计算物理第三次作业
一维薛定谔方程定态解

张楚珩（121120173）

May 28, 2014

1 问题描述

对于定态薛定谔方程

$$\begin{cases} [-\frac{d^2}{dx^2}+\frac{2m}{h^2}(V(x)-E)]\psi(x)=0 \\ \psi(x_{min})=\psi(x_{max})=0 \end{cases} \tag{1}$$

在给定势阱的情况下，求使这个问题有非零解的本征值E_0及其相应的波函数。
在本次作业中，我们将要求解如下几种势阱：

$$V(x)=(x-1)^2(x+2)^2 \tag{2}$$

$$\text{有限高方势阱} V(x)=\begin{cases} -100 & -1<x<1 \\ 0 & x<-1\text{或}x>1 \end{cases} \tag{3}$$

$$\text{抛物势阱} V(x)=x^2 \tag{4}$$

2 问题分析及算法设计

我们利用打靶法来求解该本征值问题。可以看到，本征值问题

$$\begin{cases} (\frac{d^2}{dx^2}+k^2)\psi=0 \\ \psi(x_{min})=\psi(x_{max})=0 \end{cases} \tag{5}$$

相比于定解问题，本征值问题多了一个待定的参数k，因此先猜测一个试验本征值k，同时任设一个非零参数δ，把微分方程变为初值问题

$$\begin{cases} (\frac{d^2}{dx^2}+k^2)\psi=0 \\ \psi(x_{min})=\psi(x_{max})=0, \psi'(x_{min})=\psi'(x_{max})=\delta \end{cases} \tag{6}$$

对于本题要求解的定态薛定谔方程，我们可以从x_{min}出发先去积分，可以产生一个数值解$\psi_<$。它在经典禁戒的区指数增长，转过左折点之后进入经典容许的区域，并在此区域内振荡。

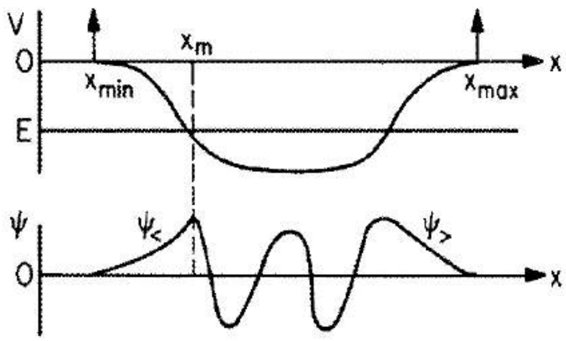

Figure 1: 波函数在势阱下的状态

如Figure 1 所示。如果越过右节点再继续积分下去，那么这个数值将变得不稳定。因为即使在一个精确的能量本征值上也可能混入一个不想要的指数增长成分，这将导致进入经典禁戒区的积分很可能是不准确的。

因此，我们采取如下做法。在每一个试探的本征值上，由x_{min}出发向前积分产生一个数值解$\psi_<$，同时，由x_{max}出发向前积分产生一个数值解$\psi_>$。为了判断这个试探能量值是不是能量本征值，我们选择一个接合点x_m上比较$\psi_<(x_m)$和$\psi_>(x_m)$是否满足如下数学表达式。

$$\begin{cases} \psi_<(x_m) = \psi_>(x_m) \\ \dfrac{\psi_<}{dx}|_{x_m} = \dfrac{\psi_>}{dx}|_{x_m} \end{cases} \tag{7}$$

在实际的算法中，我们先将右边的波函数$\psi_>$乘上一个系数$\dfrac{\psi_<(x_m)}{\psi_>(x_m)}$得到一个新的波函数

$$\psi^{new}(x) = \begin{cases} \psi_<(x) & x < x_m \\ \dfrac{\psi_<(x_m)}{\psi_>(x_m)}\psi_>(x) & x > x_m \end{cases} \tag{8}$$

同时，我们使用表达式

$$f \equiv \frac{1}{\psi(x_m)}[\psi_<(x_m - h) - \psi_>(x_m - h)] < ERROR\_TOLERANCE \tag{9}$$

