Genomic Analysis of Five Novel, Formerly-Uncultivable Oral Bacteria belonging to the family Peptostreptococcaceae

by Amanda Chinyere Chilaka

B.S. in Biology, Northeastern University

A dissertation submitted to

The Faculty of
the College of Arts and Sciences
Northeastern University
in partial fulfillment of the requirements
for the degree of Master of Science

June 25, 2014

Dissertation directed by

Slava Epstein
Professor of Biology
I would first like to acknowledge my advisor, Professor Slava Epstein, and Dr. Maria Sizova, for guiding me through the entirety of my graduate stay. I would also like to thank the members of the Epstein lab for their support and assistance with research. For teaching me basic programming, I would like to thank my post-baccalaureate mentors, Dr. Kyriacos Leptos and Dr. Andrew Tolonen, and Professor Mitchell Model of Northeastern University. Additionally I would like to thank Ashlee Earl and Margaret Priest from the Broad Institute, Manolito Torralba of JCVI, Dr. Yuri Wolf of NCBI for running the RAST annotation data through Psi-Cognitor, Dr. William Fowle for assistance with transmission electron microscopy, Markus Krummenacker and Dr. Peter Karp for technical support with Pathway Tools, Dr. Tonje M. B. Heggeset for assistance with BASys, and Dr. Michelle Giglio and Suvvi Nadendla of the Institute for Genome Sciences.

Acknowledgements
Abstract of Dissertation

According to the Human Oral Microbiome Database, approximately 700 prokaryotic species are present in the human oral cavity; 49% of these are cultivable and officially named, 17% are cultivable and unnamed, and 34% are a part of uncultivated phylotypes (Chen et al., 2010; Dewhirst et al., 2010). Cultivation of oral bacteria is important because a better understanding of their physiology could help in treating such pathologies as periodontal disease; between 2009 and 2010, approximately 47.2% of adults age 30 and up had periodontitis (Thornton-Evans et al., 2013). In our laboratory, five oral bacteria, that were formerly classified as uncultivable, were cultivated from anaerobic enrichments from the subgingival cavities of healthy human subjects as part of the Human Microbiome Project (Sizova et al., 2012). Isolates were originally designated as Eubacteriaceae bacterium ACC19a, CM2, CM5, OBRC8 and Eubacterium sp. AS15. They have since been renamed as Peptostreptococcaceae spp. ACC19a, CM2, CM5, OBRC8, and AS15. The first three isolates were sequenced at the BROAD Institute and the latter two at the JCVI.

It is hypothesized that a combination of annotation pipelines will provide a broader representation of genes present in each bacterial genome and could reveal metabolic pathways that could be missed. Supplemental tools such the Rapid Annotation using Subsystem Technology (RAST) service (Aziz et al., 2008) and Pathway Tools (Karp, Paley, Krummenacker, & al, 2010) were invaluable in determining some of the metabolic pathways that are present or absent for the isolates. One of the aims of this project is to determine what differences there are between the oral isolates and their closest relatives, as well as determining the differences between the isolates.

Abstract of Dissertation
Table of Contents

Acknowledgements Page ................................................................. ii
Abstract of Dissertation ................................................................. iii
Table of Contents ........................................................................ iv
Chapter 1: Introduction .................................................................. 1
  The Oral Microbiome .................................................................. 1
  Databases .................................................................................. 1
  Genomic Annotation ................................................................. 2
Chapter 2: The Peptostreptococaceae ............................................. 3
  Genus Filifactor ......................................................................... 4
    Filifactor alocis ........................................................................ 4
  The Peptostreptococaceae spp. ACC19a, CM2, CM5, OBRC8, and AS15 ................... 6
Chapter 3: Transmission Electron Microscopy ................................ 9
  Preparation of the Oral Isolates .................................................. 9
  TEM Images ............................................................................. 11
Chapter 4: Genomic Sequencing of the Oral Isolates ....................... 16
  The Broad Institute .................................................................... 16
    Library Construction and Sequencing ..................................... 16
    Assembly and Annotation ....................................................... 16
  The J. Craig Venter Institute (JCVI) ........................................ 17
    Library Construction and Sequencing. .................................. 17
    Assembly and Annotation ....................................................... 17
  Sequencing Results Overview .................................................. 18
Chapter 5: RAST, BASys, XBase, and Manatee ............................. 19
  1. RAST ................................................................................ 19
  2. BASys (Bacterial Annotation System) ................................... 24
  3. xBASE .............................................................................. 27
  4. Manatee and the IGS Analysis Engine .................................. 29
Chapter 6: Comparison of Annotation Pipelines ........................... 32
  Using Python to Combine Annotation Data ............................... 32
Chapter 7: Using Annotations to make Predictions about Metabolic Pathways ....................... 37
  1. RAST Subsystem and Pathway Tool Predictions .................. 37
  2. Carbohydrate Utilization ..................................................... 37
    D-Maltose Utilization ........................................................... 37
    D-Lactose Utilization ............................................................ 39
  3. Amino Acid Biosynthesis .................................................... 40
Chapter 1: Introduction

The Oral Microbiome

The oral microbiome is made up of over 600 different species of bacteria, and subsets of those species colonize different habitats within the mouth, e.g. oral biofilms and the subgingival cavities (Dewhirst et al., 2010). As much as 50% of these organisms are not cultivated (Olsen, Preza, Aas, & Paster, 2009). Oral bacteria play a crucial role in the health of organisms and an imbalance in the microbiome can lead to serious disease. Dental plaque, or biofilm, is an important ecological niche, and it is composed of oral bacteria protected by a complex of exopolysaccharides (also known as the glycocalyx). The majority of bacteria in the mouth are anaerobic and fastidious, making it difficult to understand bacteria-related pathologies. The techniques mainly used for identifying oral bacteria are PCR, cloning, and 16S rRNA gene sequencing (Olsen et al., 2009); microarrays are also used. Newer technique, such as the minitrap method, have been developed to grow newer, formerly uncultivable bacteria (Sizova et al., 2012).

Databases

In a study by Dewhirst et. al., they aimed 1.) to collect 16S rRNA gene sequences of bacteria and archaea into a curated, phylogeny-based database, known as the Human Oral Microbiome Database (HOMD), 2.) to analyze the relative abundance of taxa, and 3.) to identify novel taxa (Dewhirst et al., 2010). As of May 2014, HOMD contained 691 taxa, 355 of which have sequenced and annotated genomes; there are 13 phyla represented, including, Actinobacter,
Bacteroidetes, Chlamydiae, Chloroflexi, Euryarchaeota, Firmicutes, Fusobacteria, GN02, Proteobacteria, Spirochaetes, SR1, Synergistetes, Tenericutes, and TM7.

The Human Microbiome Project (HMP) is focused on characterizing organisms that are found within the human body 1.) to develop a reference set of microbial genome sequences and characterize the human microbiome; 2.) to interpret the relationship between disease and changes in the human microbiome; and 3.) to develop new technologies and tools for computational analyses, to name a few (NIH, 2013). The first initiative is of importance to our lab as one of our goals is to cultivate “uncultivable” bacteria of the human oral cavity. Five isolates, Peptostreptococcaceae spp. ACC19a, CM2, CM5, OBRC8, and AS15 are discussed in this thesis. The isolates were chosen for genomic sequencing based on their novelty, and the subsequent data are important in characterizing them. The isolates may form a novel genus within the family Peptostreptococcaceae.

Genomic Annotation

Full manual curation of genomic data is not routinely done for each genome sequenced because of how time-consuming and expensive the process is. It can take many months and require numerous collaborators to get through a single bacterial genome. Currently, manual curation is only done after predictions have been made in silico by an automatic annotation pipeline. Numerous annotation pipelines have been developed in the past decade. A few free and easy-to-use pipelines include: the Rapid Annotations using Subsystems Technology (RAST) (Aziz et al., 2008); the Bacterial Annotation System (BASys) (Van Domselaar et al., 2005); xBASE (Chaudhuri & Pallen, 2006); and Manatee (TIGR, 2007). It has been shown that when
output from multiple pipelines is combined, more thorough gene predictions can be made (Ederveen, Overmars, & van Hijum, 2013). Using the annotations, it is hypothesized that the presence or absence of metabolic pathways can be predicted using the genomic data and then validated by actual biological testing. Using the genomic data can save time when trying to characterize isolates and can highlight certain traits that require further testing.

Chapter 2: The Peptostreptococcaceae

According to Bergey’s Manual of Systematic Bacteriology, as of 2011, there were three genera belonging to the family Peptostreptococcaceae: *Filifactor*, *Peptostreptococcus*, and *Tepidibacter* (Ezaki, 2011). According to the National Center for Biotechnology Information (NCBI), as of May 2014, the current genera include: *Anaerosphaera*, *Filifactor*, *Murdochiella*, *Peptoclostridium*, *Peptostreptococcus*, *Sporacetigenium*, and *Tepidibacter*. Of those six genera, only *Peptoclostridium* and *Murdochiella* are without standing in nomenclature yet (http://www.bacterio.net). The family is composed of Gram-positive cocci or rods that may or may not be motile. They are obligately or facultatively anaerobic and most are catalase-negative; only the genus *Filifactor* forms endospores (Ezaki, 2011). The [*Eubacterium* yurii] species, the closest related species to the novel oral isolates, will soon be reclassified under the family Peptostreptococcaceae based on 16S rRNA gene sequence analysis. The next closest related species to the isolates belong to the *Filifactor* genus.
Genus Filifactor

The genus *Filifactor* was proposed in 1994 based on 16S rRNA gene sequence analysis of named and unnamed clostridial strains (Collins et al., 1994). The type species is *Filifactor villosus*, which was renamed from *Clostridium villosum*. Cells are long rods that are non-motile and they may form filaments. In exponential phase, cells are Gram-positive, but they turn Gram-variable or Gram-negative as the cultures age (Love & Bailey, 1979). *Filifactor* species are able to form oval, subterminal spores. The products of fermentation include: acetate, butyrate, isobutyrate, formate, and isovalerate; pyruvate is converted into butyrate. The cell walls contain ornithine-D-asparagine-type murein (type α21.4 or α22.1\(^1\)). *Filifactor villosus* weakly liquefies gelatin; and it is unable to ferment esculin, glucose, fructose, lactose, maltose, mannitol, mannose, melibiose, ribose, sucrose, and xylose.

*Filifactor alocis*

*Filifactor alocis* was formerly known as *Fusobacterium alocis* (Cato, Moore, & Moore, 1985). It was renamed because of its clustering with *Filifactor villosus* within Clostridium cluster XI; they formed a deep branch within the tree (Jalava & Eerola, 1999). Cells are non-spore-forming, obligately anaerobic rods that have been reported as Gram-negative (Ezaki, 2011) or Gram-positive (Aruni, Roy, Sandberg, & Fletcher, 2012), depending on the source. Most articles report it as Gram-positive and RAST analysis supports this; genes involved in teichoic and lipoteichoic acid biosynthesis were found. *Filifactor villosus* and *Filifactor alocis* can be distinguished by

\(^1\) The group name of murein was found from http://www.dsmz.de/fileadmin/Bereiche/Microbiology/Dateien/Key_to_Murein2.pdf
the latter’s inability: 1.) to produce isobutyrate and isovalerate, 2.) to use pyruvate, and 3.) to produce spores.

*Filifactor alocis* has been associated with periodontal disease (Aruni et al., 2012); it can induce the secretion of pro-inflammatory cytokines (IL-1β, IL-6, and TNF-α) from gingival epithelial cells (GECs), cause apoptosis in GECs, and possibly suppress MEK 1/2 (mitogen-activated protein kinase 1/2) activation (Moffatt, Whitmore, Griffen, Leys, & Lamont, 2011). Using fluorescent microscopy, Aruni et al. observed that *F. alocis* adheres to the surface of GECs and may enter the cells. *F. alocis* is able to ferment glucose, can produce acetic acid and butyric acid, is weakly stimulated by arginine, is indole negative, does not hydrolyze esculin, and can breakdown arginine (Downes et al., 2001).

