Métodos Numéricos

Facultad de Ciencias Exactas y Naturales
Universidad de Buenos Aires

17 de Abril de 2020

WARNING: unexplored cuatrimestre ahead

Se viene un cuatri distinto fuera de nuestros labos

Un cuatrimestre best-effort pero poniendole ganas

Menú del día

- Introducción al labo
- Errores Numéricos y precisión finita
- Experimentación símil TP
- Presentación TP

Introducción al labo

Los docentes del laboratorio somos:

- Federico Pousa
- Juan Manuel Perez
- Nicolás Mastropascqua
- Nicolás San Martín

Introducción al labo

Los días de laboratorio van a ser los viernes de 18:00 a 22:00 hs, y trataremos de seguir el siguiente esquema de clases:

- de 18:00 a 18:30 hs: consultas,
- de 18:30 a 21:00 hs: clase y/o presentación de TPs y
- de 21:00 a 22:00 hs: consultas y/o resolución de ejercicios.

Introducción al labo

- 3 tps, grupos de 3 alumnos.
- 4 talleres.
- Los recuperatorios de los tps son durante el cuatrimestre.
- Vamos a ser estrictos con los horarios de entrega.
- Objetivos del labo.
- Posiblemente vamos a mencionar varias cosas nuevas, pregunten!
- Lenguajes que vamos a usar: C++, Python, LaTeX.

Recordatorios lejanos

¿Qué números conocemos?

- $\mathbb{N}_0$ naturales con el cero
- $\mathbb{Z}$ enteros
- $\mathbb{Q}$ racionales
- $\mathbb{R}$ reales
- $\mathbb{C}$ complejos

¿Cuáles podremos representar en una computadora?

Representación decimal de un número

Recordemos que, dado un $x \in \mathbb{R}$ podemos escribirlo como

$$x = \sum_{j=0}^{N} a_j 10^j + \sum_{j=1}^{\infty} b_j 10^{-j} \tag{1}$$

donde el primer término es la parte entera y el segundo la decimal

Representación binaria

Análogamente, podemos escribirlos con cualquier base. En particular, base binaria

$$x = \sum_{j=0}^{N} a_j 2^j + \sum_{j=1}^{\infty} b_j 2^{-j} \tag{2}$$

Acá podemos considerar a_j y b_j como bits prendidos o apagados.

