Axial RNA-SIP Sample Processing: From Raw Reads to OTU Bar Graphs

Overview:

1. Quality Check the files with FastQC and MultiQC
2. Trim the reads with Trimmomatic
3. Combine all the forward and all the reverse read segments using cat command
4. Combine concatenated forward and reverse reads using Flash
5. Run SortMeRNA to remove rRNA from the mRNA
6. Use fq2fa from IDBA_UD to convert .fastq output of SortMeRNA to .fasta before inputting into Mothur
7. Classify taxonomy of rRNA using Mothur classify.seqs
8. Use R to properly format Mothur output
9. Use R to create a count table of taxonomic groups by desired classification (Domain, Kingdom, Phylum, Class, Order, Family, Genus, Species)
10. Use ggplot to create bar graphs

Documentation for each program:

1. FastQC - https://dnacore.missouri.edu/PDF/FastQC_Manual.pdf
2. MultiQC - https://multiqc.info/docs/
3. Flash (Fast Length Adjustment of Short reads) - https://ccb.jhu.edu/software/FLASH/
4. SortmeRNA - https://bioinfo.lifl.fr/RNA/sortmerna/code/SortMeRNA-user-manual-v2.1.pdf

- https://github.com/biocore/sortmerna

5. Mothur - classify.seqs - https://mothur.org/wiki/classify.seqs/
6. fq2fa - https://denbi-metagenomics-workshop.readthedocs.io/en/latest/assembly/idba_ud.html

- https://github.com/loneknightpy/idba

7. R code and directions will be included at the end of this document

####**Note: all of these programs were run on WHOI's cluster, Poseidon, so the code for that is included. Additionally, all programs were installed using Conda Environments**

##**1. Quality Check files with FastQC and MultiQC**

```
#Set up Conda environment like Emilie lists in her GitHub:
#https://github.com/emilieskoog/Isolate_assembly_workflow/blob/master/Huber_Lab_Isolate_assembly_README.md
#Instructions copied below:

# Step 1: Create a FastQC environment
conda create --name fastqc

conda activate fastqc
```

```
conda install -c bioconda fastqc

#Step 2: Run FastQC!
#Go into folder with files for fastqc-action to be done to them and create a shell file.
#Again:

nano fastqc.sh

#Here is an example of the code:

#!/bin/bash
#SBATCH --partition=compute                          # Queue selection
#SBATCH --job-name=parallel_FastQC1                  # Job name
#SBATCH --mail-type=ALL                              # Mail events (BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=selkassas@whoi.edu               # Where to send mail
#SBATCH --ntasks=2                                   # Run a single task
#SBATCH --cpus-per-task=36                           # Number of CPU cores per task
#SBATCH --mem=125gb                                  # Job memory request
#SBATCH --time=24:00:00                              # Time limit hrs:min:sec
#SBATCH --output=parallel_FastQC1%j.log              # Standard output/error

export OMP_NUM_THREADS=8

module load anaconda/5.1
source activate fastqc

cd /vortexfs1/scratch/selkassas/qc-trim/raw_data/trimmed

fastqc FS903_30_12L_GGATGT_L001_R1_006_trim_paired.fastq.gz
FS903_30_12L_GGATGT_L001_R2_006_trim_paired.fastq.gz
FS903_30_12L_GGATGT_L001_R1_006_trim_unpaired.fastq.gz
FS903_30_12L_GGATGT_L001_R2_006_trim_unpaired.fastq.gz



#Running files through multiqc:
#In whatever conda environment you want (I just did it in my fastqc environment,
#though you could also create a new one), install multiqc:

conda install -c bioconda -c conda-forge multiqc

#And here is an example of the code I used:

#!/bin/bash
#SBATCH --partition=compute                          # Queue selection
#SBATCH --job-name=parallel_multiqc                  # Job name
#SBATCH --mail-type=ALL                              # Mail events (BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=selkassas@whoi.edu               # Where to send mail
#SBATCH --ntasks=2                                   # Run a single task
#SBATCH --cpus-per-task=36                           # Number of CPU cores per task
#SBATCH --mem=125gb                                  # Job memory request
#SBATCH --time=24:00:00                              # Time limit hrs:min:sec
#SBATCH --output=parallel_multiqc%j.log              # Standard output/error

export OMP_NUM_THREADS=8

module load anaconda/5.1
source activate fastqc

cd /vortexfs1/scratch/selkassas/qc-trim/raw_data/fastqc
```

```
multiqc .
```

