MCOC2021-P0

Mi computador principal

- Marca/modelo: MacBook Air
- Tipo: Notebook
- Año adquisición: 2014
- Procesador:
  - Marca/Modelo: Intel Core i5-4260U
  - Velocidad Base: 1.40 GHz
  - Velocidad Máxima: 1.40 GHz
  - Numero de núcleos: 2
  - Humero de hilos: 4
  - Arquitectura: x86 Haswell
  - Set de instrucciones:
- Tamaño de las cachés del procesador
  - L3: 3 MB

- Memoria
  - Total: 4 GB
  - Tipo memoria: DDR3
  - Velocidad 1600 MHz
  - Numero de (SO)DIMM: 4
- Tarjeta Gráfica
  - Marca / Modelo: HD Gragics 5000
  - Memoria dedicada:
  - Resolución: 1366 x 768
- Disco 1:
  - Marca: Samsung
  - Tipo: SSD
  - Tamaño:
  - Particiones: 5
  - Sistema de archivos: MS-DOS FAT32
- Disco 2:
  - Marca: Samsung
  - Tipo: SSD
  - Tamaño:
  - Particiones: 2
  - Sistema de archivos: MS-DOS FAT32
- Dirección MAC de la tarjeta wifi: 64:76:ba:a8:c6:70
- Dirección IP (Interna, del router):
- Dirección IP (Externa, del ISP):
- Proveedor internet: Mi Internet

Entrega 2

- ¿Cómo difiere del gráfico del profesor/ayudante?

- La diferencia se da en l velocidad, ya que en mi caso, el gráfico, parte en aproximadamente 1s en el Tiempo transcurrido, y el termino en 1 minuto aproximadamente. Esto se da ya que, e procesador de mi comptador tiene una velocidad menor, lo que genera un mayor tiempo en la ejecución del código.

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

- Pueden haber varios factores qeu interfieran en la diferencia entre cada corrida, ya que el computador puede estar trabajando simultaneamente. Por ejemplo el uso del navegador de internet, el mail, etc, en simultaneo mientras se esta ejecutando el codigo. Tambien, se puede deber a que los procesadores entre computadores son distintos, por lo que la capacidad entre un comptador y otro no es la misma.

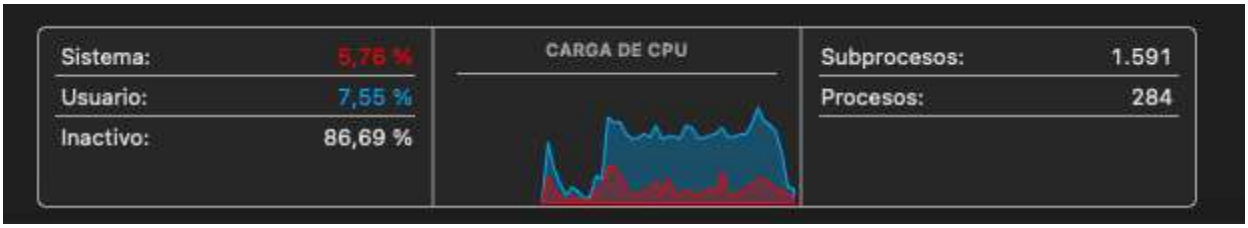
- El gráfico de uso de memoria es lineal con el tamaño de matriz, pero el de tiempo transcurrido no lo es ¿porqué puede ser?

- El tiempo en el que se demora cada ciclo va a depender de la cantidad de cosas que esté realizando el computador al momento de ejecutar cada uno. Mientras que, el proceso que realiza al ejecutar el codigo es siempre el mismo, por lo que en cada ciclo llega a donde mismo, por lo que el gasto siempre será el mismo.

