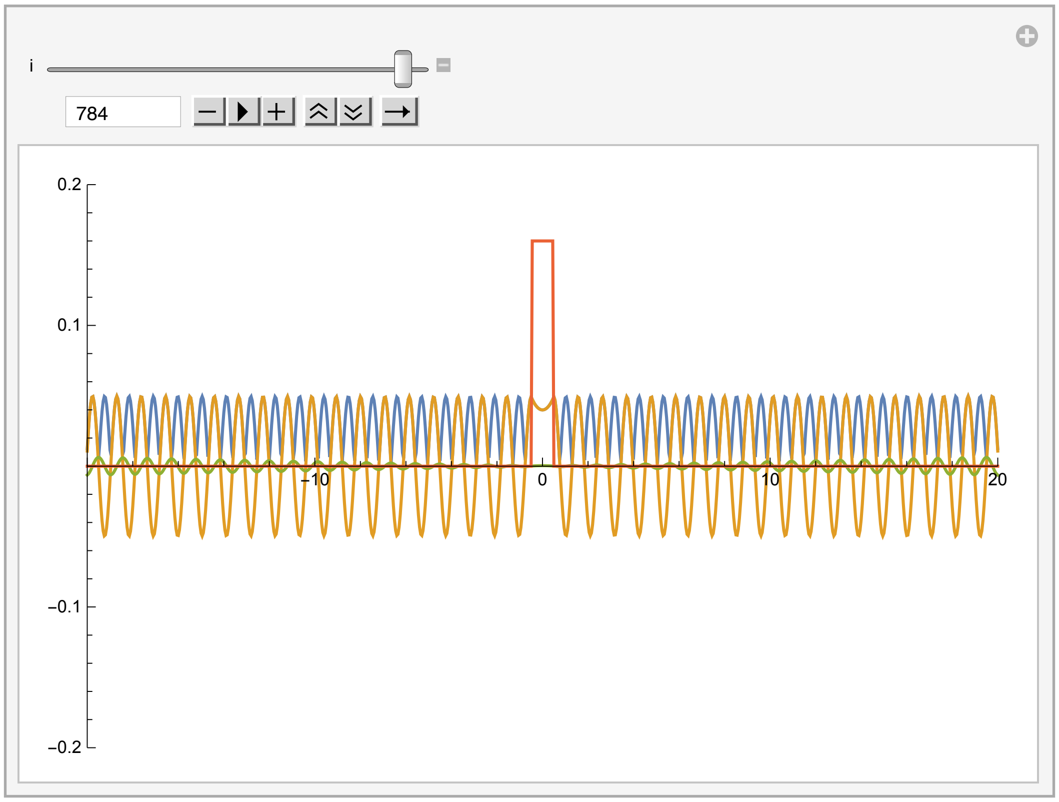
Let’s compare that to the analytical result. The analytical result is that the fraction transmitted is $T = \frac{1}{1\text{+mL^2 V0/2 hbar}}$

Note that this is independent of the initial speed of the wave. In our units, hbar is one. V is 4, m is 1, and L is also one (by our choice). This says that T should be 1/(1+2)=0.33. So we’re finding that we’ve transmitted more than we’d expect.

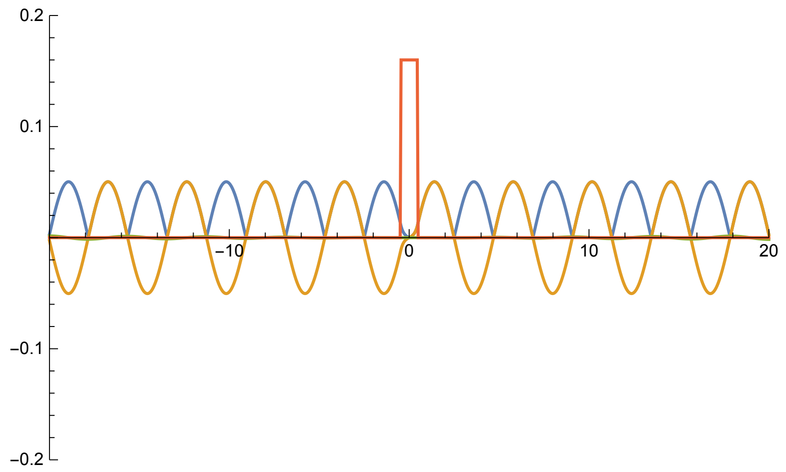
Let’s look at these eigenvectors

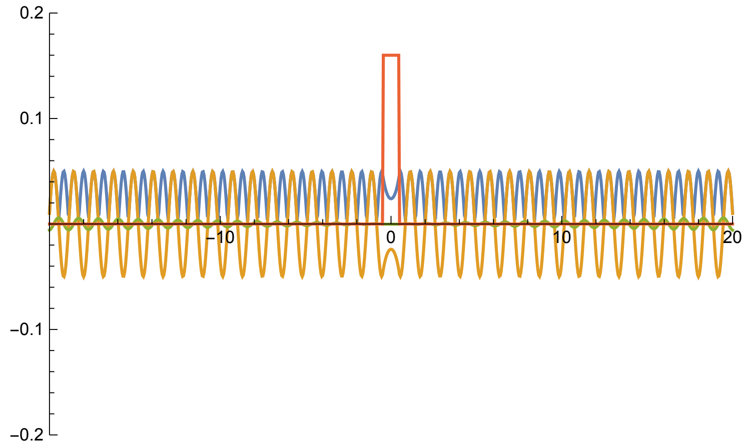
```
In[183]:= Clear[b1EVectorsRe, b1EVectorsIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 8;
            periodicChoice = 2;
             schemeChoice = 2;
            b1EValues = Import[path <> "/Runs/" <> cases[[caseChoice]] <>
                "/" <> periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <>
                "_" <> numberType[[1]] <> "_" <> eigenState[[1]] <> ".csv", "CSV"];
            Do[
             Evaluate[Symbol["b1EVectors" <> {"Re", "Im"}[[i]]]] =
                Import[path <> "/Runs/" <> cases[[caseChoice]] <> "/" <>
                  periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <> "_" <>
                  numberType[[i]] <> "_" <> eigenState[[2]] <> ".csv", "CSV"];,
             {i, 2}];
            b1EVectors = Sqrt[b1EVectorsRe^2 + b1EVectorsIm^2];
           ];

In[185]:= Manipulate[ListPlot[
            {b1EVectors[[All, i]], b1EVectorsRe[[All, i]], b1EVectorsIm[[All, i]], b1Potential / 25},
            Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
            PlotRange → {{-20, 20}, {-.2, .2}}], {i, 1, Length[b1EVectors[[All, 1]]], 1}]
```

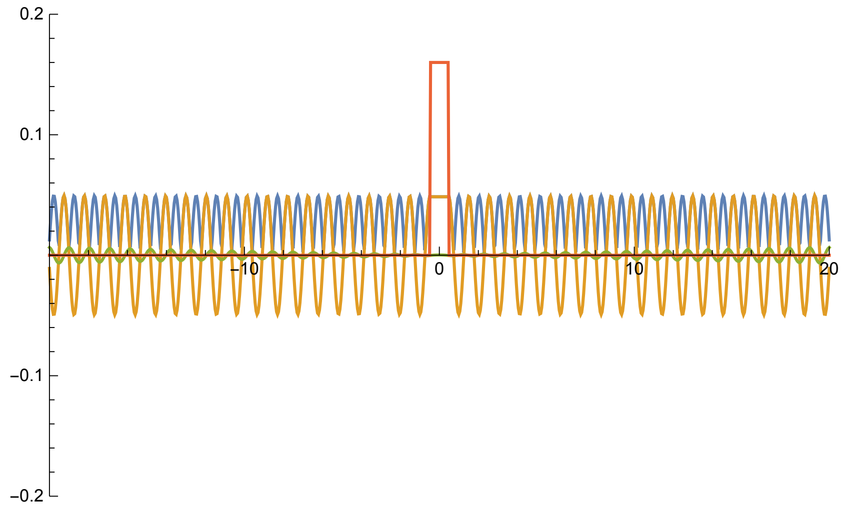
Out[185]=



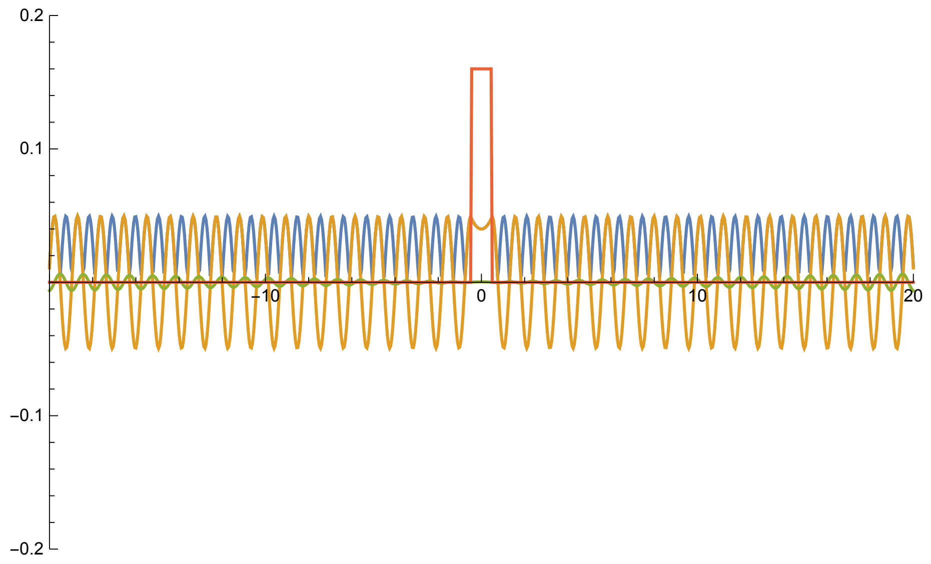
Check out some of these eigenvectors starting at 740 (740, 743, and 745 pictured below). We see oscillatory behavior. 745 shows a flat amplitude across the barrier.







