MITOCW | MIT18_03SCf11_app7.mp4

The *Euler's Method* applet helps us understand numerical methods for approximating solutions to differential equations. I can choose the differential equation using this pull down menu, and I've selected the equation y prime equals y squared minus x, the same equation that we used in the isocline applet.

The graphing window shows a slope field, the slope field of this equation. And the value of the slope field can be read off by rolling over the window. It's read off on the right-hand side here. f of x, y is various values depending on where I'm located.

I've also chosen an initial condition, initial value, of x equals-- x_0 is zero, and y_0 is minus 1. I can see the actual solution with that initial condition by pressing Actual from this set of boxes and checking Start.

Now a curve is drawn on the graphing plane. This is the solution with that initial condition. And a table of values shows up in this left table. We can see that this is one of the solutions which is sucked into the funnel. So we understand the values of y of x quite well when x is large. But what if I want to know the value of y of 1?

According to the table over here, the value is approximately minus 0.83. But how do we know that? Euler's method is the simplest numerical method. It uses the tangent line approximation. If I set the step size to be 1, I can then click Start, and this will draw a tangent line segment, with delta x equal to 1, starting at my initial condition, and with slope given by the slope field at that point.

So the tangent line approximation to y of 1 is the value zero. Well, that's not very good. But I can improve things by using a smaller step size. So let's go down to a step size of 1/4, start again. Now I've drawn a tangent line segment, but the horizontal distance is only 1/4.

Let's see if we can see this more clearly by pressing the zoom key. This will zoom in on the same picture that we had before. I can measure the slope field at the end point of this green line segment. It seems to be about 0.32. And by pressing Next Step, I can draw a line segment moving off with that slope.

So this now, it produces a polygon, the Euler polygon, which will stay closer to the actual curve than the simple tangent line approximation did. I can continue this process by continuing to say Next Step. The table of values appears on the left, and we discover that the Euler approximation to y of 1 with step size 1/4, is minus 0.75. Much better than the earlier value we had.

And I can improve things still further by choosing a smaller step size. In fact, you get as close as you want to the actual solution by selecting sufficiently small step sizes. Let's do one more example with step size of 1/8.

Now I will click 8 times to produce an Euler polygon with 8 segments. And I have an estimate of minus 0.8. All of these estimates are too large. All of these curves, these polygons, lie above the actual solution curve. Let's see if we can see why this is.

You'll notice that the slope field is given by the formula y squared minus x. So as x increases, the slope field decreases in value. So as we're moving out along one of these Euler struts, the slope field is decreasing under it. And that causes the actual solution to fall below the Euler polygon. And that process will continue as I iterate the Euler process.

So the general rule is, if the direction field is decreasing in the x-direction, you should expect the actual solution to be less than the Euler estimate. There are lots of things that can go wrong in this kind of numerical work. To see one of them, let's unzoom. Zoom back, clear the screen, redraw the actual solution, and choose step size 1.

Now instead of wanting to compute y of 1, suppose that I wanted to compute the value of the solution at x equals 6. Well if I try doing this using step size of 1, let's see what happens.

So I begin. I have the same strut I had before. It's too large, but now the slope field has a negative value so that comes back down. Things are looking better. In the next step, I've overshot. And if I take another step, then I've overshot again in the other direction, more dramatically. And now the slope field is even more negative. So when I take the next step, I've overshot yet again, more dramatically.

And if I take the next step, now my estimate for the solution, which is down here, at x equals 6 is the value 7, this is in the range where the slope field continues to increase forever. And so my estimated solution will zoom off towards infinity, while the actual curve is down here. I call this catastrophic overshoot. It's just one of a number of different things that can go wrong when you try to use these numerical methods.