Recitation 1 - Asymptotic Notation

- CSE 241 introduces the science in Computer Science!
- Asymptotic notation is the language of that science.

Asymptotic notation is not as bad as you think it is!

Intuitively we just want to describe how different functions compare to each other. The following table is not exact, but it is a good analogy.

<table>
<thead>
<tr>
<th>Asymptotic Notation</th>
<th>Analogy</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f(n) = o(g(n)) )</td>
<td>( f(n) &lt; g(n) )</td>
</tr>
<tr>
<td>( f(n) = \Omega(g(n)) )</td>
<td>( f(n) \geq g(n) )</td>
</tr>
<tr>
<td>( f(n) = \Theta(g(n)) )</td>
<td>( f(n) = g(n) )</td>
</tr>
<tr>
<td>( f(n) = \omega(g(n)) )</td>
<td>( f(n) &gt; g(n) )</td>
</tr>
</tbody>
</table>

Note: see pages 51-52 of CRLS!

This table will let you organize the various definitions we discuss later.
Big-O Notation

We review the definition of $O(g(n))$ now. Before doing so I will point out that $O(g(n))$ is a set and as such we use set-builder notation.

Eg. - $S = \{ x | \exists k \text{ s.t. } x = 2k \}$

$S$ describes the set of all even integers.

If it is not clear to you why this is then please see a TA or the professor!

Def: $O(g(n)) = \{ f(n) | \exists \text{ constants } c > 0 \text{ s.t. } 0 \leq f(n) \leq c \cdot g(n) \text{ for all } n \geq n_0 \}$

Thus, an element $f(n)$ in the set $O(g(n))$ satisfies the definition on the right, and it is said that $f(n)$ is big-$O$ of $g(n)$. This is written $f(n) = O(g(n))$ but note that this is just shorthand for $f(n) \in O(g(n))$.

Intuitively, what does $O(g(n))$ capture? It means that the rate of change or growth of $g(n)$ is greater than $f(n)$.
Consider the graph of \( f(n) \) vs \( c \cdot g(n) \).

If \( f(n) = O(g(n)) \) it means that we can pick some constant \( c \) such that we can always find some point \( n_0 \) where every \( n > n_0 \) has that \( f(n) \leq c \cdot g(n) \).

Observe how \( c \cdot g(n) \geq f(n) \) for all \( n \geq n_0 \).
Little-o notation

We're going to move onto a notation you haven't seen in class yet: little-o.

Recall that $f(x) = O(g(x))$ is similar to $f(x) \leq g(x)$ and $f(x) = o(g(x))$ is similar to $f(x) < g(x)$.

The concepts are very similar, so pay attention to how the definitions of $O$ and $o$ differ.

3. **Def:** $o(g(n)) = \{ f(x) \mid \text{for all } c > 0, \exists \text{ a } n_0 > 0 \text{ such that } 0 \leq f(n) < c \cdot g(n) \text{ for all } n \geq n_0 \}$

Where do the definitions differ?

- $O$ has $\leq$ while $o$ has $<$
- $O$ has "some $c" while $o$ has "all $c"

$O$ is an existentially quantified statement, while $o$ is a universally quantified statement.

**Question:** Why does $o \Rightarrow O$? It should be clear from the definition above.
What's the difference?

Ex: Show $n = O(2^n)$

Recall, we must find some $c$ and $A_0$ such that $n \leq c \cdot 2^n$ for all $n \geq A_0$.

Pick $c = \frac{1}{2}$ and $A_0 = 1$.

Is it true that $n \leq \frac{1}{2} \cdot 2^n$ for all $n \geq 1$? Clearly!

Thus, $n = O(2^n)$

Ex: Show $n = o(2^n)$

Recall, for every $c$ we must find some $A_0$ such that $n \leq c \cdot 2^n$ for all $n \geq A_0$.

It's not possible!

Ex: Show $10n = O(n^2)$

Recall, for every $c$ we must find some $A_0$ such that $10n \leq c \cdot n^2$ for all $n \geq A_0$.

We can solve this algebraically and we see
Big-Theta Notation

Moving on- we won't look at \( \Omega \) and \( \omega \) but they are very similar to \( O \) and \( o \).

(3) Def: \( \Theta(g(n)) = \{ f(n) \mid \exists c_1, c_2, n_0 > 0 \text{ s.t.} \)
\[
0 \leq c_1 g(n) \leq f(n) \leq c_2 g(n) \text{ for all } n \geq n_0.
\]

The \( \Theta \) notation tries to capture the idea that two functions might grow at approximately the same rate.

Good practice question—does \( \Theta \) form an equivalence relation?

Look at the graphs again, suppose:

If \( f(n) = \Theta(g(n)) \) then for some \( c_1, c_2 \) we are guaranteed to have an \( n_0 \) that satisfies the inequality above.
Examples

Is \( n^2 + n = \Theta(n^2) \)?

We must show there is a \( c_1, c_2, n_0 \) such that
\[
0 \leq c_1(n^2) \leq n^2 + n \leq c_2(n^2)
\]
fors all \( n \geq n_0 \).

The problem can be broken into cases:

\( c_1(n^2) \leq n^2 + n \)
- choose \( c_1 = 1 \) and \( n_0 = 1 \) as these clearly satisfy the inequality.

\( n^2 + n \leq c_2(n^2) \)
- If we choose \( c_2 = 2 \) and \( n_0'' = 2 \)
  we again satisfy the inequality.

Hence, we choose \( c_1 = 1, c_2 = 2, n_0 = \max(n_0', n_0'') = 2 \).

These constants satisfy the definition so \( n^2 + n = \Theta(n^2) \). Graph the function to see the long term behavior.
Last remarks

We tried to understand asymptotic notation through the definitions and through the analogy:

<table>
<thead>
<tr>
<th>notation</th>
<th>analogy</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f(n) = o(g(n)) )</td>
<td>( f(n) &lt; g(n) )</td>
</tr>
<tr>
<td>( f(n) = \theta(g(n)) )</td>
<td>( f(n) \leq g(n) )</td>
</tr>
<tr>
<td>( f(n) = \Omega(g(n)) )</td>
<td>( f(n) = g(n) )</td>
</tr>
<tr>
<td>( f(n) = \omega(g(n)) )</td>
<td>( f(n) \geq g(n) )</td>
</tr>
</tbody>
</table>

This analogy relates functions by treating them like real numbers, but it is important to know where that analogy breaks down!

For real numbers \( a + b \), either:
- \( a < b \)
- \( a = b \)
- \( a > b \)

It is not true for any two functions that they are asymptotically relatible!

E.g.: \( f(n) = n^2 \) and \( g(n) = n^3 \sin(n) \)

No matter how far you go, \( n^3 \sin(n) \) will always