Aritmética de la computadora

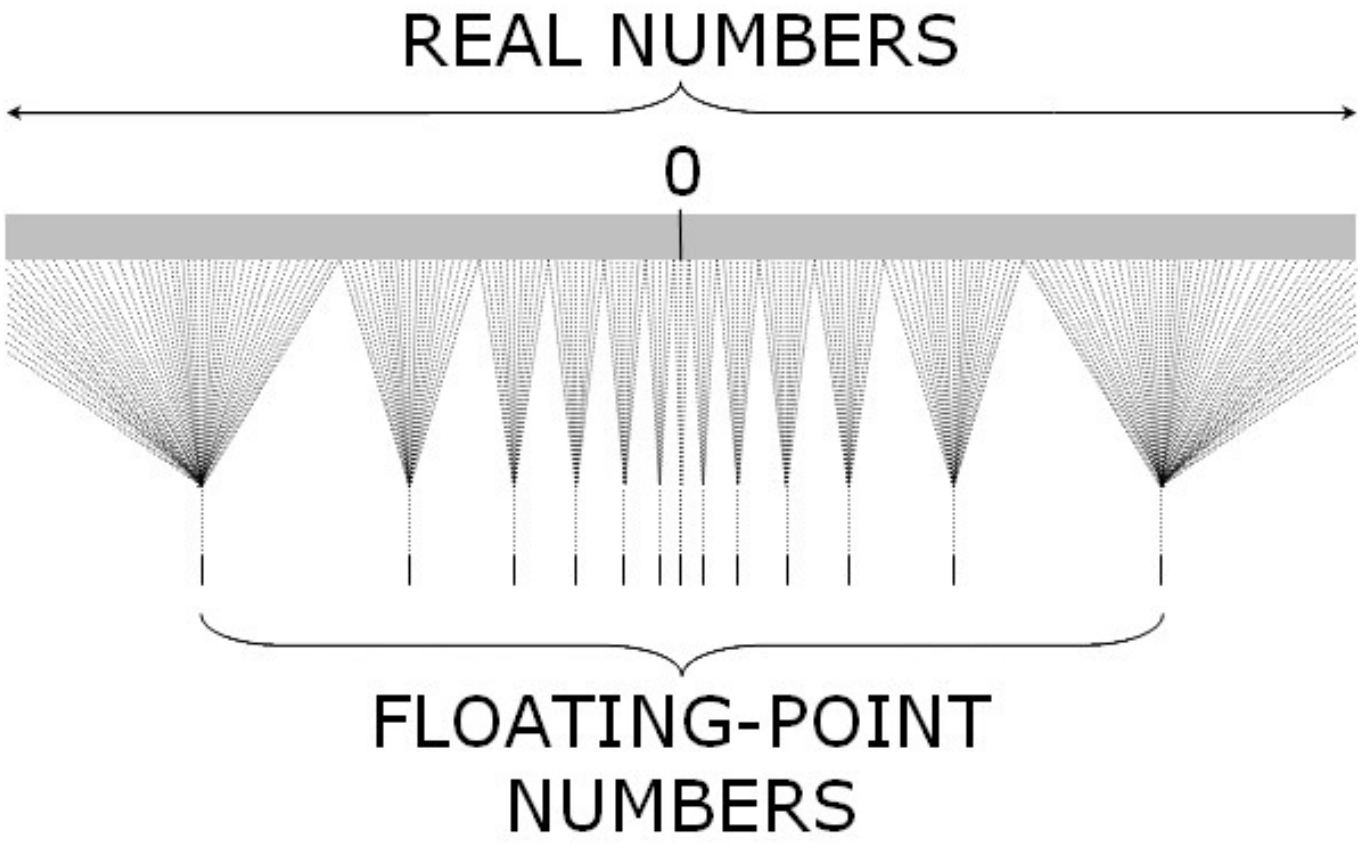
- Estándar IEEE-754 del 1985 para precisión *single* (32 bits) y *double* (64 bits)
- Sólo se pueden representar un subconjunto de $\mathbb{Q}$
- Irracionales como π o $\sqrt{3}$ no se pueden representar
- Tampoco los racionales con decimales periódicos (ej: $\frac{1}{5}$)
- Surgen **errores de redondeo** al representar números reales en la computadora

Representación de un número con precisión finita (double):

$$(-1)^s \, 2^{e-1023} \, (1+f)$$

Signo	Exponente	Mantisa
(s) 1 bit	(e) 11 bits	(f) 52 bits

Representación de punto flotante



Aritmética de la computadora

Representación de un número con precisión finita (double):

$$(-1)^s \; 2^{e-1023} \; (1+f)$$

- Los números positivos menores al mínimo representable $2^{-1023} \; (1+2^{-52}) \approx 10^{-308}$, producen **underflow**
- Los números positivos mayores al máximo representable $2^{1024} \; (2-2^{-52}) \approx 10^{308}$, producen **overflow**

Forma normalizada

Para evitar oscurecer el problema usando base 2, usemos base 10.

Forma de punto flotante normalizada de x para decimales de k dígitos

$$x = \pm\, 0, d_1 d_2 \ldots d_k \times 10^n$$

con $1 \leq d_1 \leq 9, \quad 0 \leq d_i \leq 9, \quad i = 1 \ldots k$

O sea, el primer dígito tiene que ser no nulo.

Normalizando un número

Cualquier número en el rango de representación podemos escribirlo como:

$$x = \pm\ 0, d_1 d_2 \ldots d_k, d_{k+1}, d_{k+2} \cdots \times 10^n$$

con $1 \leq d_1 \leq 9, \quad 0 \leq d_i \leq 9, \quad i = 1 \ldots k$

Dos formas de normalizar:

- Truncamiento: $fl(x) = t(x) = \pm\ 0, d_1 d_2 \ldots d_k \times 10^n$
- Redondeo: $fl(x) = t(x + 5 \times 10^{n-(k+1)})$

Notación: $fl(x)$ es la representación de punto flotante por redondeo o truncamiento de x

Missile Patriot: Arabia Saudita, año 1991, 28 muertos

El misil Patriot es un misil de intercepción de ataques enemigos.

