

# MCOC2021-P0

---

## Mi computador principal

---

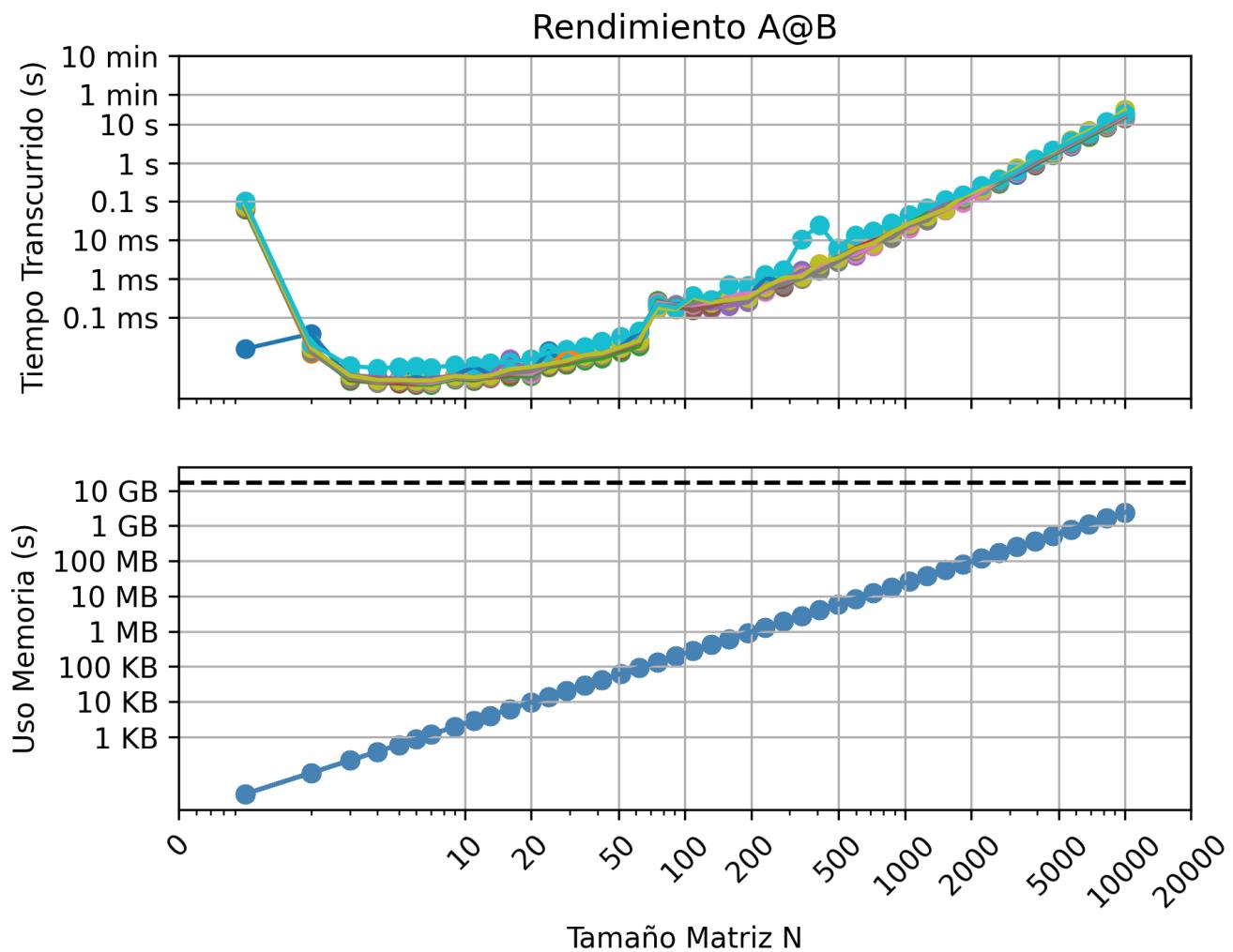
- Marca/modelo: Asus VivoBook S15 X530UF
- Tipo: Notebook
- Año adquisición: 2019
- Procesador:
  - Marca/Modelo: Intel Core i7-8550U
  - Velocidad Base: 1.80 GHz
  - Velocidad Máxima: 4.00 GHz
  - Numero de núcleos: 4
  - Numero de hilos (Procesadores lógicos): 8
  - Arquitectura: x86\_64
  - Set de instrucciones: Intel® SSE4.1, Intel® SSE4.2, Intel® AVX2
- Tamaño de las cachés del procesador
  - L1: 256KB
  - L1: 1.0MB
  - L2: 8.0MB
- Memoria
  - Total: 16 GB
  - Tipo memoria: DDR4
  - Velocidad 2400 MHz
  - Numero de (SO)DIMM: 2
- Tarjeta Gráfica
  - Marca / Modelo: Nvidia GeForce MX130
  - Memoria dedicada: 2048 MB GDDR5

- Resolución: 1920 x 1080
- Disco 1:
  - Marca: SanDisk
  - Tipo: SSD
  - Tamaño: 238GB (en teoría tiene 256GB)
  - Particiones: 3 (OS, SYSTEM, RECOVERY)
  - Sistema de archivos: NTFS (además tiene FAT32 en la partición SYSTEM)
- Disco 2:
  - Marca: Toshiba
  - Tipo: HDD
  - Tamaño: 931GB (en teoría tiene 1TB)
  - Particiones: 1
  - Sistema de archivos: NTFS
- Dirección MAC de la tarjeta wifi: 20:16:B9:44:BD:A3
- Dirección IP (Interna, del router): 192.168.100.2
- Dirección IP (Externa, del ISP): 186.67.234.139
- Proveedor internet: Entel Chile S.A. (fibra óptica)

## Desempeño MATMUL

---

En primer lugar cabe señalar que rendimiento.txt corresponde a un archivo que por cada línea es una lista de listas, en donde cada una de estas es considerada una nueva corrida y está ordenada de la siguiente manera : [[Ns],[dts],[mems]]



## Preguntas

- **¿Cómo difiere del gráfico del profesor/ayudante?**

En primer lugar se puede notar como tengo una partida similar en tiempo a las del profesor/ayudante, luego las 9 siguientes tienen un tiempo de partida mayor. Luego es interesante señalar como en matrices con tamaños entre 50 y 60 de evidencia un salto en el tiempo de memoria en el cual en mi notebook es menor que el del profesor/ayudante. Por último el comportamiento final es prácticamente igual, terminando por cada partida un poco inferior a un minuto al igual que el profesor/ayudante. Otra diferencia significativa podría ser debido a la cantidad de N en el eje x que el profesor/ayudante seleccionó vs los que yo establecí (45).

En estricto rigor se puede evidenciar como el pc del profesor/ayudante tiene un mejor rendimiento en un inicio y en el final, mientras que mi PC tiene mejor rendimiento en los valores de N intermedios.

Lo anteriormente señalado ocurre debido a las distintas características de caché, RAM, y disco entre el pc del profesor/ayudante y el mio. Pero en general, los gráficos son muy similares.

- **¿A qué se pueden deber las diferencias en cada corrida?**

Esto ocurre, ya que, python para el uso de memoria siempre intenta utilizar la menor cantidad de memoria posible. Además, la memoria funciona por bloques, entonces al necesitar más memoria irá en búsqueda del siguiente módulo de memoria y es por esto que se producen los "saltos" en el tiempo que se observan en el gráfico, en estos cambios de memoria manda la jerarquía de memoria es decir, primero caché, luego RAM y por último el disco. Este proceso de "cambio de módulo de memoria" es muy variable y es por eso que se puede evidenciar diferencias en los tiempos de ejecución en cada corrida.

Otro factor que puede influir es el sistema operativo, el cual prioriza distintos procesos (hace una cosa a la vez con distintas prioridades, por lo que hacer diferentes acciones el el pc (como navegar en internet) puede influir directamente), los cuales pueden influir directamente en el tiempo de ejecución, estas pueden ser acciones que uno hace directamente, como también acciones que hace el sistema operativo realiza por detrás sin que uno se de cuenta.

