1 Newton’s 2nd and 3rd Laws

1. Work through the tutorial called “Newton’s Second and Third Laws” on pages 31-34 in the UW “Tutorials in Introductory Physics” workbook.

2. Experiment with the momentum principle (Newton’s 2nd Law) by working through an exercise developed by the Matter and Interaction Group.

   (a) Open the program “01_newton.py” in the VPython IDLE editor. You should find this program in the p13_lab2 folder on the Desktop.

   (b) Run the program.

   (c) In the right window, hold down the left mouse button and drag a force vector. The ball in the left window moves under the influence of this single force. (The ball could be in outer space, with only a single force applied to it, or there could be several forces, including frictional forces, and the force vector represents the net force being applied.) The green arrow is the net force; the red arrow is the momentum of the ball.

   (d) Try the following exercises:

      • Make the ball move due to a constant force. What happens to the momentum?

      • Make the ball move with a constant momentum. What force is required?

      • What do you have to do to make the ball stop, then go in the opposite direction?

      • Make an example where there is a big momentum but zero force.
      • Make an example where the momentum is momentarily zero, but there is a big force.
      • Make and example where the momentum is in the opposite direction to the force.

STOP!

Checkpoint 1: Based on your experience with the examples above, discuss the following question with your partner and then your TA: Does the momentum relate directly to the force?
2 Motion with Constant Force

Now we will see how to numerically calculate the trajectory of a ball subjected to constant force.

2.1 Updating Momentum

In lab 1, you learned how to animate a ball moving at constant velocity using the position update formula. Now we’ll subject the ball to a constant force. In this case, the momentum of the ball will change, thus its velocity will change with each timestep. To accurately update the position of the ball at each timestep, we need to use the updated momentum for each timestep.

Recall the following form of the Momentum Principle

\[ \vec{p}_f = \vec{p}_i + \vec{F}_{net} \Delta t \]

This equation tells us how to calculate the new momentum of an object at time \( t + \Delta t \) if we know its initial momentum (at time \( t \)) and the net force acting on it. Let’s consider a 0.25 kg ball subjected to a constant force \( \vec{F}_{net} = (-0.4, 0, 0) \) N.

1. What is the change in momentum of the ball in the y-direction?

2. Assume the ball has initial momentum \( \vec{p}_i = (1,0,0) \) kgm/s. What is the momentum of the ball after 0.01 s?

3. What is the momentum of the ball 0.01 seconds later?

We can keep calculating the new momentum after each short time interval to find the momentum at later and later times. However, as we’ll see below, it’s easier to let the computer do so many calculations, so we’ll stop here for the moment.

2.2 Updating Position

Now that we know how to calculate the momentum of the ball at successive times, we can use it to update the position of the ball. Recall the position update formula (which comes from the definition of velocity):

\[ \vec{r}_f = \vec{r}_i + \vec{v} \Delta t \]

Using \( \vec{v} = \vec{p}/m \) for the velocity

\[ \vec{r}_f = \vec{r}_i + (\vec{p}/m) \Delta t \]
1. Assume the ball is at an initial position $\vec{r} = (-5,0,0)$ at time $t=0$. Use the initial velocity of the ball to find its position after 0.01 s. Record your answer here.

2. Use the momentum at time $t = .01$s calculated above to find the position of the ball at time $t = 0.02$ s (0.01 s later).

STOP!
Checkpoint 2: Check your answers with your TA before proceeding.

2.3 Modeling Motion with VPython

Every program that models the motion of physical objects has two main parts.

1. **Before the loop:** The first part of the program tells the computer to:
   
   (a) Create numerical values for constants we might need.
   
   (b) Create 3D objects.
   
   (c) Give the objects initial positions and momenta.

2. **The “while” loop:** The second part of the program, the loop, contains the statements that the computer reads to tell it how to increment the position of the objects over and over again, making them move on screen. These statements tell the computer:
   
   (a) How to calculate the net force on the objects.
   
   (b) How to calculate the new momentum of each object, using the net force and the momentum principle.
   
   (c) How to find the new positions of the objects, using the momenta.

Now we will use the computer to carry out the calculations you did by hand in Sections 2.1 and 2.2. We’ll also see how the ball moves under constant force. Note: The directions below don’t explicitly tell you to run the program after each step! However, you might want to run your program occasionally or after each change you make in it, so you can make sure the program runs without errors and exhibits the behavior you expect.

- Start a new vpython program. Remember to save your program and include the following statement on line 1
  
  from visual import *

- **Before the loop**
  
  - Create a ball object at position (-5,0,0). You may refer to your Lab 1 program or the Lab 1 write-up to remind you how to do this.
– Give your ball a mass. On the next line, type:
  \( m = 0.25 \)

– Add a wall to the right side of the simulation as you did in Lab 1. This will provide
  a reference for which direction the ball is moving.
  \( \text{wallR} = \text{box}(pos=\text{vector}(6,0,0), \text{size}=(.2,12,12), \text{color}=\text{color.green}) \)

– Create your “timestep” parameter, \( dt=0.01 \)
– Initialize the momentum of the ball (specify the initial momentum at time \( t=0 \))
  \( \text{ball.momentum} = \text{vector}(1,0,0) \)

• The while loop

  – Start the “while” loop remembering to include the colon
    \( \text{while } (1==1): \)
  
  – As is Lab 1, you’ll want to slow down the rate of the program. On the next line
    type:
    \( \text{rate}(50) \)
    Make sure this line (and all statements inside the loop) are indented. This is what
    tells the computer the statements are inside the loop.
  
  – First, let’s animate the ball by updating its position assuming the momentum is
    constant. We’ll follow the procedure used in the “Ball in a box” program.
    \( \text{ball.pos} = \text{ball.pos} + (\text{ball.momentum}/m)\ast dt \)

  – Now, let’s subject the ball to constant force and see what happens
    * Specify the force acting on the ball by defining a vector force. We’ll use the
      same force used in Section 2.1. Enter the following line before the statement
      that updates the ball position.
      \( \text{Fnet} = \text{vector}(-0.4,0,0) \)
      (Note: In this program, the force is constant so we could define it outside the
      loop if we wanted. It has the same value every time the computer runs through
      the loop. However, in the future, we may want to use a force that varies, so
      we’ll define it inside the loop in order to leave open this possibility.)
    * \( \text{Before} \) the statement that updates the ball position, insert a statement to up-
      date the momentum of the ball. Use the momentum update equation to do this
      and ask your TA if you need help.

  – Run your program. What happens?
– Sketch a position versus time plot for the ball (If you want, you can add a statement to make the ball leave a trail as you did in Lab 1). Sketch a velocity versus time plot.

– Compare the computer calculations with those you did by hand. Enter the following line after the position update statement

\texttt{print ball.pos}

Run your program and stop it quickly. Scroll up in the Python shell window to find where it prints the first couple of positions for the ball. Compare with your hand calculations. Do the same thing for the momentum.

\textbf{STOP!}

\textbf{Checkpoint 3:} Check your program and plots with your TA before proceeding.

* You can now experiment by making some changes to your program. For example, you might try to give the ball an initial y-component for its velocity. Or add a y-component to the force!

– First, add arrows to the display that show the momentum of the ball and force acting on it. Refer to Lab 1 for help if you need it.

– Pick two variations (anything you want), make each change, and explain what happens. Only make one change at a time (so you should have \textit{two} explanations below)!
3 Non-Constant Forces

3.1 Simple Harmonic Motion

- Change the net force inside the while loop so that the x-component of the force is proportional to the negative of the x-position of the ball. What happens and why?

- How does the velocity of the ball vary in time?

- How does the period of oscillation of the ball change if you increase the proportionality constant?

3.2 Circular Motion

- Now we will investigate what happens when the direction of the force changes in time.

- How would you define a force that is always perpendicular to the direction of the momentum of the ball? Hint: use the momentum of the ball when you define the force. You will want a force that points in the +x-direction when the ball moves in the +y-direction and a force that points in the -y-direction when the ball moves in the +x-direction, etc.

- Try it! You should see the ball move in a circular orbit. List two examples for which the force is always perpendicular to the motion.