According to an article by Uematsu et al., when supplemented with amino acids, *F. alocis* only used arginine, though enzymatic activities of arginine deiminase (EC 3.5.3.6) and ornithine carbamoyltransferase (EC 2.1.3.3) were not detected (Uematsu, Sato, Hossain, Ikeda, & Hoshino, 2003). Based on this observation, it was reasoned that arginine was converted directly to ornithine (not through citrulline), and then into butyrate and ammonia. However, according to RAST, a gene for arginine deiminase was present in the genome. RAST did not identify a gene for ornithine carbamoyltransferase.
The Peptostreptococcaceae spp. ACC19a, CM2, CM5, OBRC8, and AS15

Based on 16S rRNA gene sequence comparison, the isolates may form a novel genus within the family Peptostreptococcaceae. Figure 1 shows a phylogenetic tree containing the type species belonging to the different genera of the family Peptostreptococcaceae, a few species that have been recently reclassified under this family, the type species for the genus *Eubacterium*, and the oral isolates. According to the HOMD, strains ACC19a, CM2, CM5, and OBRC8 represent the first known cultivable members of “uncultivable” human oral taxon 081; strain AS15 is classified as a member of “cultivable” oral taxon 377 (www.homd.org). Strain AS15 is distantly related to strains ACC19a, CM2, CM5, and OBRC8 with 93.6 – 94.0% sequence similarity and is closely related to *[Eubacterium] yurii* subsp. *margaretiae* with 99.3% sequence similarity. Compared to *Filifactor alocis*, which belongs to the next closely-related group within the family Peptostreptococcaceae, the sequence similarity is 85.5%.
Figure 1: Phylogenetic Tree

Eubacterium limosum strain ATCC 8486 NR 044719
Anaerosphaera aminiphila AB298735
Peptostreptococcus anaerobius AY326462
Peptostreptococcus stomatis DQ160208
Tepidibacter thalassicus AY158079
Sporacetigenium mesophilum strain ZLJ115 AY682207
Clostridium bartlettii AY438672
Clostridium hiranonis AB023970
Clostridium irregulare X73447
Eubacterium tenue M59118
Filifactor alocis ATCC 35896 AF537211
Filifactor villosus DSM1645 AF537211
Eubacterium yurii subsp. margaretiae AY533381
Peptostreptococcaceae sp. AS15 HQ616364
Eubacterium yurii subsp. yurii L34629
Eubacterium yurii subsp. schtitka AY533382
Peptostreptococcaceae sp. ACC19a HM120213
Peptostreptococcaceae sp. CM5 HM120214
Peptostreptococcaceae sp. CM2 HM120215
Peptostreptococcaceae sp. OBRC8 HM120217

0.02
Strains ACC19a, CM2, CM5, and OBRC8 are Gram-positive rods that are distinguishable from strain AS15 by their rounded ends; strain AS15 has square ends. While ACC19a, CM2, CM5, and OBRC8 form short to long chains, AS15 appears to form rosettes, similar to *Eubacterium yurii* subsp. *margaretiae*. The width of the cells ranges from 0.4 to 0.8μm and the length ranges from 1.0 to 4.7 μm. All isolates have either a single, subpolar flagellum, as is the case with strain AS15, or multiple peritrichous flagella. The optimal temperature for growth is 37˚ C.

Based on biological testing, using API 20A test strips, all five strains produced acid in media containing glucose, maltose and sucrose, but not lactose, arabinose, cellobiose, mannose, melezitose, raffinose, rhamnose, trehalose, xylose, glycerol, mannitol, salicin, and sorbitol. In liquid medium, supplemented with 5.0 g l-1 of yeast extract, strain CM2 fermented D-glucose, D-sucrose, and D-maltose; OBRC8 fermented D-sucrose and D-maltose; and AS15 fermented D-glucose, D-sucrose, and D-maltose. For strains ACC19a, CM2, and CM5, metabolic end products were detectible and they included acetate and propionate; this trait distinguishes the isolates from the *Filifactor* genus. All isolates, except for strain AS15, were indole negative. All strains were negative for urease, catalase, oxidase, and nitrate reduction. All isolates are susceptible to erythromycin, kanamycin, penicillin, chloramphenicol, tetracycline, and bile.
Chapter 3: Transmission Electron Microscopy

Preparation of the Oral Isolates

The oral isolates were prepared for Transmission Electron Microscopy (TEM) between June and August of 2013 with the guidance of William Fowle at Northeastern University. Cultures were grown overnight in trypticase/yeast extract broth, according to previously described methods (Sizova et al., 2012), and then pelleted. The pellets were rinsed in 0.1M sodium cacodylate buffer and incubated at 4°C for 45 minutes. The used buffer was exchanged for fresh buffer twice, once after 15 minutes and once after 30 minutes. After the first 15 minutes, the pellets were diced into small pieces with a scalpel. The diced pellets were then incubated in post fixative, composed of 1.0% osmium tetroxide and 0.1M sodium cacodylate buffer, for 2 hours. The diced pellets were then rinsed in 0.1M sodium cacodylate buffer again for 45 minutes, with two changes. The diced pieces were then transferred to vials.

The pieces then went through a graded ethanol series; 30%, 50%, and 70% ethanol were used and each incubation lasted 10 minutes. The pellets were then left at 4°C overnight in 70% ethanol. The following day, the graded ethanol series was continued, using 85% (10 minutes), 95% (10 minutes), and 100% (2 hours, 2 changes) ethanol. A resin of Spurrs and Quetol (Squetol) was prepared and a 1:1 ratio of this and 100% ethanol was added to the diced pellets, after the 100% ethanol was removed. After the vials were incubated for 2 hours, the resin-ethanol mixture was replaced with 100% Squetol and allowed to incubate overnight, with two changes of fresh resin. Resin was aliquoted into Eppendorf tubes and the diced pellets were submerged into it. The samples were cured overnight in a 60°C oven. Ultrathin sections were
made using an ultramicrotome and the ultrathin sections were observed using a transmission electron microscope.

The first goal of this project was to determine the cell wall structure of the isolates; for all of the isolates, they were Gram-positive. Based on RAST annotation, the presence of genes for teichoic acid biosynthesis and the absence of genes for lipopolysaccharide biosynthesis support this finding. It was also apparent that for isolates ACC19a (Figure 2), CM2 (Figure 3), CM5 (Figure 4), and OBRC8 (Figure 5) that flagella were present and were peritrichous. For isolate AS15, the flagella were not as obvious in the TEM images (Figure 6) but, analysis of the genomic data supported the presence of flagella.
Figure 2: TEM image of Peptostreptococcaceae sp. ACC19a
Figure 3: TEM image of Peptostreptococcaceae sp. CM2
Figure 4: TEM image of Peptostreptococcaceae sp. CM5
Figure 5: TEM image of Peptostreptococcaceae sp. OBRC8
Figure 6: TEM image of Peptostreptococcaceae sp. AS15
Chapter 4: Genomic Sequencing of the Oral Isolates

The Broad Institute

Library Construction and Sequencing

The Broad Institute used 454 pyrosequencing for isolates ACC19a, CM2, and CM5 and did additional Illumina sequencing for isolate CM2. The Broad is in the process of switching over completely to Illumina sequencing. With 454, the genomic DNA (gDNA) is shattered into small fragments (about 0.6kb or 3kb) and the fragments are tailed with 454 sequencing adaptors. The fragments are circularized on a biotinylated linker and the circles are sheared. The fragments containing the biotinylated linkers are retrieved and tailed with 454 sequencing adapters again. The fragments with the adapters are sequenced from one end, creating fragment or paired reads (Institute, 2010).

Assembly and Annotation

Using Newbler version 2.3, the fragment and paired reads are put together to identify contiguous stretches of sequence, known as contigs. The contigs are ordered and linked together into larger supercontigs (scaffolds) using the paired reads. Once the assemblies are complete, PRODIGAL (Prokaryotic Dynamic Programming Gene-finding Algorithm) is used for predicting coding regions (Hyatt et al., 2010). Blast homology is searched against GenBank’s non-redundant (NR) database and e-values better than 1e-10 are used as Blast evidence. Pfam domains are found by running HMMER, using the Pfam/TIGRfam library (Punta et al., 2012) to find domains on six-frame translations of the sequence. Ribosomal RNAs are identified with RNAmmer and tRNAs are identified using tRNAscan (Fichant & Burks, 1991). Other RNA features are identified with RFAM. Reference genomes may also be used to improve automated annotations (Institute, 2010).
The J. Craig Venter Institute (JCVI)

Library Construction and Sequencing.

The JCVI uses Illumina sequencing, specifically the HiSeq™ 2000 sequencing system. The gDNA is fragmented and sheared ends are repaired and adenylated. Adaptor oligos are ligated to each end of the fragments. The fragments are then size-selected and purified. Single molecules are isothermally amplified in a flow cell. The flow cell is coated with oligos, which hybridize to the adapters. The bound fragments are extended to create copies. Each library fragment is clonally amplified through a series of extensions and isothermal bridge amplifications, creating hundreds of millions of unique clusters. Reverse strands are cleaved and washed away and ends blocked and a sequencing primer is hybridized to the DNA template and the template is ready for sequencing.

Assembly and Annotation

After sequencing, the whole genome sequence (WGS) data is assembled using Celera Assembler version 6.1. The automated Prokaryotic Annotation Pipeline was developed by JCVI to generate open reading frame (ORF) predictions and functional annotations for the assembled genomic data (JCVI, 2014). Gene finding is done using GLIMMER3 (Delcher, Bratke, Powers, & Salzberg, 2007) and non-coding RNA predictions are done using tools such as tRNAscan-SE (Lowe & Eddy, 1997) and BLAST (Altschul, Gish, Miller, Myers, & Lipman, 1990). The predicted genes are analyzed for overlaps and homology-based evidence is gathered using Hidden Markov Model (HMM) searches and BLAST. Roles and gene symbols are assigned based on the analyses, common names, GO terms and EC numbers.
Sequencing Results Overview

The general summary of genomic sequencing, coverage, and number of contigs are found in Table 1. A summary of the genome sizes, G+C content, and gene prediction are found in Table 2. Note that for isolate CM2, which used a combination of 454 and Illumina, it had the highest sequencing coverage of all the isolates; it also had the fewest number of contigs. Interestingly, isolate CM2 had the smallest genome size and the fewest number of genes annotated.

<table>
<thead>
<tr>
<th>Institute</th>
<th>Sequencer</th>
<th>Coverage</th>
<th>Contigs (scaffolds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peptostreptococcaceae sp. ACC19a</td>
<td>Broad</td>
<td>454</td>
<td>40x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. CM2</td>
<td>Broad</td>
<td>454/Illumina</td>
<td>282x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. CM5</td>
<td>Broad</td>
<td>454</td>
<td>39x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. OBRC8</td>
<td>JCVI</td>
<td>454</td>
<td>39x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. AS15</td>
<td>JCVI</td>
<td>454</td>
<td>39x</td>
</tr>
</tbody>
</table>

Table 1: General sequencing information for the five isolates

<table>
<thead>
<tr>
<th>Institute</th>
<th>Sequencer</th>
<th>Coverage</th>
<th>Contigs (scaffolds)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peptostreptococcaceae sp. ACC19a</td>
<td>Broad</td>
<td>454</td>
<td>40x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. CM2</td>
<td>Broad</td>
<td>454/Illumina</td>
<td>282x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. CM5</td>
<td>Broad</td>
<td>454</td>
<td>39x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. OBRC8</td>
<td>JCVI</td>
<td>454</td>
<td>39x</td>
</tr>
<tr>
<td>Peptostreptococcaceae sp. AS15</td>
<td>JCVI</td>
<td>454</td>
<td>39x</td>
</tr>
</tbody>
</table>

Table 2: Number of total genes and protein genes were obtained from RefSeq

To obtain the genomic data for each of the isolates, FASTA nucleotide files were downloaded from the links found in Table 3. For the gzip (.gz) files that were downloaded from NCBI, they had to be extracted using the file archiver 7-Zip (http://www.7-zip.org/download.html). Files were put into directories specific for each sequencing center and isolate for easy location. These
files were then used in different annotation pipelines, which will discussed in the following section.

<table>
<thead>
<tr>
<th>Source of FASTA download</th>
<th>Link</th>
</tr>
</thead>
<tbody>
<tr>
<td>ACC19a</td>
<td><a href="https://olive.broadinstitute.org/collections/euba_bact_acc19a.1/downloads/scaffolds.fasta">https://olive.broadinstitute.org/collections/euba_bact_acc19a.1/downloads/scaffolds.fasta</a></td>
</tr>
<tr>
<td>CM2</td>
<td><a href="https://olive.broadinstitute.org/collections/euba_bact_cm2.1/downloads/scaffolds.fasta">https://olive.broadinstitute.org/collections/euba_bact_cm2.1/downloads/scaffolds.fasta</a></td>
</tr>
<tr>
<td>CM5</td>
<td><a href="https://olive.broadinstitute.org/collections/eubact_bact_cm5.1/downloads/scaffolds.fasta">https://olive.broadinstitute.org/collections/eubact_bact_cm5.1/downloads/scaffolds.fasta</a></td>
</tr>
<tr>
<td>OBRC8</td>
<td><a href="http://www.ncbi.nlm.nih.gov/Traces/wgs/?download=ALNK01.1.fsa_nt.gz">http://www.ncbi.nlm.nih.gov/Traces/wgs/?download=ALNK01.1.fsa_nt.gz</a></td>
</tr>
</tbody>
</table>

Table 3: The links in this table were used to download the genomic data for each of the isolates.

Chapter 5: RAST, BASys, XBase, and Manatee

The four annotation pipelines that were used for comparison were chosen based on their ease-of-use and relatively quick turnaround times.

1. RAST

RAST is an annotation service that makes use of the National Microbial Pathogen Database Resource (NMPDR); the NMPDR uses a unique method of classifying predicted genes by using subsystems, or families of functional genes (FIGfams), that have been derived from the literature and experimental data (Aziz et al., 2008). At present, there are over 600 subsystems, based on genes found from over 500 bacterial and archaeal genomes. RAST can take as input either whole genome sequences or contig data, in FASTA or GenBank format, and can complete the gene and subsystem predictions in 12 to 24 hours. RAST is accessible at http://rast.nmpdr.org/.

RAST uses a 16-step process to annotate genomes (Overbeek et al., 2014). First it identifies the selenoproteins and pyrrolysoproteins, which are special-case genes, using custom algorithms. It
then uses GLIMMER3 to find gene candidates and generates an estimate of the 30 closest phylogenetic neighbors in the SEED database (http://pubseed.theseed.org). tRNAs and rRNA are identified using an in-house script named ‘search_for_rnas,’ which makes use of tRNAscan-SE and BLASTN. Genes initially identified by GLIMMER3 are tested to see if they match proteins in known subsystems using k-mers of eight amino acids. Candidates with evidence for subsystem-based function are promoted to protein-encoding genes (PEGs) and assigned a functional role. GLIMMER3 is iteratively retrained on the gene candidates that have been validated, and this is used to find genes until there are no new candidates that are similar to those in subsystems. Gene candidates are only kept if they match a gene in a subsystem and do not overlap with another gene. Convergence is usually reached after three iterations.

