Trabajo final del curso: Programación

Especialización en Bioinformática

Alumna: Lucía Dominguez

Título del trabajo: Interpretación de los resultados obtenidos en la medición del tiempo de vida de fluorescencia en células del epitelio pigmentario retinal.

Abril 2024

1. Objetivos

El objetivo principal de este trabajo es crear un algoritmo que permita modelar los datos obtenidos al realizar microscopía de tiempo de vida de fluorescencia en células del epitelio pigmentario de la retina y de este modo facilitar la comprensión de los resultados.

1.2. Objetivos específicos:

1- A partir de la matriz de intensidades localizar al núcleo celular. Convertir esta matriz en una máscara de 0 y 1 en la que el cero indique los píxeles pertenecientes al núcleo, y el 1 al citoplasma. Multiplicarla por las otras matrices, con el objetivo de anular los valores correspondientes al núcleo en todas ellas.
2- Comparar los valores promedio de cada una de las matrices, con y sin sus núcleos.
3- Crear un array multidimensional que agrupe las matrices bidimensionales correspondientes a tiempos y amplitudes de la célula.
4- Seleccionar en función de t2, que píxeles corresponden al fluoróforo lipofuscina y cuales a FAD. Graficar los resultados.
5- Comparar los resultados obtenidos con los resultados generados por un algoritmo K Means.

2. Metodología desarrollada

2.1. Librerías utilizadas: En este trabajo se emplearon Numpy [1], Matplotlib [2], Scikit-Learn [3] y Funpymodeling.

2.2. Origen y estructura de los datos utilizados

Los datos se obtuvieron mediante el software SPCImage. Este permite analizar las imágenes de tiempo de vida de fluorescencia, o exportar los datos crudos de la imagen. Estos datos corresponden a cada píxel de la imagen; cada fila de píxeles genera una matriz unidimensional y la imagen completa una matriz bidimensional. En este caso la célula evaluada tiene 85 píxeles en eje x, 83 en eje y. Por imagen se exportan 5 matrices, una correspondiente a la intensidad de fluorescencia, otra al tiempo de fluorescencia corto (“t1”), otra a la amplitud correspondiente a t1 (“a1”), otra al tiempo de fluorescencia largo (“t2”) y otra a la amplitud correspondiente a t2 (“a2”).

Los datos de intensidad son números enteros, no así el resto de las matrices. Todas ellas fueron convertidas de números decimales a enteros para facilitar el manejo de las mismas.

En la siguiente tabla se expresan valores minimos, máximos, promedio y desviación estándar de cada una de las matrices:

	Valor Mínimo	**Valor Máximo**	**Promedio**	**Desviación estándar**
Matriz Intensidad	120	585	322.19	79.95
Matriz t1	241	1044	401.99	69.60
Matriz a1	48	91	78.96	5.15
Matriz t2	2059	5735	3220.55	520.65
Matriz a2	8	51	20.03	5.15

2.3. Descripción de Metodología utilizada

Las matrices previamente guardadas en formato .csv, fueron importadas con el método “np.genfromtxt”. Para simplificar el análisis posterior, en la misma sentencia, los datos fueron transformados en enteros mediante la función INT. Los nombres de las mismas son ‘INT’, ‘T1’, ‘T2’, ‘A1’ y ‘A2’.

Posteriormente la matriz de intensidades fue convertida en una máscara (denominada ‘INT1’), dónde los píxeles correspondientes al núcleo de la célula (intensidad < 200) fueron convertidos en ceros y los correspondientes al citoplasma (intensidad > 200) en unos, mediante la función “np.where”.

Luego se multiplicó la máscara por cada una de las matrices de tiempos y amplitudes para lograr así anular los pixeles correspondientes al núcleo en todas ellas. Los nombres de estas nuevas matrices son: ‘INTSN’, ‘T1SN’, ‘T2SN’, ‘A1SN’ y ‘A2SN’.

A continuación, con la función “np.array” se creo un array multidimensional denominada ‘célula’ con las cinco matrices previamente modificadas.

Con el objetivo de evaluar si el tiempo medio de las matrices es influenciado por la presencia del núcleo o no, se obtuvo el tiempo medio de las matrices sin núcleo. El método “np.mean”, habría contabilizado los ceros correspondientes al núcleo y hubiera generado un tiempo pomedio alterado. Por esto se decidió, mediante un bucle for recorrer la matriz completa y escoger los tiempos distintos de cero, anexarlos a una lista inicialmente vacía, y luego sí calcular el valor promedio de esa lista con el método “np.mean”.

