JOURNAL OF
EMERGING INVESTIGATORS

Article

String analysis of exon 10 of the CFTR gene and the use of Bioinformatics in determination of the most accurate DNA indicator for CF prediction

Peyton Carroll[1*], Samhith Kakarla[1*], Jason Carroll[1]
[1]Fremont High School, Sunnyvale, CA, *equal contribution

SUMMARY

Cystic Fibrosis (abbreviated as CF) is a deadly disease with no cure that affects over 70,000 people worldwide every year. In this study, we aimed to discover an efficient way of diagnosing the disease and hence wanted to figure out the best predictor of a newborn's predisposal to developing CF by comparing different nucleotide patterns in the CF transmembrane conductance regulator (abbreviated as CFTR) gene. We compared nucleotide sequences from CF patients to healthy humans. We then ran string analyses over each nucleotide sequence, looking for pre-determined patterns. We then compared the patterns we observed in the diseased patients and healthy patients. The pattern that showed the most discrepancy from the diseased gene to the healthy gene was noted as the best predictor of someone's predisposition to CF. In this experiment, we focused on sequences two to eight bases long. We hypothesized the eight-base long sequence "GGGGGGGG" would be the best predictor due to it being that of the greatest length, therefore making the respective nucleotide sequences longer. Furthermore, G is the least common base, so we reasoned that this sequence will be the least common sequence. Because this sequence is least common, a single occurrence of it would be statistically significant. This differs from the current research, as the current research focuses on analyzing point mutations, and not the whole exon. By focusing on the whole exon, we propose a more accurate determination technique because more DNA is being analyzed. We can carry this work forward by using the same code and scientific process on other exons. Overall, we found "TTCCACAG" occurs 9.13 times more in the healthy nucleotide sequence than the mutant nucleotide sequence. Hence, if we can compare the DNA of a newborn to their parents' and the DNA string occurs "TTCCACAG" more often, we can conclude the child may have increased risk of developing CF.

INTRODUCTION

Cystic Fibrosis (abbreviated as CF) is a disease which causes secretion of a "sticky mucus," which can harm the function of the lungs and pancreas (1). Named after the fluid-filled sacs called cysts and scar tissue called fibrosis that it causes, this disease usually starts in childhood (75% of patients are two years old or younger), with 30,000 people living in the United States with the disease (1). Additionally, 1000 people annually are diagnosed. CF has many complications, and can lead to liver disease, lung infections, digestive problems, diabetes, and male infertility (2). Currently, there is no cure for CF. CF is a genetic disease, and researchers have found links to the CF transmembrane conductance regulator (CFTR) gene, found on chromosome 7 (2).

The CFTR gene codes for the construction of the CFTR protein (1). When the CFTR protein is mutated, it becomes unable to move chloride ions out of cells (1). This chloride normally attracts water, but the absence of it causes mucus in organs to become "thick and sticky" (2). Mucus is naturally supposed to be slippery (1). This mucus causes several problems. It clogs the digestive system, depriving the patient of nutrients (2). It also entraps bacteria in the patient's body, leading to infections (1). CF is most prevalent in Caucasians, and is currently the most common lethal inherited disease among Caucasians (2). In order to contract CF, both parents of the patient must give the patient a mutant copy of the CFTR gene (3). Just one copy of the gene is not sufficient for the occurrence of CF, and the person will be a carrier of the disease (3). An estimated 10 million people in the United States are CF carriers (3).

The CFTR protein is an ATP-binding cassette protein with two transmembrane domains (TMDs) (3). These TMDs combine to form a nucleotide-binding domain (NBD) (3). CFTR also contains a regulatory domain. This domain connects two objects called pseudo-symmetrical halves (3). These halves are constructed of a TMD and NBD (3). The CFTR protein works similarly to others in the ABC transporter family; it extracts the energy of ATP hydrolysis, allowing it to pump materials against the gradient. Of the ABC transporters family, two groups emerge. Firstly, ABC importers import materials into the cell. These importers are found solely in prokaryotic cells (4). ABC exporters, the other type of ABC transporter, are found in all types of cells. Our study focused on the CFTR protein, an ABC exporter that acts similarly to a straw used in soda cans. The CFTR protein is constructed of 1,480 amino acids, and is folded in the shape of a straw; allowing certain materials to flow from the inside to outside of the cell (5). The structure of the CFTR protein is still largely unknown to scientists, as the first high resolution picture of the protein was taken in 2017 (6).

