Investigation of the Population Structure of Methylobacteriaceae

Studies have demonstrated that species of the Methylobacteriaceae can participate in a symbiotic relationship with certain land plants. However, although it is clear that this partnership can drastically affect the fecundity and crop yield of the plant and is essential for the bacteria, very little is known about the specificity of this relationship. In order to help infer the coevolutionary dynamics of this symbiosis, and to investigate the population structure of Methylobacter symbionts , this study genotyped a large number of Methylotrophic samples from 10 plant species in Woods Hole, MA. Unfortunately, sequencing errors meant that the analysis could not be completed before the end of the course. However, presented in this paper are key insights into the type of organisms that can be isolated on methanol plates from leaf surfaces, a method for the comparison of within species population genetic variation (using standard rRNA toolkits), and an examination of the extant variation in the rRNA sequences of the Methylobacterium genus.

* * * * * * * * * * * *

Key Points For Future Students

- Lab Strains of Methylobacterium are easily lysed by boiling; however for environmental samples the freeze-thaw method is vastly preferred.

- If you see fast growing, reasonably sized white colonies on the plates you are using to isolate aerobic Methylotrophs from leaves, there is a good chance they are members of the Moraxellaceae family, and are actually eating the cycloheximide. (These should show up before the pink colonies).

- Fast growing plants appear to have fast growing bacteria!!! I think this could be a cool and feasible future class project and was sorry I did not have time to investigate it more.

- If you are a computer programming type (or if you want me to send you a program), there is some code at the end of this report that can produce some very useful numbers for you. Namely, it will produce files that contain:

  1-The number and type of segregating sites for a group of aligned sequences.
  2-The Average Hamming Distance per base pair for a group of aligned FASTA files produced by ARB, GREENGENES or any other program. This is more useful then the ARB distance matrix where there are a number of comparisons that need to be run, or when you want actual numbers instead of evolutionary distances. The distance calculation method can of course be changed.
  3-The location in a sequence that has diversity within your alignment, and the measure of it.
  The time saved by this program will be lost if you copy it by hand, so send me an email at ndelaney@fas.harvard.edu, I should have this computer (and thus the code) for the next 5 years.

- For that matter, if you do decide to study methylotrophs for your project, feel free to contact me with any and all questions.

Description of the Project

Aerobic Methylotrophic bacteria are found pretty much anywhere there is air and methanol (in between your toes, the ocean, plants etc.). The genus *Methylobacterium* is particularly interesting because they form associations with plants that seemed to me rather intricate. In some plants they nodulate the roots, in others they grow deep into the leaf tissue. They have also been known to produce phytohormones and auxins which help the plants to grow. Therefore, I envisioned that this relationship might require some specificity between the two partners, and at the very least I was curious to learn a little bit about bacterial "species" level diversity. So I set out to ask if two plants, with their different look and feel were truly different to a methylotroph, or whether a leaf, was a leaf, was a leaf. I thought a good way to go about this was to survey the population.

To do this, I sampled from 10 plants around the MBL, listed in the table below.

Species Key	Common Name	Sci. Name	Family
A	Birdfoot Deervetch	Lotus corniculatus	Fabaceae
B	Japanese meadowsweet	Spiraea japonica	Rosaceae
C	Hemerocallis fulva	Orange Daylily	Liliaceae
D	Phragmites australis	Common Reed Species	Poaceae
E	Grass leaved Goldenrod Species	Euthamia graminifolia	Asteraceae
F	Lythrum salicaria	Purple Loosestrife	Lythraceae
G	Beach Pea	Lathyrus japonicus	Fabaceae
H	American Basswood	Tilia americana	Tiliaceae
I	Common Honeylocust	Gleditsia triacanthos	Leguminosae
J	Clover	Trifolium sp.	Fabaceae

For each of the species A-F I gathered 10 leaves from ten different plants, rubbed sterile sand paper on their underside and pressed them into an agar plate with the standard Microbial Diversity recipe given below. All plates were incubated at 30 C. Note that of the plants above, the Fabaceaes seemed to reliably provide colonies, while the

Honeylocust and the Basswood were the most unreliable. I only sampled two clover plants, and received the rest of my isolates from classmates who had enriched for them earlier in the course. The Beach Pea samples were acquired by a friendly classmate, who only obtained leaves from one plant (although I did sample from another plant later).

