

## 1 Convergence of Newton's Method

As mentioned in class, when Newton's Method is applied with an initial guess reasonably close to a root of  $f(x)$ , it will home in on that root very quickly. We can check the progress of the iterations in two ways: 1) we can look at  $f(x_n)$  and stop when this is sufficiently close to zero, or 2) we can look at how much the  $x_n$  values are changing and stop when they don't change much. For approach (2), we can think of the Newton's Method formula as  $x_{n+1} = x_n - \Delta x_n$  where  $\Delta x_n = f(x_n)/f'(x_n)$ ; we stop iterating once  $|\Delta x_n|$  is small enough (which is very similar to looking at when  $|f(x_n)|$  is small enough).

The *MatLab* function `newtexample` on the course webpage shows how to implement these stopping criteria.

## 2 When Good Newtons Go Bad

In class we saw that Newton's Method can fail horribly. Obviously we have a problem if  $f'(x_n) = 0$  for any iterate  $x_n$ , but it is difficult to know, *a priori*, if this will happen. Also, we can hit problems if  $|f'(x_n)|$  is small (even if not actually zero).

However, iterates "blowing up" to infinity is not the only problem Newton's Method can encounter. Sometimes the iterates will just bounce around all over the place, not settling to any fixed value. For example, consider applying Newton's Method to  $f(x) = \sqrt{|x|}$ .

Note that if  $x > 0$ , then  $f(x) = \sqrt{x}$  so  $f'(x) = 1/(2\sqrt{x})$ . Hence,

$$x_{n+1} = x_n - \frac{\sqrt{x_n}}{1/(2\sqrt{x_n})} = x_n - 2\sqrt{x_n}\sqrt{x_n} = x_n - 2|x_n| = x_n - 2x_n = -x_n$$

Similarly, if  $x < 0$ , then  $f(x) = \sqrt{-x}$  so  $f'(x) = -1/(2\sqrt{-x})$ . And so, again,

$$x_{n+1} = x_n - \frac{\sqrt{-x_n}}{-1/(2\sqrt{-x_n})} = x_n + 2\sqrt{-x_n}\sqrt{-x_n} = x_n + 2|x_n| = x_n + 2(-x_n) = -x_n$$

(Note  $|x| = -x$  when  $x < 0$ .) So in either case, we have that  $x_{n+1} = -x_n$  and, consequently,  $x_{n+2} = -x_{n+1} = -(-x_n) = x_n$ . Hence, regardless of our choice of  $x_0$ , we will get that

$$x_1 = -x_0, x_2 = x_0, x_3 = -x_0, x_4 = x_0, x_5 = -x_0, \dots$$

So we never converge to any root ( $f(x)$  has only one root:  $x = 0$ ), but rather just jump back and forth between  $x_0$  and  $-x_0$ .

Clearly this example is special: it is chosen deliberately to be pathological. But even "harmless-looking" functions can exhibit some unexpected behavior which all demonstrates that Newton's Method, while working very well when  $x_0$  is close enough to the root, is not guaranteed to work well — or at all — in general. Moral: A good starting guess for  $x_0$  is very important!

