CALCULUS A, LAB 4
NEWTON'S METHOD

The purpose of this lab is to look at an elementary but nontrivial application of the derivative, with an eye to showing off some of the amazing power of the tools we now have. It's also slightly connected to linearization.

Newton's Method (a stupid name, since most of calculus is Newton's method) is a technique for approximating roots of equations you can't solve exactly. It's commonly used in computer algebra systems in things like *Maple's* **fsolve** command or *Sage's* **find_root** command, for instance. Newton's Method is an algorithm for taking a first approximation to the root of an equation and for producing a better second approximation. By using Newton's Method over and over, one can get increasingly accurate aproximations to the root you're seeking. I'll start out illustrating the process by approximating a root to the simple equation $x^2 - 2 = 0$, but we'll then see that the same method can be used to obtain roots of practically any equation. I'm only starting with a simple example to make the arithmetic easy and to pick a case where we can easily check the accuracy of our approximations.

Here's the idea. Suppose you want to find a lot of digits of $\sqrt{2}$, but you don't have a machine handy (or you're programming the machine to do this task). I learned in school how to take square roots by hand, but you probably didn't. What could you do?

You know that the number you're interested in, $\sqrt{2}$, is a root of the equation $x^2 - 2 = 0$, and that $1 < \sqrt{2} < 2$. A reasonable first approximation to $\sqrt{2}$ might therefore be $\sqrt{2} \doteq x_0 = 1$. Here's how Newton goes about getting a sequence of better and better approximations to $\sqrt{2}$.

1. (a) Compute the equation of the tangent line to the graph of $y = x^2 - 2$ at the point $x = x_0 = 1$.

(b) Sketch, either by machine or by hand, the graph of $y = x^2 - 2$ along with this tangent line. They had better be tangent.

(c) The curve $y = x^2 - 2$ crosses the x-axis at $x = \pm\sqrt{2}$. Is the point at which the tangent line crosses the x-axis a better or worse approximation to $\sqrt{2}$ than the original approximation $x_0 = 1$?

(d) Compute the position x_1 at which the tangent line crosses the x-axis. How far is this point from $\sqrt{2}$?

2. Now repeat this process starting from the point x_1 instead of with x_0.

(a) Compute the equation of the tangent line to the graph of $y = x^2 - 2$ at the point $x = x_1$.

(b) Sketch, either by machine or by hand, the graph of $y = x^2 - 2$ along with this tangent line. They had better be tangent.

(c) Is the point at which the tangent line crosses the x-axis a better or worse approximation to $\sqrt{2}$ than the approximation x_1?

(d) Compute the position x_2 at which the tangent line crosses the x-axis. How far is this point from $\sqrt{2}$?

Newton's Method is just this process of improving an estimate of a root by following the tangent line to the x-axis, repeating the process as many times as we like until we have as much accuracy as we need. We don't have to apply it to a simple curve like $y = x^2 - 2$; we could apply it to any function f for which we can compute values both of f and of f'.

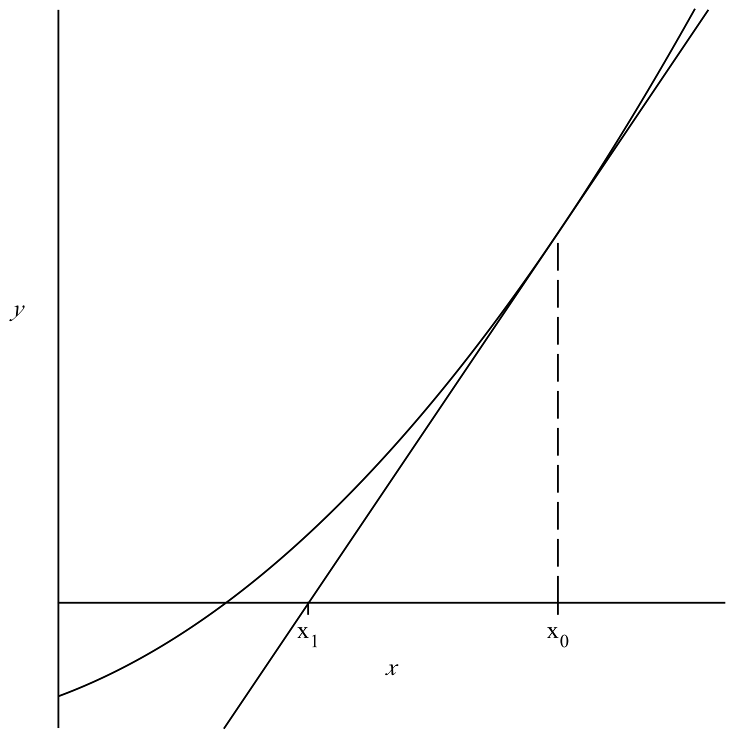

FIGURE 1. A tangent line near a root.

