ABE516 Project 3
Ken Youens-Clark
Mashing iMicrobe

Introduction: Sequence Comparison Methods

Metagenomics is the study of all the DNA from a given environment as opposed to traditional genomics where an organism is isolated, cultured, and sequenced. Most modern sequencing platforms (454, Illumina, IonTorrent) can only read short strings of DNA, usually up to 200 or 300 base pairs, which only gives a small window on the genome being studied. In traditional genomics, we know the organism we are studying is the only one present in the sample, so it's possible to somewhat confidently reassemble these short reads in order to study the genome, but in metagenomics, we do not have that luxury.

The word "meta" in Greek means "after" or "beyond," but maybe "transcendent" is a better word. Given that 99% of the world's microbes refuse to grow in a isolation and culture, we must leave behind the certainty of older methods to study genomes at a higher level. The idea of assembling millions of short reads from potentially dozens or even hundreds of unknown organisms seems certain to produce chimeric genomes that would only pollute downstream analysis. Another way to study unassembled reads could be to compare them to known genomes. A naive users might use traditional alignment tools like BLAST which are very accurate at the cost of being quite slow, far too slow to handle the millions of reads typically produced by modern sequencing platforms. This approach also has the disadvantage of database bias — we can only align to organisms we've isolated and sequenced, so we will necessarily only rediscover organisms similar to those, and we already know that what we've seen is only a very small portion of what actually exists in the world

An alternative to metagenomics can be found in amplicon sequencing where specific targets, such as the highly conserved 16S rRNA bacterial gene, are amplified and compared to known sequences, but this method also suffers from database bias most importantly in that it completely ignores the presence of fungi and viruses. So, if we don't want to assemble short reads or compare them to reference database or rely on marker genes, what is left? In this project, I will use a substring-sequence composition analysis on metagenomic samples which were sequenced on a variety of short-read platforms.

The Promise of K-mers

There are many methods we can use to compare any two strings from a given language. Rather than using local alignment, we will instead look at composition analysis. For instance, one typical assessment of DNA sequences known as "GC content" is the proportion of Gs and Cs in relation to the As and Ts. This dinucleotide representation might help to generally classify short sequences, for instance, into those from a coding region (which are found to be GC-rich) or from a particular organism (yeast is somewhat low at around 38% while some *Actinobacteria* can be as high as 72%). Finding the GC ratio of a sequence is extremely cheap (count the Gs and Cs and divide by sequence length), but it is fairly worthless for classification.

An alternate way to compare short sequences is to almost counter-intuitively make them even smaller and count not individual bases but shorter substrings called "k-mers." A k-mer is a *k*-length substring from a sequence of letters and is analogous to words like "monomer" or "dimer" to denote a structure composed of one or two parts. This pictures shows the 10 20-mers contained in a sample sequence:

```
SEQ:  ACAGCGCAAGGACGTGTTCGAGATCAAGG
 1    ACAGCGCAAGGACGTGTTCG
 2     CAGCGCAAGGACGTGTTCGA
 3      AGCGCAAGGACGTGTTCGAG
 4       GCGCAAGGACGTGTTCGAGA
 5        CGCAAGGACGTGTTCGAGAT
 6         GCAAGGACGTGTTCGAGATC
 7          CAAGGACGTGTTCGAGATCA
 8           AAGGACGTGTTCGAGATCAA
 9            AGGACGTGTTCGAGATCAAG
10             GGACGTGTTCGAGATCAAGG
```

The number of possible 20-mers from the language {A, C, T, G} is 4^20 or ~2e12; therefore, if you find a 20-mer is shared between two sequences, it's fairly unlikely to be a random event. Some regions of the genome can be quite repetitive and conserved, and so in practice we may find some k-mers are found thousands of times. We can ignore these just as we might ignore words like "a,", "an," and "the" if we were comparing any two texts from English. These are common, low-information "mers" that we can safely ignore.

Numerous algorithms exist for counting and comparing shared k-mers in genomes. I will be using Mash (Odnov et al., 2016) to compare and cluster around metagenomic samples that can be found at iMicrobe. The bulk of the data in this repository was inherited from the CAMERA project (Sun et al., 2011) after it shutdown in 2014 and includes data from very diverse biomes including acid mine drainage, soil, ocean, and host-associated (e.g., human, mouse).

