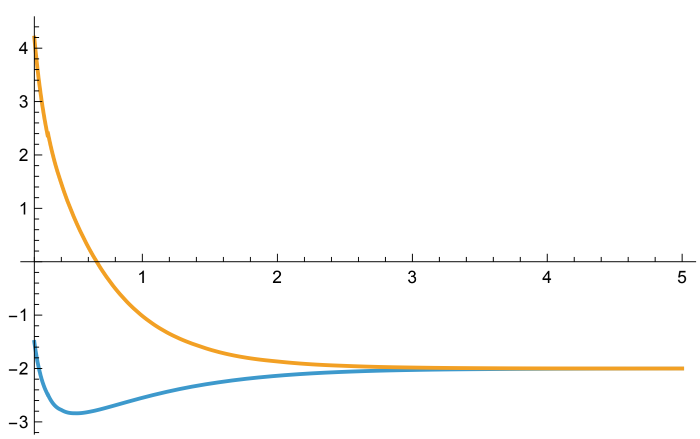
```
In[ ]:= (* this routine needs to run Vpots.nb *)

In[50]:= geradeData =
   Import["/Users/zelimir/Work/Physics-Thesis/thesis-2/ZelimirOverleaf/
      MathematicaCodeFinal/gerade1sV3.mx"];
  geradePotential = Table[{geradeData[[i, 1]], geradeData[[i, 3]]}, {i, 1, 27}];
  ungeradeData =
   Import["/Users/zelimir/Work/Physics-Thesis/thesis-2/ZelimirOverleaf/
      MathematicaCodeFinal/ungerade1sV3.mx"];
  ungeradePotential = Table[{ungeradeData[[i, 1]], ungeradeData[[i, 3]]}, {i, 1, 27}];
  long[R_] := -2.0 - 0.082 / R^4;
  geradefun = Interpolation[geradePotential];
  ungeradefun = Interpolation[ungeradePotential];
  (* Fit to a model *)
  gdata = Drop[geradePotential, {5, 27}];
  udata = Drop[ungeradePotential, {5, 27}];
  model[R_, C1_, C2_, C3_] := C1 Exp[-C2 R] / R + C3;
  gnlm = NonlinearModelFit[gdata, model[R, C1, C2, C3], {C1, C2, C3}, R];
  unlm = NonlinearModelFit[udata, model[R, C1, C2, C3], {C1, C2, C3}, R];
  gshort[R_] := model[R, C1, C2, C3] /. gnlm["BestFitParameters"];
  ushort[R_] := model[R, C1, C2, C3] /. unlm["BestFitParameters"];
  gpot[R_] = Piecewise[
    {{gshort[R], R ≤ 0.3}, {geradefun[R], 0.3 < R < 13.0}, {long[R], R ≥ 13.0}}];
  upot[R_] = Piecewise[
    {{ushort[R], R ≤ 0.3}, {ungeradefun[R], 0.3 < R < 13.0}, {long[R], R ≥ 13.0}}];
  Plot[{gpot[R], upot[R]}, {R, 0.2, 5}, PlotRange → All]

  (*e=0.1/27.21;*)
  μ = 1866.0 / 2.0;
  (*k=√(2 μ e);*)
  kB = 8.617^-5; (* Boltzman, eV/K *)
  (*λ=2π/k*)

  r0 = 0.01;
  rend = 100;

  computePhaseShifts[m_, e_, r0_, rend_] := Module[{k, eqG, eqU, solsG, solsU, fg, fu},
    If[m == 40, Print["e=", e]];
    k = √(2 μ e);
    equG := yg''[r] + yg'[r] / r - 2 μ (gpot[r] + 2) yg[r] - m^2 / r^2 yg[r] + 2 μ e yg[r] ;
    equU := yu''[r] + yu'[r] / r - 2 μ (upot[r] + 2) yu[r] - m^2 / r^2 yu[r] + 2 μ e yu[r];
    solsG = Flatten[NDSolve[
       {equG == 0, yg[r0] == 0, yg'[r0] == 1}, yg, {r, r0, rend}, MaxSteps → 200000]];
```

```
    solsU = Flatten[NDSolve[
        {equU == 0, yu[r0] == 0, yu'[r0] == 1}, yu, {r, r0, rend}, MaxSteps → 200 000]];
    fg[r_] = yg[r] /. solsG;
    fu[r_] = yu[r] /. solsU;
    If[(e > 0.002 && e < 0.004) && (m == 40 || m == 20 || m == 5 || m == 0),
     Print["k=", k];
     Print[Plot[fg[r], {r, r0, rend}, PlotLabel → StringJoin["m=",
          ToString[m], ", e=", ToString[e], "eV"], AxesLabel → "R"]]];
    (*Print[Plot[fg[r]-fu[r],{r,r0,rend}]]*)
    ylogG[r_] := fg'[r] / fg[r];
    ylogU[r_] := fu'[r] / fu[r];
    (*Print[Plot[ylogG[r],{r,r0,rend}]];
    Print[Plot[ylogU[r],{r,r0,rend}]];*)
    sl[n_, r_] := BesselJ[n, k r];
    cl[n_, r_] := HankelH1[n, k r];
    slp[n_, r_] := Derivative[0, 1][sl][n, r];
    clp[n_, r_] := Derivative[0, 1][cl][n, r];
    bG[n_, r_] :=
     (-I) ^ (Abs[n]) (slp[n, r] - ylogG[r] × sl[n, r]) / (clp[n, r] - ylogG[r] × cl[n, r]);
    bU[n_, r_] :=
     (-I) ^ (Abs[n]) (slp[n, r] - ylogU[r] × sl[n, r]) / (clp[n, r] - ylogU[r] × cl[n, r]);
    (N[Abs[bG[m, rend] - bU[m, rend]]]) ^2
    (*Sin[N[Abs[bG[m,rend]-bU[m,rend]]]]^2*)
   ];
(*computePhaseShifts[10,0.003,0.1,100]*)
```

