INTRODUCTION TO DISCRETIZATION

Today we begin learning how to write equations in a form that will allow us to produce numerical results. In introductory physics courses, almost all the equations we deal with are continuous and allow us to write solutions in closed form equations. However, in more advanced physics, it becomes necessary to be able to solve equations numerically. Discretization is the name given to the processes and protocols that we use to convert a continuous equation into a form that can be used to calculate numerical solutions.

Let' s start with some very simple examples. Suppose I want to model the motion of an object traveling at constant speed in one direction. For specificity, let' s say an object is traveling along the + x direction with a speed of 10 m/s, we would write its position vector as :

$$x(t) = x_0 + 10\,t$$

where x_0 is the position of the particle at time t=0. It is trivial to plot this motion, or to compute x(t) for any time t. But let's see how we would *discretize* this very simple equation. Discretization means we consider the motion to occur in discrete packets, and we seek to model a way of describing where the position of the particle at the n^{th} position. This procedure mimics the way you take data in a laboratory; you can't, despite your best intentions, take data at every possible value of time or position; rather, you take data at specific points or moments in time.

We approach the problem with this sort of reasoning : if the particle' s current position is x[n], then in the next time frame, it will be at position x[n + 1]. These two positions are related by the motion of the particle between the two times, so we can write :

$$(n+1)^{st}\text{ position} = n^{th}\text{ position} + \text{motion of particle between } x[n] \text{ and } x[n+1]$$

We know that the particle will travel a distance of 10 Δt for a time interval of Δt, so that we can write:

$$x[n+1] = x[n] + 10\,\Delta t$$

Equivalently, this can be written as :

$$x[n] = x[n-1] + 10\,\Delta t$$

Meaning the current position is equal to the last position plus motion since the last position

And with this formulation, we can determine the position of the particle at any time t. Let' s write this in Mathematica form, noticing which variables we have to define :

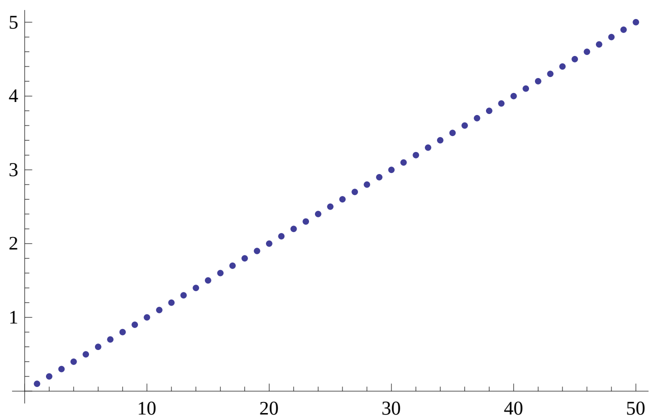
```
Clear[x, n, h, v]
v = 10; h = 0.01;
x[0] = 0;
x[n_] := x[n] = x[n − 1] + v h
x[10]
```

```
1.
```

In the code above, what are the meaning of the variables? What do v, h, and x[10] represent? What is the impact of setting x[0]=0?

Let' s see how we can plot data in this form; this will introduce us to two new and useful commands : the Table and ListPlot commands :

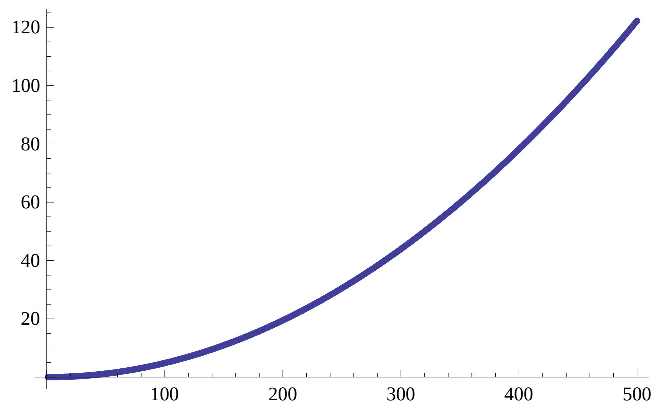
```
ListPlot[Table[x[n], {n, 50}]]
```



This plots the first 50 values of x[n] vs. n. ***Does n exactly represent time, or is n merely related to time (or is n completely unrelated to the passage of time?)***

Suppose now we want to analyze a situation where the velocity is not constant, for instance in the case of an object falling in the earth' s gravitational field under the force of gravity. We know that the velocity increases linearly with time of flight, or in equation form : v (t) = g t (where we will assume the downward direction is positive). Let' s consider the following code :

```
Clear[y, h, v, g]
g = 9.8;  h = 0.01; y[0] = 0; v[0] = 0;
v[n_] := v[n] = v[n − 1] + g h
y[n_] := y[n] = y[n − 1] + v[n − 1] h
ListPlot[Table[y[n], {n, 500}]]
```



For the input values, what is the time interval described in the plot above? Let' s see how our discrete model compares with the solution we expect?