3. Figure 1 shows a curve $y = f(x)$, an approximation x_0 to a root of $f(x) = 0$, and the tangent line to the curve at the point $(x_0, f(x_0))$. Suppose we know the height $f(x_0)$ of the curve at $x = x_0$ and the slope $f'(x_0)$ of the tangent line. Write an equation for the tangent line in terms of the "constants" $f(x_0)$ and $f'(x_0)$, and show that the point at which this tangent line strikes the x-axis is

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}. \tag{1}$$

Equation (1) lets us give a quick, formal description of Newton's Method. To find a root to the equation $f(x) = 0$, start with an approximate solution x_0. Use Equation (1) to get a closer approximation, x_1, to the root. If x_1 is still not close enough, then repeat the process using x_1 as your new x_0. Keep repeating the process until you get as close as to the root as you need to be.

4. Use Newton's Method to approximate $\sqrt{2}$ again, starting again at $x_0 = 1$. This time, though, repeat the process a bunch of times, not just twice, getting x_1 and x_2, and x_3 and so on up to maybe x_7. The attached *Sage* worksheet shows one way to do this calculation to high precision.

(a) Are the estimates produced by Newton's Method larger or smaller than $\sqrt{2}$? Can you explain why geometrically?

(b) What can you say about the error of your estimates? You could try to be precise by imagining that your starting point is $x_0 = \sqrt{2} + \varepsilon$, so that your starting error is ε, and working out how big you expect the error of x_1 to be, but if this seems too algebraically daunting, then another question you might ask is, if x_0 has n correct digits, how many correct digits do you expect x_1 to have? Of course an algebraic argument would be here, but it is more important just to try to observe and to think.

(c) If you wanted to get 1000 correct digits of $\sqrt{2}$, about how many times would you have to apply Newton's Method starting at 1? What if you wanted 1,000,000 digits of $\sqrt{2}$? (Do this by thinking, not by setting

x0 = RealField(4000000)(1)

and trying it; though in fact, *Sage* can do that calculation in a few minutes. The only part that's hard turns out to be printing the output if you're using the Notebook interface. If you use *Sage* at the command line, then on my laptop, the whole calculation takes under 4 minutes.)

5. The equation $-x^6 - 5x^2 + 2x + 50 = 0$ has a root somewhere between $x = -2$ and $x = -1$. Use Newton's Method to approximate this root to 10 digit accuracy. Are your approximants larger or smaller than the actual root? Can you explain why?

6. What conditions would a function have to satisfy in the neighborhood of a root in order for the Newton's Method approximations to be smaller than the exact root? Larger? Words like "increasing," "decreasing," "concave up" and "concave down" may be useful in your answers.

7. The equation $4x^5 - 61x^3 - 480 = 0$ has a root between $x = -3$ and $x = -2$. Explain what happens if you try to approximate this root using Newton's Method starting at $x_0 = -3$. Is there a way around this?

8. The equation $\sqrt[3]{x} = 0$ has a root at $x = 0$. What would happen if you didn't know this already and if you tried to approximate this root using Newton's Method? Is there a way around this?

9. Can you say anything in general about when Newton's Method does and does not work? An answer with a geometric flavor is fine.

newton_precise

To carry out Newton's method at all, it makes sense to define a function **newton** that takes as arguments the function f to which we want to apply Newton's method and the point x_0 at which we want to start. We'll also define **f** to be the function $f(x) = x^2 - 2$, and just make sure **newton** is doing what we think it should.

```
def newton(f, x0):
   df = diff(f)
   return(x0 - f(x0)/df(x0))
```

```
f(x) = x^2 - 2
```

```
newton(f, 1)
```

```
3/2
```

```
newton(f, 3/2)
```

```
17/12
```

Now we need to tell *Sage* that we want to work with numbers with a lot of digits. One way to do that is to start out saying

x0 = RealField(400)(1)

This tells *Sage* that we want **x0** to be 1 with 400 bits of precision. 400 bits of precision means that calculations involving **x0** will be carried out with an error of at most $2^{-400} \approx 10^{-120}$. So we want **x0** to be 1 with 120 digits of accuracy.

```
x0 = RealField(400)(1)
x0
```

```
1.000000000000000000000000000000000000000000000000000000000000000000\
00000000000000000000000000000000000000000000000000000
```

To carry out Newton's method a couple of times with 120 digits of precision starting at $x_0 = 1$, do this:

```
x1 = newton(f, x0)
x1
```

```
1.500000000000000000000000000000000000000000000000000000000000000000\
00000000000000000000000000000000000000000000000000000
```

```
x2 = newton(f, x1)
x2
```

```
1.416666666666666666666666666666666666666666666666666666666666666666\
66666666666666666666666666666666666666666666666666667
```

How far are these estimates away from $\sqrt{2}$? To find out, we just subtract, except that we have to remember to tell *Sage* that we want $\sqrt{2}$ to be computed to 400 bit precision as well:

```
x1 - RealField(400)(sqrt(2))
```

```
0.085786437626904951198311275790301921430328124623051926823320262009\
2675215378929611496124656723584272649861537690877029748
```

```
x2 - RealField(400)(sqrt(2))
```

```
0.002453104293571617864977942456968588096994791289718593489986928675\
93418820455962781627913233902509393165282043575436964122
```

Problem 4 in the Lab asks you to explore how fast the errors decrease using Newton's method by computing x_3, x_4, x_5, x_6, x_7 and seeing how much each differs from $\sqrt{2}$. You should see a recognizable pattern in roughly how many digits accuracy you get at each step.