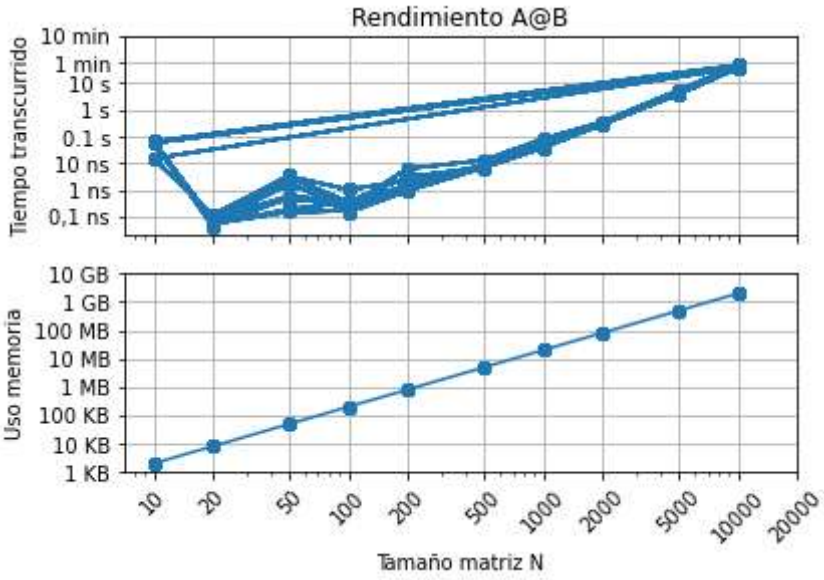
*¿Qué versión de python está usando? Python 3.8.8 MSC v.1916 64 (AMD64)

*¿Qué versión de numpy está usando? 1.20.1

*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.



