Nucleon-Nucleon scattering in Exponential potentials and Square Well potential

Rongzheng He, Xingze mao

1 Introduction

For well bound systems, Schrodinger equation can be solved to obtain discrete states, whose wavefunction asymptotically vanish in the long range. For systems near the threshold or nuclear reaction, scattering state are involved, whose boundary conditions include both incoming and outgoing part. Lippmann-Schwinger equation is usually more convenient to describe scattering states.

Cross section are quantities that can be obtained directly in experiments, which is related to the phase shift of scattered particles. Phase shift is an important bridge connecting theoretical calculations and experimental results, through which theory can be validated, prediction can be verified and experimental results can be understood.

Phase shift is related to R-matrix, which can be obtained by solving Lippmann-Schwinger equation. However, principal value of R-matrix can cause divergence issue and matrix manipulation can be burdensome with for large dimension. As an alternative, Variable Phase Approach can obtain phase shift by only integrating a differential equation, but is limited to local potential without tensor interaction.

In this work, details of both approaches are discussed in Sec. (2). Potentials used in the calculation are shown in Sec. (3). Results and discussions are presented in Sec. (4), which is followed by summary in Sec. (5). The code developed based on the two approach resides in the appendix.

2 Method

In this work, two different methods were used to calculate the phase shift of ${}^1S_0$ state of Nucleon-Nucleon scattering. The first method is the R-matrix approach by solving the Lippman-Schwinger equation directly, and details are shown in Sec. (2.1). Variable Phase Approach is another option, which is introduced in Sec. (2.2)

2.1 R-matrix Approach

For scattering states, Lippmann-Schwinger equation is more convenient to use than Schrodinger equation.

$$|\psi^{\pm}\rangle = |\phi\rangle + \frac{1}{E - \frac{q^2}{2m} \pm i\epsilon} V |\psi^{pm}\rangle \,. \tag{1}$$

In the momentum space, potential has the form:

$$V_\ell(k,k') = \int j_\ell(kr) V(r) J_\ell(k'r) r^2 dr, \tag{2}$$

where j_ℓ is the spherical Bessel function. Since only S-wave is considered in this work, only $j_0 = sin(kr)/kr$ is used.

Reaction matrix R is defined in the momentum space as an integral:

$$R(k,k') = V(k,k') + \frac{2}{\pi}\hat{P}\int_0^\infty dq q^2 V(k,q) \frac{1}{k_0^2/m - q^2/m} R(q,k'), \tag{3}$$

In the center of mass frame, reduced mass $m/2$ is used.

When $q = k_0$, Eq. (3) diverges, which is solved by $\hat{P}$ using following equation:

$$\hat{P}\int_0^\infty \frac{f(k)dk}{k^2 - k_0^2} = \int_0^\infty \frac{(f(k) - f(k_0))dk}{k^2 - k_0^2}.$$

The additional term added to the right side is zero:

$$\int_0^\infty \frac{dk}{k^2 - k_0^2} = 0,$$

which does not change the value, but cure the divergence problem.

The R-matrix is reduced to:

$$R(k,k') = V(k,k') + \frac{2}{\pi}\int_0^\infty dq \frac{q^2 V(k,q) R(q,k') - k_0^2 V(k,k_0) R(k_0,k')}{(k_0^2 - q^2)/m}, \tag{4}$$

The importance about R-matrix is its relation with the phase shift:

$$R(k_0,k_0) = -\frac{tan\delta_l}{mk_0} \tag{5}$$

The integral within the Eq. (4) is dealt with Gaussian quadrature method, where x_i and w_i are the i^{th} mesh point and the corresponding weight in the range $[-1:1]$ and $i = 0, 12...N$. A transformation is done to map the point and weight in the range $[0,\infty]$:

$$\begin{aligned} k_i &= tan\frac{\pi}{4}(1 + x_i) \\ \omega_i &= \frac{\pi}{4}\frac{w_i}{cos^2(\frac{\pi}{4}(1+x_i))} \end{aligned} \tag{6}$$