Para medir los tiempos necesarios para los cálculos iba sumando de a décimas de segundo a su clock interno.

- Un décimo es periódico en base 2.
  $1/10 \simeq 0.00011001100110011001100110011001100$
- Pero el patriot sólo podía guardar 24 bits. Entonces $1/10$ para el Patriot es $0.00011001100110011001100$
- Lo que se pierde al representar un décimo asi es $\simeq$ 0.000000095
- Al estar prendido por 100 horas, el clock del patriot está corrido $0.000000095 \times 100 \times 36000 = 0.34$ segundos
- Un misil Scud viaja a más de 1600 metros por segundo. Es decir, que en 0.34 segundos hace más o menos 5 cuadras

`http://www-users.math.umn.edu/~arnold/disasters/disasters.html`

Errores más cercanos

- Bueno, ¿pero cuando voy a tener que programar el software de un misil? Esto no me afecta...
- En C++, ¿Cuánto da 3250*0.26?
  Multiplicar 3250 y 0.26 con diferentes tipos de datos para ver cuanto da.
- En C++, ¿Es 3/7 igual a 3/7?
  Ver si 3/7 es siempre lo mismo según como se calcule
- En Python, ¿Cuánto da 0.1+0.2?

Probar en la compu en un ratito...

Las principales tres fuentes de error

... con aritmética finita claro!

Vamos a ver tres posibles operaciones conflictivas entre puntos flotantes

- Sumar números de distinta magnitud
- Restar números muy similares (Cancelación Catastrófica)
- Dividir por números pequeños/Multiplicar por números grandes

Un poco de notación

Sean $x, y \in \mathbb{R}$. Definamos las operaciones entre sus representaciones

$$
\begin{aligned}
x \oplus y &= fl(fl(x) + fl(y)) \\
x \ominus y &= fl(fl(x) - fl(y)) \\
x \odot y &= fl(fl(x) fl(y)) \\
x \oslash y &= fl(fl(x) / fl(y))
\end{aligned}
$$

Suma de números de distinta magnitud

Más conocido como "los grandes siempre ganan"

Al sumar un número real chico con otro grande en valor absoluto, la representación con aritmética finita puede hacer que el valor pequeño se pierda.

Ejemplo usando precisión de $k = 5$ dígitos
Sean $x = 0,88888888 \times 10^7$ e $y = 0,1 \times 10^2$. Entonces

$$\begin{aligned} x \oplus y &= fl(fl(x) + fl(y)) \\ &= fl(0,88888 \cdot 10^7 + 0,1 \cdot 10^2) \\ &= fl(0,888881 \cdot 10^7) \\ &= 0,88888 \cdot 10^7 \end{aligned}$$

El valor x "absorbió" a y. ¿Qué pasaría al sumar muchos números?