##2. Trim the reads with Trimmomatic

```
#Step 1: Set up a conda environment as listed on Emilie's GitHub (pasted below):
#https://github.com/emilieskoog/Isolate_assembly_workflow/blob/master/Huber_Lab_Isolate_assembly_README.md

conda create --name trimmomatic

conda activate trimmomatic

conda install -c bioconda trimmomatic

#Step 2: Add adapter file with all adapters

#If you are uncertain of what adapters were used, here is a list of them that
#trimmomatic can go through and test each one.
#If you are unsure of whether or not adapters were already removed,
#performing this step would not hurt regardless.

#1. Find path for trimmomatic using:

conda info --envs

#2. Follow the outputted path that will lead you to the trimmomatic directory
#"cd" into this trimmomatic directory and create a folder named 'adapters'

mkdir adapters

#3. Create a file with list of all adapters: all_adapters.fa

nano all_adapters.fa

#4. Copy and past the following into this newly-created all_adapters.fa file
>PrefixNX/1
AGATGTGTATAAGAGACAG
>PrefixNX/2
AGATGTGTATAAGAGACAG
>Trans1
TCGTCGGCAGCGTCAGATGTGTATAAGAGACAG
>Trans1_rc
CTGTCTCTTATACACATCTGACGCTGCCGACGA
>Trans2
GTCTCGTGGGCTCGGAGATGTGTATAAGAGACAG
>Trans2_rc
CTGTCTCTTATACACATCTCCGAGCCCACGAGAC>PrefixPE/1
AATGATACGGCGACCACCGAGATCTACACTCTTTCCCTACACGACGCTCTTCCGATCT
>PrefixPE/2
CAAGCAGAAGACGGCATACGAGATCGGTCTCGGCATTCCTGCTGAACCGCTCTTCCGATCT
>PCR_Primer1
AATGATACGGCGACCACCGAGATCTACACTCTTTCCCTACACGACGCTCTTCCGATCT
>PCR_Primer1_rc
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTAGATCTCGGTGGTCGCCGTATCATT
>PCR_Primer2
CAAGCAGAAGACGGCATACGAGATCGGTCTCGGCATTCCTGCTGAACCGCTCTTCCGATCT
>PCR_Primer2_rc
AGATCGGAAGAGCGGTTCAGCAGGAATGCCGAGACCGATCTCGTATGCCGTCTTCTGCTTG
>FlowCell1
TTTTTTTTTTAATGATACGGCGACCACCGAGATCTACAC
```

```
>FlowCell2
TTTTTTTTTTCAAGCAGAAGACGGCATACGA
>TruSeq2_SE
AGATCGGAAGAGCTCGTATGCCGTCTTCTGCTTG
>TruSeq2_PE_f
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGT
>TruSeq2_PE_r
AGATCGGAAGAGCGGTTCAGCAGGAATGCCGAG
>PrefixPE/1
TACACTCTTTCCCTACACGACGCTCTTCCGATCT
>PrefixPE/2
GTGACTGGAGTTCAGACGTGTGCTCTTCCGATCT
>PE1
TACACTCTTTCCCTACACGACGCTCTTCCGATCT
>PE1_rc
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA
>PE2
GTGACTGGAGTTCAGACGTGTGCTCTTCCGATCT
>PE2_rc
AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC>PrefixPE/1
TACACTCTTTCCCTACACGACGCTCTTCCGATCT
>PrefixPE/2
GTGACTGGAGTTCAGACGTGTGCTCTTCCGATCT>TruSeq3_IndexedAdapter
AGATCGGAAGAGCACACGTCTGAACTCCAGTCAC
>TruSeq3_UniversalAdapter
AGATCGGAAGAGCGTCGTGTAGGGAAAGAGTGTA

#move all of your files to a raw_data folder.
#It will make it easier to run a bunch of things at once.
#However, this is by no means an automated process.
#It will be better to use Sarah's SnakeMake pipeline for scalability and reproducibility.
#However, in some cases (like if your file names do not match what Sarah has in her
#SnakeFile), you've got to do things the old fashion way as shown below.

#Here is an example of the code:
#!/bin/bash
#SBATCH --partition=compute                          # Queue selection
#SBATCH --job-name=parallel_trimmomatic9             # Job name
#SBATCH --mail-type=ALL                              # Mail events (BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=selkassas@whoi.edu               # Where to send mail
#SBATCH --ntasks=2                                   # Run a single task
#SBATCH --cpus-per-task=36                           # Number of CPU cores per task
#SBATCH --mem=100gb                                  # Job memory request
#SBATCH --time=24:00:00                              # Time limit hrs:min:sec
#SBATCH --output=parallel_trimmed9%j.log              # Standard output/error

export OMP_NUM_THREADS=8

module load anaconda/5.1
source activate trimmomatic

cd /vortexfs1/scratch/selkassas/qc-trim/raw_data

trimmomatic PE FS891_RNA_GGACCC_L001_R1_001.fastq.gz
FS891_RNA_GGACCC_L001_R2_001.fastq.gz
FS891_RNA_GGACCC_L001_R1_001_trim_paired.fastq.gz
FS891_RNA_GGACCC_L001_R1_001_trim_unpaired.fastq.gz
FS891_RNA_GGACCC_L001_R2_001_trim_paired.fastq.gz
FS891_RNA_GGACCC_L001_R2_001_trim_unpaired.fastq.gz
ILLUMINACLIP:/vortexfs1/home/selkassas/.conda/envs/trimmomatic/all_adapters.fa:2:40:15
CROP:140 LEADING:10 TRAILING:10 SLIDINGWINDOW:25:10 MINLEN:50
```

##**3. Combine all the forward and all the reverse read segments using cat command**

```
#Preparing your samples
# Make sure you trim and quality control your reads before running through sortmerna!
#Use the "paired" output of trimmomatic as your input for sortmerna.
#Now, sortmerna can take merged, paired reads or unmerged, paired reads.
#However, if you have unmerged, paired reads, you will need to specify
#the names of your two outputs.
#I'll show an example of running sortmerna with each one. See below.

#If you do want to merged, paired reads, you must prepare them by:
#1) concatenating all of the forward reads together,
#then concatenating all of the reverse reads together.
#Note: there may be a lot of forward and reverse reads
#if your sample was downloaded from the Short Read Archive (SRA),
#which splits your reads up into smaller segments. See example:

#My original sample had these forward reads (this sample was split
#into 7 forward read files):

FS903_30_12L_GGATGT_L001_R1_001_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_002_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_003_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_004_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_005_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_006_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_007_trim_paired.fastq

#and these reverse reads(this sample was split into 7 reverse read files):
FS903_30_12L_GGATGT_L001_R1_001_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_001_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_002_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_003_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_004_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_005_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_006_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_007_trim_paired.fastq

#Here is the concatenation code:

#####general concatenation code:

cat file1 file2 file3...file# > newfile

#####concatenation code for the forward and reverse reads above:

cat FS903_30_12L_GGATGT_L001_R1_001_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_002_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_003_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_004_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_005_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_006_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R1_007_trim_paired.fastq >
FS903_30_12L_GGATGT_L001_trim_paired_cat_forward_reads.fastq

cat FS903_30_12L_GGATGT_L001_R2_001_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_002_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_003_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_004_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_005_trim_paired.fastq
```

```
FS903_30_12L_GGATGT_L001_R2_006_trim_paired.fastq
FS903_30_12L_GGATGT_L001_R2_007_trim_paired.fastq >
FS903_30_12L_GGATGT_L001_trim_paired_cat_reverse_reads.fastq
```

