```
In[•]:= SetDirectory[NotebookDirectory[]];
        Import["init.wl"];
```

Time-Dependent Dynamics

In this example, we examine how to propagate a time-dependent solution to the Schrodinger Equation and use the evolution to compute the spectrum of the potential from the wavefunction autocorrelation function.

We begin by briefly reviewing time-dependent quantum mechanics, the solutions, $\psi(x, t)$, to the time-dependent Schrodinger equation,

$$-i\,\hbar\,\partial_t\;\psi[x,t] = \frac{-\hbar^2}{2\,m}\,\partial_{\{x,2\}}\;\psi[x,t] + V[x]\,\psi[x,t]$$

This we can solve formally as,

$$\psi[x,t] = \int \left\langle x \mid e^{-i\,H\,t/\hbar} \mid y \right\rangle \psi[y,0]\,dy$$

where H is the Hamiltonian operator, which we take to be time-independent. Since the kinetic energy operator and the potential operator do not commute, we cannot simply break the propagator (in brackets) into separate kinetic energy and potential energy contributions.

What we can do, however, is use the semi-group property of the propagator to write,

$$U(t_f, t_i) = U(\delta t_n)\,U(\delta t_{n-1})\ldots U(\delta t_o) = \mathcal{T}\prod_{i=1}^{n} U[\delta t_n]$$

where $\mathcal{T}$ is the time-ordering operator and $\delta\,t_n$is a short time interval $\delta\,t_n = t_n - t_{n-1}$ and,

$$U(\delta\,t) = \left\langle x \mid e^{i\,H\,\delta\,t/\hbar} \mid x' \right\rangle$$

Since $\hat{H} = \hat{T} + \hat{V}$ we can use Baker-Campbell-Hausdorff relation to break the exponential into a product of three-exponentials.

This is the Trotter Product and is accurate to third order in $\delta\,t_i$,

$$U(\delta\,t_i) = e^{-i\,T\,t/2}\,e^{-i\,V\,t}\,e^{-i\,T\,t/2} + O\left[\delta\,t^3\right]$$

To use this, we need to insert complete sets of states between each term,

$$U(\delta\,t_i) = \int \left\langle x \mid e^{-i\,T\,t/2} \mid x' \right\rangle \left\langle x' \mid e^{-i\,V\,t} \mid x' \right\rangle \left\langle x' \mid e^{-i\,T\,t/2} \mid x'' \right\rangle dx'$$

where we have taken advantage of the fact that V is simply a function of x. Now, we insert complete sets of kinetic energy states $T\left|n\right\rangle = \epsilon_n\left|n\right\rangle$ on either side of the kinetic energy terms,

$$U(\delta\,t_i) = \sum_{n=1}^{N}\sum_{m=1}^{N}\int \langle x \mid n\rangle\, e^{-i\,\epsilon_n\,t/2}\,\langle n \mid x'\rangle\, e^{-i\,V(x')\,t}\,\langle x' \mid m\rangle\, e^{-i\,\epsilon_m\,t/2}\,\langle m \mid x''\rangle\, dx'$$

Lastly, we approximate the integral over x$'$ via Gaussian quadrature by using the DVR points and their associated weights,

$$U_{kj}(\delta\,t_i) = \sum_{n=1}^{N} T_{nk}\,e^{-i\,\epsilon_n\,\delta\,t/2}\sum_{i=i}^{N} T_{ni}\,e^{-i\,V_i\,\delta\,t}\sum_{m=1}^{N} T_{mi}\,e^{-i\,\epsilon_m\,\delta\,t/2}\,T_{mj}$$

Computationally, it is a bit more effective to break this into parts.

- First, compute the contribution from the kinetic energy part.

$F = e^{-i \epsilon \delta t/2}$

where *F* is a diagonal matrix.