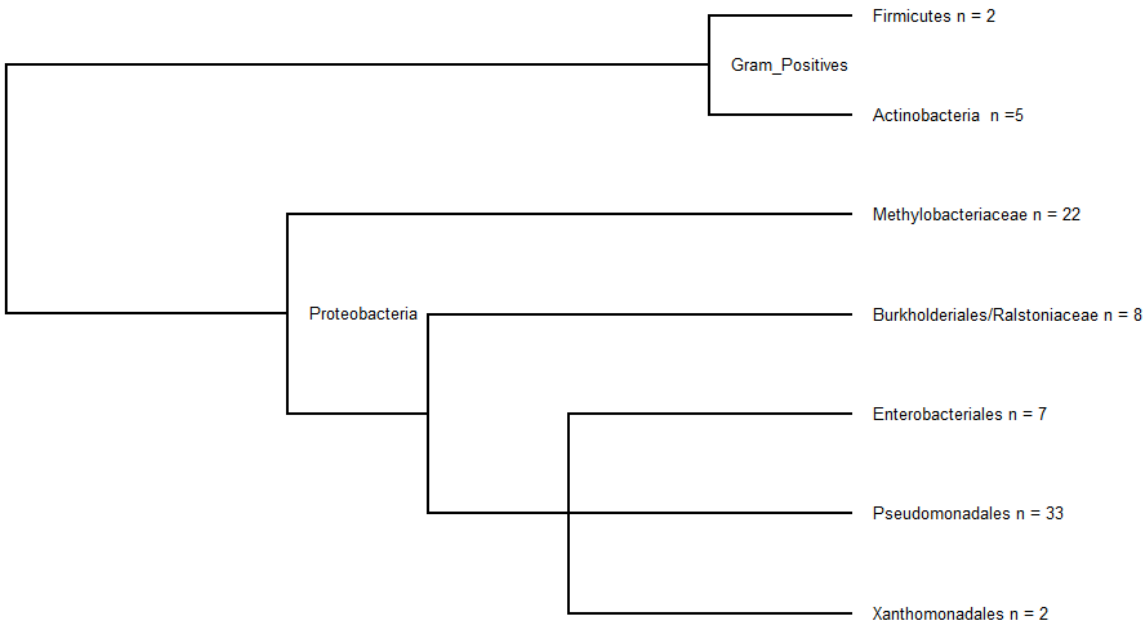
Methylotroph Media

- 10 ml 100x Fresh Water Base
- 10 ml .5 M NH4Cl
- 10 ml 150 mM K Phosphate Solution
- 1 ml 1 M Sodium Sulfate
- 5 ml 1 M MOPS Buffere, pH 7.2
- 1 ml Trace Elements
- 10 ml Vitamin Solution
- 15 g Washed Agar
- Add to 1 L milliQ Water

After Autoclaving add

- .1 ml Multivitamin Solution
- 125 ug Cycloheximide
- 2 ml Methanol

I wound up growing a rather large diversity of colonies types, which I then tried to sequence (using a combination of primers, 1492R, 519F and 8F, although due to cost constraints the 519F is overrepresented in the sample). Although I attempted many PCRs, most of the bacteria proved hard to amplify, and so only a few wound up in the final collection of sequences. Of these some I did not trust as not being the result of a false amplification (although the negative controls always were negative). An overview tree showing the types (and number of colonies) that were isolated is shown below.



A Brief Description of my thoughts on each of these major groups is included below. A more complete listing of each sequenced sample and its place in the tree of life is given at the end. This discussion should be prefaced by saying that the base calling software was very lax, and so I am still in the process of refining the sequences/alignments.

Members of the Order Burkholderiales

The colonies that generated these sequences were very diverse, some pink, some white, and some fuzzy, indicating that these sequences were the result of contamination. Discussion with some of my classmates indicated that they had amplified similar sequences. To test whether our sequences were similar, I built a BLAST database with 5 other classmates Burholderiales sequences, all results scanned showed that exact matches were obtained over the region of reliable sequence. As such, this data was removed from further consideration. Although there was some indication that one of the

isolates from *Hemerocallis fulva* might be unique, the likelihood of contamination forced these sequences aside.

Members of the Order Pseudomonadales, Actinomycetales, Xanthomonadales and Bacillales (With the exception of the Moraxellaceae family)

These colonies did contain some members that had small and similarly colored colony morphologies. In particular, distinctive colony morphologies tended to coincide with distinctive phylogenetic positions. However, it also contained members that clearly seemed to be pink pigmented facultative Methylotrophs. Do to uncertainty in the authenticity of this data and time constraints, further examination of this data was not obtained at this point.

Members of the Order Enterobacteriales

These colonies all had a similar cell morphology, being very small and clear. As such, these colonies are being held for further characterization and analysis. Who knows how they eked out this small living of theirs, vitamins?

Members of the Family Moraxellaceae (representing 13 of the sequences within the Pseudomonadales)

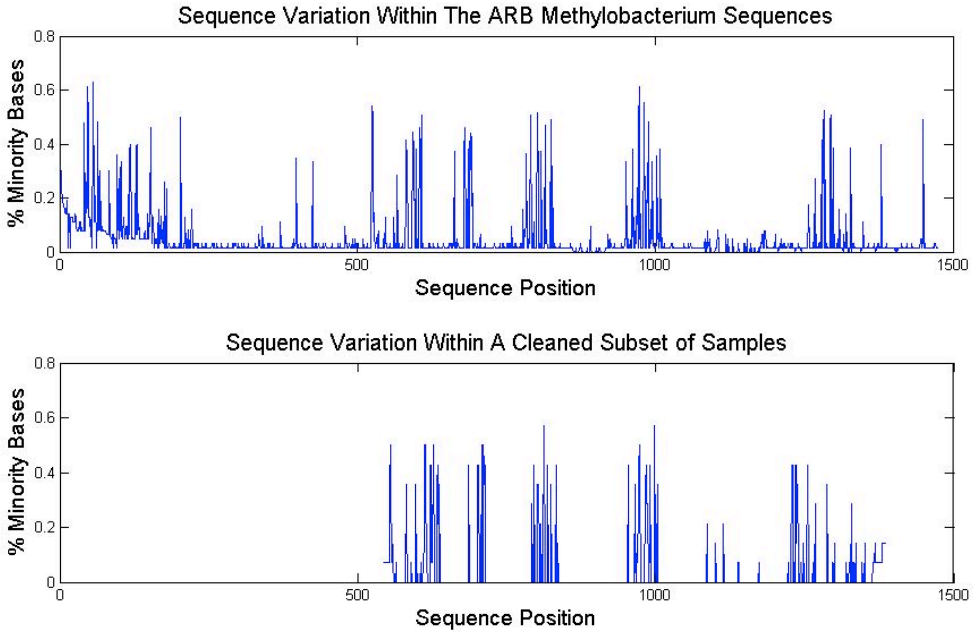
These colonies all had a similar cell morphology, being very small and clear. As such, these colonies are being held for further characterization and analysis. I am extremely confident that these colonies are Moraxellaceae, as they were very fast growing and isolated much earlier then the other colonies, and they yielded consistently scorching PCR bands. By changing the type of media I was growing these organisms on, I inferred that they were actually eating the cycloheximide, and not the methanol.

Although truly I only tested that they do grow on methanol and cycloheximide in base media but do not grow on base media alone, it seemed to me unlikely that they were growing on *both* methanol and cycloheximide since it would be unknown for this group.

Members of the Methylobacterium

It seems that I likely isolated a few different species of Methylobacterium, based on the present diversity. If this holds up, I think it is reasonably exciting. The clustering of specific genotypes closely together, and there replicate existence on different plants suggests that this biodiversity is being maintained by some force.

As a first pass look at where diversity existed in the 16S of Methylobacterium, I went through all the sequences listed as Methylobacterium in the current ARB database we were using (n=63, the database was from Jan 04). I then went through and calculated the % representation of all the non-consensus basepairs along the length of the alignment, ignoring any of the basepairs that were represented by a '-' or a "." in every sequence. I then did the same thing for a subset of the alignments of the *Methylobacterium* sequences whose electropherogram calls I had enough time to check by hand (n=14).



