Final: BLAST 2 Sequences

AS.410.712.81.SP21 Advanced Practical Computer Concepts for Bioinformatics

Gregory Sprenger

BLAST, or Basic Local Alignment Search Tool, has been around for decades and has been used by many scientists and students. It identifies similarities between nucleotide or protein sequences against other sequences or a database. Once matches are found, it calculates the statistical significance of each match. This allows for inferences to be made about the sequences, such as functionality and relationships. Furthermore, not every BLAST search needs to be matched with an entire database. For instance, if two sequences are already known to be homologous, then comparing just these two sequences would return a much quicker run time. Therefore, due to the familiarity with BLAST and the quick execution of its command-line tool bl2seq, it was chosen to be the back-end tool.

To determine the validity of the results obtained from the bl2seq tool, two lactoferrin sequences were used, DQ821936.1 and AJ005203.1. To test the matching abilities of bl2seq, DQ821936.1 was compared against itself. This returned a one-hundred percent identity. To further test bl2seq, DQ821936.1 and AJ005203.1 were compared against each other using the nucleotide search tool, blastn. The percent identity should be high as they are both of a lactoferrin gene. For the plus strands, it returned 55/57 identities which is a ninety-six percent match (figure 2). It also matched the plus and minus of one-hundred percent. BLAST’s webserver was used to validate these results, in which it returned the same identities matched and overall match percentage (figure 1). Though, it did not display matches on the plus/minus strand. Therefore, the use of bl2seq on the back-end of this implementation proved to offer valid results.

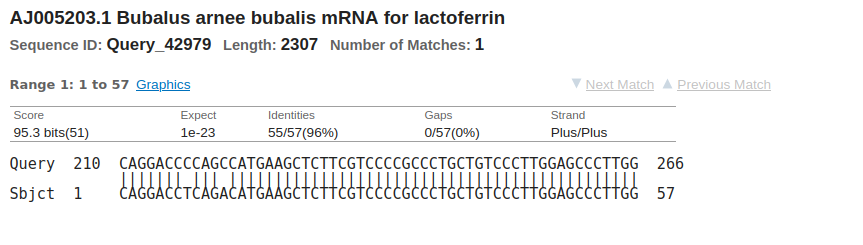


Figure 1: BLAST’s webserver blastn comparison of lactoferrin genes DQ821936.1 and AJ005203.1

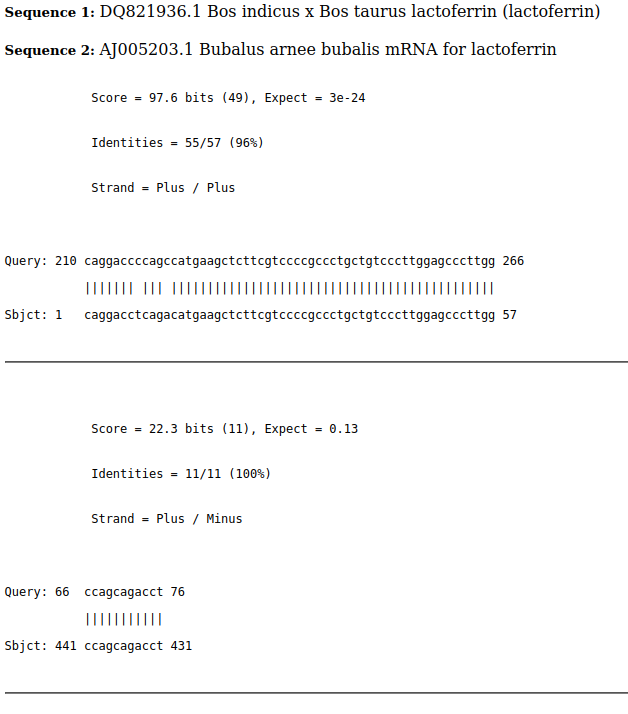


Figure 2: bl2seq results of a blastn comparison of lactoferrin genes DQ821936.1 and AJ005203.1

When working on a server with minimum permissions, it makes the implementation a front-end for a tool difficult. The biggest obstacle was trying to create and write to files after obtaining the data from a form in the index.html file. It was originally planned to create new temporary files for each submission, with python’s mkstemp, so that they could be erased at the end of their analysis. Instead, separate files were created and given read-write-execute permissions so that they could be rewritten during each analysis. Future plans are to add a file upload button on a less restricted machine. Also, creating a jQuery script to alert the user if any text area was left blank proved to be harder than it actually was. After adding an id to the text area forms and checking them on submit for emptiness, it became much easier. Overall, this project allowed the skills learned over the course of the class to be applied and rigorously tested.