**核安全综合保障**

**基于SPH方法的U材料力学性能探究**

|  |  |  |
| --- | --- | --- |
| 姓名 |  | 刘丰硕 |
| 学号 |  | 201900501007 |

# 练习1：基于SPH方法的导数求解器

## 1.SPH方法基本原理

1. SPH基本思想

将流体离散为一系列具有质量，任意分布的拉格朗日粒子，利用核近似和粒子近似的方法用粒子所携带的物理量对宏观物理量以及空间导数进行近似。

核近似：对函数的进行微分可转换为对光滑函数进行微分，由此可以减小对场函数连续性的要求。（归一化，狄拉克，紧支性；二阶精度）

粒子近似：支持域内所有粒子叠加求和的离散化形式。

粒子i的任一函数值都可以通过应用光滑函数对紧支域内所有粒子所对应的函数值进行加权平均近似，也就是任意粒子处的函数及其导数的连续积分都转换成了在任意排列的粒子上的离散化求和。

1. 主要求解的问题

数学上：将偏微分方程组在问题域内离散化；获取任意一点的变量函数及其导数的近似值；将近似函数应用于偏微分方程组来获得一系列离散化的、只与时间相关的常微分方程。

物理上：对NS方程进行空间域离散化，推导出适用于广义流体动力学的SPH方程，得到一系列与时间相关的常微分方程。这一系列常微分方程可以通过对时间的积分进行求解。

1. 优势

SPH方法是一种纯拉格朗日性质的无网格粒子自适应算法，在处理大变形、跟踪运动界面或自由表面，以及获取变量的时间历程等方面有着很大优势。

## 2.可执行源代码

**Matlab代码**

syms x1 x2 N h length dilf;

x1=input('x1=');

x2=input('x2=');

N=input('N=');

hanshu=input('求导的原函数=',"s");

length=x2-x1;

h=2.2\*length/N;

a=5/(4\*h);

dd=[];

xx=[];

x=0;

xj=0;

x0=0;

for i=x1:length/N:x2

dilf=0;

for j=i-length/2:length/N:i+length/2

if j~=i

d=length/N;

R=abs(i-j)/h;

dr=(i-j)/(abs(i-j)\*h);

if R>1

dilw=0;

else

dilw=-a\*12\*R\*(1-R)^2\*dr;

end

x=j;

dilf=dilf+d\*eval(hanshu)\*dilw;

else

continue

end

end

dd=[dd;dilf];

xx=[xx;i];

end

y2=max(dd);

y1=min(dd);

figure

hold on;

h1=plot(xx,dd,'rx-','linewidth',2,'MarkerSize',2);

xlim([x1 x2]);

ylim([y1 y2]);

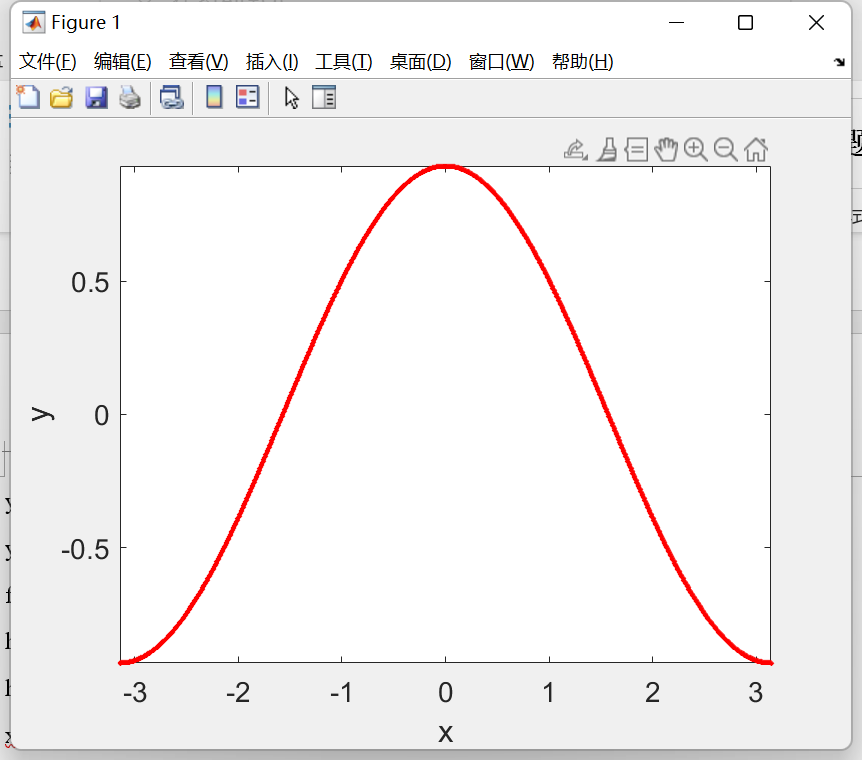
set(gca,'fontsize',14);