##4. Combine concatenated forward and reverse reads using Flash

```
#2) Merge the concatenated forward reads file with the concatenated reverse reads
#file using a program called flash.
#Here is how to install it into your sortmerna environment and then run the program:

conda install -c bioconda flash (mine was v. 1.2.11)

#Here is the flash code:

    #general format:

flash <insert concatenated forward reads file> <insert concatenated reverse reads file> -r
<insert read length> -d
<insert output directory path> -o <insert output file prefix - this will generate a fastq
file>

    #flash code for the example above:

flash FS903_30_12L_GGATGT_L001_trim_paired_cat_forward_reads.fastq
FS903_30_12L_GGATGT_L001_trim_paired_cat_reverse_reads.fastq -r 110 -d
/vortexfs1/scratch/selkassas/qc-trim/raw_data/merged_files_flash -o
FS903_30_12L_GGATGT_L001_trim_paired_merged

THE OUTPUT FROM FLASH THAT YOU WILL USE FOR SORTMERNA IS THE .extendedFrags.fastq FILE!!!
```

##5. Run SortMeRNA to remove rRNA from the mRNA

```
#SortMeRNA - very intensive pipeline.

#This program is tricky to use, so follow my directions closely.
#This takes a lot of computational power.
#I recommend using the following slurm script and running only a
#few samples at a time (like 10 maximum):

1) conda create --name sortmerna
2) conda activate sortmerna
3) conda install -c bioconda sortmerna

#This will give you the newest version. As of 1/18/21, it is 4.2.0.
#Any 'indexdb_rna' commands will not work.
#There is only one command now, 'sortmerna'.
#Everything, including indexing your databases will be done with this.

#Download the GitHub repo and its contents:

git clone https://github.com/biocore/sortmerna.git

#Now you will have a directory called "sortmerna" in your conda path.
cd /vortexfs1/home/selkassas/.conda/envs/sortmerna
cd sortmerna
#the sortmerna directory is a subdirectory of the main sortmerna path created by conda

#Move two additional databases into the 'rRNA_databases' directory
mv silva-bac-16s-database-id85.fasta rRNA_databases
```

```
mv silva-arc-16s-database-id95.fasta rRNA_databases

#How to index your databases:
https://github.com/biocore/sortmerna/blob/master/scripts/test.jinja.yaml
#L347 --> look at
#test17 in this file. It will give you an example of how to index.
#However, just look at my code below and it will work for you too!
#Please note that sortmerna indexes your databases as it runs.
#You will have to index your databases each time you run a new sample.
#Because of this, you will have to delete the 'run' directory within
#the workdir it creates before running each new sample.
#See below examples with two samples - the first example is with merged, paired reads,
#and the second is with unmerged, paired reads (as promised!):

#SortmeRNA code - merged, paired reads

    #general format:

sortmerna --ref <insert path to reference database 1> --ref <insert path to reference
database 2> --ref <insert path to reference database 3> --reads <insert merged reads file
from Flash (will end in .extendedFrags.fastq> --aligned <insert file name_rRNA> --fastx
--other <insert file name_mRNA> --otu_map -v

#some notes: you can list as many reference databases as you want, just make sure you
#indicate it using the --ref flag. The --aligned flag indicates reads mapped
#to the specified reference databases (i.e. your rRNA).
#The --fastx flag gives your outputs as fastq files.
#--other flag indicates reads that did not map to the specified reference databases
#(i.e. your mRNA). The --otu_map flag is a potentially useful output for taxonomic classification.
#I haven't found it useful, though it is a very small file.
#So no harm in generating it anyways!
#The -v flag indicates you want a verbose output i.e.
#a nice log to check out what percent of reads mapped and some other
#info that may be useful.

    #example using samples I've run.
#I also includes my slurm script as an example of the type of memory you're looking to allot.
#This will work for about 10 samples (in my experience) and will take about 2 hours per
#sample (This depends on how big your samples are, of course!)

#!/bin/bash
#SBATCH --partition=compute                          # Queue selection
#SBATCH --job-name=parallel_sortmerna                # Job name
#SBATCH --mail-type=ALL                              # Mail events (BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=selkassas@whoi.edu               # Where to send mail
#SBATCH --ntasks=1                                   # Run a single task
#SBATCH --cpus-per-task=36                           # Number of CPU cores per task
#SBATCH --mem=180gb                                  # Job memory request
#SBATCH --time=24:00:00                              # Time limit hrs:min:sec
#SBATCH --output=parallel_sortmerna%j.log            # Standard output/error

export OMP_NUM_THREADS=16

module load anaconda/5.1
source activate sortmerna

cd /vortexfs1/scratch/selkassas/qc-trim/raw_data/merged_files_flash/extendedFrags_for_sortmerna

rm -r /vortexfs1/home/selkassas/sortmerna/run

sortmerna --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
```

```
/silva-bac-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-bac-16s-i
d90.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/silva-arc-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-i
d95.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/rfam-5s-database-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/rfam-5.8s-datab
ase-id98.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_data
bases/silva-bac-16s-database-id85.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-d
atabase-id95.fasta --reads FS908_80_13H7_GGATGT_L001_trim_paired_merged.extendedFrags.fastq
--aligned FS908_80_13H7_GGATGT_L001_trim_paired_merged_rRNA --fastx --other
FS908_80_13H7_GGATGT_L001_trim_paired_merged_mRNA --otu_map -v

rm -r /vortexfs1/home/selkassas/sortmerna/run