- Then, we hook everything together,

$U_{jk} = \left(U^K.\exp(-i\,\delta\,t\,V).U^K\right)_{jk}$

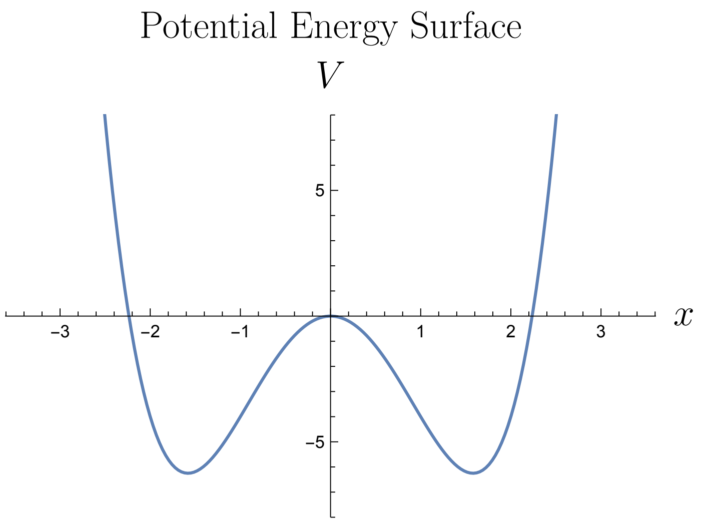
where, again, $e^{-i V \delta t}$, is a diagonal matrix computed by evaluating the potential over the DVR points, then exponentiating.
After putting everything together we have an approximation for the quantum propagator over a short time-step.

```
In[]:= tcheby[npts_Integer, xmin_, xmax_] := Module[
          {del, pts, t, fbrke, w},
          del = xmax - xmin;
          pts = N[Table[(i del)/(npts + 1) + xmin, {i, npts}]];
          t = N[Table[Sqrt[2./(npts + 1)] Sin[((i j) π)/(npts + 1)], {i, npts}, {j, npts}]];
          fbrke = N[Table[((i π)/del)^2, {i, npts}]];
          w = N[Table[del/(npts + 1), {i, npts}]];
          Return[{pts, t, fbrke, w}]];
       dv2fb[dvr_, t_] := t.dvr.Transpose[t];
       fb2dv[fbr_, t_] := Transpose[t].fbr.t;
```

```
In[•]:= npts = 200;
xmin = -3.5;
xmax = 3.5;
xo = 1.5;
β = 0.5;
ℏ = 1;
δt = 0.02;
m = 1;

{pts, t, fbrke, w} = tcheby[npts, xmin, xmax];
fbrke = fbrke * ℏ^2/(2 m);

v[x_] := -5 x^2 + x^4;
vdvr = Thread@v[pts];
hdvr = (fb2dv[DiagonalMatrix[fbrke], t] + DiagonalMatrix[vdvr]);

expV = (Exp[-ⅈ v[pts] δt / ℏ]) // Thread;
expK = (Exp[-ⅈ / ℏ fbrke δt / 2]) // Thread;
tTrans = Transpose[t];

uk = t.DiagonalMatrix[expK].tTrans;
u = uk.DiagonalMatrix[expV].uk;
```

```
In[•]:= plotV = ListPlot[
   Transpose[{pts, vdvr}], PlotRange → {-8, 8}, Joined → True,
   AxesLabel → {MaTeX["x", FontSize → 20], MaTeX["V", FontSize → 20]},
   PlotLabel → MaTeX["\\text{Potential Energy Surface}", FontSize → 20]
  ]
```