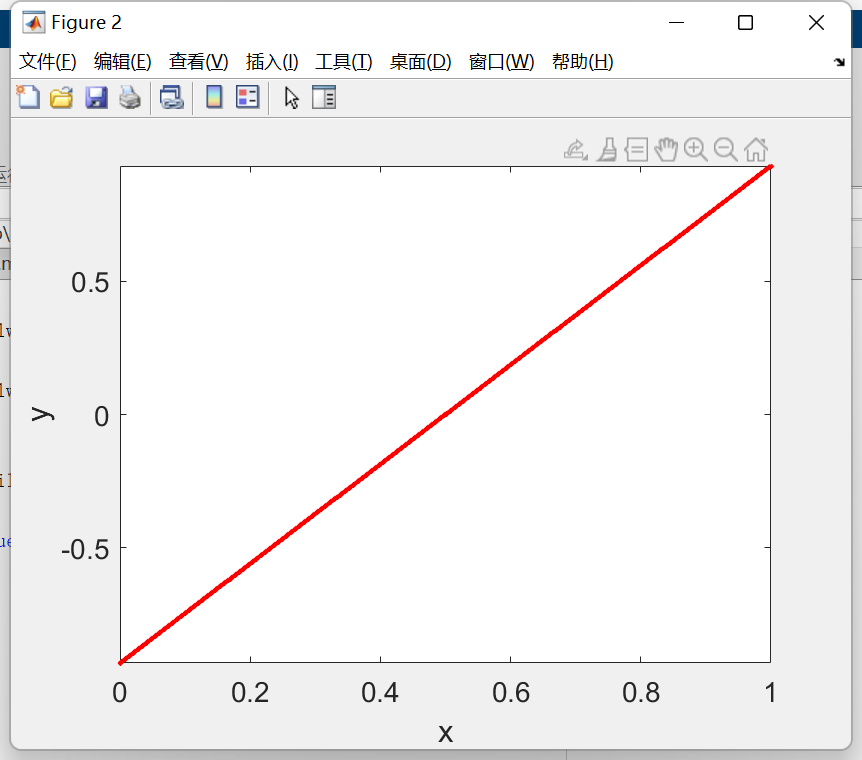
xlabel('x')

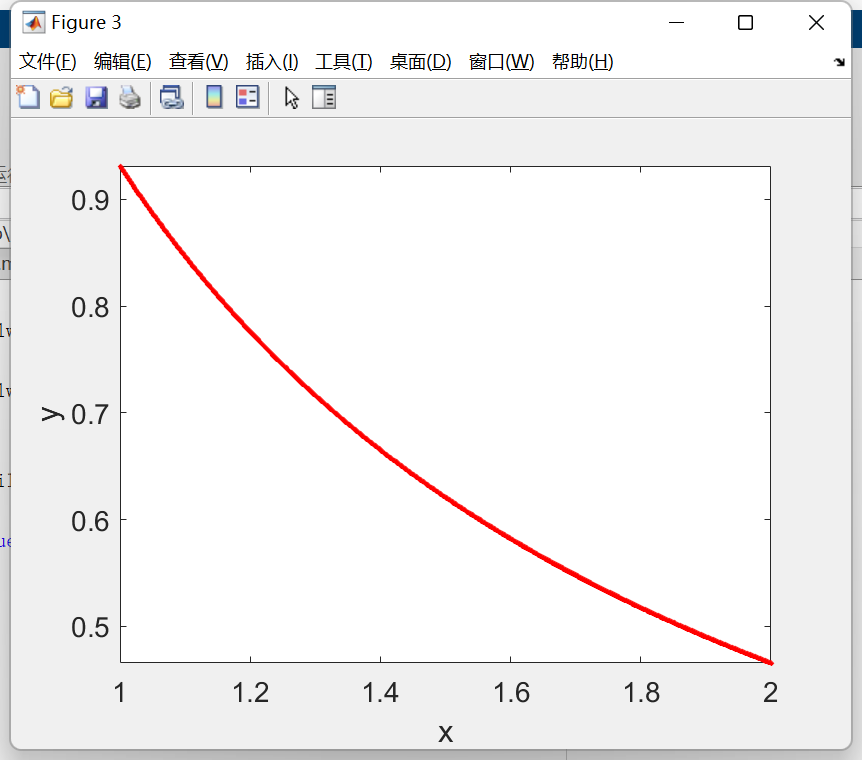
ylabel('y');