Mash uses MinHash, a min-wise independent permutations locality sensitive hashing scheme, to create a signature of a sample. This algorithm was originally developed in 1997 at AltaVista, one of the earliest Internet search engines, to determine the similarity of any two given web pages so as to avoid indexing duplicate data. Rather than counting every k-mer found in a sample, Mash uses a "sketching" step to select a "minimum hash value" which is like a signature for a sample. These sketches can be compared to determine how similar one sample is from another.

Data Availability

The iMicrobe project has create Mash indexes for samples (https://www.imicrobe.us/#/samples) and placed them into the Cyverse Data Store (/iplant/home/shared/imicrobe). Using the iRODS command line tool "ils" to recursively list all files and filtering for those ending in ".msh," I created a list of 4,158 files which I used "iget" to retrieve to my local disk. The size of the original data this represents is on the order of 10T, but the local Mash files occupy around 50M, a compression of orders of magnitude. Of course, such a scale of compression also includes inevitable data loss, but in this exploration we will determine if Mash can still group samples accurately.

Creating a Distance Matrix

The time for Mash to sketch a given set of samples is relatively fast, and the time to compare Mash sketches is almost trivial. The use of almost any other comparison method to do a complete pairwise comparison of 3684 samples likely would be impractical, but Mash is able to create a distance matrix in around 3 minutes:

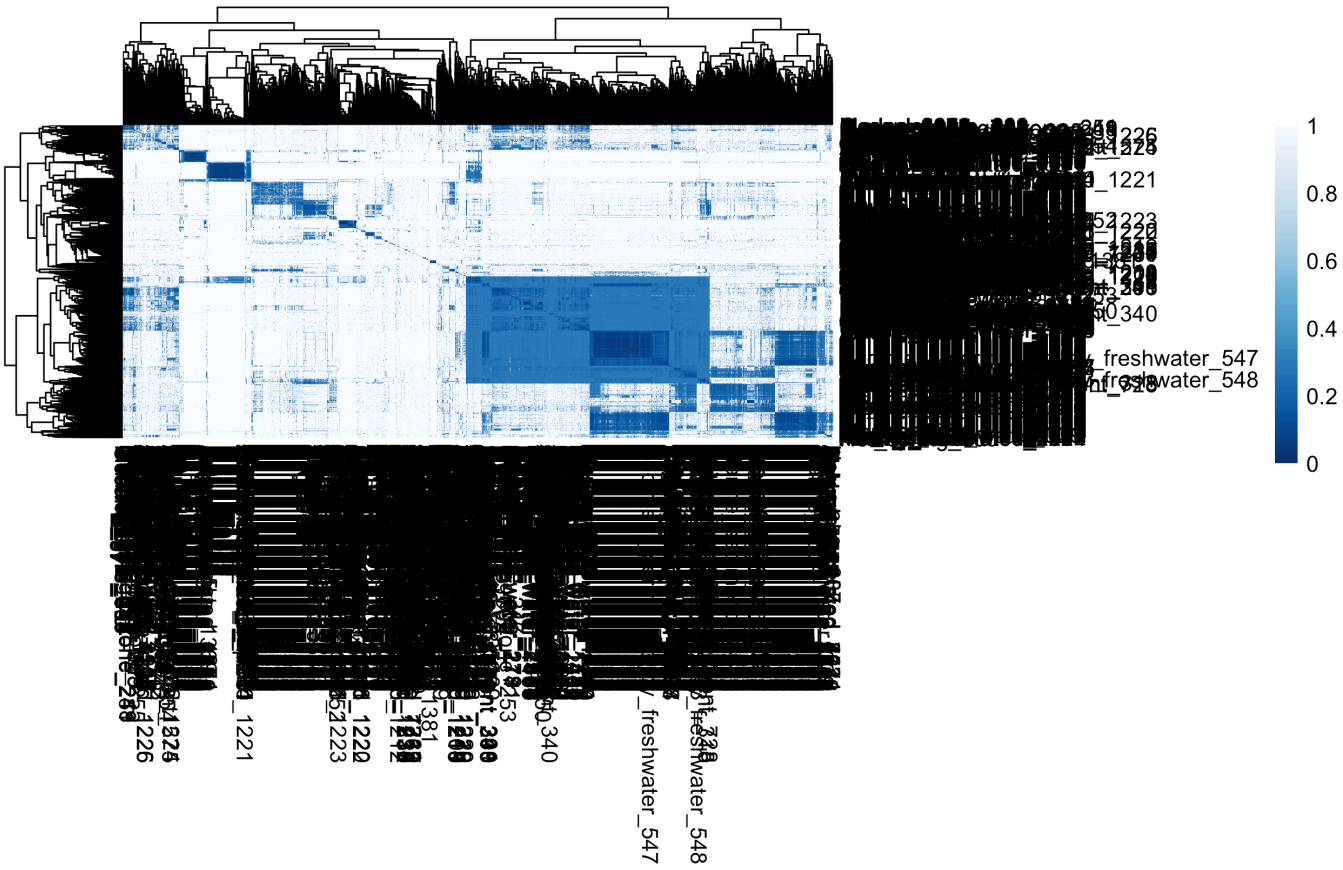
```
$ ls -1 mash/*.msh | wc -l
    3684
$ find mash -name \*.msh > all-files
$ mash paste -l all all-files
Writing all.msh...
time mash dist -t all.msh all.msh > mash-dist.txt

real  2m49.756s
user  2m53.846s
sys   0m0.785s
```