The R-matrix can be then expressed using the mesh points and weights:

$$R(k,k') = V(k,k') + \frac{2}{\pi}\sum_{j=1}^{N}\frac{\omega_j k_j^2 V(k,k_j)R(k_j,k')}{(k_0^2-k_j^2)/m} - \frac{2}{\pi}k_0^2 V(k,k_0)R(k_0,k')\sum_{n=1}^{N}\frac{\omega_n}{(k_0^2-k_n^2)/m} \tag{7}$$

Defining

$$\begin{aligned} u_j &= \frac{2}{\pi}\frac{w_j k_j^2}{(k_0^2-k_j^2)/m} \quad j=1,...,N, \\ u_{N+1} &= -\frac{2}{\pi}\sum_{j=1}^{N}\frac{k_0^2\omega_j}{(k_0^2-k_j^2)/m}, \\ A_{i,j} &= \delta_{i,j} - V(k_i,k_j)u_j, \end{aligned} \tag{8}$$

Eq. (7) can be expressed as $A_{i,l}R_{l,j} = V_{i,j}$, where all A,R and V are $(N+1)\times(N+1)$ dimension matrix. R-matrix can be obtain by applying inverse of A to V: $R = A^{-1}V$. Phase shift can be calculated from $R(k_0,k_0) = R(k_{N+1},k_{N+1})$ using Eq.(5)

2.2 Variable Phase Approach

Variable Phase Approach can be applied only to local potential without tensor forces. Its limitation in generality is made up by simplicity and better control of numerical accuracy. Phase shift is calculated for truncated potential $V_\rho(r) = V(r)\theta(\rho - r)$, where $V(r)$ is the real potential and $\theta(r)$ is the step function. Define $\delta k,\rho$ as the phase shift for V_ρ at momentum k. In the limit $\rho\to\infty$, $\delta(k,\rho)\to\delta(k)$. The relation between potential and phase shift for S-wave is:

$$\frac{d\delta(k,r)}{dr} = -\frac{2mV(r)}{k}sin^2[kr+\delta(k,r)]. \tag{9}$$

The initial condition is $\delta(k,0) = 0$, since phase shift is 0 when there is no potential.

3 Potentials

Two different potentials are used in the work: Exponential potential and Square well potential.

3.1 Exponential potential

Potentials in the exponential form are used to fit the experimental phase shifts. Three exponential potentials with different effective ranges and strengths are used to account for the short range ($a = 1$), intermediate range ($b = 4$) and long range ($c = 7$) nuclear interaction.

$$V(r) = V_a\frac{e^{-ax}}{x} + V_b\frac{e^{-bx}}{x} + V_c\frac{e^{-cx}}{x}, \tag{10}$$

where $x = \mu r$, $\mu = 0.7\text{fm}^{-1}$, $V_a = -10.463$ MeV, $V_b = -1650.6$ MeV, $V_c = -6484.3$ MeV

$\hbar = c = 1$ is used in the calculation. Length is in unit of fm and energy is in fm^{-1}. With $\hbar c = 1 = 197$ MeV $\cdot$ fm, we get $\text{fm}^{-1} = 197$ MeV, with which all energy in units of MeV can be transformed into fm^{-1} unit.

In the R-matrix approach, Lippmann-Schwinger equation is solved in the momentum space, which requires us to transform the exponential potentials into momentum space with the help of Eq. (2). Since only S-wave is considered, $j_0(kr) = sin(kr)/kr$ is used.