box on;

## 3.结果展示







练习2：三物块弹簧链接动力学计算

## 基本物理模型

三个物块，质量分别为1kg、2kg、3kg，由两个弹性系数为k=10N/m的弹簧连接。

计算在给定初始条件(1)、(2)下，此系统在0-10s内的位移-时间图像、能量-时间图像。

## 

1. v0=3m/s, x0=0m
2. v0=0m/s, x0=1m

## 2. 可执行源代码

**Matlab代码**

%初始条件

%质量

m1=1;

m2=2;

m3=3;

%位置

x1=1;

x2=5;

x3=10;

%弹簧弹性系数

k1=10;

k2=10;

%弹簧原长

yc1=5;

yc2=5;

%速度

v1=0;

v2=0;

v3=0;

%时间步长

t2=10;

t1=0;

length=t2-t1;

N=10000;

%初始化

x11=[];

x22=[];

x33=[];

tt=[];

epp=[];

ekk=[];

ee=[];

for t=t1:length/N:t2

dt=length/N;%时间步长

%位置数组

x11=[x11;x1];

x22=[x22;x2];

x33=[x33;x3];

tt=[tt;t];

%受力计算

dx1=x2-x1;

dx2=x3-x2;

F1=k1\*(dx1-yc1);

F2=k2\*(dx2-yc2);

%加速度计算

a1=F1/m1;

a2=(F2-F1)/m2;

a3=-F2/m3;

%速度更新

v1=v1+a1\*dt;

v2=v2+a2\*dt;

v3=v3+a3\*dt;

%能量计算

ep=1/2\*k1\*(dx1-yc1)^2+1/2\*k2\*(dx2-yc2)^2;

ek=1/2\*m1\*v1^2+1/2\*m2\*v2^2+1/2\*m3\*v3^2;

e=ep+ek;

epp=[epp;ep];

ekk=[ekk;ek];

ee=[ee;e];

%位置更新

x1=x1+v1\*dt;

x2=x2+v2\*dt;

x3=x3+v3\*dt;

end

%x-t图像

figure

plot(tt,x11,'linewidth',2,'MarkerSize',2)

hold on

plot(tt,x22,'linewidth',2,'MarkerSize',2)

hold on

plot(tt,x33,'linewidth',2,'MarkerSize',2)

hold on

legend('m1','m2','m3')

xlabel('时间/s')

ylabel('位置/m')

set(gca,'fontsize',14);

%e-t图像

figure

plot(tt,epp,'linewidth',2,'MarkerSize',2)

hold on

plot(tt,ekk,'linewidth',2,'MarkerSize',2)

hold on

plot(tt,ee,'linewidth',2,'MarkerSize',2)

hold on

legend('ep','ek','e')