The CFTR gene, first discovered in 1989 by Lap-Chee Tsui, is 189 kb long and contains 27 exons. The CFTR gene is located at 7q31.2, and is located between nucleotides 117,480,025 and 117,668,665 in Homo sapiens. We decided to focus on a particular exon due to limitations of computing power. We decided to analyze exon 10 because four of the most common mutations in the CFTR gene occur within exon 10 (12). We reasoned this exon would therefore contain the most discrepancies compared to healthy genes (6).

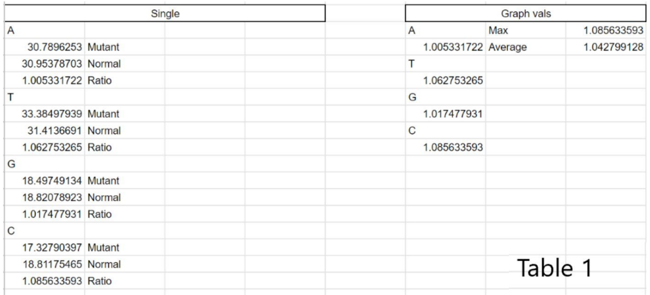
When looking for nucleotide patterns in exon 10, we used the computer language Java in conjugation with the IDE (Integrated Development Environment), Intellij. To test for patterns, we found a database of both healthy exon 10 nucleotide sequences and CF nucleotide sequences. We converted these nucleotide sequences into text files to run our program on, and parsed the files, transforming the text files into strings. We then ran our program over each nucleotide sequence, searching for patterns. The patterns we looked for were chunks of DNA, 2 to 8 bases long. The reason we did not look for chunks with lengths 9 bases or larger was because of the limit of computing power. There are four different nucleotides: adenine, thymine, guanine, and cytosine. Therefore, the number of different possible DNA chunks we are searching for is equal to four to the power of the length of the chunk. Once we found the frequency of these various chunks of DNA, we found the frequency of those chunks in terms of the number of occurrences over 100 nucleotide bases. We then compared the frequency of these chunks in patient DNA and the healthy DNA. The large differences in frequency were noted. The longer the length of nucleotides we used, the more unique it was and therefore it was less likely to appear. Because of this, we hypothesized the nucleotide sequence that was eight nucleotides long would be the best predictor of CF in the DNA.

This question is of interest due to the long-term damage incurred by CF. CF becomes increasingly more deadly and debilitating the longer the disease goes undiagnosed. Creating an accurate DNA testing for newborns will allow us to predict CF in the future, allowing doctors and parents to monitor predisposed children for symptoms.

We concluded the “TTCCACAG” sequence was the best indicator of CF, as it occurred 6.33 times out of 1,000,000 bases for the mutant DNA and 57.78 out of the 1,000,000 bases for normal DNA. This was the most clear and obvious ratio from the lab.

RESULTS

We examined exon 10 of the CFTR gene to determine the best genetic indicator of CF to provide the best diagnosis. The goal of our project was to discover the best indicator of CF in exon 10 of the CFTR gene. We looked for the greatest difference in frequency of the different DNA sequences. We acquired a database of exon 10 genes from 8,000 healthy individuals and 3,000 CF patients. This step took the most amount of time due to the fact that we needed to find a large database of DNA to ensure that our data was accurate. We saved all of these DNA sequences into a text file, which we then analyzed, looking for patterns. We then displayed the frequencies of the sequences per 100 bases for normal and mutant DNA, along with the ratio between them (**Table 1**).