Genes that do not significantly overlap an existing gene are included if they are similar to any protein found in the 30 closest neighbors using BLASTP. Remaining genes that do not significantly overlap and existing gene are kept. Gene fragments are then detected by comparing them to the template genes in the 30 nearest neighbors. If there are stretches of DNA longer than 1500bp that do not contain a gene, they are ‘backfilled’ with gene candidates using BLASTX. Products of genes that were not assigned in previous steps are given functions using BLASTP. If a gene has not been given a subsystem-based function, it is compared to the nearest gene neighbors; if all three genes are bidirectional best hits (BBHs) to a corresponding set of three genes in a neighboring genome, the gene assignment is replaced by a subsystem-based functional role. Missed genes are identified by looking at the remaining gaps that are flanked by BBHs. Gene candidates that do not have a subsystem or BLAST support are removed if they overlap with another gene or if they are less than 90 nucleotides long. Subsystem analysis and metabolic reconstruction are carried out, and the variant is estimated; annotations are connected to the metabolic model. Pairs of close bidirectional best hits are computed against genomes in
PubSEED; estimates of functional coupling are supported by this. Lastly, genome data is exported in a number of different formats, including GenBank and tab-delimited.

The steps involved in submitting genomic data to RAST is described here. Step one requires uploading the FASTA file. Information about the organism then has to be verified in step 2. The taxonomy ID, which can be found in the NCBI Taxonomy browser, can be entered in the Taxonomy ID Field; after clicking the “Look up taxonomy ID at NCBI,” the rest of the fields will automatically be filled in (Figure 7). In step 3, there are optional questions to answer including, what sequencing method was used and what the coverage value was. The other options for each RAST annotation were left in their default settings and the upload was finished. The same process was done for all five isolates. When the jobs were done, the annotations were downloaded as FASTA amino acid files (.faa).
Figure 7: RAST genome upload, step 2
Figure 8: RAST genome upload, step 3
A summary of the genes predicted by RAST can be found in Table 4, along with the predicted number of subsystems and percent of genes found in those subsystems. For comparison, the genomic data of *Eubacterium limosum* KIST612, *[Eubacterium] yurii subs. margaretiae*, and *Filifactor alocis* were also run through RAST.

<table>
<thead>
<tr>
<th>Species</th>
<th># gene predictions</th>
<th># subsystems</th>
<th>% genes in subsystem</th>
</tr>
</thead>
<tbody>
<tr>
<td>Peptostreptococcaceae bacterium ACC19a</td>
<td>2345</td>
<td>277</td>
<td>40%</td>
</tr>
<tr>
<td>Peptostreptococcaceae bacterium CM2</td>
<td>2044</td>
<td>273</td>
<td>43%</td>
</tr>
<tr>
<td>Peptostreptococcaceae bacterium CM5</td>
<td>2390</td>
<td>279</td>
<td>39%</td>
</tr>
<tr>
<td>Peptostreptococcaceae bacterium OBRC8</td>
<td>2296</td>
<td>266</td>
<td>36%</td>
</tr>
<tr>
<td>Peptostreptococcaceae bacterium AS15</td>
<td>2300</td>
<td>293</td>
<td>41%</td>
</tr>
<tr>
<td><em>Eubacterium limosum</em> KIST612</td>
<td>4319</td>
<td>340</td>
<td>33%</td>
</tr>
<tr>
<td><em>[Eubacterium] yurii subs.margaretiae</em></td>
<td>2097</td>
<td>286</td>
<td>43%</td>
</tr>
<tr>
<td><em>Filifactor alocis</em></td>
<td>1896</td>
<td>240</td>
<td>38%</td>
</tr>
</tbody>
</table>

Table 4: RAST summary for the five oral isolates and three other species for comparison

2. BASys (Bacterial Annotation System)

BASys is a web server that performs automated annotations of bacterial genomes (Van Domselaar et al., 2005). BASys is accessible at https://www.basys.ca. A personal account was created and analysis was run on BASys Server 4. BASys only takes complete whole genome sequences, not files that contain multiple contigs or scaffolds. To get around this, a dummy sequence, with stop codons in all six reading frames, was inserted in between each contig. The dummy sequence used was “nnnnncacacacttaattaattaagtgtgtgmn.” It was obtained through email correspondence with another researcher that used the BASys software for a publication (Heggeset et al., 2012). For each isolate, an annotation report was created with multiple output file types to download. The one selected for comparison to the other annotation pipelines was the FASTA amino acid file. Figure 9 shows a screen shot of the BASys genome submission
page, where information must be entered manually. Figure 10 shows a screenshot of the BASys site after logging into Server 4; all jobs that have been run are stored for future access.

![Screenshot of BASys Genome Submission page](image)

Figure 9: Screenshot of BASys Genome Submission page
Figure 10: Screenshot of BASys site after logging into a Server.
3. xBASE

xBASE was the fastest of the annotation pipelines used in this analysis; annotations can be completed in as little as 20 minutes (Chaudhuri et al., 2008). xBASE can be accessed at http://www.xbase.ac.uk/annotation/. A screenshot of the service can be found in Figure 11. In order to annotate a genome, a reference genome is required, which can be selected from a dropdown menu on the site. The software only takes FASTA nucleotide sequences and first uses GLIMMER (Delcher et al., 2007) to make gene predictions. To find tRNA genes, xBASE uses tRNAScan-SE (Lowe & Eddy, 1997) and to find rRNA genes, it uses RNAammer (Lagesen et al., 2007). A protein BLAST is run against the reference genome, using the translated coding sequences. The best BLAST search results are used in the annotations. The output can be downloaded in GenBank or FASTA protein format. While the GenBank file does include predicted product names, one drawback about xBASE is that the FASTA protein file does not include product names.

At the time of data submission to xBASE, which was on July 9th, 2013, the reference genome used was Eubacterium rectale ATCC 33656. It was selected because the isolates at one point were believed to be Eubacterium species, based on their close 16S rRNA gene similarity to the Eubacterium yurii species. Attempts to rerun the annotations using “Eubacterium yurii subsp. margaretiae ATCC 43715” as a reference genome in May of 2014 failed because of an error with the service. Names of the output files were modified to be more descriptive and files were put into respective directories for comparison to the other pipelines.
Figure 11: Screenshot of the xBASE bacterial annotation service
4. Manatee and the IGS Analysis Engine

Manatee is a tool used to perform manual functional annotations and uses the Institute for Genome Sciences (IGS) Analysis Engine. Manatee was developed by the IGS and it is currently under continuing development both by the IGS and J. Craig Venter Institute (JCVI). The pipeline uses GLIMMER3 (Delcher et al., 2007) to make gene predictions, tRNA-scanSE (Lowe & Eddy, 1997) to predict tRNAs, and RNAmmer (Lagesen et al., 2007) to predict rRNAs. Further information about the pipeline can be found on the IGS website (IGS, 2014 #222). This link, http://ae.igs.umaryland.edu/cgi/index.cgi, brings you to a page where you can contact someone and submit genomes for annotation.

The FASTA nucleotide files were sent to Dr. Michelle Giglio of IGS and the data was run through Manatee by Suvvi Nadendla in August of 2013. Since the genomic data for each of the isolates contained multiple contigs, for each isolate, the contigs were ordered from longest to shortest and run as a pseudomolecule. Annotations were completed in about 3 weeks. An email was received with an access link to Manatee, http://manatee.igs.umaryland.edu/, a user name, password, and list of database names. See Figure 12 for an image of the Manatee Login page.

After logging in, using each database name to access the different isolate annotations, the FASTA protein files and GenBank files were downloaded from a link at the bottom of the page (Figure 13).
Figure 12: Screenshot of Manatee Login page at http://manatee.igs.umaryland.edu/
Figure 13: Screenshot of Manatee Database
Chapter 6: Comparison of Annotation Pipelines

Using Python to Combine Annotation Data

After the FASTA nucleotide files, containing the assembled contigs, were submitted to the four other annotation pipelines, RAST, BASys, XBase, and Manatee, protein predictions were collected in FASTA amino acid format. A Python script named compare_pipelines_faa_03.19.14.py was written and used to compare the predictions (See supplement 1). The program works on annotation output for one isolate at a time and, at present, the script must be manually edited to change which isolate is analyzed. For each annotation output, the FASTA file is parsed and dictionaries are created with the protein translations as the key and the product name(s) as the value. So for example, when you want to look up the name of a protein, you can give the program the amino acid sequence and retrieve product names.

Once all proteins are obtained from each annotation pipeline output, a master list of sequences is made. The next step is to find amino acid sequences that overlap; this takes into account sequences that were predicted with different start sites but, that are otherwise the same annotation. For example, for strain ACC19a, between all five annotation pipelines, there were two overlapping sequences that coded for the same protein:

- MDNKILIDEIVNWLKMQVKSSNSKGLLVGISGGIDSVAVANLIKLACPNNSLGVILPINSSENSVEDANLLVKQCKSSLTVDISEEHSKLFESMEKLNIGLYKDEYSRMTDANLRARLRMSTLYAIANNLGYMVVGTDNADETYTGYFTKYGDGGVDILPLKKIFKSDVYEMGKILGWPSILSKAPSADLWENQTEDEAEVGVSYDSIEKYMREGEKAKVSEKDEKISNLHKKSEHKRNLPCCFEI
- MVVGTDNADETYTGYFTKYGDGGVDILPLKKIFKSDVYEMGKILGWPSILSKAPSADLWENQTEDEAEVGVSYDSIEKYMREGEKAKVSEKDEKISNLHKKSEHKRNLPCCFEI
The output for this protein is shown in Figure 14. A module that was found online was incorporated into the script and then used to search the master list of proteins for overlapping sequences (https://neil.fraser.name/news/2010/11/04/).

A new master list is created that contains list of overlapping amino acid sequences. For sequences that are unique and have no overlapping sequence, the second value in the list is entered as a ‘None;’ see Figure 15 for an example. Lastly, for each annotation pipeline, each annotation, including those predictions that have multiple start sites, is reviewed and an array is created.
Figure 14: Example output for an NAD synthetase found in the genome of strain ACC19a

Figure 15: Example output for a transposase found in the genome of strain ACC19a
In Table 5 and Figure 16 you can see the number of proteins found for each different combination of annotation pipeline. Most proteins were predicted by all annotation methods.

<table>
<thead>
<tr>
<th></th>
<th>ACC19a</th>
<th>CM2</th>
<th>CM5</th>
<th>OBRC8</th>
<th>AS15</th>
</tr>
</thead>
<tbody>
<tr>
<td>all annotation pipelines</td>
<td>1545</td>
<td>1430</td>
<td>1545</td>
<td>1596</td>
<td>1604</td>
</tr>
<tr>
<td>not Manatee</td>
<td>98</td>
<td>67</td>
<td>97</td>
<td>84</td>
<td>74</td>
</tr>
<tr>
<td>not XBASE</td>
<td>276</td>
<td>214</td>
<td>263</td>
<td>260</td>
<td>316</td>
</tr>
<tr>
<td>not Manatee and XBASE</td>
<td>143</td>
<td>100</td>
<td>136</td>
<td>127</td>
<td>110</td>
</tr>
<tr>
<td>not BROAD/JCVI</td>
<td>13</td>
<td>9</td>
<td>19</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>not BROAD/JCVI and Manatee</td>
<td>9</td>
<td>8</td>
<td>9</td>
<td>10</td>
<td>12</td>
</tr>
<tr>
<td>not BROAD/JCVI and XBASE</td>
<td>31</td>
<td>17</td>
<td>30</td>
<td>5</td>
<td>2</td>
</tr>
<tr>
<td>RAST and BASYS only</td>
<td>16</td>
<td>9</td>
<td>20</td>
<td>9</td>
<td>10</td>
</tr>
<tr>
<td>not BASYS</td>
<td>8</td>
<td>7</td>
<td>9</td>
<td>18</td>
<td>16</td>
</tr>
<tr>
<td>not BASYS and Manatee</td>
<td>12</td>
<td>16</td>
<td>13</td>
<td>19</td>
<td>19</td>
</tr>
<tr>
<td>not BASYS and XBASE</td>
<td>12</td>
<td>6</td>
<td>17</td>
<td>21</td>
<td>23</td>
</tr>
<tr>
<td>RAST and BROAD/JCVI only</td>
<td>53</td>
<td>51</td>
<td>61</td>
<td>73</td>
<td>49</td>
</tr>
<tr>
<td>not BASYS and BROAD/JCVI</td>
<td>5</td>
<td>3</td>
<td>8</td>
<td>4</td>
<td>0</td>
</tr>
<tr>
<td>RAST and XBASE only</td>
<td>6</td>
<td>4</td>
<td>8</td>
<td>3</td>
<td>4</td>
</tr>
<tr>
<td>RAST and Manatee only</td>
<td>6</td>
<td>7</td>
<td>9</td>
<td>3</td>
<td>3</td>
</tr>
<tr>
<td>RAST only</td>
<td>48</td>
<td>30</td>
<td>61</td>
<td>27</td>
<td>24</td>
</tr>
<tr>
<td>not RAST</td>
<td>12</td>
<td>9</td>
<td>12</td>
<td>5</td>
<td>5</td>
</tr>
<tr>
<td>not RAST and Manatee</td>
<td>5</td>
<td>7</td>
<td>6</td>
<td>2</td>
<td>0</td>
</tr>
<tr>
<td>not RAST and XBASE</td>
<td>14</td>
<td>5</td>
<td>20</td>
<td>14</td>
<td>20</td>
</tr>
<tr>
<td>BASYS and BROAD/JCVI only</td>
<td>15</td>
<td>5</td>
<td>19</td>
<td>10</td>
<td>10</td>
</tr>
<tr>
<td>not RAST and BROAD/JCVI</td>
<td>19</td>
<td>25</td>
<td>26</td>
<td>22</td>
<td>15</td>
</tr>
<tr>
<td>BASYS and XBASE only</td>
<td>11</td>
<td>15</td>
<td>13</td>
<td>18</td>
<td>8</td>
</tr>
<tr>
<td>BASYS and Manatee only</td>
<td>48</td>
<td>27</td>
<td>79</td>
<td>60</td>
<td>53</td>
</tr>
<tr>
<td>BASYS only</td>
<td>94</td>
<td>65</td>
<td>106</td>
<td>92</td>
<td>123</td>
</tr>
<tr>
<td>not RAST and BASYS</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>BROAD/JCVI and XBASE only</td>
<td>1</td>
<td>2</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>BROAD/JCVI and Manatee only</td>
<td>6</td>
<td>4</td>
<td>4</td>
<td>13</td>
<td>12</td>
</tr>
<tr>
<td>BROAD/JCVI only</td>
<td>15</td>
<td>15</td>
<td>25</td>
<td>7</td>
<td>9</td>
</tr>
<tr>
<td>XBASE and Manatee only</td>
<td>118</td>
<td>83</td>
<td>114</td>
<td>105</td>
<td>85</td>
</tr>
<tr>
<td>XBASE only</td>
<td>59</td>
<td>52</td>
<td>66</td>
<td>73</td>
<td>35</td>
</tr>
<tr>
<td>Manatee only</td>
<td>51</td>
<td>43</td>
<td>52</td>
<td>35</td>
<td>39</td>
</tr>
</tbody>
</table>

