Preface

Here are my online notes for my differential equations course that I teach here at Lamar University. Despite the fact that these are my “class notes”, they should be accessible to anyone wanting to learn how to solve differential equations or needing a refresher on differential equations.

I’ve tried to make these notes as self contained as possible and so all the information needed to read through them is either from a Calculus or Algebra class or contained in other sections of the notes.

A couple of warnings to my students who may be here to get a copy of what happened on a day that you missed.

1. Because I wanted to make this a fairly complete set of notes for anyone wanting to learn differential equations I have included some material that I do not usually have time to cover in class and because this changes from semester to semester it is not noted here. You will need to find one of your fellow class mates to see if there is something in these notes that wasn’t covered in class.

2. In general I try to work problems in class that are different from my notes. However, with Differential Equation many of the problems are difficult to make up on the spur of the moment and so in this class my class work will follow these notes fairly close as far as worked problems go. With that being said I will, on occasion, work problems off the top of my head when I can to provide more examples than just those in my notes. Also, I often don’t have time in class to work all of the problems in the notes and so you will find that some sections contain problems that weren’t worked in class due to time restrictions.

3. Sometimes questions in class will lead down paths that are not covered here. I try to anticipate as many of the questions as possible in writing these up, but the reality is that I can’t anticipate all the questions. Sometimes a very good question gets asked in class that leads to insights that I’ve not included here. You should always talk to someone who was in class on the day you missed and compare these notes to their notes and see what the differences are.

4. This is somewhat related to the previous three items, but is important enough to merit its own item. THESE NOTES ARE NOT A SUBSTITUTE FOR ATTENDING CLASS!! Using these notes as a substitute for class is liable to get you in trouble. As already noted not everything in these notes is covered in class and often material or insights not in these notes is covered in class.
Euler’s Method

Up to this point practically every differential equation that we’ve been presented with could be solved. The problem with this is that these are the exceptions rather than the rule. The vast majority of first order differential equations can’t be solved.

In order to teach you something about solving first order differential equations we’ve had to restrict ourselves down to the fairly restrictive cases of linear, separable, or exact differential equations or differential equations that could be solved with a set of very specific substitutions. Most first order differential equations however fall into none of these categories. In fact even those that are separable or exact cannot always be solved for an explicit solution. Without explicit solutions to these it would be hard to get any information about the solution.

So what do we do when faced with a differential equation that we can’t solve? The answer depends on what you are looking for. If you are only looking for long term behavior of a solution you can always sketch a direction field. This can be done without too much difficulty for some fairly complex differential equations that we can’t solve to get exact solutions.

The problem with this approach is that it’s only really good for getting general trends in solutions and for long term behavior of solutions. There are times when we will need something more. For instance, maybe we need to determine how a specific solution behaves, including some values that the solution will take. There are also a fairly large set of differential equations that are not easy to sketch good direction fields for.

In these cases we resort to numerical methods that will allow us to approximate solutions to differential equations. There are many different methods that can be used to approximate solutions to a differential equation and in fact whole classes can be taught just dealing with the various methods. We are going to look at one of the oldest and easiest to use here. This method was originally devised by Euler and is called, oddly enough, Euler’s Method.

Let’s start with a general first order IVP

\[
\frac{dy}{dt} = f(t, y) \quad y(t_0) = y_0
\]  // (1)

where \(f(t,y)\) is a known function and the values in the initial condition are also known numbers. From the second theorem in the Intervals of Validity section we know that if \(f\) and \(f_y\) are continuous functions then there is a unique solution to the IVP in some interval surrounding \(t = t_0\). So, let’s assume that everything is nice and continuous so that we know that a solution will in fact exist.