sortmerna --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/silva-bac-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-bac-16s-i
d90.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/silva-arc-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-i
d95.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/rfam-5s-database-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/rfam-5.8s-datab
ase-id98.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_data
bases/silva-bac-16s-database-id85.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-d
atabase-id95.fasta --reads FS907_80_13H8_TTCAGC_L001_trim_paired_merged.extendedFrags.fastq
--aligned FS907_80_13H8_TTCAGC_L001_trim_paired_merged_rRNA --fastx --other
FS907_80_13H8_TTCAGC_L001_trim_paired_merged_mRNA --otu_map -v


#SortmeRNA code - unmerged, paired reads

    #general format:

sortmerna --ref <insert path to reference database 1> --ref <insert path to reference
database 2> --ref <insert path to reference database 3> --reads <insert forward reads>
  --reads <insert reverse reads> --aligned <insert file name_rRNA> --fastx --other <insert
file name_mRNA> --otu_map -v --out2 --paired_out

#some notes: you can list as many reference databases as you want, just make sure you
#indicate it using the --ref flag. The --aligned flag indicates reads mapped to the specified
#reference databases (i.e. your rRNA). The --fastx flag gives your outputs as fastq files.
#other flag indicates reads that did not map to the specified reference databases (i.e. your
#mRNA). The --otu_map flag is a potentially useful output for taxonomic classification. I
#haven't found it useful, though it is a very small file. So no harm in generating it
#anyways! The -v flag indicates you want a verbose output i.e. a nice log to check out what
#percent of reads mapped and some other info that may be useful. The --out2 flag outputs
#paired reads into two separate files. You can choose one of the following outputs to
#combine your files too. I used the --paired_out in the example above.

 --paired_in       BOOL        Optional  If one of the paired-end reads is Aligned,            False
                                           put both reads into Aligned FASTA/Q file
                                           Must be used with 'fastx'.
                                           Mutually exclusive with 'paired_out'.

 --paired_out      BOOL        Optional  If one of the paired-end reads is Non-aligned,        False
                                           put both reads into Non-Aligned FASTA/Q file
```

```
                                        Must be used with 'fastx'.
                                        Mutually exclusive with 'paired_in'.

    #example using samples I've worked with. I've never run this before, but this is the format
#that should work for you if you decide to use unmerged, paired reads. I also includes my
#slurm script as an example of the type of memory you're looking to allot. This will work for
#about 10 samples (in my experience).

#!/bin/bash
#SBATCH --partition=compute                          # Queue selection
#SBATCH --job-name=parallel_sortmerna                # Job name
#SBATCH --mail-type=ALL                              # Mail events (BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=selkassas@whoi.edu               # Where to send mail
#SBATCH --ntasks=1                                   # Run a single task
#SBATCH --cpus-per-task=36                           # Number of CPU cores per task
#SBATCH --mem=180gb                                  # Job memory request
#SBATCH --time=24:00:00                              # Time limit hrs:min:sec
#SBATCH --output=parallel_sortmerna%j.log            # Standard output/error

export OMP_NUM_THREADS=16

module load anaconda/5.1
source activate sortmerna

cd /vortexfs1/scratch/selkassas/qc-trim/raw_data/merged_files_flash/extendedFrags_for_sortmer
na

rm -r /vortexfs1/home/selkassas/sortmerna/run

sortmerna --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/silva-bac-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-bac-16s-i
d90.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/silva-arc-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-i
d95.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/rfam-5s-database-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/rfam-5.8s-datab
ase-id98.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_data
bases/silva-bac-16s-database-id85.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-d
atabase-id95.fasta --reads
FS908_80_13H7_GGATGT_L001_trim_paired_merged_cat_forward_reads.fastq --reads
FS908_80_13H7_GGATGT_L001_trim_paired_merged_cat_reverse_reads.fastq --aligned
FS908_80_13H7_GGATGT_L001_trim_paired_merged_rRNA --fastx --other
FS908_80_13H7_GGATGT_L001_trim_paired_mRNA --otu_map -v --out2 --paired out


rm -r /vortexfs1/home/selkassas/sortmerna/run

sortmerna --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/silva-bac-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-bac-16s-i
d90.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/silva-arc-23s-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-i
d95.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases
/rfam-5s-database-id98.fasta --ref
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/rfam-5.8s-datab
ase-id98.fasta --ref /vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_data
bases/silva-bac-16s-database-id85.fasta --ref
```

```
/vortexfs1/home/selkassas/.conda/envs/sortmerna/sortmerna/data/rRNA_databases/silva-arc-16s-d
atabase-id95.fasta --reads
FS907_80_13H8_TTCAGC_L001_trim_paired_merged.cat_forward_reads.fastq --reads
FS907_80_13H8_TTCAGC_L001_trim_paired_mergedcat_reverse_reads.fastq--aligned
FS907_80_13H8_TTCAGC_L001_trim_paired_merged_rRNA --fastx --other
FS907_80_13H8_TTCAGC_L001_trim_paired_merged_mRNA --otu_map -v --out2 --paired out
```