Table 5: Results of protein predictions made by the annotation pipelines
Figure 16: Chart showing the number of proteins predicted by different combinations of annotation pipelines. For example, the first set of bars represents proteins that were predicted by all annotation pipelines.
Chapter 7: Using Annotations to make Predictions about Metabolic Pathways

1. RAST Subsystem and Pathway Tool Predictions

In order to make predictions about metabolic pathways, a combination of RAST subsystem predictions and Pathway Tools metabolic pathway predictions was used. For example, carbohydrate utilization and amino acid biosynthesis was accessed.

2. Carbohydrate Utilization

Based on the results of the API20A test strips, all five strains produced acid in media containing glucose, maltose and sucrose, but not lactose, arabinose, cellobiose, mannose, melezitose, raffinose, rhamnose, trehalose, xylose, glycerol, mannitol, salicin, and sorbitol. It was hypothesized that these results could be supported by the genomic data and RAST annotations. For example, for all five isolates, genes were found that belonged to the “sucrose utilization” subsystem in RAST. In Pathway Tools version 17.5, for all five isolates, complete “sucrose degradation III (sucrose invertase)” pathways were found. As an example, the D-maltose utilization pathway will be discussed in more detail.

D-Maltose Utilization

D-maltose is the disaccharide subunit of starch and glycogen and it is abundant in the human diet. It would make sense that the oral isolates have the ability to degrade maltose and use it as a carbon source. In Pathway Tools, maltose is part of multiple pathways, including the starch degradation I through V pathways and the glycogen degradation I pathway. In RAST, for all five isolates the “maltose and maltodextrin utilization” subsystem was present, suggesting that enough genes were found for the pathway to be present. In all five isolates, the following
proteins were found: a multiple sugar ABC transporter (MsmFG), a multiple sugar ABC transporter ATP-binding protein, a 4-α-glucanotransferase, and two glycogen phosphorylases. MsmFG is found in the cell membrane and may also be responsible for transporting maltose into the cell. In other organisms, such as Streptococcus mutans (ATCC 700610), MsmFG is involved in the uptake of melibiose, raffinose, and isomaltotriose. The other multiple sugar ABC transporter ATP-binding protein may also be able to transport maltose. 4-α-glucanotransferase (amylomaltase, EC 2.4.1.25) directly breaks down D-maltose and it is found in the starch degradation V pathway and the glycogen degradation I and III pathways.

ACC19a, CM2, CM5, and OBRC8 also have a maltose operon transcriptional repressor MalR, LacI Family and two glycogen phosphorylases. Contrary to the name, the role of MalR is to activate maltose metabolism by stimulating the maltose translocation system; it is an exception within the family of LacI-GalR proteins (Andersson & Rådström, 2002). In the glycogen degradation III pathway, a glycogen phosphorylase and 4-α-glucanotransferase are required, among other proteins. ACC19a, CM2, and CM5 have a neopullulanase and AS15 has a pullulanase. Pullulanases are debranching enzymes that hydrolyzes the α-1,6 glycosydic linkages in starch, amylpectin, pullulan, and other oligosaccharides (Hii, Tan, Ling, & Ariff, 2012).

Lastly, CM2, OBRC8, and AS15 have a cytoplasmic α-amylase. The starch degradation V pathway begins with the breakdown of starch into D-maltose by an α-amylase. As mentioned before, the absence of this gene can be explained by the presence of amylases in the saliva, removing the need to have this gene: a trait shared by ACC19a and CM5.

Theoretically, CM2, OBRC8 and AS15 are capable of breaking down starch outside of the cell, while ACC19a and CM5 depend on salivary amylases. Pullulanases are also used to break down starch.
D-Lactose Utilization

Since the tests supplemented with lactose, arabinose, cellobiose, mannose, melezitose, raffinose, rhamnose, trehalose, xylose, glycerol, mannitol, salicin, or sorbitol did not have a positive result in the API20A test strips, it was hypothesized that the isolates did not have complete pathways for the utilization of these carbohydrates. As an example, D-lactose utilization will be discussed in more detail. According to RAST, all five isolates also have genes belonging to the “Lactose and Galactose Uptake and Utilization” subsystem, but the genes annotated only belonged to galactose utilization. No genes were found for lactose uptake and utilization. In Pathway Tools, there are 3 instances of lactose degradation: 1.) lactose and galactose degradation I, 2.) lactose degradation II, and 3.) lactose degradation III. A species comparison for the isolates was done for each of these pathways and they were marked as not present in the isolates. For comparison, since that lactose and galactose pathways are connected, the galactose degradation I (Leloir pathway) was also analyzed and this pathway was present in all the isolates. Overall the isolates do not have the ability to utilize lactose, but do have the ability to use galactose.
3. Amino Acid Biosynthesis

Based on publically-available genomes on RefSeq, amino acid biosynthesis phenotype predictions for all of the isolates were made on the Integrated Microbial Genomes (IMG) website (Markowitz et al., 2012). The site is accessible at https://img.jgi.doe.gov/cgi-bin/w/main.cgi. An IMG account is required to use the site and it can be requested at https://img.jgi.doe.gov/cgi-bin/submit/request.cgi. In Table 6 you can see the predictions made by IMG. The following section goes through each prediction and determines whether or not the predictions were accurate.

<table>
<thead>
<tr>
<th>Amino Acid</th>
<th>ACC19a</th>
<th>CM2</th>
<th>CM5</th>
<th>OBRC8</th>
<th>AS15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alanine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Arginine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Asparagine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Aspartate</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Cysteine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glutamate</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Glutamine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Glycine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Histidine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Isoleucine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Leucine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Lysine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Methionine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Phenylalanine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Proline</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Serine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Threonine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Tryptophan</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Tyrosine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Valine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>L-Selenocysteine</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 6: Table of amino acid biosynthesis predictions found on the IMG website
The Pyruvate Family

Alanine Biosynthesis

According to IMG, the five isolates are alanine-auxotrophs; five pathways were not asserted: L-alanine synthesis from 1.) D-alanine, 2.) L-glutamate, 3.) L-serine, 4.) L-valine, and 5.) reductive amination of pyruvate. In RAST, there is an “alanine biosynthesis” subsystem that was predicted for all of the isolates, specifically variant C: “alanine biosynthesis III pathway (cysteine desulfurase) and alanine racemase.’ This variant pathway requires at least a cysteine desulfurase and an alanine racemase; a branched-chain amino acid (BCAA) aminotransferase may or may not be present. Based on the RAST data, all the isolates have at least one gene for the following proteins: 1.) a BCAA aminotransferase (EC 2.6.1.42), 2.) an alanine racemase (EC 5.1.1.1), 3.) a cysteine desulfurase (EC 2.8.1.7), and 4.) an iron-sulfur cluster regulator IscR.

Going through the list of IMG pathways, first, alanine racemase converts L-alanine into D-alanine, and vice versa; each isolate has one gene for this enzyme: this pathway should have been asserted. L-glutamate can be converted into L-alanine using a glutamate-pyruvate aminotransferase\(^2\) (EC 2.6.1.2, aka alanine transaminase) and pyruvate, but none of the isolates have this enzyme. L-serine can be converted into L-alanine using a serine-pyruvate aminotransferase (EC 2.6.1.51) and pyruvate. While not predicted as part of the “alanine biosynthesis” subsystem, this protein was found in isolates ACC19a, CM2, CM5, and OBRC8, but not AS15 or *Eubacterium yurii margaretiae*. This pathway should have been asserted in IMG for those isolates.

L-valine can be converted into L-alanine using a valine-pyruvate transaminase (EC 2.6.1.66) and pyruvate, but the isolates also do not have a gene for this protein. The last 3 enzymes

\(^2\) Transaminase and aminotransferase are interchangeable names
mentioned all require pyridoxal 5’-phosphate\(^3\) (PLP) as a cofactor. Lastly, pyruvate can be converted into L-alanine using an L-alanine dehydrogenase (EC 1.4.1.1), pyruvate, ammonium, and NADH; the isolates also do not have this enzyme.

Branched-chain amino acid aminotransferases catalyze the reaction between 2-oxoglutarate and BCAAs to form L-glutamate and another byproduct, depending on the BCAA. This could feed into the L-alanine biosynthesis pathway, but as mentioned before, the isolates lack the necessary alanine transaminase.

Cysteine desulfurase is a PLP-dependent enzyme that can convert L-cysteine into L-alanine and sulfane sulfur through the formations of a cysteine persulfide intermediate (Mihara & Esaki, 2002). In addition to this, cysteine desulfurase is involved in the synthesis of numerous other molecules, such as iron-sulfur clusters, thiamine, biotin, etc.... Each isolate has at least 4 genes for this enzyme, including one specifically annotated as belonging to the SufS subfamily. Therefore, if L-cysteine is present, then the isolates hypothetically could synthesize L-alanine. Overall, it is hypothesized that the isolates are alanine-prototrophic.

---

\(^3\) Pyridoxal 5’-phosphate is the active form of vitamin B\(_6\).
Valine and Leucine Biosynthesis

Valine Biosynthesis

According to IMG, the five isolates are valine-auxotrophs; one pathway was not asserted: L-valine synthesis from pyruvate. According to PathwayTools, the following proteins are needed to convert pyruvate into L-valine (superpathway of leucine, valine, and isoleucine biosynthesis):

- Acetolactate synthase IlvGM (EC 2.2.1.6)
- Ketol-acid reductoisomerase IlvC (EC 1.1.1.86)
- Dihydroxyacid dehydratase IlvD (EC 4.2.1.9) and
- Branched-chain amino acid aminotransferase (EC 2.6.1.42)

According to the RAST predictions, the five isolates and Eubacterium yurii margaretiae, have genes for all these proteins, but Filifactor alocis does not. It is hypothesized that the isolates are valine-prototrophic and this trait can distinguish them from Filifactor species.

Leucine Biosynthesis

According to IMG, the five isolates are leucine-auxotrophs; one pathway was uncertain or not asserted, 2-oxoisovalerate (3-methyl-2-oxobutanoate) synthesis from pyruvate, and one pathway was not asserted, L-leucine synthesis from 2-oxoisovalerate. These two pathways are part of one pathway, starting from pyruvate and ending in L-leucine. According to PathwayTools, the following proteins are needed to convert pyruvate into L-leucine:

- Acetolactate synthase IlvGM (EC 2.2.1.6)  *also part of valine synthesis*
- Ketol-acid reductoisomerase IlvC (EC 1.1.1.86)  *also part of valine synthesis*
- Dihydroxy-acid dehydratase IlvD (EC 4.2.1.9)  *also part of valine synthesis*
- 2-isopropylmalate synthase LeuA (EC 2.3.3.13)
- 3-isopropylmalate dehydratase LeuCD (EC 4.2.1.33)
- 3-isopropylmalate dehydrogenase LeuB (EC 1.1.1.85) and
- Branched-chain amino acid aminotransferase (EC 2.6.1.42)  *multiple pathways*

According to RAST, the “Branched-Chain Amino Acid Biosynthesis” subsystem and the “Leucine biosynthesis” subsystem were predicted for isolates CM2, OBRC8, and AS15 and
*Eubacterium yurii margaretiæ.* The first subsystem contains all the genes for the proteins mentioned above. For ACC19a and CM5, the subsystem was not predicted; a few genes, specific for leucine biosynthesis were missing —- 2-isopropylmalate synthase LeuA (EC 2.3.3.13) and 3-isopropylmalate dehydratase LeuC D (EC 4.2.1.33) —- while some were present but not assigned to this subsystem. It is hypothesized that CM2, OBRC8, and AS15 are leucine-prototrophs and ACC19a and CM5 are leucine-auxotrophs.