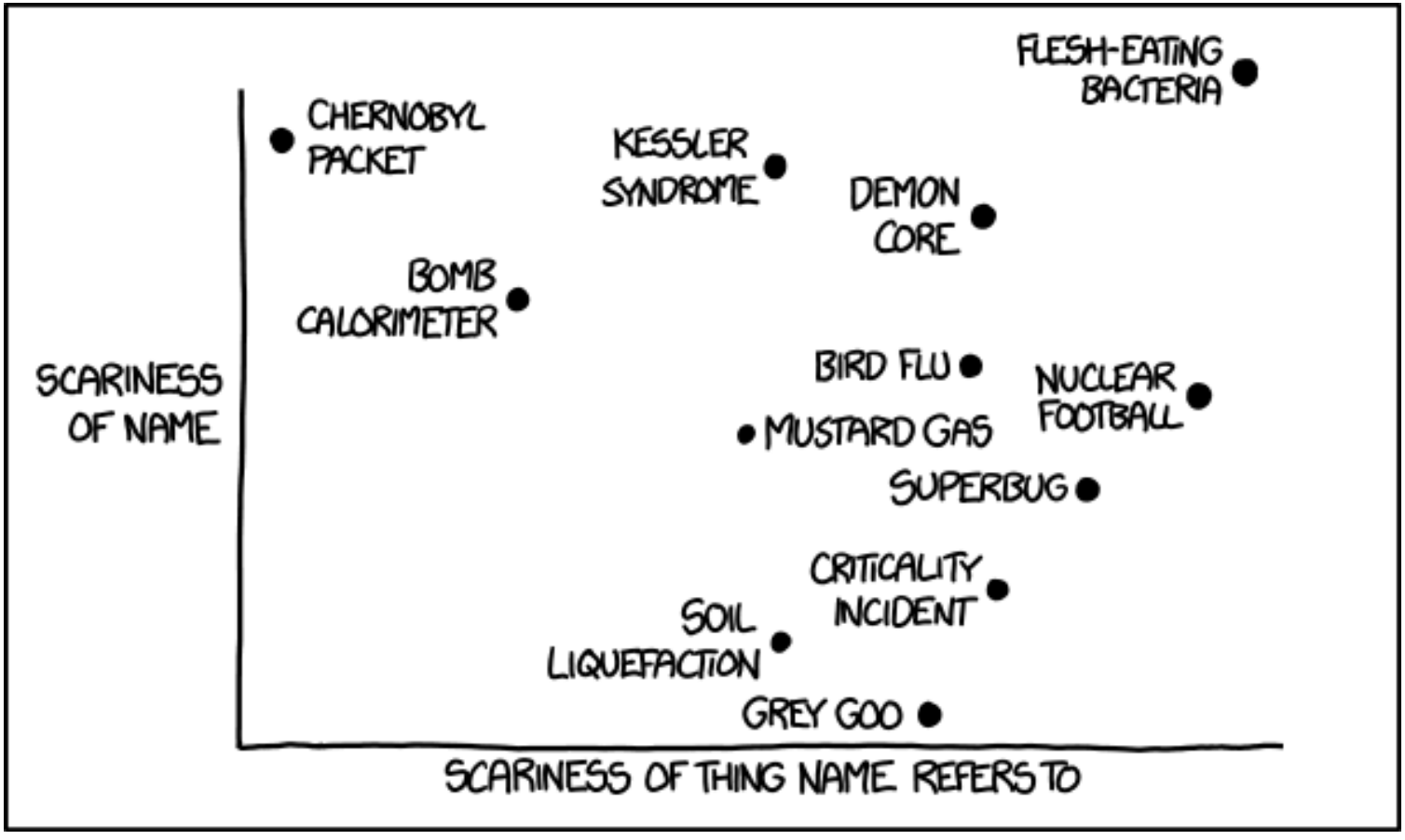
División por números pequeños

- Al multiplicar por un número grande el error de redondeo se amplifica proporcionalmente
- Lo mismo ocurre al dividir por un número muy chico

Ejemplo. Sean:

- $fl(z) = z + \delta$ donde δ es el error de representación o introducido en un cálculo anterior.
- $\epsilon = 10^{-n}$ donde $n > 0$
- $\frac{z}{\epsilon} \approx fl(\frac{fl(z)}{fl(\epsilon)}) = (z + \delta) \times 10^n$
  Error absoluto $= |\delta| \times 10^n$

Cancelación Catastrófica



`https://xkcd.com/1242/` y si no se entiende el chiste...

`https://www.explainxkcd.com/wiki/index.php/1242:_Scary_Names`

Cancelación Catastrófica

- La cancelación catastrófica es la perdida de dígitos significativos que produce la resta de dos números parecidos.
- Dados $x > y$ representados con k cifras significativas

$$fl(x) = 0, d_1 d_2 \dots d_p \alpha_{p+1} \alpha_{p+2} \dots \alpha_k \times 10^n$$

$$fl(y) = 0, d_1 d_2 \dots d_p \beta_{p+1} \beta_{p+2} \dots \beta_k \times 10^n$$

$$fl(fl(x) - fl(y)) = 0, \sigma_1 \sigma_2 \dots \sigma_{k-p} \times 10^{n-p}$$

- Tiene a lo sumo $k - p$ cifras significativas.
- Los cálculos involucrados de aquí en más utilizarán menos cifras.

Cancelación Catastrófica

- Veamos con un ejemplo: Graficar

$$\frac{1 - cos(x)}{x^2}$$

entre -4×10^{-8} y 4×10^{-8}

- Hagamoslo de nuevo pero usando la ecuación

$$\frac{2 \times sin^2(\frac{x}{2})}{x^2}$$

Probar en la compu en un ratito...

