Homework_2_soln

September 28, 2017

1 PHYS-330 - Classical Mechanics - Fall 2017

1.1 Homework 2

Due: 14th Sept 2017 by the start of class. Anything later will be considered late. Instructions: Complete all of the questions below. You are encouraged to use Jupyter Notebooks to complete any numerical work and written. While the use of python is encouraged, you can use any programming language you want. You can either email me your assignment or provide me with a hard copy in class.

1. 2.11 from Taylor
2. 2.21 from Taylor
3. Consider a projectile (near the earth's surface) such that $|\mathbf{v}_0| = 1$ at an angle of θ from the horizontal.

    a. what is the maximum range of the projectile (ignoreing any drag)?

    b. Now consider the same projectile which is now subjected to linear drag. What is the range (exact to 5 decimal places) of the projectile if $\theta = \pi/4$? (choose units such that $g = 1$, $v_0 = 1$ and $v_{\text{ter}} = 1$)

    c. determine the angle (exact to at least 4 decimal places) of maximum range for the projectile in the previous question.
4. 2.42 from Taylor
5. A particle of mass m and charge q is subject to a non-uniform magnetostatic field

$$\mathbf{B} = (0, -\alpha y, 1 + \alpha z)B,$$

    where α and B are positive constants.

    a. Show that equations of motion are

$$\frac{dX}{d\tau} - (1+Z)Y = D$$

$$\frac{d^2Y}{d\tau^2} + (1+Z)^2 Y + D(1+Z) = 0$$

$$\frac{d^2Z}{d\tau^2} + (1+Z)Y^2 + DY = 0$$

where D is a constant and $(X, Y, Z) = (\alpha x, \alpha y, \alpha z)$ and $\tau = \dfrac{qBt}{m} = \omega t$.

b. Consider the initial conditions at $t = 0$ as $(X, Y, Z)_0 = (0, Y_0, 0)$ and $\mathbf{V}_0 = (U_1, 0, U_3)$ where $U_i = \alpha u_i/\omega$ are dimensionless velocities. Take $Y_0 = 0.66$, $U_1 = 0.02$, and $U_3 = 0.08$. Numerically solve the system of differential equations and make plots of $X(\tau)$, $Y(\tau)$ and $Z(\tau)$. Also plot $Y(Z)$ and $Y(X)$. (Note: $D = U_1 - Y_0 = -0.64$)

1.2 Answer to 1

a. you should get the equations

$$v_y = -v_{ter} + (v_0 + v_{ter})e^{-t/\tau} \quad \text{and} \quad v(t) = -v_{ter}t + (v_0 + v_{ter})\tau(1 - e^{-t/\tau})$$

b. to get the max height we set $v_y = 0$ and solve for t. This gives a time

$$t_m = \tau \ln(1 + v_0/v_{ter})$$

and thus $y_{\text{max}} = y(t_m) = v_0\tau - v_{ter}\tau \ln(1 + v_0/v_{ter})$
c. if $v_{ter} \to \infty$ then $y_{\text{max}} \to v_0^2/(2g)$

1.3 Answer to 2

```
In [8]: import numpy as np
        from matplotlib import pyplot as plt
        from pylab import rcParams
        rcParams['figure.figsize'] = 8, 8

        def question2(theta,rho):
            g = 9.81
            v02 = 4
            return rho*np.tan(theta)- (g*rho**2)*(1/(2*v02))*(1/np.cos(theta))
        g = 9.81
        theta = np.pi/6
```

2.21 ⋆⋆⋆ The problem is axially symmetric about the z axis, so we can focus on any one direction ϕ. To be definite, let's take $\phi = 0$. If the gun is fired at an angle θ above the horizontal, then its trajectory is

$$z = v_o t \sin\theta - \tfrac{1}{2} g t^2 \quad \text{and} \quad \rho = v_o t \cos\theta.$$

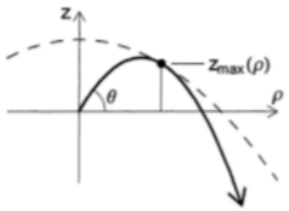
Solving the second equation for t and substituting into the first, we get the equation of the trajectory:

$$z = \rho \tan\theta - \frac{g\rho^2}{2v_o^2} \sec^2\theta.$$

This is the solid curve in the figure. For any distance ρ from the gun, the highest the shell can go, $z_{\max}$, is found by differentiating z with respect to θ and finding where $\partial z/\partial\theta = 0$. This gives $\tan\theta = v_o^2/g\rho$. (You can easily check that $\partial^2 z/\partial\theta^2 < 0$, so this is a maximum of z.) Substituting into the expression for z, we find