## 3 A MatLab Function To Play With

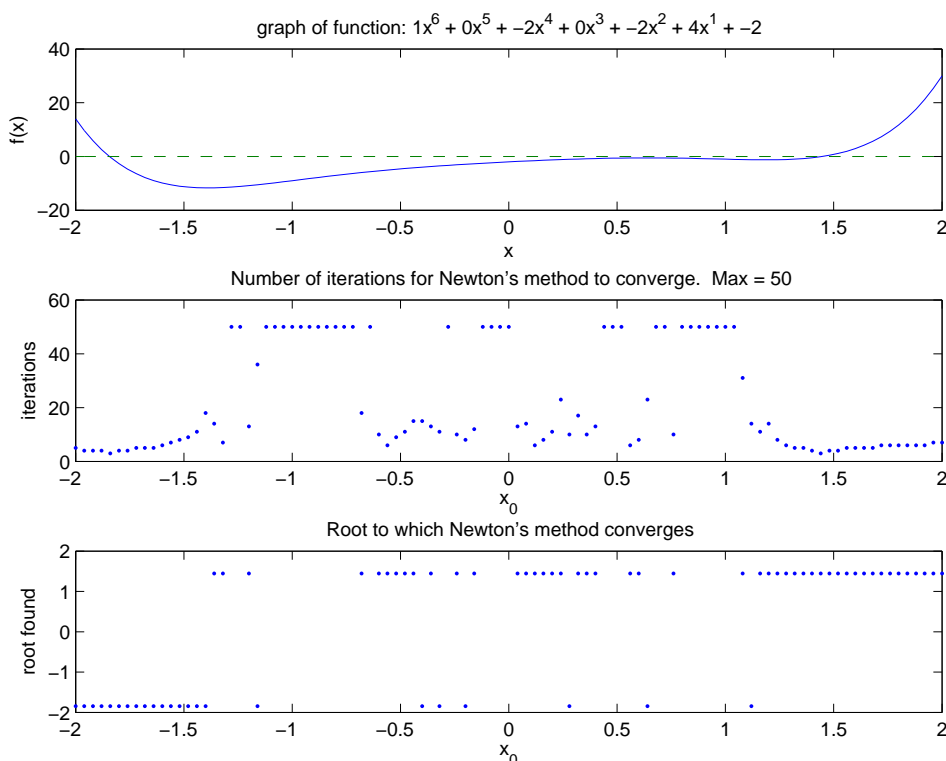
The *MatLab* function `newtexample` on the course webpage shows some interesting behavior of Newton iterates for the simple case of finding the roots of polynomials. This function takes an array of coefficients that represent a polynomial, an interval  $[a, b]$ , and some other optional arguments; it produces three graphs. The first graph is simply the graph of the function over the given interval. The second is a graph of how many iterations were needed for Newton's Method to converge (to within some given tolerance), as a function of  $x_0$ ; that is, the function takes each value of  $x$  in the interval  $[a, b]$  and uses it as  $x_0$  in Newton's Method, counting the iterations required to find a root, given that starting guess. The third graph shows which root the Newton iterations converge to, again as a function of starting guess  $x_0$ .

To use this function:

- download the file from the course webpage to a directory on your computer

- start *MatLab*
- at the top of the *MatLab* Command Window is a directory bar; use this to change directory to wherever you just saved the file
- at the prompt (>>) type `help newtexample` and hit the return key
- if the directory is set correctly, you should see some help text
- if so, you can use the function; the help can guide you, but as a first test, try: `newtexample([1,0,-2,0,-2,4,-2],[ -2,2]);`
- you can use the up arrow key to bring up the previous command(s), then you can edit the command and hit return to run a slightly different command
- play with it!

The example above produces the graph:



The first graph shows that this function ( $x^6 - 2x^4 - 2x^2 + 4x - 2$ ) has two roots, one near  $x = 1.5$  and one near  $x = -1.8$ . The second and third graphs show that starting with  $x_0 \in [-2, -1.5]$  converges to the root near  $-1.8$  in only a few iterations (as expected) and, similarly  $x_0 \in [1.2, 2]$  converges to the root near  $1.5$  very quickly also. However, some strange things happen in between! Looking at, say,  $x_0 \in [-0.5, 0]$ , we see that some values of  $x_0$  produce convergence to one root and some to the other (third graph); some don't converge at all (second graph where the number of iterations is 50, which is the maximum value allowed). Even when there is convergence to one root or other, there is no real pattern to how many iterations it will take — one value of  $x_0$  will produce an answer in just a few iterations, but a value only slightly different will take numerous iterations (or not converge at all). Clearly Newton's Method is very sensitive to the choice of  $x_0$ ! Furthermore, try zooming in on the region  $x_0 \in [-0.5, 0]$  by entering `newtexample([1,0,-2,0,-2,4,-2],[ -0.5,0],501);` The weirdness is not resolved by zooming in (or by using more points) — in fact, it gets worse!

Let's say this one more time: Newton's Method is very sensitive to the choice of  $x_0$ . One could almost say that it behaves quite "chaotically" ...