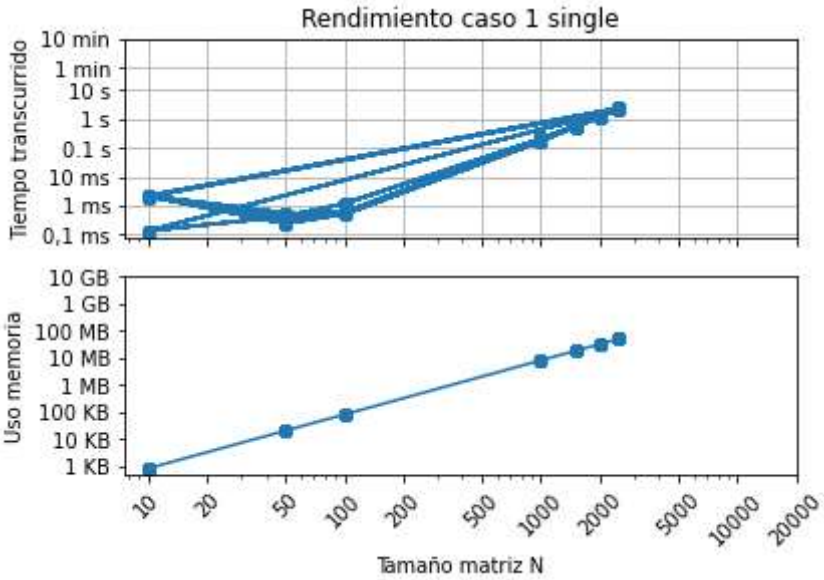
- Gáfico

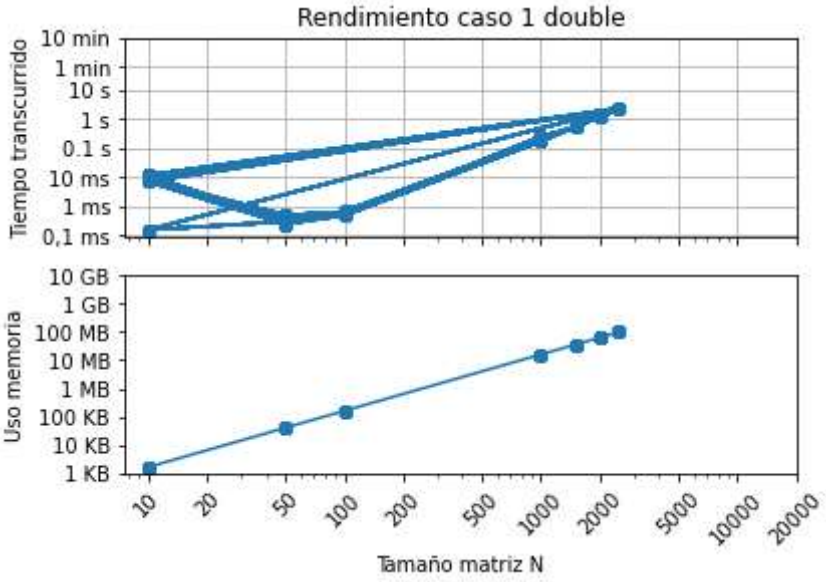


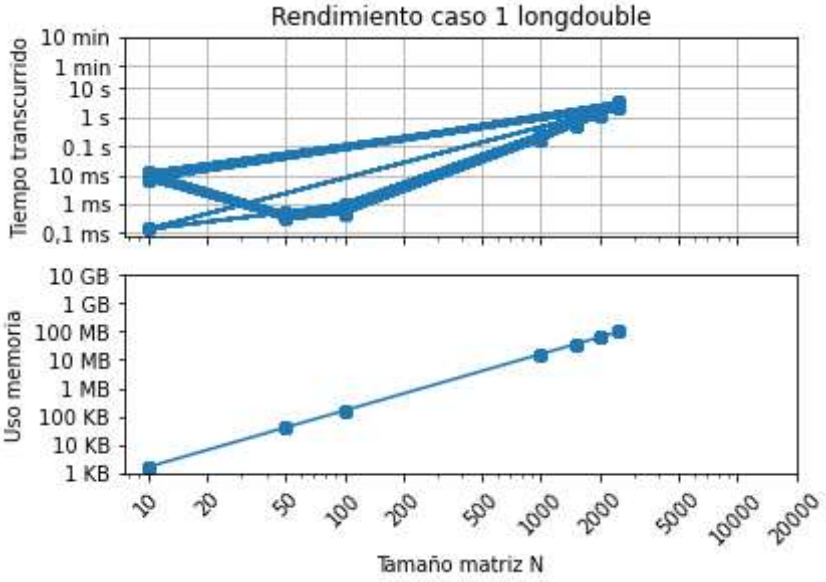
Entrega 3

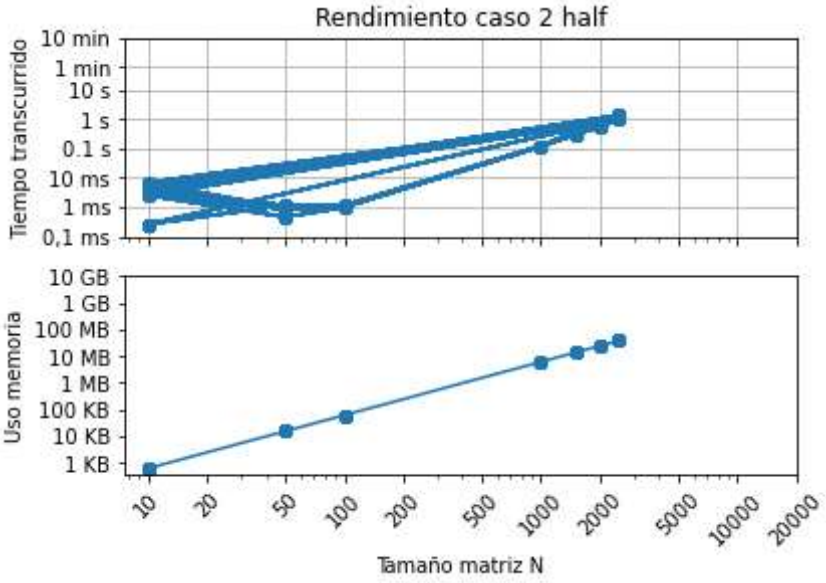
Caso 1 - Half:

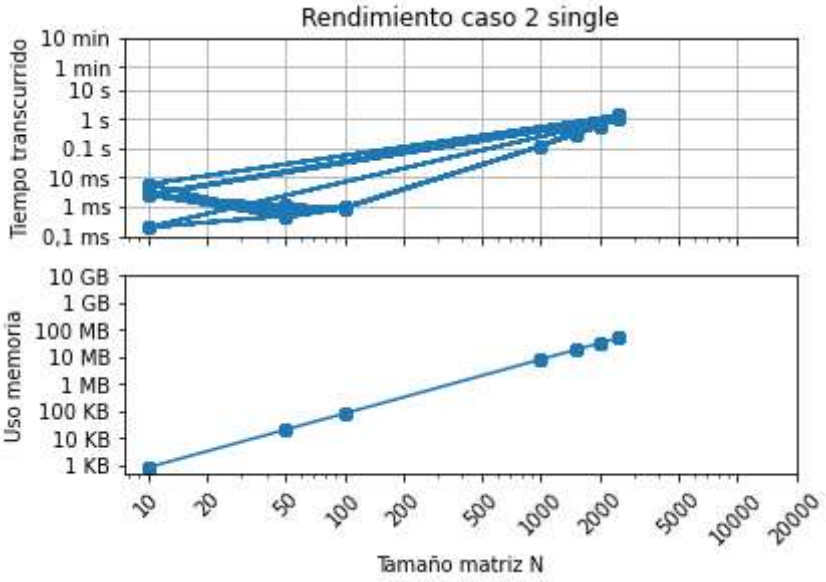
```
<module>
    Am1=inv(A)

  File "<__array_function__ internals>", line 5, in inv

  File "C:\Users\sandra\anaconda3\lib\site-packages\numpy\linalg\linalg.py",
line 541, in inv
    t, result_t = _commonType(a)

  File "C:\Users\sandra\anaconda3\lib\site-packages\numpy\linalg\linalg.py",
line 146, in _commonType
    raise TypeError("array type %s is unsupported in linalg" %

TypeError: array type float16 is unsupported in linalg
```