We want to approximate the solution to (1) near \(t = t_0\). We’ll start with the two pieces of information that we do know about the solution. First, we know the value of the solution at \(t = t_0\) from the initial condition. Second, we also know the value of the derivative at \(t = t_0\). We can get this by plugging the initial condition into \(f(t,y)\) into the differential equation itself. So, the derivative at this point is.

\[
\left. \frac{dy}{dt} \right|_{t=t_0} = f(t_0, y_0)
\]
Now, recall from your Calculus I class that these two pieces of information are enough for us to write down the equation of the tangent line to the solution at \( t = t_0 \). The tangent line is

\[
y = y_0 + f(t_0, y_0)(t - t_0)
\]

Take a look at the figure below

If \( t_1 \) is close enough to \( t_0 \) then the point \( y_1 \) on the tangent line should be fairly close to the actual value of the solution at \( t_1 \), or \( y(t_1) \). Finding \( y_1 \) is easy enough. All we need to do is plug \( t_1 \) in the equation for the tangent line.

\[
y_1 = y_0 + f(t_0, y_0)(t_1 - t_0)
\]

Now, we would like to proceed in a similar manner, but we don’t have the value of the solution at \( t_1 \) and so we won’t know the slope of the tangent line to the solution at this point. This is a problem. We can partially solve it however, by recalling that \( y_1 \) is an approximation to the solution at \( t_1 \). If \( y_1 \) is a very good approximation to the actual value of the solution then we can use that to estimate the slope of the tangent line at \( t_1 \).

So, let’s hope that \( y_1 \) is a good approximation to the solution and construct a line through the point \((t_1, y_1)\) that has slope \( f(t_1, y_1) \). This gives

\[
y = y_1 + f(t_1, y_1)(t - t_1)
\]

Now, to get an approximation to the solution at \( t = t_2 \) we will hope that this new line will be fairly close to the actual solution at \( t_2 \) and use the value of the line at \( t_2 \) as an approximation to the actual solution. This gives.

\[
y_2 = y_1 + f(t_1, y_1)(t_2 - t_1)
\]

We can continue in this fashion. Use the previously computed approximation to get the next approximation. So,

\[
y_3 = y_2 + f(t_2, y_2)(t_3 - t_2)
\]

\[
y_4 = y_3 + f(t_3, y_3)(t_4 - t_3)
\]

etc.
In general, if we have \( t_n \) and the approximation to the solution at this point, \( y_n \), and we want to find the approximation at \( t_{n+1} \) all we need to do is use the following.

\[
y_{n+1} = y_n + f\left(t_n, y_n\right) \cdot (t_{n+1} - t_n)
\]

If we define \( f_n = f\left(t_n, y_n\right) \) we can simplify the formula to

\[
y_{n+1} = y_n + f_n \cdot (t_{n+1} - t_n)
\]  \( \text{(2)} \)

Often, we will assume that the step sizes between the points \( t_0, t_1, t_2, \ldots \) are of a uniform size of \( h \). In other words, we will often assume that

\[
t_{n+1} - t_n = h
\]

This doesn’t have to be done and there are times when it’s best that we not do this. However, if we do the formula for the next approximation becomes.

\[
y_{n+1} = y_n + hf_n
\]  \( \text{(3)} \)

So, how do we use Euler’s Method? It’s fairly simple. We start with (1) and then decide if we want to use a uniform step size or not. Then starting with \((t_0, y_0)\) we repeatedly evaluate (2) or (3) depending on whether we chose to use a uniform set size or not. We continue until we’ve gone the desired number of steps or reached the desired time. This will give us a sequence of numbers \( y_1, y_2, y_3, \ldots, y_n \) that will approximate the value of the actual solution at \( t_1, t_2, t_3, \ldots, t_n \).

What do we do if we want a value of the solution at some other point than those used here? One possibility is to go back and redefine our set of points to a new set that will include the points we are after and redo Euler’s Method using this new set of points. However this is cumbersome and could take a lot of time especially if we had to make changes to the set of points more than once.

Another possibility is to remember how we arrived at the approximations in the first place. Recall that we used the tangent line

\[
y = y_0 + f\left(t_0, y_0\right)\left(t - t_0\right)
\]

to get the value of \( y_1 \). We could use this tangent line as an approximation for the solution on the interval \([t_0, t_1]\). Likewise, we used the tangent line

\[
y = y_1 + f\left(t_1, y_1\right)\left(t - t_1\right)
\]

to get the value of \( y_2 \). We could use this tangent line as an approximation for the solution on the interval \([t_1, t_2]\). Continuing in this manner we would get a set of lines that, when strung together, should be an approximation to the solution as a whole.