Single		Graph vals		
A		A	Max	1.085633593
30.7896253	Mutant	1.005331722	Average	1.042799128
30.95378703	Normal	T		
1.005331722	Ratio	1.062753265		
T		G		
33.38497939	Mutant	1.017477931		
31.4136691	Normal	C		
1.062753265	Ratio	1.085633593		
G				
18.49749134	Mutant			
18.82078923	Normal			
1.017477931	Ratio			
C				
17.32790397	Mutant			
18.81175465	Normal			
1.085633593	Ratio			

Table 1

Table 1. Shows the ratio between normal and mutant DNA for the occurrences of all possible nucleotide sequences of base length 1 in the normal and mutant DNA data. It also shows their max and average ratios.

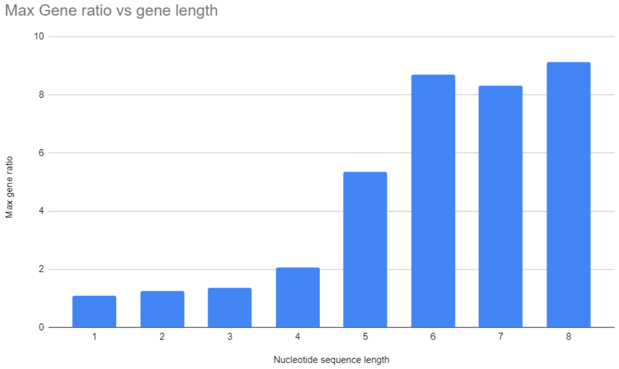


Figure 1. Shows the largest ratios from each nucleotide sequence from length 1 - 8.

We displayed the DNA sequences which were length two, three, four, five, six, seven, and eight in the same manner. We then graphed the ratios of the sequences (determined by frequency of normal DNA divided by frequency of mutant DNA) for each set as can be seen in example in **Table 1**. We then used a sheet formula to find the largest ratio for each sequence length (**Figure 1**).

In this data, we saw a clear exponential relationship. As we increased the sequence length, we observed an increase in the ratio. Note that sequence 7 did not follow this pattern, but 1-6 and 8 followed the pattern. We concluded that the 8-base long sequence “TTCCACAG” had the greatest ratio, occurring 9.13 times more in the normal DNA than mutant DNA. Because this ratio was the greatest, “TTCCACAG” was the best indicator, since a patient lacking this sequence, is more likely to have CF.

DISCUSSION

Our sequence of interest, “TTCCACAG” appeared 9 times more often in normal DNA compared to mutant DNA. This was by far the greatest difference in the incidence of any nucleotide sequence between normal and mutant DNA hence becoming the best indicator of the disease. We interpret this result to mean that if an individual’s DNA contains the nucleotide sequence “TTCCACAG”, the individual could be at a higher risk for CF. We think comparing a patient’s DNA to a healthy immediate family member’s DNA is the best way to use our method of detection as the patient and the family member should have very similar DNA and hence if such a large difference in their DNA is detected it would be safe to assume the patient has CF (13).

This result has large implications in the diagnosis of CF. Because there is no cure for CF, the quality of life of the patient depends on how quickly the disease can be diagnosed. With the tool we have developed, we can test the DNA of a newborn and determine if they will develop CF. This will help doctors make an advanced diagnosis on the patient.

One remaining question is whether there are other patterns which yield a greater ratio and are longer than eight bases. This question was asked because the longest sequence we tested was eight bases. In the future, we plan to experiment with RNA to see if the patterns we observed in DNA occur again. One factor that has heavily influenced our results is the size of our data. We had about 8,000 sequences of normal DNA and a bit over 3,000 sequences of mutant DNA. This, although being enough to provide us with somewhat reliable results, isn't a very large amount of data. To better detect patterns between normal and mutant DNA it would be best to take DNA samples from hundreds of thousands of volunteers and build a database from those sequences of DNA and then analyze these DNA samples to find results. We didn't encounter any traditional experimental errors or biases as this wasn't an experiment done physically with any sort of equipment but rather done all on a computer. However, it is possible for errors to have occurred with our code that could have affected our results. We also could have collected incorrect data which would have led to inaccurate results.