Out[66]=



In[72]:=

```
(* e = collision energy in eV, r0, rend - R range *)
computeCrossSect[eV_, r0_, rend_] := Module[{allBs, eH, k, crossSection},
   eH = eV / 27.21;  (* energy Hartree *)
   k = Sqrt[2 μ eH];
   (* m - partial waves *)
   allBs = Table[{m, computePhaseShifts[m, eH, r0, rend]}, {m, 0, 40}];
   crossSection = 1/k Total[allBs, {1}][[2]];
   {eV, crossSection}
  ];

crossSection = Table[computeCrossSect[e, r0, rend], {e, 0.01, 0.1, 0.01}];
crossSection // TableForm

crossSectionGraph = Interpolation[crossSection];
```

e=0.000367512

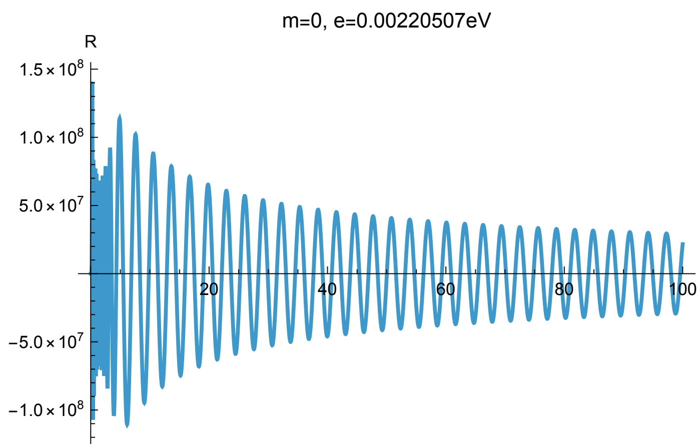
e=0.000735024

e=0.00110254

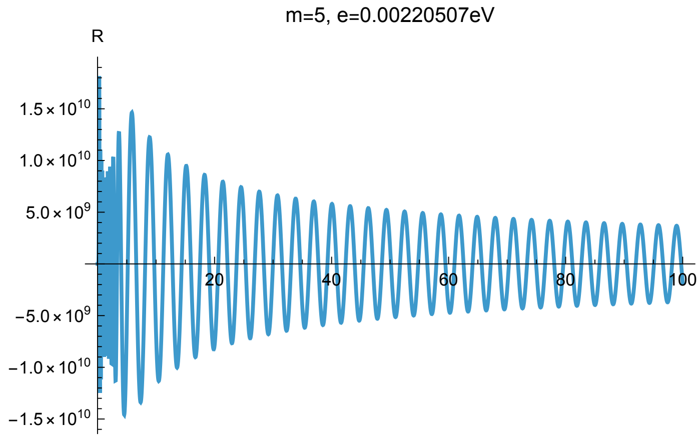
e=0.00147005

e=0.00183756

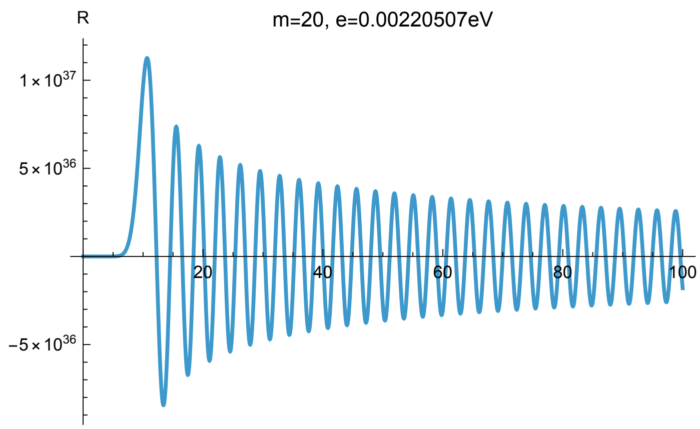
k=2.02846



k=2.02846

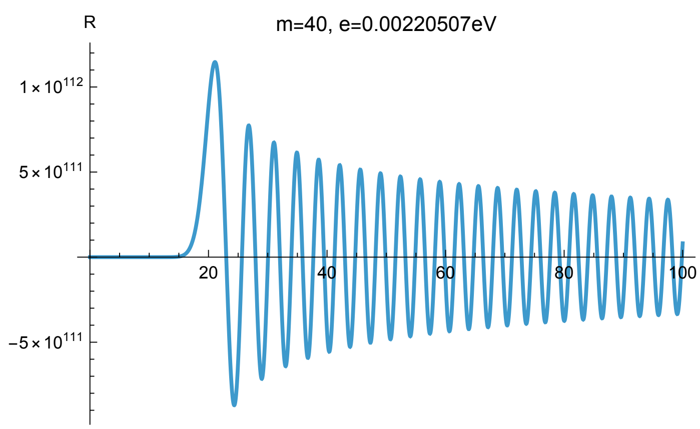


k=2.02846

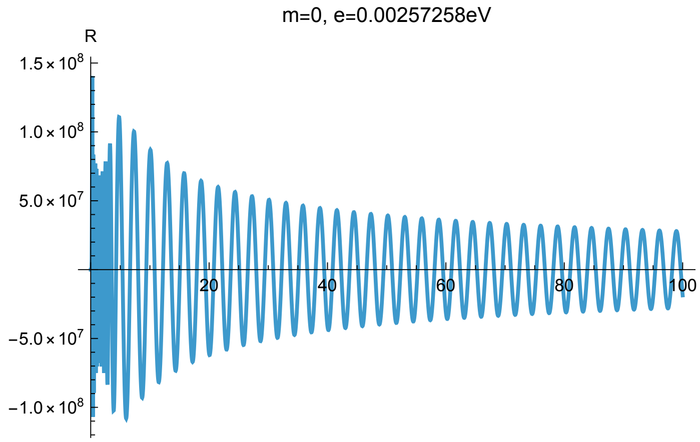


e=0.00220507

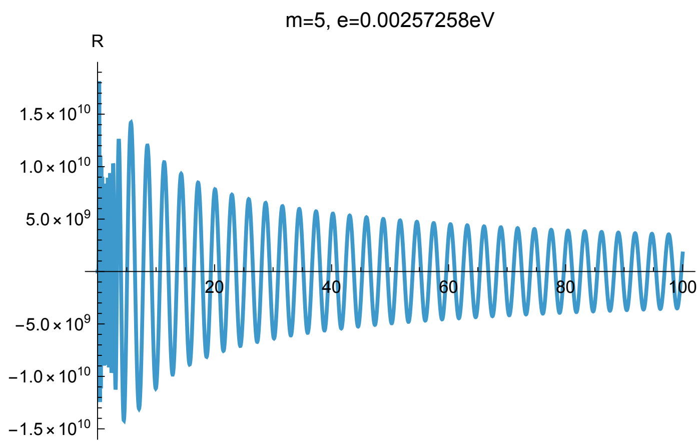
k=2.02846



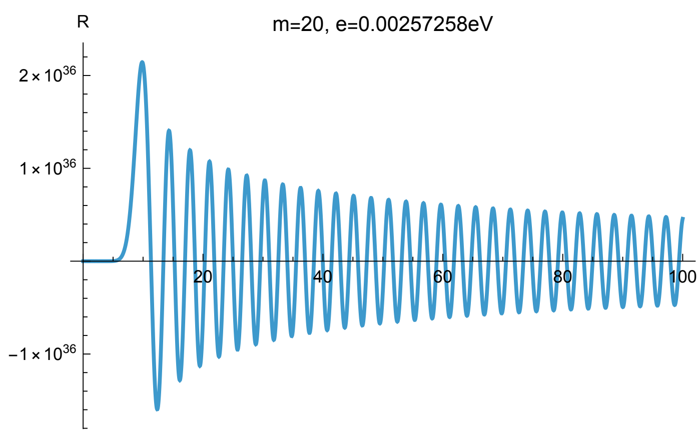
k=2.19099



k=2.19099

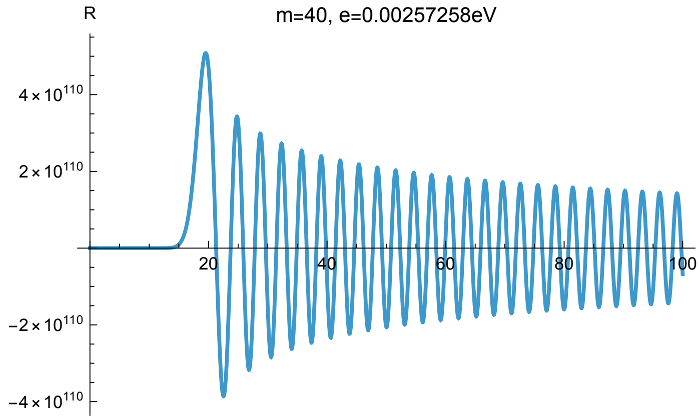


k=2.19099