In practice you would need to write a computer program to do these computations for you. In most cases the function \( f(t, y) \) would be too large and/or complicated to use by hand and in most serious uses of Euler’s Method you would want to use hundreds of steps which would make doing this by hand prohibitive. So, here is a bit of pseudo-code that you can use to write a program for Euler’s Method that uses a uniform step size, \( h \).
1. define \( f(t, y) \).
2. input \( t_0 \) and \( y_0 \).
3. input step size, \( h \) and the number of steps, \( n \).
4. for \( j \) from 1 to \( n \) do
   a. \( m = f(t_0, y_0) \)
   b. \( y_1 = y_0 + h * m \)
   c. \( t_1 = t_0 + h \)
   d. Print \( t_1 \) and \( y_1 \)
   e. \( t_0 = t_1 \)
   f. \( y_0 = y_1 \)
5. end

The pseudo-code for a non-uniform step size would be a little more complicated, but it would essentially be the same.

So, let’s take a look at a couple of examples. We’ll use Euler’s Method to approximate solutions to a couple of first order differential equations. The differential equations that we’ll be using are linear first order differential equations that can be easily solved for an exact solution. Of course, in practice we wouldn’t use Euler’s Method on these kinds of differential equations, but by using easily solvable differential equations we will be able to check the accuracy of the method.

Knowing the accuracy of any approximation method is a good thing. It is important to know if the method is liable to give a good approximation or not.

**Example 1**

For the IVP

\[
y' + 2y = 2 - e^{-4t} \quad y(0) = 1
\]

Use Euler’s Method with a step size of \( h = 0.1 \) to find approximate values of the solution at \( t = 0.1, 0.2, 0.3, 0.4, \) and \( 0.5 \). Compare them to the exact values of the solution as these points.

**Solution**

This is a fairly simple linear differential equation so we’ll leave it to you to check that the solution is

\[
y(t) = 1 + \frac{1}{2}e^{-4t} - \frac{1}{2}e^{-2t}
\]

In order to use Euler’s Method we first need to rewrite the differential equation into the form given in (1).

\[
y' = 2 - e^{-4t} - 2y
\]

From this we can see that \( f(t, y) = 2 - e^{-4t} - 2y \). Also note that \( t_0 = 0 \) and \( y_0 = 1 \). We can now start doing some computations.

\[
f_0 = f(0, 1) = 2 - e^{-4(0)} - 2(1) = -1
\]

\[
y_1 = y_0 + h f_0 = 1 + (0.1)(-1) = 0.9
\]

So, the approximation to the solution at \( t_1 = 0.1 \) is \( y_1 = 0.9 \).

At the next step we have
\[ f_1 = f(0.1, 0.9) = 2 - e^{-0.1} - 2(0.9) = -0.470320046 \]
\[ y_2 = y_1 + h f_1 = 0.9 + (0.1)(-0.470320046) = 0.852967995 \]

Therefore, the approximation to the solution at \( t = 0.2 \) is \( y_2 = 0.852967995 \).

I’ll leave it to you to check the remainder of these computations.

\[ f_2 = -0.155264954 \]
\[ f_3 = 0.023922788 \]
\[ f_4 = 0.1184359245 \]
\[ y_3 = 0.837441500 \]
\[ y_4 = 0.839833779 \]
\[ y_5 = 0.851677371 \]

Here’s a quick table that gives the approximations as well as the exact value of the solutions at the given points.