**The 3-phosphoglycerate Family**

**Serine and Glycine Biosynthesis**

*Serine Biosynthesis*

According to IMG, the five isolates are serine-auxotrophs; the only pathway that was not asserted was the L-serine synthesis from 3-phosphoglycerate pathway. According to PathwayTools, the following proteins are needed to convert 3-phospho-D-glycerate into L-serine and L-glycine:

- D-3-phosphoglycerate dehydrogenase SerA (EC 1.1.1.95)
- 3-phosphoserine aminotransferase SerC (EC 2.6.1.52)
- Phosphoserine phosphatase SerB (EC 3.1.3.3) *final step for L-serine*
- hydroxymethyltransferase GlyA (EC 2.1.2.1) *to synthesize glycine from L-serine*

In RAST, there is a “Serine Biosynthesis” subsystem, and isolate AS15 and *Eubacterium yurii margaretiæ* each have four genes predicted to be a part of this subsystem: three genes for D-3-phosphoglycerate dehydrogenase SerA (EC 1.1.1.95) and one for serine hydroxymethyltransferase GlyA (EC 2.1.2.1). Initially, isolates ACC19a, CM2, CM5, and OBRC8 did not have genes belonging to this subsystem, but a manual search found that these isolates also have multiple genes for D-3-phosphoglycerate dehydrogenase. From the data, it appears that all five isolates are auxotrophic for L-serine, as IMG predicted, because they lack at least two of the proteins that are a part of the pathway.
**Glycine Biosynthesis**

According to IMG, the five isolates are glycine-prototrophs; there are four different pathways used for this prediction. For isolates ACC19a, CM2, CM5, and OBRC8, three pathways were not asserted: 1.) L-serine conversion to glycine and methylene-THF, 2.) L-threonine conversion to glycine and acetyl-CoA, and 3.) Glycine synthesis from L-alanine by transamination; and one was asserted, L-threonine conversion to glycine and acetaldehyde. For isolate AS15, the presence of the L-serine conversion to glycine and methylene-THF pathway was unknown. For the one pathway that was asserted in all the isolates the following protein is required: L-aldolase GlyB (EC 4.1.2.5). The gene for this protein was present in all of the isolates and this supports the IMG prediction.

Only isolate AS15 and *Eubacterium yurii margaretiae* have a gene for serine hydroxymethyltransferase GlyA (EC 2.1.2.1), which converts L-serine and THF into glycine and methylene-THF; this pathway should have been asserted in AS15. For the L-threonine conversion to glycine and acetyl-CoA pathway, the following proteins are required: threonine dehydrogenase (EC 1.1.1.103) and 2-amino-3-ketobutyrate (EC 2.3.1.29). Isolate AS15 and *Eubacterium yurii margaretiae* only have a gene for threonine dehydrogenase and the other isolates do not. None of the isolates have a gene for 2-amino-3-ketobutyrate. The glycine synthesis from L-alanine by transamination pathway is actually found in eukaryotic organisms, but the following protein is needed: alanine--glyoxylate transaminase (EC 2.6.1.44). None of the isolates have this gene.

**Cysteine Biosynthesis**
IMG does not make a prediction about cysteine metabolism for the isolates. According to Pathway Tools, L-serine can be converted into L-cysteine using two enzymes: serine acetyltransferase CysE (EC 2.3.1.30) and cysteine synthase CysK (EC 2.5.1.47). According to RAST, all the isolates have genes for both proteins and may be cysteine-prototrophic.

The Ribose 5-phosphate Family
Histidine Biosynthesis

According to IMG, the five isolates are histidine-auxotrophs; the one pathway needed to synthesis histidine was not asserted. Histidine synthesis is fed from the pentose phosphate pathway, which produces 5-phospho-α-D-ribose-1-diphosphate (PRPP). According to Pathway Tools, the following proteins are required to convert PRPP to L-histidine:

- ATP phosphoribosyltransferase HisA (EC 2.4.2.17)
- Phosphoribosyl-ATP pyrophosphatase (EC 3.6.1.31)
- Phosphoribosyl-AMP cyclohydrolase (EC 3.5.4.19)
- N-(5’-phospho-D-ribosylformimino)-5-amino-1-(5”-phosphoribosyl)-4-imidazolecarboxamide isomerase (EC 5.3.1.16)
- Imidazole glycerol phosphate synthase (EC 2.4.2.-)
- Imidazoleglycerol phosphate dehydratase (EC 4.2.1.19)
- Histidinol-phosphate aminotransferase (EC 2.6.1.9)
- Histidinol phosphate phosphatase (EC 3.1.3.15) and
- Histidinol/Histodinal dehydrogenase (EC 1.1.1.23)

In RAST, there is a “histidine biosynthesis” subsystem, but it was not predicted for any of the isolates. Further manual analysis showed that no genes for any of these proteins were present in the isolates.

The Oxaoloacetate/Aspartate Family
Aspartate

According to IMG, the isolates are L-aspartate-auxotrophic; three pathways were not asserted: L-aspartate synthesis from 1.) L-glutamate, 2.) asparagine deamidation, and 3.) D-aspartate. In
PathwayTools, there is basically one pathway for producing aspartate. An aspartate aminotransferase (EC 2.6.1.1) is used to convert oxaloacetate and L-glutamate into L-aspartate and 2-oxoglutarate. The isolates all have multiple genes for this protein and this pathway in IMG should have been asserted. For asparagine deamidation, an asparaginase (EC 3.5.1.1) is required, and none of the isolates have a gene for this protein. In order for D-aspartate to be converted into L-asparate, an aspartate racemase (EC 5.1.1.13) is required, but none of the isolates have a gene for this protein. Based on the presence of an aspartate aminotransferase, the pathway should have been predicted.

**Lysine Biosynthesis**

According to IMG, the five isolates are lysine-auxotrophs; five pathways were not asserted: L-lysine synthesis from 1.) acelated intermediates, 2.) succinylated intermediates, 3.) aminoadipate pathway (prokaryotic), 4.) Diaminopimelic (DAP) aminotransferase, and 5.) DAP dehydrogenase. According to PathwayTools, there are six instances of lysine biosynthesis, named lysine biosynthesis I through VI. Instances I through III and VI start with the conversion of aspartate, and IV and V start with the conversion of 2-oxoglutarate. In RAST, the isolates all have genes that are part of the “Lysine Biosynthesis DAP Pathway, GJO scratch” subsystem. In addition, isolate AS15 has additional genes that are a part of the “Lysine Biosynthesis DAP pathway” subsystem. This suggests that the isolates can synthesize lysine from L-aspartate and pyruvate through DAP. The DAP pathway is also important in the synthesis of cell wall peptidoglycan. Based this information and some of the proteins found the isolates, the “L-lysine synthesis with DAP dehydrogenase pathway” in IMG was further investigated. The following enzymes are required for this pathway:

- Aspartate kinase (EC 2.7.1.4)
- Aspartate semialdehyde dehydrogenase (EC 1.2.1.11)
Based on the proteins found for the isolates and the proteins required in the pathway being investigated, it suggests that isolate AS15 and *Eubacterium yurii margaretiae* may be able to synthesize lysine. The other isolates are missing three key enzymes and are auxotrophic.

**Asparagine**

According to IMG, the isolates are L-asparagine-prototrophic; one pathway was asserted and two were not. The pathway that was asserted was “asparagine synthesis using ammonium;” an aspartate-ammonia ligase (EC 6.3.1.1) is required. This enzyme was found in all isolates and is responsible for catalyzing the reaction: $\text{ATP} + \text{L-aspartate} + \text{NH}_3 \rightarrow \text{AMP} + \text{diphosphate} + \text{L-asparagine}$. This pathway was correctly asserted in IMG. The non-asserted pathways involved synthesis from L-glutamine or a tRNA-dependent synthesis. For the first pathway, an asparagine synthase (EC 6.3.5.4) is required; none of the isolates have a gene for this protein. For the second, an aspartate-tRNA ligase (EC 6.1.1.12) is required; none of the isolates have a gene for this protein either.

**Methionine Biosynthesis**

IMG does not make a prediction about methionine metabolism for the isolates. Pathway Tools shows five instances of methionine biosynthesis: 1.) methionine biosynthesis I through III, 2.) methionine biosynthesis by sulphonylation, and 3.) methionine biosynthesis by transsulfuration. According to RAST, there is a “Methionine Biosynthesis” subsystem and there are two ways in which methionine can be synthesized: via transsulfuration from cysteine or sulphonylation from
inorganic sulfur. All the isolates, have this subsystem predicted and have at least 15 different proteins for it. For isolates ACC19a, CM2, and AS15, the 4th RAST variant of this pathway was predicted; this pathway is similar to the one found in *Streptococcus spp.*, with a full sulphydrylation pathway via O-succinyl-L-homoserine and with methionine transport. Unexpectedly, isolate CM5 had the 8th variant of this pathway predicted, which is similar to the pathway found in *Enterococcus faecalis*; there is no de novo pathway but there is a methionine transporter; it was expected that OBRC8 would have the same pathway variant as CM2 because of the proteins they have in common. A variant type was not predicted for isolate OBRC8.

According to Pathway Tool, to synthesize methionine via sulphydrylation, starting with L-homoserine and hydrogen sulfide, the following enzymes are required:

- Homoserine O-acetyltransferase (EC 2.3.1.31),
- O-acetylhomoserine sulphydrylase (EC 2.5.1.49), and
- a Methionine synthase: either 5-methyltetrahydrofolate-homocysteine methyltransferase (EC 2.1.1.13) or 5-methyltetrahydropteroyltriglutamate--homocysteine S-methyltransferase (2.1.1.14)

Based on the genes that the isolates have, they do not follow the sulphydrylation pathway exactly. The presence of a homoserine dehydrogenase (EC 1.1.1.3) suggests that the isolates can produce L-homoserine from L-aspartate-semialdehyde. L-homoserine is required in methionine biosynthesis both by sulphydrylation and transsulfuration. For sulphydrylation, the next gene required is homoserine O-acetyltransferase (EC 2.3.1.31), which converts L-homoserine into O-acetyl-L-homoserine; the isolates do not appear to have this gene. Instead, the isolates have a gene for an homoserine O-succinyltransferase (EC 2.3.1.46), which converts L-homoserine into O-succinyl-L-homoserine.

Next, the isolates have a gene that is annotated in two ways, either as an O-acetylhomoserine sulphydrylase (EC 2.5.1.49) or an O-succinylhomoserine sulphydrylase (EC 2.5.1.48). O-acetylhomoserine sulphydrylase can convert O-acetyl-L-homoserine (which the isolates may not
be able to produce) and methanethiol directly into L-methionine and acetate, according to ExPASy. In PathwayTools, this enzyme is shown to convert O-acetyl-L-homoserine and hydrogen sulfide into L-homocysteine. O-succinylhomoserine sulphydrylase, on the other hand, can convert O-succinyl-L-homoserine (which the isolates can produce) and cysteine into L-cystathionine and succinate. The latter enzyme, along with the presence of a cysteine synthase, suggests that the isolates use transsulfuration. However, with transsulfuration, there is an additional step of converting L-cystathionine into L-homocysteine, using a cystathionine-β-lyase (EC 4.4.1.8); the isolates do not have this enzyme.

If L-homocysteine can be produced, one final step is necessary to synthesize L-methionine from it; a methionine synthase (EC 2.1.1.13), which all the isolates have. Overall, regardless of which pathway could be used, there are enzymes missing from either one that are necessary to synthesis L-methionine.

**Threonine**

IMG does not make a prediction about methionine metabolism for the isolates. Pathway Tools shows a “threonine biosynthesis” pathway that requires the following enzymes:

- Aspartate transaminase (EC 2.6.1.1, aka aspartate aminotransferase)
- Aspartate kinase (EC 2.7.2.4)
- Aspartate semialdehyde dehydrogenase (EC 1.2.1.11)
- Homoserine dehydrogenase (EC 1.1.1.3)
- Homoserine kinase (EC 2.7.1.39) and
- Threonine synthase (EC 4.2.3.1)

All of the isolates have genes for these six proteins and therefore may be threonine-prototrophic.

**Isoleucine Biosynthesis**
According to IMG, the five isolates are isoleucine-auxotrophs; three pathways were either unknown or not asserted: 1.) L-leucine synthesis from 2-oxobutanoate and pyruvate, 2.) L-threonine deamination to 2-oxobutanoate, and 3.) 2-oxobutanoate synthesis via R-citramalate.