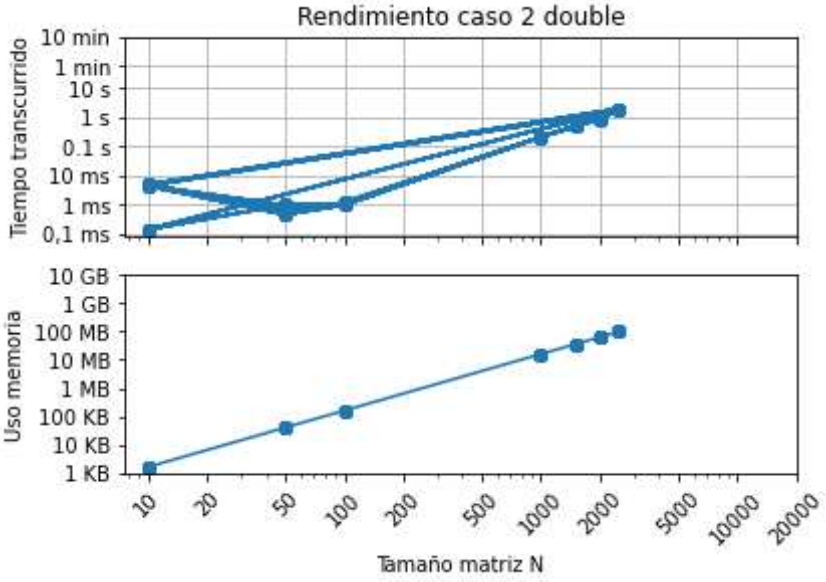


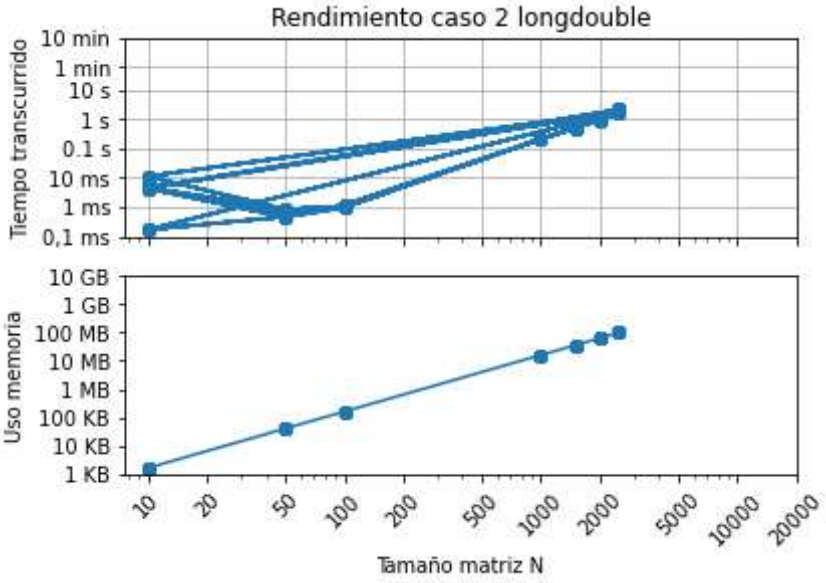

Rendimiento caso 1 double
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N

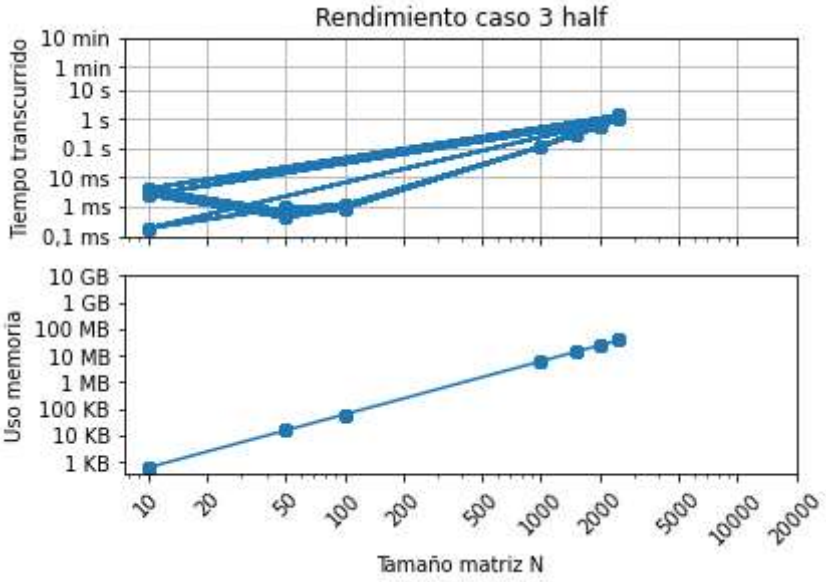

Rendimiento caso 1 longdouble
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N