<table>
<thead>
<tr>
<th>Time, ( t_n )</th>
<th>Approximation</th>
<th>Exact</th>
<th>Error</th>
</tr>
</thead>
<tbody>
<tr>
<td>( t_0 = 0 )</td>
<td>( y_0 = 1 )</td>
<td>( y(0) = 1 )</td>
<td>0 %</td>
</tr>
<tr>
<td>( t_1 = 0.1 )</td>
<td>( y_1 = 0.9 )</td>
<td>( y(0.1) = 0.925794646 )</td>
<td>2.79 %</td>
</tr>
<tr>
<td>( t_2 = 0.2 )</td>
<td>( y_2 = 0.852967995 )</td>
<td>( y(0.2) = 0.889504459 )</td>
<td>4.11 %</td>
</tr>
<tr>
<td>( t_3 = 0.3 )</td>
<td>( y_3 = 0.837441500 )</td>
<td>( y(0.3) = 0.876191288 )</td>
<td>4.42 %</td>
</tr>
<tr>
<td>( t_4 = 0.4 )</td>
<td>( y_4 = 0.839833779 )</td>
<td>( y(0.4) = 0.876283777 )</td>
<td>4.16 %</td>
</tr>
<tr>
<td>( t_5 = 0.5 )</td>
<td>( y_5 = 0.851677371 )</td>
<td>( y(0.5) = 0.883727921 )</td>
<td>3.63 %</td>
</tr>
</tbody>
</table>

We’ve also included the error as a percentage. It’s often easier to see how well an approximation does if you look at percentages. The formula for this is,

\[
\text{percent error} = \left| \frac{\text{exact} - \text{approximate}}{\text{exact}} \right| \times 100
\]

We used absolute value in the numerator because we really don’t care at this point if the approximation is larger or smaller than the exact. We’re only interested in how close the two are.

The maximum error in the approximations from the last example was 4.42%, which isn’t too bad, but also isn’t all that great of an approximation. So, provided we aren’t after very accurate approximations this didn’t do too badly. This kind of error is generally unacceptable in almost all real applications however. So, how can we get better approximations?

Recall that we are getting the approximations by using a tangent line to approximate the value of the solution and that we are moving forward in time by steps of \( h \). So, if we want a more accurate approximation, then it seems like one way to get a better approximation is to not move forward as much with each step. In other words, take smaller \( h \)’s.

**Example 2** Repeat the previous example only this time give the approximations at \( t = 1 \), \( t = 2 \), \( t = 3 \), \( t = 4 \), and \( t = 5 \). Use \( h = 0.1 \), \( h = 0.05 \), \( h = 0.01 \), \( h = 0.005 \), and \( h = 0.001 \) for the approximations.

**Solution**

Below are two tables, one gives approximations to the solution and the other gives the errors for each approximation. We’ll leave the computational details to you to check.
We can see from these tables that decreasing $h$ does in fact improve the accuracy of the approximation as we expected.

There are a couple of other interesting things to note from the data. First, notice that in general, decreasing the step size, $h$, by a factor of 10 also decreased the error by about a factor of 10 as well.

Also, notice that as $t$ increases the approximation actually tends to get better. This isn’t the case completely as we can see that in all but the first case the $t = 3$ error is worse than the error at $t = 2$, but after that point, it only gets better. This should not be expected in general. In this case this is more a function of the shape of the solution. Below is a graph of the solution (the line) as well as the approximations (the dots) for $h = 0.1$.

Notice that the approximation is worst where the function is changing rapidly. This should not be too surprising. Recall that we’re using tangent lines to get the approximations and so the value of the tangent line at a given $t$ will often be significantly different than the function due to the rapidly changing function at that point.
Also, in this case, because the function ends up fairly flat as $t$ increases, the tangents start looking like the function itself and so the approximations are very accurate. This won’t always be the case of course.

Let’s take a look at one more example.

**Example 3** For the IVP

$$y' - y = -\frac{1}{2} e^t \sin(5t) + 5e^t \cos(5t) \quad y(0) = 0$$

Use Euler’s Method to find the approximation to the solution at $t = 1$, $t = 2$, $t = 3$, $t = 4$, and $t = 5$. Use $h = 0.1$, $h = 0.05$, $h = 0.01$, $h = 0.005$, and $h = 0.001$ for the approximations.

**Solution**

I’ll leave it to you to check the details of the solution process. The solution to this linear first order differential equation is.

$$y(t) = e^t \sin(5t)$$

Here are two tables giving the approximations and the percentage error for each approximation.