According to PathwayTools, the following proteins are needed to convert L-threonine into L-isoleucine:

- Threonine deaminase IlvA (EC 4.3.1.19)
- Acetohydroxybutanoate synthase IlvGM (EC 2.2.1.6, aka acetolactate synthase)
- Ketol-acid reductoisomerase IlvC (EC 1.1.1.86)
- 2,3-dihydroxy-3-methylvaerate hydrolase IlvD (EC 4.2.1.9, aka dihydroxy-acid dehydratase) and
- Branched-chain amino acid aminotransferase (EC 2.6.1.42)

In RAST, isoleucine biosynthesis proteins are included in the “Branched Chain amino acid biosynthesis” subsystem. Isolates CM2, OBRC8, AS15 and Eubacterium yurii margaretiae each have genes for all these proteins, while ACC19a and CM5 do not. It is hypothesized that CM2, OBRC8, and AS15 are isoleucine-prototrophic and ACC19a and CM5 are isoleucine-auxotrophic.

The Erythrose 4-phosphate and Phosphoenolpyruvate Family: Aromatic Amino Acids

Phenylalanine and Tyrosine

According to IMG, the five isolates are phenylalanine-auxotrophs; four pathways were not asserted: 1.) chorismate synthesis, 2.) archaeal chorismate synthesis, 3.) L-phenylalanine synthesis via phenylpyruvate, and 4.) L-phenylalanine synthesis via L-arogenate. Also according to IMG, the five isolates are tyrosine-auxotrophs; four pathways were not asserted: 1.) chorismate synthesis, 2.) archaeal chorismate synthesis, 3.) L-tyrosine synthesis via 4-hydroxyphenylpyruvate, and 4.) L-tyrosine synthesis via L-arogenate.
In Pathway Tools, there are two pathways for phenylalanine biosynthesis in bacteria. All proteins that are a part of these pathways are in the following list and the Roman numerals represent which pathway they are in:

- Chorismate mutase (EC 5.4.99.5) I
- Prephenate dehydratase (EC 4.2.1.51) I
- Phenylalanine transaminase (EC 2.6.1.57) I
- Prephenate aminotransferase (EC 2.6.1.79) II
- Arogenate dehydratase (EC 4.2.1.91) II

In Pathway Tools, there are four pathways for tyrosine biosynthesis. The first four start from chorismate and the last one starts from L-phenylalanine. All proteins that are a part of these pathways are in the following list and the Roman numerals represent what pathway they are in:

- Chorismate mutase (EC 5.4.99.5) I II III
- Prephenate dehydrogenase (EC 1.3.1.12) I
- Tyrosine aminotransferase (EC 2.6.1.57) I
- Prephenate aminotransferase (EC 2.6.1.79) II III
- Arogenate dehydrogenase (1.3.1.78) II
- Arogenate dehydrogenase (1.3.1.43) III
- Phenylalanine hydroxylase (1.14.16.1) IV

In RAST there is a “Phenylalanine and Tyrosine Branches from Chorismate” subsystem, and phenylalanine biosynthesis was also not predicted for the isolates. However, each of the isolates has at least two enzymes belonging to this subsystem: 1.) one annotated as “Prephenate and/or arogenate dehydrogenase (unknown specificity) (EC 1.3.1.12) (EC 1.3.1.43)” and 2.) prephenate dehydratase (EC 4.2.1.51). The enzyme of uncertain function can either convert prephenate and NAD+ into 4-hydroxyphenylpyruvate, CO₂, and NADH, or it can convert L-arogenate and NAD+ into L-tyrosine, NADH, and CO₂. Isolate AS15 and Eubacterium yurii margaretiae also have a gene for chorismate mutase I (EC 5.4.99.5). Overall, the isolates do not have complete pathway for phenylalanine and tyrosine biosynthesis.
Tryptophan

According to IMG, the five isolates are tryptophan-auxotrophs; three pathways were not asserted: 1.) chorismate synthesis, 2.) archaeal chorismate synthesis, and 3.) L-tryptophan synthesis. The first two pathways have been discussed in the phenylalanine and tyrosine section.

In Pathway Tools, there is one pathway for tryptophan biosynthesis and the following enzymes are required:

- Anthranilate synthase (EC 4.1.3.27)
- Anthranilate phosphoribosyltransferase (EC 2.4.2.18)
- Phosphoribosylanthranilate isomerase (EC 5.3.1.24)
- Indole-3-glycerol phosphate synthase (EC 4.1.1.48)
- Indoleglycerol phosphate synthase (EC 4.1.2.8) and
- Tryptophan synthase α and β chains (EC 4.2.1.122)

According to Pathway Tools, based on the RAST annotations, all the isolates have the β chain of the tryptophan synthase protein. All the isolates, except for ACC19a, are missing the other four enzymes. Isolate ACC19a also has a phosphoribosylanthranilate isomerase-like protein. In RAST, the “tryptophan biosynthesis” subsystem was not predicted for the isolates. Upon further investigation, the tryptophan synthase beta chain (EC 4.2.1.20) gene was assigned to a different subsystem: “Chorismate: Intermediate for synthesis of Tryptophan, PAPA antibiotics, PABA, 3-hydroxyanthranilate.” A search for the word tryptophan in the Role column brought up another protein for isolates ACC19a, CM2, CM5, and OBRC8: Substrate-specific component TrpP of tryptophan ECF transporter. Based on these findings, it appears that the isolates are able to transport tryptophan, but are unable to synthesize it.

The α-ketoglutarate Family

Glutamate
According to IMG, the isolates are L-glutamate-prototrophic. In PathwayTools, there are many pathways that lead to the production of L-glutamate. In one pathway, a two-subunit protein, glutamate synthase (EC 1.4.1.13), is required to convert L-glutamine, one 2-oxoglutarate, and NADPH, into two L-glutamates and NADP+ (“glutamate biosynthesis I”). In another pathway, glutamate dehydrogenase (EC 1.4.1.3) is required to convert two 2-oxoglutarates, NAD(P)H, H+, and ammonium into L-glutamate, NAD(P)+, and water (“glutamate biosynthesis II”). All the isolates have genes for both the alpha and beta subunits of glutamate synthase, supporting the assertion in IMG. They also have a gene for an NAD-specific glutamate dehydrogenase (EC 1.4.1.2), which is similar to glutamate dehydrogenase (EC 1.4.1.3), but it uses NADH instead of NADPH. This suggests that in IMG, the “L-glutamate synthesis from L-glutamine and 2-oxoglutarate” pathway should have been asserted for the isolates. The isolates also have a glutamate racemase (EC 5.1.1.3), which converts L-glutamate to D-glutamate, and vice versa.

Glutamine

In IMG, glutamine metabolism was not available for the isolates. In PathwayTools, there are four instances for glutamine biosynthesis. One of the pathways requires a glutamine synthetase (EC 6.3.1.2), which catalyzes the following reaction: ATP + L-glutamate + NH₃ ↔ ADP + phosphate + L-glutamine. All of the isolates have a gene for this protein. The isolates also have a gene for glutaminase (EC 3.5.1.2), which catalyzes the following reaction: L-glutamine + H(2)O ↔ L-glutamate + NH₃. The isolates appear to be prototrophic for glutamine.

Proline

According to IMG, the five isolates are proline-auxotrophs; two pathways were not asserted: 1.) L-proline synthesis from L-glutamate and 2.) L-proline synthesis via ornithine cyclodeaminase.
In RAST, the “proline biosynthesis” subsystem was not predicted for any of the isolates. All proteins that are a part of these pathways are in the following list and the Roman numerals represent what pathway they are in:

- Glutamate 5-kinase (EC 2.7.2.11) I
- Glutamate-5-semialdehyde dehydrogenase (EC 1.2.1.41) I
  - aka γ-glutamyl phosphate reductase
- Pyrroline-5-carboxylate reductase (EC 1.5.1.2) I
- Ornithine cyclodeaminase (EC 4.3.1.12) II

Pathway I in IMG is the same as the “proline biosynthesis I” pathway in Pathway Tools. According to Pathway Tools and the RAST annotations, only isolate AS15 and *Eubacterium yurii margaretiae* have the enzymes necessary for proline biosynthesis: specifically the enzymes found in pathway I of IMG, so this pathway should have been asserted. Isolates ACC19a, CM2, CM5, and OBRC8 did not have any of the enzymes found in either IMG pathway for proline biosynthesis

Arginine Biosynthesis

According to IMG, the five isolates are arginine-auxotrophs; three pathways were not asserted:

1.) L-ornithine synthesis via N-acetylglutamate, 2.) L-arginine synthesis from L-ornithine, and 3.) L-arginine synthesis via N-acetyl-L-citrulline. These pathways are actually part of a single pathway starting from L-glutamate and ending with L-arginine. The following proteins are required:

- N-acetylglutamate synthase (EC 2.3.1.1)
- Acetylglutamate kinase (EC 2.7.2.8)
- N-acetylglutamylphosphate (EC 1.2.1.38)
- Acetylornithine aminotransferase (EC 2.6.1.11)
- Acetylornithine deacetylase (EC 3.5.1.16)
- Ornithine carbamoyltransferase (EC 2.1.3.3) *to produce carbamoyl-phosphate*
- Argininosuccinate synthase (EC 6.3.4.5) and
- Argininosuccinate lyase (EC 4.3.2.1)
For all of the isolates, only a gene for ornithine carbamoyltransferase was found. This supports the IMG prediction that the isolates are arginine-auxotrophs.

Selenocysteine

According to IMG, the five isolates are selenocysteine-prototrohs; two of three pathways were not asserted: 1.) selenophosphate synthesis and 2.) bacterial selenocysteine synthesis. Understandably, the only pathway that was not asserted was the archael/eukaryotic selenocysteine synthesis pathway. For the first pathway, selenocysteine synthesis, the only required protein is a selenophosphate synthase (EC 2.7.9.3, aka selenide,water dikinase). For the second pathway, a seryl-tRNA synthetase (EC 6.1.1.11) and an L-seryl-tRNA (Sec) selenium transferase (EC 2.9.1.1) are required. The RAST data supports this; for all five isolates, these three proteins were found.
Hypothesis on Amino Acid Biosynthesis

Based on the *in silico* analyses of the amino acid biosynthesis pathways, some differences were found compared to what was predicted for the Refseq data on IMG. The RAST annotations seem to provide different findings. Hypothetically, the differences in isoleucine, leucine, methionine, and proline biosynthesis could be used to distinguish isolates ACC19a and CM5, isolates CM2 and OBRC8, and isolate AS15 (Table 7).

<table>
<thead>
<tr>
<th></th>
<th>ACC19a</th>
<th>CM2</th>
<th>CM5</th>
<th>OBRC8</th>
<th>AS15</th>
</tr>
</thead>
<tbody>
<tr>
<td>Alanine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Arginine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Asparagine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Aspartate</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Cysteine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Glutamate</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Glutamine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Glycine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Histidine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Isoleucine</td>
<td>auxotroph</td>
<td>prototroph</td>
<td>auxotroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Leucine</td>
<td>auxotroph</td>
<td>prototroph</td>
<td>auxotroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Lysine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Methionine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Phenylalanine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Proline</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Serine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Threonine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>Tryptophan</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Tyrosine</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
<td>auxotroph</td>
</tr>
<tr>
<td>Valine</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
<td>prototroph</td>
</tr>
<tr>
<td>L-Selenocysteine</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
<td>+</td>
</tr>
</tbody>
</table>

Table 7: Amino acid biosynthesis predictions based on RAST and Pathway Tools analysis
Effect of Amino Acids on Growth

Methods

To see whether the addition of amino acids could improve growth, medium containing the amino acids that the isolates were auxotrophic for was tested. Due to the fastidious nature of the isolates, it was found that yeast extract is essential for growth and this was also included in the media; the concentration of yeast extract was kept as low as possible, at 2g/L. This experiment was compared to media with 1.) only yeast extract, 2.) media with yeast extract and glucose, and 3.) media with yeast extract, amino acids, and glucose. To make the media, two bottles, labeled A and B, were set up, one with 0.2 g yeast extract and the other with 0.2g yeast extract and 0.2 g D-Glucose. A 50x phosphate buffer solution was made by dissolving 13.8 g of potassium phosphate monobasic (KH$_2$PO$_4$) and 17 g of potassium phosphate dibasic (K$_2$HPO$_4$) into 100mL of water. To bottles A and B, 1mL of the 50x phosphate buffer was added. The solutions were topped off with 98 mL of water. Each of the bottles was divided in half and relabeled, to A1 and A2 (no glucose) and B1 and B2 (with glucose). To bottles A2 and B2, amino acids were added. The 50x Gibco MEM Amino Acids Solution (Catalog # 11130036) was selected because it contained many of the amino acids that the isolates were auxotrophic for; 1mL of this solution was added to bottles A2 and B2. In addition, a 50x solution of L-proline and L-serine was prepared by dissolving 575.5 mg of proline and 525.45 mg of serine in 5mL of water; 1 mL of this solution was also added to bottles A2 and B2. The solutions were aliquoted into anaerobic tubes, capped within an anaerobic chamber, sealed, and autoclaved. Tubes were inoculated with approximately 0.2 mL of freshly-grown culture. OD 600 measurements were taken at time 0, day 1, day 2, day 3, and day 6.
Results

As a preliminary experiment, cultures were grown under four conditions: 1.) media with 2 g/L yeast extract, 2.) media with 2 g/L yeast extract and 2 g/L glucose, 3.) media with 2 g/L yeast extract and amino acids that the isolates were auxotrophic for, and 4.) media with 2 g/L yeast extract, 2 g/L glucose, and amino acids that the isolates were auxotrophic for. For all isolates except CM2, growth was not visible by the naked eye, but for all isolates growth was detectible by spectrophotometer at OD 600. For isolate ACC19a, there was no difference in growth between the four variables (17). Amino acids did not appear to improve growth, nor did it deter it. For isolate CM2, growth was slightly more robust than the other isolates, but the only factor that appeared to enhance growth was glucose (18). For isolates CM5, OBRC8, and AS15, there was no significant improvement in growth for cultures supplemented with glucose or amino acids (19, 20, 21).