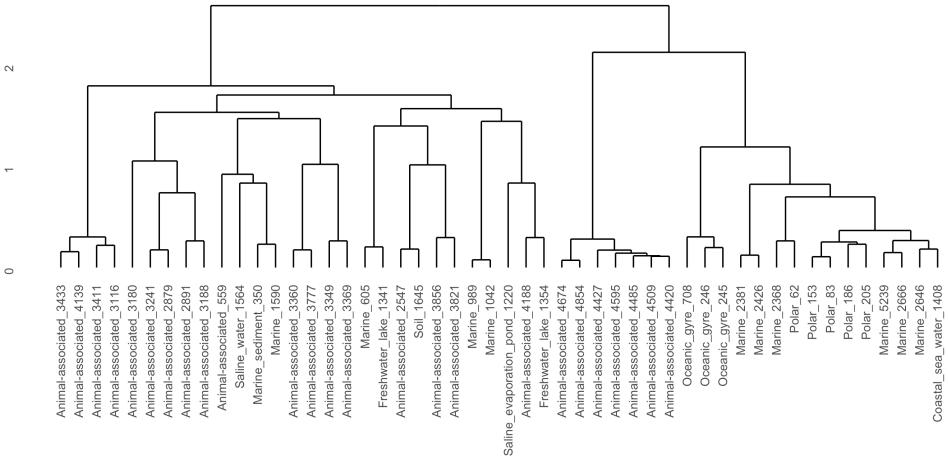
Mash has an annoying output format in that the first line starts with a "#" symbol ("#query") which is almost universally regarded as a comment. I wrote various Python programs to fix this as well as to annotate my sample names with biome or other metadata.