Interestingly, a couple of vectors starting at 784 have significant imaginary components (in green):

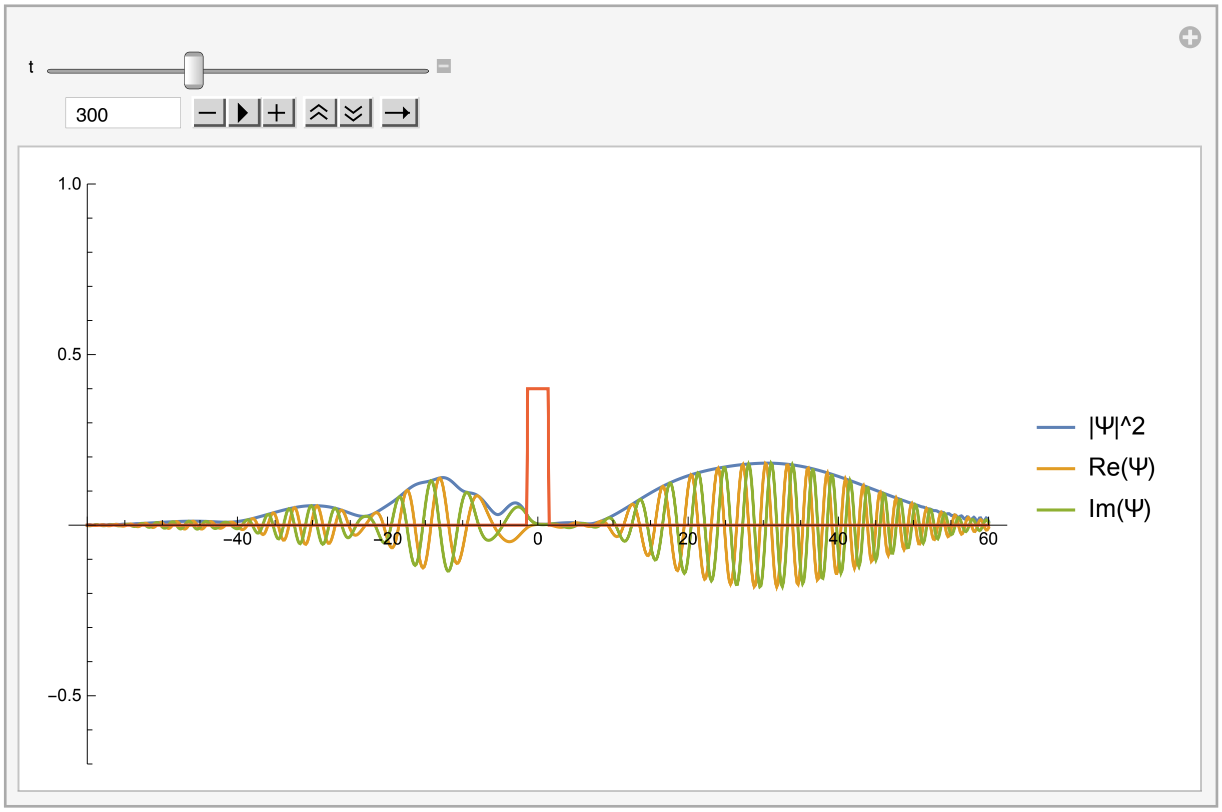


In[186]:=

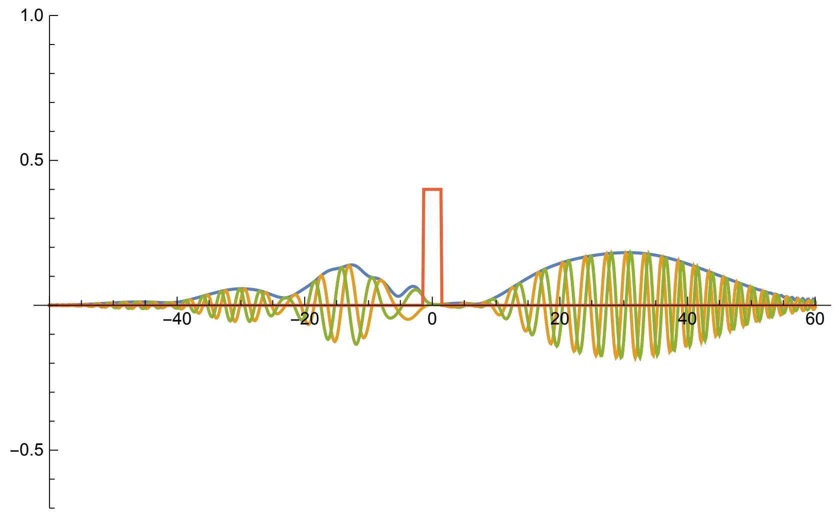
Height < E

```
In[187]:= Clear[b2CNRe, b2CNIm, b2CN, b2Potential];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 9;
            periodicChoice = 2;
             schemeChoice = 2;
            Do[
             Evaluate[Symbol["b2CN" <> {"Re", "Im"}⟦i⟧]] = Import[
                 path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
                  scheme⟦schemeChoice⟧ <> "_" <> numberType⟦i⟧ <> ".csv", "CSV"];,
             {i, 2}];
            b2Potential = Flatten[Import[
                path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
                 scheme⟦schemeChoice⟧ <> "_" <> "Potential" <> ".csv", "CSV"]];
            b2CN = Sqrt[b2CNRe^2 + b2CNIm^2];
           ];
          Manipulate[ListPlot[{b2CN⟦t, All⟧, b2CNRe⟦t, All⟧, b2CNIm⟦t, All⟧, b2Potential / 5},
            PlotRange → {-.7, 1}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            ImageSize → 500, Joined → True, DataRange → {-60, 60},
            AxesOrigin → {-60, 0}], {t, 1, Length[b2CNRe⟦All, 1⟧], 1}]
```

Out[189]=



Let’s look at timestep 300 for this case (shown below).



```
In[190]:= Total[Table[b2CN[[300, i]], {i, 1, 397}]] / Total[b2CN[[300, All]]]
Out[190]= 0.313061541774715315
```

And the fraction transmitted is

```
In[191]:= Total[Table[b2CN[[300, i]], {i, 404, 801}]] / Total[b2CN[[300, All]]]
Out[191]= 0.686448569932912189
```

Those numbers make sense based on the picture. Let’s check the analytical result. We expect that the fraction transmitted is

In[192]:=
$$T = \frac{1}{1 + \frac{\text{V0^2}\sin\text{^2}\left(\sqrt{2\,m(E-V)/\text{hbar^2}}\;L\right)}{4\,E(E-V)}}$$

Out[192]=
$$\frac{1}{1 + \frac{\text{L}\,\text{sin}^2\,\text{V0}^2\sqrt{\frac{\text{m}[\text{e}-\text{V}]}{\text{hbar}^2}}}{2\sqrt{2}\;\text{e}[\text{e}-\text{V}]}}$$