##6. Use fq2fa from IDBA_UD to convert .fastq output of SortMeRNA to .fasta before inputting into Mothur

```
#Convert your sample_name_rRNA.fastq files from SortMeRNA to fasta using fq2fa
#(already installed in my idba-ud_assembly_env)

conda activate idba-ud_assembly_env

    #general format:

fq2fa sample_name_rRNA.fastq sample_name_rRNA.fasta

    #Example using one of my samples:

fq2fa FS903_30_12H_AAGACG_L001_trim_paired_merged_rRNA.fastq
FS903_30_12H_AAGACG_L001_trim_paired_merged_rRNA.fasta
```

##7. Classify taxonomy of rRNA using Mothur classify.seqs

```
Running rRNA samples through mothur to classify rRNA takes a bit of preparation.

#useful links:
https://mothur.org/wiki/classify.seqs/
https://mothur.org/wiki/degap.seqs/

#First, get your mothur environment set up on conda:

conda create --name mothur
conda activate mothur
conda install -c bioconda mothur

#Download the GitHub repo and its contents:

git clone https://github.com/mothur/mothur.git

#Download the taxonomy and template files to Poseidon:

https://mothur.org/wiki/silva_reference_files/ -> scroll down to the version you want, and copy the URL of the '
by right clicking, then clicking "copy link"

#then

wget -N <insert URL> username@poseidon.whoi.edu:path/to/where/you/want/these/databases/to/go

    #As of 2/8/21 the Silva-132 taxonomy and align files are in the following
#folder in the huber lab directory:

        #taxonomy
        /vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax

        #align
        /vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.align
```

```
        #nogap align file (also known as the 'reference' or 'template' in the classify.seqs code
        /vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta

    #If you need a newer version though, you will have to re-download the files to Poseidon.

#Create a nogap version of your .align file,
#so you are able to align your rRNA without issues. Mothur has a build in command for this:

    #enter mothur

mothur

    #use degap.seqs command

degap.seqs(fasta=mothur.silva.nr_v132.align)

    #The align file is a fasta file, so no need to convert it to have the .fasta extension.
  #It will do it for you.
    #Also, you do need to enter mothur first before this command can be run!
  #If you try to run it without first entering  mothur, it will give you the error:

bash: syntax error near unexpected token `('
```

```
#I have already created the nogap file for Silva v132.
#You will only need to do this step if you downloaded a newer version of the
#silva .tax and .align files.

#Run mothur classify.seqs command to get your rRNA taxonomy! Remember to enter mothur first!

#enter mothur

mothur

    #general format:

classify.seqs(fasta=rRNA_file.fasta, template=nogap_file.align,
taxonomy=silva_taxonomy_file.tax)

    #example with one of my samples:

classify.seqs(fasta=FS891_RNA_merged_rRNA.fasta,
template=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta
,taxonomy=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax)

#Slurm script - for running a lot of samples at once:

1.) make sure you paste all of your commands in a .txt file first:

nano mothur.txt

#paste your commands (Here are some of mine as an example).
#It is best to only do five at a time. It will take almost 24 hours to run 5 samples.

classify.seqs(fasta=FS891_RNA_merged_rRNA.fasta,
template=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta,
taxonomy=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax)

classify.seqs(fasta=FS903_30_12H_AAGACG_L001_trim_paired_merged_rRNA.fasta,
template=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta,
taxonomy=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax)
```

```
classify.seqs(fasta=FS903_30_12L_GGATGT_L001_trim_paired_merged_rRNA.fasta,
template=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta,
taxonomy=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax)

classify.seqs(fasta=FS903_30_13H_CCTCGG_L001_trim_paired_merged_rRNA.fasta,
template=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta,
taxonomy=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax)

classify.seqs(fasta=FS903_30_13L_TTCGCT_L001_trim_paired_merged_rRNA.fasta,
template=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta,
taxonomy=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax)

classify.seqs(fasta=FS903_55_12H_AAGGGA_L002_trim_paired_merged_rRNA.fasta,
template=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.ng.fasta,
taxonomy=/vortexfs1/omics/huber/db/silva-db/mothur.silva.nr_v132.tax)

#2.) Write your submit script:

nano mothur.sh

#!/bin/bash
#SBATCH --partition=compute                          # Queue selection
#SBATCH --job-name=parallel_mothur                   # Job name
#SBATCH --mail-type=ALL                              # Mail events (BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=selkassas@whoi.edu               # Where to send mail
#SBATCH --ntasks=2                                   # Run a single task
#SBATCH --cpus-per-task=36                           # Number of CPU cores per task
#SBATCH --mem=100gb                                  # Job memory request
#SBATCH --time=24:00:00                              # Time limit hrs:min:sec
#SBATCH --output=parallel_mothur%j.log               # Standard output/error

export OMP_NUM_THREADS=12

module load anaconda/5.1
source activate mothur

cd /vortexfs1/scratch/selkassas/qc-trim/raw_data/merged_files_flash/extendedFrags_for_sortmer
na/rRNA_files/rRNA_fasta_files

mothur mothur.txt

#3.) Submit to slurm:

sbatch mothur.sh

#You will get these three outputs per rRNA file:
filename_rRNA.nr_v132.wang.flip.accnos
filename_rRNA.nr_v132.wang.taxonomy
filename_rRNA.nr_v132.wang.tax.summary
```

###The one you need to work with in R is the **taxonomy** file and NOT the **tax.summary** file!! It is often gb in size!

---

8. Use R to properly format Mothur output

```
#I will just show an example of this, and you can use the code to process yours.
#This ensures that the .csv file, which we will input into our count table is properly
#formatted. The count table code WILL NOT work unless this code is run first.
#input file is the taxonomy output (not the tax.summary file) file from Mothur.

##80C 2014 All Locations
```

```
install.packages(tidyverse)
library(tidyverse)

bad906 <- read.csv("~/OneDrive - Massachusetts Institute of Technology/Notebooks/R with
Sarah/80C_2014_SIP_for_figures/FS906_80_TP1_13H_AAGGGA_L006_trim_paired_merged_rRNA.nr_v132.w
ang.taxonomy.csv", header = FALSE)
bad907 <- read.csv("~/OneDrive - Massachusetts Institute of Technology/Notebooks/R with
Sarah/80C_2014_SIP_for_figures/FS907_80_13H8_TTCAGC_L001_trim_paired_merged_rRNA.nr_v132.wang
.taxonomy.csv", header = FALSE)
bad908 <- read.csv("~/OneDrive - Massachusetts Institute of Technology/Notebooks/R with
Sarah/80C_2014_SIP_for_figures/FS908_80_TP1_13H_GGATGT_L006_trim_paired_merged_rRNA.nr_v132.w
ang.taxonomy.csv", header = FALSE)

FS906 <- bad906 %>% filter(!is.na(V1)) %>%
  select(x = V1)
FS907 <- bad907 %>% filter(!is.na(V1)) %>%
  select(x = V1)
FS908 <- bad908 %>% filter(!is.na(V1)) %>%
  select(x = V1)

write.csv(FS906, "/Users/sabrinaelkassas/OneDrive\ -\ Massachusetts\
          Institute\of\Technology/Notebooks/R\with\Sarah/80C_2014_SIP_for_figures/FS906_80_TP
          1_13H_AAGGGA_L006_trim_paired_merged_rRNA.nr_v132.wang.taxonomy.NEW.csv")
write.csv(FS907,"/vortexfs1/scratch/selkassas/qc-trim/raw_data/rRNA_files/rRNA_fasta_files/ot
          her_tax/taxonomy_files_for_r/80C_2014_SIP_for_figures/FS907_80_13H8_TTCAGC_L001_tri
          m_paired_merged_rRNA.nr_v132.wang.taxonomy.NEW.csv")
write.csv(FS908, "/vortexfs1/scratch/selkassas/qc-trim/raw_data/rRNA_files/rRNA_fasta_files/o
          ther_tax/taxonomy_files_for_r/80C_2014_SIP_for_figures/FS908_80_TP1_13H_GGATGT_L006
          _trim_paired_merged_rRNA.nr_v132.wang.taxonomy.NEW.csv")
```