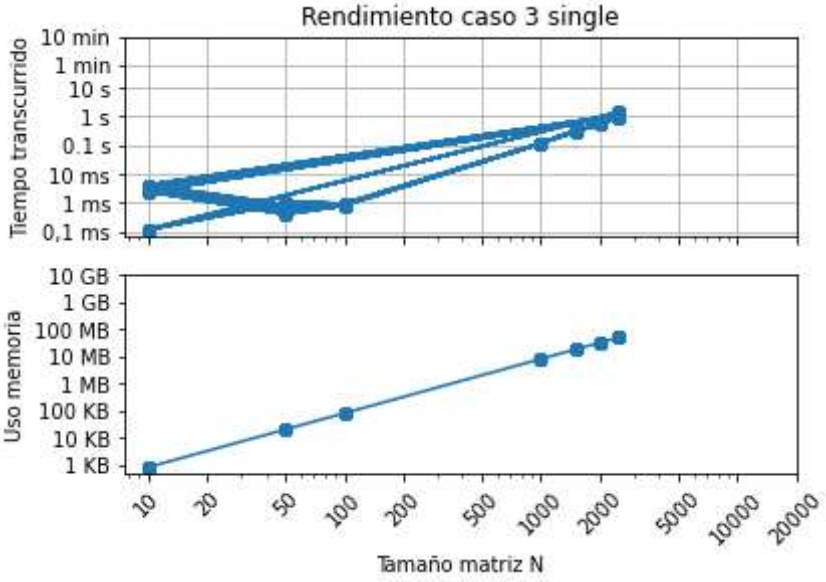

Rendimiento caso 2 half
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N

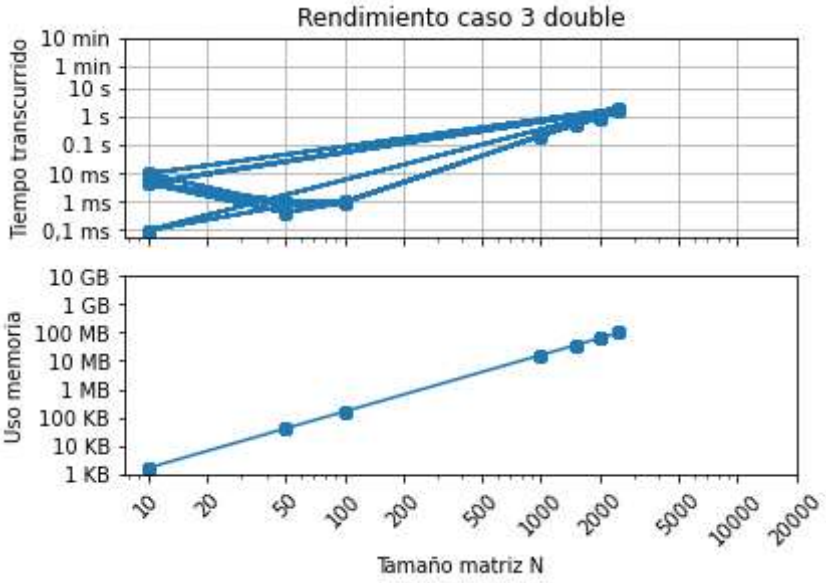

Rendimiento caso 2 single
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N

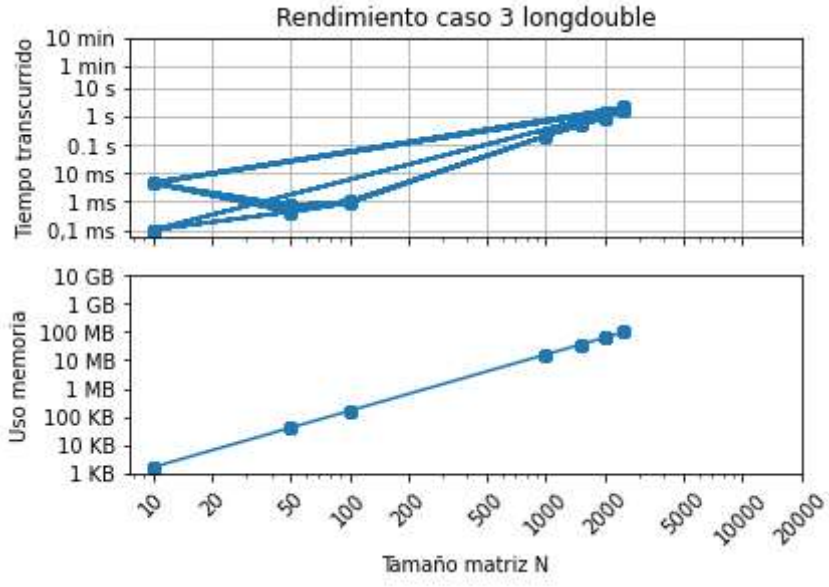

Rendimiento caso 2 double
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N


Rendimiento caso 2 longdouble
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N


Rendimiento caso 3 half
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N


Rendimiento caso 3 single
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N


Rendimiento caso 3 double
Tiempo transcurrido
10 min
1 min
10 s
1 s
0.1 s
10 ms
1 ms
0,1 ms
Uso memoria
10 GB
1 GB
100 MB
10 MB
1 MB
100 KB
10 KB
1 KB
10
20
50
100
200
500
1000
2000
5000
10000
20000
Tamaño matriz N



Entrega 4

- ¿Como es la variabilidad del tiempo de ejecucion para cada algoritmo? ¿Qué algoritmo gana (en promedio) en cada caso? ¿Depende del tamaño de la matriz? ¿A que se puede deber la superioridad de cada opción? ¿Su computador usa más de un proceso por cada corrida? ¿Que hay del uso de memoria (como crece)?

- Al comparar el tiempo entre el uso de eigh y solve, eigh toma un tiempo considerable al ejecutar el codigo cuando la matriz es grande, como al utilizar N=10000. Este tiempo es mucho mayor que el que se gasta al utilizar solve.

- En cuanto al algoritmo que gana en promedio, se puede decir que ese es solve, ya que es mucho mas eficiente, debido que, para un N=8000, que fue el maximo valor para realizar la matriz, el tiempo promedio que tomo el codigo en ejecutar fue de aproximadamente 20 a 30 segundos, a diferencia de eigh que uso un teimpo promedio de aproximandamente 2,5 minutos para el mismo N.

- El tiempo de ejecución sí depende del tamaño de la matriz, ya que mientas más grande sea esta, mayor será el tiempo que tarde el codigo en finalizar.

Entrega 5

- Código matriz laplaciana.

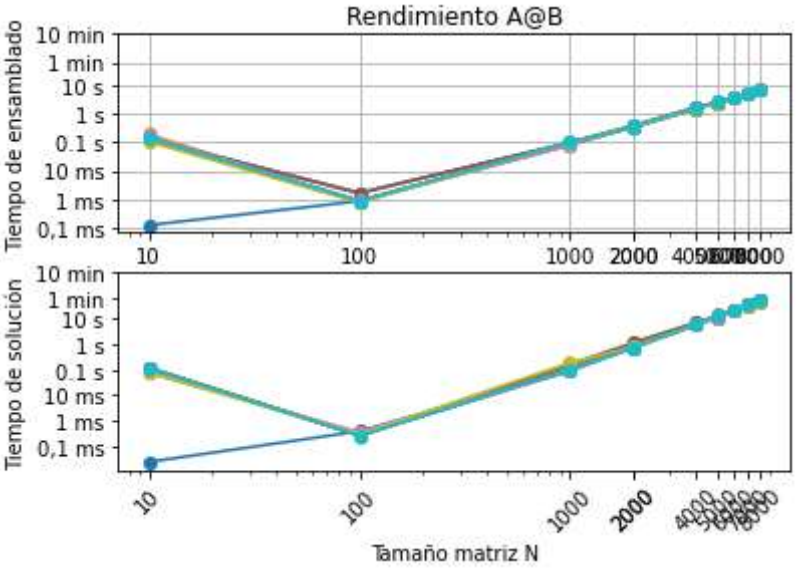
def matriz_laplaciana(N,t=double): e=eye(N)-eye(N,N,1) return t(e+e.T)

for i in range(10):