Clustering

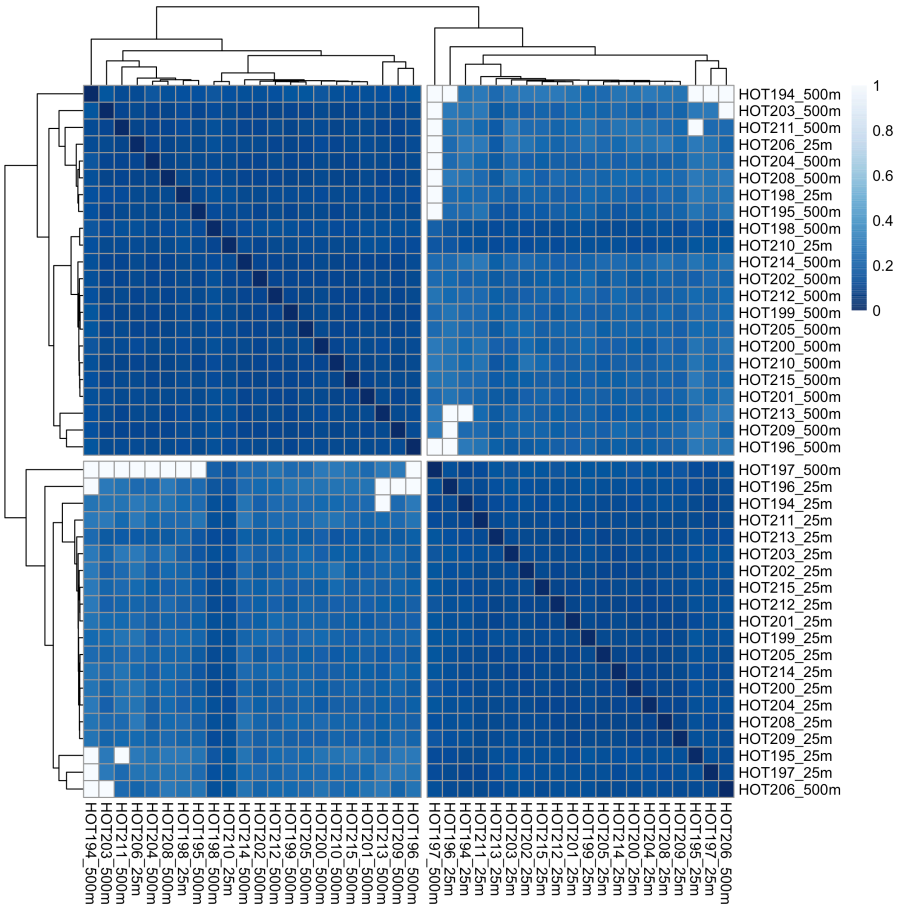
While it is possible for R to cluster all 3600 samples, it's rather difficult to read the names of the clusters to understand which samples are grouping. Still, we definitely see some significant clustering:



To better see which samples Mash is clustering, I decided to randomly sample 50 metagenomes for hierarchical clustering. I ran this 10 times and include the output in the appendix, but we can take one to examine here. What we can see in this cluster is that Mash is fairly good at clustering samples into “Animal-associated” and water (saline, marine, ocean, polar).



One might expect Mash to be able to discriminate samples from entirely different biomes. To see if Mash can resolve samples which are more closely related, I chose to cluster the 42 samples from a collection of Hawaiian Ocean Timeseries (HOT) which were taken at depths of 25 and 500m. I found that Mash does quite well at clustering these highly similar samples.



Conclusion

Mash's selection of k-mers is clearly imperfect as can be seen how some samples are placed into unexpected clusters. Further investigation (e.g., taxonomic or functional annotation) might reveal actual similarities driving the clustering, but it seems unlikely, for instance, that ocean samples collected at 25m (photic) should cluster with 500m (aphotic) samples as the organisms present and their metabolic profiles ought to be significantly different. Regardless, Mash is redeemed by the speed with which it can be used to identify closely related samples in a very quick fashion.

Code Availability

All code is available in https://github.com/kyclark/abe516/tree/master/project3

```
library("ggplot2")
library("ggdendro")
library("pheatmap")
library("RColorBrewer")
library("R.utils")

wd = "~/work/abe516/project3"
setwd(wd)
full.df = read.table("dist.txt", header = T, check.names = F)
colors = colorRampPalette(rev(brewer.pal(9, "Blues")) )(255)
pheatmap(full.df, col = colors, filename = file.path(wd, "full-heatmap.png"))

# Subsample full dataset
num.samples = 50
num.iter = 10
for (i in 1:num.iter) {
  print(paste("Sample", i))
  sample.cols = sample(nrow(full.df), num.samples)
  df = full.df[sample.cols, sample.cols]
  fit = hclust(as.dist(df), method = "ward.D2")
  dg = ggdendro::ggdendrogram(fit, rotate=F)
  ggsave(file = file.path(wd, paste0("sample", i, ".png")),
         limitsize = FALSE, width = 10, height = 5, plot = dg)
}

# HOT
hot.df = read.table("hot-dist.txt", header = T, check.names = F)
hot.fit = hclust(as.dist(hot.df), method = "ward.D2")
dg = ggdendro::ggdendrogram(hot.fit, rotate=F)
ggsave(file = file.path(wd, "hot.png"),
       limitsize = FALSE, width = 10, height = 5, plot = dg)
pheatmap(hot.df,
         col = colors,
         cutree_rows = 2,
         cutree_cols = 2,
         filename = file.path(wd, "hot-heatmap.png"))
```

Additional Images