Micro-simulacro de TP: Orden de sumatorias

- Hoy sólo nos vamos a centrar en el desarrollo.
- Desarrollo:
  - No es:
    - Código de como implementaron una solución al problema
    - Casos de tests para probar que su implementación es correcta
  - Si es:
    - Dejar bien en claro los objetivos de la investigación.
    - Explicar los diferentes métodos utilizados para la experimentación
    - Plantear las hipótesis presentes antes de la experimentación
    - Proponer y fundamentar la experimentación

Micro-simulacro de TP: Orden de sumatorias

Investigación del día: Sumar muchos números

¿Por qué? ¿Qué puede tener de interesante?

Como:

- Así como vengan. Porque es lo más simple
- Usando el algoritmo de Kahan. Porque la catedra nos lo dijo
- Ordenándolos de menor a mayor. Para no perder los numeritos chiquitos
- Ordenándolos de mayor a menor. Porque quiero romper todo

Micro-simulacro de TP: Orden de sumatorias

- Dejar bien en claro los objetivos de la investigación
- Explicar los diferentes métodos utilizados para la experimentación
- Plantear las hipótesis presentes antes de la experimentación
- Proponer y fundamentar la experimentación

Micro-simulacro de TP: Ejemplo de visualización

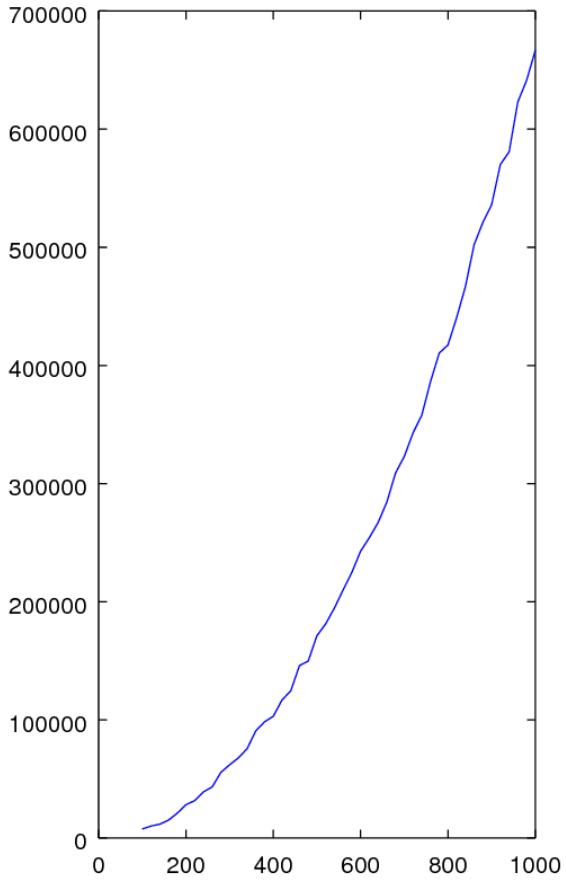
Supongamos que quiero medir los tiempos de mi función de sumatoria para dos algoritmos distintos.

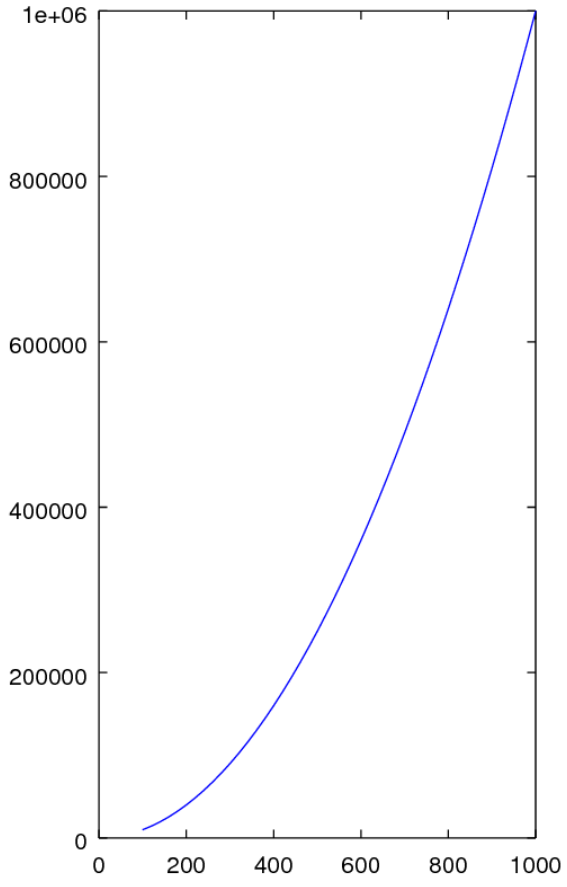
Quiero ver el crecimiento del tiempo a medida que aumenta el tamaño de los vectores.

