Από το Βιβλίο Matter and Interactions
Ruth W. Chabay, Bruce A. Sherwood
4th ed. Wiley
2.6 σελ. 65 – 72
2.7 σελ. 72 – 75

Η μέθοδος Euler-Kromer
(αριθμητική επίλυση της διαφορικής εξίσωσης κίνησης $\Delta\vec{p} = \vec{F}_{net}\Delta t$)
Παράδειγμα: δύναμη ελατηρίου

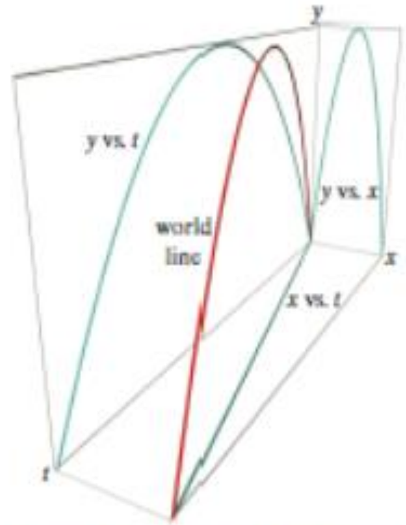

Figure 2.32 A "world line" and its projections.

Projections of a "World Line"

Figure 2.32 illustrates an interesting way to think of the graphs of y vs. x, y vs. t, and x vs. t. The red curve in Figure 2.32 is a graphical description of the motion in x, y, and t of a ball that moves in the xy plane. Imagine shining a light on the red curve to project a shadow onto the xy plane, or the yt plane, or the xt plane. These "projections" are shown in cyan in the figure. Compare these projections with the individual graphs shown in Figures 2.29, 2.30, and 2.31.

The red curve, a function of x, y, and t, is called a "world line." World lines play an important role in textbooks on special relativity. In reality, a world line should be a function of x, y, z, and t, but it's very difficult even with a computer to draw a four-dimensional curve!

2.6 ITERATIVE PREDICTION: VARYING NET FORCE

A familiar example of a force that is not constant is the force exerted by a spring that is stretched or compressed. The magnitude of this force depends on the amount of stretch or compression of the spring. The force is directed along the line of the spring.

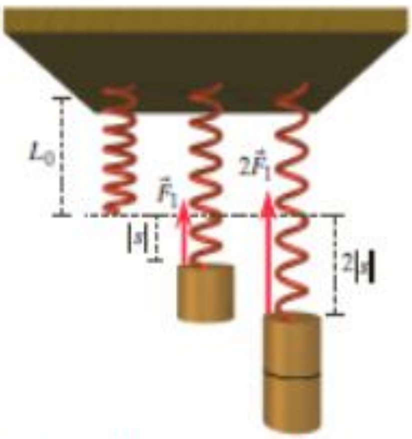

Figure 2.33 The magnitude of the force exerted by a spring is proportional to the absolute value of the stretch of the spring. For an elongated spring, stretch is positive.

The Spring Force

A "force law" describes mathematically how a force depends on the situation. For a spring, it is determined experimentally that the force exerted by a spring on an object attached to the spring is given by the following equation:

MAGNITUDE OF THE SPRING FORCE

$$|\vec{F}_{spring}| = k_s|s|$$

$|s|$ is the absolute value of the stretch: $s = L - L_0$.

L_0 is the length of the relaxed spring.

L is the length of the spring when stretched or compressed.

k_s is the "spring stiffness" (also called "spring constant").

The force acts in a direction to restore the spring to its relaxed length.

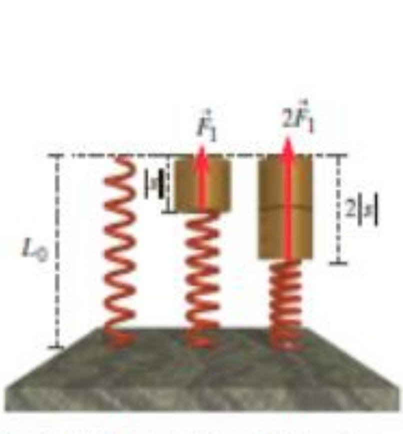

Figure 2.34 The magnitude of the force exerted by a spring is proportional to the absolute value of the stretch of the spring. For a compressed spring, stretch is negative.

The constant k_s is a positive number, and is a property of the particular spring: the stiffer the spring, the larger the spring stiffness, and the larger the force needed to stretch the spring. Note that s is positive if the spring is stretched ($L > L_0$, Figure 2.33) and negative if the spring is compressed ($L < L_0$, Figure 2.34). This equation is sometimes called "Hooke's law." It is valid as long as the spring is not stretched or compressed too much.

QUESTION Suppose a certain spring has been calibrated so that we know that its spring stiffness k_s is 500 N/m. You pull on the spring and observe that it is 0.01 m (1 cm) longer than it was when relaxed. What is the magnitude of the force exerted by the spring on your hand?

