

Chapter 3

Limits and Derivative Concepts

- §1. Average Rate of Change
- §2. Using Tables to Investigate Limits
- §3. Symbolic Limits and the Derivative Definition
- §4. Graphical Derivatives
- §5. Numerical Derivatives
- §6. Newton's Method and the Bisection Method

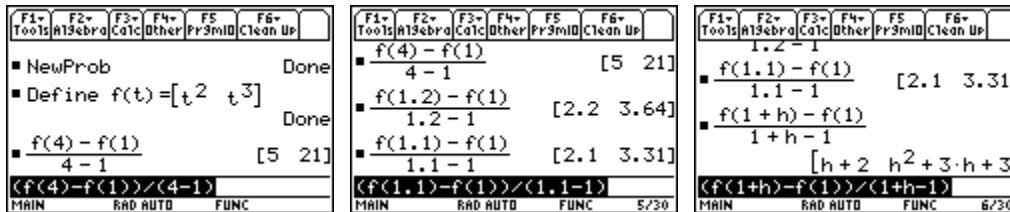
We consider in this chapter symbolic and numerical computations that motivate and define the concepts of limit and derivative. This is loosely coordinated with Chapter 3 of the text *Calculus with Early Vectors*, by Phillip Zenor, Edward Slaminka, and Donald Thaxton, Prentice Hall, 1999.

1. Average Rate of Change

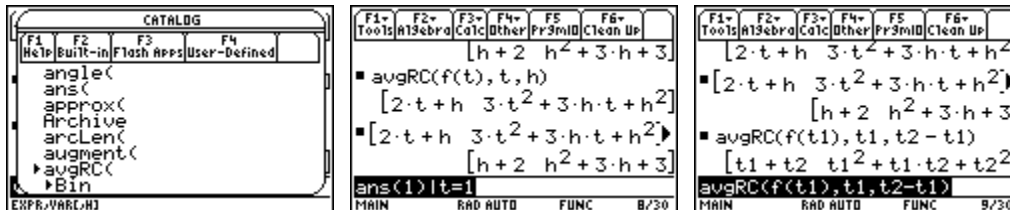
The concept of average velocity and average rate of change for a function $\vec{f} : \mathbb{R} \rightarrow \mathbb{R}^n$ involve the formula

$$\frac{\vec{f}(t_2) - \vec{f}(t_1)}{t_2 - t_1}$$

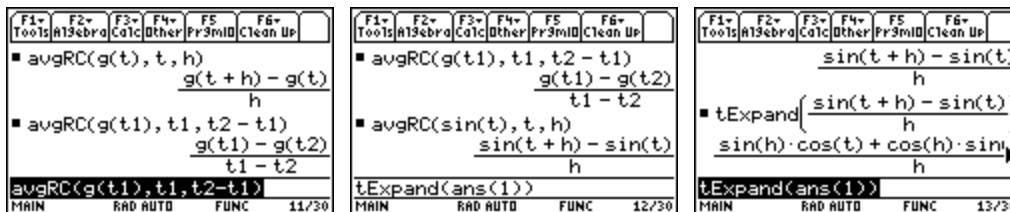
For a given vector-valued function, we can do numerical and symbolic computations directly with this formula.



After using the formula directly a few times, you might seek a shortcut. This is such an important concept, that the average rate of change is given as a command. The easiest way to find this command is to use the CATALOG (pressing the key with “A” above it to move to the part of the catalog with commands beginning with “A”).



As with many commands which we are just starting to use, we can make sure of what the command is doing by applying the command to undefined variables.

