Inducción - Sumatorias y Productorias

Pablo L. De Nápoli

Departamento de Matemática
Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

Álgebra I - Primer cuatrimestre de 2020

Introducción - Sucesiones

En la clase de hoy vamos a ver un poco más en detalle como trabajar con sumatorias y productorias.

El símbolo de sumatoria ya lo introdujimos en la clase previa sobre inducción y los números de Fibonacci. Si no vieron, recomiendo que lo vean primero.

Concretamente en este video vamos a resolver el ejercicio 6 de la práctica 1. Sugiero que intenten resolverlo antes de ver este video.

Sucesiones

Comencemos recordando que una sucesión de números reales es una función $a : \mathbb{N} \to \mathbb{R}$. También podríamos considerar sucesiones de números complejos $a : \mathbb{N} \to \mathbb{C}$). $a_n = a(n)$ se denomina el n-ésimo término en la sucesión. Intuitivamente es como dar una lista infinita de números

$$a_1, a_2, a_3, \ldots, a_n, \ldots$$

Algunos ejemplos:

1. La sucesión de los números naturales $a_n = n$

$$1, 2, 3, \ldots, n, \ldots$$

2. La sucesión de los números pares $a_n = 2n$

$$2, 4, 6, \ldots, 2n, \ldots$$

3. La sucesión de los cuadrados $a_n = n^2$

$$1, 4, 9, 16, 25, \ldots, n^2, \ldots$$

4. La sucesión de Fibonacci F_n

Sumatorias

El símbolo de sumatoria $\sum$ proporciona una forma compacta y precisa de expresar sumas.´
Dada una sucesión (a_k) el símbolo

$$\sum_{k=i}^{n} a_k$$

representa la suma de los términos de la sucesión (a_k) donde el índice (o variable) k varia entre el valor inicial i y el valor final n, es decir: $i \leq k \leq n$.
Por ejemplo:

$$\sum_{k=1}^{n} k = 1 + 2 + 3 + \ldots + n$$

$$\sum_{k=2}^{n} k^2 = 2^2 + 3^2 + \ldots + n^2$$

Bonus track: programando una sumatoria

Como vimos en el video sobre números de Fibonacci, es posible escribir un programita para calcular sumatorias en forma recursiva. Otra forma de hacerlo es mediante un ciclo y una variable que acumule las sumas parciales
Por ejemplo, supongamos que queremos calcular la suma de los impares en un cierto rango

$$\sum_{i=1}^{n}(2i-1)$$

Sumando mediante un ciclo en Python 3

```
def ejemplo(n):
 s=0
 for i in range (1,n+1):
         s=s+(2*i-1)
 return s
```

Bonus track 2: La salida del programa

Ahora mediante otro ciclo podemos imprimir una tablita de valores:

Imprimimos una tablita mediante un ciclo en Python 3

```
for n in range(1,10):
 print("n=",n,"ejemplo(n)=",ejemplo(n))
```

Salida del programa

```
n= 1 ejemplo(n)= 1
n= 2 ejemplo(n)= 4
n= 3 ejemplo(n)= 9
n= 4 ejemplo(n)= 16
n= 5 ejemplo(n)= 25
n= 6 ejemplo(n)= 36
n= 7 ejemplo(n)= 49
n= 8 ejemplo(n)= 64
n= 9 ejemplo(n)= 81
```

Una conjetura

Eso nos lleva a conjeturar la fórmula

$$\sum_{i=1}^{n}(2i-1) = n^2$$

que aparece en el ejercicio 6 de la práctica 2, donde se indican diferentes formas de demostrarla.

Una manera de hacerlo es por inducción.

El Principio de inducción matemática

Tenemos una propiedad $P(n)$ que depende de un número natural $n \in \mathbb{N} = \{1, 2, 3, \ldots\}$. Técnicamente se denomina una función proposicional de n (tenemos una proposición para cada n). Podemos formular entonces el

Principio de Inducción Matemática

Si $P(1)$ es verdadera, y para todo $n \in \mathbb{N}$ se verifica que

$$P(n) \Rightarrow P(n+1)$$

entonces $P(n)$ es cierta para todo $n \in \mathbb{N}$.

En nuestro ejemplo

$P(n)$:

$$\sum_{i=1}^{n} (2i - 1) = n^2$$

El caso base

Demostremos $P(1)$. Si $n = 1$,

$$\sum_{i=1}^{n}(2i - 1) = 1$$

mientras que

$$n^2 = 1$$

luego la fórmula es válida cuando $n = 1$.

El paso inductivo

Suponemos que vale $P(n)$, queremos probar entonces que vale $P(n+1)$.

$$\sum_{i=1}^{n+1}(2i-1) = \sum_{i=1}^{n}(2i-1) + [2(n+1)-1]$$

usando la hipótesis inductiva esto es

$$= n^2 + [2(n+1)-1] = n^2 + 2n + 1 = (n+1)^2$$

Por el principio de inducción matemática, concluimos que la fórmula es cierta para todo n natural.

Otra manera de probarla

Un problema de la inducción es que es necesario saber de antemano (conjeturar) qué fórmula queremos demostrar.Otra manera de hacerlo es usando las propiedades de linealidad de la sumatoria

$$\sum_{i=1}^{n}(a_i + b_i) = \left(\sum_{i=1}^{n} a_i\right) + \left(\sum_{i=1}^{n} b_i\right) \quad \text{(propiedad asociativa)}$$

$$\sum_{i=1}^{n} c \cdot a_i = c \cdot \left(\sum_{i=1}^{n} a_i\right) \quad \text{(propiedad distributiva)}$$

y otra sumatoria ya conocido (el ejercicio 3 de la práctica 2):

$$\sum_{i=1}^{n} i = \frac{n(n+1)}{2}$$

En efecto,

$$\sum_{i=1}^{n}(2i-1) = 2\left(\sum_{i=1}^{n} i\right) - \left(\sum_{i=1}^{n} 1\right) = 2\frac{n(n+1)}{2} - n = n^2 + n - n = n^2$$

Productorias

De modo similar la las sumatorias, la productorias permiten expresar de forma compacta y precisa productos. El símbolo

$$\prod_{k=i}^{n} a_k$$

denota un producto de los elementos de la sucesión a_k donde k varía en el rango $i \leq k \leq n$.
Por ejemplo:

$$\prod_{k=1}^{n} a = a^n$$

$$n! = \prod_{k=1}^{n} k = 1 \cdot 2 \cdot 3 \cdots n \quad \text{factorial de } n$$

Bonus track 3: programando una productoria

Si queremos programar una productoria, podemos hacerlo como antes mediante un ciclo y acumulando los productos parciales. Por ejemplo la fórmula para el factorial

$$n! = \prod_{k=1}^{n} k$$

se programaría así

Multiplicando mediante un ciclo en Python 3

```
def factorial(n):
 p=1
 for k in range (1,n+1):
         p=p*k
 return p
```

Conclusión

- En la clase de hoy, vimos qué son sumatorias y productorias.
- Aprendimos a hacer una prueba por inducción con sumatorias.
- Vimos como relacionar una sumatoria con otra por medio de las propiedades de linealidad.
- Finalmente aprendimos a programar sumatorias y productorias mediante un ciclo y una variable acumulando resultados parciales.