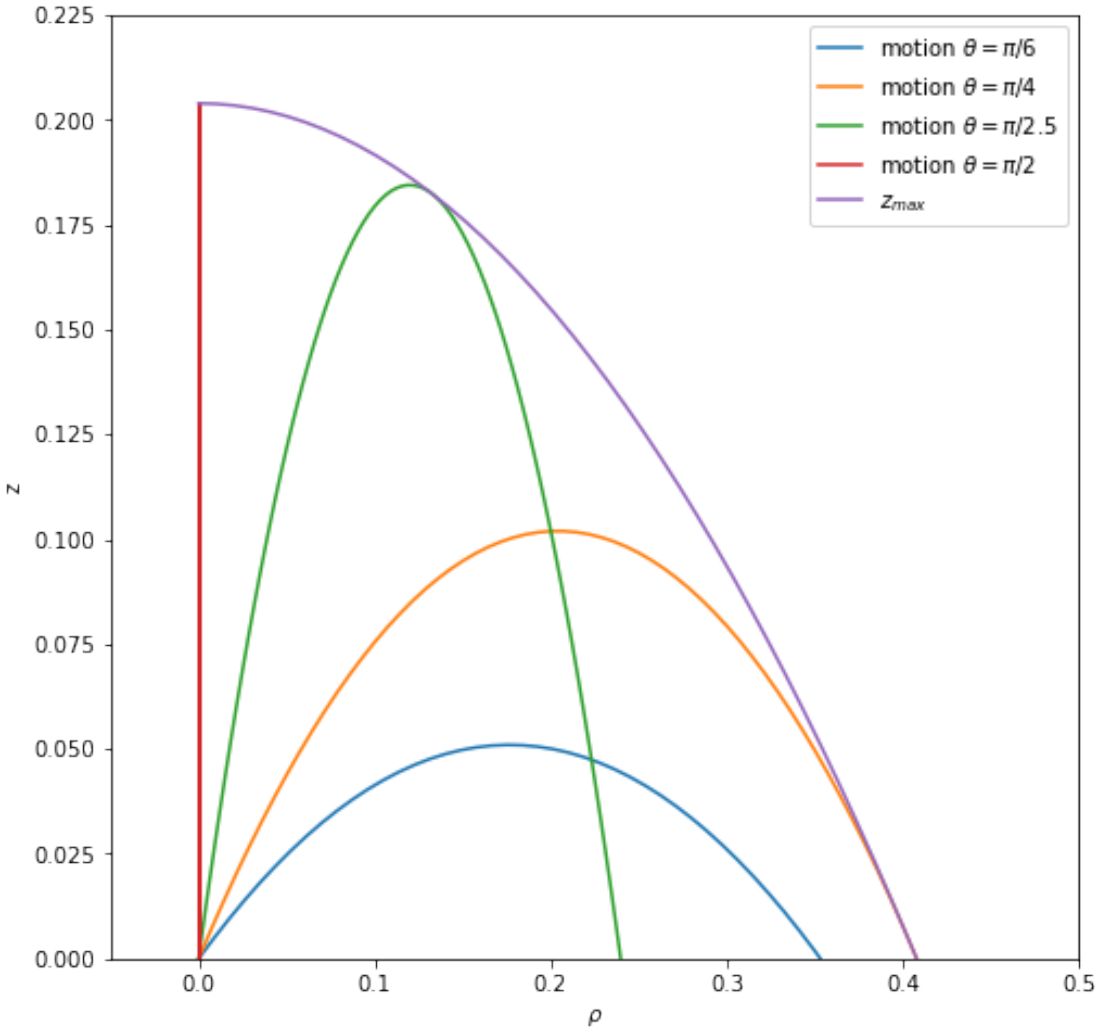
$$z_{\max}(\rho) = \frac{v_o^2}{2g} - \frac{g}{2v_o^2}\rho^2.$$