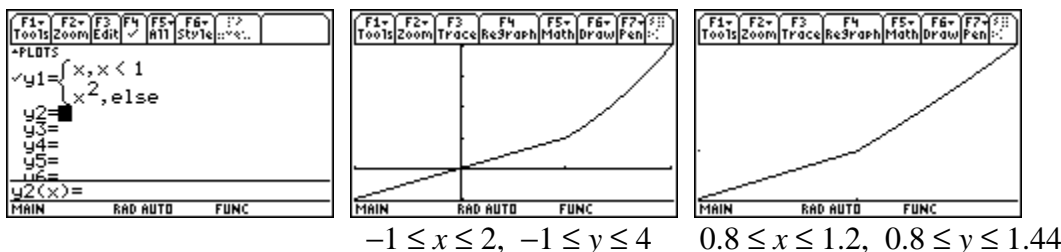


2. Using Graphs and Tables to Investigate Limits

The limit concept is first intuitively approached by looking at graphs. Consider the piecewise-defined function

$$f(x) = \begin{cases} x, & x < 1 \\ x^2, & 1 \leq x \end{cases}$$

where the most interesting question is what happens when x approaches 1.

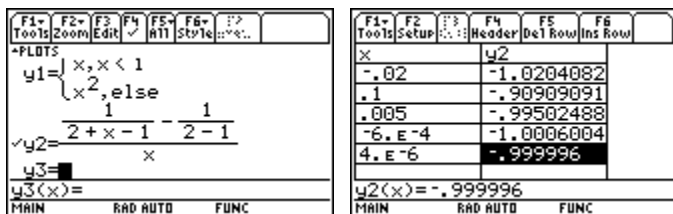


Thus there is graphical evidence that $\lim_{x \rightarrow 1} f(x) = 1$ and that this function is continuous. We can also explore this limit numerically in a table. For limits, it is best to select the “ASK” mode for the Table Setting involving the independent variable. Clear the Table of old results, and enter in x -values approaching 1 (both above and below).



The most interesting limits arise from taking the average rate of change over a smaller and smaller interval. E.g.

$$\lim_{h \rightarrow 0} \frac{f(x_0 + h) - f(x_0)}{h} \quad \text{where } f(x) = \frac{1}{x-1} \text{ and } x_0 = 2.$$



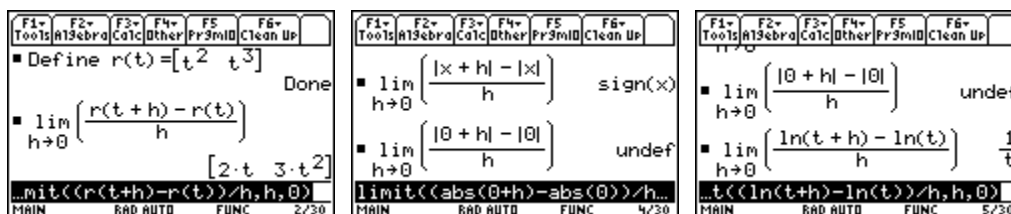
The numerical evidence seems to suggest that the limit here is about -1 .

3. Symbolic Limits and the Derivative Definition

In the Home screen, we can actually ask the calculator to find symbolic limits. It knows the patterns and rules to find many limits exactly. Appropriately the limit command is in the F3 Calculus menu (as well as in the CATALOG).



The optional last argument for the limit command gives one-sided limits (with the limit from the left if the number is negative and with the limit from the right if the number is positive). Highly interesting to us now are limits involving the definition of the derivative.

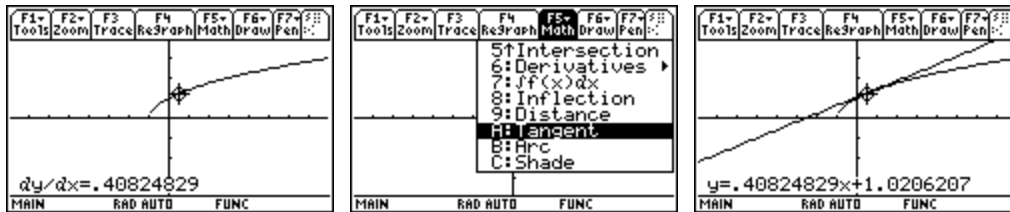


Notice that the calculator can handle limits of vector functions, by taking the limits component by component. The history indicates that a limit does not exist by the notation “undef”.