Out[•]=



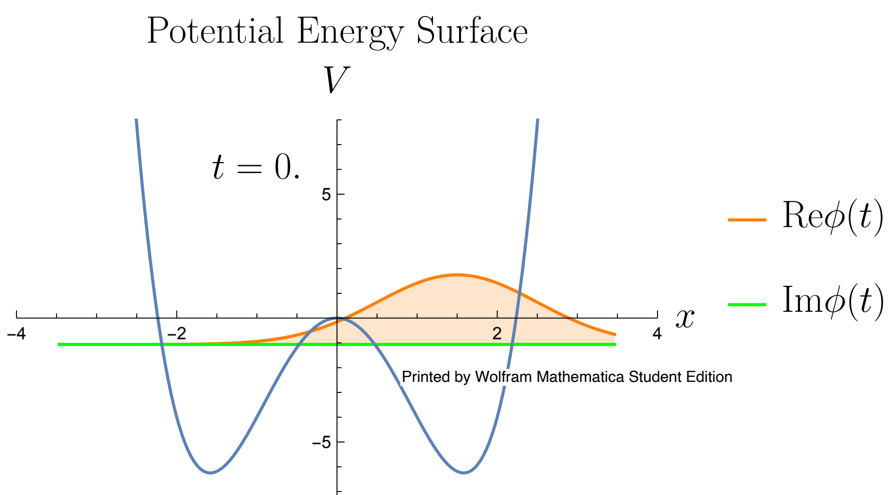
Now, we evolve the initial state, ϕ_o, for a few time-steps and compute the overlap of the time-evolve state with the initial state.
What we're interested in computing is the evolution of the overlap of the wavefunction at time *t* with the initial wavefunction.

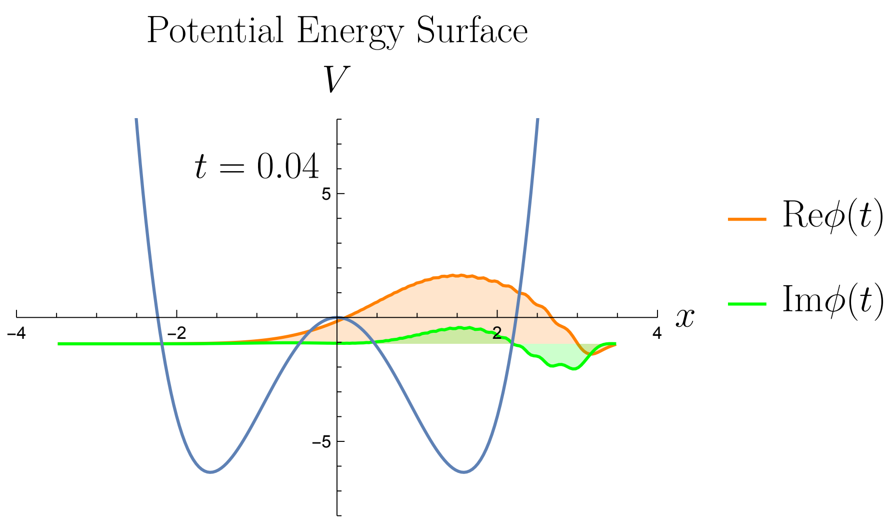
$$c(t) = \langle\psi(0)\mid\psi(t)\rangle = \sum_{n=1}^{N}\langle\psi(0)\mid n\rangle\, e^{i\,\omega_n t}\langle n\mid\psi(0)\rangle$$

Thus, taking the Fourier transform of $c(t)$ produces a spectrum in ω where the peaks give the location of the energy eigenvalues and the intensities give the projection $|\langle\psi(0)\mid n\rangle|^2$.

