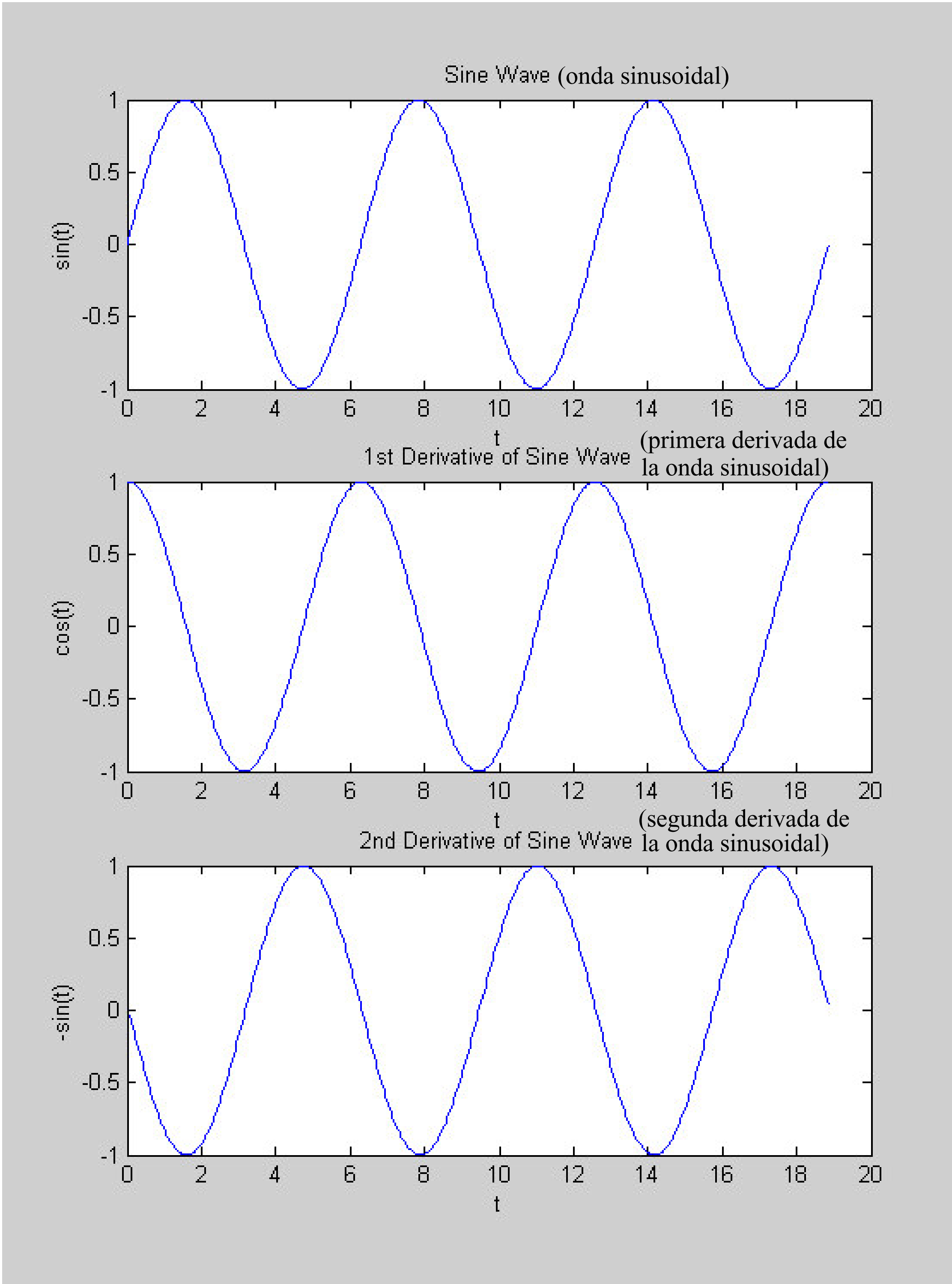
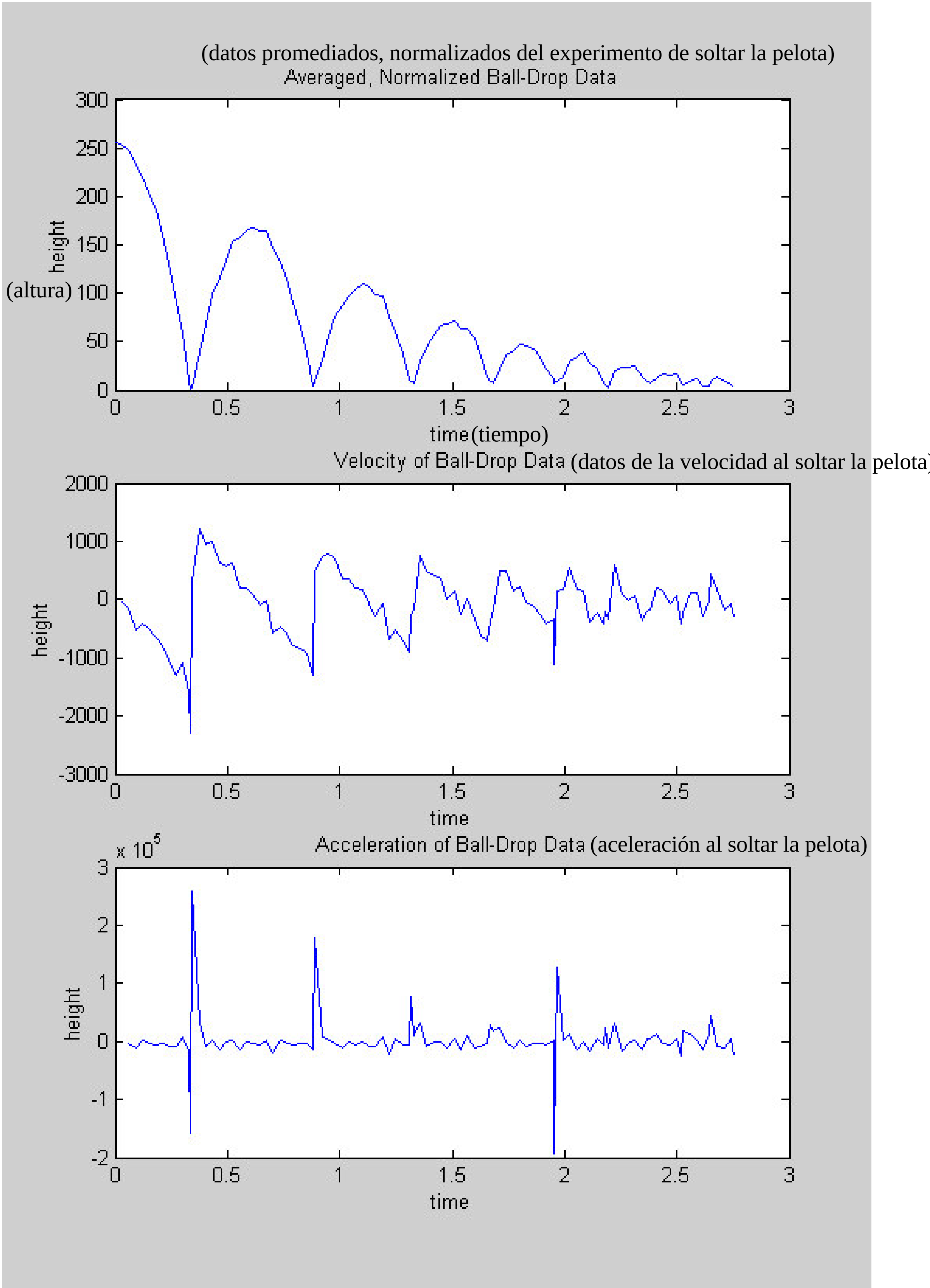
```
%Práctica 2, Apartado A


%primero, determine los datos de la práctica 1 de nuevo
load('drop_data.mat')
d=(d1+d2+d4+d5)/4;
s=min(d);
d=d-s+.1;

%a continuación, realice una onda sinusoidal para el otro test
t=linspace(0,6*pi,500);      %otra forma de crear un vector. Este
                             %tiene 500 puntos, mientras que el método
                             %t=[0:.1:6*pi] crea un array  con puntos
                             %separados por intervalos de 0.1
g=sin(t);

%apartado a)  Primera derivada
[dt,dg] = derivative(t,g);
[dTime,dv] = derivative(Time,d);

%apartado b) Segunda derivada
[ddt,ddg] = derivative(dt,dg);
[ddTime,da] = derivative(dTime,dv);


figure(1)
subplot(3,1,1)
plot(t,g), title('Sine Wave'),xlabel('t'),ylabel('sin(t)')
subplot(3,1,2)
plot(dt,dg),title('1st Derivative of Sine Wave'),xlabel('t'),
      ylabel('cos(t)')
subplot(3,1,3)
plot(ddt,ddg),title('2nd Derivative of Sine Wave'),xlabel('t'),
      ylabel('-sin(t)')


figure(2)
subplot(3,1,1)
plot(Time,d),title('Averaged, Normalized Ball-Drop Data'),
      xlabel('time'),ylabel('height')
subplot(3,1,2)
plot(dTime,dv),title('Velocity of Ball-Drop Data'),
      xlabel('time'),ylabel('height')
subplot(3,1,3)
plot(ddTime,da),title('Acceleration of Ball-Drop Data'),
      xlabel('time'),ylabel('height')

------------------

function [dervx, dervy] = derivative(x,y)

dervy = diff(y)./diff(x);       %aquí podía haber utilizado también un bucle FOR.
dervy = dervy';                 %la matriz se traslada por alguna razón.

dervx = x(2:length(x));         %acortar el vector x según corresponda
dervx = dervx';
```