- Vamos tomando los tiempos para varios tamaños y graficamos.
- ¿Cómo construimos casos? ¿Cuáles son casos adecuados? ¿Cualquier caso nos sirve?
- ¿Qué tamaños usamos? ¿Tomamos varias mediciones por tamaño? ¿Por qué?
- ¿Qué hacemos con nuestras mediciones? ¿Puede haber outliers? ¿Cómo los tratamos?
- ¿Qué sabemos de nuestra función? ¿Qué deberíamos esperar de los resultados?
- Y por último, ¿tienen sentido éstos resultados que obtuve?

Micro-simulacro de TP: Resultados

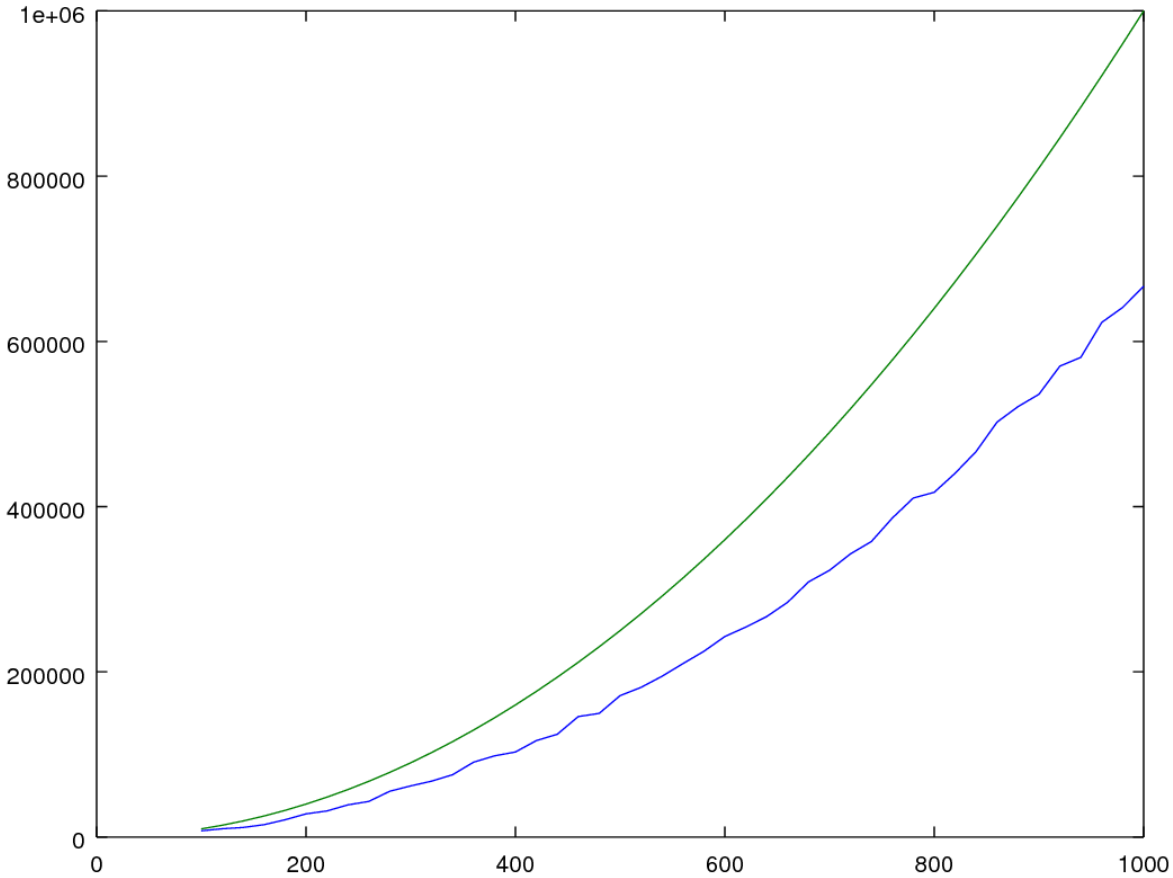
¿Qué sabemos de la función? ¿Qué opciones hay para graficar/mostrar esto?