The force law gives $|\vec{F}_{spring}| = (500\ \text{N/m})(|+0.01\ \text{m}|) = 5\ \text{N}$. Note that the total length of the spring doesn't matter; it's just the amount of stretch or compression that matters. Because of the "reciprocity" of the electric forces between the protons and electrons in the spring and those in your hand, as we'll study in Chapter 3, the force you exert on the spring is equal in magnitude and opposite in direction to the force the spring exerts on your hand.

QUESTION Suppose that instead of pulling on the spring, you push on it, so the spring becomes shorter than its relaxed length. If the relaxed length of the spring is 10 cm, and you compress the spring to a length of 9 cm, what is the magnitude of the force exerted by the spring on your hand?

The stretch of the spring in SI units is

$$s = L - L_0 = (0.09\,\text{m} - 0.10\,\text{m}) = -0.01\,\text{m}$$

The force law gives

$$|\vec{F}_{\text{spring}}| = (500\,\text{N/m})(|-0.01\,\text{m}|) = 5\,\text{N}$$

The magnitude of the force is the same as in the previous case. Of course the direction of the force exerted by the spring on your hand is now different, so we will need to write a full vector equation to incorporate this information.

Checkpoint 6 (1) You push on a spring whose stiffness is 11 N/m, compressing it until it is 2.5 cm shorter than its relaxed length. What is the magnitude of the force the spring now exerts on your hand? (2) A different spring is 0.17 m long when it is relaxed. **(a)** When a force of magnitude 250 N is applied, the spring becomes 0.24 m long. What is the stiffness of this spring? **(b)** This spring is compressed so that its length is 0.15 m. What magnitude of force is required to do this?

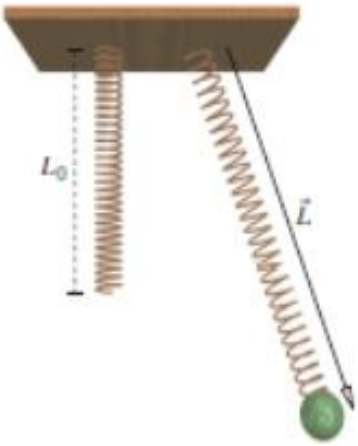

Figure 2.35 The vector $\vec{L}$ points from the location where the spring is attached to the ceiling to the location of the mass. The arrow is offset for clarity in the diagram. L_0 is the length of the relaxed spring.

The Spring Force as a Vector

To write an equation for the spring force as a vector, combining magnitude and direction into one expression, we need to consider the general case, in which the spring may be stretched both vertically and horizontally. Since the force exerted by the spring will be directed along the axis of the spring, we need a unit vector in that direction. We will get this unit vector from the vector $\vec{L}$, shown in Figure 2.35, which is a relative position vector: it specifies the position of the movable end of the spring relative to the fixed end of the spring. $\vec{L}$ extends from the point at which the spring is attached to a support to the mass at the other end of the spring. We can factor the relative position vector $\vec{L}$ into the product of a magnitude (the length of the spring) and a unit vector:

$$\vec{L} = |\vec{L}|\hat{L}$$

The stretch of the spring can now be written as

$$s = |\vec{L}| - L_0$$

- When the spring is longer than its relaxed length, the value of s will be positive, but the direction of the force will be given by $-\hat{L}$.
- When the spring is shorter than its relaxed length, the value of s will be negative, but the direction of the force will be given by $+\hat{L}$.

The general expression for the vector spring force is therefore:

THE VECTOR SPRING FORCE

$$\vec{F} = -k_s s\hat{L}$$

The stretch $s = |\vec{L}| - L_0$ and may be positive or negative.
The scalar L_0 is the length of the relaxed spring.
The vector $\vec{L}$ extends from the point of attachment of the spring to the movable end.
k_s is the "spring stiffness" (also called "spring constant").

QUESTION Consider the two stretched springs in Figure 2.33. What is the unit vector $\hat{L}$ for each stretched spring (assuming the usual coordinate system)? Does $\hat{L}$ depend on how much the spring is stretched?

In each case $\hat{L} = \langle 0,-1,0\rangle$. Although the magnitude of $\vec{L}$ depends on the stretch, the unit vector $\hat{L}$, which has magnitude 1, is the same in both cases. Similarly, for both compressed springs in Figure 2.34, $\hat{L} = \langle 0,1,0\rangle$.

EXAMPLE **Force Due to a Stretched Spring**

Suppose that the stiffness of the rightmost spring shown in Figure 2.35 is 9 N/m, and its relaxed length is 21 cm. At the instant shown, the location of the green mass is $\langle 0.07,-0.33,0\rangle$ m relative to an origin at the point of attachment of the spring. What is the force exerted by the spring on the green mass at this instant?

Solution

$$\vec{L} = \langle 0.07,-0.33,0\rangle\ \text{m} - \langle 0,0,0\rangle\ \text{m} = \langle 0.07,-0.33,0\rangle\ \text{m}$$

$$|\vec{L}| = \sqrt{(0.07\ \text{m})^2 + (-.33\ \text{m})^2} = 0.337\ \text{m}$$

$$\hat{L} = \frac{\langle 0.07,-0.33,0\rangle\ \text{m}}{0.337\ \text{m}} = \langle 0.208,-.979,0\rangle$$

$$s = 0.337\ \text{m} - 0.21\ \text{m} = 0.127\ \text{m}$$

$$\vec{F} = -(9\ \text{N/m})(0.127\ \text{m})\langle 0.208,-.979,0\rangle = \langle -0.238, 1.12, 0\rangle\ \text{N/m}$$

Check: The x component of the force is negative, and the y component is positive, as they should be.