This is the highest the shell can reach at the distance ρ. The locus of $z_{\max}$ is the dashed parabola shown. By changing the angle θ we can get the shell to pass through any point lower than $z_{\max}$; therefore, all points below the dashed curve are accessible to the gun. When the parabola is rotated about the z axis we get a paraboloid of revolution. Its vertex is at $z = v_o^2/2g$ (well known as the maximum height of a projectile launched vertically up) and its radius in the plane $z = 0$ is $\rho = v_o^2/g$ (well known to be the maximum horizontal range of a gun with muzzle speed v_o).



title

```
v02 = 2
g1 = 9.81
t11 = np.linspace(0,.5,100)
plt.plot(v02*t11*np.cos(theta),v02*t11*np.sin(theta)-.5*g1*t11**2,label="mo
theta = np.pi/4
plt.plot(v02*t11*np.cos(theta),v02*t11*np.sin(theta)-.5*g1*t11**2, label="m
theta = np.pi/2.5
plt.plot(v02*t11*np.cos(theta),v02*t11*np.sin(theta)-.5*g1*t11**2, label="m
theta = np.pi/2
plt.plot(v02*t11*np.cos(theta),v02*t11*np.sin(theta)-.5*g1*t11**2,label="mo
rho = np.linspace(0,1,100)
plt.plot(rho,v02**2/(2*g)-(g/(2*v02**2))*rho**2,label='$z_{max}$')
plt.axis([-0.05,.5,0,.225])
plt.legend()
plt.xlabel("$\\rho$")
plt.ylabel("z")
plt.show()
```



the curve $z_{\max}$ is rotated around the z axis which creates a paraboloid. The cannon can hit any point within the paraboloid.

1.4 Answer to 5

```
In [10]: import numpy as np
         from matplotlib import pyplot as plt
         from scipy.integrate import odeint
         from pylab import rcParams
         rcParams['figure.figsize'] = 8, 10

         def problem5(x,t):
             #function is the system of equations listed in problem
             D=-0.64
             z1 = (1+x[2])
             q1 = z1*x[1]+D
```

We remark that the field (1) satisfies $\boldsymbol{\nabla} \cdot \mathbf{B} = 0$ (as any magnetic field must) and $\boldsymbol{\nabla} \times \mathbf{B} = 0$ (which is required of static fields in source-free regions). Note also that α^{-1} is a length scale.

(a) For the field (1), the equation of motion $m\ddot{\mathbf{r}} = q\dot{\mathbf{r}} \times \mathbf{B}$ has components

$$\ddot{x} = \omega\left\{\dot{y} + \alpha\frac{d}{dt}(yz)\right\} \tag{8}$$

$$\ddot{y} = -\omega(1+\alpha z)\dot{x} \tag{9}$$

$$\ddot{z} = -\omega\alpha y\dot{x}\,, \tag{10}$$

where $\omega = qB/m$ is the cyclotron frequency for the uniform field $(0, 0, B)$ corresponding to $\alpha = 0$. Integration of (8) gives

$$\dot{x} = \omega(1+\alpha z)y + D_0\,, \tag{11}$$

where D_0 is a constant of integration. By substituting (11) in (9) and (10) we obtain two coupled equations involving y and z only:

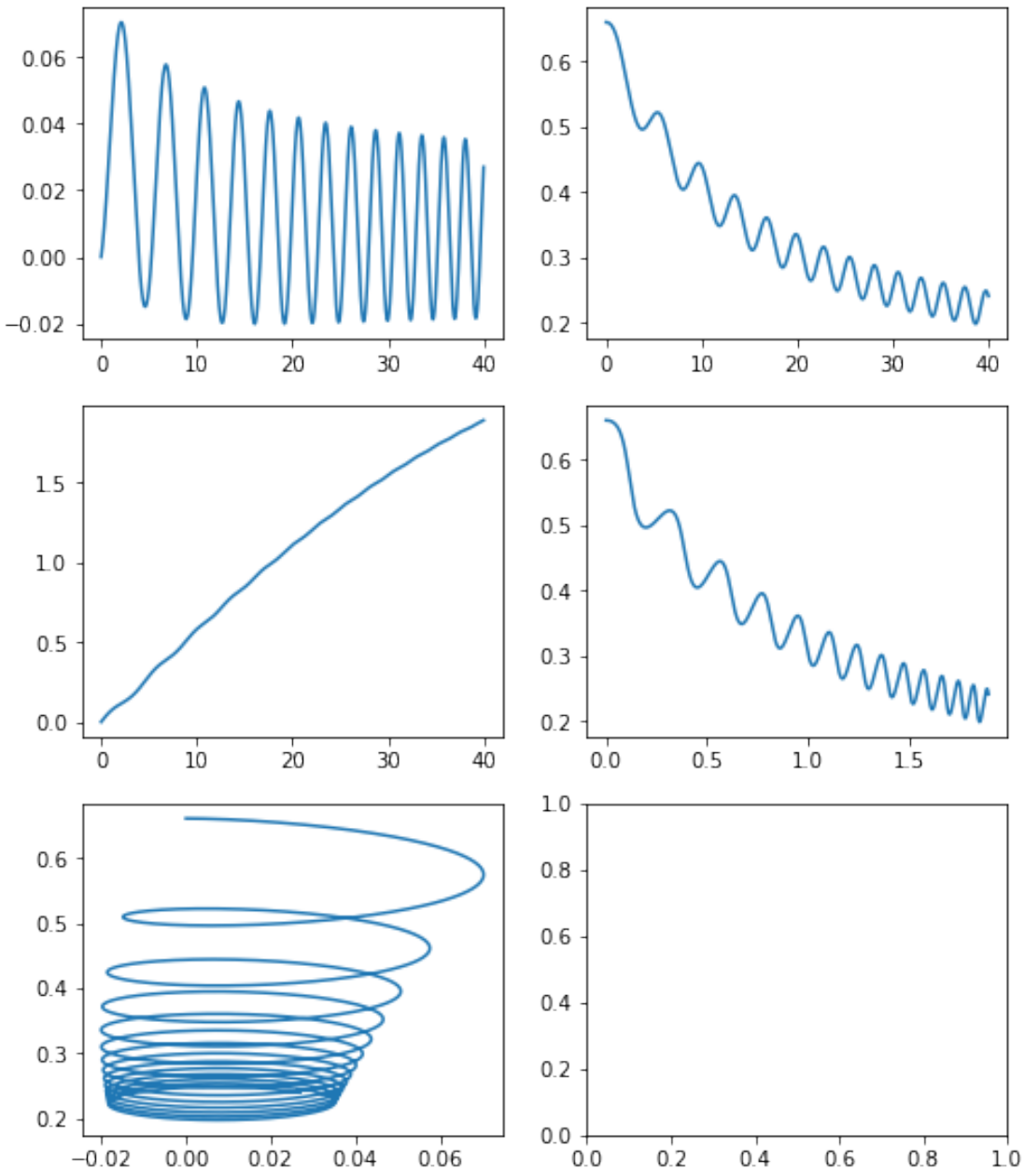
$$\ddot{y} = -\omega^2(1+\alpha z)^2 y - \omega D_0(1+\alpha z) \tag{12}$$

$$\ddot{z} = -\omega^2\alpha(1+\alpha z)y^2 - \omega D_0\alpha y\,. \tag{13}$$

If we multiply (11) by α/ω and (12) and (13) by α/ω^2, and use (5), we obtain the dimensionless forms (2)–(4) with $D = \alpha D_0/\omega$.

title

```
    q2 = x[3]
    q3 = x[4]

    p1 = -D*z1-z1**2*x[1]
    p2 = -D*x[1]-z1*x[1]**2
    return [q1,q2,q3,p1,p2]

#set up initial conditions
x0 = [0,0.66,0,0,0.08]
#set time intervals
t0 = np.linspace(0,40.0,10000)

#solve system
soln = odeint(problem5,x0,t0)

f, axarr = plt.subplots(3,2)
axarr[0,0].plot(t0,soln[:,0])
axarr[0,1].plot(t0,soln[:,1])
axarr[1,0].plot(t0,soln[:,2])
axarr[1,1].plot(soln[:,2],soln[:,1])
axarr[2,0].plot(soln[:,0],soln[:,1])
#plt.plot(t0,soln[:,4],lw=.5)
plt.show()
```