来判定，所试探的能量值是否为符合精度要求的本征值。

3 代码

下面给出了C语言编写的代码：

```
//
//   main.c
//   Schrodinger
```

```
//
//  Created by ZHANG CH on 14-4-26.
//  Copyright (c) 年2014 NJU. All rights reserved.
//

//问题陈述:
// 利用打靶法解一维薛定谔方程本征值与本征波函数 :
//      d2u/dx2 + k2(x)u = 0
// 其中      k2(x) = 2m/h^2 [E - V(x)]
//
//二阶差分公式:
//  d2u/dx2 = (u_i-1 - 2u_i + u_i+1)/(h^2) = k2(x)u
// 推得:  u[i+1] = (2-h^2 k2(x)) u[i] - u[i-1]

#include <stdio.h>
#include <math.h>

#define X_MIN  (-4.0)   //的下界x
#define X_MAX (4.0)     //的上界x
#define DELTA (1.0)     //边界处的导数
#define h (0.001)       //步长
#define MAXN (8000)     //数组大小 应等于(X_MAX-X_MIN)/h
#define MIDN (4000)     //处的数组下标x_m 即x_m=0
#define COEFFIENT (1.0) //归一化系数 2m/h^2
#define E_START (-10.0) //起始处的能量试探值 需比真实本征值小
#define DELTA_E_START (0.1)     //最开始的时候能量试探值的增量
#define ERROR_TOLERANCE (0.0000001)     //最好的评估函数的误差容忍f

double u[MAXN] = {0};   //储存波函数值的数组,即波函数psi
double E = E_START;     //试探能量值

double function_k2(double E, double x);
double function_f(double E);
double function_V(double x);
void run();
void print();

int main(int argc, const char * argv[])
{
    run();
    print();
    return 0;
}

void run()
{
```

```
    double dE = DELTA_E_START;
    double f_old = function_f(E);
    double f_new = function_f(E += dE);
    while (1) {
        if (f_old*f_new <0)
        {
            dE = -(dE/2.0);
        }
        f_old = f_new;
        f_new = function_f(E += dE);
        if (fabs(f_new) < ERROR_TOLERANCE) {
            break;
        }
    }
}

void print()
{
    FILE * fp;
    int i;
    fp = fopen("output.txt", "w+");
    fprintf(fp, "E = %lf\n", E);
    fprintf(fp, "[");
    for (i=0; i<MAXN; i++) {
        if (i != 0) fprintf(fp, ",");
        fprintf(fp, "%lf ", u[i]);
    }
    fprintf(fp, "]");
    fclose(fp);
}

double function_f(double E)
{
    int i;
    double Umr,Uml;
    u[0] = 0; u[1] = DELTA; u[MAXN-1] = 0; u[MAXN-2] = DELTA;
    //使用二阶差分计算关于的定解问题u
    //正向
    for (i=MAXN-3; i>=MIDN; i--) {
        u[i] =(2-h*h*function_k2(E, X_MIN+(i*h)))*u[i+1] -u[i+2];
    }
    Umr = u[MIDN];
    //逆向
    for (i=2; i<=MIDN; i++) {
        u[i] = (2-h*h*function_k2(E, X_MIN+(i*h)))*u[i-1] -u[i-2];
    }
```

```
    Uml = u[MIDN];
    for (i=MIDN+1; i<MAXN; i++) {
        u[i] = u[i]*Uml/Umr;
    }
    return (1/Uml) * (2*u[MIDN] - u[MIDN+1] - u[MIDN-1]);
}

double function_k2(double E, double x)
{
    return COEFFIENT*(E - function_V(x));
}

//改变一下函数的值可以求出不同势垒下的本征值和本征波函数
double function_V(double x)
{
    //作业题目中给定的势垒
    return (x-1)*(x-1)*(x+2)*(x+2);
    //有限高方势垒
    //return ((x<1) && (x>-1))? -100: 0;
    //抛物势垒
    //return x*x;
}
```

另外给出了MATLAB下的代码，代码调用了MATLAB内置的四阶和五阶混合龙格库塔法ode45。ode45用4阶方法提供候选解，5阶方法控制误差，是一种自适应步长（变步长）的常微分方程数值解法，其整体截断误差为Δx^5。

```
%在文件 main.m 中
function main
global E
clc;
n=9;
eold=0;
tol=1e-2;
xturn=-0.5;
kk=0.1;
xmin=-4;
%xmax=3;

for k=1:n
   E=eold+abs(eold)/70;

   dE=1;

   [x1,u1]=ode45(@fun,[xmin xturn],[0 kk]);
   fold = u1(length(x1),2);

