

Guessing Limits Numerically

Michael Penna, Indiana University – Purdue University, Indianapolis

Objective

To guess the limit of a function at a point numerically.

Narrative

Prior to having the theorems on limits at our disposal, there are two major issues surrounding the limit of a function at a point: The first is guessing what the limit is, if it even exists; this issue can often be approached either graphically or numerically. The second issue involves proving that the guess you made is correct; this issue involves using the formal definition of limit.

In this project we address the issue of guessing limits numerically. We also illustrate Mathematica's **For** loop: The general syntax of a **For** loop in Mathematica is:

For[start condition, end condition, increment condition, {body}]

For example, the following code initializes **TheSum** to 0, and then increments the value of **TheSum** by **n** = 1, 2, 3, 4. The condition **n<5** tells Mathematica how long to continue incrementing, and the notation **n++** tells Mathematica to increment **n** by 1 on each pass through the loop.

```
In[1]:= TheSum=0
Out[1]:= 0
In[2]:= For[n=1, n<5, n++, {TheSum = TheSum+n, Print[TheSum]}]
Out[2]:= 1
          3
          6
          10
```

Tasks

1. Type the command lines in the left-hand column below into Mathematica in the order in which they are listed. These commands will help you estimate $\lim_{x \rightarrow 0} \frac{1 - \cos x}{x^2}$ numerically, if it exists.

```
In[1]:= (* Your name, today's date *)
In[2]:= (* Guessing Limits Numerically *)
In[3]:= f[x_]:= (1-Cos[x])/x^2
In[4]:= Plot[f[x], {x,-1,1}]
In[5]:= f[0]
In[6]:= For[n=1, n<7, n++,{Print[n, " ", N[f[1/2^n]], 6]}]
In[7]:= For[n=1, n<7, n++,{Print[n, " ", N[f[-1/2^n]], 6]}]
```

Let $f(x) = (1 - \cos x)/x^2$.
Plot the graph of f .
What is $f(0)$? (You should get an error message!)
Let's first look at some values of $f(x)$ for $x > 0$.
Now let's look at some values of $f(x)$ for $x < 0$.

At this time make a hard-copy of your typed input and Mathematica's responses. Then:

2. On the graphic you created in Task 1, plot the points on the graph of f whose y values you computed in the two **For** loops. (The easiest way to do this is to plot x values along the x -axis and then draw lines parallel to the y -axis through these points, until they meet the graph of f .)

3. On the basis of this data, do you think $\lim_{x \rightarrow 0} \frac{1 - \cos x}{x^2}$ exists? If so, what do you think it is (to 4 decimal places of accuracy)? Justify your answer.

Your lab report will be a hard copy of your typed input and Mathematica's responses, and your written responses.

Comments

1. You can *guess* whether or not $\lim_{x \rightarrow 0} \frac{1 - \cos x}{x^2}$ exists, and if it does exist, what its value is, on the basis of numerical "evidence" (as we did in this project), but you *cannot say for sure* that you're correct: you can never perform more than a finite number of computations, and however close x is to 0, you may *miss* some critical behavior of $f(x) = \frac{1 - \cos x}{x^2}$ that might affect your guess. It is because of this that we *must* turn to the formal concept of the limit.
2. Different rates of convergence can be achieved by replacing $1/2^n$ by $1/n^2$ (this produces a slower rate of convergence) or $1/n^n$ (this produces a faster rate of convergence).
3. The physical limitations of your computer may limit the accuracy of your computations.
4. Mathematica has a built in **Limit** function that allows you to compute limits automatically. Since we are interested not just in what limits are, but how they are computed, we intentionally avoided using this command in this project.
5. At the end of a **For** loop, the counter will have the last value assigned to it. For example, at the end of the loop

```
For[n=0, n<6, n++, {Print[n, " ", Factorial[n]]}]
```

the counter **n** will have the value 5. You can verify this by entering the code

```
In[1] := For[n=0, n<6, n++, {Print[n, " ", Factorial[n]]}]
In[2] := n
```

Thus if you want to reuse **n** as a variable in subsequent code, you will need to clear its value using:

```
In[3] := Clear[n]
```

6. In addition to **For** loops, Mathematica has **Do** loops and **While** loops. See Mathematica's documentation for more information on these.