Por último, otros factores que pueden influir son que el computador este utilizando una gran cantidad de recursos que relentece el proceso (por ejemplo que la temperatura del PC aumente).

- El gráfico de uso de memoria es lineal con el tamaño de matriz, pero el de tiempo transcurrido no lo es ¿porqué puede ser?

Esto ocurre ya que, en una multiplicación de matrices existe un número predeterminado de "acciones" por lo que la memoria utilizada en la operación siempre será la misma (es por eso que al graficar las 10 corridas sigue viendose una única gráfica). Es decir, la memoria utilizada esta establecida por la operación y el porte de las matrices y no influira si esta se desarrolla antes o después a diferencia de lo que ocurre con el tiempo.

Con respecto a la linealidad del gráfico, la memoria utilizada es lineal ya que si una matriz es el doble de grande simplemente utilizará el doble de memoria sin importar otros factores (lo mismo ocurre con la operación MATMUL), pero en el caso del tiempo, a medida que se multiplica una matriz con valores de N más grande, los recursos necesarios van creciendo exponencialmente (es decir una matriz el doble de grande no demorara necesariamente el doble del tiempo sino que más), esto debido a la complejidad de la operación MATMUL que va aumentando exponencialmente a medida que aumenta los valores de N.

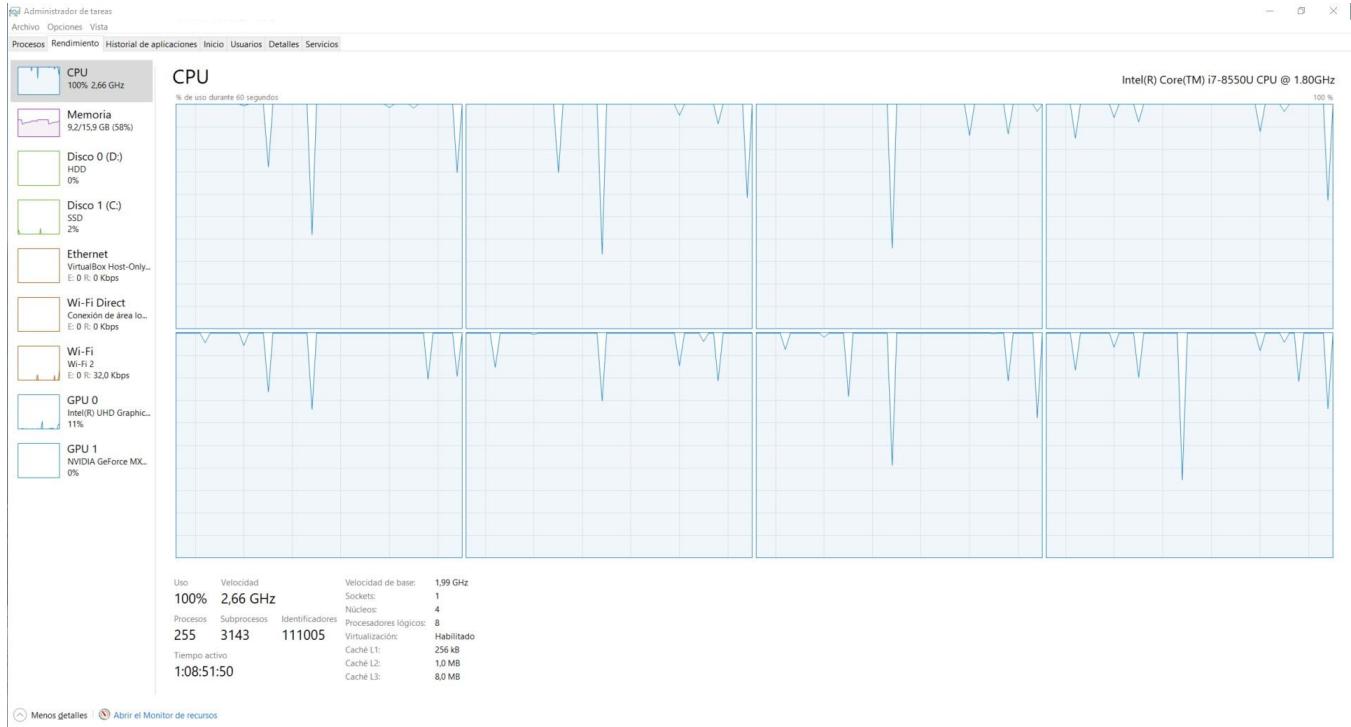
- ¿Qué versión de python está usando?

Python Version: 3.9.6

- ¿Qué versión de numpy está usando?

NumPy Version: 1.21.1

- Durante la ejecución de su código ¿se utiliza más de un procesador? Muestre una imagen (screenshot) de su uso de procesador durante alguna corrida para confirmar.



Tal como se evidencia en la imagen, durante la ejecución del código se utilizan 8 procesadores, los cuales corresponden a todos los de mi CPU.

El hecho de que utilice los 8 procesadores (y con tanto %uso), es ya que el código necesita demasiados recursos y de esta manera utiliza demasiada CPU a tal punto de necesitar el 100% de su capacidad.

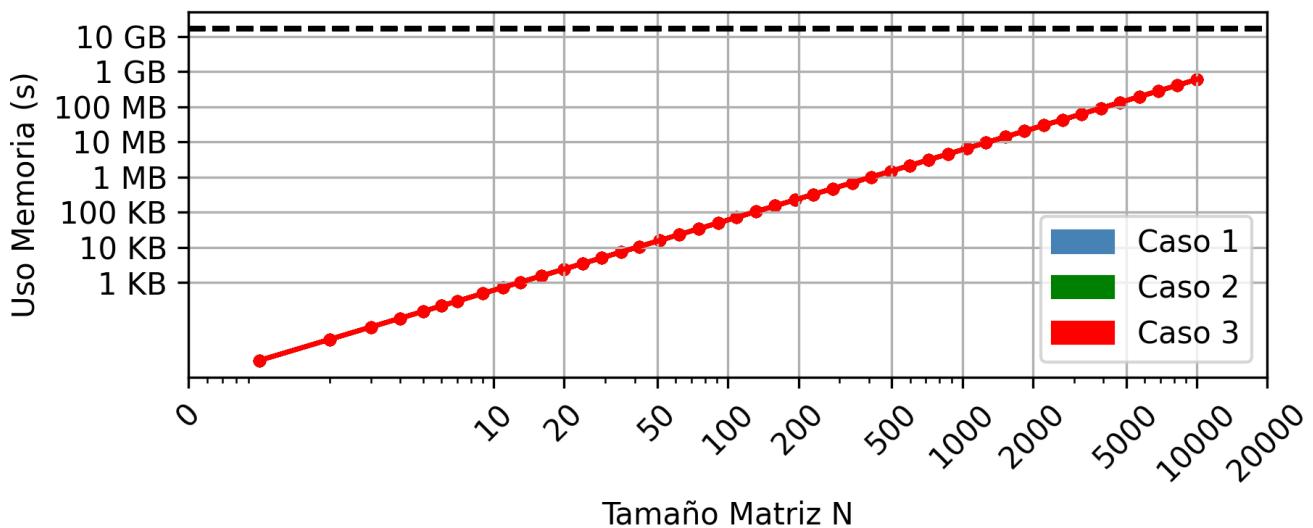
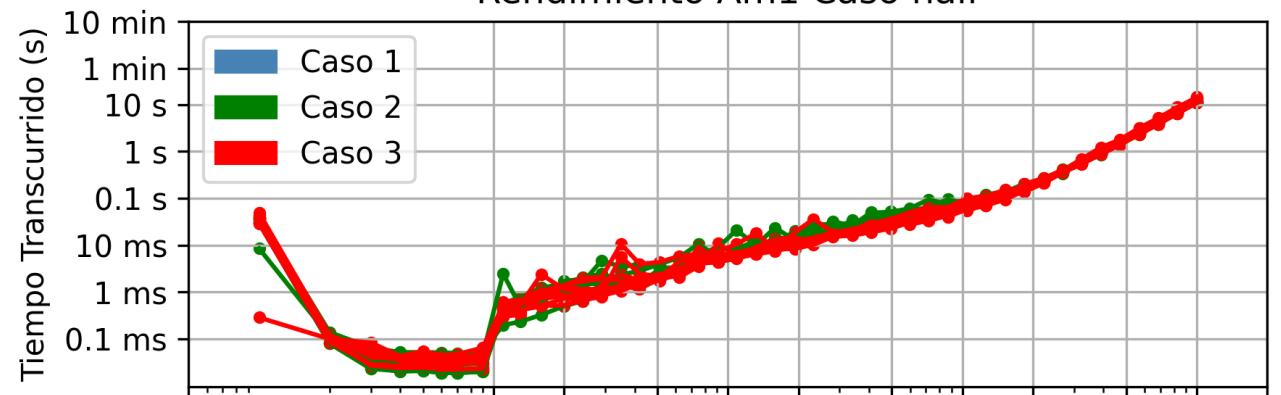
## Desempeño de INV

Se realizan 12 archivos .txt, uno por cada tipo en cada uno de los tres casos. Es importante señalar que tanto el tipo half y longdouble en el caso 1 (usando librería numpy) no se logran realizar, ya que son tipos que linalg de numpy no soportan.

