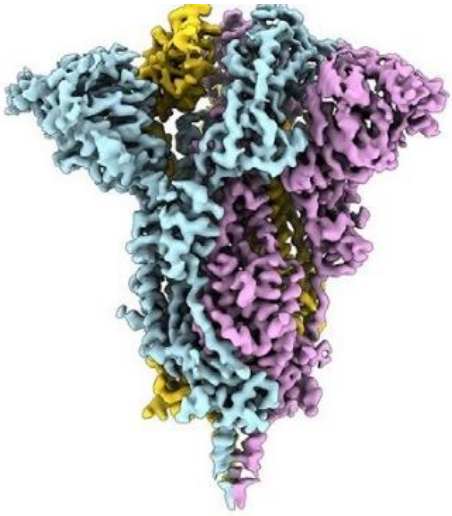
Instituto Politécnico
Nacional
Escuela Superior de
Cómputo
Bioinformatics
Introduction to Colab
Tinoco Videgaray Sergio
Ernesto
Rosas Trigueros Jorge Luis
07/03/2024
14/03/2024

Marco teórico.

La proteína recombinante del SARS-CoV-2 producida y purificada por el Grupo de Alérgenos Vegetales del CBGP se obtuvo tras la identificación de la secuencia de ADN del gen que codifica la proteína Spike del virus por parte del grupo del CSIC-CNB que participa en esta iniciativa, dicha secuencia fue facilitada al Grupo de Alérgenos Vegetales del CBGP para comenzar con las tareas de producción y purificación de la proteína. Cabe destacar que la proteína obtenida por parte del grupo del CBGP-UPM es una proteína truncada (es decir, un trozo y no la proteína completa) que se corresponde con la región expuesta al exterior del virus (y que esta emplea para unirse al receptor de las células humanas y entrar dentro de ellas). Esto se hizo así debido al gran tamaño y complejidad de la proteína Spike (figura 1) [1].



Estructura de la glicoproteína de pico SARS-CoV-2 A372T

Existen dos representaciones principales cuando trabaja con estructuras de proteínas: átomo completo y centroide.

En un mundo ideal con tiempo infinito y potencia informática, se podrían realizar todas las simulaciones con todos los átomos. Pero en la práctica, intentar realizar un muestreo extenso de la cadena principal y al mismo tiempo incluir todos los átomos de la cadena lateral resulta, en el mejor de los casos, poco práctico.

El primer problema es que almacenar todos los átomos es costoso, porque el cálculo de las interacciones entre todos los pares de átomos crece rápidamente (n^2) con el número de átomos. El mayor problema es que el espacio conformacional totalmente atómico es *muy accidentado*, por lo que Monte Carlo rechaza la mayoría de los movimientos.

Para solucionar este problema, las estructuras a menudo se transforman al modo centroide para partes de un protocolo que requieren un amplio muestreo de la columna vertebral (por ejemplo, las etapas iniciales de la predicción de la estructura de novo . En el modo centroide, la columna vertebral sigue siendo completamente atómica, pero la representación de cada cadena lateral se simplifica a un solo *pseudoátomo* de tamaño variable. [2].

RCSB es el centro de datos de estados unidos para el archivo global del Banco de Datos de Proteínas (PDB por sus siglas en ingles) que contienen datos estructurales en 3D para grandes moléculas biológicas (proteínas, ADN y ARN) que son esenciales para la investigación y la educación en biología fundamental, salud, energía y biotecnología. La enorme riqueza de datos de estructuras 3D almacenados en el PDB ha sustentado avances significativos en la comprensión de la arquitectura de las proteínas, que culminaron en avances recientes en la predicción de la estructura de las proteínas acelerados por enfoques de inteligencia artificial y métodos de aprendizaje de maquina [3].

Colaboratory, o "Colab" para abreviar, es un producto de Google Research que permite a cualquier usuario escribir y ejecutar código de Python en el navegador. Es especialmente adecuado para tareas de aprendizaje automático, análisis de datos y educación. Desde un punto de vista más técnico, Colab es un servicio de cuaderno alojado de Jupyter que no requiere configuración y que ofrece acceso sin coste adicional a recursos informáticos, como GPUs. [4].

Material y equipo.

Para esta práctica se utilizaron los siguientes materiales y recursos:

- Un equipo con acceso a internet.
- La plataforma de RCSB PDB la cual proporciona el banco de datos de la mayoría de las proteínas de todo el mundo. Así como distintas herramientas de visualización de datos de proteínas como Jmol.
- El archivo PDB de la glicoproteína spike SARS-CoV-2
- Un entorno de desarrollo para Python como Colab o Spyder.

Desarrollo.

1. Ir al portal de PDB dentro de RCSB buscar la entrada 7FCD.

En este paso me dirigí a la página de www.rcsb.org y en la barra de búsqueda ingresé la cadena “7fcd” como se muestra en la figura 2.

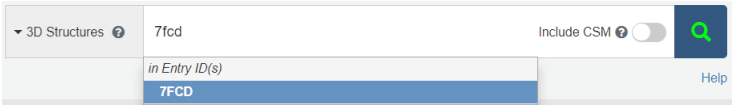


Figura 2. Barra de búsqueda del portal de RCSB

Al seleccionar la primera opción la página me dirigió a la siguiente página donde se incluye información de la Ubiquitina junto con una vista previa de su estructura en 3D (figura 3).

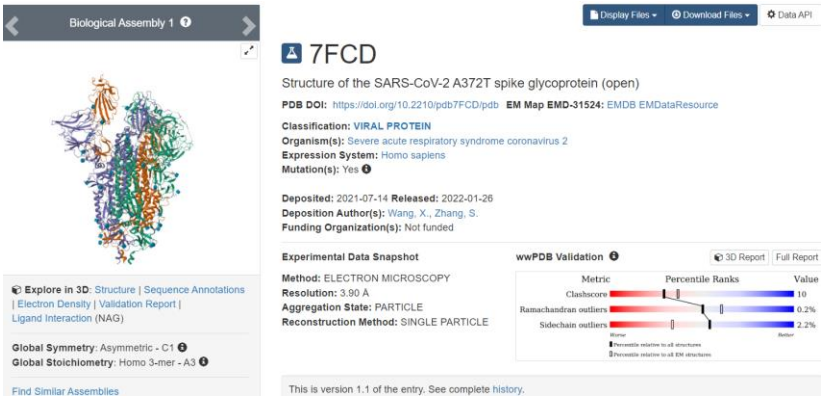


Figura 3. Información de la glicoproteína 7FCD mostrada en RCSB.

Posteriormente, descargue el archivo PDB de la proteína glicoproteína 7FCD (figura 4).

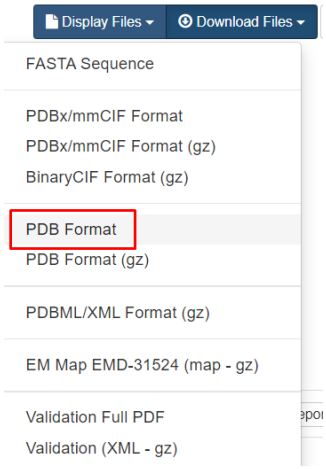


Figura 4. Ventana de descarga del archivo PDB.

Posteriormente, me dirigí al entorno de Colab y conecté mi carpeta de drive donde subí el archivo PDB de la glicoproteína 7FCD. (figura 5).

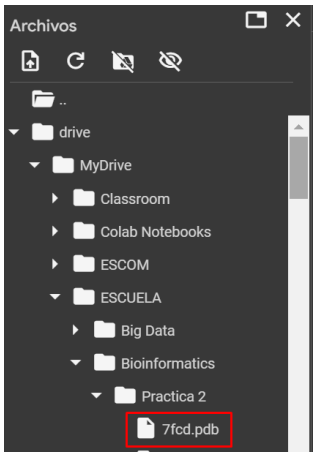


Figura 5. Vista de mi carpeta de drive desde Colab.

Se codificó la función para obtener las coordenadas de cada cadena de la glicoproteína 7FCD mediante expresiones regulares y bucles for (figura 6).

```
#Obtener coordenadas de las cadenas A,B y C
def getChainsCoords(pdb):
    import re
    #Get A coords
    A_chain=re.findall("(?<=(ATOM.{17}A.{9})).((\d+.\d+\s+){3})",pdb)
    A_chainCoords=[[float(coord) for coord in a[1].split()] for a in A_chain] #Omitir grupos de la Regex

    #Get B coords
    B_chain=re.findall("(?<=(ATOM.{17}B.{9})).((\d+.\d+\s+){3})",pdb)
    B_chainCoords=[[float(coord) for coord in a[1].split()] for a in B_chain] #Omitir grupos de la Regex

    #Get C coords
    C_chain=re.findall("(?<=(ATOM.{17}C.{9})).((\d+.\d+\s+){3})",pdb)
    C_chainCoords=[[float(coord) for coord in a[1].split()] for a in C_chain] #Omitir grupos de la Regex

    #Agrupar cadenas por coordenadas x,y,z
    a=(list(zip(*A_chainCoords))[0],list(zip(*A_chainCoords))[1],list(zip(*A_chainCoords))[2])
    b=(list(zip(*B_chainCoords))[0],list(zip(*B_chainCoords))[1],list(zip(*B_chainCoords))[2])
    c=(list(zip(*C_chainCoords))[0],list(zip(*C_chainCoords))[1],list(zip(*C_chainCoords))[2])

    return [a,b,c]
```

Figura 6. Código de la función que obtiene las coordenadas de cada cadena de la glicoproteína 7FCD.

De define la función para obtener los centroides de cada cadena calculando la media de cada coordenada x, y, z de la glicoproteína 7FCD (figura 7).

```
#Calcular centroides de cada cadena
def calcCentroids(x,y,z):

    #Calcular medias
    x_centroid=sum(x)/len(x)
    y_centroid=sum(y)/len(y)
    z_centroid=sum(z)/len(z)

    return (x_centroid,y_centroid,z_centroid)
```

Figura 7. Código de la función de calcula los centroides de las cadenas de la glicoproteína 7FCD.

De igual forma se define la función que grafica tanto las cadenas A, B y C como los centroides de la glicoproteína 7FCD como se muestra en la figura 8.
Para visualizar con mayor claridad la proteína se aplico un muestreo de las coordenadas por medio de un parámetro que define el tamaño de cada muestra.

```python
#Graficar cadenas y centroides
def graficar(chains,centroids,sample):
    from matplotlib import pyplot as plt
    import random
    # Crear figure
    fig = plt.figure(figsize = (10, 7))
    ax = plt.axes(projection ="3d")

    #Lista de colores
    colors=[["r","#DE0000"],["g","#10FF00"],["b","#00DCFF"]]
    sample_size=sample
    #Iterar sobre las cadenas centroides y la lista de colores
    for chain,color,centroid in zip(chains,colors,centroids):
        chain_sample=random.sample(list(zip(chain[0],chain[1],chain[2])),sample_size) #Obtener muestra de coordenadas
        #Separar coordenadas x,y,z
        x=list(zip(*chain_sample))[0]
        y=list(zip(*chain_sample))[1]
        z=list(zip(*chain_sample))[2]
        #Generar grafica
        ax.scatter3D(x,y,z, color = color[0])
        ax.scatter3D(centroid[0],centroid[1],centroid[2], color = color[1],marker="*",s=sample_size*3)
    #Titulo
    plt.title("7FCD CENTROIDS")
    #Show plot
    plt.show()
```

Figura 8. Código de la función que grafica las cadenas de la glicoproteína 7FCD y sus centroides.

Finalmente se invocan las funciones descritas anteriormente desde la función principal (figura 9).

```python
_7fcd=readPDB() #Abrir archivo PDB
chains=getChainsCoords(_7fcd)   #Obtener cadenas
#Calcular centoides por cada cadena
centroids=[[centroid for centroid in calcCentroids(chain[0],chain[1],chain[2])] for chain in chains]
#Imprimir centroides
for centroid in centroids:
  print(centroid)
#graficar(chains,centroids)  #Graficar cadeas con sus centroides
```

Figura 9. Código de la función principal que invoca las funciones definidas.

Resultados

Se imprimió en consola las coordenadas de cada centroide (figura 10).

```
[67.98650651816247, 143.4993150234147, 149.14732527528133]
[53.57548907479079, 174.63504882007717, 162.4284844549885]
[45.06297252610644, 147.07956650919968, 148.34585094480315]
```

Figura 10. Coordenadas de los centroides de la proteína.

Se grafico la proteína utilizando una muestra de 30 átomos por cadena (90 átomos) como se muestra en la figura 11. Donde los puntos de colores representan una cadena diferente y las estrellas de colores representan el centroide al que pertenecen.

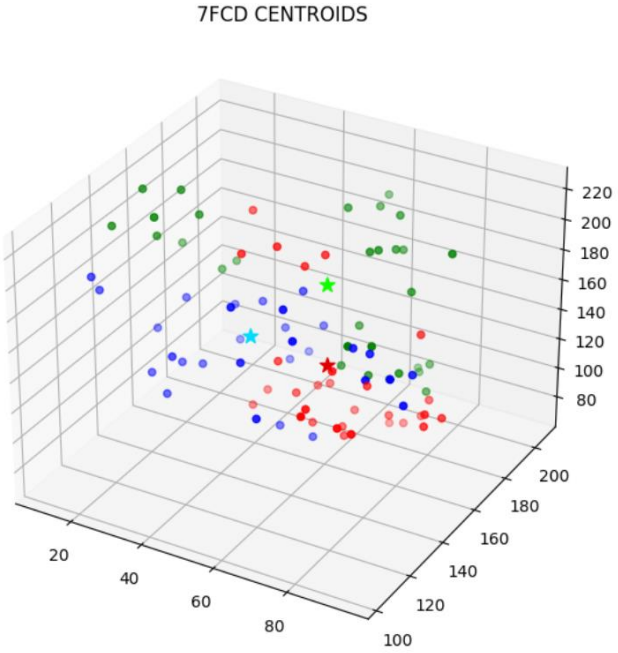


Figura 11. Gráfica de la glicoproteína 7FCD usando una muestra de 30 átomos por cadena.

De igual forma se graficó la glicoproteína, pero ahora con una muestra de 300 átomos por cadena (900 átomos en total) como se ve en la figura 12.

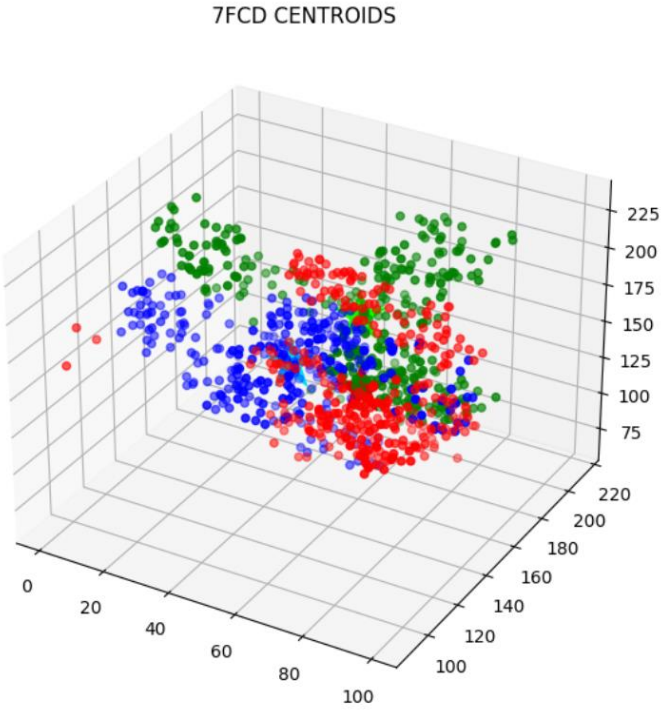


Figura 12. Gráfica de la glicoproteína 7FCD usando una muestra de 300 átomos por cadena.

Finalmente, se grafico la glicoproteína usando una muestra de 3,000 átomos por cadena (9,000 átomos en total) como se muestra en la figura 13.

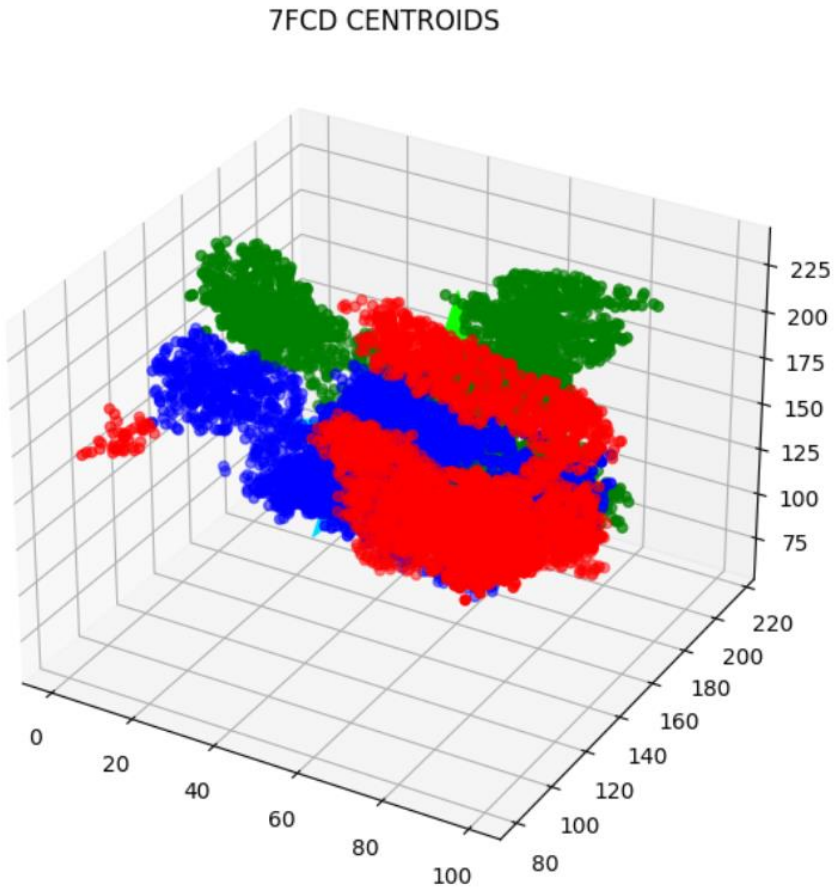


Figura 13. Gráfica de la glicoproteína 7FCD usando una muestra de 3,000 átomos por cada cadena.

Conclusiones y recomendaciones.

En esta práctica se trabajó sobre un archivo PDB que almacena la información de una glicoproteína a manera de texto, lo que permitió extraer la información necesaria para obtener las coordenadas de cada átomo y almacenarlas en una matriz de números. Para posteriormente calcular los centroides de cada cadena de la proteína y visualizarlos en una gráfica tridimensional.

Por otro lado, se utilizo la plataforma de Google Colab que permite escribir y ejecutar código en lenguaje Python en un entorno remoto, ya que los servicios necesarios para ejecutar el código Python junto con sus componentes se proporcionan de forma remota en un servidor de Google.

Esta herramienta resulta útil para fines didácticos ya que sirve para realizar pruebas e implementaciones de programas tanto en lenguajes como Python y C.

Referencias.

[1] CTB (2020). “¿Cómo se obtiene la proteína Spike del virus SARS-CoV-2 utilizada en el ensayo Piloto para el COVID19 que se está desarrollando en el CTB-UPM?” http://www.ctb.upm.es/como-se-obtiene-la-proteina-spike-del-virus-sars-cov-2/

[2] Shourya S. (2016) “Full Atom Representation vs Centroid Representation" https://new.rosettacommons.org/demos/latest/tutorials/full_atom_vs_centroid/fullatom_centroid

[3] RCSB PDB. (2022). "About RCSB PDB: Enabling Breakthroughs in Scientific and Biomedical Research and Education" https://www.rcsb.org/pages/about-us/

[4] Google. (2020). “Colaboratory Preguntas frecuentes” https://research.google.com/colaboratory/intl/es/faq.html#:~:text=Colaboratory%2C%20o%20%22Colab%22%20para,an%C3%A1lisis%20de%20datos%20y%20educaci%C3%B3n