1.5 Answer to 3

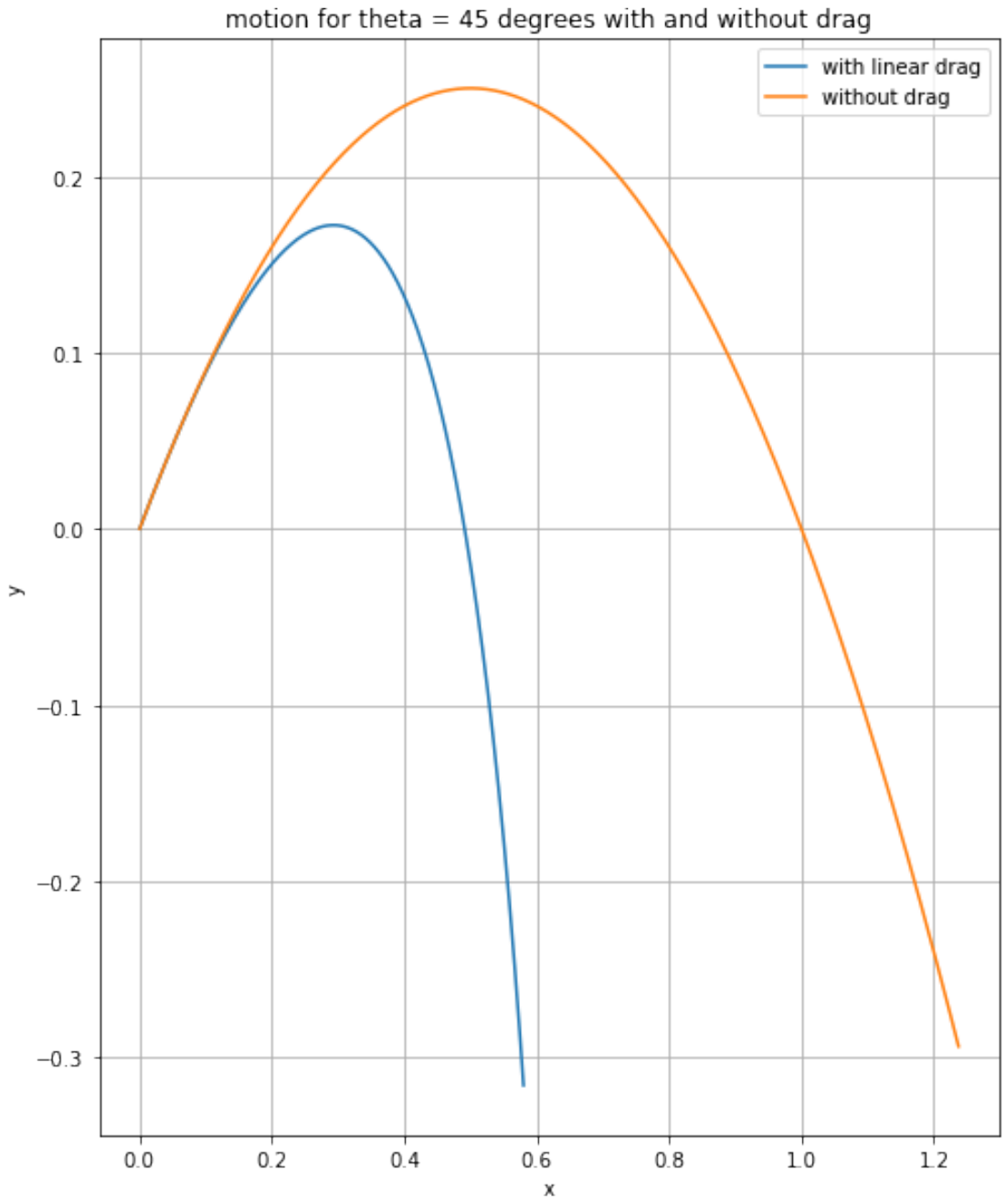
(a) Maximum range for the projectile without any drag forces is determined by