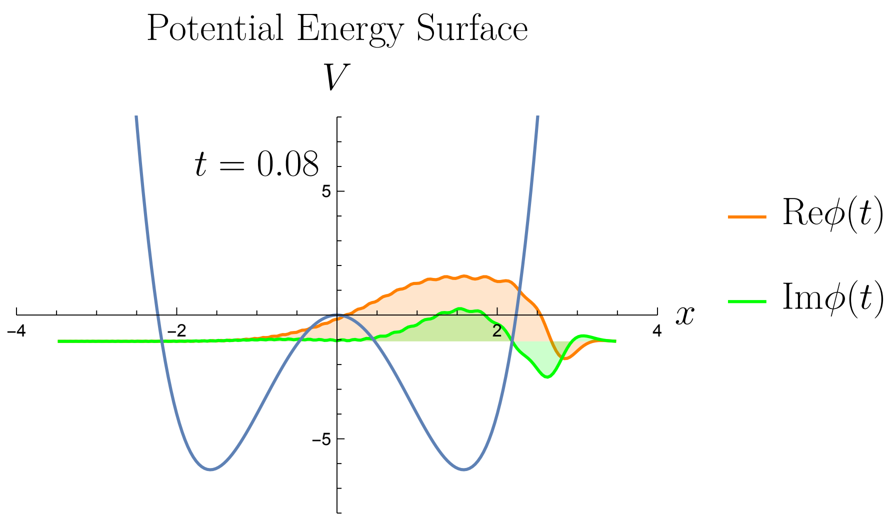
```
In[•]:= ψo[x_] := (β/π)^(1/4) e^(-β (x-xo)^2); ϕo = Thread[ψo[pts]];
amp = 20.0;
norm = Sqrt[ϕo.ϕo]; ϕo = ϕo/norm; ψt = {ϕo}; ct = {1};
ene = ϕo.hdvr.ϕo;
evolve = {{ListPlot[Transpose[{pts, amp * ϕo + ene}], Joined → True, PlotRange →
      {{-4, 4}, {-8, 8}}, PlotStyle → Orange, DisplayFunction → Identity,
     Filling → ene, PlotLegends → {MaTeX["\\text{Re}\\phi(t)", FontSize → 20]},
     Epilog → Inset[MaTeX["t = " <> ToString[0.00], FontSize → 20], {-1.0, 6}],
     AxesLabel → {MaTeX["x", FontSize → 20], MaTeX["V", FontSize → 20]},
     PlotLabel → MaTeX["\\text{Potential Energy Surface}", FontSize → 20]],
    ListPlot[Transpose[{pts, amp * Im[ϕo] + ene}], Joined → True,
     PlotRange → {{-4, 4}, {-8, 8}}, PlotStyle → Green,
     DisplayFunction → Identity, Filling → ene,
     PlotLegends → {MaTeX["\\text{Im}\\phi(t)", FontSize → 20]}]}};
nsteps = 20;
 ϕt = ϕo;
 Do[ϕt = u.ϕt;
  c = ϕo.ϕt;
  ct = Append[ct, c];
  If[Mod[n, 1] == 0, ψt = Append[ψt, ϕt];
   rePlot = ListPlot[Transpose[{pts, amp * Re[ϕt] + ene}], Joined → True, PlotRange →
      {{-4, 4}, {-8, 8}}, PlotStyle → Orange, DisplayFunction → Identity,
     Filling → ene, PlotLegends → {MaTeX["\\text{Re}\\phi(t)", FontSize → 20]},
     Epilog → Inset[MaTeX["t = " <> ToString[n δt], FontSize → 20], {-1.0, 6}],
     AxesLabel → {MaTeX["x", FontSize → 20], MaTeX["V", FontSize → 20]},
     PlotLabel → MaTeX["\\text{Potential Energy Surface}", FontSize → 20]];
   imPlot = ListPlot[Transpose[{pts, amp * Im[ϕt] + ene}], Joined → True,
     PlotRange → {{-4, 4}, {-8, 8}}, PlotStyle → Green, DisplayFunction → Identity,
     Filling → ene, PlotLegends → {MaTeX["\\text{Im}\\phi(t)", FontSize → 20]}];
   evolve = Append[evolve, {rePlot, imPlot}]], {n, nsteps}]; ct1 = ct;
```

```
In[•]:= TableForm@Table[e = evolve[[i]]; Show[e[[1]], e[[2]], plotV,
    PlotRange → {{-4, 4}, {-8, 8}}, ImageSize → Medium], {i, 1, 20, 2}]
Out[•]//TableForm=
```