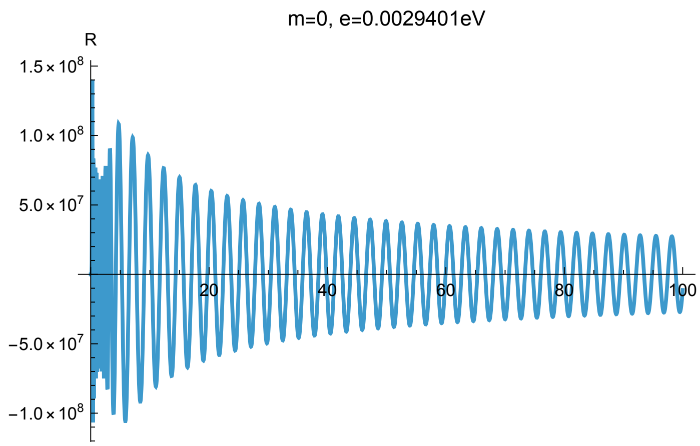


e=0.00257258

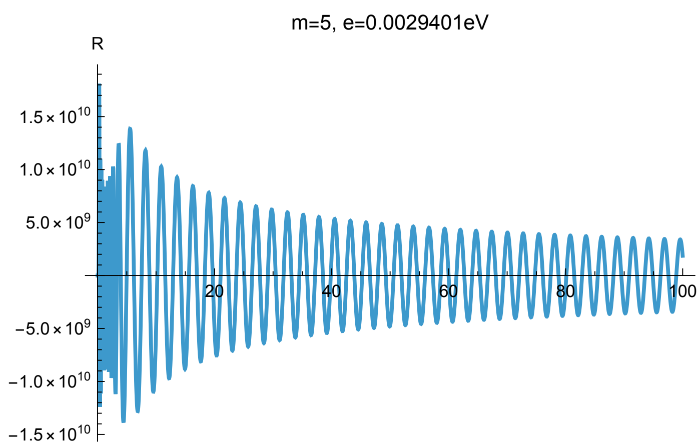
k=2.19099



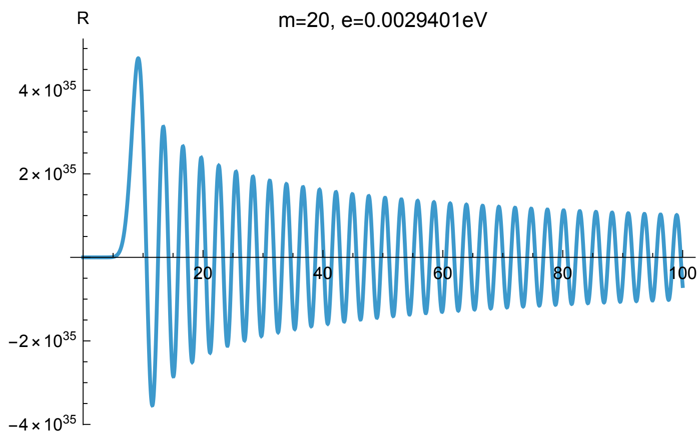
k=2.34227



k=2.34227

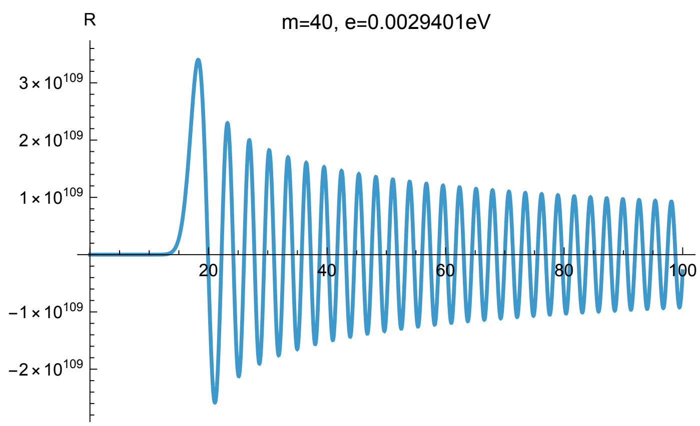


k=2.34227

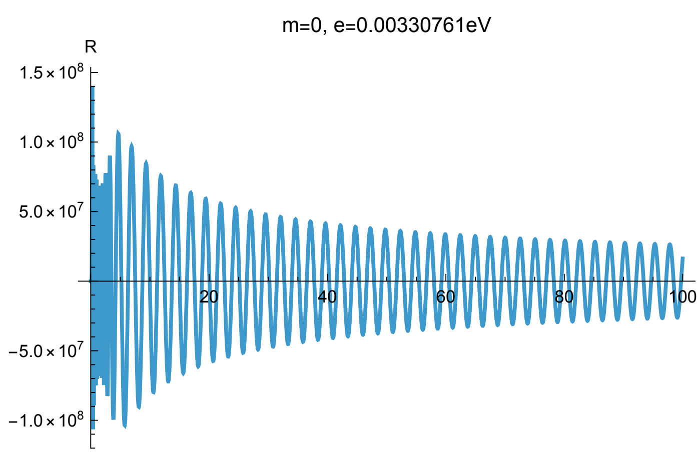


e=0.0029401

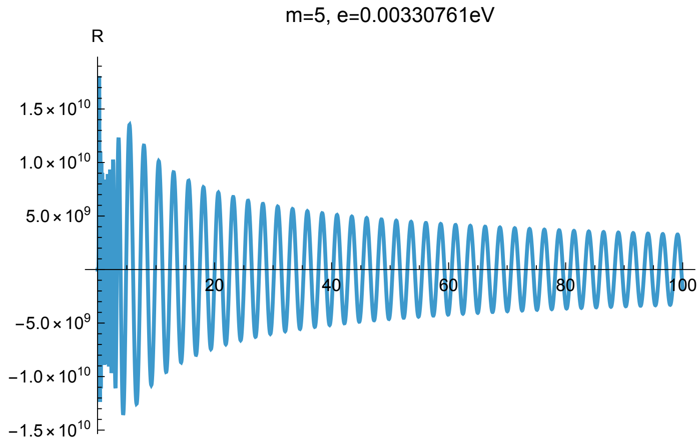
k=2.34227



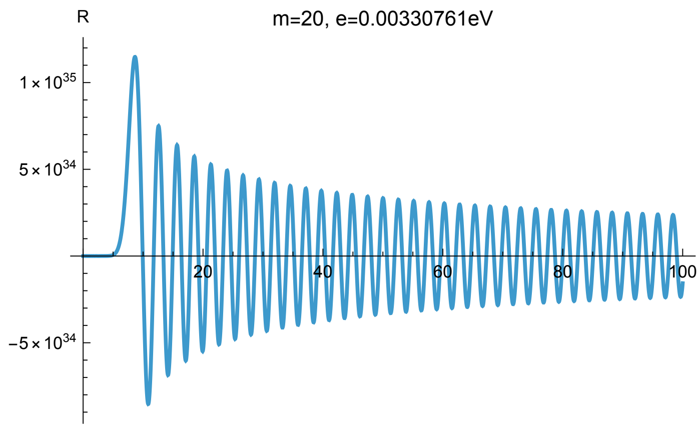
k=2.48435



k=2.48435

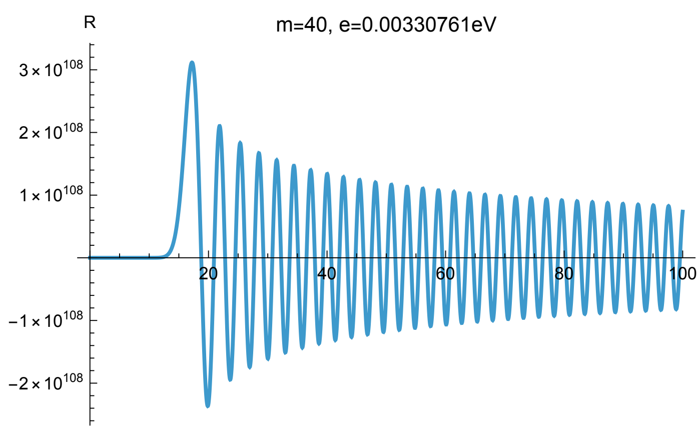


k=2.48435

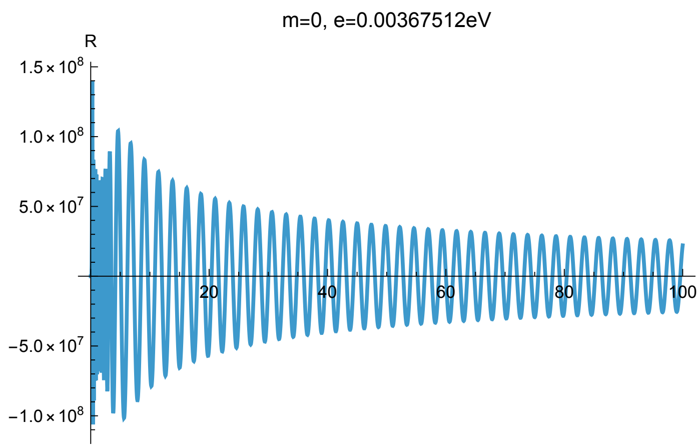


e=0.00330761

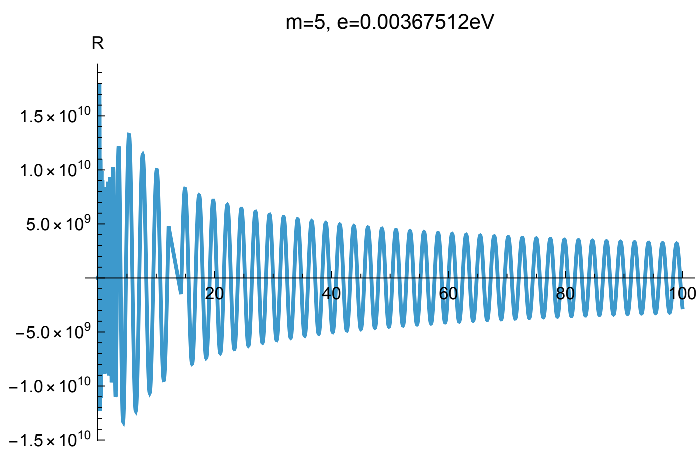