The figure shown below does give some insight into the dynamics. The frequent occurence of minority frequency peaks at the same height suggest that there may be at least two types of sequences segregating, and it will be interesting to see if the smaller peaks allow further refined analysis of these putative dominant groups.

Note that a tree of these sequences has not been constructed yet because I did not have enough confidence in the sequence data, however I would likely briefly state my intended analysis methods for anyone interested. Basically the idea is to look at the difference between a pair of sequences taken from different plants of the same species, to a pair of sequences taken from two different species. If the populations are distinct, the difference between pairs should be higher between species then within species. The significance of this can be tested by bootstrapping.

So how to determine the number of differences between pairs of sequences? I was originally planning on using a Hamming distance, and the code in the appendix is useful for calculating this. However, given the unexpectedly high level of diversity, a

model of distance calculation that accounts for multiple substitutions might be more interesting, and weights gaps less might be more applicable (but still treats gaps as a 5th character state).

Table 1: A Key of the different plants with successful sequencing and there phylogenetic position.

The Plate Key should be interpreted as follows:
Plant Species - Plant # - Primary Isolation Streak – Secondary Isolation Streak
If no primary isolation streak or secondary isolation streak is given, this means that the colony PCR was performed from a plate with a likely mixed culture, but whose colony appeared visually distinct.

Plate Key	Phylum	Class	Order	Family
G-5-2	Firmicutes	Bacilli	Bacillales	Bacillaceae
G-5-1	Firmicutes	Bacilli	Bacillales	Bacillaceae
E-3-1	Proteobacteria	Gammaproteobacteria	Enterobacteriales	Enterobacteriaceae
E-3-2	Proteobacteria	Gammaproteobacteria	Enterobacteriales	Enterobacteriaceae
C-5-2	Proteobacteria	Gammaproteobacteria	Enterobacteriales	Enterobacteriaceae
C-5-1	Proteobacteria	Gammaproteobacteria	Enterobacteriales	Enterobacteriaceae
F-3-1	Proteobacteria	Gammaproteobacteria	Enterobacteriales	Enterobacteriaceae
L-1	Actinobacteria	Actinobacteria	Actinomycetales	Gordoniaceae
F-8-2	Actinobacteria	Actinobacteria	Actinomycetales	Intrasporangiaceae
C-8-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
C-7-2	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
C-7-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
C-6-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
M-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
C-8-2	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
C-2-2	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
C-3-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
B-2-3	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
B-10-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
A-9-3	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
A-6-3	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
A-6-2	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
A-4-3	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
A-3-5	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae

Plate Key	Phylum	Class	Order	Family
C-2-3	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
J-5	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
G-11	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
J-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
J-4	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
F-1-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
K-1	Proteobacteria	Alphaproteobacteria	Bradyrhizobiales	Methylobacteriaceae
d-9-3	Actinobacteria	Actinobacteria	Actinomycetales	Microbacteriaceae
d-4	Actinobacteria	Actinobacteria	Actinomycetales	Microbacteriaceae
E-4-1-1	Actinobacteria	Actinobacteria	Actinomycetales	Micrococcaceae
A-1-2-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-4-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-9-1-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-1-2-Replicate	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-7-2	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-7-2-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-7-2-2	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-8-2-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
C-3-2	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-9-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
A-4-1-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
C-3-3	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
E-6-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Moraxellaceae
C-2-1	Proteobacteria	Betaproteobacteria	Burkholderiales	Oxalobacteraceae
A-7-3	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Pseudomonadaceae
C-5-4	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Pseudomonadaceae
A-3-4	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Pseudomonadaceae
A-3-1	Proteobacteria	Gammaproteobacteria	Pseudomonadales	Pseudomonadaceae
A-5-2	Proteobacteria	Gammaproteobacteri	Pseudomonadales	Pseudomonadaceae