Checkpoint 7 (1) A spring of stiffness 13 N/m, with relaxed length 20 cm, stands vertically on a table as shown in Figure 2.36. Use the usual coordinate system, with $+x$ to the right, $+y$ up, and $+z$ out of the page, toward you. **(a)** When the spring is compressed to a length of 13 cm, what is the unit vector $\hat{L}$? **(b)** When the spring is stretched to a length of 24 cm, what is the unit vector $\hat{L}$? (2) A different spring of stiffness 95 N/m, and with relaxed length 15 cm, stands vertically on a table, as shown in Figure 2.36. With your hand you push straight down on the spring until your hand is only 11 cm above the table. Find **(a)** the vector $\vec{L}$, **(b)** the magnitude of $\vec{L}$, **(c)** the unit vector $\hat{L}$, **(d)** the stretch s, **(e)** the force $\vec{F}$ exerted on your hand by the spring.

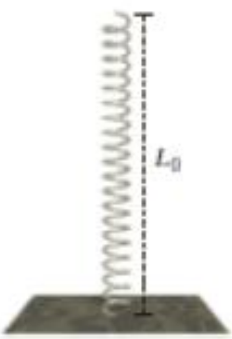

Figure 2.36 A relaxed vertical spring. The tabletop lies in the xz plane, and y is up, as usual.

Motion of a Block–Spring System

If we attach a block to the top of a spring, push down on the block, and then release it, the block will oscillate up and down. (This repetitive motion is described as "periodic.") As the spring stretches and compresses, the force exerted on the block by the spring changes in magnitude and direction. There is also a constant gravitational force on the block. Because the net force on the block is continually changing, we can't use a one-step calculation to predict its motion. We need to apply the Momentum Principle iteratively to predict the location and velocity of the block at any instant.

QUESTION Can we use the equations we derived for the constant force situation?

No, those equations would give us the wrong answer here, because the net force on the block is not constant. We need to solve the problem iteratively, using the same procedure we used to find the path of a ball.

We need two initial values:

- The initial position of the block
- The initial momentum of the block

We will do the following calculation iteratively (repeatedly):

- Find the net force $\vec{F}_{\text{net,now}}$ on the block.
- Apply the Momentum Principle to find $\vec{p}_{\text{future}}$.
- Use $(\vec{p}_{\text{future}}/m)$ to approximate the average velocity, and use this to find the new position of the block.

QUESTION Why is it important to calculate the net force for each iteration? Couldn't we just calculate it once and use this value in each iteration?

As the block moves up and down, the length of the spring changes, so the force exerted by the spring on the block is different each time we take a step.

EXAMPLE **Block on Spring: 1D, Nonconstant Net Force**

A spring has a relaxed length of 20 cm (0.2 m) and its spring stiffness is 8 N/m (Figure 2.36). You glue a 60 g block (0.06 kg) to the top of the spring, and push the block down, compressing the spring so its total length is 10 cm (Figure 2.37). You make sure the block is at rest, then you quickly move your hand away. The block begins to move upward, because the upward force on the block by the spring is greater than the downward force on the block by the Earth. Make a graph of y vs. time for the block during a 0.3 s interval after you release the block.

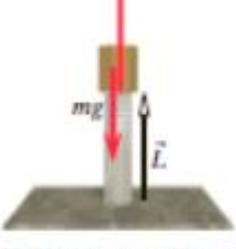

Figure 2.37 You compress the spring, make sure the block is at rest, then release the block. The red arrows show the forces on the block, due to the surroundings, at the instant just after you release the block.

Solution System: Block
Surroundings: Spring, Earth
Diagram: Figure 2.37

For convenience we place the origin at the base of the spring. We will use the shorter notation $\vec{p}_i$ and $\vec{p}_f$ rather than $\vec{p}_{\text{now}}$ and $\vec{p}_{\text{future}}$ to refer to the momentum at the beginning and end of each time step.

$$\vec{L} = \langle 0, 0.1, 0\rangle - \langle 0, 0, 0\rangle = \langle 0, 0.1, 0\rangle \text{ m}$$

$$|\vec{L}| = 0.1$$

$$\hat{L} = \langle 0, 1, 0\rangle$$

$$\vec{F}_{\text{spring}} = -k_s(|\vec{L}| - L_0)\langle 0, 1, 0\rangle = \langle 0, -k_s(|\vec{L}| - L_0), 0\rangle$$

$$\vec{F}_{\text{Earth}} = \langle 0, -mg, 0\rangle$$

The initial momentum of the block is zero, since it is at rest when you release it:

$$\vec{p}_i = \langle 0,0,0\rangle$$

The net force on the block will be the sum of the forces on the block by the Earth and by the spring. Because both the force by the spring and the gravitational force by the Earth act in the $\pm y$ direction, and the initial x and z components of the block's momentum are zero, we could consider only the y components of force, momentum, and position in our solution. However, writing out all the vectors helps considerably in avoiding sign errors in forces and momenta.