k=2.48435



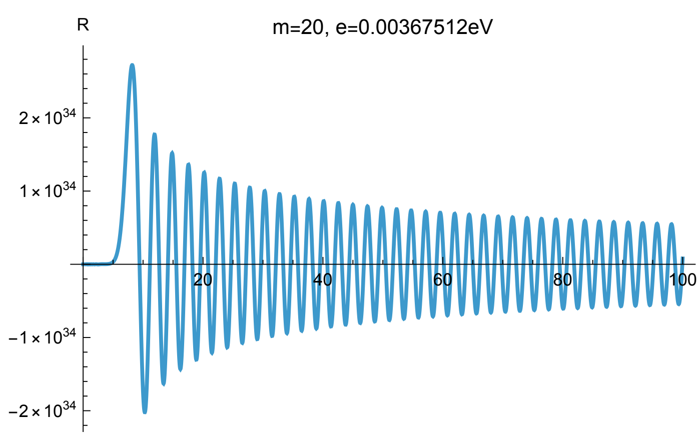
k=2.61873



k=2.61873

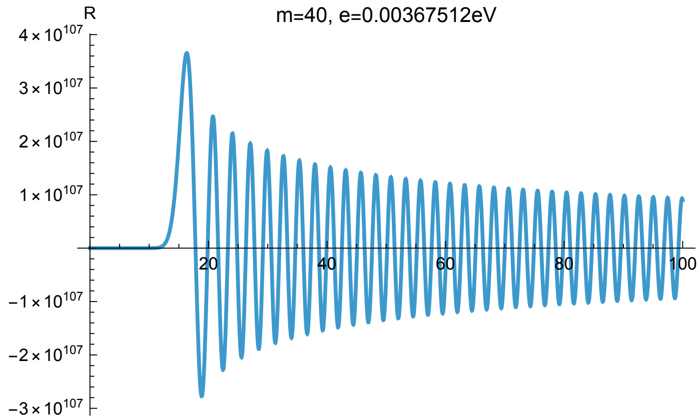


k=2.61873



e=0.00367512

k=2.61873



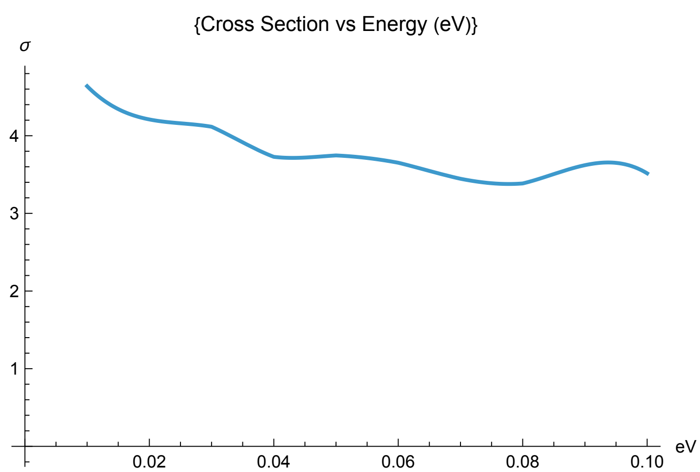
Out[74]//TableForm=

0.01	4.63768
0.02	4.20954
0.03	4.11535
0.04	3.72823
0.05	3.74606
0.06	3.652
0.07	3.44569
0.08	3.38539
0.09	3.62067
0.1	3.51548

In[78]:=

```
Plot[crossSectionGraph[e], {e, 0.01, 0.1}, AxesOrigin → {0, 0},
 AxesLabel → {"eV", "σ"}, PlotLabel → {"Cross Section vs Energy (eV)"}]
```

Out[78]=



Note the convergence of b_(m=40) as a function of R; b_(40) is already small, you need to calculate all b_{m} m<40, before total cross section sum over partial waves converges

The above code demonstrates the convergence of b_{m} (related to a phase shift) defined by Eq. (23) in my notes. This code is for the gerade potentials, do the same for the ungerade potentials, calculated in

Vpots.nb. Use Eq. (38) in my notes to determine CRT cross sections.

You need to calculate b_{m} for many partial waves m, before σ converges for large m (note above m=40, almost converged as |b_{40}|^2 <<1. You may want to re-write this code in python, to speed it up.

When complete, make a plot of the total cross sections, as a function of collision energy.
In order to educate the reader you may want to include additional figures, such as the dependence of the partial wave cross sections, or plot of the wave function as shown above. Your thesis should be comprehensive and offer a narrative to help the, non-specialist reader, understand what and why you are doing.