Plugging in our numbers yields

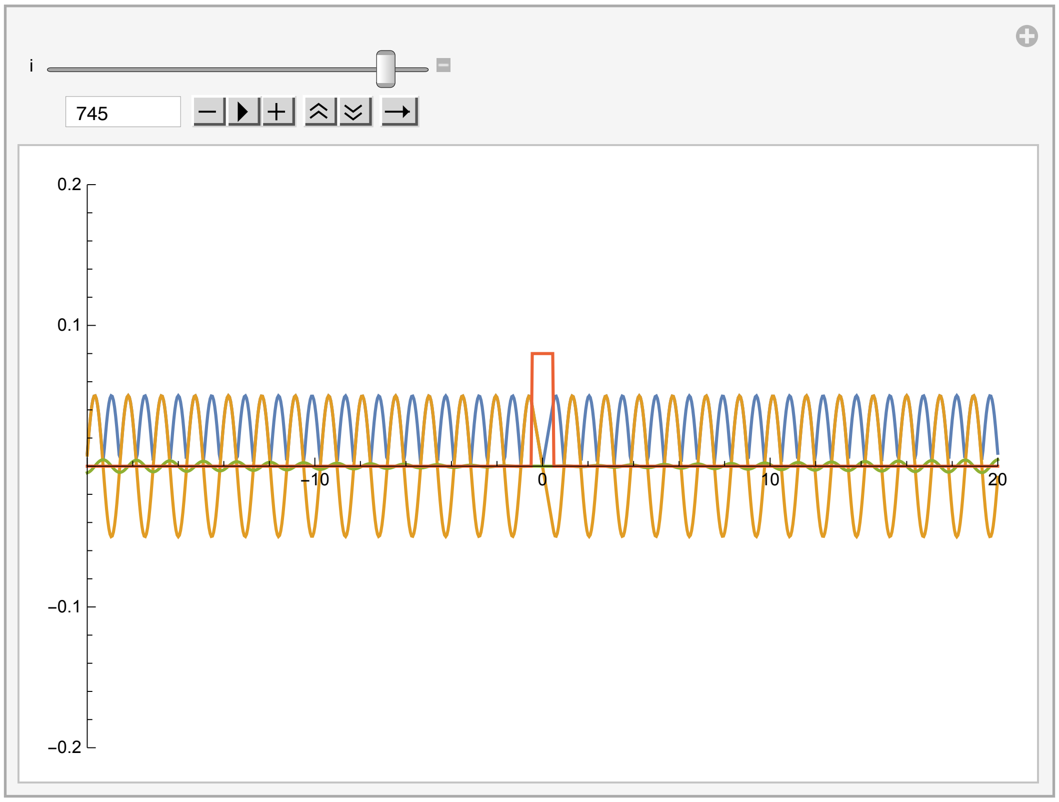
```
In[193]:= T = 1 / (1 + (2^2 (Sin[2*(2)*1])^2) / (4*4*2)) // N
Out[193]= 0.933189
```

That’s not at all close to what we got. The analytical result makes sense--quite a bit should get over that barrier. Having our numerical solution underestimate this quantity by 50% isn’t a very good fit.

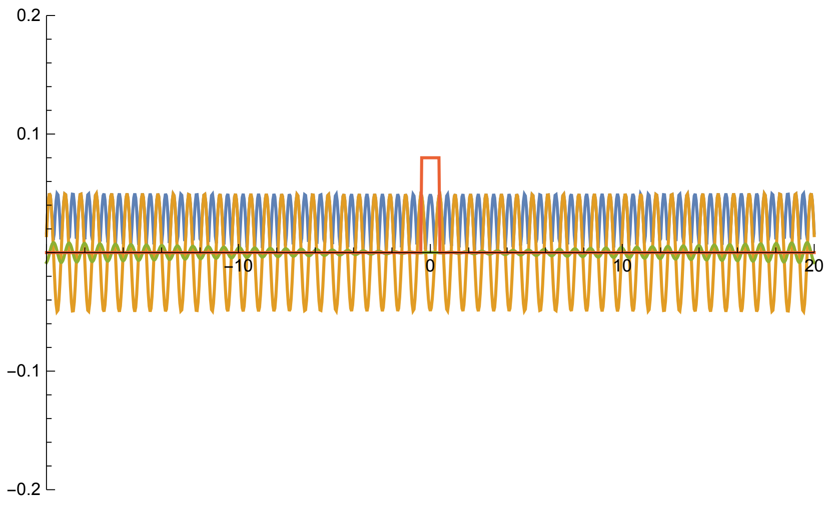
Let’s look at the eigenvectors

```
In[194]:= Clear[b2EVectorsRe, b2EVectorsIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 9;
            periodicChoice = 2;
             schemeChoice = 2;
            kpEValues = Import[path <> "/Runs/" <> cases[[caseChoice]] <>
                "/" <> periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <>
                "_" <> numberType[[1]] <> "_" <> eigenState[[1]] <> ".csv", "CSV"];
            Do[
             Evaluate[Symbol["b2EVectors" <> {"Re", "Im"}[[i]]]] =
                Import[path <> "/Runs/" <> cases[[caseChoice]] <> "/" <>
                  periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <> "_" <>
                  numberType[[i]] <> "_" <> eigenState[[2]] <> ".csv", "CSV"];,
             {i, 2}];
            b2EVectors = Sqrt[b2EVectorsRe^2 + b2EVectorsIm^2];
           ];

In[196]:= Manipulate[ListPlot[
            {b2EVectors[[All, i]], b2EVectorsRe[[All, i]], b2EVectorsIm[[All, i]], b2Potential / 25},
            Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
            PlotRange → {{-20, 20}, {-.2, .2}}], {i, 1, Length[b2EVectors[[All, 1]]], 1}]
```

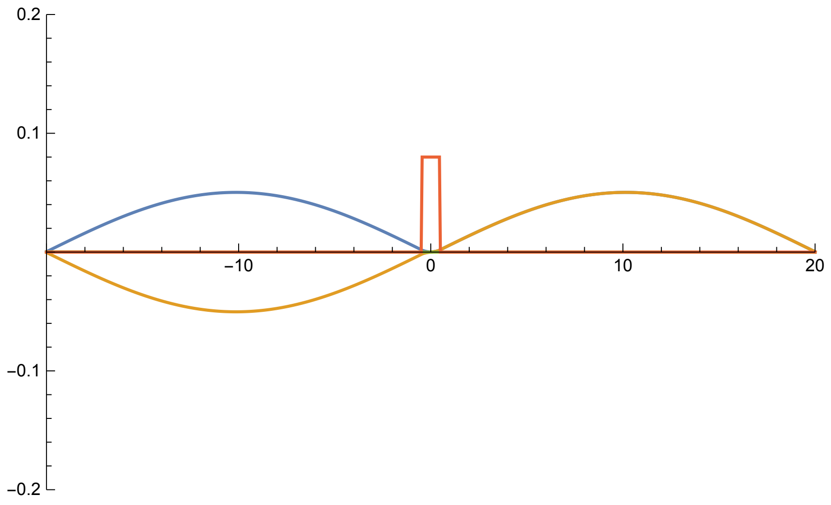
Out[196]=



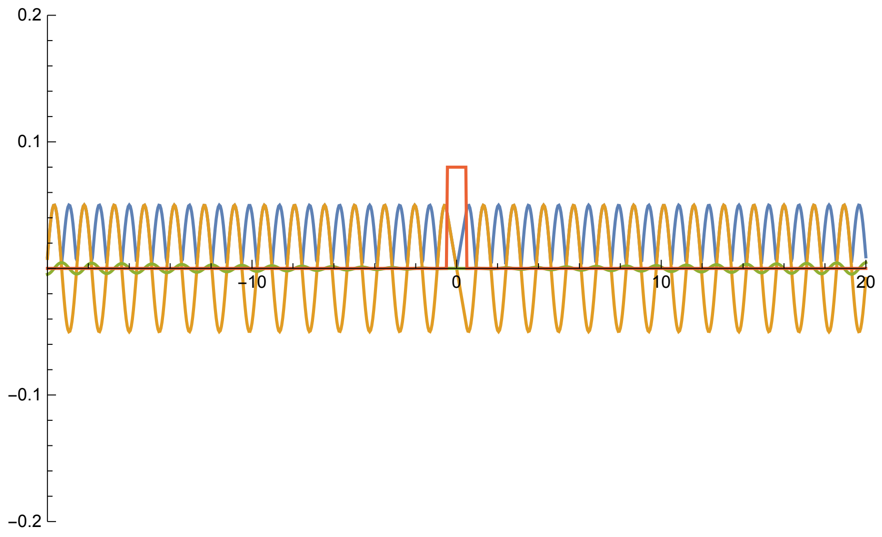
Some of the low(er) energy eigenstates can be seen around 706:



and the ground state appears at 728:



Around 745 we see some where there is more clear interaction with the barrier:



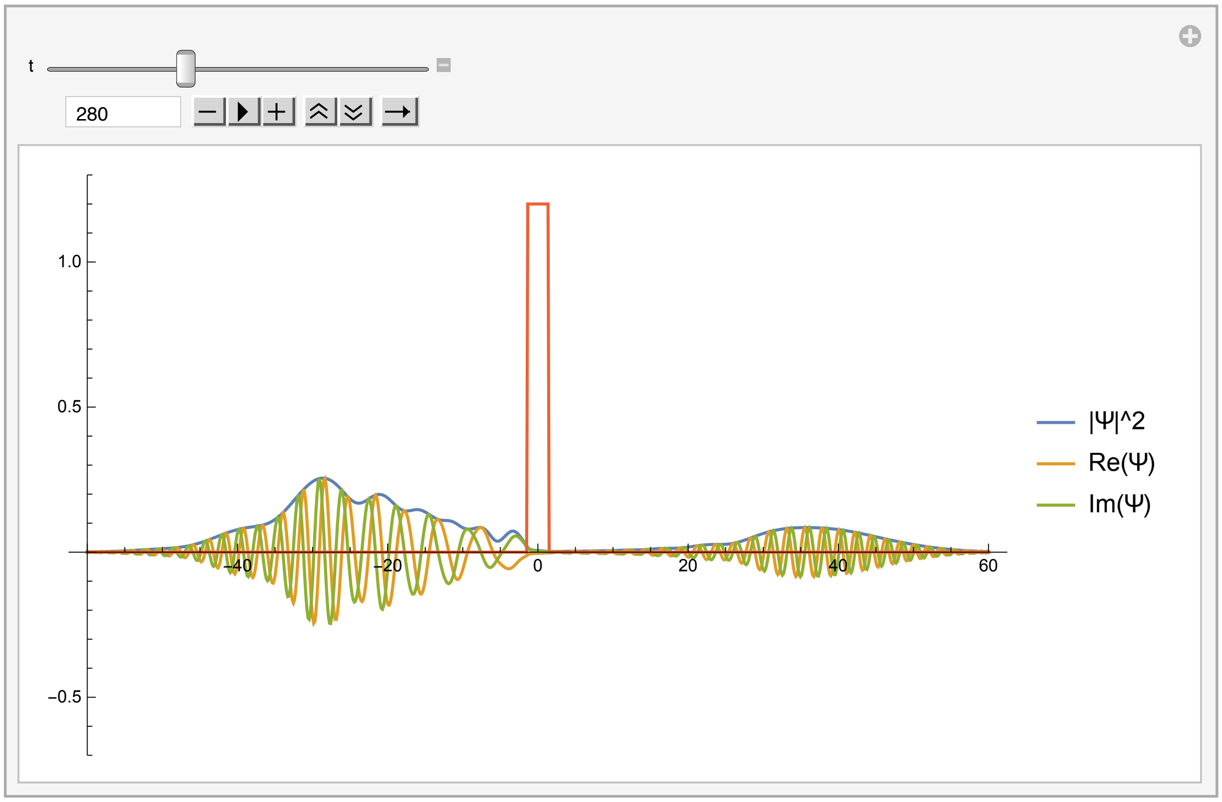
We expect all of our barriers to have essentially the same eigenvectors, just scaled (and perhaps ordered) differently.

Height > E

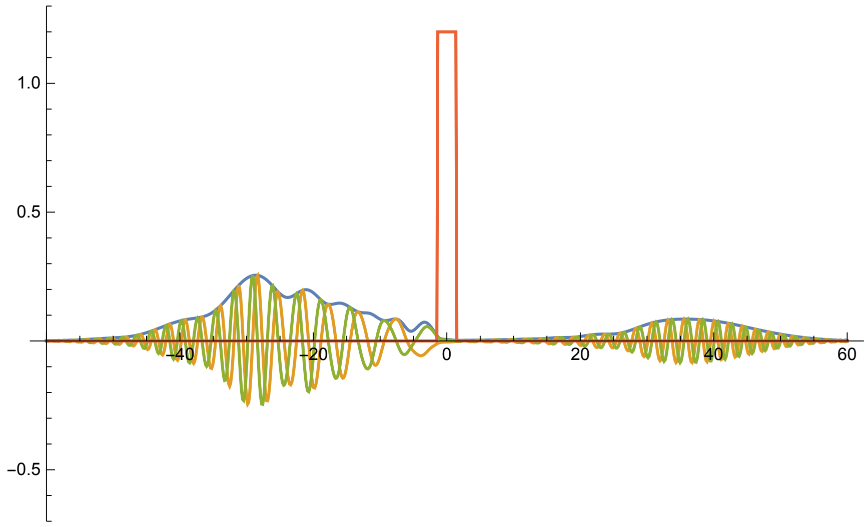
```
In[197]:= Clear[b3CNRe, b3CNIm, b3CN, b3Potential];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 10;
            periodicChoice = 2;
             schemeChoice = 2;
            Do[
             Evaluate[Symbol["b3CN" <> {"Re", "Im"}⟦i⟧]] = Import[
                path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
                 scheme⟦schemeChoice⟧ <> "_" <> numberType⟦i⟧ <> ".csv", "CSV"];,
             {i, 2}];
            b3Potential = Flatten[Import[
               path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <> periodicType⟦periodicChoice⟧ <>
                scheme⟦schemeChoice⟧ <> "_" <> "Potential" <> ".csv", "CSV"]];
            b3CN = Sqrt[b3CNRe^2 + b3CNIm^2];
           ];
```

```
In[199]:= Manipulate[ListPlot[{b3CN⟦t, All⟧, b3CNRe⟦t, All⟧, b3CNIm⟦t, All⟧, b3Potential / 5},
            PlotRange → {-.7, 1.3}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
            ImageSize → 500, Joined → True, DataRange → {-60, 60},
            AxesOrigin → {-60, 0}], {t, 1, Length[b3CNRe⟦All, 1⟧], 1}]
```

Out[199]=



Let's look at timestep 280 for this one:



The fraction reflected is

```
In[200]:= Total[Table[b3CN[[280, i]], {i, 1, 397}]] / Total[b3CN[[280, All]]]

Out[200]= 0.748817863813825201
```

And the fraction transmitted is

```
In[201]:= Total[Table[b3CN[[280, i]], {i, 404, 801}]] / Total[b3CN[[280, All]]]

Out[201]= 0.250499939965893153
```

Those numbers make sense based on the picture. Let’s look at the analytical result. We should have

In[202]:=
$$T = \frac{1}{1 + \frac{\mathrm{V0\hat{}2\,sinh\hat{}2}\left(\sqrt{2\,m(E-V)/\mathrm{hbar\hat{}2}}\;L\right)}{4\,E(V-E)}}$$

Out[202]=
$$\frac{1}{1 + \frac{\mathrm{L}\,\mathrm{sinh}^2\,\mathrm{V0}^2\sqrt{\frac{\mathrm{m}[\mathrm{e}-\mathrm{V}]}{\mathrm{hbar}^2}}}{2\sqrt{2}\;\mathrm{e}[-\mathrm{e}+\mathrm{V}]}}$$

Plugging in our numbers yields

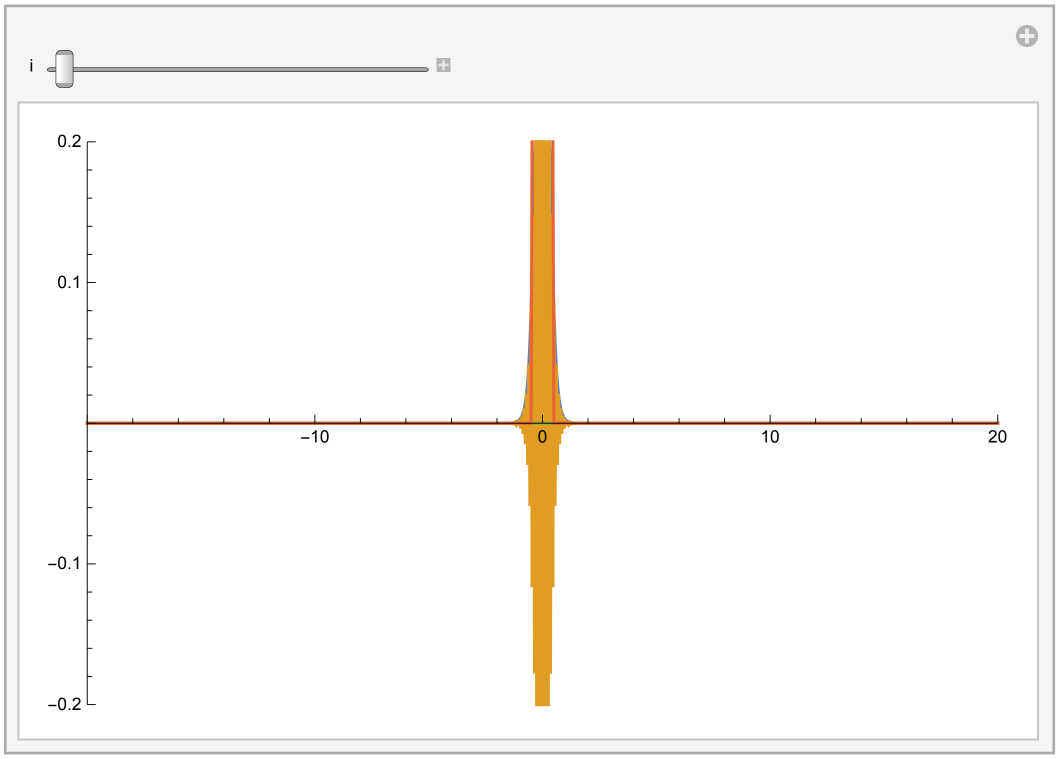
In[203]:=
$$\mathtt{T} = \frac{1}{1 + \frac{\mathtt{2\hat{}2\ (Sinh[2*(2)*1])\hat{}2}}{\mathtt{4*4*2}}}\ \mathtt{//\ N}$$

```
Out[203]= 0.0106278
```

