DISCRETIZATION II

The Harmonic Oscillator

Review of Harmonic Oscillators

One of the most frequently studied systems in physics is the harmonic oscillator. In the two systems considered above, the acceleration of the system was constant (a = 0 or a = g). In the harmonic oscillator, the acceleration varies with the position of the particle.

Algebraically, this is described via Hooke's Law:

$$F = -kx$$

Combined with Newton' s second law, we have

$$F = ma \Rightarrow ma = -kx \Rightarrow a = -\left(\frac{k}{m}\right)x$$

In differential terms, this yields the second order differential equation :

$$\frac{d^2x}{dt^2} = \left(\frac{-k}{m}\right)x \qquad (1)$$

or in "dot" notation :

$$\ddot{x} = -\left(\frac{k}{m}\right)x$$

where a "dot" indicates differentiation with respect to time, so that

$$\dot{x} = \frac{dx}{dt} \text{ and } \ddot{x} = \frac{d^2x}{dt^2}$$

Now, we know that the solutions to the harmonic oscillator problems are sin and cos. This makes sense if you consider the differential equation in eq. (1). This equation relates the second derivative of a function to the negative of the original function (times a constant). What functions do we know that when differentiated twice, return the negative of the original function? And the answer, as you learned in intro calc, are the sin and cos functions. Recall that :

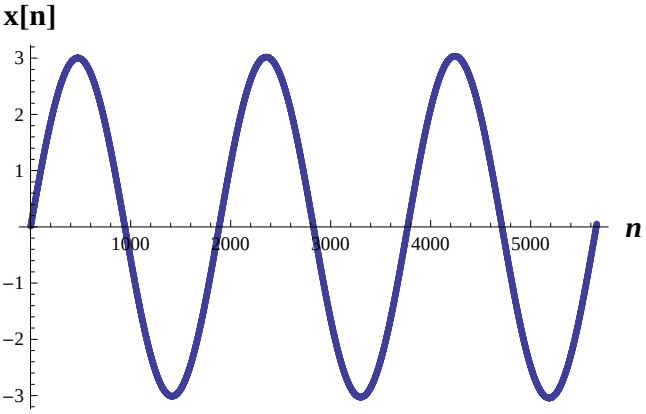
$$\frac{d}{dx}\sin x = \cos x; \quad \frac{d^2 \sin x}{dx} = \frac{d}{dx}\cos x = -\sin x$$

Modelling via Mathematica

You have learned to solve the simple harmonic oscillator in introductory physics. Let' s see if we can model this system via Mathematica. Remember that acceleration is now non - constant and is a function of the position, x. We will need to solve for x[n], v[n], and also for an appropriate function of acceleration :

In[1]:=
```
Clear[x, v, h, k, m]
x[0] = 0; h = 0.001; v[0] = 10; k = 1111.11; m = 100;
a[x_] := -(k / m) x;
v[n_] := v[n] = v[n - 1] + a[x[n - 1]] h
x[n_] := x[n] = x[n - 1] + h (v[n] + v[n - 1]) / 2
ListPlot[Table[x[n], {n, 5660}],
 AxesLabel → {Style[n, 14, Bold], Style["x[n]", 14, Bold]}]
```

Out[6]=



Checking the accuracy of the model: The Period:

Let' s look at the output and see if it matches what we expect. If we have properly modeled this system, we expect to reproduce the motion of a harmonic oscillator of mass 100 kg and spring constant 1111.11 Nm (why the odd choice for k?). First, we expect to observe periodic motion, and the graph above (which has the x position on the vertical axis and the value of n on the horizontal axis) certainly appears to be sinusoidal. (Remember, n is related to, but not exactly equal to the time elapsed. But is this the correct representation of the motion of this system?)