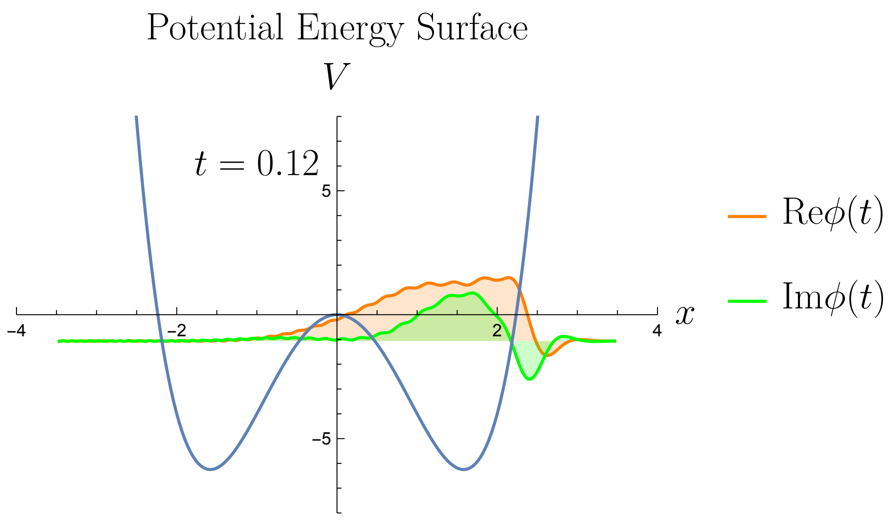




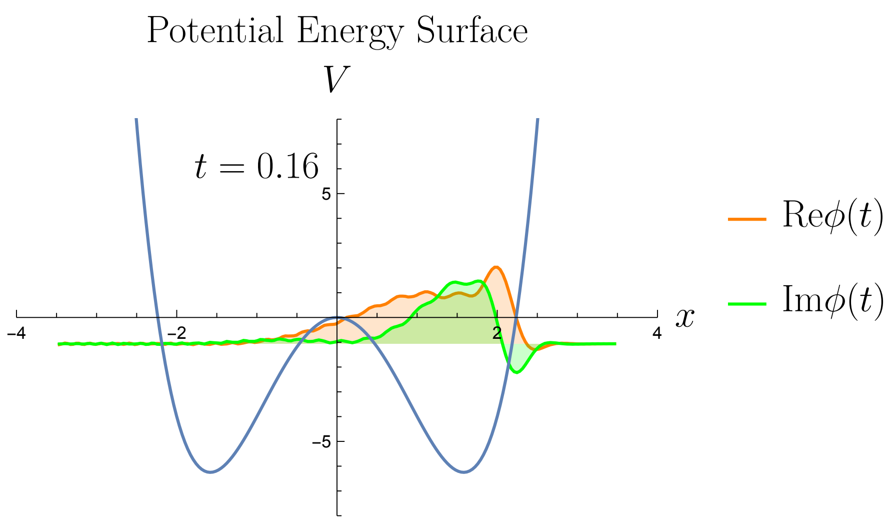
Potential Energy Surface
V
$t = 0.04$
5
−5
−4
−2
2
4
x
$\mathrm{Re}\phi(t)$
$\mathrm{Im}\phi(t)$

