The Definition of Limit

Michael Penna, Indiana University – Purdue University, Indianapolis

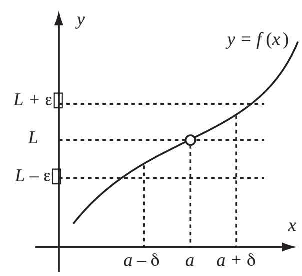
Objective

To investigate the precise definition of limit.

Narrative

To prove that the number L which we *guess* to be the limit of $f(x)$ at $x = a$, *really is* the limit of $f(x)$ at $x = a$, we must verify the condition in the formal definition of limit. This condition requires that for each real number $\epsilon > 0$, there is a real number $\delta > 0$ such that the values of $f(x)$ for all x in the interval $(a - \delta, a + \delta)$ — except possibly at $x = a$ itself — lie between $L - \epsilon$ and $L + \epsilon$; that is,

$$0 < |x - a| < \delta \Rightarrow |f(x) - L| < \epsilon.$$



In this project we investigate the graphical implications of this condition.

In this project we also illustrate "delayed plotting" using the `plots` package command `display`, and how to draw a line segment from the point $P(a, b)$ to the point $Q(c, d)$ using the command "<segment name> := [[a,b],[c,d]]".

Task

1. Type the command lines below into Maple in the order in which they are listed. These commands are concerned with $\lim_{x\to 2}(-x^3/12 + x^2/2 + 5/3)$. Note that in response to the command "`with(plots):`", Maple will issue the warning "`Warning, the name changecoords has been redefined`"; warnings such as this are perfectly natural. (On the other hand, error messages such as "`Error ...`" are not! A message such as this is telling you that a mistake has been made and that must be corrected before going on!) Also note that we terminate the "`plot0 := ...`" and "`plot1 := ...`" commands with a colon "`:`" rather than a semicolon "`;`", suppressing the immediate display of these plot structures; later we display them using the `plots` command `display`.

```
> # Your name, today´s date
> # The Definition of a Limit
> # Task 1
> restart;
> with(plots):
> f := x -> -x^3/12+x^2/2+5/3;
> a := 2.0;
> L := limit(f(x),x=a);
> xeqa := [[a,L-2],[a,L+2]];
> plot0 := plot({f(x),L,xeqa},x=a-2..a+2,y=L-2..L+2,color=blue):
> display(plot0);
> e := 0.5;
> plot1 := plot({L-e,L+e},x=a-2..a+2,y=L-2..L+2,color=red):
> display({plot0,plot1});
```

Observe that Maple uses different scales on the x- and y-axes to improve the appearance of the graphics. If you want to use the same scales on both axes you could use the "`scaling=constrained`" option to over-ride this feature: this option forces Maple to use the same units on the x- and y-axes.

2. Continue by typing the command lines below into Maple in the order in which they are listed.

```
> # Task 2
> e := 0.2;
> plot1 := plot({L-e,L+e},x=a-2..a+2,y=L-2..L+2,color=red):
> display({plot0,plot1});
```

At this time make a hard-copy of your typed input and Maple's responses. Then on the second graphic you produced in Task 1 and on the graphic you produced in Task 2:

3. Label by hand the graphs of $y = f(x)$, $y = L$, $y = L \pm e$, and $x = a$. Estimate by eye and state a value of `d` for which the values of `f(x)` for all `x` in the interval `a-d..a+d` — except possibly at $x = a$ — lie between `L-e` and `L+e` when `e = 0.5`. And finally draw the lines whose equations are `x = a+d` and `x = a-d` by hand, and highlight that part of the graph of $y = f(x)$ for which $x \in (a - \delta, a + \delta)$.

Your lab report will be a hard copy of your typed input and Maple's responses (both text and hand-labeled graphics).

Comments

1. In this project we are *not* actually proving that $L = \lim_{x \to a} f(x)$. On one hand, we are just verifying that an appropriate `d` exists for *two* given `e`'s: to verify that $L = \lim_{x \to a} f(x)$, we would have to do this *for every* `e`, not just two, three, four, or any finite number of `e`'s. On the other hand, since Maple draws the graphs of functions by "connecting-the-dots", some significant behavior could occur *between* the dots that is not revealed by Maple, so we cannot trust Maple's graphics to be completely accurate. This is one of the big reasons the $\epsilon\delta$-analysis of limits is so important.
2. You have to tell Maple you want to use a package of routines, such as `plots`, by saying "`with(plots)`" only once at the beginning (or right after the `restart`) of a session — not every time you use a routine in the package.

COPYRIGHT ©2005 Brooks/Cole, a division of Thomson Learning, Inc.