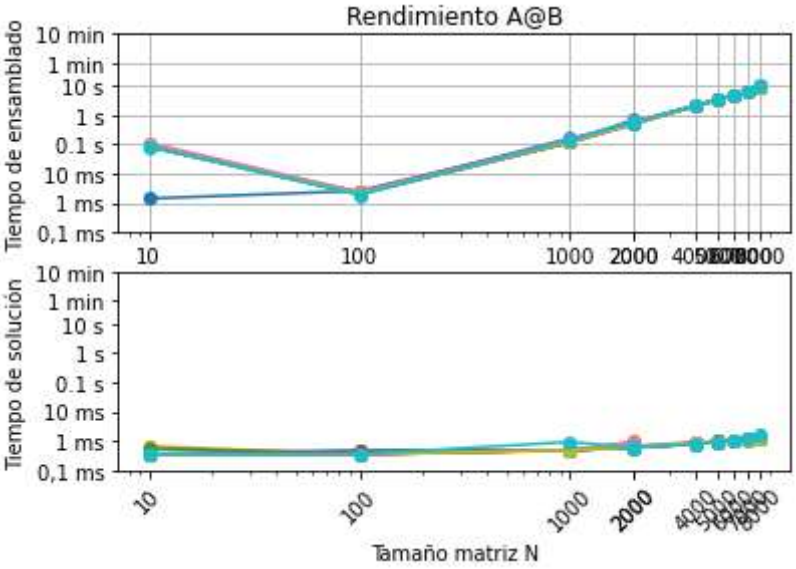
```
  for N in Ns:
```

```
t1=perf_counter()
A=matriz_laplaciana(N)
b=matriz_laplaciana(N)
Acsr1=sparse.csr_matrix(A)
Acsr2=sparse.csr_matrix(b)
t2=perf_counter()
x=Acsr1@Acsr2
t3=perf_counter()

dt=t2-t1 #tiempo ensamblado
dt2=t3-t2 #tiempo de solución
Dts.append(dt)
Dts2.append(dt2)
```

- Matriz llena:



- Matriz dispersa:



- En relación a los resultados obtenidos, se puede ver que el tiempo que demora la multiplicación es bstante menor en la matriz dispersa, lo que se debe

principalemente a que en esta, no se consideran los ceros como es en la matriz llena, lo que hace que el teimpo se reduzca considerablemente.

Entrega 6

- Código matriz laplaciana para inversa de matriz llena

def matriz_laplaciana(N,t=double): e=eye(N)-eye(N,N,1) return t(e+e.T)

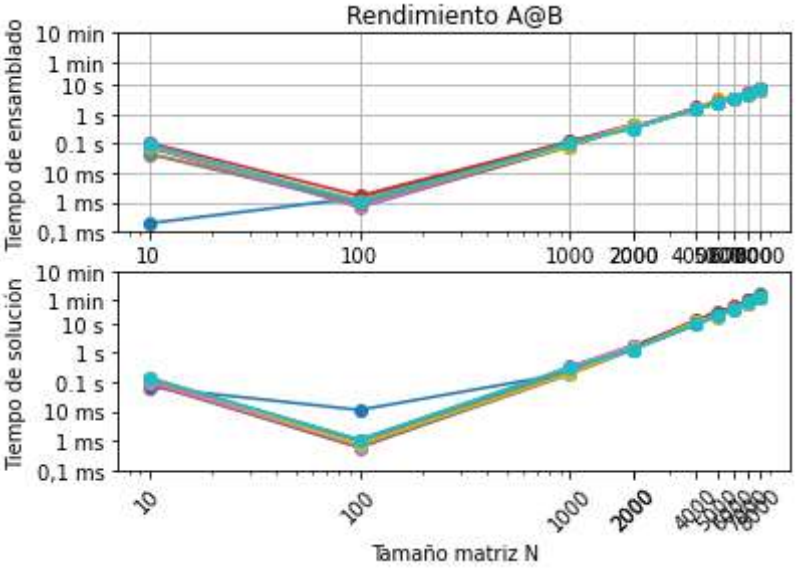
for i in range(10):

```
for N in Ns:

    t1=perf_counter()
    A=matriz_laplaciana(N)
    b=matriz_laplaciana(N)
    Acsr1=sparse.csr_matrix(A)
    Acsr2=sparse.csr_matrix(b)
    t2=perf_counter()
    x=inv(A)
    t3=perf_counter()

    dt=t2-t1 #tiempo ensamblado
    dt2=t3-t2 #tiempo de solución
    Dts.append(dt)
    Dts2.append(dt2)
```