To get an approximate answer, let's divide the 0.3 s time interval into three intervals each 0.1 s long. (It would be better to use even shorter intervals, but this would be unduly tedious if done by hand.)

First time step (Figure 2.38):

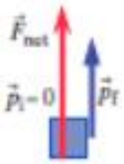

Figure 2.38 At the beginning of time step 1, the net force on the block is in the $+y$ direction, so the y component of the block's momentum will increase.

$$|\vec{L}| = 0.1\ \text{m}$$
$$s = 0.1\ \text{m} - 0.2\ \text{m} = -0.1\ \text{m}$$
$$\vec{F}_{spring} = -8\ \text{N/m}(-0.1\ \text{m})\langle 0,1,0\rangle = \langle 0,+0.8,0\rangle\ \text{N}$$
$$\vec{F}_{Earth} = \langle 0,-0.06\ \text{kg}\cdot 9.8\ \text{N/kg},0\rangle = \langle 0,-0.588,0\rangle\ \text{N}$$
$$\vec{F}_{net} = \langle 0,+0.212,0\rangle\ \text{N}$$
$$\vec{p}_f = \langle 0,0,0\rangle + \langle 0,+0.212,0\rangle\ \text{N}\ (0.1\ \text{s})\quad \text{(Momentum Principle)}$$
$$\vec{p}_f = \langle 0,+0.0212,0\rangle\ \text{kg}\cdot\text{m/s}$$
$$\vec{v}_{avg} \approx \vec{v}_f$$
$$\vec{v}_f = \frac{\vec{p}_f}{m} = \frac{\langle 0,+0.0212,0\rangle\ \text{kg}\cdot\text{m/s}}{0.06\ \text{kg}} = \langle 0,+0.353,0\rangle\ \text{m/s}$$
$$\vec{r}_f = \langle 0,0.1,0\rangle\ \text{m} + \langle 0,+0.353,0\rangle\ \text{m/s}\ (0.1\ \text{s})\quad \text{(position update)}$$
$$\vec{r}_f = \langle 0,0.135,0\rangle\ \text{m}$$

Second time step (Figure 2.39):

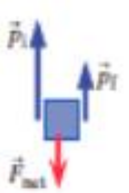

Figure 2.39 At the beginning of time step 2, the net force on the block is now in the $-y$ direction. The block will move upward, but its upward momentum will decrease.

Now we advance the clock. At the beginning of time step 2, we need to recalculate $F_{spring,y}$, because the length of the spring has changed. We find that both the magnitude and the direction of the net force are different from the values we found at the beginning of step 1. The momentum of the block at this time ("now") reflects the forces that the block experienced during step 1.

$$|\vec{L}| = 0.135\ \text{m}$$
$$s = -0.0647\ \text{m}$$
$$\vec{F}_{spring} = \langle 0,+0.520,0\rangle\ \text{N}$$
$$\vec{F}_{net} = \langle 0,-0.0707,0\rangle\ \text{N}$$
$$\vec{p}_f = \langle 0,0.0141,0\rangle\ \text{kg}\cdot\text{m/s}$$
$$\vec{r}_f = \langle 0,0.159,0\rangle\ \text{m}$$

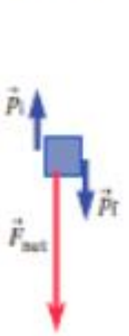

Figure 2.40 At the beginning of time step 3, the net force on the block is in the $-y$ direction, and the final momentum of the block will be downward.

QUESTION What value did we use for $\vec{p}_i$ in time step 2?

$\vec{p}_f$ from step 1 became $\vec{p}_i$ in step 2, as we advanced the clock.

Third time step (Figure 2.40):

At the beginning of the third time step we find that the net force on the block has changed again. The momentum of the block reflects the impulses it has experienced during the first two steps:

$$|\vec{L}| = 0.159 \text{ m}$$
$$s = -0.0411 \text{ m}$$
$$\vec{F}_{net} = \langle 0, -0.259, 0 \rangle \text{ N}$$
$$\vec{p}_f = \langle 0, -0.0118, 0 \rangle \text{ kg-m/s}$$
$$\vec{r}_f = \langle 0, 0.139, 0 \rangle \text{ m}$$

The graph of y vs. time for our three-step calculation is shown in Figure 2.41.

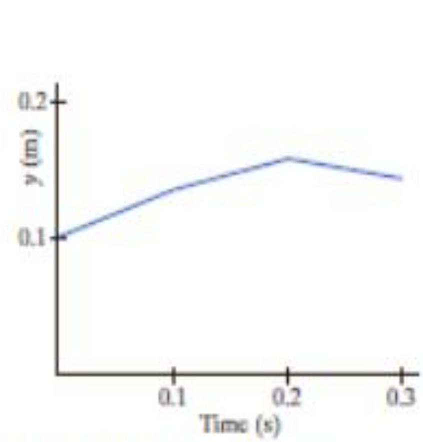

Figure 2.41 A graph of the y component of the block's position vs. time for the three-step iterative calculation.