##9. Use R to create a count table of taxonomic groups by desired classification (Domain, Kingdom, Phylum, Class, Order, Family, Genus, Species)

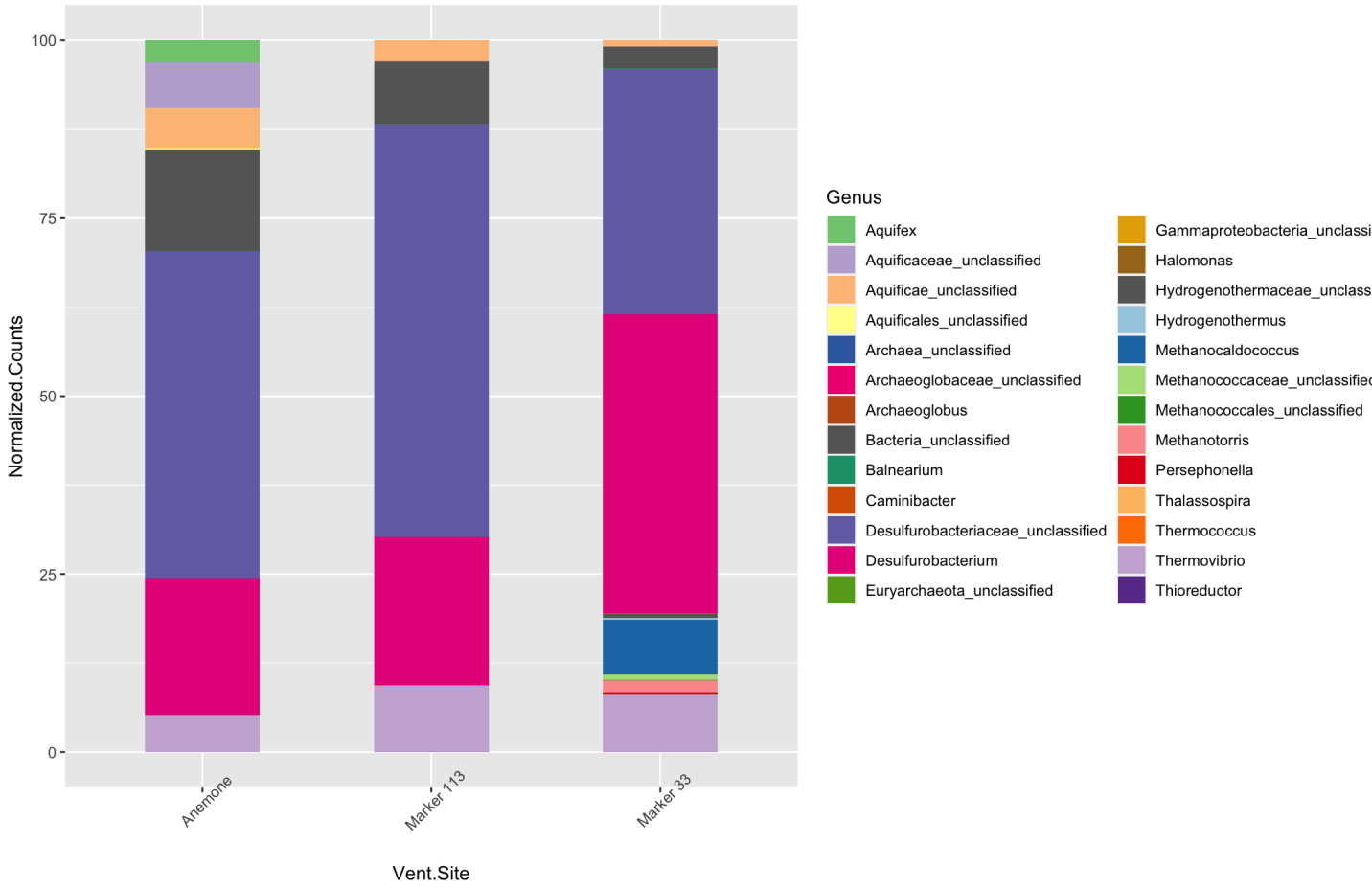
```
#Again, I will paste an example of my code and slurm script. It is easily adaptable
#to properly formatted .csv's.

#for loop to import rRNA read taxonomy assignments
##Input path to all data files
##07/30/21

#install packages
library(tidyverse)

#Set up files
#select correct files for HPC
taxa_raw <- list.files(path =
"/vortexfs1/scratch/selkassas/qc-trim/raw_data/rRNA_files/rRNA_fasta_files/other_tax/taxonomy
_files_for_r/80C_2014_SIP_for_figures", pattern =
"wang.taxonomy.NEW.csv", full.names = FALSE)

#path to files for HPC
path_data <- "/vortexfs1/scratch/selkassas/qc-trim/raw_data/rRNA_files/rRNA_fasta_files/other
_tax/taxonomy_files_for_r/80C_2014_SIP_for_figures/"

#For loop

for(a in taxa_raw){
  #import files, use paste to string together path and file names
  imported_tax <- read.csv(paste(path_data, a, sep = ""))
```

```
  #Extract sample name from "a", and split at ".wang"
  sample_names <- unlist(strsplit(a, ".wang"))
  #Modify imported data
  output_tmp <- imported_tax %>%
  #Adding in the sample name
  mutate(SAMPLE = sample_names[1]) %>%
    #filter out unknowns
    filter(!(grepl("unknown_unclassified", x))) %>%
    select(useful = x) %>%
    separate(useful, into = c("ACCESSION_NUMBER", "taxonomy"), sep = "\t") %>%
    # use regex to modify taxonomy column
    mutate(new_tax = str_replace_all(taxonomy, pattern = "\\(\\d+\\)", replacement = "")) %>%
    # parse taxonomy lineage name by semicolon
    separate(new_tax,
             into = c("Domain", "Phylum", "Class", "Order", "Family", "Genus", "Species"), sep = ";") %>%
    # add artifical count column
    add_column(COUNT = 1) %>%
    # add sample information
  group_by(sample_names[1], Phylum, Class, Order, Family, Genus, Species) %>%
    summarise(SUM = sum(COUNT))
  cat("Processing...", sample_names[1], "/n/n")
  # if else statement to facilitate row bind
  if (!exists("tax_table")){
    tax_table <- output_tmp
  } else {
    tax_table <- bind_rows(tax_table, output_tmp)
  }
  rm(output_tmp)
}

#rm(output_tmp)
#run the rm(tax_table) every time you run the for-loop so it doesn't add files twice.
#rm(tax_table)

write_delim(tax_table, file  = "output-tax-table_80C_2014_SIP.txt", delim = "\t")


####SLURM SCRIPT

#!/bin/bash
#SBATCH --partition=compute                  # Queue selection
#SBATCH --job-name=80C_2014_SIP              # Job name
#SBATCH --mail-type=ALL                      # Mail events (BEGIN, END, FAIL, ALL)
#SBATCH --mail-user=elks@mit.edu             # Where to send mail
#SBATCH --ntasks=1                           # Run a single task
#SBATCH --cpus-per-task=36                   # Number of CPU cores per task
#SBATCH --mem=100gb                          # Job memory request
#SBATCH --time=24:00:00                      # Time limit hrs:min:sec
#SBATCH --output=80C_2014_SIP.log            # Standard output/error
#export OMP_NUM_THREADS=8

module load anaconda/5.1

source activate R_environment

cd /vortexfs1/scratch/selkassas/qc-trim/raw_data/rRNA_files/
  rRNA_fasta_files/other_tax/taxonomy_files_for_r/80C_2014_SIP_for_figures

Rscript rRNA-tax-loop.R
```