Se creo una lista en la que se fueron añadiendo los valores promedios de cada matriz con y sin núcleo. Se realizo un gráfico de barras con librería Myplotlib con los valores promedio de cada matriz.

De acuerdo a la información expresada en la bibliografía [4], los fluoróforos que fueron excitados son FAD y lipofuscina (esto depende de la longitud de onda del laser de excitación -405 nm- y del filtro de emisión -500 a 550 nm-). Con el objetivo de clasificar qué píxel corresponde a lipofuscina y cuál a FAD, se los diferenció en función de su t2.

Los píxeles con t2 superior a 2200 ps y menor 2500 ps, se clasificaron como lipofuscina, y los píxeles con t2 entre 2500 y 3000 ps, como FAD. Los píxeles restantes fueron atribuidos al núcleo y al ruido propio de la imagen. Para lograrlo, se llevo a cabo el siguiente procedimiento.

En primer lugar tomar las matrices T1 y T2 previamente multiplicadas por la máscara de intensidades, y aplanarlas a matrices unidimensionales con el método “np.concatenate”.

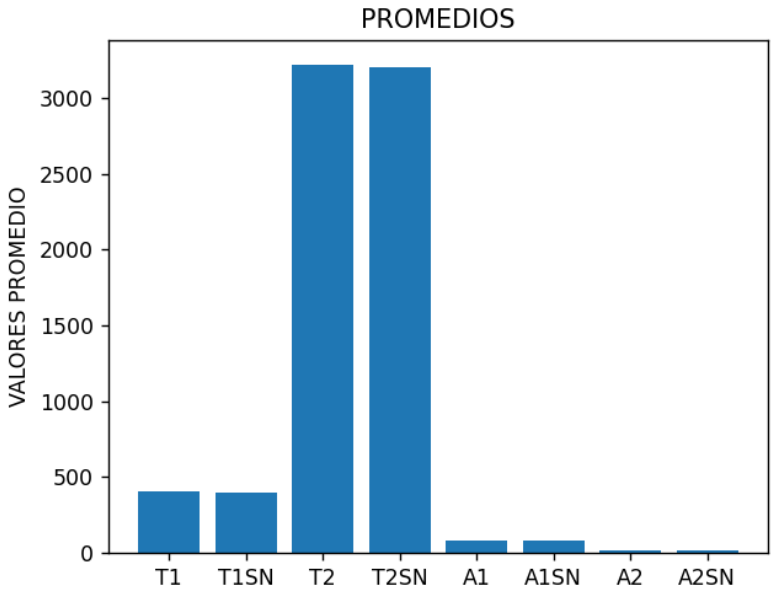
Luego, para que los píxeles de ambas matrices se correspondan, se unieron las dos matrices unidimensionales en un vector de dos elementos con las funciones “np.array”, “list”, “zip” y “reshape”. Posteriormente este vector fue convertido en un data frame con la función “todf” de la librería funpymodelling.

Luego se declaró una lista vacía, y con un bucle for se recorrió la columna del dataframe correspondiente a t2. Mediante un if se seleccionó los valores > 2200 y < 2500, y se los clasificó como lipofuscina, agregando un 1 a la lista vacía. A los valores >2500 y <3000, se los clasificó como FAD, agregando un 2 a la lista. A los valores iguales a 0, se los clasificó como núcleo, agregando un 0 a la lista. Y por cada valor restante, no clasificables en ningún grupo, se agregó un 4 a la lista.