$$\begin{aligned} I &= V_\eta \int_0^\infty dr r^2 \frac{sin(kr)}{kr} \frac{e^{-\eta x}}{x} \frac{sin(k'r)}{k'r} \\ &= \frac{V_\eta}{\mu k k'} \int_0^\infty dx \frac{sin(kr) e^{-\eta x} sin(k'r)}{x} \\ &= \frac{V_\eta}{2\mu k k'} \int_0^\infty dx (cos(\alpha x) - cos(\beta x)) \frac{e^{-\eta x}}{x}, \end{aligned} \tag{11}$$

where the last step use the relation $sinusinv = \frac{1}{2}[cos(u-v) - cos(u+v)]$ and $\alpha = \frac{k-k'}{\mu}$, $\beta = \frac{k+k'}{\mu}$. By adding 1 and subtracting 1 and using standard integral tables, the integral can be rewritten as:

$$\begin{aligned} I &= \frac{V_\eta}{2\mu k k'} \Big[\int_0^\infty dx (1 - cos(\beta x)) \frac{e^{-\eta x}}{x} - \int_0^\infty dx (1 - cos(\alpha x)) \frac{e^{-\eta x}}{x} \Big] \\ &= \frac{V_\eta}{4\mu k k'} ln \Big[\frac{(\mu\eta)^2 + (k+k')^2}{(\mu\eta)^2 + (k-k')^2} \Big] \end{aligned} \tag{12}$$

3.2 Square well potential

Square well potentials have the form:

$$V(r) = \begin{cases} -V_0 & \text{if } r < R, \\ 0 & \text{otherwise.} \end{cases}$$

When $r < R$, the potential is constant attractive or repulsive based on the sign of V_0. Outside of the range R, potential vanishes.

4 Results and discussion

4.1 Phase shifts for exponential potentials

A parametrized potential is used to calculate the singlet S-state $^1S_0$ between a proton and neutron, where spin $S = 0$ and orbital momentum $\ell = 0$. Both R-matrix method and VPA are used to calculate the phase shifts at different scattering energy.

Fig. (1) shows the phase shifts calculated using R-matrix approach, where kinetic energy in the lab frame is along the horizontal axis and vertical axis shows the phase shifts at different energies.

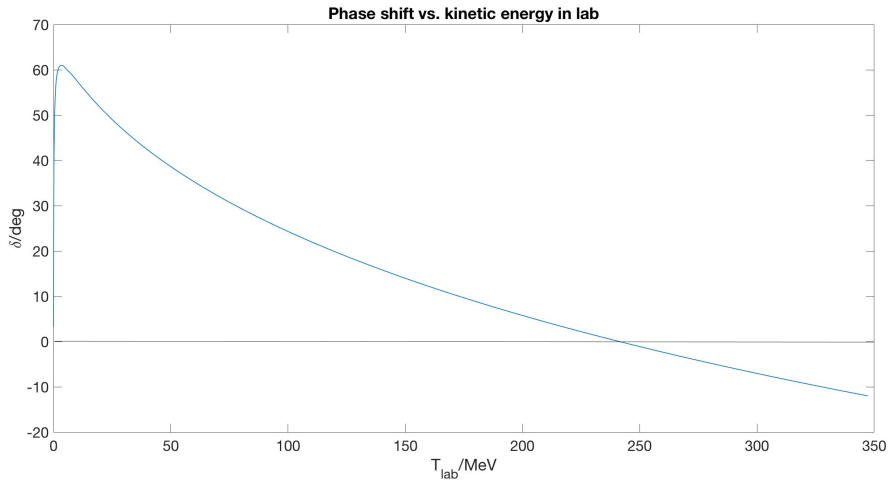

Figure 1: Phase shifts for ${}^1S_0$ state of Exponential potentials using R-matrix approach