Again we see something that is not at all consistent with our numerical results.

```
In[204]:= Clear[b3EVectorsRe, b3EVectorsIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 10;
            periodicChoice = 2;
             schemeChoice = 2;
            kpEValues = Import[path <> "/Runs/" <> cases[[caseChoice]] <>
                "/" <> periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <>
                "_" <> numberType[[1]] <> "_" <> eigenState[[1]] <> ".csv", "CSV"];
            Do[
             Evaluate[Symbol["b3EVectors" <> {"Re", "Im"}[[i]]]] =
                Import[path <> "/Runs/" <> cases[[caseChoice]] <> "/" <>
                  periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <> "_" <>
                  numberType[[i]] <> "_" <> eigenState[[2]] <> ".csv", "CSV"];,
             {i, 2}];
            b3EVectors = Sqrt[b3EVectorsRe^2 + b3EVectorsIm^2];
           ];

In[206]:= Manipulate[ListPlot[
            {b3EVectors[[All, i]], b3EVectorsRe[[All, i]], b3EVectorsIm[[All, i]], b3Potential / 25},
            Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
            PlotRange → {{-20, 20}, {-.2, .2}}], {i, 1, Length[b3EVectors[[All, 1]]], 1}]
```

Out[206]=



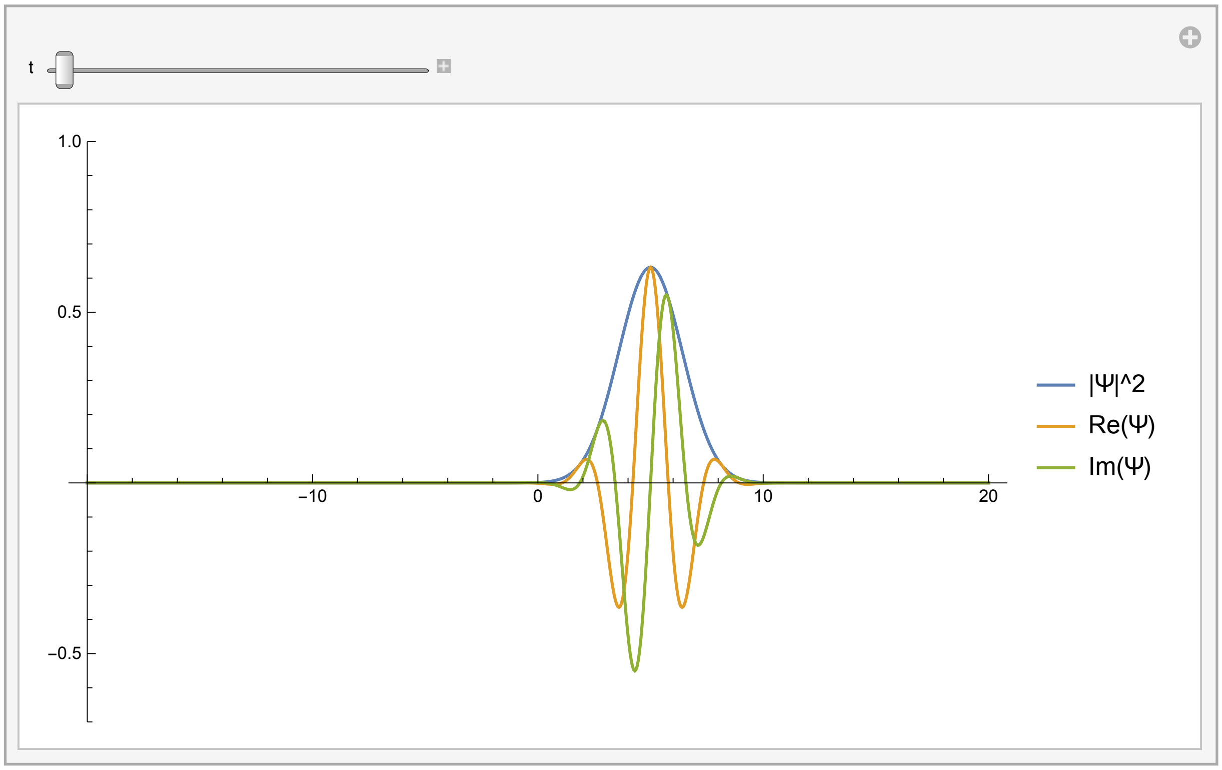
Check out these eigenvectors starting at 695. As expected, this looks like the previous eigenvectors. The ground state is at 726. A snapshot is below.

V=ix

Here is the imagineary potential.

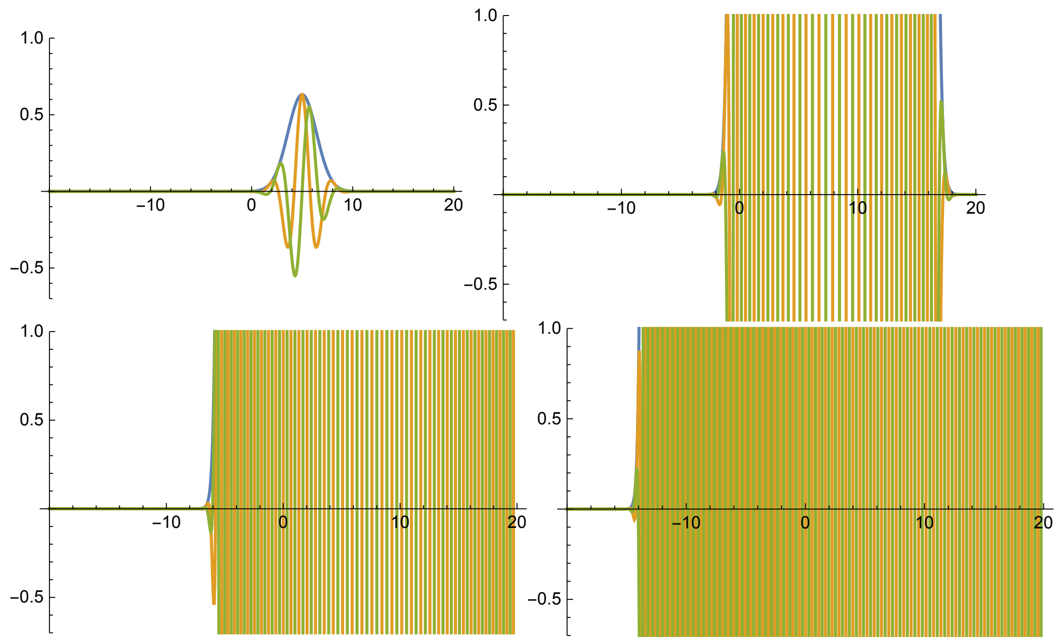
In[207]:=
```
Clear[imagCNRe, imagCNIm, imagCN, imagPotential];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 6;
  periodicChoice = 2;
   schemeChoice = 2;
  Do[
   Evaluate[Symbol["imagCN" <> {"Re", "Im"}[[i]]]] = Import[
      path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
       scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"] ;,
    {i, 2}];
  imagPotential = Flatten[Import[
     path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
      scheme[[schemeChoice]] <> "_" <> "Potential" <> ".csv", "CSV"]];
  imagCN = Sqrt[imagCNRe^2 + imagCNIm^2];
 ];
Manipulate[
 ListPlot[{imagCN[[t, All]], imagCNRe[[t, All]], imagCNIm[[t, All]], Im[imagPotential]},
  PlotRange → {-.7, 1}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
  ImageSize → 500, Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}],
 {t, 1, Length[complexCNRe[[All, 1]]], 1}]
```

Out[209]=



In[210]:=

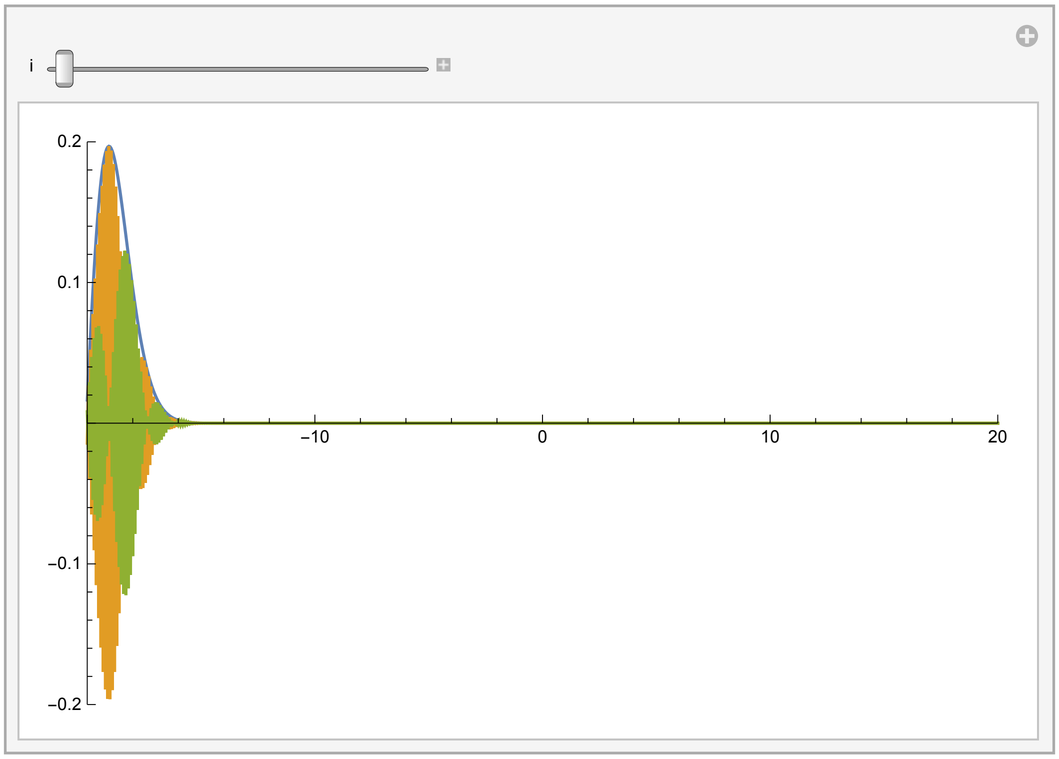
Something really strange seems to happen with our wave packet--it grows without bound. That's a bit alarming! So even though this potential supposedly has real eigenvalues, it seems like energy is not conserved. That's a bit of a problem. Check out time steps 1, 10, 20 and 50 below.



