MATH 1351 TI-85 EXERCISE VIII
Graphing derivatives

Name: _____ SID: _____

The relationship between a function and its' first and second derivatives is often central to the use of these concepts in practice. Thus it might be nice to view the graphs of all three functions on the same axes. There are several ways to graph a function **y = f (x)** together with its' first derivative, **y = f'(x),** and its' second derivative, **y = f''(x)**, on the TI-85. We will explore these three ways by considering a very simple example:

f (x) = x^2 - 2x + 1.

(For each of the following exercises use the **SimulG** and **DrawLine** options from the **FORMT** menu. This lets you "see" the relation between the function and its' derivatives as the points are being plotted. Of course you have to figure out which is which.)

Way 1. Define **y1 = f (x)** , compute the derivatives directly (by hand) and define **y2 = f'(x)** and **y3 = f''(x)** and graph them.

In our example this means **y1 = x^2 - 2x + 1, y2 = 2x - 2** and **y3 = 2.** Sketch the resulting graphs in the RANGE **-3.15 ≤ x ≤ 3.15** and **-5 ≤ y ≤ .5.**

What is the relation between the sign of y2 and the behavior of the graph of y1?_____

_____

How is y3 related to y2? _____

Way 2. Define **y1 = f (x), y2 = nDer(y1,x),** and **y3 = nDer(y2,x).** (nDer stands for *numerical derivative* and is found in the Calculus menu which is accessed by choosing **2nd CALC** from the keyboard. So y3 is the derivative of the derivative of y1, hence the second derivative of y1.)

In our example, again **y1 = x^2 - 2x + 1**, **y2 = nDer(y1,x)**, and **y3 = nDer(y2,x)**. (Notice that this has the advantage of allowing us to change the function y1 and y2 will automatically be the derivative of the new y1 and y3 its' second derivative.) The graphs should be the same as above. If not you did something wrong, check it out.

Way 3. Define **y1 = f (x), y2 = der1(y1,x)** and **y3 = der2(y1,x).** (der1 stands for the first derivative and der2 stands for the second derivative. Both are found in the same place as nDer.)

In our example, again **y1 = x^2 - 2x + 1, y2 = der1(y1,x)** and **y3 = der2(y1,x)**. (Again this has the advantage of allowing us to change the function y1 and y2 will automatically be the first derivative and y3 the second derivative of the new y1.) The graphs should be the same as above.

If not you did something wrong, check it out.

This begs the question: "What's the difference between the programs **nDer** and **der1**? The answer is a bit technical, but is needed to explain several apparent deficiencies we may observe in using these packages. The program **nDer** computes a very particular difference quotient to obtain a numerical approximation to the derivative:

nDer(f(x),x) = (f (x + *delta*) - f (x - *delta)*) / 2 *delta*

for a numerical value of ***delta*** which we can set. (The default value of ***delta*** is .001. To check the setting in your machine choose **2nd TOLER** (short for tolerance) from the keyboard. If ***delta*** = .001, go back to GRAPH, otherwise set ***delta*** to be .001 by *arrowing down* to it and changing it to .001.)

The **der1** program apparently looks through some stored set of functions for a formula for the derivative of **f(x)** and, if it can find such a formula, uses it to compute derivative.

There are advantages and disadvantages to each program. If **der1** can't find a stored formula it will not work, while **nDer** works on any function you can define. However **nDer** can give some very misleading; indeed, flat wrong answers. To illustrate let's look at some examples we can analyze by hand and then see what the machine does with each package.

Define **y1 = abs(x-1) + 1.**

What is the value of dy1/dx (x) for any value of x < 1 ?_____

What is the value of dy1/dx (x) for any value of x > 1 ?_____

What is the value of dy1/dx (x) for the value x = 1? _____

Graph **y2 = der1(y1,x)** in the ZDECM window**.** Sketch this graph.

Use TRACE to find the value of y2 corresponding to each of the following values of x:
x = .8 -> y = __ ; x = .9 -> y = __ ; x = 1 -> y = __ ; x = 1.1 -> y = __ ; x = 1.2 -> y = ___

For comparison graph **y2 = nDer(y1,x)** in the same window. The graph should look the same; however there is a significant difference in the TI-85 picture. Again use TRACH to find the value of y2 corresponding to each of the following values of x:
x = .8 -> y = __ ; x = .9 -> y = __ ; x = 1 -> y = __ ; x = 1.1 -> y = __ ; x = 1.2 -> y = ___

Explain why **nDer** "thinks" the derivative of this function at x = 1 is 0.(We know it is not defined there because the left and right limits of the DQ don't match up at x = 1.) _____

_____

A more interesting wrong response for **nDer** is illustrated by the following example: **y1 = (1/x)^2**. Notice that **nDer** thinks the derivative at the x = 0 is 0 even though the function is not even

defined at 0. In fact if anything the derivative a 0 should be infinity since the graph is asymptotic to the y axis there. Notice that **der1** correctly yields the derivative of y1 undefined at x = 0.

In fact if **f** is any <u>even function</u>, then **nDer** thinks that **f'(0) = 0.** Why? _____

_____

Sometimes **der1** gives no information at all. For example where is the derivative of **y = int (x)** defined? _____ And what is its value where it is defined? _____ Try graphing y1 = int (x) and y2 = der1(y1,x). What happens? _____ Now try graphing y2 = nDer(y1,x)? (Use DrawLine format.) What happens? _____ Sketch the graph of y2 = nDer(y1,x). Explain this curious graph._____

_____

Where are the "spikes?' _____

How high up do they go? _____ Why? _____

_____

For each of the following functions sketch the graph of the function and its' first and second derivatives on the same axes. Label each.

1. **f (x) = x^3 - 3 x^2 + 2**

2. f (x) = x / (x + 1)

3. f (x) = (x^3 - 1) / (x^2 - 4)