Micro-simulacro de TP: Resultados

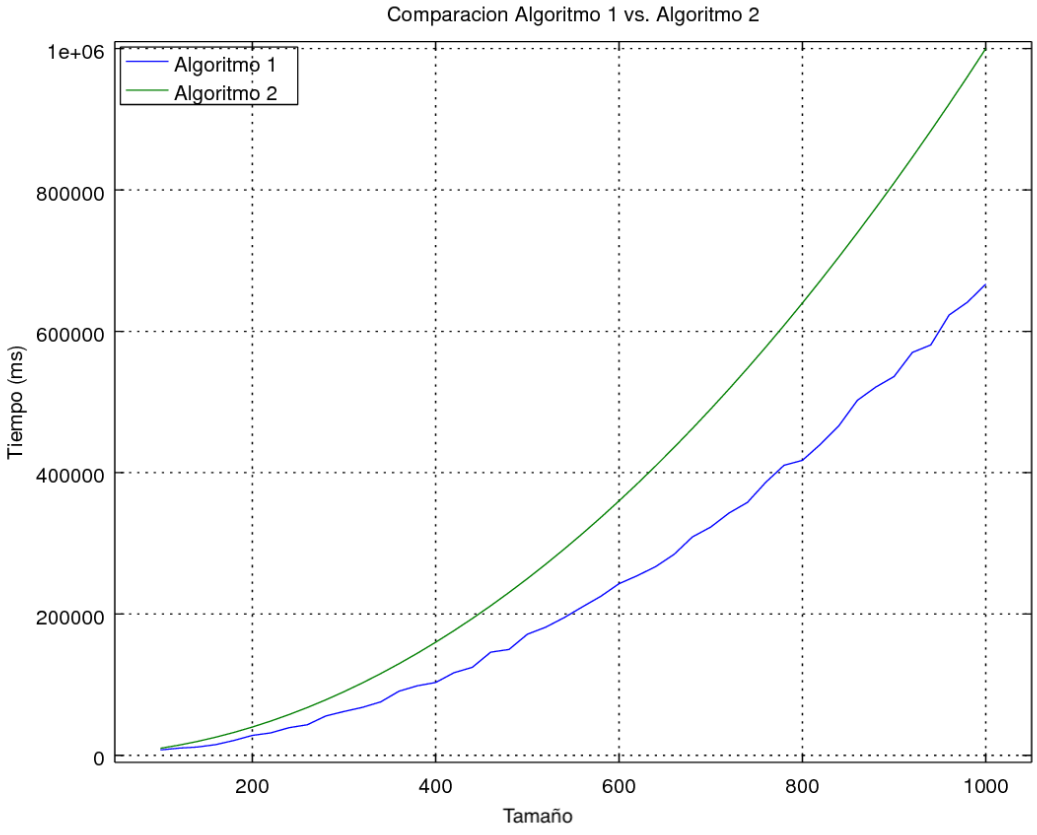
Finalmente armamos nuestro gráfico...



- ¿Qué tiene de mal (o bien) nuestro gráfico? ¿Faltan cosas?

Micro-simulacro de TP: Resultados

Lo mejoramos un poco...



- Supongamos que nos interesa "corroborar" los tiempos de nuestra experimentación...

Micro-simulacro de TP: Orden de sumatorias

¿Y las otras secciones del informe?

- Introducción
- Desarrollo
- Discusión
- Resultados
- Conclusiones

Recomendaciones rápidas

- Bibliografía usada al final VS links como notas la pie
- **Referenciar todas las figuras**
- Todo gráfico o tabla tiene un propósito

LEER

- Pautas de laboratorio en el Campus
- ¡Ver la TPChecklist.pdf antes de entregar!