In[211]:=
```
Clear[imagEVectorsRe, imagEVectorsIm];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 6;
  periodicChoice = 2;
   schemeChoice = 2;
  imagEValuesRE = Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <>
      "/" <> periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <>
      "_" <> numberType⟦1⟧ <> "_" <> eigenState⟦1⟧ <> ".csv", "CSV"];
  imagEValuesIm = Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <>
      "/" <> periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <>
      "_" <> numberType⟦2⟧ <> "_" <> eigenState⟦1⟧ <> ".csv", "CSV"];
  Do[
   Evaluate[Symbol["imagEVectors" <> {"Re", "Im"}⟦i⟧]] =
      Import[path <> "/Runs/" <> cases⟦caseChoice⟧ <> "/" <>
        periodicType⟦periodicChoice⟧ <> scheme⟦schemeChoice⟧ <> "_" <>
        numberType⟦i⟧ <> "_" <> eigenState⟦2⟧ <> ".csv", "CSV"];,
   {i, 2}];
  imagEVectors = Sqrt[imagEVectorsRe^2 + imagEVectorsIm^2];
 ];
```

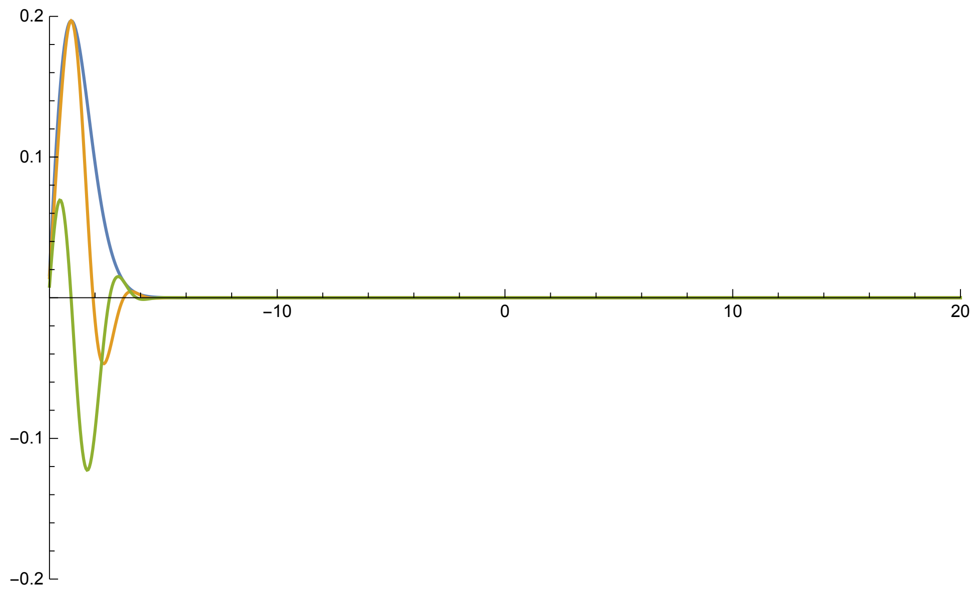
```
In[213]:= Manipulate[ListPlot[{imagEVectors[[All, i]], imagEVectorsRe[[All, i]],
    imagEVectorsIm[[All, i]], imagPotential / 25}, Joined → True, DataRange → {-20, 20},
   AxesOrigin → {-20, 0}, ImageSize → 500, PlotRange → {{-20, 20}, {-.2, .2}}],
  {i, 1, Length[imagEVectors[[All, 1]]], 1}]
```

Out[213]=

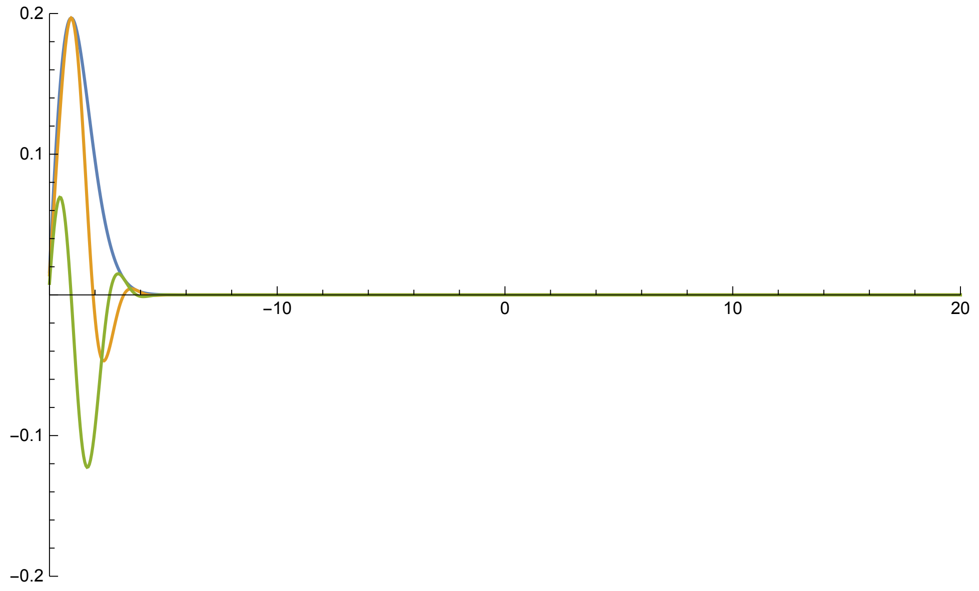


These eigenstates look like complex wave packets. Check out 319:

In[214]:=

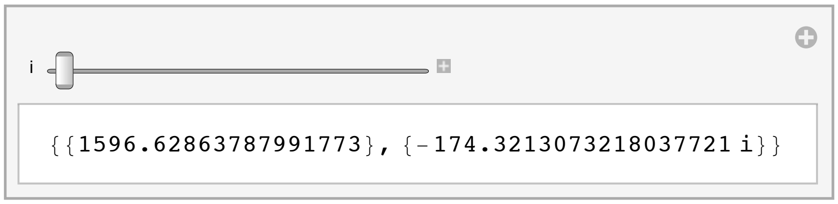


Out[214]=



In[215]:=
```
Manipulate[{imagEValuesRE[[i]], imagEValuesIm[[1]] "i"},
 {i, 1, Length[imagEValues], 1}]
```

Out[215]=



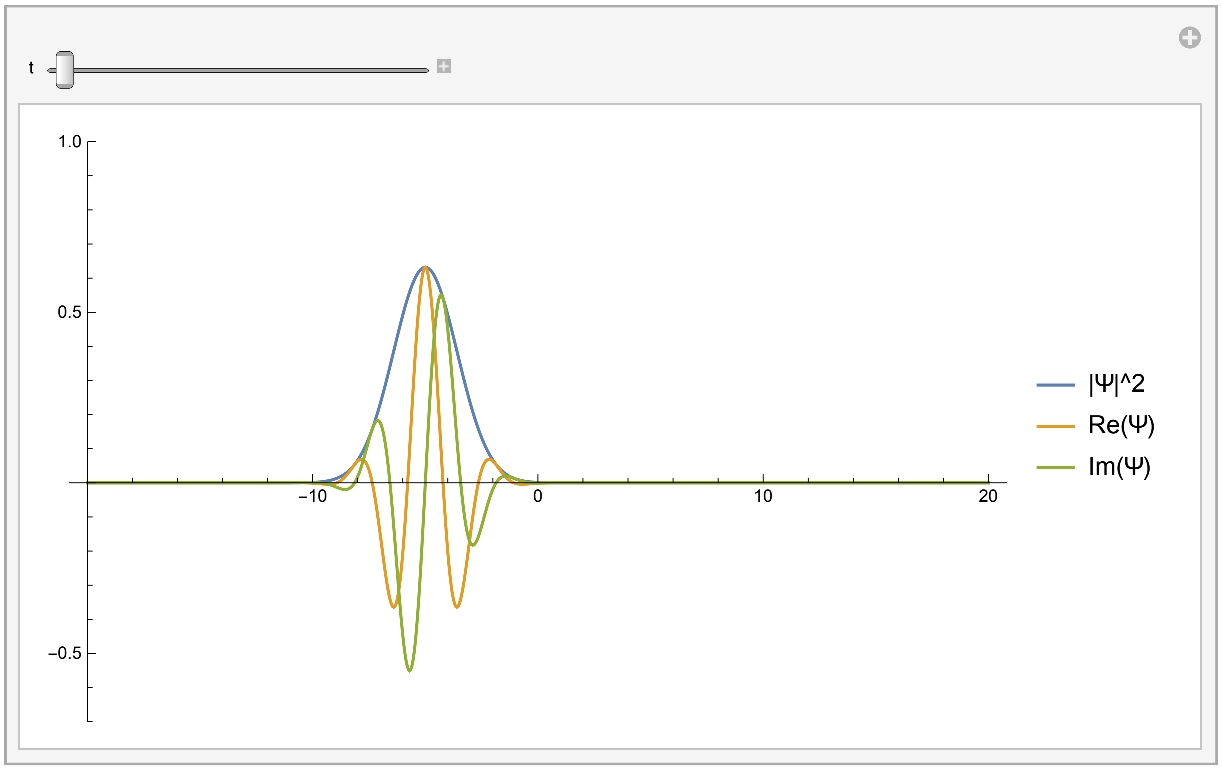
```
{{1596.62863787991773}, {-174.3213073218037721 i}}
```