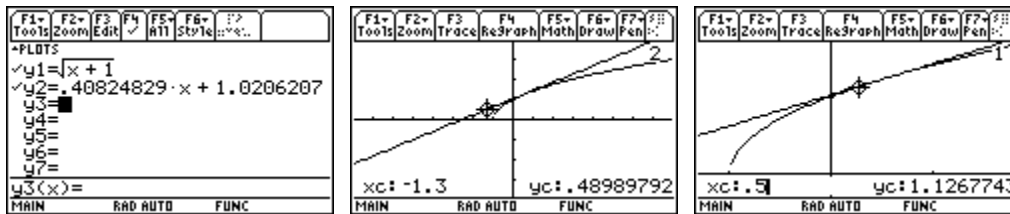
4. Graphical Derivatives

While looking at the graph of a function, we can evaluate the derivative at any point on the graph or have a tangent line drawn. We consider first real-valued functions.



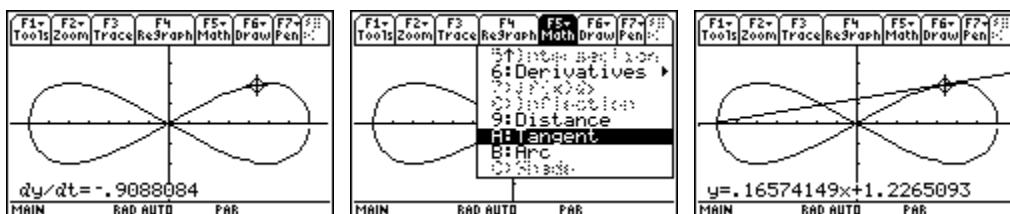
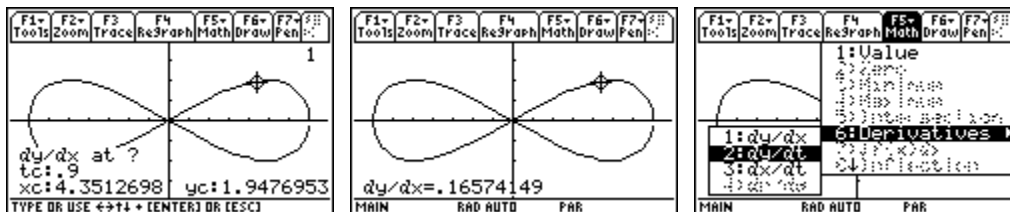
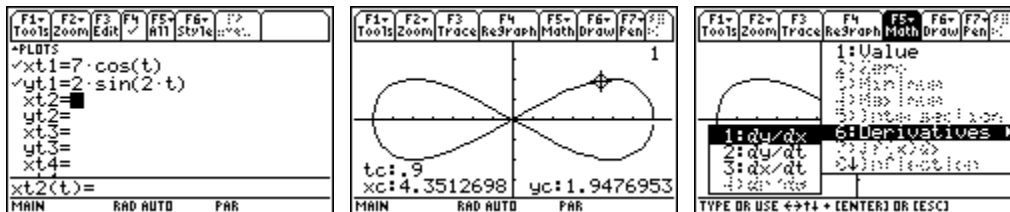


The tangent line produced by the F5 MATH command is a drawn object on the plot. Thus you cannot trace along the tangent line, and this line will disappear when you make any change to the graph (zoom, scroll, change the functions selected, or change the window). You would need to type the tangent line as a Y= function to get an active graph including it.

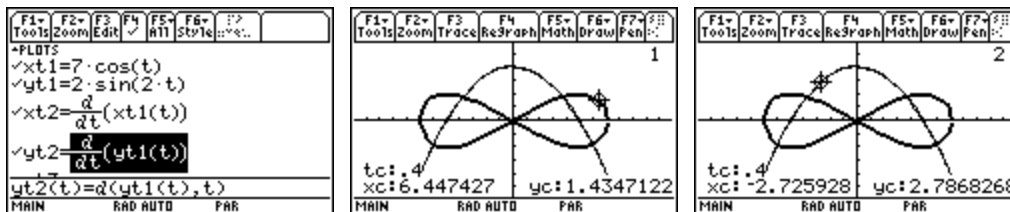


after a ZoomBox

We can also look at the derivatives for a vector-valued function with *two* components using parametric graphing.