Figure 17: Impact of Growth on Isolate ACC19a; y/e: yeast extract; y/e+glu: yeast extract + glucose; y/e+aa: yeast extract + amino acids; y/e+glu+aa: yeast extract + glucose + amino acids
Figure 18: Impact of Growth on Isolate CM2; y/e: yeast extract; y/e+glu: yeast extract + glucose; y/e+aa: yeast extract + amino acids; y/e+glu+aa: yeast extract + glucose + amino acids

Figure 19: Impact of Growth on Isolate CM5; y/e: yeast extract; y/e+glu: yeast extract + glucose; y/e+aa: yeast extract + amino acids; y/e+glu+aa: yeast extract + glucose + amino acids
Figure 20: Impact of Growth on Isolate OBRC8; y/e: yeast extract; y/e+glu: yeast extract + glucose; y/e+aa: yeast extract + amino acids; y/e+glu+aa: yeast extract + glucose + amino acids

Figure 21: Impact of Growth on Isolate AS15; y/e: yeast extract; y/e+glu: yeast extract + glucose; y/e+aa: yeast extract + amino acids; y/e+glu+aa: yeast extract + glucose + amino acids
The simplest conclusion is that supplementing the media with amino acids that the isolates were auxotrophic for does not improve growth. There is another growth factor, found in yeast extract, that the isolates require.

4. Amino Acid Degradation

Based on the amino acid biosynthesis predictions, it is thought that for those amino acids the isolates were auxotrophic for, there would be corresponding pathways leading to the degradation of those amino acids. For example, according to RAST, isolate ACC19a has proteins belonging to the Histidine, Arginine and Ornithine, Methionine, Threonine, and Lysine degradation subsystems, and isolate ACC19a is auxotrophic for histidine, arginine, methionine, and lysine.

Histidine fermentation

During histidine fermentation, urocanate and glutamate are produced by *Clostridium tetanomorphum* (Wickremasinghe & Fry, 1954); a similar metabolic pathway is believed to be shared by the isolates. Variations of this pathway have been studied and are shown in detail in Pathway Tools as the histidine degradation pathways I through VI. Pathway I was not described as absent in the isolates and included the following proteins: histidine ammonia-lyase (EC 4.3.1.3, aka histidase), urocanate hydratase (EC 4.2.1.49, aka urocanase), imidazolonepropionase (EC 3.5.2.7), and formiminoglutamase (EC 3.5.3.8). The only enzyme that was missing, for all isolates, was formiminoglutamase. The first three proteins of this pathway were all predicted as part of the “histidine degradation” subsystem in RAST. In addition, the isolates had glutamate formiminotransferase (EC 2.1.2.5) predicted.
In pathway II, the first three proteins of pathway I are required, as well as formiminoglutamate deiminase (EC 3.5.3.13) and N-formylglutamate amidohydrolase (EC 3.5.1.68). The last two enzymes of pathway II are missing in all the isolates. Pathway III also shares the first three proteins with pathway I and additionally has glutamate formimidoyltransferase (EC 2.1.2.5), formiminotetrahydrofolate cyclodeaminase (EC 4.3.1.4), and methenyltetrahydrofolate cyclohydrolase (EC 3.5.4.9). These proteins were present in all isolates, and so, pathway III is complete for all isolates. Based on a search for “histidine degradation” in the “Features in Subsystems” tab, Formiminotetrahydrofolate cyclodeaminase and methenyltetrahydrofolate cyclohydrolase were not predicted as part of the “histidine degradation” subsystem; they are included in the “one-carbon metabolism by tetrahydropterines” subsystem.

In pathway IV, two enzymes are required: histidine-2-oxoglutarate aminotransferase (EC 2.6.1.38) and imidazol-pyruvate reductase (EC 1.1.1.111). These enzymes were missing in all five isolates. In pathway V, two enzymes are required: histidine-pyruvate aminotransferase (EC 2.6.1.58) and imidazole-lactate dehydrogenase. These enzymes were also missing in all five isolates. Lastly, pathway VI shares the first three proteins with pathway I and additionally has 4-imidazolone-5-propionate oxidase (EC 1.17.3.-). All five isolates were missing 4-imidazolone-5-propionate oxidase. Overall, histidine degradation pathways I and III may be present for all isolates; pathways II, IV, V, and VI are either not complete or completely absent.

Arginine fermentation

As mentioned in the section on *Filifactor alocis* (see page 4), *F. alocis* utilizes arginine and converts it to butyrate and ammonia (Uematsu et al., 2003). According to RAST, the isolates
each have proteins belonging to the “arginine and ornithine degradation” subsystem; some of the proteins included in this subsystem that were found in at least four of the isolates were: arginine deiminase (EC 3.5.3.6), ornithine carbamoyltransferase (EC 2.1.3.3), carbamate kinase (EC 2.7.2.2), and arginine decarboxylase (EC 4.1.1.19). ACC19a, CM2, CM5, and OBRC8 have each of these four proteins. AS15 is missing arginine deiminase. The first three of these proteins match to the arginine degradation V (arginine deiminase) pathway in Pathway Tools. Some of the products of this pathway are ammonium, L-ornithine, ATP, and CO₂. This pathway provides organisms with energy, carbon, and nitrogen. The arginine deiminase pathway may also be a critical factor in oral biofilm pH homeostasis, through production of ammonia (Burne & Marquis, 2000).

Methionine fermentation

In RAST, for all five isolates, the “methionine degradation” subsystem was predicted. In all five isolates, the following proteins were predicted: methionine ABC transporter proteins, S-adenosylmethionine synthetase (EC 2.5.1.6, formerly EC 2.4.2.13), S-adenosylhomocysteine nucleosidase (EC 3.2.2.9), and pyruvate-flavodoxin oxidoreductase (EC 1.2.7.-). Additionally, ACC19a, CM2, CM5, and OBRC8 had a methionine gamma-lyase (EC 4.4.1.11) and AS15 had an S-ribosylhomocysteine lyase (EC 4.4.1.21). In Pathway Tools there are three pathways for methionine degradation; in the methionine degradation II pathway, methionine gamma-lyase simultaneously deiminates and dethiomyethylates methionine to 2-oxobutanoate. This pathway is complete in the isolates.

In the methionine degradation I pathway, which converts L-methionine to L-homocysteine, S-adenosylmethionine synthetase (EC 2.5.1.6) is required in the first step. This isolates all have this protein. In the second step, a methyltransferase (EC 2.1.1.-) that can convert S-adenosyl-L-
methionine to S-adenosyl-L-homocysteine is needed. In the final step, an S-adenosylhomocysteine hydrolase (EC 3.3.1.1) converts S-adenosyl-L-homocysteine into L-homocysteine. The isolates do not have these last two proteins, so this pathway is not complete for the isolates.

In the methionine degradation III pathway, the following proteins are required: L-phenylalanine: 2-oxo-4-methylthiobutanoate aminotransferase (EC 2.6.1.88?), 2-oxo-4-methylthiobutanoate carboxy-lyase, and methionol dehydrogenase (EC 1.1.1.1). The isolates did not have any of these proteins, and therefore the pathway is absent in all the isolates.

Threonine fermentation

Interestingly, ACC19a is predicted to be prototrophic for threonine and also predicted to have the “threonine degradation” subsystem; the same is true for the other isolates. The proteins that belong to the “threonine degradation” subsystem, that were found for all the isolates, include a threonine dehydratase (EC 4.3.1.19) and a low-specificity L-threonine aldolase (EC 4.1.2.5). Isolate AS15 additionally has a threonine dehydrogenase. In Pathway Tools, there are four threonine degradation pathways. Pathway I requires the following proteins: a threonine dehydratase, 2-iminobutanoate deaminase (EC 4.3.1.19), 2-ketobutyrate formate-lyase (EC 2.3.1.29), phosphate propionyltransferase (EC 2.3.1.222), and propionate kinase (EC 2.7.2.15). This pathway is not complete for the isolates.

In pathway II, a threonine dehydrogenase (EC 1.1.1.103) and 2-amino-3-ketobutyrate CoA ligase (EC 2.3.1.29) are required. These proteins are missing from isolates ACC19a, CM2, CM5, and OBRC8; AS15 is missing 2-amino-3-ketobutyrate CoA ligase. Pathway III also requires a threonine dehydrogenase, and additionally an aminoacetone oxidase (EC 1.4.3.21). This pathway
is also not complete in the isolates. Lastly, pathway IV requires a threonine aldolase (EC 4.1.2.5) and an acetaldehyde dehydrogenase (EC 1.2.1.10). All the isolates have a threonine aldolase but not an acetaldehyde dehydrogenase. Overall, it appears that the isolates do not have complete threonine degradation pathways.

Lysine fermentation

According to RAST, the “lysine degradation” subsystem was predicted for all the isolates; the only “lysine degradation” protein that was predicted for each isolate was a lysine decarboxylase (EC 4.1.1.18). It should be noted that for each isolate there was some ambiguity with the annotation; the protein was annotated as having two functions, the second of which was being an arginine decarboxylase (EC 4.1.1.19). According to Pathway Tools, there are ten lysine degradation pathways as well as a “lysine fermentation to acetate and butyrate” pathway. L-lysine decarboxylase was found as part of the lysine degradation I and X pathways. In pathway I, lysine decarboxylase is the only required protein, and it is responsible for converting L-lysine to cadaverine. In strict anaerobes, cadaverine is produced as an essential component of peptidoglycan. In E. coli, cadaverine is secreted to protect the cells when the environment is acidic. Some organisms can further degrade cadaverine to acetyl CoA (lysine degradation X pathway), but this does not occur with the isolates.
Chapter 8: Conclusions

The five oral isolates, Peptostreptococcaceae spp. ACC19a, CM2, CM5, OBRC8, and AS15, form a novel branch within the family Peptostreptococcaceae. Isolate AS15 is a strain of the species \textit{Eubacterium} yurii subs. margaretiae, while the other isolates are novel species. While DDH values are not enough to distinguish whether strains ACC19a, CM2, CM5, and OBRC8 are different species, based on amino acid biosynthesis predictions, it is hypothesized that stains strains ACC19a and CM5 are of the same species and strains CM2 and OBRC8 are of a different species.

Genomic annotations performed by the two sequencing centers, the BROAD and the JCVI, were not similar enough to do comparative analysis, but the use of a combination of publicly-available annotation pipelines was able to aid with this problem. At least 50% of all proteins predicted for each isolate was identified by 5 different annotation methods, and predictions that were made in pipelines other than the ones used by the sequencing centers were able to assist in making metabolic predictions. While it was found that there were differences in amino acid auxotrophy, experimentally, we were unable to observe a result suggesting that these differences were unique to the three proposed species: ACC19a and CM5, CM2 and OBRC8, and AS15. Due to the fastidious nature of the isolates, all that was ascertained was that the isolates require some factor found in yeast extract and that 5 g/L of yeast extract, as opposed to anything less, supported growth. The hypothesis that amino acids found in the yeast extract were the growth factor that the isolates needed was disproven and further testing of the composition of yeast extract should be the next step in analysis.
References


# The goal of this script is to compare the protein FASTA outputs from the five different annotation pipelines used for the selected genome. Some of the proteins match identically between the files, but some proteins have different start positions. Some proteins have a different starting peptides. The difficulty is matching the protein sequences that overlap.