We expect to see real eigenvalues... but we see THE SAME imaginary component for all eigenvalues. That’s odd.

V=x+ix

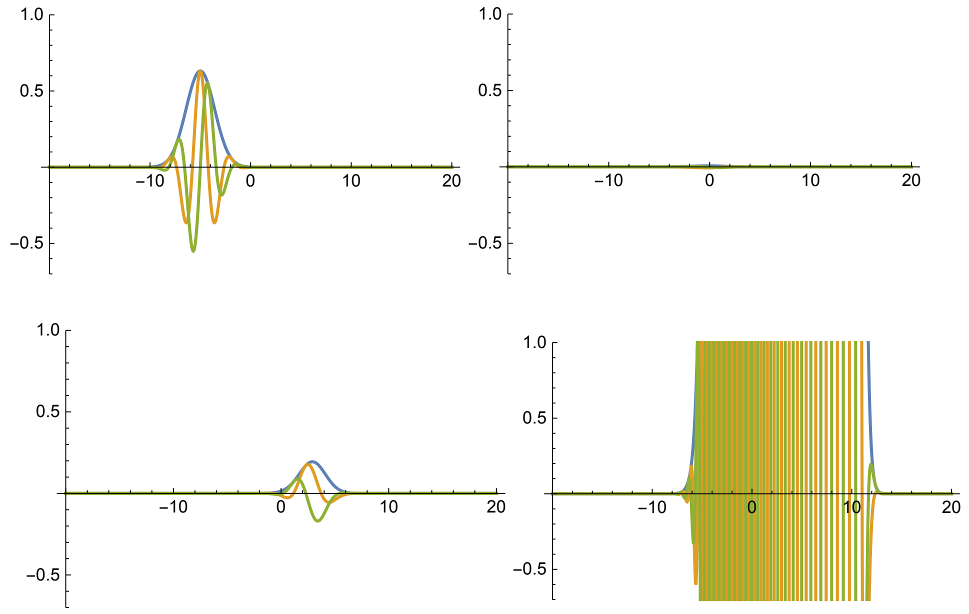
In[216]:=
```
Clear[complexCNRe, complexCNIm, complexCN, complexPotential];
Block[{caseChoice, periodicChoice, schemeChoice},
  caseChoice = 7;
  periodicChoice = 2;
   schemeChoice = 2;
  Do[
   Evaluate[Symbol["complexCN" <> {"Re", "Im"}[[i]]]] = Import[
      path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
       scheme[[schemeChoice]] <> "_" <> numberType[[i]] <> ".csv", "CSV"] ;,
   {i, 2}];
  complexPotential = Flatten[Import[
     path <> "/Runs/" <> cases[[caseChoice]] <> "/" <> periodicType[[periodicChoice]] <>
      scheme[[schemeChoice]] <> "_" <> "Potential" <> ".csv", "CSV"]];
  complexCN = Sqrt[complexCNRe^2 + complexCNIm^2];
 ];
Manipulate[ListPlot[
  {complexCN[[t, All]], complexCNRe[[t, All]], complexCNIm[[t, All]], complexPotential},
  PlotRange → {-.7, 1}, PlotLegends → {"|Ψ|^2", "Re(Ψ)", "Im(Ψ)"},
  ImageSize → 500, Joined → True, DataRange → {-20, 20}, AxesOrigin → {-20, 0}],
 {t, 1, Length[complexCNRe[[All, 1]]], 1}]
```

Out[218]=

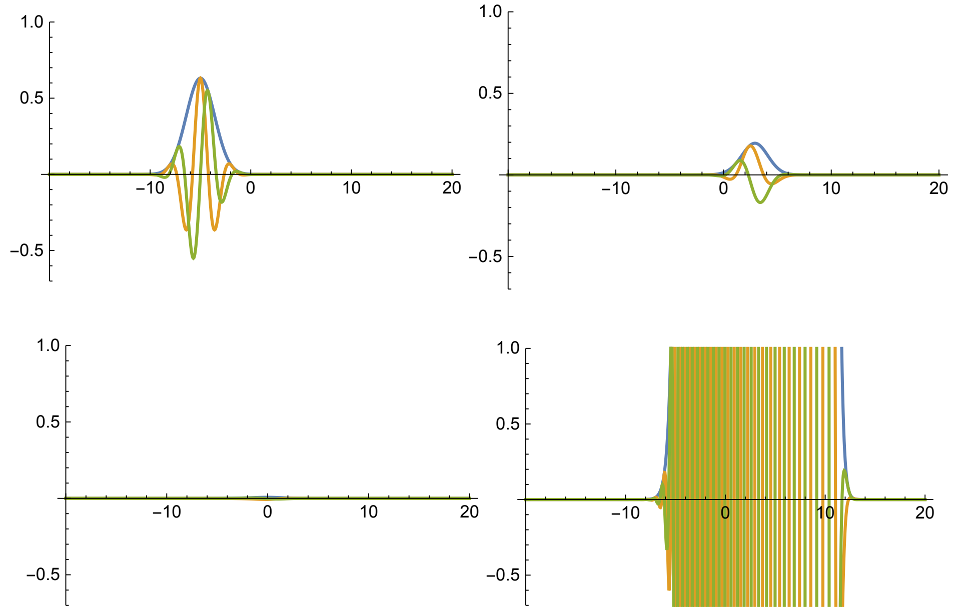


This seems to suffer from the similar problem as the last potential, except first it collapses. See timesteps 1, 10, 20, and 50 below (left to right order).