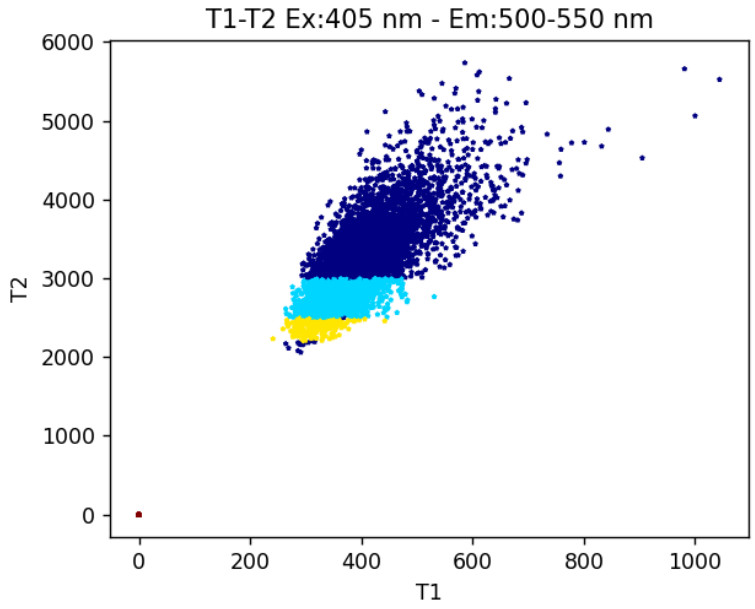
Posteriormente esa lista fue agregada al dataframe formando una tercera columna.
A continuación, con la función “plt.scatter” de la librería Myplotlib – pyplot, se realizó un grafico de dispersión. En el eje x se graficó t1 y en el eje y, t2, y los puntos se colorearon según la clase asignada 0, 1, 2 o 3.
Solo a modo de comparación, se entrenaron los datos con un algoritmo K-Means, utilizando 4 de clústeres: uno correspondiente al núcleo, uno a FAD, uno a lipofuscina, y uno al resto de los datos que no se pueden clasificar en los grupos anteriores. Posteriormente se grafico el resultado del clustering con la función “plt.scatter”.

3. Resultados encontrados:

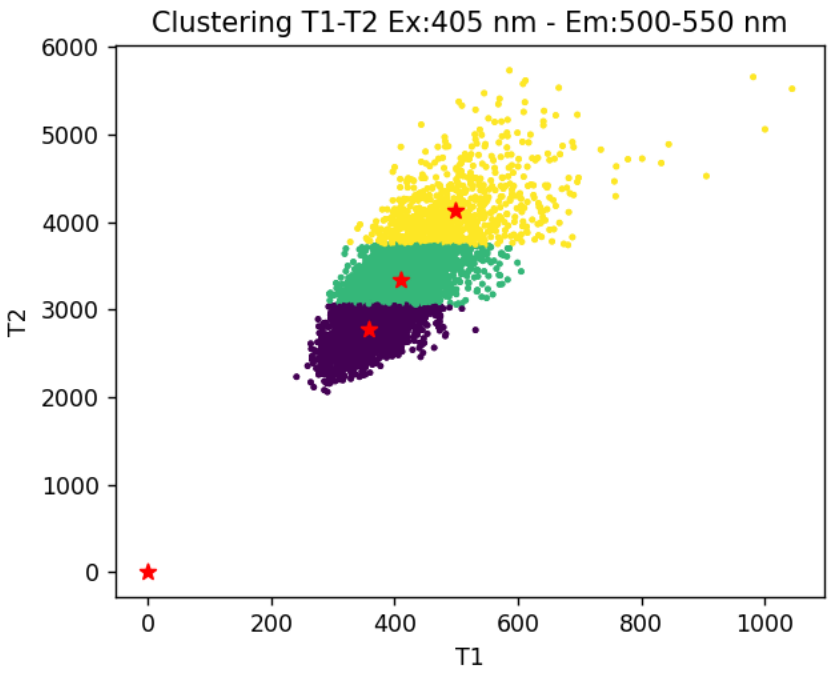
Al comparar los tiempos y amplitudes promedios correspondientes a la célula con su núcleo incluido y sin el mismo, observamos que no diferencias importantes en los tiempos ni en las amplitudes.



Se clasificaron los pares de datos (t1 – t2) en función de su t2. El gráfico de estos datos evidencia en color amarillo los pixeles correspondientes a lipofuscina, en celeste los correspondientes a FAD, el azul los datos que no se pueden clasificar de acuerdo a sus tiempos, y en marrón los correspondientes al núcleo.



Los datos fueron entrenados con un algoritmo K-Means de 4 clústeres, con el objetivo de comparar esta clasificación con la realizada manualmente.



Las diferencias que se observan fundamentalmente son la extensión de los tiempos correspondientes a lipofuscina y a FAD.

4. Código utilizado

https://github.com/ludominguezzz/C-digo-TP-programaci-n

5. Bibliografía:

1- Harris, C.R., et al. Array programming with NumPy. Nature 585, 357–362 (2020).

2- Hunter J. D., Matplotlib: A 2D Graphics Environment. Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95 (2007).

3- Pedregosa *et al.* Scikit-learn: Machine Learning in Python. JMLR 12, pp. 2825-2830 (2011).

4- Dysli C, et al. Fluorescence lifetime imaging ophthalmoscopy. Prog Retin Eye Res. 60:120-143 (2017).