Introduction

Scientific Question: How does the DNA-binding domain of elephant p53 compare to the DNA-binding domain of human p53?

It has long been known that elephants, despite their enourmous size, have incredibly low rates of cancer as a species. Many studies have aimed to unravel this mystery by studying various things. One particular study by Sulak et al. from 2016 revealed that Loxodonta (African Elephants) have twenty copies of their p53 gene throughout their genome. This study also found that as the elephant species evolved through history, the number of copies of their p53 gene increased along with their body size. This supports the idea that more copies of p53 are needed to suppress cancer while allowing increases in body mass. Another study discovered that elephant p53 when added to human cancer cell lines in vitro was able to induce cell death more often than natural human p53 (Abegglen, 2017). Building off of these studies, I wanted to examine what excatly about elephant p53 made it so effective at preventing cancer in African Elephants.

Scientific Hypothesis: If elephants have lower rates of cancer than humans, then the DNA-binding domain of their p53 protein has a significantly different 3D structure compared to the DNA-binding domain of human p53.

The protein and nucleotide sequence data used in my project comes from the NCBI nucleotide database, whereas the 3D protein structure is courtesy of the C-I-TASSER online structure prediction tool and the PDB (https://zhanggroup.org/C-I-TASSER/).

In this project, I first downloaded the nucleotide sequences of the DNA-binding domain of human and elephant p53 from the NCBI Nucleotide database. I used these sequences to perform a pairwise sequence alignment, allowing us to see exactly where these sequences are the same and different. Next, I took the protein sequences of human, mouse, and elephant p53 DNA-binding domain and generated a sequence logo using the LogoMaker package. This allows us to visualize conserved areas of the p53 protein across 3 different species, which helps us answer our scientific question.

Loading in Packages

Pairwise2: Pairwise2 is a module found within BioPython that allows the programmer to perform a pairwise sequence alignment between 2 protein or nucleotide sequences. Pairwise2 provides an algorithm to generate either local or global alignments, depending on what the user needs. In my project, I perform a global alignment to visualize areas where human and elephant p53 genes are similar and different. You can also provide your own gap penalties and match parameters, but the defaults are fine for this project. You can learn more about Pairwise2 at: https://biopython.org/docs/1.76/api/Bio.pairwise2.html#:~:text=Bio.-,pairwise2%20module,all%20characters%20in%20two%20sequences.

SeqIO: SeqIO is a package used to format protein and nucleotide sequences for use with other biopython packages and modules. In this project, I use SeqIO to read in and format the files that contain the nucleotide sequence information for humans and loxodonta p53. The IO in SeqIO stands for Input/Output. It handles sequencenses as SeqRecord objects, and is widely used. For more information visit: https://biopython.org/wiki/SeqIO

NumPy: NumPy is a widely used Python package that is mainly used for scientific math and computing. One of the most crucial features of numpy is the numpy array. A numpy array is an array object that can have any number of dimensions, and functions or code can be applied to each component of the array. Numpy arrays are important in the world of bioinformatics, as large biological datasets are often stored in numpy arrays. For more information visit: https://numpy.org/

pandas: Pandas is a widely used Python package that makes data analysis and manipulation easy. The defining feature of pandas is its DataFrame object, which allows data scientists to easily organize and read massive datasets. In a pandas DataFrame, one can add info to rows or columns, remove info, label columns and rows, and much more. Many different file types can be used to generate a pandas dataframe, including CSV, text, and excel files. Data sets can also be seamlessly merged or joined together. For more information visit: https://pandas.pydata.org/about/index.html

Logomaker: LogoMaker is a free online Python package that can be used to create beautiful sequence logos easily. To use it, simply insert the protein or nucleotide sequence you wish to generate a sequence logo for as a string into the logomaker function, and set optional parameters and arguments. The result is a clear, neat sequence logo that can give you a lot of information about the relatedness of two or more sequences. In this project, 3 sequences from 3 different species were compared. For more information visit: https://logomaker.readthedocs.io/en/latest/