$$R_{\text{no drag}} = v_{0x}\tau$$

and for our units $R_{\text{no drag}} = 1$

```
In [11]: # Define the pre-defined variables
         vter = 1.0  #terminal velocity
```

```
g = 1.0      # acceleration due to gravity
v0 = 1.0     # initial speed
theta = np.pi/4 #initial angle
tau = vter/g      # tau

#define x and y components of velocity
v0x = v0*np.cos(theta)
v0y = v0*np.sin(theta)

# x displacement in vaccuum
Rvac = v0x*tau

x = np.linspace(0,.58,1000)
y = x*(v0y + vter)/v0x + vter*tau*np.log(1.0-x/(v0x*tau))

t = np.linspace(0,1.75,1000)
xvac = v0x*t
yvac = v0y*t-0.5*g*t**2

#plt.axis([.48,.53,.052,-.052])
plt.plot(x,y,label="with linear drag")
plt.plot(xvac,yvac,label="without drag")
plt.grid()
plt.title("motion for theta = 45 degrees with and without drag")
plt.xlabel("x")
plt.ylabel("y")
plt.legend()
plt.show()
```



In [12]:
```
# the scipy.optimize library has a newton's method i can use to find a roo
import scipy.optimize as sci

def func(x):
    #this is the function we want to find a root of (eqn 2.39 in book)
    return x*(v0y+vter)/v0x + vter*tau*np.log(1-x/(v0x*tau))
```

```
        #now we use newton's method
        print("X displacement = ",sci.newton(func,x0=.42,tol=1.0e-12))
```

```
X displacement =  0.490986359267
```

```
In [13]: def range(theta):
             # this function returns the range  of the projectile for a given theta
             vter = 1.0  # set units
             g = 1.0     # given in problem
             v0 = 1.0    # initial velocity

             def func(x): #this function return value of equation 2.39 in book
                 return x*(v0y + vter)/v0x + vter*tau*np.log(1.0-x/(v0x*tau))#you r

             tau = vter/g
             v0x = v0*np.cos(theta) # x component of velocity
             v0y = v0*np.sin(theta) # y component of velocity
             return sci.newton(func,x0=.5,tol=1.0e-12)  #return root of equation 2.


         #below is a simple script which is reminicient of a binomial search
         thetamax = np.pi # we know theta can't be this big to maximize R
         thetamin = 0.0   # initializing theta
         step = .01
         flag = True      # change this to exit loop when we have found R
         R = 0             # initial range.This will change as we find larger R
         while flag == True:
                 theta = thetamin + step  # update to new theta to test
                 Rnew = range(theta)      # calculate range for current theta
                 if Rnew < R:             # if xnew is less than current range
                     step = step/2.0           # decrease step size
                 else:                    # otherwise check rel error from previous
                     if np.abs(theta-thetamin)/theta < 1.0e-10: # if relative error
                         print("theta for max displacement =",theta, "rad")
                         flag = False
                     else:     # otherwise update thetamin and new range
                         thetamin = theta
                         R = Rnew
```

```
theta for max displacement = 0.6212500000002913 rad
```

2 Answer to 4

If we continue to measure y upward as in Problem 2.41, the equation of motion for the downward journey is $m\dot{v} = -mg + cv^2 = -mg(1 - v^2/v_{\text{ter}}^2)$. Using the "$vdv/dx$" rule (again as in Problem 2.41), this becomes $d(v^2)/(1 - v^2/v_{\text{ter}}^2) = -2\,g\,dy$. If we integrate this from the top ($v = 0$ and $y = y_{\max}$) to an arbitrary position (velocity v and height y), we get

$$-v_{\text{ter}}^2 \ln(1 - v^2/v_{\text{ter}}^2) = -2g(y - y_{\max})$$

whence

$$v = -v_{\text{ter}}\sqrt{1 - e^{2g(y-y_{\max})/v_{\text{ter}}^2}}.$$

(Note that I took the negative square root since the velocity is downward.) Here $y_{\max}$ is given by Eq.(2.89) and, when the ball hits the ground, $y = 0$. Thus the velocity just before it hits the ground is $v = -v_{\text{ter}}\sqrt{1 - v_{\text{ter}}^2/(v_{\text{ter}}^2 + v_{\text{o}}^2)} = -v_{\text{ter}}v_{\text{o}}/\sqrt{(v_{\text{ter}}^2 + v_{\text{o}}^2)}$, as claimed. Putting in the numbers, we find that $v = -17.4$ m/s at landing. In a vacuum the speed at landing would be the same as that at launch, so v would be -20 m/s.

title