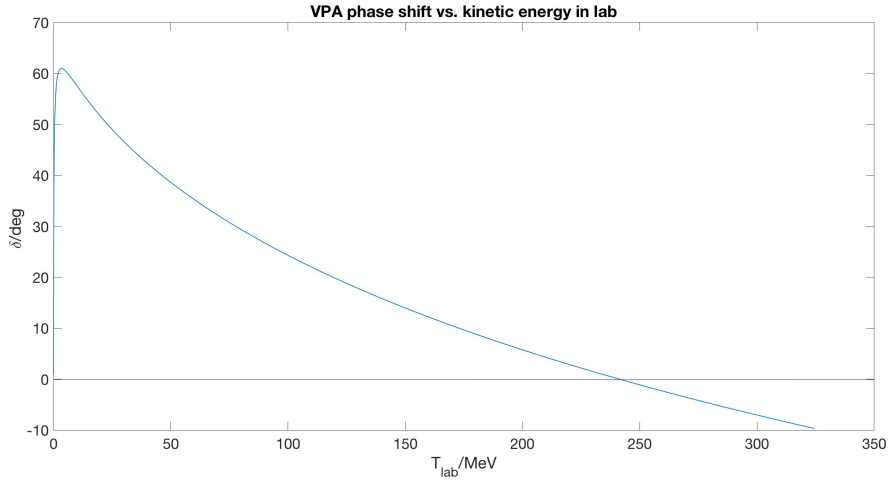

Figure 2: Phase shifts for ${}^1S_0$ state of Exponential potentials using VPA

Results for VPA is in Fig. (2). Both methods have almost identical results. The maximum phase shift is about 61^o when scattering energy is $E_{lab} = 6$

MeV, after which phase shift decreases for higher scattering energy. Phase shift become 0 at about 250 MeV. Thus there are no resonances or bound states for such a potential. Our calculations are similar to the results for $^1S_0$ state shown in [1].

4.2 Phase shifts for square well potential with VPA

Square well potentials have been used a lot in quantum mechanical systems due to its simplicity, which makes analytical derivation possible. For square well potentials with width R and depth V_0, its phase shift follows the relation [2]:

$$\delta k = arctan\Big(\sqrt{E/(E+V_0)} * tan\big(R\sqrt{2m(E+V_0)}\big)\Big) - R\sqrt{2m*E} \qquad (13)$$

In this part, calculations are carried out for square well potential with $V_0 = 4.0$ fm^{-1} and $R = 1$ fm employing VPA. The results are compared with the analytical solutions in Fig. (3). Figure on the left side shows the $kcot(\delta k)$ versus k and momentum dependence of phase shift is on the right side. It can be clearly seen that numerical calculation using VPA matches well with analytical solution. Due to the multiplicity of function $f(x) = cotx$ and analytical constraints the phase shift in the range $[-\pi, \pi]$, the phase shift on the right side has some mismatch, which does not affect the result.

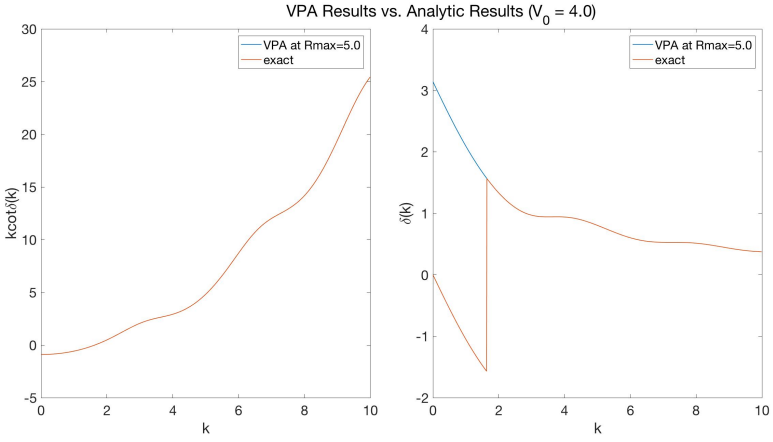

Figure 3: Phase shifts for $^1S_0$ state of square well potential using VPA

4.3 Phase shift and bound state

Eq. (9) is the fundamental differential equation in VPA. If V_0 is fully positive or negative, considering the property that $sin^2(x) \geq 0$ always hold, the RHS of Eq.(9) will have definite sign for all the r. For a fully positive V_0, the phase

shift decreases with increasing r. For a fully negative V_0l, the phase shift increases with r. So a fully attractive potential gives positive phase shift and fully repulsive potentials support negative phase shift.