We can check our numerical results against the results of analytic computations. We know, for instance, that the period of a harmonic oscillator is given by:

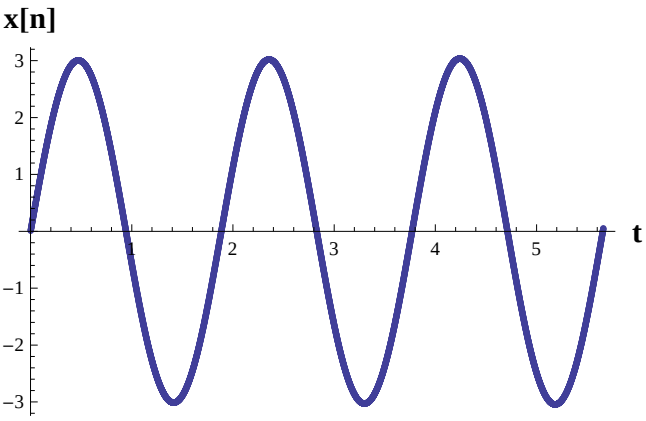
$$P = 2\pi\sqrt{m/k} = 2\pi\sqrt{100\,\text{kg}/1111.11\,\text{Nm}} = 1.884\,\text{s}$$

This means that it should take 3 x .1884 s = 5.652 s to complete three cycles. Below is a slightly different plot of this motion. Notice how the ListPlot[Table command now includes two arguments; we are plotting the quantity n h on the x axis and x[n] on the y axis.

Remember that n is the value of the time step, and h is the duration of each time step, therefore, the quantity n h is simply the time elapsed. Notice also that we are plotting the first 5660 n values (or the first 5660 snapshots of the motion of the oscillator); if each time step has a value of 0.001 s, then the total time plotted is 5.66 seconds. If you look carefully, you can see that the model predicts just slightly more than 3 cycles will be completed in 5.66 s, in good agreement with our analytical calculation. The graph of x[n] vs. time is produced below; notice this time we are plotting the quantity "n h" on the x-axis and x[n] on the y-axis, producing what is essentially a graph of x(t) vs. t.

In[7]:=
```
ListPlot[Table[{n h, x[n]}, {n, 5660}],
  AxesLabel → {Style["t", 14, Bold], Style["x[n]", 14, Bold]} ]
```

Out[7]=



Checking the Accuracy of the Model: The Amplitude:

We can also compare the amplitude of the predicted motion with the results of calculation. Our model assumes that x = 0 at t = 0, and that v = 10 m/s at t = 0. This means that at t = 0, all the energy is in the form of kinetic energy, and energy conservation tells us that :

$$\text{PE} + \text{KE} = \text{constant} = \text{total Energy}$$

In this case, we can write :

$$\frac{1}{2} k x^2 + \frac{1}{2} m v^2 = E \tag{2}$$

When x = 0, all the energy is kinetic; when x is at its maximum distance from equilibrium, all the energy is potential, if we call this distance x_{max}, we have:

$$\frac{1}{2} k x^2_{max} =$$

$$\frac{1}{2} m v(0)^2 \Rightarrow x_{max} = v(0) \sqrt{m/k} = 10\,\text{m/s} \sqrt{100\,\text{kg}/1111.11\,\text{Nm}} = 3\,\text{m}$$

As you can guess, the constants were chosen to produce an integer result; this calculation predicts the amplitude of motion will be 3 m.Our graph above appears to be in good agreement, but let' s dive deeper to do more than a visual scan.We have already determined that the period of one cycle is 1.884 seconds, therefore the time for the particle to move from the origin to its maximum distance should be 1/4 of a cycle, or 0.47124 seconds.Now, since we are sampling the position of the particle every 0.001 s, we are predicting that the particle should reach this maximum amplitude between the 470 th and 480 th time step (remember each time step is 0.01 s in duration).Thus, let' s check the values of x[n] in the vicinity of n = 470 :

In[8]:=
```
Block[{$RecursionLimit = Infinity}, Do[Print[x[n]], {n, 470, 480}]]
```

```
3.0039
3.00393
3.00393
3.0039
3.00383
3.00373
3.00359
3.00343
3.00323
3.00299
3.00272
```