Plate Key	Phylum	Class	Order	Family
		a		
A-9-2	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
A-3	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
C-1	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
A-7-1	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
C-6-2	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
A-7	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
A-8-1	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
C-2	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
d-2-1	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
C-1-1	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
G-9-2	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
A-1-1	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
B-2-2	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
G-9-1	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
A-2	Proteobacteria	Gammaproteobacteri a	Pseudomonadales	Pseudomonadaceae
A-1-2	Proteobacteria	Betaproteobacteria	Burkholderiales	Ralstoniaceae
C-5	Proteobacteria	Betaproteobacteria	Burkholderiales	Ralstoniaceae
J-2	Proteobacteria	Betaproteobacteria	Burkholderiales	Ralstoniaceae
F-1-3	Proteobacteria	Betaproteobacteria	Burkholderiales	Ralstoniaceae
F-5-3	Proteobacteria	Betaproteobacteria	Burkholderiales	Ralstoniaceae
B-1-1	Proteobacteria	Betaproteobacteria	Burkholderiales	Ralstoniaceae
C-4-1	Proteobacteria	Betaproteobacteria	Burkholderiales	Ralstoniaceae
F-1-2	Proteobacteria	Gammaproteobacteri a	Enterobacteriales	Unclassified
E-1-1	Proteobacteria	Gammaproteobacteri a	Enterobacteriales	Unclassified
A-10-1	Proteobacteria	Gammaproteobacteri a	Xanthomonadales	Xanthomonadaceae
A-10-2	Proteobacteria	Gammaproteobacteri a	Xanthomonadales	Xanthomonadaceae