### Approximations

<table>
<thead>
<tr>
<th>Time</th>
<th>Exact</th>
<th>$h = 0.1$</th>
<th>$h = 0.05$</th>
<th>$h = 0.01$</th>
<th>$h = 0.005$</th>
<th>$h = 0.001$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t = 1$</td>
<td>-1.58100</td>
<td>-0.97167</td>
<td>-1.26512</td>
<td>-1.51580</td>
<td>-1.54826</td>
<td>-1.57443</td>
</tr>
<tr>
<td>$t = 2$</td>
<td>-1.47880</td>
<td>0.65270</td>
<td>-0.34327</td>
<td>-2.18657</td>
<td>-1.35810</td>
<td>-1.45453</td>
</tr>
<tr>
<td>$t = 3$</td>
<td>2.91439</td>
<td>7.30209</td>
<td>5.34682</td>
<td>3.44488</td>
<td>3.18259</td>
<td>2.96851</td>
</tr>
<tr>
<td>$t = 4$</td>
<td>6.74580</td>
<td>15.56128</td>
<td>11.84839</td>
<td>7.89808</td>
<td>7.33093</td>
<td>6.86429</td>
</tr>
<tr>
<td>$t = 5$</td>
<td>-1.61237</td>
<td>21.95465</td>
<td>12.24018</td>
<td>1.56056</td>
<td>0.0018864</td>
<td>-1.28498</td>
</tr>
</tbody>
</table>

### Percentage Errors

<table>
<thead>
<tr>
<th>Time</th>
<th>$h = 0.1$</th>
<th>$h = 0.05$</th>
<th>$h = 0.01$</th>
<th>$h = 0.005$</th>
<th>$h = 0.001$</th>
</tr>
</thead>
<tbody>
<tr>
<td>$t = 1$</td>
<td>38.54 %</td>
<td>19.98 %</td>
<td>4.12 %</td>
<td>2.07 %</td>
<td>0.42 %</td>
</tr>
<tr>
<td>$t = 2$</td>
<td>144.14 %</td>
<td>76.79 %</td>
<td>16.21 %</td>
<td>8.16 %</td>
<td>1.64 %</td>
</tr>
<tr>
<td>$t = 3$</td>
<td>150.55 %</td>
<td>83.46 %</td>
<td>18.20 %</td>
<td>9.20 %</td>
<td>1.86 %</td>
</tr>
<tr>
<td>$t = 4$</td>
<td>130.68 %</td>
<td>75.64 %</td>
<td>17.08 %</td>
<td>8.67 %</td>
<td>1.76 %</td>
</tr>
<tr>
<td>$t = 5$</td>
<td>1461.63 %</td>
<td>859.14 %</td>
<td>196.79 %</td>
<td>100.12 %</td>
<td>20.30 %</td>
</tr>
</tbody>
</table>

So, with this example Euler’s Method does not do nearly as well as it did on the first IVP. Some of the observations we made in Example 2 are still true however. Decreasing the size of $h$ decreases the error as we saw with the last example and would expect to happen. Also, as we saw in the last example, decreasing $h$ by a factor of 10 also decreases the error by about a factor of 10.

However, unlike the last example increasing $t$ sees an increasing error. This behavior is fairly common in the approximations. We shouldn’t expect the error to decrease as $t$ increases as we saw in the last example. Each successive approximation is found using a previous approximation. Therefore, at each step we introduce error and so approximations should, in general, get worse as $t$ increases.

Below is a graph of the solution (the line) as well as the approximations (the dots) for $h = 0.05$. 
As we can see the approximations do follow the general shape of the solution, however, the error is clearly getting much worse as \( t \) increases.

So, Euler’s method is a nice method for approximating fairly nice solutions that don’t change rapidly. However, not all solutions will be this nicely behaved. There are other approximation methods that do a much better job of approximating solutions. These are not the focus of this course however, so I’ll leave it to you to look further into this field if you are interested.

Also notice that we don’t generally have the actual solution around to check the accuracy of the approximation. We generally try to find bounds on the error for each method that will tell us how well an approximation should do. These error bounds are again not really the focus of this course, so I’ll leave these to you as well if you’re interested in looking into them.