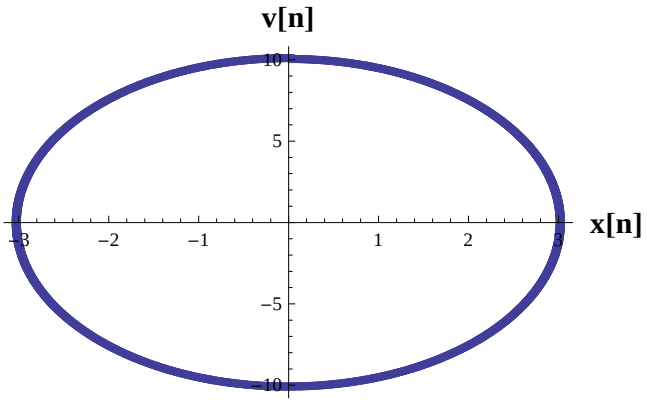
These data indicate the maximum displacement occurs just after 0.47 seconds, and the amplitude is slightly greater than 3m, suggesting that our model is fairly accurate, and with a smaller choice of step size, should approximate the analytical results even more closely.

Phase Diagrams

Let' s look again at equation (2) and ask the question : What should a graph of v (t) vs. x (t) look like? Let' s construct a plot of v[n] vs. x[n] for three cycles of this oscillator:

In[9]:=
```
ListPlot[Table[{x[n], v[n]}, {n, 5660}],
 AxesLabel → {Style["x[n]", 14, Bold], Style["v[n]", 14, Bold]}]
```

Out[9]=



What a beautiful planetary orbit, I mean, ellipse. Why do you think this is the shape of the v (t) vs. x (t) graph? Notice that v = 10 when x = 0; this is what you expect from simple energy considerations. The particle has maximum velocity when it passes through the equilibrium point. When the particle is at greatest distance from the origin (x = 3 and - 3), the velocity is zero, consistent with your understanding of conservation of energy in harmonic oscillators.

This type of v (t) vs. x (t) diagram is called a **phase diagram**, and is a useful took in analyzing the nature of dynamical systems.

Adding friction to the fray

Let' s take our initial system and add some friction. The typical way of writing the differential equation for a damped oscillator is :

$$m\,\ddot{x} + c\,\dot{x} + k\,x = 0$$

Let' s take our initial system and add some friction. The typical way of writing the differential equation for a damped oscillator is :

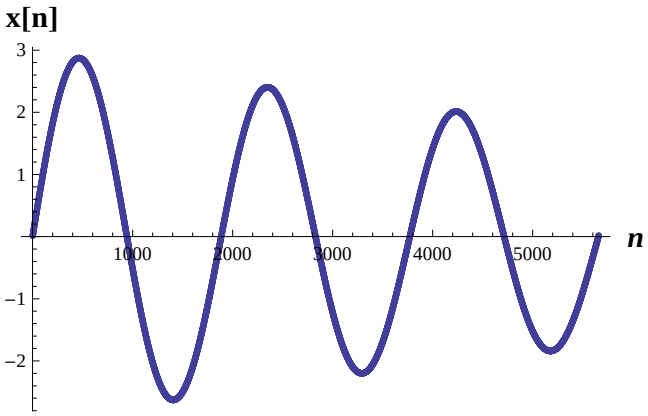
$$m\,\ddot{x} + c\,\dot{x} + k\,x = 0$$

where again we use the "dot" notation to denote time derivatives. If we solve for acceleration, we have :

$$\ddot{x} = -(c/m)\,\dot{x} - (k/m)\,x$$

Where c is some constant, and the amount of friction is proportional to the velocity of the oscillator. I arbitrarily set c/m = 0.2:

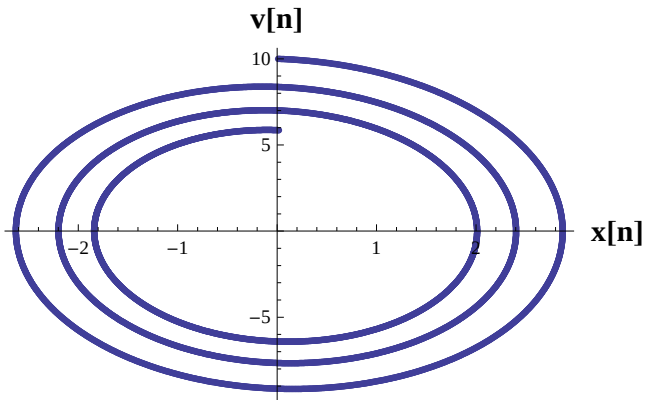
```
Clear[x, v, h, c, k, m]
x[0] = 0; h = 0.001; v[0] = 10; k = 1111.11; m = 100; c = 0.2;
a[x_, v_] := -(k / m) x - 0.2 v

v[n_] := v[n] = v[n - 1] + (a[x[n - 1], v[n - 1]]) h
x[n_] := x[n] = x[n - 1] + h (v[n - 1] + v[n - 1]) / 2
ListPlot[Table[x[n], {n, 5660}], AxesLabel → {Style[n, 14, Bold], Style["x[n]", 14, Bold]}]
```



How do these oscillations compare to the undamped case? What causes this difference?

Below is the phase diagram for three cycles of the damped harmonic oscillator. What is the physical significance of the spiraling in this diagram? What is the physical significance of the lack of spiralling in the phase diagram for the undamped oscillator?

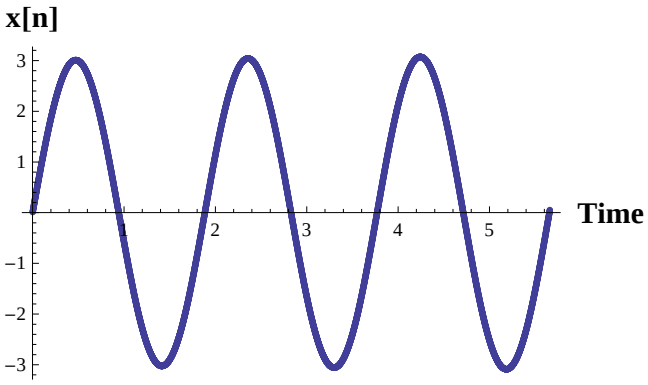
```
ListPlot[Table[{x[n], v[n]}, {n, 5660}],
 AxesLabel → {Style["x[n]", 14, Bold], Style["v[n]", 14, Bold]}]
```



Investigating the Effect of Step - Size :

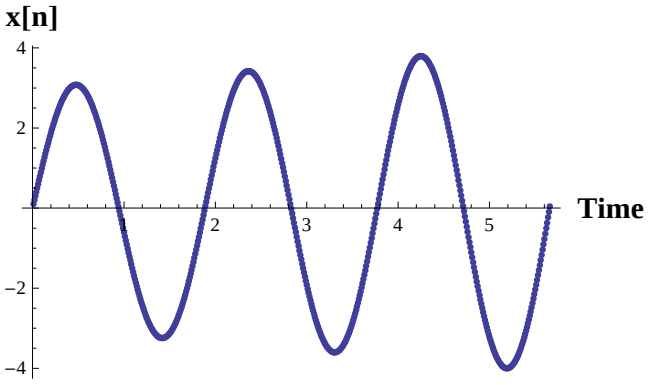
We can see the importance of choosing the proper step - size using our original program and set of parameters :

```
Clear[x, v, h, k, m]
x[0] = 0; h = 0.001; v[0] = 10; k = 1111.11; m = 100;
a[x_] := -(k / m) x;
v[n_] := v[n] = v[n - 1] + a[x[n - 1]] h
x[n_] := x[n] = x[n - 1] + h (v[n - 1] + v[n - 1]) / 2
g1 = ListPlot[Table[{n h, x[n]}, {n, 5660}],
  AxesLabel → {Style["Time", 14, Bold], Style["x[n]", 14, Bold]}]
```