Appendix 1: A Method To Generate Data Regarding Segregating Polymorphisms From A File Containing a Greengenes, ARB or any other Arbitrary Alignment. Note that I deliberately treated each gap as a different character state, since they are normally lumped together.

```
using System;
using System.Collections.Generic;
using System.ComponentModel;
using System.Data;
using System.IO;
using System.Text;
using System.Collections;
using System.Text.RegularExpressions;

private void MakeComparisonFiles()
        {
            //This method takes an aligned file from the arb database and determines
            //the pairwise differences
            string ProcessDirectory=@"C:\Users\Nigel Delaney\Documents\MBL\Project\";
            string InFile = ProcessDirectory+"Aligned_Methylo.fasta";
            StreamReader SR = new StreamReader(InFile);
            //StreamWriter SW2 = new StreamWriter(@"C:\Users\Nigel
       Delaney\Documents\Output_consenus.txt");
            //First to scan through and initialize the arrays

            string line;
            line = SR.ReadLine();
            line = SR.ReadLine();
            int NumofCharactersinSeq = line.Length;
            int NumofSequences = 1;
            while((line=SR.ReadLine())!=null)
            {
                if(line.StartsWith(">"))
                {
                    NumofSequences++;
                }
            }
            SR.Close();

            int[,] PairWiseStat = new int[NumofCharactersinSeq, 2];
            //This will hold the following values in each column
            //1-Num of Differences between sequences
            //2-Times A comparison was made at this base

            int numCharacters=16;
            int numCharactersToOutput = 6;//This variable will be used later to give
counts of the guys
            char[] CharactersPossible ={ '-', '.', 'U', 'A', 'C',
'G','N','S','Y','V','R','H','M','W','B','K'};
            Regex RNARegExp = new Regex("[ACGUacgu]");
            int[,] NumTimesPresent = new int [NumofCharactersinSeq,numCharacters];

            string[,] Sequences = new string[NumofSequences,2];
            //Holds all the sequences,and their names

            //Now to read them in
            SR = new StreamReader(InFile);
            int SeqIndexer = 0;
            while((line=SR.ReadLine())!=null)
            {
                if(line.StartsWith(">"))
                {
                    Sequences[SeqIndexer, 1] = line;
                    Sequences[SeqIndexer, 0] = SR.ReadLine();
                    SeqIndexer++;
                }
            }
            //Now to do Two Things, 1st get the overall diversity data, then make the
pairwise comparison matrix
```

```
            //For the first job.
            for (int j = 0; j < NumofCharactersinSeq; j++)
            {
                for (int i = 0; i < NumofSequences; i++)
                {
                    char Value = Sequences[i, 0][j];
                    int Position = -1;

                    for (int CharLoopIndex = 0; CharLoopIndex < numCharacters;
harLoopIndex++)
                    {
                        if (CharactersPossible[CharLoopIndex] == Value)
                        {
                            Position = CharLoopIndex;
                            break;                        }
                    }
                    try
                    {
                        NumTimesPresent[j, Position]++;

                    }
                    catch { MessageBox.Show("Unexpected Character :" + Value); }
                }
            }
            //NOW TO OUPUT FIRST JOB
            StreamWriter SingleOutput = new StreamWriter(ProcessDirectory +
"Overall_Diversity.csv");
            SingleOutput.Write("AlignPos,SeqPos,");
            for(int iii=0;iii<numCharactersToOutput;iii++)
            {SingleOutput.Write(CharactersPossible[iii]+",");}
            SingleOutput.Write("\n");
            int OutputCharacterPositionCounter = 1;
            for (int OutputIndex = 0; OutputIndex < NumofCharactersinSeq; OutputIndex++)
            {
                if ((NumTimesPresent[OutputIndex, 0] + NumTimesPresent[OutputIndex, 1])
!= NumofSequences)
                {

if((NumTimesPresent[OutputIndex,0]+NumTimesPresent[OutputIndex,1])!=NumofSequences)
                    {
                        SingleOutput.Write(OutputIndex.ToString() + "," +
OutputCharacterPositionCounter.ToString() + ",");
                        for (int iii = 0; iii < numCharactersToOutput; iii++)
                        { SingleOutput.Write(NumTimesPresent[OutputIndex,iii].ToString()
+ ","); }
                        SingleOutput.Write("\n");
                        OutputCharacterPositionCounter++;
                    }
                }
            }
            SingleOutput.Close();


            //Now for the second job, to make all pairwise comparisons......
            double[,] AverageHammmingDistanceMatrix=new
                            double[NumofSequences,NumofSequences];
            //Above will be a sparsely populated matrix with
            for (int Sequence1Index = 0; Sequence1Index < NumofSequences;
                            Sequence1Index++)
            {

                for (int Sequence2Index = Sequence1Index + 1; Sequence2Index <
                  NumofSequences; Sequence2Index++)
                {
                    //Going to remake the Sequence strings to make life easier
                    string Seq1 = Sequences[Sequence1Index, 0];
                    string Seq2 = Sequences[Sequence2Index, 0];
                    //Now to find the start and end of the overlap of the sequences
                    int[] LowIndex = new int[2];
                    int[] HighIndex = new int[2];
                    //SEQ1
```

```
                    Match BasePairHit = RNARegExp.Match(Seq1);
                    LowIndex[0] = BasePairHit.Index;
                    while ((BasePairHit = BasePairHit.NextMatch()).Success)