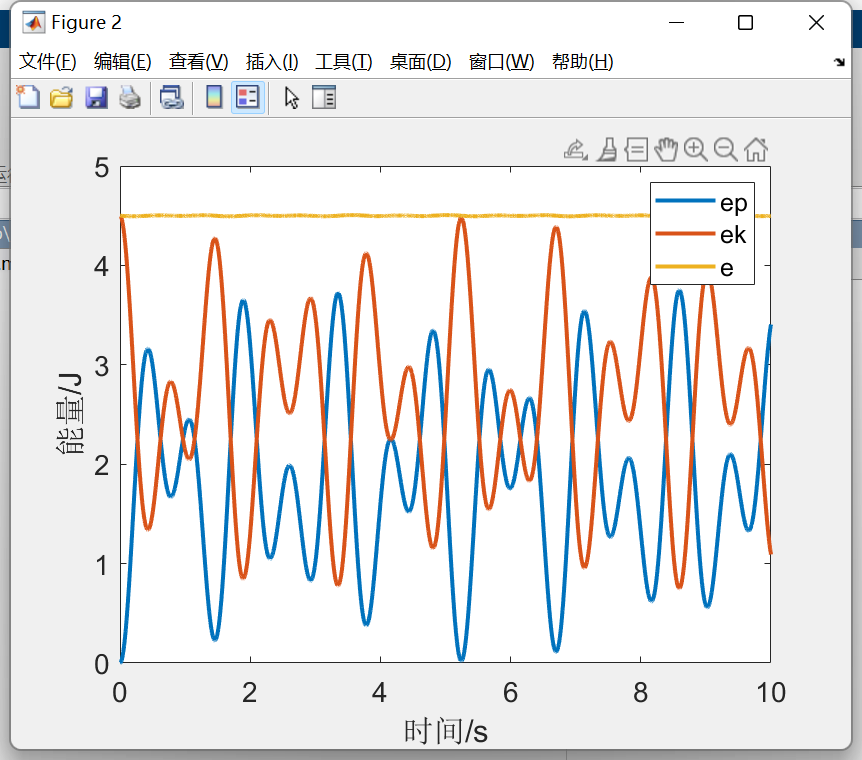
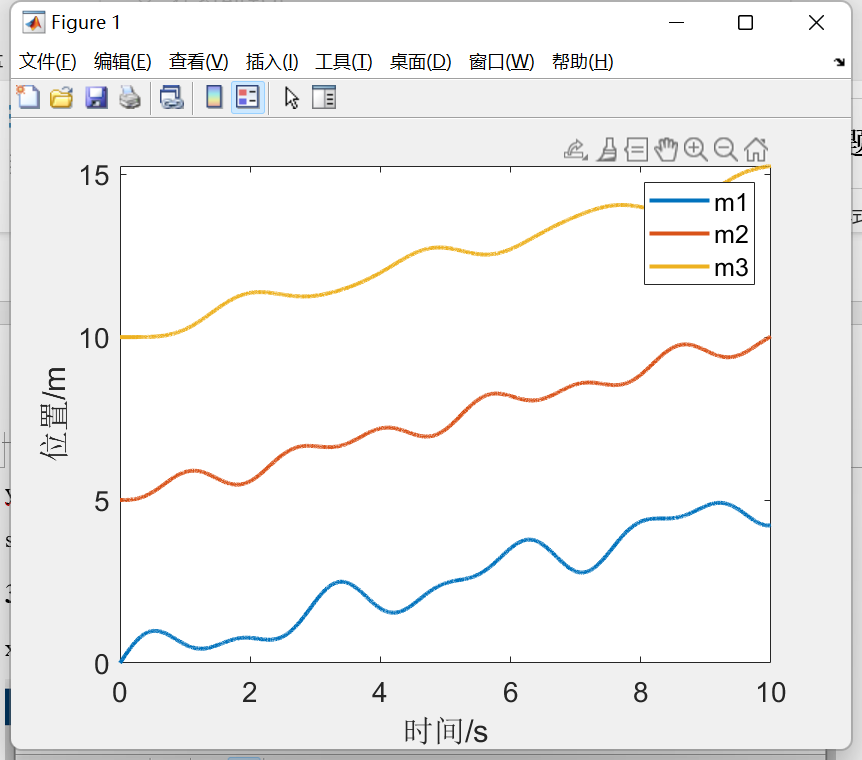
xlabel('时间/s')

ylabel('能量/J')

set(gca,'fontsize',14)

## 3.结果展示

当x=0 v=3时



当x=1 v=0时