>> lab2


Sine Wave (onda sinusoidal)
sin(t)
t
1st Derivative of Sine Wave (primera derivada de la onda sinusoidal)
cos(t)
t
2nd Derivative of Sine Wave (segunda derivada de la onda sinusoidal)
-sin(t)
t

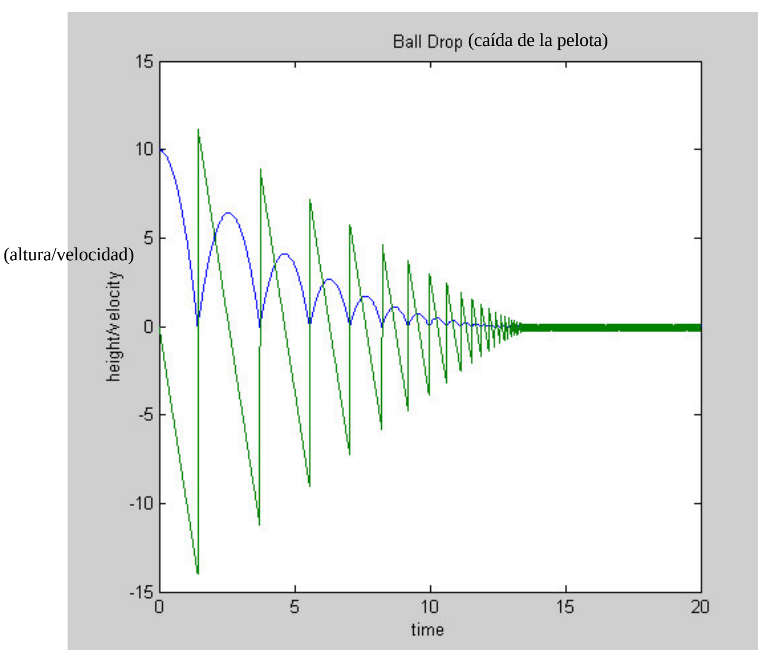

(datos promediados, normalizados del experimento de soltar la pelota)
Averaged, Normalized Ball-Drop Data
height
(altura)
time (tiempo)
Velocity of Ball-Drop Data (datos de la velocidad al soltar la pelota)
height
time
x 10^5
Acceleration of Ball-Drop Data (aceleración al soltar la pelota)
height
time