                    {
                        HighIndex[0] = BasePairHit.Index;
                    }
                    //SEQ2
                    BasePairHit = RNARegExp.Match(Seq2);
                    LowIndex[1] = BasePairHit.Index;
                    while ((BasePairHit = BasePairHit.NextMatch()).Success)
                    {
                        HighIndex[1] = BasePairHit.Index;
                    }
                    int Start = Max(LowIndex);
                    int End = Min(HighIndex);
                    int ComparisonsMade = 0;
                    int Differences = 0;
                    //Now to calculate the average hamming distance, for now going to
                      assume that the - and . are ignored
                    for (int CharacterIndex = Start; CharacterIndex < End + 1;
                                    CharacterIndex++)
                    {
                        //This is a really bad 4am solution to the issue of
                unexpected sequences that showed in the sequence, this needs to be
                shortened
                        if ((Seq1[CharacterIndex] == '.' || Seq1[CharacterIndex] == '-'
                            || Seq1[CharacterIndex] == 'N' || Seq1[CharacterIndex] == 'Y'
                            || Seq1[CharacterIndex] == 'S' || Seq1[CharacterIndex] == 'V'
                            || Seq1[CharacterIndex] == 'H' || Seq1[CharacterIndex] == 'R'
                            || Seq1[CharacterIndex] == 'W' || Seq1[CharacterIndex] == 'M'
                            || Seq1[CharacterIndex] == 'B' || Seq1[CharacterIndex] ==
'K')

                            && (Seq2[CharacterIndex] == '.' || Seq2[CharacterIndex] == '-
'
                            || Seq2[CharacterIndex] == 'N' || Seq2[CharacterIndex] == 'Y'
                            || Seq2[CharacterIndex] == 'S' || Seq2[CharacterIndex] == 'V'
                            || Seq2[CharacterIndex] == 'R' || Seq2[CharacterIndex] == 'H'
                            || Seq1[CharacterIndex] == 'W' || Seq1[CharacterIndex] == 'M'
                            || Seq1[CharacterIndex] == 'K' || Seq1[CharacterIndex] ==
'B'))
                        {
                            continue;
                        }
                        else if (Seq1[CharacterIndex] != Seq2[CharacterIndex])
                        {
                            Differences++;
                            ComparisonsMade++;
                            PairWiseStat[CharacterIndex, 0]++;//Increase Number of
Differences Observed Made
                            PairWiseStat[CharacterIndex, 1]++;//Increase the Number of
Comparisons Made
                        }
                        else { ComparisonsMade++; PairWiseStat[CharacterIndex, 1]++; }
                    }
                    if (Differences != 0)
                    {
                        AverageHammmingDistanceMatrix[Sequence1Index, Sequence2Index] =
((double)Differences / (double)ComparisonsMade);
                    }
                    else
                    {
                        AverageHammmingDistanceMatrix[Sequence1Index, Sequence2Index] = -
999;
                    }
                }
            }

            SR.Close();
            //NOW TO OUTPUT EVERYTHING-Note this could be incorporated above, but thought
best seperate for now.
```

```
            StreamWriter SW = new StreamWriter(@"C:\Users\Nigel
Delaney\Documents\MBL\Project\HammingDistanceMatrix.csv");
            SW.Write(",");//
            for (int SeqTitleOutput = 0; SeqTitleOutput < NumofSequences;SeqTitleOutput++
)
            {
                SW.Write(Sequences[SeqTitleOutput, 1] + ",");
            }
            SW.Write("\n");
            for(int OutputIndex=0;OutputIndex<NumofSequences;OutputIndex++)
            {
                SW.Write(Sequences[OutputIndex,1]+",");
                //Now some more commaspacer
                for(int commaSpacer=0;commaSpacer<OutputIndex+1;commaSpacer++)
                {SW.Write(",");}
                for (int ElementSpacer = OutputIndex + 1; ElementSpacer < NumofSequences;
ElementSpacer++)
                {

SW.Write(AverageHammmingDistanceMatrix[OutputIndex,ElementSpacer].ToString("n5")+",");
                }
                SW.Write("\n");
            }
            SW.Close();
            SW = new StreamWriter(ProcessDirectory + "OutputPairwiseVariation.csv");
            SW.WriteLine("Align_Pos,Seq_Pos,Num_Comparisons,Percent_Dif");
            int Seq_Pos_Counter = 1;//This gives the position in the sequence with
useless information removed
            for (int CharacterIndex2 = 0; CharacterIndex2 < NumofCharactersinSeq;
CharacterIndex2++)
            {
                if(PairWiseStat[CharacterIndex2,1]!=0)//Check to make sure this is an
informative site
                {
                    double PercentDifferent =
(Convert.ToDouble(PairWiseStat[CharacterIndex2,0])/Convert.ToDouble(PairWiseStat[Characte
rIndex2,1]));

SW.Write(CharacterIndex2.ToString()+","+Seq_Pos_Counter.ToString()+","+PairWiseStat[Chara
cterIndex2,1].ToString()+","+PercentDifferent.ToString()+"\n");
                    Seq_Pos_Counter++;
                }
            }
            SW.Close();
        }

        private int Max(int[] ArrayOfValues)
        {
            int Max = ArrayOfValues[0];
            for (int i = 1; i < ArrayOfValues.Length; i++)
            {
                if (ArrayOfValues[i] > Max)
                {
                    Max = ArrayOfValues[i];
                }
            }
            return Max;
        }
        private int Min(int[] ArrayOfValues)
        {
            int Min=ArrayOfValues[0];
            for (int i = 1; i < ArrayOfValues.Length; i++)
            {
                if (ArrayOfValues[i] < Min)
                {
                    Min = ArrayOfValues[i];
                }
            }
            return Min;
        }
```