If we actually enter the derivative parametric functions, we can trace on both to investigate the graphical meaning of the derivative for a vector-valued function.



5. Numerical and Symbolic Derivatives

From the derivative definition, we know that a derivative (if it exists) can be approximated by the following.

$$f'(x_0) \approx \frac{f(x_0 + h) - f(x_0)}{h}$$

When $h = r > 0$, this is called a *forward difference approximation* $\frac{f(x_0 + r) - f(x_0)}{r}$, and when

$h = -r < 0$, this is called a *backward difference approximation*

$$\frac{f(x_0 - r) - f(x_0)}{-r} = \frac{f(x_0) - f(x_0 - r)}{r}$$

The command “averRG(f(x),x,h)” gives us a convenient way to compute these. In general, one of these approximations will tend to be larger than the derivative and the other will tend to be smaller. Thus it is reasonable to expect the average of the two to be a more accurate approximation. [See p. 120 in Zenor, et. al. for another explanation and exercises 13-16.]

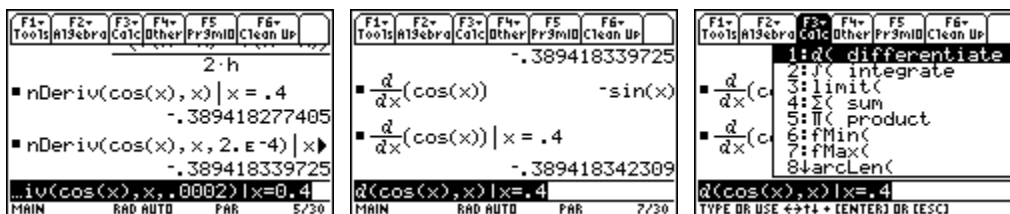
Centered Difference Approximation

$$\frac{1}{2} \frac{f(x_0 + h) - f(x_0)}{h} + \frac{1}{2} \frac{f(x_0) - f(x_0 - h)}{h} = \frac{f(x_0 + h) - f(x_0 - h)}{2h}$$

This numerical approximation is provided by the F3 Calc command A : nDeriv(Note that all of these derivative approximations can always be computed, even when the derivative does not exist (or even later when there is no rule to find the derivative).



Default $h = 0.001$



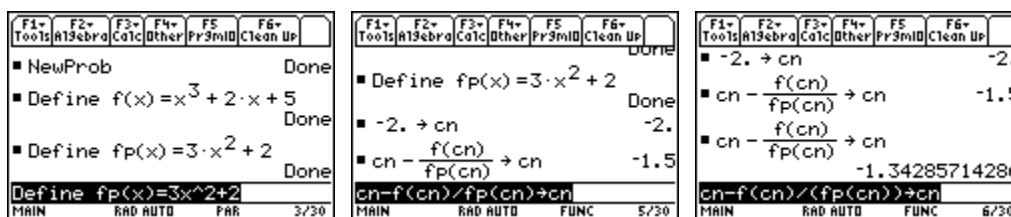
The symbolic derivative operation is provided in the F3 Calc menu by the command 1 : d(differentiate. Note that you can also get this command on the keyboard (2nd “8”) and that this is a special “slanted” character on the command line, which is different from the letter “d”. We will explore the symbolic derivative command more in the next chapter.

6. Newton’s Method and the Bisection Method

A very common task is to solve the equation $f(x) = 0$. We call the solutions the *zeros* or the *roots* of the function f . We have already considered in the introductory chapter how to solve equations symbolically in the home screen, numerically in the numeric solver application, and numerically and graphically in the graph screen. The two methods explored here are numerical methods that have a long history. They show very nice applications of the concepts of the derivative and the concept of continuity. While the built-in numeric solver is a little more sophisticated than these, it works in a similar fashion.