Apartado B

Se suponía que este apartado estaría también relacionado con la pelota que rebota, no obstante, dada la ecuación, no resultaba tan obvio mostrar el trazado de dicha trayectoria y cómo y cuándo golpearía la pelota el suelo. Es totalmente correcto haber trazado la trayectoria parabólica que cayó hasta -2000, como si alguien cayese desde un precipicio o similar. No obstante, a continuación proporcionamos un archivo m que calcula la posición y la velocidad e incluye el rebote y un coeficiente de restitución igual a 0.8, tal como se utilizará posteriormente en el problema.

```
function [z,v,impact] = drop(z0,v0,tf)

g=-9.8;
e=0.8;
t=[0:0.01:tf];

t0=0;
j=0;
for i=1:length(t)
    z(i) = 0.5*g*(t(i)-t0)^2 + v0*(t(i)-t0) + z0;
    v(i) = g*(t(i)-t0) + v0;

    if z(i) < 0          %cuando la pelota baja del nivel del suelo,
        j = j+1;         %invierta el signo de la velocidad y
        v0 = -e*v(i);    %reduzca ésta mediante el coeficiente de
        z0 = 0;          %restitución.  Añada los tiempos de impacto al vector.
        t0 = t(i);       %t0 es ahora el último tiempo de impacto.
        impact(j) = t(i);
    end
end

plot(t,z,t,v),title('Ball Drop'),xlabel('t'),ylabel('height/velocity')
```

```
>> [z,v,im] = drop(10,0,20);
```



```
>> im(1:10)'

ans =

    1.4300
    3.7200
    5.5600
    7.0400
    8.2300
    9.1900
    9.9700
   10.6100
   11.1300
   11.5500
```

He encontrado mis 10 aproximaciones para el tiempo de impacto, lo cual será de utilidad para los apartados B3 en adelante.

B3) se supone que el primer tiempo de colisión se da en 1.43. En ese momento, la pelota vuelve a rebotar hacia arriba desde el suelo. A continuación, utilizaré la función fzero para hallar t1 de forma más precisa. Debería funcionar con un filename de función como argumento, aunque aquí simplemente tendremos que introducir la dinámica como una función en línea (inline).

```
>> fzero(inline('-.5*9.8*t.^2+10'),1.43)

ans =

    1.4286
```

B4) la velocidad en t1=1.4286 is -9.8*1.4286 = -14m/s.

B5) el coeficiente de restitución conforma la velocidad de salida = 14*.8 = 11.2m/s. También podemos observar esto en el diagrama anterior.

B6) si ejecutamos de nuevo el proegrama con las nuevas condiciones iniciales de z0 = 0 y v0 = 11.2, hallamos que el siguiente tiempo de impacto es en 3.7186 seg, tal como se predijo anteirormente.

```
>> [z,v,im]=drop(0,11.2,20);
>> im(1)+1.4286

ans =

    3.7186
```

y realizando el cálculo correctamente:
```
>> fzero(inline('-.5*9.8*t.^2+11.2*t'),2.3)

ans =

    2.2857

>> ans+1.4286

ans =

    3.7143
```

la velocidad en este tiempo es -9.8*2.2857 + 11.2 = -11.1999 m/s. Es bueno ver que se ha conservado la energía ... después del rebote, v = 0.8*11.2 = 8.96 m/s. y así sucesivamente en todos los rebotes. ¡Listo! Por fin, gracias a Dios.