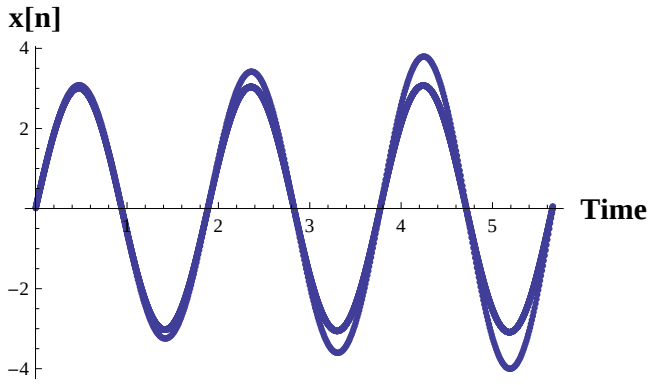
Now, let' s see what happens if I use exactly the same code with a step size of 0.01 (since I am increasing the step size by a factor of 10, I will reduce the number of steps by a similar factor, so will only plot 566 points) :

```
Clear[x, v, h, k, m]
x[0] = 0; h = 0.01; v[0] = 10; k = 1111.11; m = 100;
a[x_] := -(k / m) x;
v[n_] := v[n] = v[n - 1] + a[x[n - 1]] h
x[n_] := x[n] = x[n - 1] + h (v[n - 1] + v[n - 1]) / 2
g2 = ListPlot[Table[{n h, x[n]}, {n, 566}],
  AxesLabel → {Style["Time", 14, Bold], Style["x[n]", 14, Bold]}]
```



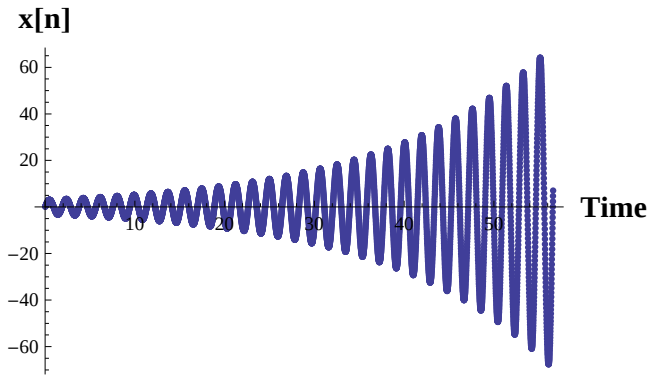
Do you see any difference between the two graphs? Let' s superimpose the plots and see if any differences emerge :

```
Show[g1, g2]
```



The two graphs do not completely coincide, and show significant differences by the third cycle. We can emphasize the nature of this difference by producing 30 cycles of our model using a step size of h = 0.01 :

```
Clear[x, v, h, k, m]
x[0] = 0; h = 0.01; v[0] = 10; k = 1111.11; m = 100;
a[x_] := -(k / m) x;
v[n_] := v[n] = v[n - 1] + a[x[n - 1]] h
x[n_] := x[n] = x[n - 1] + h (v[n - 1] + v[n - 1]) / 2
ListPlot[Table[{n h, x[n]}, {n, 5660}],
 AxesLabel → {Style["Time", 14, Bold], Style["x[n]", 14, Bold]}]
```



The behavior of this model is clear. As time elapses, the errors in this numerical model propagate in such a way as to increase the amplitude of oscillation. We know that this is incorrect since there is no forcing term in our equations, so what is causing this increase in amplitude?

Our code above calculates the acceleration, velocity and position of the oscillator at specific, discrete times, and uses the last set of values to predict the next set of values. The assumption we are using is that the velocity and acceleration remain constant during the interval associated with the step-size; however we know this is not a perfectly accurate assumption since we know the acceleration and velocity change continuously with the position of the oscillator. Therefore, if we make our step-size too large, we introduce too much error into our computations. As the graphs above show, determining the proper step-size for a problem is a critical element of successful programming.

It should not be too surprising that the choice of step-size is an important factor. When you first learned about the concept of a derivative in calculus, you learned that the derivative is equal to the slope of the straight line that connects the two points (x+h, f(x+h)) and (x, f(x)), but only if h is small enough.

This might suggest that your best strategy is always to make the step-size as small as possible, but this will require many more iterations of computation, thereby slowing down the execution of your program. The key is to find the most appropriate step-size that will preserve the accuracy of your model without unnecessarily slowing down your calculations.