Since $sin^2(x) \leq 1$, the RHS of Eq. (9) has the maximum absolute value $\frac{2MV}{k}$. In the high energy limit $k \to \infty$, this maximum value vanishes. Integrating the differential equation returns $\delta(k_\infty) = 0$. With high enough scattering energy, a high speed proton can travels so fast that it's insensitive to the potential generated by a static neutron. The phase shift for this reaction is negligible. In Levinson's theorem:

$$\delta(k = 0) - \delta(k_\infty) = n\pi, \tag{14}$$

where n is the number of bound state. It can be obtained $\delta(0) = n\pi$, where connection between number of bound states and the phase shift at zero energy is built.

In Fig. (4), phase shifts for square well potentials with different depths are calculated. Potential depths are adjusted near the threshold where the first bound state emerges. Phase shifts are in the unit of Radian and x-axis shows the momentum of scattering particle. With $V_0 = 1.21$ fm^{-1}, maximum of the phase shift barely pass 1.4 rad. As strength V_0 gets bigger, maximum of the phase shift increase sharply at $V_0 = 1.24$ fm^{-1} and exceeds 3.14. The first bound state is formed based on the Levinson's theorem. More calculations on phase shifts are obtained in Fig. (5) for stronger potential V_0. Although the phase shift for $V_0 = 5$ fm^{-1} and $V_0 = 10$ fm^{-1} only cross $\delta = \pi$ once, they do show the trend that more bound states will be formed with stronger potential.