A continuación se muestran las comparaciones de los casos por cada tipo de dato:

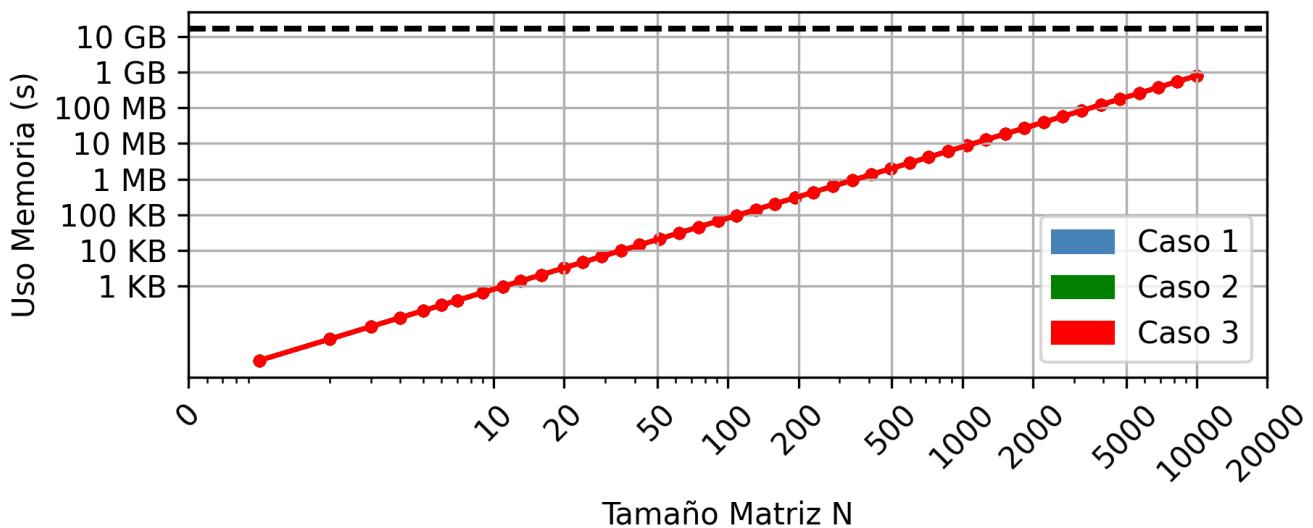
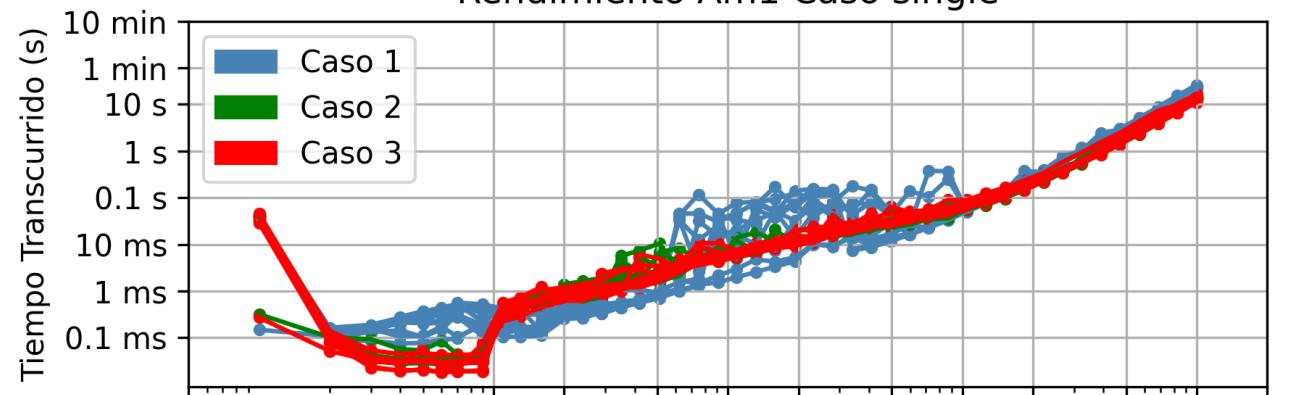
DTYPE: HALF

