# INFORME LAB4

## Usando Python con Jupyter para cálculo científico

En general la resolución de esta práctica me ha resultado muy interesante.

Respecto al informe no voy a extenderme en apartados concretos, sino que más bien quiero mencionar aquellas cosas que han llamado mi atención.

En primer lugar, en esta práctica se vuelve a utilizar GitHub, lo cual es satisfactorio al venir con la soltura de la práctica anterior y poder interactuar de forma muy fluida cuando creamos repositorios, los vinculamos a un repositorio de GitHub, realizamos commits, etc.

Por otra parte, en esta práctica se trabaja con la librería sys, lo que, unido a nuestros conocimientos de Python, nos permite comenzar a ejecutar scrips con parámetros desde la propia línea de sbatch al lanzar trabajos a un cluster, esto me resulta muy interesante.

Adentrándonos al objetivo de la práctica, que a mi parecer es la optimización de tiempos de cómputo al realizar operaciones via cpu. La librería de NumPy aparece como una potente herramienta para optimizar estos procesos gracias a sus funciones y su forma de gestionar el almacenamiento continuo de datos en memoria, tal como nos explicaron en sistemas bioinformáticos y de nuevo en esta asignatura. El único impedimento para no usar numpy sería no querer familiarizarse con las distintas funciones, por tanto, para procesos iterativos con gran cantidad de datos es una herramienta indispensable.

Otra de las herramientas propuestas ha sido el uso de decoradores en funciones para suplir la carencia de Python (compila línea a línea el código), compilando funciones para así ahorrar tiempo de ejecución. Me parece también muy potente su uso, pero combinado con numpy hay que tener consideraciones. Me he dado cuenta de que no son totalmente combinables todas las funciones de NumPy con las optimizaciones de numba, pudiendo aparecer errores al compilar esta ultima funciones de NumPy

Respecto a cosas llamativas, en esta práctica se afianzan dos formas de medir tiempo mediante time.time() (algo menos preciso) y %%timeit (más precisa al repetir varias veces el proceso a medir y sacar la media) pero solo usable en jupyter. Ambas son herramientas que voy a considerar en futuros proyectos del master que impliquen Python.

Por último, destacar que este lab lo tenia resuelto para la primera entrega, pero no lo hice puesto que el aparatado extra me daba problemas, no lograba llegar al tiempo de optimización. Fue culpa mía, puesto que media solo 1 la función con el time, por lo que no consideraba a numba que necesita una ejecución para compilar y otra para valorar la velocidad de verdad. Al medir de nuevo la función, una vez compilada con numba obtuve una velocidad considerable.

He aprendido bastante con este lab, creo que es muy recomendable y útil para saber manejar Python de una forma más potente.