QUESTION Is this result reasonable?

Yes, because we expect the block to oscillate up and down on the spring.

QUESTION What approximations were made in this calculation?

We made the approximation that the net force did not change significantly over each small time step. During each time step, we also used the final velocity as an approximation for the average velocity. (This is simpler than computing the arithmetic average, and not necessarily worse in a situation where the force is changing. In this case, it turns out that using the arithmetic average would actually have given a less accurate answer.)

Figure 2.42 shows the graph of y vs. *t* produced when the iterative calculation above is carried out for 10 time steps of 0.1 s.

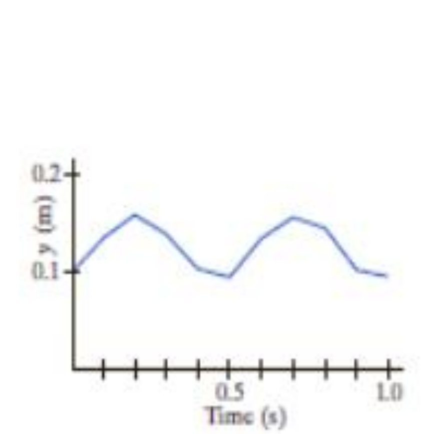

Figure 2.42 A graph of the y component of the block's position vs. time for an iterative calculation carried out for 10 steps of 0.1 s.

QUESTION What features of the graph in Figure 2.42 reflect inaccuracies in the calculations?

Although the graph does show the oscillatory motion of the block, its jagged lines reflect the fact that 0.1 s is too large a step size to produce an accurate result. There are two issues:

- The graph is jagged because we're connecting temporally distant computed points by long straight lines.
- The computed points themselves are inaccurate because we used a rather long time step of 0.1 s, which is clearly large compared to the time scale of the changes occurring in the motion.

Since the motion of the block–spring system repeats over and over, this motion is called *periodic* motion. The time interval between maxima in the plot of y vs. *t* is called the *period* of the motion, and is usually represented by the symbol *T*. From the graph in Figure 2.42 we can see that for this particular mass-spring system the period *T* appears to be around half a second. To get a better estimate of the period, we would need a better graph.

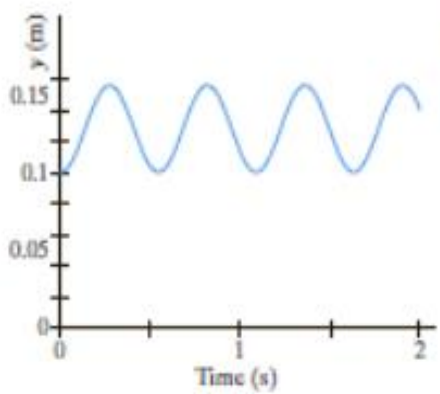

Figure 2.43 A graph of the y component of the block's position vs. time for an iterative calculation using a step size of 0.02 s.

> **Checkpoint 8** **(a)** In step 2 of the mass–spring example above, the net force on the block was downward, but the block moved upward. Explain why this was possible, and what the effect of the net force was during this time step. **(b)** In step 3 of the mass–spring example above, only the results of the calculations are given. Carry out the calculations for this step yourself, to be sure you understand in detail the procedure used. Compare your results to the values given above.

Improving Accuracy

QUESTION How could we improve the accuracy of the preceding calculations?

In general, decreasing the time step size will produce more accurate results, because the assumption that F_{net} is constant during the time Δt will be more valid. Using a step size of 0.02 s produces a smoothly oscillating graph of y vs. t, as shown by the curve in Figure 2.43. Now the straight lines connecting the computed points are short, making the graph look more smooth, and the computed points themselves are more accurate thanks to using a shorter time step, so that the force and velocity are more nearly constant during each time interval. This prediction is in agreement with experiments with masses oscillating on springs.

QUESTION Suppose that we had chosen to use a Δt of 1 s. What effect would this have had on our prediction of the block's motion?

The periodicity of the system's motion is due to the particular way in which the net force on the block is changing with time. Since 1 s is actually longer than the period of the motion, assuming that the net force is constant over this interval will lead to an extremely inaccurate result. The resulting graph of y vs. t, shown in Figure 2.44, shows that the predicted y position of the mass after 1 s would be more than 3 m above the floor! A 1 s time step is clearly too large for this system; if the motion we want to predict is periodic, we need a time step that is much smaller than the period of the motion.

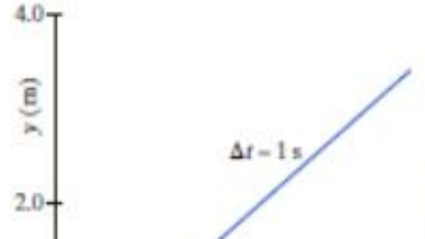

Figure 2.44 If we use a time step of 1 s to predict the motion of the mass–spring system, we predict that the mass will be more than 3 m above the floor! Note that the vertical scale of this graph is very different from the scale of the preceding graphs.

There is no single rule for picking an appropriate time step, and sometimes we will need to use a guess-and-check strategy to refine our choice. One approach to deciding if we have chosen an appropriate value for Δt is to try reducing the step size significantly, and see if the predicted motion changes. However, it will be much less tedious to do calculations for many small time steps if we can instruct a computer to do the calculation for us.

QUESTION Would you expect the iterative calculation to be more accurate if we updated the position using $(\vec{v}_i + \vec{v}_f)/2$ instead of using the velocity at the end of the time interval?

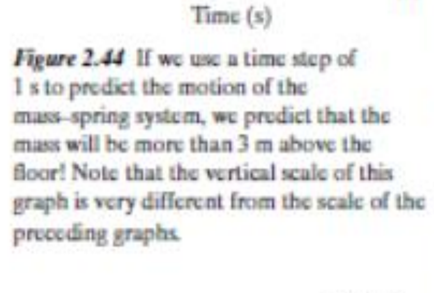

Figure 2.45 Two graphs of the y component of the block's position vs. time for an iterative calculation using a step size of 0.02 s. Blue curve: the velocity at the end of the time interval, $\vec{v}_f$, was used to update the position. Red curve: the arithmetic average, $(\vec{v}_i + \vec{v}_f)/2$, was used. In this calculation the arithmetic average gives significantly worse results.

For the blue curve in Figure 2.45 the velocity at the end of the time interval, $\vec{v}_f$, was used to update the position, whereas for the red curve the arithmetic average, $(\vec{v}_i + \vec{v}_f)/2$, was used. Using the arithmetic average (red curve) clearly produces a very inaccurate prediction: the oscillation will not grow larger and larger in the real world! Here we see an example of the fact that the arithmetic average isn't necessarily the best way to estimate the average velocity in an iterative calculation. In more advanced courses on computational modeling one learns special numerical techniques for optimizing speed and accuracy. For our purposes in this introductory course, we get good results by calculating the net force, then updating the momentum, then updating the position using the final velocity.

Why Not Just Use Calculus?

You might wonder why we don't simply use calculus to predict the motion of physical systems. There are two answers to this question:

First: In fact, we actually are using calculus, in its most fundamental form. Step by step, we add up a large number of small increments of the momentum of an object, and a large number of small displacements of the object, to calculate a large change in its momentum and position over a long time, and this corresponds to a numerical evaluation of an integral:

$$\Delta\vec{p}_1 + \Delta\vec{p}_2 + \cdots \approx \int_i^f d\vec{p} = \int_i^f \vec{F}_{net}\, dt$$

$$\Delta\vec{r}_1 + \Delta\vec{r}_2 + \cdots \approx \int_i^f d\vec{r} = \int_i^f \vec{v}_{avg}\, dt$$

When we calculate these summations with finite time steps, we call the process *numerical integration* (Figures 2.46 and 2.47). The integral sign used in calculus is a distorted "S" meaning "sum" of an infinite number of infinitesimal quantities, from the initial time t to the final time f. Evaluating a definite integral in calculus corresponds to taking an infinite number of infinitesimally small time steps in a summation.

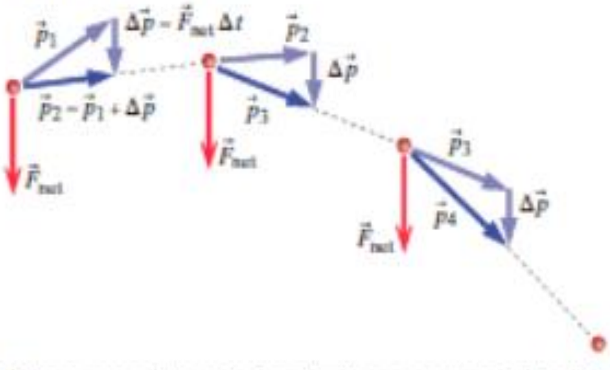

Figure 2.46 When we predict motion iteratively, we are numerically adding a large number of small increments ($\Delta\vec{p}_i$) to the starting momentum to get the momentum at some future time, as shown in Figure 2.47. This is called "numerical integration."

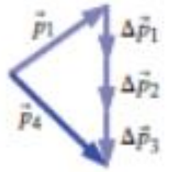

Figure 2.47 Numerical integration: adding all the $\Delta\vec{p}_i$s to $\vec{p}_1$ (Figure 2.46) gives the future momentum $\vec{p}_4$.

Second: A more interesting answer is that the motion of most physical systems actually *cannot* be predicted using calculus in any way other than by numerical integration. In a few special cases calculus does give a general result without carrying out a numerical integration. For example, we have seen that an object subjected to a constant force has a constant rate of change of momentum and velocity, and calculus can be used to obtain a prediction for the position as a function of time. An analytical solution can be derived for the motion in one dimension of a mass attached to a very low mass spring, in the absence of air resistance or friction, as we will see in Chapter 4. The elliptical orbits of two stars around each other can be predicted mathematically without an iterative approach, although the math is quite challenging.

However, in only slightly more complicated situations, an analytical solution may not be possible at all. For example, the general motion of three stars around each other has never been successfully analyzed in this way. The basic problem is that it is usually relatively easy to take the derivative of a known function, but it is often impossible to determine in algebraic form the integral of a known function, which is what would be involved in long-term prediction (adding up a large number of small momentum increments due to known forces).

In contrast, a step-by-step procedure of the kind we carried out for the mass–spring system can easily be extended to three or more bodies in three dimensions. It is also possible to include the effects of various kinds of friction and damping in an iterative calculation. This is why we learn to use the step-by-step prediction method: because it is a powerful technique of increasing importance in modern science and engineering, thanks to the availability of powerful computers to do the repetitive work for us.

2.7 ITERATIVE CALCULATIONS ON A COMPUTER

While the iterative scheme is very general, doing it by hand is tedious. It is not difficult to program a computer to do these calculations repetitively. Computers are now fast enough that it is possible to get high accuracy simply by taking very short time steps, so that during each step the net force and velocity do not change much.

A computer program is simply a sequence of instructions that specify how to perform a calculation. There are many programming languages that can be

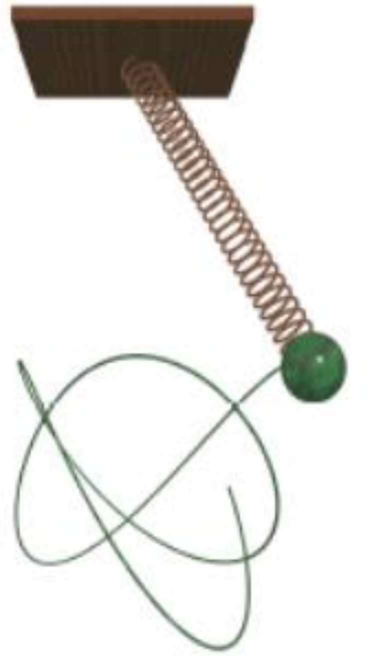

Figure 2.48 A snapshot from a computer program to calculate and animate the motion of a mass–spring system in 3D. This program was written in the language VPython (http://vpython.org), which generates real-time 3D visualizations of motion.

used to do this, but the basic organization of the program will be the same in almost all cases:

STRUCTURING ITERATIVE CALCULATIONS ON A COMPUTER

- Define the values of constants such as g or k_s to use in the program.
- Specify the masses, initial positions, and initial momenta of the interacting objects.
- Specify an appropriate value for Δt, small enough that the objects don't move very far during one update.
- Create a "loop" structure for repetitive calculations:

Repeat
- Calculate the net (vector) force $\vec{F}_{net}$ acting on the system.
- Update the momentum of the system: $\vec{p}_f = \vec{p}_i + \vec{F}_{net}\,\Delta t$.
- Update the position: $\vec{r}_f = \vec{r}_i + \vec{v}_{avg}\,\Delta t$.

As before, we use the approximation $\vec{v}_{avg} \approx \vec{p}_f/m$.

Figure 2.48 shows a display created by such a computer program.

QUESTION Do computer calculations give answers that are exactly correct?

Neither a computer nor your own calculator give "exact" answers, for several reasons. First, when we take a finite time step Δt, we are making the approximation that $\vec{F}_{net}$ is constant over this time interval. In a situation in which the net force is varying, this approximation may never be exactly true. However, if we take small enough time steps, this approximation can give very good results.

Second, real numbers (typically called "floating-point numbers" in a computational context) cannot be represented infinitely precisely inside a calculator or computer. When arithmetic operations are done with floating-point numbers there is always a small round-off error. For example, if you start with zero and repeatedly add to it the number 1×10^{-5}, doing this 1×10^{5} times, you may get the result 0.99999999999998 instead of 1.0.

Although round-off error does accumulate as the number of steps taken increases, it turns out that taking more, smaller steps does make calculations more accurate. The increased accuracy of a single smaller step more than compensates for the accumulation of round-off error in many steps.

Time Step Size in a Computer Calculation

When doing a calculation by hand, there is a trade-off between accuracy and time required to do many iterations (the smaller the time step, the more calculations must be done). Since computers are quite fast, it is reasonable to use much smaller time steps in a computer calculation than one would use by hand. However, even a fast computer can take a very long time to do calculations that use an unnecessarily tiny time step—it would not be reasonable to use a time step of 1×10^{-20} s in a prediction of the Earth's motion around the Sun.

If the motion is periodic, as in the case of an oscillating mass–spring system or a planet orbiting a star, it is important to use a time step that is much shorter than the period of the motion (the repetition time).

A standard method of checking the accuracy of a computer calculation is to decrease the time step and repeat the calculation. If the results do not change significantly, the original time step was adequately small.