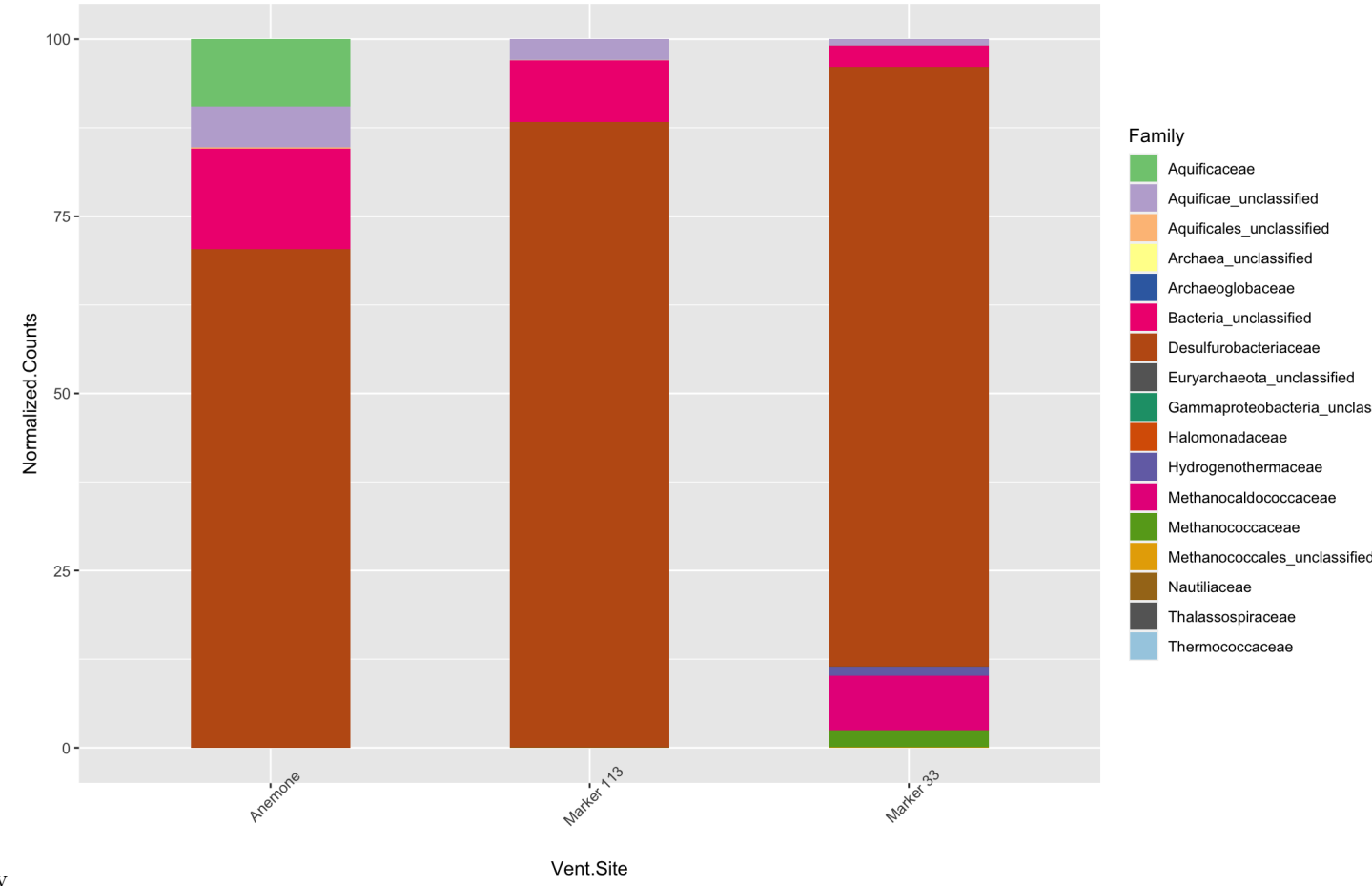
10. Use ggplot to create bar graphs

```
##Here is the code I used, after converting the taxonomy count table outputs from
#the previous step into a .csv and imposing cutoffs of counts (for example, not
#including anything that has a count less than 100)
#all plots are shown below the code.

#######GRAPH 1: SIP_80_2014_all_vents
##Read in the data table

library(tidyverse)
library(reshape2)
library(RColorBrewer)
library(ggplot2)

#read in the count table
SIP_80_2014_tax_table <-
  read.delim("/Users/sabrinaelkassas/Desktop/output-tax-table_80C_2014_SIP_normalized.txt",
             header = TRUE, sep = "\t")

#converting to a formatted .csv, so I can impose cutoffs

write.csv(SIP_80_2014_tax_table,
/Users/sabrinaelkassas/Desktop/output-tax-table_80C_2014_SIP.csv")
```

```
#choose enough colors for the plot
n <- 16
qual_col_pals = brewer.pal.info[brewer.pal.info$category == 'qual',]
col_vector = unlist(mapply(brewer.pal, qual_col_pals$maxcolors, rownames(qual_col_pals)))
pie(rep(1,n), col=sample(col_vector, n))

#ggplot code - 80C 2014
library(ggplot2)
ggplot(SIP_80_2014_tax_table, aes(x = Vent.Site, y = Normalized.Counts, fill = Family)) +
  geom_bar(stat = "identity", width = 0.5) + theme(axis.text.x.bottom = element_text(angle =
  45)) + scale_fill_manual(values=col_vector)
```