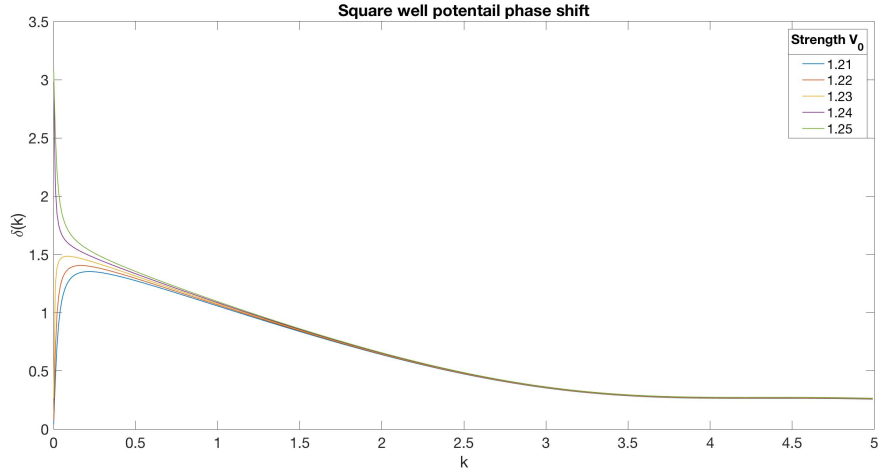

Figure 4: Phase shifts for square potentials with different depths using VPA

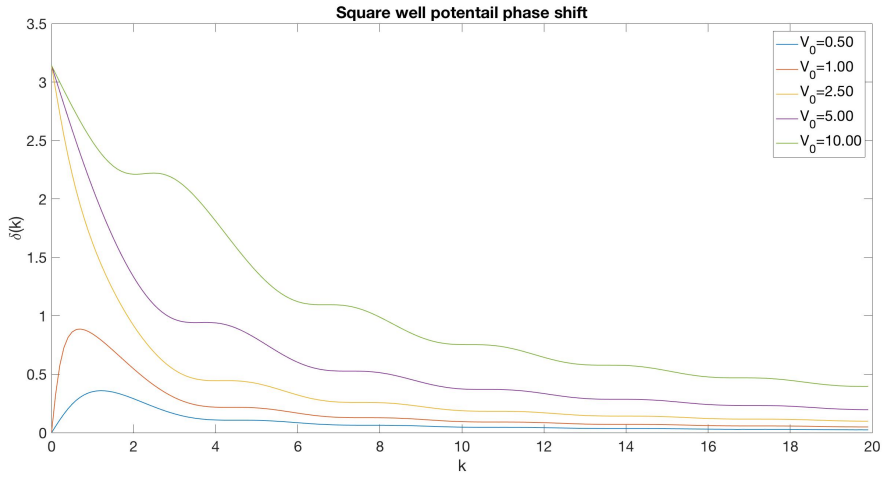

Figure 5: Phase shifts for square potentials with different depths using VPA

5 Summary

In this work, two sets of codes were developed based on R-matrix approach and Variable Phase Approach, whose details corresponding code were also discussed. Both approaches were applied to calculate the phase shift for ${}^0S_1$ state of nucleon nucleon scattering, whose results are consistent to the experimental results. VPA is used to calculated the phase shift for square well potential, whose analytical solution is available. VPA gave almost identical results with the analytical results. Relation between phase shift and number of bound states can be reveal clearly using VPA. Threshold is found for square well potential using VPA.

References

[1] Stoks, V. G. J. and Klomp, R. A. M. and Rentmeester, M. C. M. and de Swart, J. J. Phys. Rev. C, 48 792-815, 1993

[2] Morten Hjorth-Jensen, https://github.com/ManyBodyPhysics/NuclearForces/blob/master/doc/Projects/2017/Project1/VariablePhaseApproach.ipynb