NGLview: NGLview is a Jupyter widget used to visualize the 3D structure of virtually any protein. NGLview does not predict tertiary protein structure from a polypeptide sequence. Instead, it takes a file that already has 3D structure information in it and displays it in a jupyter notebook. The required input file type is .PDB, and these file types were obtained from the C-I-TASSER online tool for this project. .PDB files can also be found on the PDB website. For more information visit: https://github.com/nglviewer/nglview

In [16]:
```
from Bio import pairwise2
from Bio import SeqIO

import numpy as np
import pandas as pd
import logomaker
import nglview
import matplotlib.pyplot as plt
```

Note: If you are using the pip method, this following line of code must be carried out in its own cell, and you may need to restart the kernel to use it once it finishes installing

In [17]:
```
pip install logomaker
```

```
Requirement already satisfied: logomaker in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (0.8)
Requirement already satisfied: matplotlib in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from logomaker) (3.4.3)
Requirement already satisfied: pandas in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from logomaker) (1.3.4)
Requirement already satisfied: numpy in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from logomaker) (1.20.3)
Requirement already satisfied: pyparsing>=2.2.1 in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from matplotlib->logomaker) (3.0.4)
Requirement already satisfied: pillow>=6.2.0 in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from matplotlib->logomaker) (8.4.0)
Requirement already satisfied: python-dateutil>=2.7 in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from matplotlib->logomaker) (2.8.2)
Requirement already satisfied: cycler>=0.10 in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from matplotlib->logomaker) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from matplotlib->logomaker) (1.3.1)
Requirement already satisfied: six in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from cycler>=0.10->matplotlib->logomaker) (1.16.0)
Requirement already satisfied: pytz>=2017.3 in /Users/zachturman/opt/anaconda3/lib/python3.9/site-packages (from pandas->logomaker) (2021.3)
Note: you may need to restart the kernel to use updated packages.
```

Pairwise Sequence Alignment:

Pairwise sequence alignment is a bioinformatics method used to compare two and only two protein or nucleotide sequences, hence the "pair" in "pairwise". There is also something called multiple sequence alignments where more than 2 sequences are compared, but those will not be used in this study. There are two types of pairwise sequence alignments: global and local alignments. A global pairwise alignment algorithm, like the one used in this project, compares the 2 sequences to find the most matches, whereas a local alignment algorithm aims to find the biggest chunk where the sequecnes match perfectly. Each possible alignment in a global alignment is given a score based on how many individual "matches" there are between nucleotides, and the alignment with the highest score is ultimately the one we want to use. The data types used in the pairwise sequence alignment of this project are fasta files that contain the nucleotide sequences of human and loxodonta (African Elephant) DNA-binding regions of p53. A global alignment is performed, and the results are displayed below.

In [18]:
```
#https://biopython.org/docs/1.75/api/Bio.pairwise2.html
#http://biopython.org/DIST/docs/tutorial/Tutorial.html#sec99
#https://www.ncbi.nlm.nih.gov/nuccore/NM_000546.6
#https://www.ncbi.nlm.nih.gov/nuccore/XM_003416902.3

#read in the nucleotide sequences of the p53 gene that are involved in DNA-binding from fasta files from NCBI Nucleotide database
#Both elephantseq1 and humanseq2 are global variables that contain the nucleotide sequences for Loxodonta and Human p53 DNA-binding domain, respectively.
elephantseq1 = SeqIO.read("loxodontap53_nucleotide_DNAbinding.fasta", "fasta")
humanseq2 = SeqIO.read("humanp53_nucleotide_DNAbinding.fasta", "fasta")
#set up the gobal variable 'alignment' using appropriate functions and methods from pairwise2 to
# generate the pairwise sequence alignments
alignments = pairwise2.align.globalxx(elephantseq1.seq, humanseq2.seq)
#reveal the completed pairwise sequence alignments
print(pairwise2.format_alignment(*alignments[0]))
```

```
CACCA-TG---G--CCATC-TACAA--GAAGTCAGAGC-
  |  ||   |  ||  |  | |    |  |   |||
---C-GTGTTTGTGCC-T-GT-C--CTG--G---GAG-A
  Score=14
```

