CS-415
Assignment 1

Colby Blair

February 26th, 2011

1 Robbery

Top Results

Accession	Description	Max score	Total score	Query coverage	E value
AAD44164.1	cytochrome b [Elephas maximus indicus]	94.7	94.7	83%	$2e-17$
AAR06165.1	cytochrome b [Loxodonta cyclotis]	91.3	91.3	85%	$2e-16$
AAR06156.1	cytochrome b [Loxodonta cyclotis]	91.3	91.3	85%	$2e-16$

The creature with the highest **Total Score** is the Elephas maximus indicus, or the Indian Elephant, and is the most likely culprit to the robbery. The **Max Score** was the same as the total score. The **Query Coverage** is reasonably good and about the max coverage for all candidates. The **E value** (or expected score) is extremely low compared to the total score, so there seems to be a pretty clear indication that we have the culprit.

One observation was that the E value, although low in value, was relatively high compared to other searches. Perhaps this protien may actually be fairly common, but this is only my hypothesis in my extremely low experience with sequence alignments.

2 Is Frequency Important?

2.1 First Protein

The first protein most likely belongs to a Mountain Degu (Octodontomys gliroides). It is a species of rodent found in Argentian, Bolivia and Chile. It could also belong to a Talas Tuco-tuco (Ctenomys talarum), which is another rodent found in Argentina.

Top Results

Accession	Description	Max score	Total score	Query coverage	E value
Q94WW5.1	[Octodontomys gliroides]	234	234	98%	2e-59
AAC52545.1	[Trinomys sp. MN-31459]	233	233	100%	$4e-59$
Q94WX2.1	[Ctenomys talarum]	232	232	100%	$7e-59$

2.2 Second Protein

The second animal is most likely a North American porcupine (Erethizon dorsatum). It could also potential be a Bicolored-spined Porcupine (Coendou bicolor), found in Bolivia, Colombia, Ecuador, and Peru. The animal could also be a Brazilian Agouti (Dasyprocta leporina), found in Venezuela, Guyana, French Guiana and Brazil.

Top Results

Accession	Description	Max score	Total score	Query coverage	E value
ACI89451.1	[Erethizon dorsatum]	161	161	96%	$1e-37$
AAC52559.1	[Coendou bicolor]	160	160	96%	$3e-37$
AAC52558.1	[Coendou bicolor]	160	160	96%	$3e-37$
AAQ04519.1	[Dasyprocta leporina]	158	158	96%	$2e-36$

2.3 Comparison with Frequency

The first protein is 90% decomposed with a random residue. The second protein is 90% decomposed with a random residue from within itself. This results in the second protein having a much larger **E value** (or expected value) than the first. This indicates that residue frequency is a significant factor in identification, and **frequency is somewhat proportion with E value**.

3 Only a Partial Print

3.1 First Protein

The first protein most likely belongs to a Bighorn sheep (Ovis canadensis) of North America. It also could belong to a Desert Bighorn Sheep (Ovis canadensis nelsoni) of Southwest desert regions of the United States and in the northern desert regions of Mexico. It can also belong to many other species of bighorn sheep. The **amount of candidates** with Total Scores comparible to the leader was significantly larger than our other search, suggesting an imperical idea that decreasing the length of the protein creates a lot more potential matches.

Top Results

Accession	Description	Max score	Total score	Query coverage	E value
ACB78005.1	[Ovis canadensis canadensis]	422	422	100%	$2e-116$
O78779.1	[Ovis canadensis mexicana]	421	421	100%	$2e-116$
ADA85741.1	[Ovis canadensis nelsoni]	421	421	100%	$3e-116$
ACB78006.1	[Ovis canadensis canadensis]	421	421	100%	$4e-116$

3.2 Second Protein

The second protein search resulted in many of the same candidates as the first search. The only significant results different from the first were that since the string/residue length of the second protein is even less than the first, the **E value** is even higher. This is expected. The total score is also less, since there are less characters/residues to match (in part).