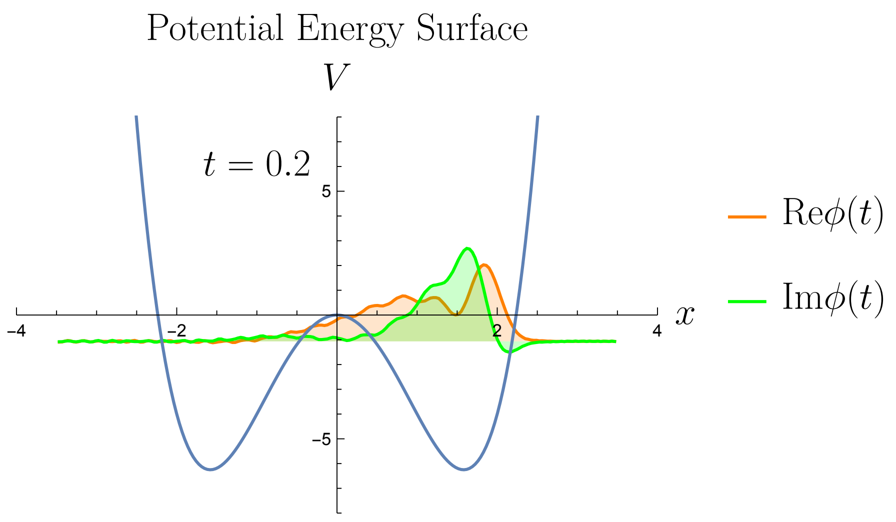


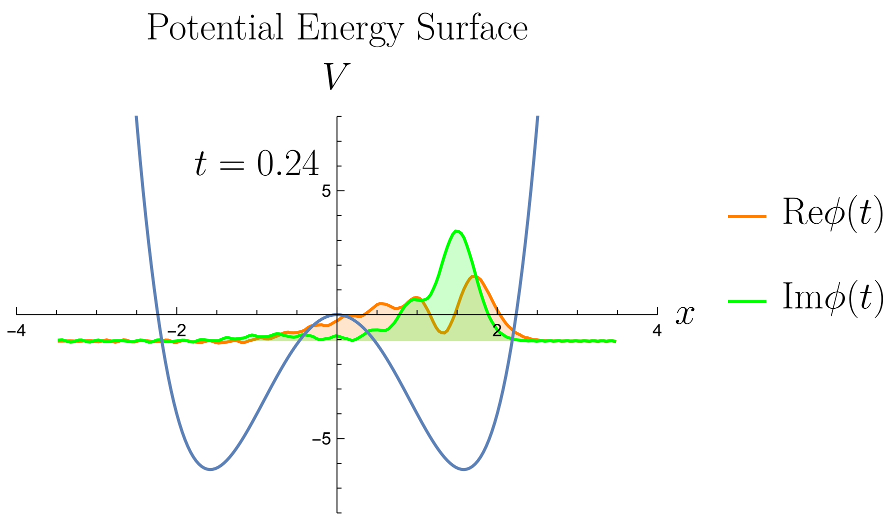
Potential Energy Surface
V
$t = 0.08$
5
−5
−4
−2
2
4
x
$\mathrm{Re}\phi(t)$
$\mathrm{Im}\phi(t)$