Three different codes are listed here in the appendix: 1)R-matrix approach for Exponential potential (in C++):

```
#include<cmath>
#include<iostream>
#include<fstream>
```

```
#include<iomanip>
#include<stdlib.h>
#include<stdio.h>
#include "armadillo"

const double mass=939.0/197.0;
// parameters for potential, h_bar*c = 1
#define a 1
#define b 4
#define c 7
#define miu 0.7
#define Va -10.463/197.0
#define Vb -1650.6/197.0
#define Vc 6484.3/197.0
#define ZERO   1.0E-10

using namespace arma;
using namespace std;

double Poten(double k1, double k2);
void GaussLegendreQuadrature(double, double, double *, double*, int);

int main()
{
     int n=100;
//   set up the mesh points and weights
     double *x = new double [n+1];
     double *w = new double [n+1];
     double *r = new double [n+1];
     double *s = new double [n+1];
//   set up matrices V, A and R
     mat A = zeros<mat>(n+1,n+1);
     mat R = zeros<mat>(n+1,n+1);
     mat V = zeros<mat>(n+1,n+1);
     mat u = zeros<vec>(n+1);
     GaussLegendreQuadrature(-1.0, 1.0, x, w, n);
     double pi = 3.14159265359;
     for ( int i = 0;  i < n; i++){
        double xx=0.25*pi*(x[i]+1.0);
        r[i]= tan(xx);
        s[i]=0.25*pi/(cos(xx)*cos(xx))*w[i];
     }

// start loop for different k0
    for(int np = 0; np < 100; ++np){
        r[n] = np*0.03 + 0.01;
```

```
      for (int i = 0; i < n; ++i){
         u(i) = 2.0/pi*s[i]*r[i]*r[i]*mass/(r[n]*r[n]-r[i]*r[i]);}

         u(n) = 0;   //########  initialize u[n]
         for (int i =0; i < n; ++i){
                 u(n) = u(n)-2.0/pi*mass*s[i]/(r[n]*r[n]-r[i]*r[i]);}
         u(n) = u(n)*r[n]*r[n];

         for(int i = 0; i < n+1; ++i){
             for(int j = 0; j < n+1; ++j){
                 V(i,j) = Poten(r[i],r[j]);
                 if (i != j){A(i,j) = -V(i,j)*u(j);}
                 else{ A(i,j) = 1.0-V(i,j)*u(j);}
             }
         }

         mat A_inv = inv(A);
         R = A_inv*V;
         double sigma = atan(-R(n,n)*mass*r[n]);
         double Energy = r[n]*r[n]/mass*197;
        cout << sigma/pi*180 << " " << Energy << endl;
      }
      delete [] x;
      delete [] w;
      delete [] s;
      delete [] r;
      return 0;
}

//  this function defines potential in momentum space
double Poten(double k1, double k2)
{
  double value = 0.0;
  value = value + 0.25*Va/(miu*k1*k2) * log((pow(k1+k2,2)+ pow(miu*a,2)) \
                /(pow(k1-k2,2)+ pow(miu*a,2)));
  value = value + 0.25*Vb/(miu*k1*k2) * log((pow(k1+k2,2)+ pow(miu*b,2)) \
                /(pow(k1-k2,2)+ pow(miu*b,2)));
  value = value + 0.25*Vc/(miu*k1*k2) * log((pow(k1+k2,2)+ pow(miu*c,2))
\
                /(pow(k1-k2,2)+ pow(miu*c,2)));
  return value;
}

void GaussLegendreQuadrature(double x1, double x2, double x[], double w[], int n
{
```

```
int          m,j,i;
double       z1,z,xm,xl,pp,p3,p2,p1;
double       const  pi = 3.14159265359;
double       *x_low, *x_high, *w_low, *w_high;

m  = (n + 1)/2;                                  // roots are symmetric in the int
xm = 0.5 * (x2 + x1);
xl = 0.5 * (x2 - x1);

x_low  = x;                                          // pointer initialization
x_high = x + n - 1;
w_low  = w;
w_high = w + n - 1;

for(i = 1; i <= m; i++) {                                 // loops over desired r
   z = cos(pi * (i - 0.25)/(n + 0.5));

        /*
        ** Starting with the above approximation to the ith root
        ** we enter the mani loop of refinement bt Newtons method.
        */

   do {
      p1 =1.0;
      p2 =0.0;

        /*
        ** loop up recurrence relation to get the
        ** Legendre polynomial evaluated at x
        */

      for(j = 1; j <= n; j++) {
         p3 = p2;
         p2 = p1;
         p1 = ((2.0 * j - 1.0) * z * p2 - (j - 1.0) * p3)/j;
      }

        /*
        ** p1 is now the desired Legrendre polynomial. Next compute
        ** ppp its derivative by standard relation involving also p2,
        ** polynomial of one lower order.
        */

      pp = n * (z * p1 - p2)/(z * z - 1.0);
      z1 = z;
      z  = z1 - p1/pp;                       // Newton's method
```

```
        } while(fabs(z - z1) > ZERO);

            /*
            ** Scale the root to the desired interval and put in its symmetric
            ** counterpart. Compute the weight and its symmetric counterpart
            */

        *(x_low++)  = xm - xl * z;
        *(x_high--) = xm + xl * z;
        *w_low      = 2.0 * xl/((1.0 - z * z) * pp * pp);
        *(w_high--) = *(w_low++);
    }
}
```

2) VPA for exponential potential (in MATLAB)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% In this program we use variable phase approach to extract phase
% shift for a toy nucleon-nucleon potential.
% This code can calculate several things:
% 1. Phase shift for different truncated position r for certain k;
% 2. Phase shift depending on k or energy;
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear all;
close all;
format long;
opts = odeset('RelTol',1e-6,'abstol',1e-6);

% parameters for potential and constants
global M M_pi Va Vb Vc k;
M = 939.0/197.0;
M_pi = 0.7;
Va = -10.463/197.0;
Vb = -1650.6/197.0;
Vc = 6484.3/197.0;
pi = 3.141592653589793;


k = 0.001;
rspan = 0.001:0.1:15;
y0 = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Defining the exponential potential
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
function dydr = VNN(r,y)
```