---

# Input files

This section will be changed into queries once the rest of the script is functional.

```python
rast_faa_filename = "..\RAST\RAST Eubacteriaceae bacterium ACC19 6666666.41977.faa"
basy_faa_filename = "..\BASY\BASYS Eubacteriaceae ACC19.faa"
broadjcvi_faa_filename = "..\BROAD\eubacterium_ACC19\euba_bact_acc19a.1.genes.fasta"
xbase_faa_filename = "..\XBASE\XBASE Eubacteriaceae bacterium ACC19.faa"
manatee_faa_filename = "..\Manatee\ACC19a_protein_multifasta_seq.fsa"

rast_faa_filename = "..\RAST\RAST Eubacteriaceae bacterium CM2 6666666.41978.faa"
basy_faa_filename = "..\BASY\BASYS Eubacteriaceae CM2.faa"
broadjcvi_faa_filename = "..\BROAD\eubacterium_cm2\euba_bact_cm2.1.genes.fasta"
xbase_faa_filename = "..\XBASE\XBASE Eubacteriaceae bacterium CM2.faa"
manatee_faa_filename = "..\Manatee\CM2_protein_multifasta_seq.fsa"

rast_faa_filename = "..\RAST\RAST Eubacteriaceae bacterium CM5 6666666.41979.faa"
basy_faa_filename = "..\BASY\BASYS Eubacteriaceae CM5.faa"
broadjcvi_faa_filename = "..\BROAD\eubacterium_CM5\eubact_bact_cm5.1.genes.fasta"
xbase_faa_filename = "..\XBASE\XBASE Eubacteriaceae bacterium CM5.faa"
manatee_faa_filename = "..\Manatee\CM5_protein_multifasta_seq.fsa"

rast_faa_filename = "..\RAST\RAST Eubacteria sp. OBRC8 6666666.41745.faa"
basy_faa_filename = "..\BASY\BASYS Eubacteria OBRC8.faa"
broadjcvi_faa_filename = "..\JCVI\OBRC8\JCVI Eubacteriaceae sp OBRC8.fasta"
```
xbase_faa_filename = "..\XBASE\XBASE Eubacteriaceae bacterium OBRC8.faa"
manatee_faa_filename = "..\Manatee\OBRC8_protein_multifasta_seq.fsa"

rast_faa_filename = "..\RAST\RAST Eubacterium sp. AS15 6666666.42048.faa"
basys_faa_filename = "..\BASYS\BASYS Eubacteriaceae AS15.faa"
broadjcvi_faa_filename = "..\JCVI\AS15\JCVI Eubacterium AS15.fasta"
xbase_faa_filename = "..\XBASE\XBASE Eubacteriaceae bacterium AS15.faa"
manatee_faa_filename = "..\Manatee\AS15_protein_multifasta_seq.fsa"

#==================================
#DICTIONARIES
RAST_translation2product = {}
BASYS_translation2product = {}
BROADJCVI_translation2product = {}
XBASE_translation2product = {}
MANATEE_translation2product = {}
pipelines = [RAST_translation2product, BASYS_translation2product, BROADJCVI_translation2product, XBASE_translation2product, MANATEE_translation2product]

#==================================
#OPEN THE RAST FILE AND SEPARATE THE SEQUENCES INTO A LIST
input_handle1 = open(rast_faa_filename, "r")
f = input_handle1.readlines()
f2 = ".join(f)
f3 = f2.split('>')

#GO THROUGH THE LIST AND CREATE A DICTIONARY {SEQUENCE:PRODUCT NAME}
for item in f3:
    if len(item) > 1:
        items = item.split('n')
        product_name = items[0]
        translation = items[1:]
        translation = ".join(translation)
        translation = translation.replace('n','')
        if translation not in RAST_translation2product:
            RAST_translation2product[translation] = product_name
        else:
            #There are some situations where a protein sequence will show up more than once
            #in the genome, such as mobile elements
            p = RAST_translation2product[translation]
            if type(p) == list:
                #There was one instance where a sequence showed up three times
                p.append(product_name)
            RAST_translation2product[translation] = p
else:
    RAST_translation2product[translation] = [product_name, p]
input_handle1.close()

#==================================================================================================
==
#OPEN THE BASYS FILE AND SEPARATE THE SEQUENCES INTO A LIST
input_handle2 = open(basys_faa_filename, "r")

f = input_handle2.readlines()
f2 = ".join(f[:])
f3 = f2.split(">")

#GO THROUGH THE LIST AND CREATE A DICTIONARY {SEQUENCE:PRODUCT NAME}
for item in f3:
    if len(item) > 1:
        items = item.split("
")
        product_name = items[0]
        translation = items[1:]
        translation = ".join(translation)
        translation = translation.replace("\n","")
        if translation not in BASYS_translation2product:
            BASYS_translation2product[translation] = product_name
        else:
            p = BASYS_translation2product[translation]
            if type(p) == list:
                p.append(product_name)
                BASYS_translation2product[translation] = p
            else:
                BASYS_translation2product[translation] = [product_name, p]

input_handle2.close()

#==================================================================================================
==
#OPEN THE BROAD/JCVI FILE AND SEPARATE THE SEQUENCES INTO A LIST
input_handle3 = open(broadjcvi_faa_filename, "r")

f = input_handle3.readlines()
f2 = ".join(f[:])
f3 = f2.split(">")

#GO THROUGH THE LIST AND CREATE A DICTIONARY {SEQUENCE:PRODUCT NAME}
for item in f3:
    if len(item) > 1:
        items = item.split("
")
        product_name = items[0]
if "JCVI" in broadjcvi_faa_filename:
    product_name = product_name.split('|
    product_name = product_name[4]
elif "BROAD" in broadjcvi_faa_filename:
    product_name = product_name.split(':')
    product_name = product_name[-1]
    product_name = product_name.replace('*','')
else:
    product_name = product_name.split('|
    product_name = product_name[-1]
    product_name = product_name.replace('*','')
translation = items[1:]
translation = ''.join(translation)
translation = translation.replace('n','')
translation = translation.replace('*','')

if translation not in BROADJCVI_translation2product:
    BROADJCVI_translation2product[translation] = product_name
else:
    p = BROADJCVI_translation2product[translation]
    if type(p) == list:
        p.append(product_name)
        BROADJCVI_translation2product[translation] = p
    else:
        BROADJCVI_translation2product[translation] = [product_name,p]

input_handle3.close()

#===================================================
==
#OPEN THE XBASE FILE AND SEPARATE THE SEQUENCES INTO A LIST
input_handle4  = open(xbase_faa_filename, "r")
f = input_handle4.readlines()
f2 = ''.join(f)
f3 = f2.split('>')

#GO THROUGH THE LIST AND CREATE A DICTIONARY {SEQUENCE:PRODUCT NAME}
for item in f3:
    if len(item) > 1:
        items = item.split('n')
        product_name = items[0]
        translation = items[1:]
        translation = ''.join(translation)
        translation = translation.replace('n','')
        translation = translation.replace('*','')

        if translation not in XBASE_translation2product:
            XBASE_translation2product[translation] = product_name
        else:
            p = XBASE_translation2product[translation]
if type(p) == list:
    p.append(product_name)
    XBASE_translation2product[translation] = p
else:
    XBASE_translation2product[translation] = [product_name,p]
input_handle4.close()

#OPEN THE MANATEE FILE AND SEPARATE THE SEQUENCES INTO A LIST
input_handle5  = open(manatee_faa_filename, "r")
f = input_handle5.readlines()
f2 = ".\join(f)
f3 = f2.split('>')

#GO THROUGH THE LIST AND CREATE A DICTIONARY \{SEQUENCE:PRODUCT NAME\}
for item in f3:
    if len(item) > 1:
        items = item.split('n')
        product_name = items[0]
        product_name = product_name.split('t')
        product_name = product_name[1]
        translation = items[1:]
        translation = ".\join(translation)
        translation = translation.replace('n','')
        if translation not in MANATEE_translation2product:
            MANATEE_translation2product[translation] = product_name
        else:
            p = MANATEE_translation2product[translation]
            if type(p) == list:
                p.append(product_name)
                MANATEE_translation2product[translation] = p
            else:
                MANATEE_translation2product[translation] = [product_name,p]
input_handle5.close()
print ('files closed')

#MAKE A LIST OF ALL THE DIFFERENT SEQUENCES FOUND, EVEN IF THERE IS AN OVERLAP
#OR MISS-TRANSLATED CODON IN THE BEGINNING OF THE SEQUENCES
Proteins = []
for pipeline in pipelines:
    for protein in pipeline:
        if protein not in Proteins:
            Proteins.append(protein)
print (len(Proteins))
def commonOverlapIndexOf(text1, text2):
    # Cache the text lengths to prevent multiple calls.
    text1_length = len(text1)
    text2_length = len(text2)
    # Eliminate the null case.
    if text1_length == 0 or text2_length == 0:
        return 0
    # Truncate the longer string.
    if text1_length > text2_length:
        text1 = text1[-text2_length:]
    elif text1_length < text2_length:
        text2 = text2[-text1_length:]
    # Quick check for the worst case.
    if text1 == text2:
        return min(text1_length, text2_length)
    # Start by looking for a single character match
    # and increase length until no match is found.
    best = 0
    length = 1
    while True:
        pattern = text1[-length:]
        found = text2.find(pattern)
        if found == -1:
            return best
        length += found
        if text1[-length:] == text2[:length]:
            best = length
            length += 1

#FIND PROTEINS THAT ARE OVERLAPPING, DECREASING THE NUMBER OF UNIQUE SEQUENCES.
#THE OVERLAPPING PROTEINS WILL BE ADDED TO THE LIST OF PROTEINS AS A TWO-VARIABLE LIST

PROTEINS = []
PROTEINS_used = []

for protein_a in Proteins:
    match_found = False
    for protein_b in Proteins:
        overlap = commonOverlapIndexOf(protein_a, protein_b)
        if overlap > 25 and protein_a != protein_b:
            match_found = True
overlapping_proteins = [protein_a,protein_b]
overlapping_proteins.sort()
# print (overlapping_proteins)

if protein_a not in PROTEINS_used and protein_b not in PROTEINS_used:
    if overlapping_proteins not in PROTEINS:
        PROTEINS.append(overlapping_proteins)
    if protein_a not in PROTEINS_used:
        PROTEINS_used.append(protein_a)
    if protein_b not in PROTEINS_used:
        PROTEINS_used.append(protein_b)
if match_found == False:
    if protein_a not in PROTEINS_used:
        protein = [protein_a,'None']
        # print (protein)
        # print ('n')
        if protein not in PROTEINS:
            PROTEINS.append(protein)
print (len(PROTEINS))
print ('finished looking for overlaps')
#====================================================================
==
# PRINT THE OUTPUT FOR EACH UNIQUE PROTEIN SEQUENCE
protein2occurrence ={}
translation2product = {}
for protein in PROTEINS:
    occurrence =[]
    # if RAST_translation2product.has_key(protein[0]):
    if protein[0] in RAST_translation2product:
        rast_product = RAST_translation2product[protein[0]]
        # print ('RAST'+rast_product +'
        occurrence.append('1a')
        # elif RAST_translation2product.has_key(protein[1]):
        elif protein[1] in RAST_translation2product:
            rast_product = RAST_translation2product[protein[1]]
            occurrence.append('1b')
    else:
        rast_product = 'None'
        occurrence.append('0')
    #===================================================
    # if BASYS_translation2product.has_key(protein[0]):
    if protein[0] in BASYS_translation2product:
        basys_product = BASYS_translation2product[protein[0]]
        occurrence.append('1a')
    # elif BASYS_translation2product.has_key(protein[1]):
    elif protein[1] in BASYS_translation2product:
        basys_product = BASYS_translation2product[protein[1]]
        occurrence.append('1b')
else:
    basys_product = 'None'
    occurrence.append('0')
#===================================================
#if BROADJCVI_translation2product.has_key(protein[0]):
if protein[0] in BROADJCVI_translation2product:
    broadjcvi_product = BROADJCVI_translation2product[protein[0]]
    occurrence.append('1a')
#elif BROADJCVI_translation2product.has_key(protein[1]):
elif protein[1] in BROADJCVI_translation2product:
    broadjcvi_product = BROADJCVI_translation2product[protein[1]]
    occurrence.append('1b')
else:
    broadjcvi_product = 'None'
    occurrence.append('0')
#===================================================
#if XBASE_translation2product.has_key(protein[0]):
if protein[0] in XBASE_translation2product:
    xbase_product = XBASE_translation2product[protein[0]]
    occurrence.append('1a')
#elif XBASE_translation2product.has_key(protein[1]):
elif protein[1] in XBASE_translation2product:
    xbase_product = XBASE_translation2product[protein[1]]
    occurrence.append('1b')
else:
    xbase_product = 'None'
    occurrence.append('0')
#========================================
#if MANATEE_translation2product.has_key(protein[0]):
if protein[0] in MANATEE_translation2product:
    manatee_product = MANATEE_translation2product[protein[0]]
    occurrence.append('1a')
#elif MANATEE_translation2product.has_key(protein[1]):
elif protein[1] in MANATEE_translation2product:
    manatee_product = MANATEE_translation2product[protein[1]]
    occurrence.append('1b')
else:
    manatee_product = 'None'
    occurrence.append('0')
#===================================================
print (str(protein))
print (occurrence)
#protein2occurrence[protein[0]] = occurrence
protein2occurrence[str(protein)] = occurrence #NEEDS IMPROVEMENT HERE BECAUSE OF PROTEINS SEQUENCES THAT OVERLAP
translation2product[str(protein)] =
[rast_product,basys_product,broadjcvi_product,xbase_product,manatee_product]
print
(str(rast_product)+'\n'+str(basys_product)+'\n'+str(broadjcvi_product)+'\n'+str(xbase_product)+'\n'+str(occurrence))
n'+str(manatee_product))
    print (\n')
#=======================================================
print (len(PROTEINS))
union = 0
for protein in protein2occurrence:
    if protein2occurrence[protein] == ['1a','1a','1a','1a','1a']:
        union+=1
    #else:
    #    print (protein2occurrence[protein])
print (union)
print (\n\n')
#Look for patterns in protein calls:
patterns = {}
for protein in protein2occurrence:
    occurrence = protein2occurrence[protein]
    list_occurences = occurrence
    #====================================
    #Change values to 1 to establish a pattern
    if '1a' in list_occurences:
        #list_occurences[list_occurences.index('1a')] = '1'
        for n,i in enumerate(list_occurences):
            if i == '1a':
                list_occurences[n] = '1'
    if '1b' in list_occurences:
        #list_occurences[list_occurences.index('1b')] = '1'
        for n,i in enumerate(list_occurences):
            if i == '1b':
                list_occurences[n] = '1'
    #====================================
    if str(list_occurences) not in patterns:
        patterns[str(list_occurences)] = 1
    else:
        number = patterns[str(list_occurences)]
        number +=1
        patterns[str(list_occurences)] = number
total_proteins_check = 0
for pattern in patterns:
    print (pattern + '\t'+str(patterns[pattern]))
    total_proteins_check += patterns[pattern]
print (total_proteins_check)