```

```
    while (abs(fold) > tol)
        E = E + dE;
        [x1,u1]=ode45(@fun,[xmin xturn],[0 kk]);
        fnew = u1(length(x1),2);
        if (fold*fnew < 0)
            dE = -(dE./2);
        end
        fold = fnew;
    end
    eold = E;
    subplot(3,3,k);
    plot(x1,u1(:,1),'b',(-1).*ones(size(x1))-x1,u1(:,1),'b');
    axis tight;
    title(['第',num2str(k),'个本征态 ','E=',num2str(eold)]);
end
```

```
%在文件 fun.m 中
function yy=fun(x,psi)
global E
    yy=[psi(2);(-E+((x-1)^2)*((x+2)^2))*psi(1)];
```

4 计算结果

4.1 三个不同势阱下的波函数

对于双峰抛物线势阱$V(x)=(x-1)^2(x+2)^2$可以得到相应的本征值$E_0=2.583008$，相应的波函数如图Figure2。

对于有限高方势阱可以得到相应的本征值$E_0=97.962109$，相应的波函数如图Figure3。

对于抛物势阱可以得到相应的本征值$E_0=0.999414$，相应的波函数如图Figure4。

4.2 双峰抛物线下的不同本征值下的波函数

对于双峰抛物线势阱$V(x)=(x-1)^2(x+2)^2$可以得到不同的本征值E，由此，图5给出了程序计算得到的前九个能级的波函数。

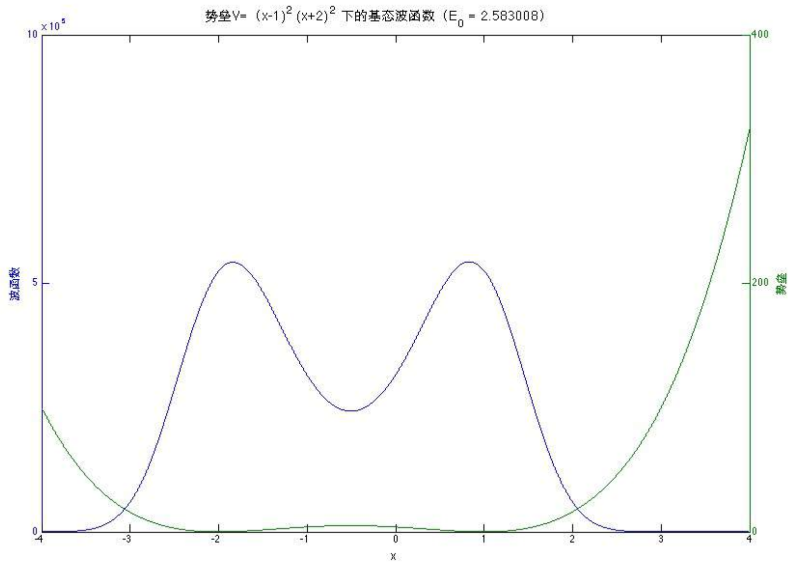

Figure 2: 双峰抛物线势阱下的波函数

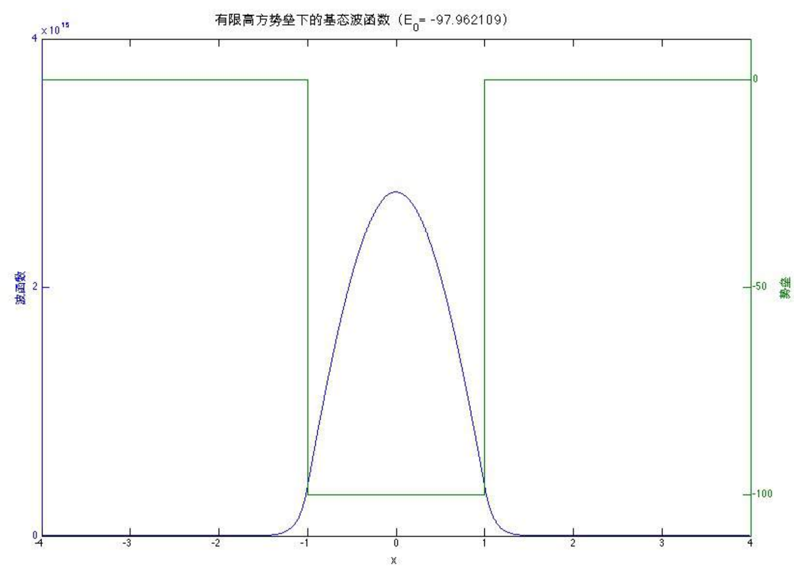

Figure 3: 有限高方势阱下的波函数

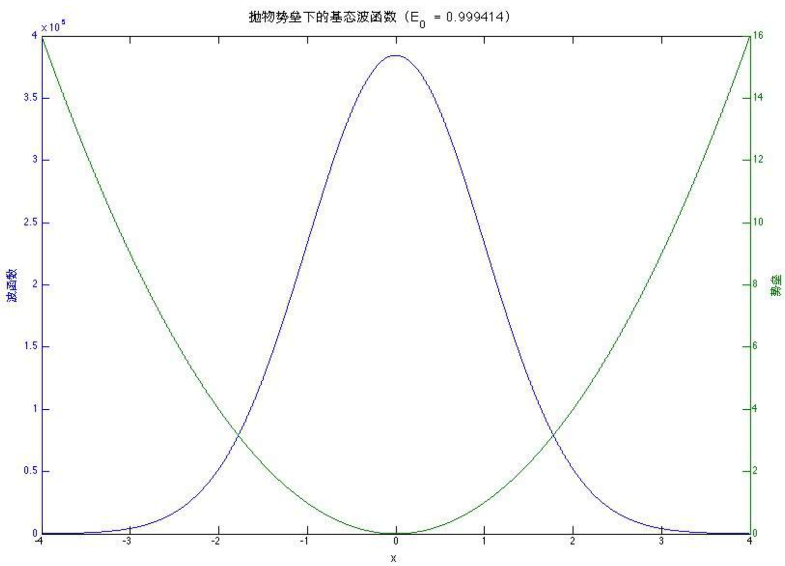

Figure 4: 抛物势阱下的波函数

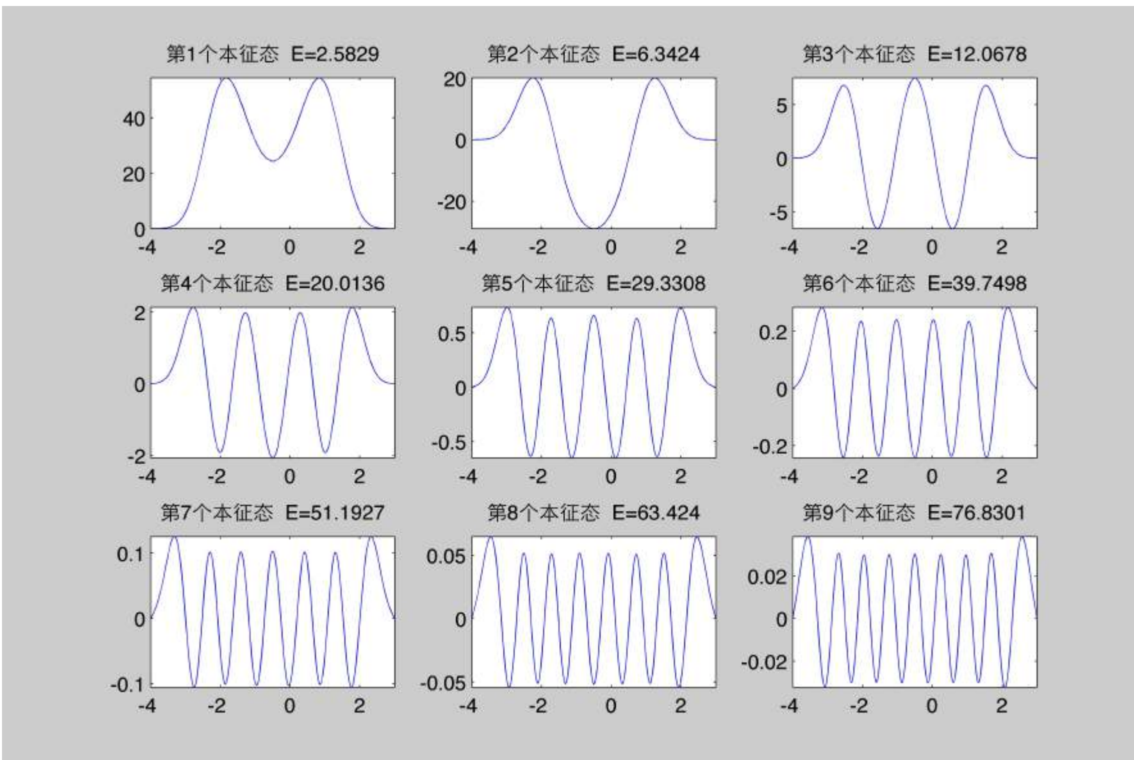

Figure 5: 双峰抛物线势阱下前九个能级的波函数