In[219]:=

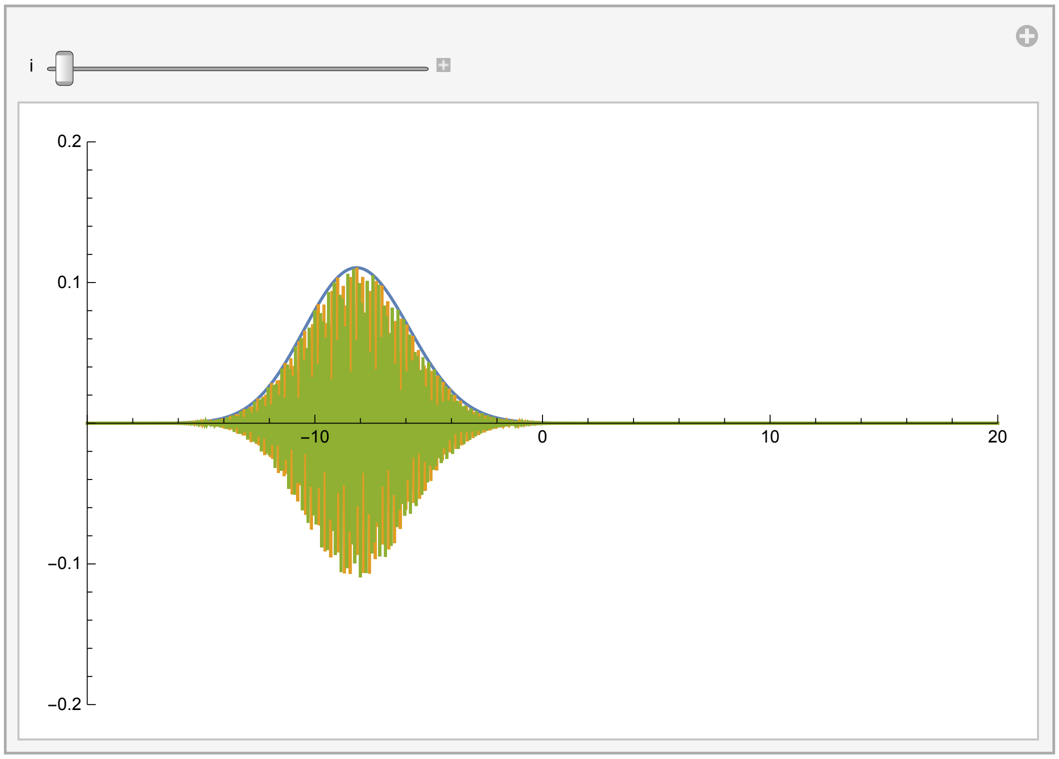


Out[219]=



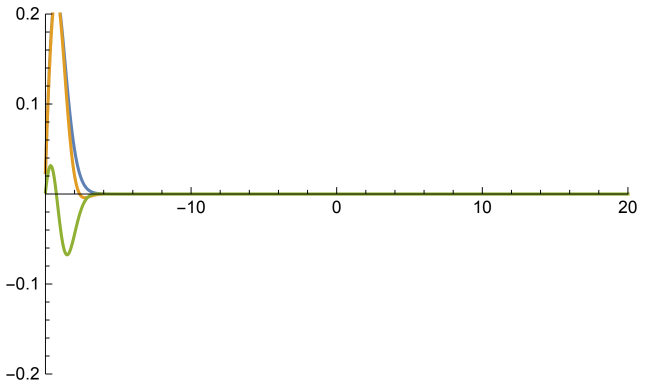
```
In[220]:= Clear[complexEVectorsRe, complexEVectorsIm];
          Block[{caseChoice, periodicChoice, schemeChoice},
            caseChoice = 7;
            periodicChoice = 2;
             schemeChoice = 2;
            complexEValuesRE = Import[path <> "/Runs/" <> cases[[caseChoice]] <>
                "/" <> periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <>
                "_" <> numberType[[1]] <> "_" <> eigenState[[1]] <> ".csv", "CSV"];
            complexEValuesIm = Import[path <> "/Runs/" <> cases[[caseChoice]] <>
                "/" <> periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <>
                "_" <> numberType[[2]] <> "_" <> eigenState[[1]] <> ".csv", "CSV"];
            Do[
             Evaluate[Symbol["complexEVectors" <> {"Re", "Im"}[[i]]]] =
                Import[path <> "/Runs/" <> cases[[caseChoice]] <> "/" <>
                  periodicType[[periodicChoice]] <> scheme[[schemeChoice]] <> "_" <>
                  numberType[[i]] <> "_" <> eigenState[[2]] <> ".csv", "CSV"];,
              {i, 2}];
            complexEVectors = Sqrt[complexEVectorsRe^2 + complexEVectorsIm^2];
           ];

In[222]:= Manipulate[ListPlot[{complexEVectors[[All, i]], complexEVectorsRe[[All, i]],
              complexEVectorsIm[[All, i]], complexPotential / 25}, Joined → True,
            DataRange → {-20, 20}, AxesOrigin → {-20, 0}, ImageSize → 500,
            PlotRange → {{-20, 20}, {-.2, .2}}], {i, 1, Length[complexEVectors[[All, 1]]], 1}]
```

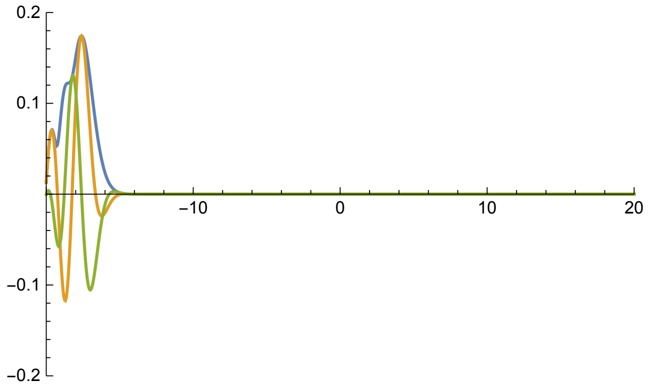
Out[222]=

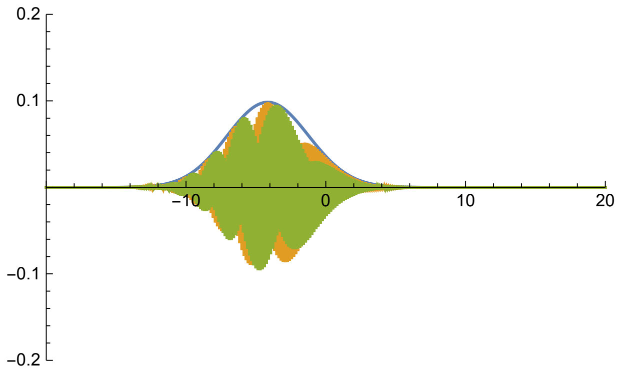


These look a lot like the eigenvectors of our last potential. Check out 220, 222, and 226 (left to right)

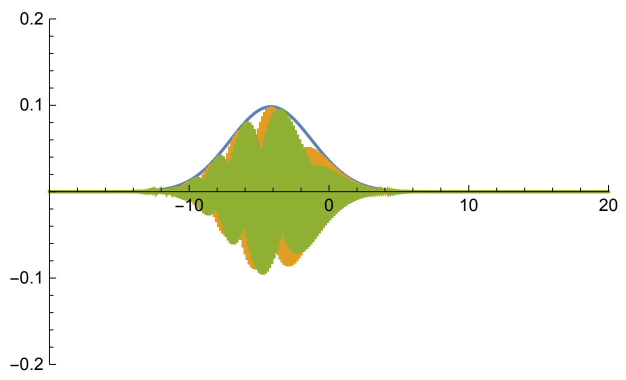
In[223]:=

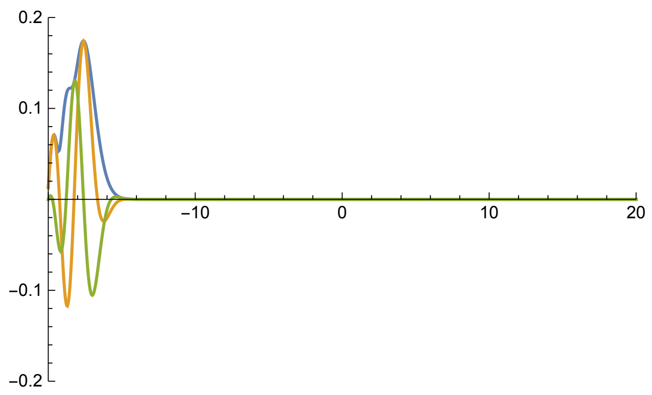


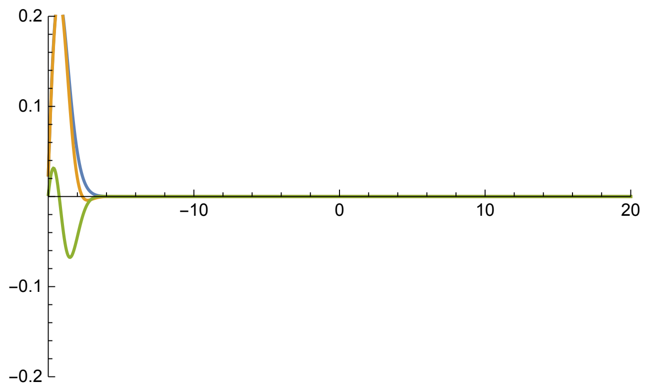




Out[223]=

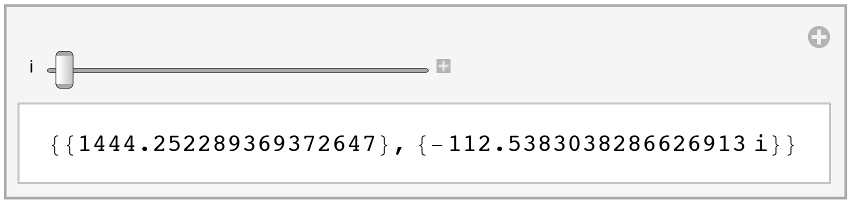






In[224]:=
```
Manipulate[{complexEValuesRE[[i]], complexEValuesIm[[1]] "i"},
  {i, 1, Length[complexEValuesRE], 1}]
```

Out[224]=



```
{{1444.252289369372647}, {-112.5383038286626913 i}}
```

This has suffered the same fascinating fate as the previous potential--we have the same imaginary part of each eigenvalue. So perhaps it’s some quirk of our scheme.

Conclusion

Broadly, our results using the Crank-Nicolson scheme predict the behavior that we would expect to see for wave packets. For the square well and harmonic oscilator potentials, the Crank-Nicolson scheme largely preserves the eigenstates over time. For the barrier, the results are qualitatively consistent with analytical results, though they are off significantly when it comes to the exact numbers. This may be due to the vary narrow barrier, which cases more sensitivity to rounding errors (both in our analytical calculation and in the numerical approximations).

Intriguingly, initial wave packets incident on the complex potentials seem to explode. We don't have a good explanation for this. We also have a baffling constant imaginary term in the eigenvalues for each of these which is puzzling.