Rendimiento Am1 Caso half



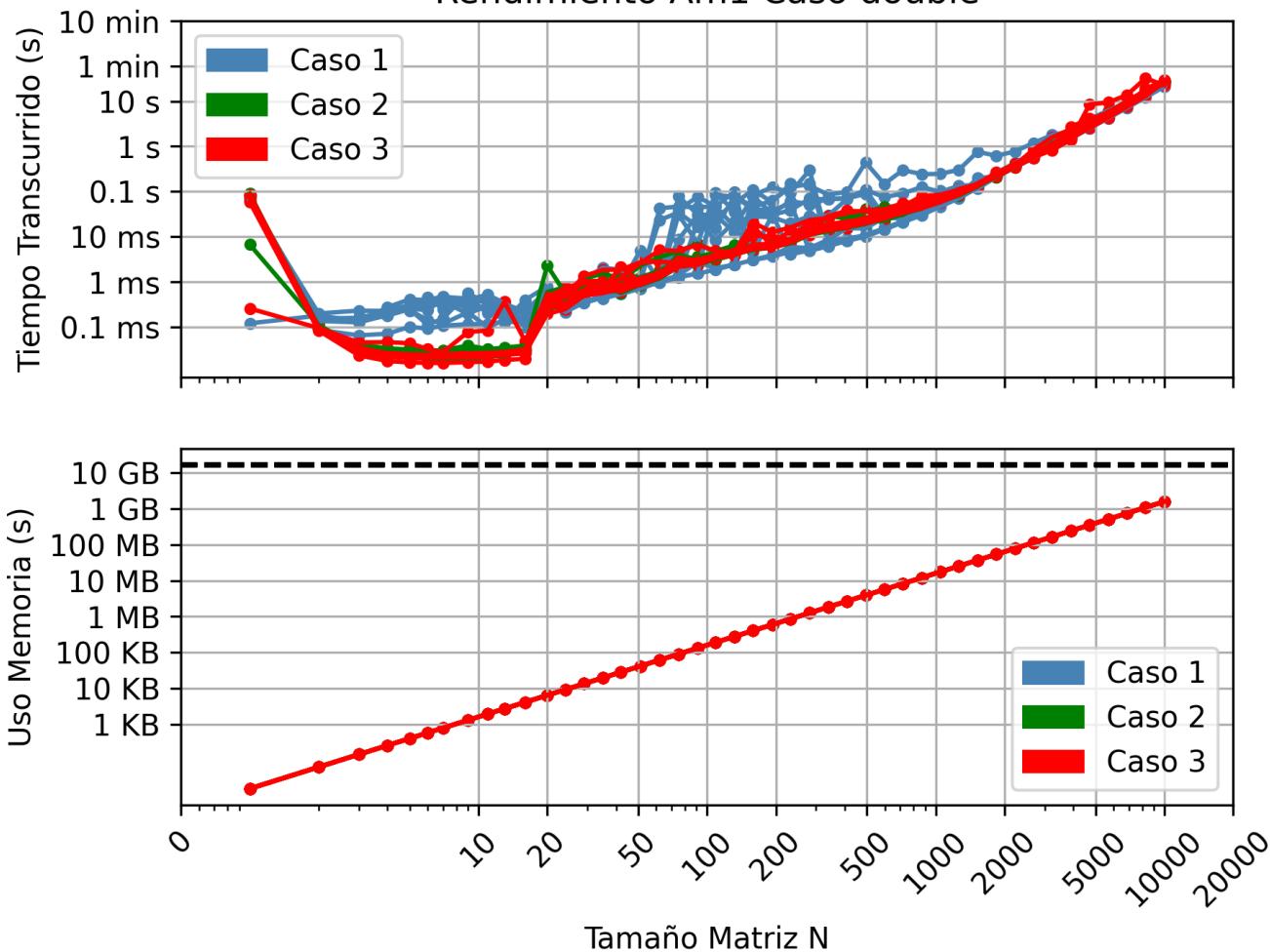
DTYPE: SINGLE

Rendimiento Am1 Caso single

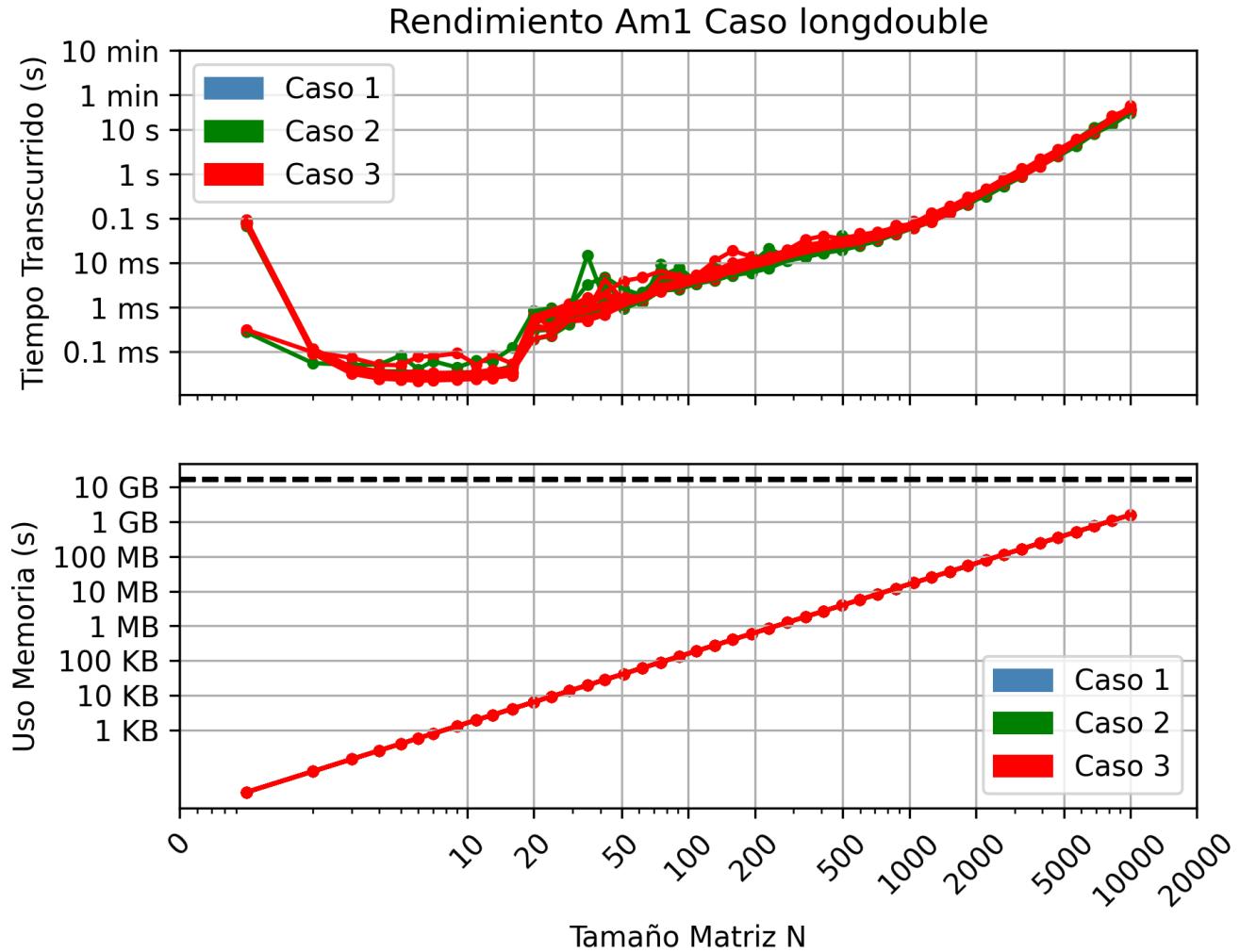


DTYPE: DOUBLE

Rendimiento Am1 Caso double

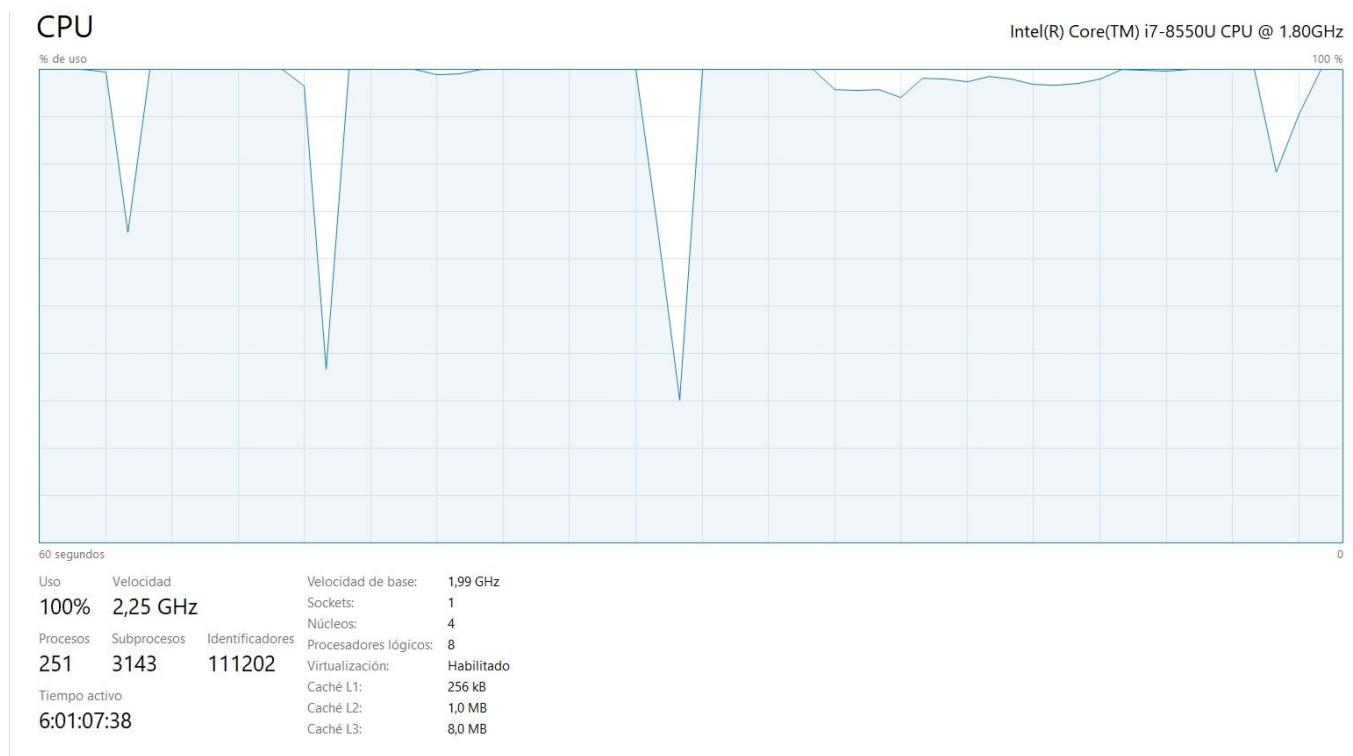


DTYPE: LONGDOUBLE



Además tanto el desempeño del procesador como de la memoria en todos los casos que los códigos funcionaron los resultado fueron prácticamente idénticos, es decir el procesador usando casi todos sus recursos (todos incluso), mientras que la memoria prácticamente no cambiaba y si lo hacía era muy leve. Si hubo alguna diferencia fue prácticamente imperceptible.