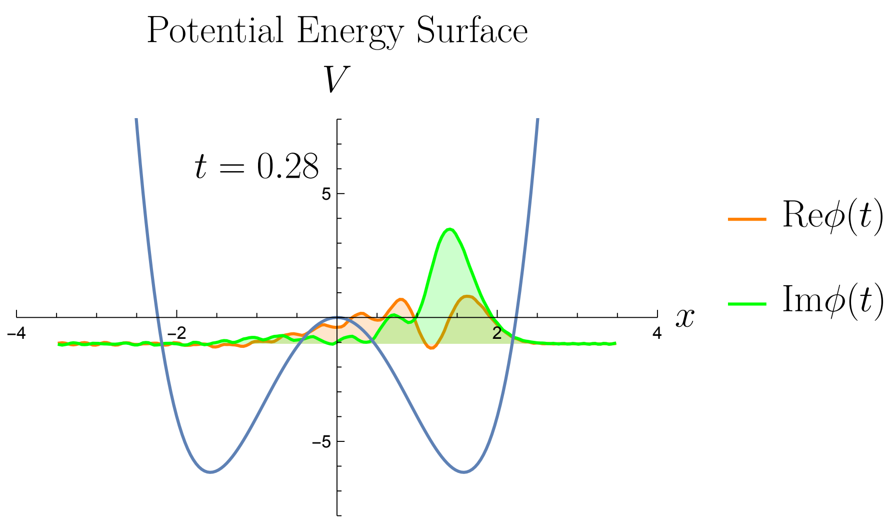


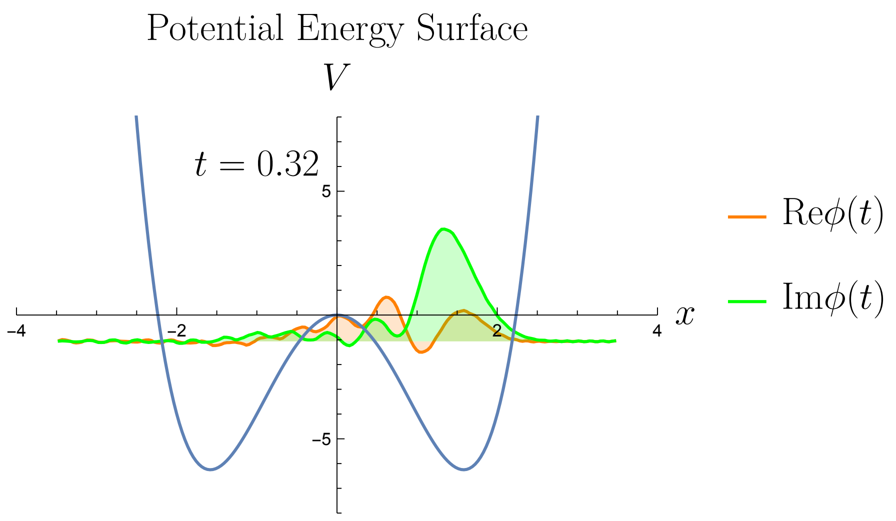
Potential Energy Surface
V
$t = 0.12$
5
−5
−4
−2
2
4
x
$\mathrm{Re}\phi(t)$
$\mathrm{Im}\phi(t)$

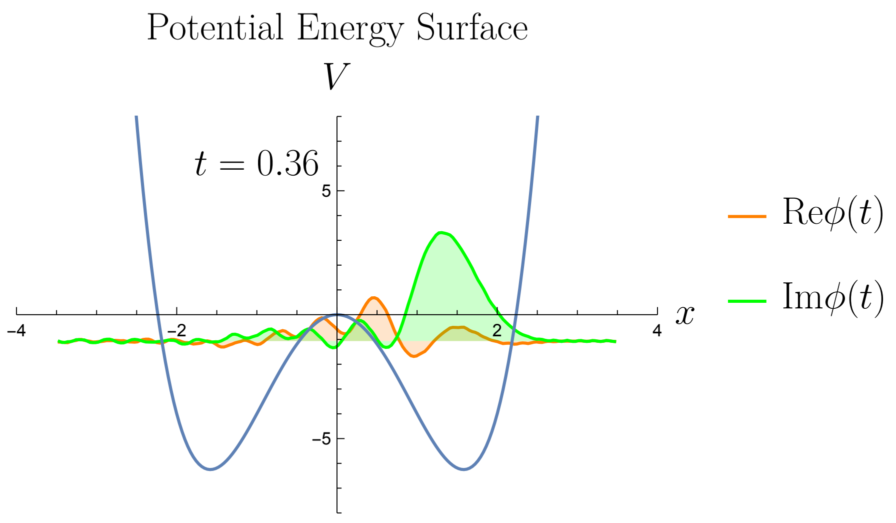

Potential Energy Surface
V
$t = 0.16$
5
−4
−2
2
4
x
−5
$\mathrm{Re}\phi(t)$
$\mathrm{Im}\phi(t)$


Potential Energy Surface
V
$t = 0.2$
5
−4
−2
2
4
x
−5
$\mathrm{Re}\phi(t)$
$\mathrm{Im}\phi(t)$


Potential Energy Surface
V
$t = 0.24$
5
−4
−2
2
4
x
−5
$\mathrm{Re}\phi(t)$
$\mathrm{Im}\phi(t)$


Potential Energy Surface
V
$t = 0.28$
5
−4
−2
2
4
x
−5
$\text{Re}\phi(t)$
$\text{Im}\phi(t)$


Potential Energy Surface
V
$t = 0.32$
5
−4
−2
2
4
x
−5
$\text{Re}\phi(t)$
$\text{Im}\phi(t)$


Potential Energy Surface
V
$t = 0.36$
5
−4
−2
2
4
x
−5
$\text{Re}\phi(t)$
$\text{Im}\phi(t)$