Sequence Logos

Sequence logos are a popular way to visualize the similarity and differences between two or more nucleotide or peptide chains. For sequence logos to work, the sequences you are comparing must all be the same length. For this project specifically, sequence logos operate by displaying the proportions of which amino acids are found at a given site. For example, all 3 of the polypeptide chains that are compared in this study have phenylalanine (F) at their C terminus. Therfore, the sequence logo displays a large "F" as the first character. Looking at the second amino acid in the chains, we see some differences. The second amino acid in human and elephant p53 DNA-binding domains is Arginine (R), but in mice it's Histidine (H). This discrepancy is shown on the sequence logo, where a large R appears with a small H under it. This shows us that at this location along the amino acid chain, most of the sequences have an Arginine here, but at least one sequence has Histidine. Repeat this for the rest of the chain and you get a beautifully crafted sequence logo that you can look at and instantly know where the peptide chains are different and where they are the same.

In [19]:
```
#https://logomaker.readthedocs.io/en/latest/
```

In [20]:
```
#I was not able to find a way to read the amino acid sequences into a jupyter notebook for use with LogoMaker,
#so I unfortunately had to hard code them into my notebook.

#The 3 protein sequences, which represnt the DNA-binding domains of human, elephant, and mouse
#p53 protein have been pasted into the global 'proteins' variable.
proteins = ["FRLGFLHSGTAKSVTCTYSPALNKMFCQLAKTCPVQLWVDSTPPCTRVRAMAIYKQSQHMTEVVRRCPHHERCSDSDGLAPPQHLIRVEGNLRVEYLDDRNTFRHSVVVPYEPPEVGSDCTTIHYNYMCNSSCMGGMNRRPILTIITLEDSSGNLLGRNSFEVRVCACPGRDRRTEEEN", "FRLGFLHSGTAKSVTCTYSPDLNKLFCQLAKT
#LogoMaker package requires the sequences to be in pandas database in order to generate a sequence logo.
#"proteindf" is a global variable that contains the dataframe of the protein sequences that will be used to
# generate a sequence logo.
proteindf = pd.DataFrame(proteins)

#I added column and index names to make the dataframe more clear, then printed some of the dataframe.
proteindf.columns =['Peptide Sequences']
proteindf.index = ['Human', 'Elephant', 'Mouse']
print(proteindf.head())

#crp_sites_list is a global variable that accesses the 'Peptide Sequences' column of the protein dataframe.
crp_sites_list = proteindf['Peptide Sequences'].values

#Now we will generate the sequence logo using logomaker functions and our crp_sites_list global variable to access the
# protein sequences in the proteins dataframe.
crp_counts_df2 = logomaker.alignment_to_matrix(sequences=crp_sites_list, to_type='counts')

#build the actual sequence logo in the logo global variable, selecting which dataframe the sequence logo is built from,
# as well as the font and shade of the logo and the size of the actual sequence logo.
#The sequence logo automatically displays in the Jupyter notebook after this code.
logo = logomaker.Logo(df=crp_counts_df2,
                      font_name='Arial',
                      fade_below=0.5,
                      shade_below=0.5,
                      figsize=(800,20))


#result is from C to N terminus
```

```
                                  Peptide Sequences
Human     FRLGFLHSGTAKSVTCTYSPALNKMFCQLAKTCPVQLWVDSTPPPG...
Elephant  FRLGFLHSGTAKSVTCTYSPDLNKLFCQLAKTCPVQLWVASPPPPG...
Mouse     FHLGFLQSGTAKSVNCTYSPPLNKLFCQLAKTCPVQLWVSATPPAG...
```

In [21]:
```
#code check to make sure human, elephant, and mouse peptide chains are the same length.
print(len(proteins[0]))
print(len(proteins[1]))
print(len(proteins[2]))
```

```
180
180
180
```