## Procesador durante corridas:



## Memoria durante corridas:



## Preguntas

- ¿Qué algoritmo de inversión cree que utiliza cada método (ver wiki)? Justifique claramente su respuesta.

**NumPy:** numpy.linalg.inv(A) lo que hace es llamar a la función numpy.linalg.solve(A,I) (esta función realiza  $Ax = I$ , en donde x correspondería en este caso a la matriz inversa de A), en donde I corresponde a la matriz identidad, y luego mediante lapack's LU factorization (paquete de Descomposición LU) resuelve el sistema de solve. Esto significa que finalmente ocupa eliminación Gaussiana en donde la ortogonalidad no se detecta por default. Cabe destacar que este método aumenta el error si el array dado no es cuadrado o la inversión falla.

**SciPy:** Muy parecido a NumPy, lo que hace Scipy (scipy.linalg.inv(A)) es llamar directamente a los paquetes LAPACK (get\_lapack\_funcs("función que se quiere")) y desde ahí mediante descomposición LU realiza la inversión de la matriz. Es importante señalar que la opción overwrite\_a lo que realiza es reemplazar los valores de la matriz (los va descartando), es por eso que esta opción en True podría incrementar el rendimiento. Importante decir que se puede notar que scipy realiza en general el proceso más rápido ya que llama directamente a los paquetes para realizar la descomposición Lu, en cambio, numpy.linalg.inv() llama a numpy.linalg.solve(), y es esta función la que llama al mismo paquete, es decir "utiliza un paso más"

- ¿Cómo incide el paralelismo y la estructura de caché de su procesador en el desempeño en cada caso? Justifique su comentario en base al uso de procesadores y memoria observado durante las corridas.

La lógica del paralelismo es realizar varias tareas al mismo tiempo, para esto el computador divide el procesador en "mini procesadores" que cada uno realiza diferentes acciones. Sabiendo esto, al correr el programa con datos como half vs longdouble el procesador con half ("menos información"), trabaja de mejor manera con estos "mini procesadores", ya que esa tarea puede distribuirla mejor entre ellos sin tener que colapsar los caché, en cambio con longdouble el procesador necesita usar mayores recursos destinando más cantidad de "mini procesadores" y así dificulta y alarga más el tiempo de ejecución, ya que estos estarán entregando mayor rendimiento con un tamaño de caché al máximo.

Es importante señalar que en mi caso, para todos los tipos de datos el tiempo de corrida fue similar, levemente los datos más pequeños fueron un poco más rápido, pero es de manera prácticamente imperceptible. Esto quiere decir que mi pc, para poder correr datos de half utiliza varios de estos "mini procesadores" y con la capacidad de los caché si no es al máximo, muy cercano, en el caso de un dato más grande como longdouble, probablemente este usando todo los recursos.

Por último, para el caso de la memoria, al tener una gran cantidad y con mucho desuso no significó un gran problema para todos los tipos de datos, sin embargo con datos pequeños esta trabajo leve e imperceptiblemente menos que con datos más grandes.

## Desempeño de SOLVE y EIGH

---

Se realizaron 4 archivos .py (uno para todos los puntos del caso A, otro para todos los puntos del caso B y esto por cada tipo de dato), además de estos se obtuvieron 4 archivos .txt los cuales por línea contienen los subcasos de A y B, pero solo los valores promedios de las 10 corridas por subcaso (son del tipo [[Ns],[dts]]).

A continuación se muestra una tabla con los desempeños del procesador y la memoria por cada caso (subcaso) realizado.

CASO	PROCESADOR (CPU)	MEMORIA (RAM)
------	------------------	---------------

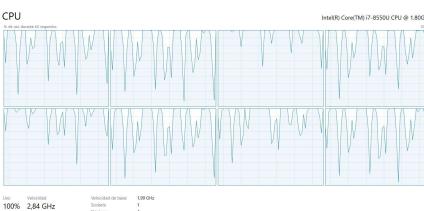
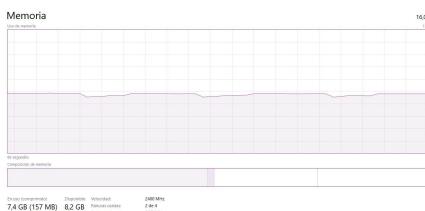
CASO	PROCESADOR (CPU)	MEMORIA (RAM)
A.I (float) ( $x = \text{Am1} \times b$ )	<p><b>CPU</b></p> <p>100% 2.71 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Velocidad de base: 1.8 GHz Sistema: 4 Procesadores: 1 Memoria: 8 GB Vidriadoras: 128GB Habilitado: 128GB Cache L1: 3.5 MB Cache L2: 12.5 MB Cache L3: 8.0 MB Tiempo actual: 8:05:41:23</p>	<p><b>Memoria</b></p> <p>16.0 GB 11:12:28 Al asignar: 8.8 GB (585 MB) Disponible: 7.0 GB Velocidad: 2400 MHz Z en 4: 12.5/18.3 GB 5.3 GB Reservado para hardware: 117 MB Borrar caché: 718 MB Borrar todo: 1014 MB</p>
A.II (float) ( <code>scipy.linalg.solve</code> by default)	<p><b>CPU</b></p> <p>77% 2.25 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Velocidad de base: 1.8 GHz Sistema: 4 Procesadores: 1 Memoria: 8 GB Vidriadoras: 128GB Habilitado: 128GB Cache L1: 3.5 MB Cache L2: 12.5 MB Cache L3: 8.0 MB Tiempo actual: 8:05:55:10</p>	<p><b>Memoria</b></p> <p>16.0 GB 11:12:28 Al asignar: 7.5 GB (551 MB) Disponible: 8.2 GB Velocidad: 2400 MHz Z en 4: 12.5/18.3 GB 5.3 GB Reservado para hardware: 117 MB Borrar caché: 718 MB Borrar todo: 1012 MB</p>
A.III (float) (using <code>assume_a='pos'</code> )	<p><b>CPU</b></p> <p>100% 2.25 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Velocidad de base: 1.8 GHz Sistema: 4 Procesadores: 1 Memoria: 8 GB Vidriadoras: 128GB Habilitado: 128GB Cache L1: 3.5 MB Cache L2: 12.5 MB Cache L3: 8.0 MB Tiempo actual: 8:05:59:34</p>	<p><b>Memoria</b></p> <p>16.0 GB 11:12:28 Al asignar: 7.7 GB (587 MB) Disponible: 8.0 GB Velocidad: 2400 MHz Z en 4: 12.5/18.3 GB 5.3 GB Reservado para hardware: 117 MB Borrar caché: 718 MB Borrar todo: 1016 MB</p>
A.IV (float) (using <code>assume_a='sym'</code> )	<p><b>CPU</b></p> <p>100% 2.67 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Velocidad de base: 1.8 GHz Sistema: 4 Procesadores: 1 Memoria: 8 GB Vidriadoras: 128GB Habilitado: 128GB Cache L1: 3.5 MB Cache L2: 12.5 MB Cache L3: 8.0 MB Tiempo actual: 8:06:04:52</p>	<p><b>Memoria</b></p> <p>16.0 GB 11:12:28 Al asignar: 7.0 GB (429 MB) Disponible: 8.7 GB Velocidad: 2400 MHz Z en 4: 12.5/18.3 GB 5.1 GB Reservado para hardware: 117 MB Borrar caché: 718 MB Borrar todo: 1016 MB</p>
A.V (float) (using <code>overwrite_a=True</code> )	<p><b>CPU</b></p> <p>74% 2.52 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Velocidad de base: 1.8 GHz Sistema: 4 Procesadores: 1 Memoria: 8 GB Vidriadoras: 128GB Habilitado: 128GB Cache L1: 3.5 MB Cache L2: 12.5 MB Cache L3: 8.0 MB Tiempo actual: 8:06:09:37</p>	<p><b>Memoria</b></p> <p>16.0 GB 11:12:28 Al asignar: 7.3 GB (488 MB) Disponible: 8.5 GB Velocidad: 2400 MHz Z en 4: 12.5/18.3 GB 5.3 GB Reservado para hardware: 117 MB Borrar caché: 719 MB Borrar todo: 1016 MB</p>
A.VI (float) (using <code>overwrite_b=True</code> )	<p><b>CPU</b></p> <p>49% 2.68 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Velocidad de base: 1.8 GHz Sistema: 4 Procesadores: 1 Memoria: 8 GB Vidriadoras: 128GB Habilitado: 128GB Cache L1: 3.5 MB Cache L2: 12.5 MB Cache L3: 8.0 MB Tiempo actual: 8:06:14:40</p>	<p><b>Memoria</b></p> <p>16.0 GB 11:12:28 Al asignar: 7.5 GB (521 MB) Disponible: 8.2 GB Velocidad: 2400 MHz Z en 4: 12.5/18.3 GB 5.4 GB Reservado para hardware: 117 MB Borrar caché: 719 MB Borrar todo: 1017 MB</p>