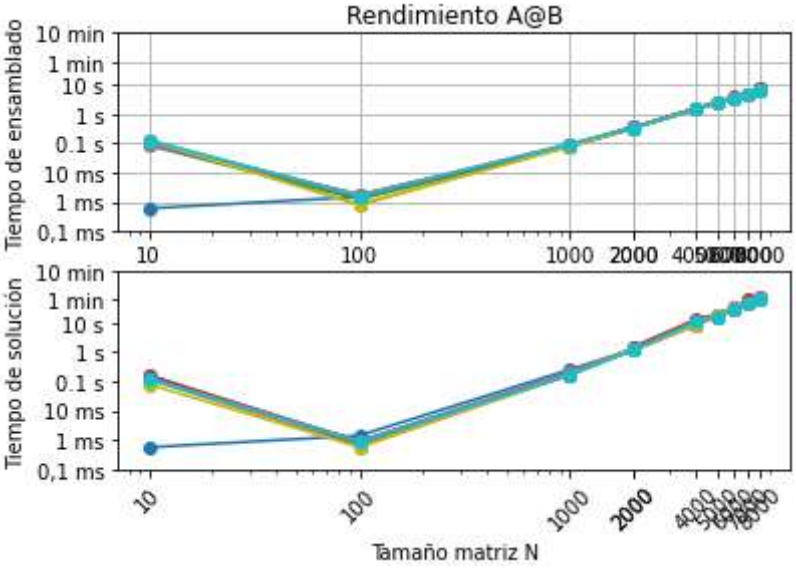
- Para el caso de solve, la matriz dispersa fue la que más se demoró en completar los diez ciclo, en aproximadamente 4350 segundos, mientras que la matriz llena en completar los diez ciclos ocupo un tiempo de 2300 segundos aproximadamente.
- En cuanto al caso Inv, el tiempo que demoró la matriz dispersa fue de aproximadament 4500 segundos y la matriz llena 7500 segundos aproximadamente.
- Esto indica que el mejor método es utilizando solve, ya que el tiempo que domoraron en completarse los 10 ciclos en la matriz dispersa y llena son menores que usando Inv
- Para Inv en la matriz llena cada uno de los ciclos tienen tiempos casi iguales, lo que hace que se bastante estable. Para la matriz dispersa en promedio, entre un ciclo y otro hubo una variacion de 0,4 segundos en cada uno de los N de la matriz, lo cual, aun que sea una diferencia minima, si es considerable em el tiempo final una vez terminados los 10 ciclos

- Para solve, la matriz llena, tiene diferencias de menos de 0,1 segundos por cada N en los 10 ciclos, lo cual no genera mayores aumentos en el tiempo final una vez terminados los 10 ciclos, a diferencia de la amtriz dispersa que hay un tiempo aproximado de 0,5 segundos de diferencia entre cada N de cada ciclo.

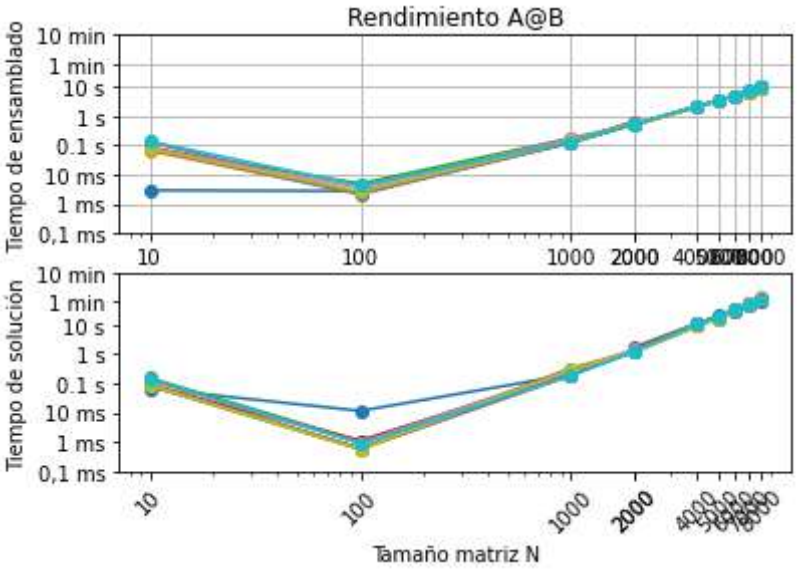
- Matriz llena Solve:



- Matriz llena Inv:



- Matriz dispersa Solve:



- Matriz dispersa Inv: