MATH 210: 10-18 WORKSHEET

L'Hôpital's rule (originally discovered by Johan Bernoulli) lets you compute limits of fractions that gave an indeterminate form of $0/0$ or ∞/∞.

$$\lim_{x\to a}\frac{n(x)}{d(x)} = \lim_{x\to a}\frac{n'(x)}{d'(x)},$$

provided either $\lim_{x\to a} n(x) = \lim_{x\to a} d(x) = 0$ or both limits are $\pm\infty$. It also works for one-sided limits, or if a is $\pm\infty$.

(1) Use L'Hôpital's rule to compute the limit

$$\lim_{x\to\pi}\frac{\sin x}{\ln(x/\pi)}.$$

(2) To use the limit definition of the derivative to work out the rule for the derivative of $\sin x$, we needed to know the two limits

$$\lim_{\Delta x\to 0}\frac{\sin(\Delta x)}{\Delta x} \qquad \text{and} \qquad \lim_{\Delta x\to 0}\frac{\cos(\Delta x)-1}{\Delta x}.$$

Use L'Hôpital's rule to compute these two limits. [Hint: it doesn't matter if we call the variable Δx or h or x or anything else for using L'Hôpital.]

(3) Use L'Hôpital's rule to compute the limits

$$\lim_{x\to\infty}\frac{2^x}{x}, \qquad \lim_{x\to\infty}\frac{2^x}{x^2}, \qquad \lim_{x\to\infty}\frac{2^x}{x^3}.$$

Figure out the general answer to

$$\lim_{x\to\infty}\frac{2^x}{x^n}, \qquad x \text{ is a positive integer.}$$

These calculations should support the idea that exponential functions grow infinitely faster than polynomials.

(4) L'Hôpital's rule also works for $0\cdot\infty$ indeterminate forms, by thinking of the multiplication as division by a reciprocal. Compute

$$\lim_{x\to 0^+} x\ln x$$

by rewriting the expression as $\frac{\ln x}{1/x}$ and using L'Hôpital's rule.

(5) Calculate

$$\lim_{x\to -\infty} x^2 e^x.$$

(6) Limits with other indeterminate forms, such as $\infty-\infty$, can be calculated by L'Hôpital's rule by first rewriting them as a fraction in the $0/0$ or ∞/∞ indeterminate forms. Use this idea to calculate

$$\lim_{x\to 0^+}\frac{1}{x^2}-\frac{1}{\tan x}.$$

(7) Show that

$$\lim_{x\to 1}\frac{x^2+5}{3x+4} = \frac{6}{7}.$$

In computer science, *Big O notation* is used to understand the runtime of programs. It gives a rough accounting of many steps it will take to run a program, based on the size of its input. Roughly, a program's runtime is $O(f(n))$ if it takes approximately $f(n)$ steps given an input of size n.[1] For example, the bubble sort[2] algorithm for sorting a list has $O(n^2)$ runtime while the merge sort[3] algorithm has a runtime of $O(n \log n)$.

In order for Big O analysis to be helpful, we need to know how to compare them. Is $O(n^2)$ slower or faster than $O(n \log n)$? Limits can be used to get a precise comparison: $O(f(n))$ is faster than $O(g(n))$ if

$$\lim_{n\to\infty} \frac{g(n)}{f(n)} = \infty \qquad \text{or equivalently, } \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0.$$

Intuitively, this says that if $O(f(n))$ is faster than $O(g(n))$, then for large input sizes n we expect to the first runtime to be much faster than the second runtime.[4] If you have some programming know-how you can test this yourself: write merge sort and bubble sort programs, then run them on large lists of random numbers. You'll see that the merge sort program will finish running much sooner than the buble sort program.

But enough about computer science, let's get back to calculus. Determine which runtimes of these pairs are faster using L'Hôpital's rule. For some of these, you may have calculated the limits already.

$$O(n^2) \text{ versus } O(2^n)$$

$$O(2^n) \text{ versus } O(n^k),\ k \text{ is a positive integer}$$

$$O(n^2) \text{ versus } O(n \log n)$$

$$O(n) \text{ versus } O(\log n)$$

$$O(\sqrt{n}) \text{ versus } O(\log n)$$

[1]More precisely, if there is a constant K so that for large enough n the number of steps is $\leq K \cdot f(n)$.

[2]`https://en.wikipedia.org/wiki/Bubble_sort`

[3]`https://en.wikipedia.org/wiki/Merge_sort`

[4]Somewhat confusingly, this comparison is known as "little o": we'd say that $f(n)$ is $o(g(n))$ if that limit fact holds.