CASO	PROCESADOR (CPU)	MEMORIA (RAM)
A.VII (float) (using overwrite_a=True and overwrite_b=True)	<p>CPU</p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 82% 2.22 GHz</p> <p>Procesador: 266 3264 127908</p> <p>Memoria: 8.06:1937</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 1.00 GHz</p> <p>Procesador: 266 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p> <p>Velocidad: 100% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>	<p>Memoria</p> <p>16.0 GB 11:38</p> <p>8.0 GB (592 MB) 7.3 GB</p> <p>240 MHz 2.044 500MHz</p> <p>11.0/18.3 GB 5.3 GB</p> <p>720 MB 1014 MB</p> <p>Velocidad: 100% 1.00 GHz</p> <p>Procesador: 266 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>
A.I (double) ( x = Am1 x b )	<p>CPU</p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>	<p>Memoria</p> <p>16.0 GB 11:38</p> <p>9.1 GB (548 MB) 6.7 GB</p> <p>240 MHz 2.044 500MHz</p> <p>13.0/18.3 GB 5.5 GB</p> <p>724 MB 1020 MB</p> <p>Velocidad: 100% 1.00 GHz</p> <p>Procesador: 266 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>
A.II (double) ( scipy.linalg.solve by default)	<p>CPU</p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>	<p>Memoria</p> <p>16.0 GB 11:38</p> <p>8.0 GB (607 MB) 6.7 GB</p> <p>240 MHz 2.044 500MHz</p> <p>13.0/18.3 GB 5.6 GB</p> <p>725 MB 1020 MB</p> <p>Velocidad: 100% 1.00 GHz</p> <p>Procesador: 266 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>
A.III (double) (using assume_a='pos' )	<p>CPU</p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 100% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:2227</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:3000</p>	<p>Memoria</p> <p>16.0 GB 11:38</p> <p>7.2 GB (492 MB) 8.5 GB</p> <p>240 MHz 2.044 500MHz</p> <p>11.0/18.3 GB 5.4 GB</p> <p>724 MB 1017 MB</p> <p>Velocidad: 100% 1.00 GHz</p> <p>Procesador: 266 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>
A.IV (double) (using assume_a='sym' )	<p>CPU</p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 100% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:2227</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:3000</p>	<p>Memoria</p> <p>16.0 GB 11:38</p> <p>8.4 GB (491 MB) 7.3 GB</p> <p>240 MHz 2.044 500MHz</p> <p>12.0/18.3 GB 5.4 GB</p> <p>724 MB 1020 MB</p> <p>Velocidad: 100% 1.00 GHz</p> <p>Procesador: 266 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>
A.V (double) (using overwrite_a=True )	<p>CPU</p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 100% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p> <p>Ventilador: 100% 1.00 GHz</p> <p>Temperatura: 40°C</p> <p>Velocidad: 100% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>	<p>Memoria</p> <p>16.0 GB 11:38</p> <p>7.3 GB (510 MB) 8.5 GB</p> <p>240 MHz 2.044 500MHz</p> <p>11.0/18.3 GB 5.4 GB</p> <p>724 MB 1018 MB</p> <p>Velocidad: 100% 1.00 GHz</p> <p>Procesador: 266 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.43 GHz</p> <p>Procesador: 264 3144 127270</p> <p>Memoria: 8.07:0053</p> <p>Velocidad: 100% 2.71 GHz</p> <p>Procesador: 266 3236 128277</p> <p>Memoria: 8.07:1403</p> <p>Velocidad: 100% 2.87 GHz</p> <p>Procesador: 265 3249 128055</p> <p>Memoria: 8.07:1828</p> <p>Velocidad: 80% 2.77 GHz</p> <p>Procesador: 264 3239 128393</p> <p>Memoria: 8.07:2227</p> <p>Velocidad: 65% 3.17 GHz</p> <p>Procesador: 266 3340 127601</p> <p>Memoria: 8.07:3000</p>

CASO	PROCESADOR (CPU)	MEMORIA (RAM)
A.VI (double) (using overwrite_b=True )	<p><b>CPU</b></p> <p>31% 3,34 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Processador: 3261 128547 Memoria: 8073226</p> <p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>	<p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>
A.VII (double) (using overwrite_a=True and overwrite_b=True	<p><b>CPU</b></p> <p>94% 2,53 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Processador: 3360 128517 Memoria: 8073659</p> <p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>	<p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>
B.I (float) ( scipy.linalg.eigh by default)	<p><b>CPU</b></p> <p>87% 2,67 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Processador: 3407 136153 Memoria: 8132635</p> <p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>	<p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>
B.II.1 (float) ( driver='ev' and overwrite_a=False )	<p><b>CPU</b></p> <p>34% 3,71 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Processador: 281 3694 141251 Memoria: 8133749</p> <p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>	<p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>
B.II.2 (float) ( driver='ev' and overwrite_a=True )	<p><b>CPU</b></p> <p>42% 3,56 GHz Procesador: Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz Processador: 269 3365 135005 Memoria: 8135808</p> <p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>	<p><b>Memoria</b></p> <p>16,0 GB 112 MB</p>

