Learning Objectives

By the end of this worksheet, you will:

- Prove and disprove statements using the definition of Big-Oh.
- Investigate properties of Big-Oh for some common families of functions.

Note: In Big-Oh expressions, it will be convenient to just write down the "body" of the functions rather than defining named functions all the time. We'll always use the variable n to represent the function input, and so when we write "$n \in \mathcal{O}(n^2)$," we really mean "the functions defined as $f(n) = n$ and $g(n) = n^2$ satisfy $f \in \mathcal{O}(g)$."

As a reminder, here is the formal definition of Big-Oh:

$$g \in \mathcal{O}(f): \ \exists c, n_0 \in \mathbb{R}^+, \ \forall n \in \mathbb{N}, \ n \geq n_0 \Rightarrow g(n) \leq cf(n) \qquad \text{where } f, g : \mathbb{N} \to \mathbb{R}^{\geq 0}$$

1. **Comparing polynomials.** Our first step in comparing different families of functions is looking at different powers of n. Consider the following statement, which generalizes the fact that $n \in \mathcal{O}(n^2)$:

$$\forall a, b \in \mathbb{R}^+, \ a \leq b \Rightarrow n^a \in \mathcal{O}(n^b)$$

(a) Rewrite the above statement by expanding the definition of Big-Oh.

there exists n_0 ,c in R_+, for all n in N, n >= n_0, ==> n^a <= cn^b

(b) Prove the above statement. **Hint**: you can actually pick c and n_0 to both be 1. Even though this is pretty simple, take the time to write the formal proof as a good warm-up for the rest of this worksheet.

Let a, b be arbitrary nums in R_+. Let c = 1, n_0 = 1.
Assume a <= b,
since n>= 1
n^a <= n^b

2. **Comparing logarithms.** One slight oddity about the definition of Big-Oh is that it treats all logarithmic functions "the same". Your task in this question is to investigate this by proving the following statement:[1]

$$\forall a, b \in \mathbb{R}^+,\ a > 1 \land b > 1 \Rightarrow \log_a n \in \mathcal{O}(\log_b n)$$

We won't ask you to expand the definition of Big-Oh, but if you aren't quite sure, then we highly recommend doing so before attempting even your rough work!

Hint: use the "change of base rule" for logarithms.

n >= n_0 ==> loga(n) <= c* logb(n)

[1] If you are concerned by the fact that $\log n$ is not defined at $n = 0$, you can replace $\log_a n$ with $\log_a(1 + n)$ in the above, and similarly with $\log_b$. We usually don't worry about this subtlety, since our concern is with the value of the functions for larger values of n. Picking an $n_0 > 0$ avoids the evaluation worry.

3. **Sum of functions.** Now let's look at one of the most important properties of Big-Oh: how it behaves when adding functions together. Let $f, g : \mathbb{N} \to \mathbb{R}^{\geq 0}$. We define the **sum of f and g** as the function $f + g : \mathbb{N} \to \mathbb{R}^{\geq 0}$ such that $\forall n \in \mathbb{N},\ (f + g)(n) = f(n) + g(n)$. For example, if $f(n) = 2n$ and $g(n) = n^2 + 3$, then $(f + g)(n) = 2n + n^2 + 3$.

   Consider the following statement:

$$\forall f, g : \mathbb{N} \to \mathbb{R}^{\geq 0},\ g \in \mathcal{O}(f) \Rightarrow f + g \in \mathcal{O}(f)$$

   In other words, if g is Big-Oh of f, then $f + g$ is no bigger than just f itself, asymptotically speaking.

   Your task for this question is to prove this statement. Keep in mind this is an implication: you're going to *assume* that $g \in \mathcal{O}(f)$, and you want to *prove* that $f + g \in \mathcal{O}(f)$. It will likely be helpful to write out the full statement (with the definition of Big-Oh expanded), and use subscripts to help keep track of the variables.

there exists c, n_0, n>= n_0 ==> g(n) <= c*f(n)   g(n) + f(n) <= cf(n) +f(n)

there exist c1, n1, n>= n1 ==> g + f <= c1* f(n)   c1 = c + 1, n1 = n_0

Assume g in O(f).
Want to show f+g in O(f).
let c, n_0 be such that n>= n_0 ==> g(n) <= c*f(n).
let c1 = c + 1, let n1 =n_0.
let n >= n_0, g(n) + f(n) <= c*f(n) + f(n) = (c+1)f(c) = c1*f(n).

Prove for all n in N n >= 1, g_1 .. g_n are all in O(f). Want to show g_1 + g_2 + …+ g_n in O(f)
- can use induction from what we just proved

Define P = U_{i = 1, inf} O(n^i)
g_1, g_2.. g_n in P , want to prove g_1 + g_2 .. + g_n in P
- find the biggest i = i', let n^i' be f from last question.

"the set P is closed under summation."

g_1 <= c1* n^i
g_2 <= c2* n^i2
…

c1* n^i + c2* n^i2…. <=