```
    global M M_pi Va Vb Vc k;
    Mu = M/2.0;
    V = Va*exp(-1.0*M_pi*r)/(M_pi*r);
    V = V + Vb*exp(-4.0*M_pi*r)/(M_pi*r);
    V = V + Vc*exp(-7.0*M_pi*r)/(M_pi*r);
    dydr = (-2.0*Mu/k)*V*(sin(k*r + y))^2;
end

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 1. calculate phase shift depending on truncated position of potential
% [r,y] = ode45(@VNN, rspan, y0, opts);
% plot(r,y/pi*180)
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%


% 2. phase shift vs. k(or Energy)
delta = zeros(1,100);
for i = 1:100
    [r,y] = ode45(@VNN, rspan, y0, opts);
    delta(i) = y(length(y));
    E(i) = k^2/M*197.0*2;
    kcot(i) = k/tan(delta(i));
    k = k + 0.1;
end

x = 0.001:0.1:10;
%figure(1);
%plot(x, delta/pi*180);

%figure(2);
%plot(E, delta/pi*180,'linewidth',1);
trans_kcot = transpose(kcot);
trans_x = transpose(x);
plot(x, kcot)
title('VPA phase shift vs. kinetic energy in lab')
xlabel('T_{lab}/MeV')
ylabel('\delta/deg ')
```

3) VPA for square well potential (in MATLAB)

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% In this program we use variable phase approach to extract phase
% shift for square well potential with width R and depth -V0.
% This code can calculate several things:
% 1. Phase shift for different truncated position r for certain k;
% 2. Phase shift depending on k or energy;
% 3. Find the critical strength for a bound state first appears.
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
clear all;
close all;
format long;
opts = odeset('RelTol',1e-6,'abstol',1e-6);

global Mu V0 R k;

%for j = 1:5
  Mu = 1.0;
  R = 1.0;
  V0 = 4.0;
  %V0 = 1.1819; % 3. try to find a bound state
  k = 0.001;
  rspan = [0:0.05:5];
  y0 = 0;

%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Define the square well potential
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
  function dydr = Vsw(r,y)
     global Mu V0 R k;
     if r > R | r < 0
        V = 0;
     else
        V = -V0;
     end
     dydr = (-2.0*Mu/k)*V*(sin(k*r + y))^2;
  end


%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% 1. calculate phase shift depending on truncated position of potential
% [r,y] = ode45(@Vsw,rspan,y0,opts);
% plot(r,y)
% xlim([-0.1 5])
% ylim([-0.1 0.8])
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

% 2. calculate phase shift for different k
delta = zeros(1,1000);
for i = 1:1000
   [r,y] = ode45(@Vsw,rspan,y0,opts);
   delta(i) = y(length(y));
   kcot(i) = k/tan(delta(i));
   E = k^2/2.0/Mu;
```

```
    delta_exact(i) = atan(tan(analytic(E)));
    kcot_exact(i) = k/tan(delta_exact(i));
    k = k + 0.01;
end

x = 0.001:0.01:10;
figure;
subplot(1,2,1);
plot(x,kcot,'linewidth',1.0);
hold on;
plot(x,kcot_exact,'linewidth',1.0);
xlabel('k')
ylabel('kcot\delta(k)')
legend("VPA at Rmax=5.0","exact")
hold off;

subplot(1,2,2);
plot(x,delta,'linewidth',1.0);
hold on;
plot(x,delta_exact,'linewidth',1.0)
xlabel('k')
ylabel('\delta(k)')
legend("VPA at Rmax=5.0","exact")
hold off;

% trans_x = transpose(x);
% trans_kcot = transpose(kcot);
%plot(x,delta,'linewidth',1);
%legend("V_0=0.50","V_0=1.00","V_0=2.50",...
%       "V_0=5.00","V_0=10.00","V_0=20.00");
%end
%title('Square well potentail phase shift')
```