Checkpoint 9 Jupiter goes around the Sun in 4333 Earth days. Which of the following would be a reasonable value to try for Δt in a *computer* calculation of the orbit? **(a)** 1 day **(b)** 4333 days **(c)** 0.01 second **(d)** 800 days

The Euler–Cromer Method of Numerical Integration

If you have progressed far enough in your study of calculus, you may have recognized that the iterative procedures we have been using to predict motion involve the Euler method of numerical integration. Actually we have been using a variation on this method, called the Euler–Cromer method, which much improves the accuracy of these iterative predictions. The key feature of this method is the order of the steps:

- First calculate the net force using current positions
- Second, use this force to update the momentum of the system
- Third, use this new momentum to update the position

Doing these calculations in a different order gives much less accurate predictions. (The proof of this is beyond the scope of this textbook; if you are interested you can find more information online.)

Iterative Calculations in VPython

■ The computational homework problems at the end of Chapter 1, and the instructional videos referenced there, introduce you to the key concepts needed to understand VPython code.

The most important part of an iterative computer model is the calculation loop, which contains all the instructions that need to be repeated for each time step. Consider the case of a fan cart moving under the influence of the nearly constant force of the air on the fan. Our computational loop might look like this code, which is an excerpt from a complete program:

```
while True:
    rate(100)
    F_fan = vector(-0.4,0,0)
    F_net = F_fan
    p_cart = p_cart + F_net * deltat
    cart.pos = cart.pos + (p_cart/m_cart) * deltat
```

In contrast, the computational loop in a program modeling a block hanging from a vertical spring might look like this:

```
while True:
    rate(100)
    F_grav = vector(0, -g*m_block, 0)
    L = block.pos - spring.pos
    Lhat = L/mag(L)
    s = mag(L) - L0
    F_spring = -ks * s * Lhat
    F_net = F_grav + F_spring
    p_block = p_block + F_net * deltat
    block.pos = block.pos + (p_block/m_block) * deltat
```

QUESTION What code is essentially the same in both loops?

The last two lines of code are essentially the same. In the next-to-last line, although the objects involved have different names (one is a block, one is a cart), the Momentum Principle is used to update the momentum of each object. In the last line, the position update equation is used to update the position of each object by approximating the average velocity by $\vec{p}_f/m$.

QUESTION What is the major difference between these two computations?

The forces on the two objects are very different. The fan cart is subject only to a constant force, while the net force on the block is the sum of two forces: a constant gravitational force and a spring force that varies with the stretch of the spring.

The instructions that specify how to calculate the spring force mirror what you would do on paper, using your calculator. If $\vec{r}_0$ is the location of the fixed end of the spring, then here is a translation of the algebraic expressions to VPython expressions:

$\vec{L}=\vec{r}-\vec{r}_0$	`L = block.pos - spring.pos`
$\hat{L}=\vec{L}/\|\vec{L}\|$	`Lhat = L/mag(L)`
$s=\|\vec{L}\|-L0$	`s = mag(L) - L0`
$\vec{F}=-k_s\cdot s\cdot\hat{L}$	`F_spring = -ks * s * Lhat`

QUESTION Where is $|\vec{L}|$ calculated?

VPython provides a function for calculating the magnitude of a vector. In the code above, the `mag()` function is used to get the magnitude of $\vec{L}$. We could have calculated it separately instead, giving it a name such as Lmag, as is done below:

```
Lmag = sqrt(L.x**2 + L.y**2 + L.z**2)
Lhat = L/Lmag
```

Checkpoint 10 Some code would need to be added in front of each computational loop discussed above in order to make a runnable program. What things would you need to instruct the computer to do before beginning the loop? **(a)** Create objects **(b)** Define constants **(c)** Set initial positions **(d)** Set initial momentum **(e)** Calculate the net force

2.8 *DERIVATION: SPECIAL-CASE AVERAGE VELOCITY

Here are two proofs, one geometric and one algebraic (using calculus), for the following special-case result concerning average velocity:

$$v_{avg,x}=\frac{(v_{ix}+v_{fx})}{2}\quad \text{if } v_x \text{ changes at a constant rate.}$$

The results are similar for $v_{avg,y}$ and $v_{avg,z}$.

Figure 2.49 Graph of v_x vs. t (constant force), divided into narrow vertical slices each of height v_x and width Δt.

Geometric Proof

If $F_{net,x}$ is constant, $p_{fx}=p_{ix}+F_{net,x}\Delta t$ implies that p_x changes at a constant rate. At speeds small compared to the speed of light, $v_x\approx p_x/m$, so a graph of v_x vs. time is a straight line, as in Figure 2.49. Using this graph, we form narrow vertical slices, each of height v_x and narrow width Δt.

Within each narrow slice v_x changes very little, so the change in position during the brief time Δt is approximately $\Delta x=v_x\Delta t$. Therefore the change in x is approximately equal to the area of the slice of height v_x and width Δt (Figure 2.50).

If we add up the areas of all these slices, we get approximately the area under the line in Figure 2.49, and this is also equal to the total displacement $\Delta x_1+\Delta x_2+\Delta x_3+\cdots=x_f-x_i$. If we go to the limit of an infinite number of slices, each with infinitesimal width, the sum of slices really *is* the area, and this