Structural Bioinformatics

Structural bioinformatics is a branch of bioinformatics that deals with the 3D structure of proteins. By visualizing 3D protein structure, scientists can answer many questions such as how this protein might interact with other proteins, and what particular active sites look like on the protein, which could determine drug development. Another major part of structural bioinformatics is the scientific community's recently developed ability to predict 3D protein structure with computer models and artificial intelligence. This new breakthrough is truly revolutionary. Before computers such as AlphaFold and C-I-TASSER were able to predict 3D protein structure based on the peptide sequence alone, it would take scientists countless hours to experimentally determine 3D protein structure. Structure-determining methods include X-ray crystallography and NMR, neither of which were easy or quick. For this project, a FASTA file containing the peptide sequences of the DNA-binding domain of human and elephant p53 were fed into the C-I-TASSER server, and the resulting structure was returned as a .PDB file for use in this project.

In [22]:
```
#https://github.com/nglviewer/nglview
#http://nglviewer.org/nglview/release/v0.5.1/#usage

#The global variable view1 accesses the correct PDB file in my directory for human p53 DNA-binding domain.
view1 = nglview.show_file("humanp53_DNA_binding_structure.pdb")
#says we want the 3D protein structure to appear in the "cartoon" style, as shown below.
view1.add_representation('cartoon')
#"licorice" means the molecular structure representation. In this case, we apply that visual to every
# Alanine and Glutamine in the protein.
view1.add_licorice('ALA, GLU')
#to view the final result
view1
```

In [23]:
```
#The global variable view2 accesses the correct PDB file in my directory for the loxodonta p53 DNA-binding domain.
view2 = nglview.show_file("loxodontap53_DNAbinding_structure.pdb")
#says we want the 3D protein structure to appear in the "cartoon" style, as shown below.
view2.add_representation('cartoon')
#"licorice" means the molecular structure representation. In this case, we apply that visual to every
# Alanine and Glutamine in the protein.
view2.add_licorice('ALA, GLU')
#to view the final result
view2
```

In [24]:
```
#the 'Box' function will allow us to put the two protein structures we want to compare
#into a 'box' side-by-side for easy comparison.
from ipywidgets.widgets import Box

#set w1 and w2 as the two global variables that hold the protein structures we want to compare.
w1=view1
w2=view2
#build and display the actual box.
Box(children=(w1,w2))
```

Analysis of Results

All of the data analysis methods used in this project deal with the DNA-binding region of the p53 gene or protein in either humans or African Elephants (Loxodonta). The global pairwise sequence alignment compares the nucleotide sequences of the p53 gene in humans and African elephants. From the results, we can see that the best alignment received a score of 14, which is quite high considering there are only 24 nucleotides in this region that we are analyzing. Visually, we can see that there is a high degree of sequence similarity at the beginning of these genes, but they begin to differ towards the end. Perhaps the promoter regions of the p53 gene are conserved across humans and loxodonta?

Moving on to the sequence logo comparing the DNA-binding domain protein sequences of human, mouse, and elephant p53, we see a similar pattern. At the C terminus, as shown on the left side of the sequence logo visual, the protein sequences show a high degree of similarity, often containing the same amino acid at the exact same location as their different-species counterparts. This similarity begins to fade towards the N terminus of the amino acid chains, indicating that the N terminus is less conserved across species. Perhaps the uniqueness of the N terminus is what gives loxodonta their unique cancer-fighting abilities.

Finally, examining the 3D protein structure of the DNA-binding domain of human and loxodonta p53 can show us just how similar their tertiary structuers are, even though the peptide sequences can vary greatly at certain locations. Of course, the side by side comparison only shows us the DNA-binding domain of each species' p53 protein, but we can still draw an important conclusion from the data: The DNA-binding domain of human and loxodonta p53 have very similar structure, therefore they have very similar functions. Unfortunately this study was not able to conclude that African elephants have a remarkable p53 structure, therefore there must be something else that contributes to African elephants' remarkably low rates of cancer. Perhaps other proteins that interact with p53 like MDM2 or p21 have a unique peptide sequence or 3D structure in elephants that grants them the ability to avoid cancer. There are countless possible avenues for further exploration and research, and the results may be used to take cancer treatment to the next level.

In [ ]:
```
```