We plan to use the software we have developed to determine the occurrence of other genetic-based diseases such as: sickle-cell anemia, Marfan syndrome, Duchenne muscular dystrophy, and Huntington disease (12). Currently, scientists are working to find specific mutations in exon 10 of the CFTR gene. Some examples are: Arg117His, Arg334Trp and Ala455Glu which are mutations of specific sequences. However, we are looking for patterns overall in the whole of exon 10, a more holistic approach instead of just looking at a section of the DNA. We also plan to use patient samples to test if our code can find the predicted sequence.

MATERIALS AND METHODS

In this experiment, we used a dataset consisting of nucleotide sequences of exon 10 of the CFTR gene. We obtained these sequences from The National Center for Biotechnology Information and CF Mutation Database (10, 11). These nucleotide sequences were converted into a text file. This text file was added to an idea project in Intellij. We then wrote a program to read these text files. We used the java utility *scanner* to read these files. We then used a *for* loop to loop over all the nucleotides. The patterns we were looking for were pre-programmed into the program. In addition, we created a counter which would increase by one every time we saw a pattern. We divided the counter by the nucleotide sequence length to determine the frequency of that pattern.

	Starting with A	Starting with T	Starting with G	Starting with C
Ending with A	AA	TA	GA	CA
Ending with T	AT	TT	GT	CT
Ending with G	AG	TG	GG	CG
Ending with C	AC	TC	GC	CC

Table 2. The ratio between normal and mutant DNA for the occurrences of all possible nucleotide sequences of base length two in the normal and mutant DNA data. It also shows their max and average ratios (we only chose to display tables for base length one and two because the rest of them are very long and it would be impractical to display them. We still included the graphs for their ratios).

The patterns we were searching for were nucleotide base sequences two to five bases long. In each of the sequences, we tried every possible sequence of bases. For example, in the two base-long sequence, we tested 16 different nucleotide sequences (**Table 2**).

We searched for the nucleotides in **Table 2**. We then recorded the frequency of all the nucleotide sequences we were looking for in a table.

The code we used to analyze the CFTR gene had several parts. Firstly, we needed to design code to read a text file of the nucleotide sequences. This was done by using the *Scanner* class in java. We iterated using a *while* loop. We created an empty string to hold to all these values. This string had no immediate value to us though, as we needed two separate array lists to hold the mutant DNA and normal DNA. We used the *string split* method to split the string at all spaces, as the format for the DNA nucleotide sequence is "sequence, m" for mutants of "sequence, n" for normal. Because the *split* command returns an array, we generated an array filled with the nucleotide sequences as strings. Because we split the large string at spaces, we needed to take into account that the previous nucleotide sequence was indicated either mutant or normal by the string after it. For example, in the case "atgc, mcgta, n", we see the atgc nucleotide sequence is mutant indicated by m and the cgta is normal as indicated by the n. If we split this by the space, we get two strings, "atgc", "cgta". M in front of cgta indicates the string in the index before cgta is mutant. It does not indicate cgta is mutant. This was taken into account for our code. The array we created from the *split* command of the large string was looped over, and the values in the array were separated into two different arrayLists. These arraylists contained the nucleotide sequence for the mutant DNA in one list and normal DNA in the other list. We decided to split the DNA into two lists so that we could have an easier analysis of the different types of DNA. We used an arrayList because we did not know how many nucleotide sequences of normal and mutant DNA we had, so we needed a data structure with flexible bounds in order to account for the unknown amount of data. We used a method to separate the values in the large array into the two arrayLists by using the *startswith* method. This method takes a string as an input and returns the value which it starts with. With this knowledge, we used an if statement to organize the data. If the string at a certain index returns n from the *startswith* method, then the index before that string is normal. The same logic can be used for m and mutated. With the knowledge of which strings are mutant and normal, we used the *.add* command to add these values to either the normal nucleotide sequence arrayList or the mutant nucleotide sequence arrayList. Now that we have two separate arrayLists with the mutant and normal nucleotide sequences, we created a method to search for patterns. This method takes two values as input; an arrayList and a string for which we are looking for. We looped over every index to look for the pattern, and looped through the string at that index to find the pattern. We initialized two counters with values of 0; a counter for the total number of nucleotide bases and the number of occurrences of the pattern. We increased the value of the nucleotide bases counter by one every time we looped over a nucleotide base, and added one to the pattern counter every time we encountered the pattern. Once the loop finished looking for the pattern, we created a double and equated it to the value of the total number of patterns by the total nucleotides. We multiplied this number by 100 to find the total number of occurrences every 100 bases, and returned this value.