Top Results

Accession	Description	Max score	Total score	Query coverage	E value
ACB78004.1	[Ovis canadensis canadensis]	119	119	100%	$2e-25$
ADA85741.1	[Ovis canadensis nelsoni]	119	119	100%	$2e-25$
O78779.1	[Ovis canadensis mexicana]	119	119	100%	$2e-25$
CAI30088.1	[Ovis nivicola]	119	119	100%	$2e-25$
CAI30087.1	[Ovis nivicola]	119	119	100%	$2e-25$
ACB78006.1	[Ovis canadensis canadensis]	118	118	100%	$2e-25$
ACB78005.1	[Ovis canadensis canadensis]	118	118	100%	$2e-25$

3.3 Comparison and Conclusion

Both proteins were substantially smaller in size than our other searches. This should increase potential matches in many organisms. This is why we see so many **more candidates with top Total Scores**. We also see **E values** (or expected scores) go up, since with a shorter protein, there are so many more expected matches. Also, it was noticable that **Query Coverage** was significantly higher.

4 Alignment

Running the code:

```
./align.py blosum62.txt v1.fasta v2.fasta
./align.py blosum62.txt v2.fasta v1.fasta
```

- produce the results:

```
...
Score: 7.0
-A-VIDVEGASL–ASFINGERLING–S-
RICHVI-VALASVEGASSINGERDEEDCITY
Score: 7.0
RICHVI-VALASVEGASSINGERDEEDCITY
-A-VIDVEGASL–ASFINGERLING–S-
```

So there is no difference (other than the strings are swapped in the output). There should be no reason why any switching ever would make a difference. A difference would indicate to the user that their BLOSSUM matrix was incorrect (duplicate pairs), or that gap calculations for e and s were unequal.

5 A Tie

In align.py, the code that checks the best score so far is:

```
(whichway, f[s][e]) = maxarg([
            f[s][e-1] + gap, # for east
            f[s-1][e] + gap, # for south
            f[s-1][e-1] + subst[substDict[southStr[s-1]]][substDict[ eastStr[e-1]]], #  for both
])
```

The result of **./align1.py blosum62.txt v5.fasta v6.fasta** is:

```
Score: 70.0
AACCCAAAACCCCCAAAAAA
A-CC-AAA-CCCC-AAAAA-
```

maxarg() checks from 1 to n arguments it gets looking for a max, in that order. In the case of align.py above: 1 checks East; 2 checks South; 3 checks Diagonal. maxarg() only makes a recognition of a new max, so in the case of a ties, **it recognizes the first max**.

To check 1 Diagonal, 2 East, 3 South, we simply need to change the order of the maxarg() args in the align2.py code:

```
(whichway, f[s][e]) = maxarg([
            f[s-1][e-1] + subst[substDict[southStr[s-1]]][substDict[ eastStr[e-1]]], #  for both
            f[s][e-1] + gap, # for east
            f[s-1][e] + gap, # for south
])
```

The result of **./align2.py blosum62.txt v5.fasta v6.fasta** is:

```
Score: 70.0
AACCCAAAACCCCCAAAAAA
-A-CC-AAA-CCCC-AAAAA
```

The result is no difference in **Score**, which is no suprise. The resulting gaps and matches, however, are quite different, which shows the change in how ties are handled due to the align2.py modifications listed above.

6 Local Alignment

The local alignment took some trial an error to get from the global alignment. We essentially zero out the borders, and then record a zero score if a location's east, south, and diagonal values have a maximum as a negative number. Also, if this is the case, the algorithm sets the direction matrix location to 'stop'.

When we align the fruit fly HOX gene with this human HOX gene, we get the following result: The result of **./alignlocal.py blosum62.txt drosophilaHOX.fasta humanHOX.fasta** is:

```
Score: 373.0

Score: 373.0

284 LYPWMRSQFGKCQERKRGRQTYTRYQTLELEKEFHFNRYLTRRRRIEIAHALCLT
    GSGG-EGDEITP
118 IYPWMRSS-GP–DRKRGRQTYTRYQTLELEKEFHFNRYLTRRRRIEIAHALCLTERC
    PTAAAAPEG-AV-P
real 0m4.816s
user 0m3.919s
sys 0m0.136s
```