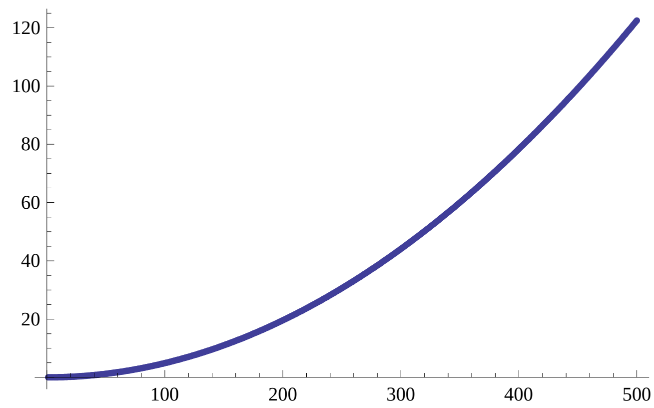
```
(1 / 2 g 5 ^ 2) / y[500]
```

```
1.002
```

Pretty close but not exact. Our two answers are off by 0.2 %. What might be causing even this small amount of error?

Let' s see what happens if we amend the code slightly :

```
Clear[y, h, v, g]
g = 9.8;  h = 0.01; y[0] = 0; v[0] = 0;
v[n_] := v[n] = v[n - 1] + g h
y[n_] := y[n] = y[n - 1] + 1 / 2 (v[n] + v[n - 1]) h
ListPlot[Table[y[n], {n, 500}]]
```




```
(1 / 2 g 5^2) / y[500]
1.
```

What did we do that was different, how else might we have reduced our error?

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 :

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

or in "dot" notation :

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

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

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

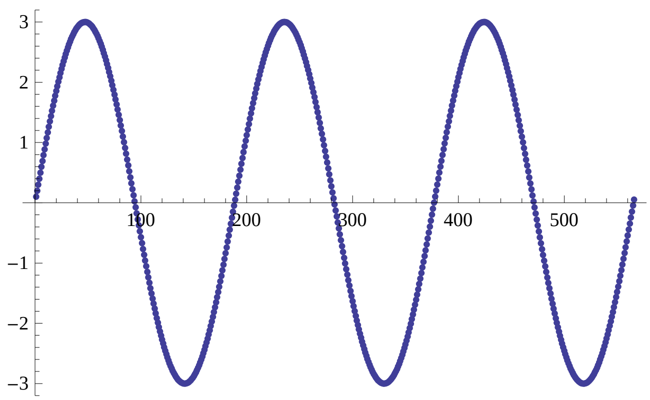
Now, we know that the solutions to the harmonic oscillator problems are sin and cos. This makes sense if you consider the differential equation in eq. (1). This equation relates the second derivative of a function to the negative of the original function (times a constant). What functions do we know 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 :

```
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]] h
x[n_] := x[n] = x[n − 1] + h (v[n − 1] + v[n − 1]) / 2
ListPlot[Table[x[n], {n, 566}]]
```



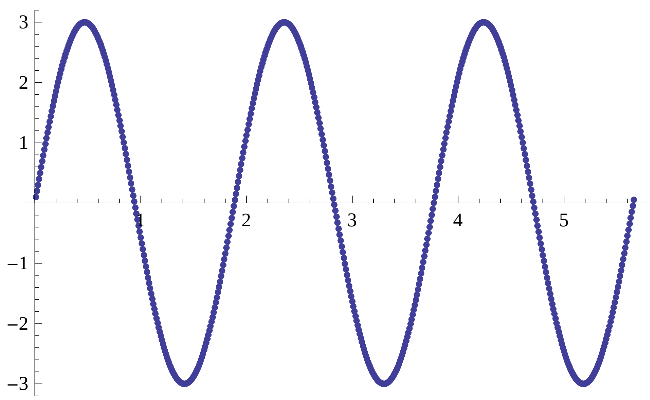
Let' s look at the output and see if it matches what we expect. If we have properly modeled this system, we expect to reproduce the motion of a harmonic oscillator of mass 100 kg and spring constant 1111.11 Nm (why the odd choice for k?). First, we expect to observe periodic motion, and the graph above (which has the x position on the vertical axis and the value of n on the horizontal axis. 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:

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

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 566 n values (or the first 566 snapshots of the motion of the oscillator); if each time step has a value of 0.01 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.

```
ListPlot[Table[{n h, x[n]}, {n, 566}]]
```



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 :

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

In this case, we can write :

$$\frac{1}{2} k x^2 + \frac{1}{2} m v^2 = E \qquad (2)$$

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:

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

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 0.01 s, we are predicting that the particle should reach this maximum amplitude between the 47 th and 48 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 = 47 :

{x[45], x[46], x[47], x[48], x[49]}

```
{2.99292, 2.99832, 3.00039, 2.99913, 2.99454}
```

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

Testing the effect of step - size :

If I redo the calculation with the step size set = 0.001, we then expect the maximum displacement to reach a value of 3 between x[471] and x[472]. The results of this calculation are :

```
Block[{$RecursionLimit = Infinity}, {x[470], x[471], x[472], x[473]}]
```

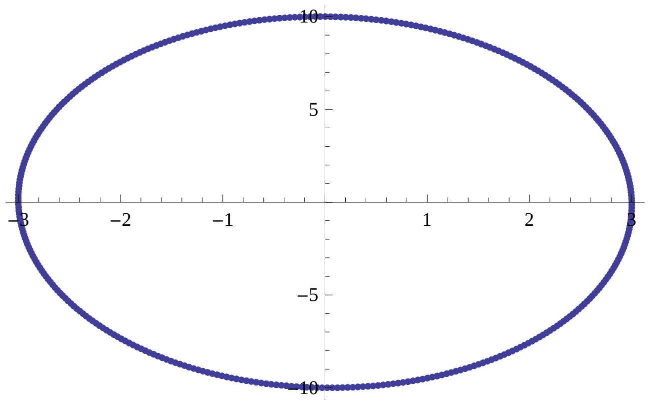
```
{2.99998, 3., 3., 2.99995}
```

(The "Block[{$RecursionLimit=Infinity}" statement tells Mathematica not to cutoff calculations.)

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 one cycle

```
ListPlot[Table[{x[n], v[n]}, {n, 985}]]
```



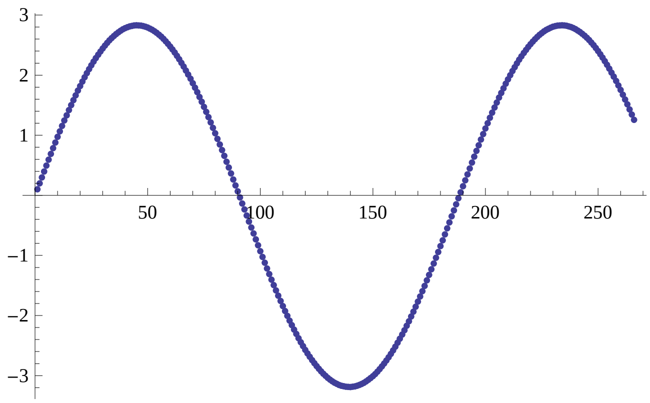
What a beautiful planetary orbit, I mean, ellipse. Why do you think this is the shape of the v (t) vs. x (t) graph? Notice that v = 10 when x = 0; this is what you expect from simple energy considerations. The particle has maximum velocity when it passes through the equilibrium point. When the particle 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 a constant amount of friction to the system as if our mass is sliding on a table that has a constant coefficient of friction. Let ' s say that friction with the table changes the nature of the acceleration, so that our code becomes (notice the change in the definition of acceleration) :

```
Clear[x, v, h, c, k, m]
x[0] = 0; h = 0.01; v[0] = 10; k = 1111.11; m = 100; c = 2;
a[x_] := - (k / m) x - c
v[n_] := v[n] = v[n - 1] + a[x[n]] h
x[n_] := x[n] = x[n - 1] + h (v[n - 1] + v[n - 1]) / 2
ListPlot[Table[x[n], {n, 266}]]
```




```
ListPlot[Table[{x[n], v[n]}, {n, 200}]]
```