Received: January 06, 2020
Accepted: June 15, 2020
Published: July 12, 2020

REFERENCES

1. McIntosh, James. “Cystic Fibrosis: Symptoms, Causes, and Management.” *Medical News Today*, MediLexicon International, 11 Jan. 2018, https://www.medicalnewstoday.com/articles/147960.php.

2. “About Cystic Fibrosis.” *CF Foundation*, www.cff.org/Whatis-CF/About-Cystic-Fibrosis/. Accessed: 10 Dec, 2019.

3. “Basics of the CFTR Protein.” *CF Foundation*, www.cff.org/Research/Research-Into-the-Disease/RestoreCFTR-Function/Basics-of-the-CFTR-Protein/. Accessed: 21 Dec, 2019.

4. “Cystic Fibrosis.” *ERS*, https://www.erswhitebook.org/chapters/cystic-fibrosis/. Accessed: 2 Jan, 2020.

5. “CYSTIC FIBROSIS TRANSMEMBRANE CONDUCTANCE REGULATOR; CFTR.” *Online Mendelian Inheritance in Man (OMIM)*, www.omim.org/entry/602421. Accessed: 17 Dec, 2019.

6. "CFTR". *UniProt. Swiss Institute of Bioinformatics*, https://www.uniprot.org/uniprot/Q9UML8, Accessed: 11 Dec. 2019

7. Bhargava, Hansa D. “Cystic Fibrosis (CF): Symptoms, Causes, Diagnosis and Treatment.” *WebMD*, https://www.webmd.com/children/what-is-cystic-fibrosis#1. Accessed: 12 Dec, 2019.

8. Hwang, Tzyh-Chang, and Kevin L Kirk. “The CFTR Ion Channel: Gating, Regulation, and Anion Permeation.” *Cold Spring Harbor Perspectives in Medicine*, Cold Spring Harbor Laboratory Press, 1 Jan. 2013

9. "Cystic Fibrosis Transmembrane Conductance Regulator (CFTR) Gene". *The Embryo Project Encyclopedia*, https://embryo.asu.edu/pages/cystic-fibrosis-transmembrane conductance-regulator-CFTR-gene. Accessed: 30 Dec, 2019.

10. “Human Cystic Fibrosis (CFTR) Gene”. *National Center for Biotechnology Information, U.S*. National Library of Medicine, www.ncbi.nlm.nih.gov/nuccore/?term=CFTR+human.

11. *Cystic Fibrosis Mutation Database*, http://www.genet.sickkids.on.ca/CFTRdnasequence/CFTRdnasequence0_180000.txt?endPoint=180000&startPoint=0&fbclid=IwAR0BxAZR1JW4v4R852NXh_ryi6J6cF2h2CoOd3vEQVuYZLzVQQGvuk994jw. Accessed: 1 Jan, 2020.

12. “Genetic Disease: 4 Types and List of 39.” *EMedicineHealth*, www.emedicinehealth.com/types_and_list_of_genetic_diseases/article_em.htm. Accessed: 25 Dec, 2019.

13. Belov, Artem. “Do Siblings Have The Same DNA: How Your Parents Shaped You.” *Atlas Blog*, atlasbiomed.com/blog/do-siblings-have-the-same-dna-thefacts-on-family-genetics/. Accessed: 3 Jan, 2019.