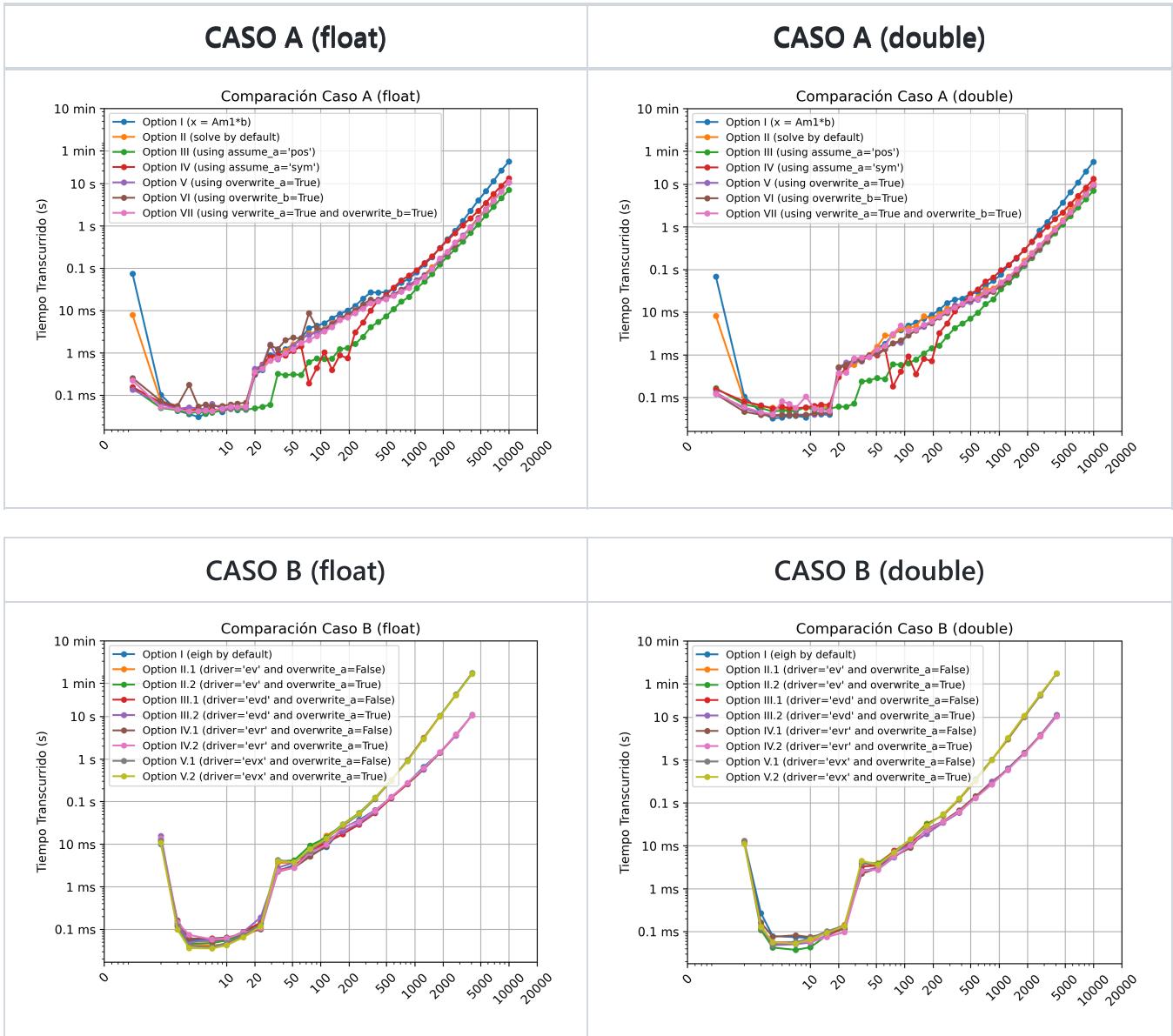
CASO	PROCESADOR (CPU)	MEMORIA (RAM)
B.III.1 (float) ( driver='evd' and overwrite_a=False )	<p><b>CPU</b></p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Uso: 100% Velocidad: 2.73 GHz Velocidad de base: 1.00 GHz</p> <p>Sobrecarga: 4 Sockets: 1 Procesadores: 8 Procesadores físicos: 8</p> <p>Memoria: 264 MB Cache L1: 32 KB Cache L2: 256 KB Cache L3: 8 MB</p> <p>Firma activa: 8:14:23:05</p> <p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 6.8 GB (535 MB) Disponible: 8.1 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.2/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>918 MB 1.1 GB</p>	<p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 6.8 GB (535 MB) Disponible: 8.1 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.2/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>918 MB 1.1 GB</p>
B.III.2 (float) ( driver='evd' and overwrite_a=True )	<p><b>CPU</b></p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Uso: 92% 2.68 GHz Velocidad: 1.00 GHz</p> <p>Sobrecarga: 4 Sockets: 1 Procesadores: 8 Procesadores físicos: 8</p> <p>Memoria: 265 MB Cache L1: 32 KB Cache L2: 256 KB Cache L3: 8 MB</p> <p>Firma activa: 8:14:25:02</p> <p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.0 GB (535 MB) Disponible: 8.9 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.5/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>919 MB 1.1 GB</p>	<p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.0 GB (535 MB) Disponible: 8.9 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.5/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>919 MB 1.1 GB</p>
B.VI.1 (float) ( driver='evr' and overwrite_a=False )	<p><b>CPU</b></p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Uso: 99% 2.69 GHz Velocidad: 1.00 GHz</p> <p>Sobrecarga: 4 Sockets: 1 Procesadores: 8 Procesadores físicos: 8</p> <p>Memoria: 265 MB Cache L1: 32 KB Cache L2: 256 KB Cache L3: 8 MB</p> <p>Firma activa: 8:14:27:22</p> <p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 6.9 GB (535 MB) Disponible: 8.9 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.2/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>920 MB 1.1 GB</p>	<p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 6.9 GB (535 MB) Disponible: 8.9 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.2/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>920 MB 1.1 GB</p>
B.VI.2 (float) ( driver='evr' and overwrite_a=True )	<p><b>CPU</b></p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Uso: 100% 2.65 GHz Velocidad: 1.00 GHz</p> <p>Sobrecarga: 4 Sockets: 1 Procesadores: 8 Procesadores físicos: 8</p> <p>Memoria: 264 MB Cache L1: 32 KB Cache L2: 256 KB Cache L3: 8 MB</p> <p>Firma activa: 8:14:30:02</p> <p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.1 GB (522 MB) Disponible: 8.7 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.4/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>920 MB 1.1 GB</p>	<p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.1 GB (522 MB) Disponible: 8.7 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.4/16.3 GB 6.8 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>920 MB 1.1 GB</p>
B.V.1 (float) ( driver='evx' and overwrite_a=False )	<p><b>CPU</b></p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Uso: 39% 3.57 GHz Velocidad: 1.00 GHz</p> <p>Sobrecarga: 4 Sockets: 1 Procesadores: 8 Procesadores físicos: 8</p> <p>Memoria: 270 MB Cache L1: 32 KB Cache L2: 256 KB Cache L3: 8 MB</p> <p>Firma activa: 8:14:36:00</p> <p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.2 GB (522 MB) Disponible: 8.6 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.5/16.3 GB 6.7 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>920 MB 1.1 GB</p>	<p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.2 GB (522 MB) Disponible: 8.6 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.5/16.3 GB 6.7 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>920 MB 1.1 GB</p>
B.V.2 (float) ( driver='evx' and overwrite_a=True )	<p><b>CPU</b></p> <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Uso: 42% 3.57 GHz Velocidad: 1.00 GHz</p> <p>Sobrecarga: 4 Sockets: 1 Procesadores: 8 Procesadores físicos: 8</p> <p>Memoria: 261 MB Cache L1: 32 KB Cache L2: 256 KB Cache L3: 8 MB</p> <p>Firma activa: 8:14:59:51</p> <p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.0 GB (505 MB) Disponible: 8.8 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.0/16.3 GB 6.7 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>921 MB 1.1 GB</p>	<p><b>Memoria</b></p> <p>16.0 GB</p> <p>Al asignar: 0 Al liberar: 0 Al reservar: 0 Al liberar: 0</p> <p>Datos compresión: 7.0 GB (505 MB) Disponible: 8.8 GB Memoria usada: 2.4 GB (200MB) Reservada para hibernación: 111 MB</p> <p>11.0/16.3 GB 6.7 GB</p> <p>Borrar todo: Borrar los cambios</p> <p>921 MB 1.1 GB</p>