We certainly could implement Newton’s method in a program (and you are encouraged to do so if you chose). This is more of an educational activity than a truly needed process (since the numeric solver does this very easily). Thus we will look at how to do this in the home screen, and then we will save our work in a text file for later reference. Key to the success of the method is a *good* starting value c_0 (perhaps you should look at a graph of f if this is not a homework or test question with a specified starting value). The iterative formula for the next approximation to

the zero is $c_{n+1} = c_n - \frac{f(c_n)}{f'(c_n)}$.



Just press ENTER

Define the function with one variable name and the derivative of the function with another variable name. If you need to do so, you could let the calculator find the derivative symbolically. Make sure the Exact/Approx Mode is set to “AUTO” and that you type the starting value using a decimal point so that you get floating-point numbers (rather than exact results). Usually we will also want to see lots of digits, so the Display Digits Mode set to “FLOAT” is nice. After you have typed the command line “ $cn - f(cn)/fp(cn) \rightarrow cn$ ” once, you can simply press ENTER repeatedly to have the same command redone (each time using the most recently stored value for the variable “cn”). When things converge, you will eventually get virtually no change in the result. (*No change* does not guarantee that you have an exact zero, but it does imply that the “correction” in the next step of Newton’s method is so small that we cannot see a change.)



We save the work in the history section by the command F1 2: Save Copy As. The dialog box will ask you to provide a name for the resulting text file. Here we give it the name “newton”. This process will save the work, even if you clear the history area.

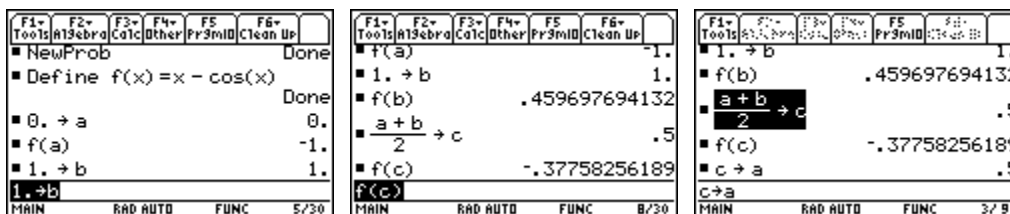
Press the APPS key and select the Text Editor. Choose to open the text we named “newton”. In this Text Editor, we can edit the file to change the starting value or to change the function and its derivative. Press F3 View to choose a Split view so that we can see both this text file (at the top) and the home screen (at the bottom). Move the cursor to some command line (with a beginning C:) and press F4 Execute to paste that command into the home screen and have it executed.



Respond with ESC

Repeatedly press F4

In the Bisection Method (also called the Interval-Halving Method), we do not need to know the derivative of the function. We simply need to know the function is continuous on some subinterval $[a, b]$ and that the value of the function at one endpoint is positive and the value at the other endpoint is negative (thus there must be a zero in between by the Intermediate Value Theorem). We compute the midpoint c of the subinterval and evaluate the function there. Depending upon the sign of $f(c)$, we select either $[a, c]$ or $[c, b]$ as the new subinterval that must contain the zero. Then we repeat. Again for educational purposes, we just implement this in the home screen and “save a copy” of the work as a text file for repeated use.



From this point on (after you have computed c once and decided whether it should replace a or b), you can simply cursor up in the history to get a recent command that you want to repeat. Certainly if you have to do the Bisection Method very many times, you will want to consider writing a program. In the program, the “If ... Then ... Else ...” structure is very nice for making the decision about whether c replaces a or b before you repeat.

We finish this section by noting that some of the features and characteristics of Newton's method and the Bisection Method are present in the numeric solver available as an application. If you type a "good" starting value into the variable in the numeric solver before pressing F2 Solve, you effectively tell the internal algorithm to start its iteration with the given value. This can help it to converge more rapidly, much as in Newton's method. There is also the option to specify an interval (the default bound ={-1.E14,1.E14} is effectively no bound at all). The internal algorithm stays within the interval given, much as in the Bisection method.