CASO	PROCESADOR (CPU)	MEMORIA (RAM)
B.IV.1 (double) ( driver='evr' and overwrite_a=False )	 <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 100% 2.64 GHz</p> <p>Procesos: 257 2913 122567</p> <p>Memoria: 4 16 GB</p> <p>Cache L1: 256 MB 1.048 KB</p> <p>Cache L2: 1.024 MB 8.096 KB</p> <p>8:22:18:21</p>	 <p>Memoria: 16.0 GB (12.38)</p> <p>En uso: 16.0 GB</p> <p>Disponible: 0.0 GB</p> <p>Velocidad: 2400 MHz</p> <p>2400 MHz 2.048 GB SODIMM DDR4</p> <p>9.5/18.3 GB 4.0 GB</p> <p>9.5/18.3 GB 4.0 GB</p> <p>690 MB 1.0 GB</p>
B.IV.2 (double) ( driver='evr' and overwrite_a=True )	 <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 100% 2.64 GHz</p> <p>Procesos: 253 2651 120097</p> <p>Memoria: 8 16 GB</p> <p>Cache L1: 256 MB 1.048 KB</p> <p>Cache L2: 1.024 MB 8.096 KB</p> <p>8:22:21:12</p>	 <p>Memoria: 16.0 GB (12.38)</p> <p>En uso: 16.0 GB</p> <p>Disponible: 0.0 GB</p> <p>Velocidad: 2400 MHz</p> <p>2400 MHz 2.048 GB SODIMM DDR4</p> <p>9.6/18.3 GB 4.1 GB</p> <p>9.6/18.3 GB 4.1 GB</p> <p>688 MB 1.0 GB</p>
B.V.1 (double) ( driver='evx' and overwrite_a=False )	 <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 42% 3.43 GHz</p> <p>Procesos: 248 2650 118731</p> <p>Memoria: 4 16 GB</p> <p>Cache L1: 256 MB 1.048 KB</p> <p>Cache L2: 1.024 MB 8.096 KB</p> <p>8:22:25:16</p>	 <p>Memoria: 16.0 GB (12.38)</p> <p>En uso: 16.0 GB</p> <p>Disponible: 0.0 GB</p> <p>Velocidad: 2400 MHz</p> <p>2400 MHz 2.048 GB SODIMM DDR4</p> <p>9.1/18.3 GB 4.1 GB</p> <p>9.1/18.3 GB 4.1 GB</p> <p>689 MB 1.0 GB</p>
B.V.2 (double) ( driver='evx' and overwrite_a=True )	 <p>Intel(R) Core(TM) i7-8550U CPU @ 1.80GHz</p> <p>Velocidad: 44% 3.53 GHz</p> <p>Procesos: 258 3029 124466</p> <p>Memoria: 4 16 GB</p> <p>Cache L1: 256 MB 1.048 KB</p> <p>Cache L2: 1.024 MB 8.096 KB</p> <p>8:22:53:27</p>	 <p>Memoria: 16.0 GB (12.38)</p> <p>En uso: 16.0 GB</p> <p>Disponible: 0.0 GB</p> <p>Velocidad: 2400 MHz</p> <p>2400 MHz 2.048 GB SODIMM DDR4</p> <p>9.6/18.3 GB 4.3 GB</p> <p>9.6/18.3 GB 4.3 GB</p> <p>697 MB 1.0 GB</p>

Se puede evidenciar que en general en las corridas el procesador alcanza su máximo en los puntos con mayores valores de N, y es en esos casos que la memoria ram tiende a subir un poco, para ayudar a procesar la información. Otro aspecto interesante a señalar, que para el caso B opción II y V el procesador no usaba en promedio ni el 50% y la memoria nunca se vio afectada, lo curioso es que estos casos fueron los que más tiempo demoraron en ejecutarse (esto explica mejor su tiempo de duración ya que al utilizar menos recursos su tiempo de realización es más lento, quizás esto puede deberse a que la operación no tenía una dificultad tan grande pero si un proceso extenso).

A continuación se muestran los gráficos encontrados:



De los gráficos para el caso A, se puede apreciar que son muy similares entre ellos, casi no hay diferencias, y si las hay no tienen un claro patrón hasta el final (es decir hasta los valores más grandes de N), queda claro que la opción de calcular la inversa y luego multiplicar esta por la matriz b es la manera más lenta de realizar la operación, mientras que la más rápida es utilizando `scipy.linalg.solve` asumiendo una matriz A positiva (`assume_a='pos'` ). Es importante señalar que los resultados entre el caso A usando datos de tipo float vs usando datos de tipo double son prácticamente idénticos.

Por otro lado, observando lo ocurrido en el caso B, se puede apreciar que si existen mayores diferencias entre las diferentes opciones, en donde las opciones II.1, II.2, V.1 y V.2 ( `driver='ev` y `driver=evx` ) son evidentemente más lentas, mientras que todas las demás tienen tiempos prácticamente iguales obteniendo un mejor rendimiento. Al igual que el caso anterior, la diferencia entre el tipo de dato `double` y `float` es prácticamente imperceptible.

## Preguntas

- **Haga un comentario completo respecto de todo lo que ve en términos de desempeño en cada problema.**

Tal como se menciona antes se puede apreciar como en el caso A son todos los casos muy parecidos, siendo el método de utilizar la inversa el más lento, mientras que el más rápido es asumiendo que la matriz A es positiva.

Para el caso B, usar `driver='ev` y `driver=ev` relentizan considerablemente el proceso, mientras que los demás trabajan de manera más eficiente.

(Más comentarios debajo de los gráficos...).

- **¿Como es la variabilidad del tiempo de ejecucion para cada algoritmo?**

En el caso A se puede evidenciar que no hay tanta variabilidad de tiempo, pero que para valores de N entre aproximadamente 60 y 220 el caso más optimo será asumir una matriz A simétrica, mientras que para los todos los demás casos de N (menores y mayores), el tiempo de ejecución de usar `scipy.linalg.solve` asumiendo una matriz positiva es mejor.

Para el caso B utilizando `scipy.linalg.eigh`, claramente hay una diferencia notable de tiempo entre los subcasos con `driver='ev'` y `driver=evx`, los cuales serán siempre más lentos, mientras que todos los demás casos tienen un comportamiento casi identico con un tiempo de ejecución considerablemente mejor.

Cabe agregar que el método `scipy.linalg.solve` es evidentemente más rápido que `scipy.linalg.eigh`.

- **¿Qué algoritmo gana (en promedio) en cada caso?**

En el caso A en promedio el algoritmo más rápido es `scipy.linalg.solve(assume_a='pos')`.

En el caso B en promedio el algoritmo más rápido es `scipy.linalg.eigh(driver="evd", overwrite_a=True)` --> Es levemente más rápido, prácticamente imperceptible.

- **¿Depende del tamaño de la matriz?**

Para el caso A, si es influyente el tamaño de la matriz, porque como ya mencioné antes el subcaso IV tiene mejor rendimiento que cualquiera con matrices de tamaño entre  $60 < N < 220$ . Para el caso B, el porte de la matriz no es un factor influyente, siempre existe el mismo comportamiento (si hay diferencias es prácticamente imperceptible).

- ¿A qué se puede deber la superioridad de cada opción?

La superioridad de cada opción tanto en el caso A como en el caso B, se debe plenamente a los paquetes y el proceso que reliza "por detrás" los diferentes métodos con sus diferentes parámetros. Por ejemplo, si se utiliza un argumento `overwrite_a=True`, se utilizarán menos recursos, ya que no se está creando una nueva matriz, sino que se está reescribiendo la misma, otros ejemplos podrían ser que al asumirla de un tipo (positiva o simétrica), en donde el cálculo es más fácil, y el paquete al estar informado, realiza "trucos" de resolución más óptimos.

- ¿Su computador usa más de un proceso por cada corrida?

Tal como se evidencia en las imágenes del procesador, este utiliza los 8 procesadores lógicos para la mayoría de los casos, hay un par de excepciones en donde utiliza 2 al máximo y los demás en mitad del rendimiento, pero en estricto rigor siempre está utilizando cerca del 100% de la memoria de la CPU en los casos con N altos. Que el computador utilice más de un proceso significa que está realizando muchas acciones diferentes simultáneamente y esto viene definido en parte por el computador (por qué decide usar), pero también por las librerías y paquetes que se utilicen, los cuales pueden requerir de hacer varias acciones en simultáneo para hacerlas de manera más eficiente.

- ¿Qué hay del uso de memoria (como crece)?

En mi caso, la memoria se vio muy poco afectada, ya que mi procesador es bastante potente, solo cambiaba levemente cuando el procesador sobrepasaba el 100% de su rendimiento, y cuando lo hacía de todas maneras lo que cambiaba la memoria era muy poco. Probablemente esto haya ocurrido para los valores de N que eran más grandes.

Todo lo anteriormente señalado, ocurre por la jerarquización de la memoria en los computadores.