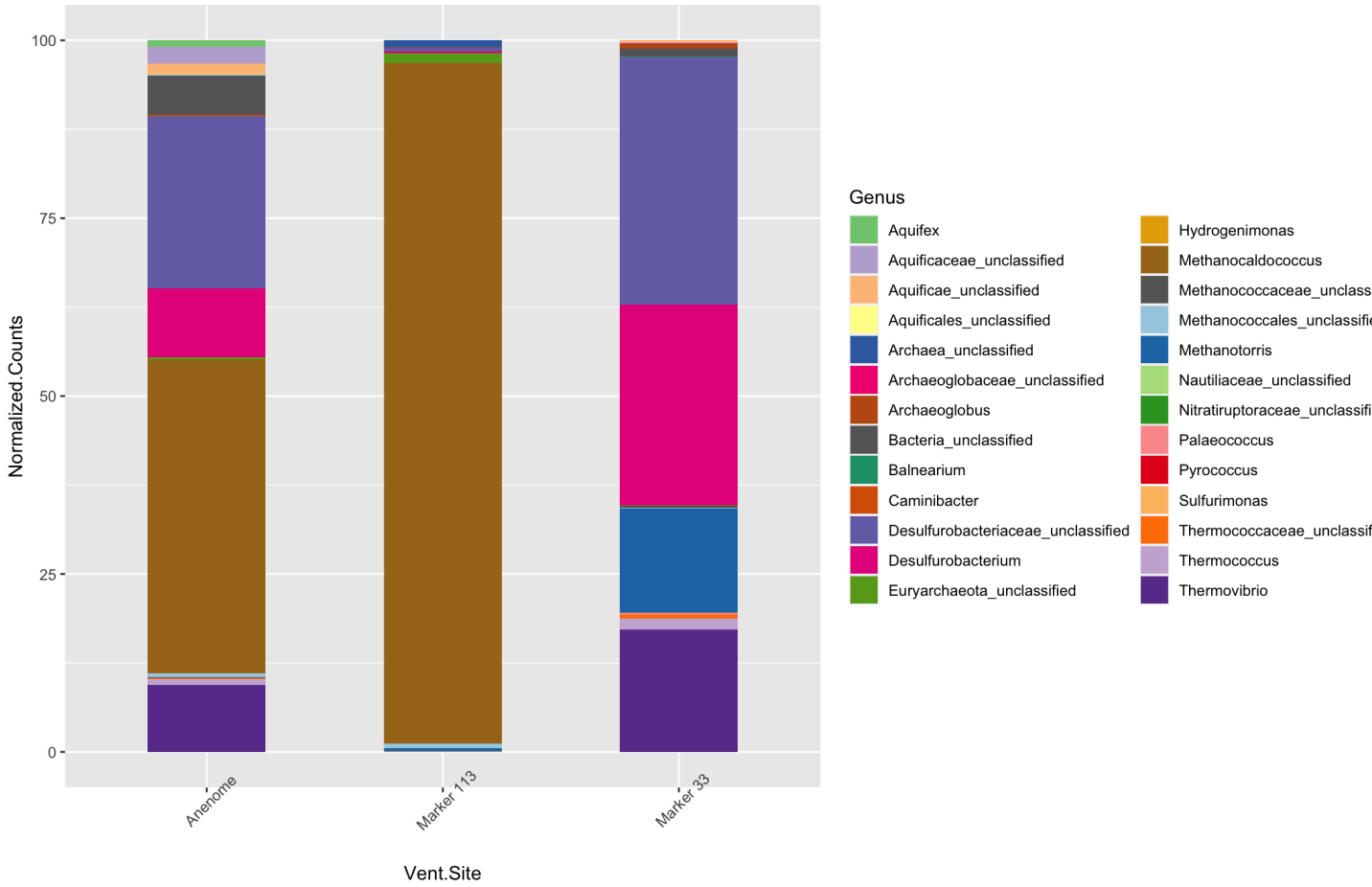


#by genus

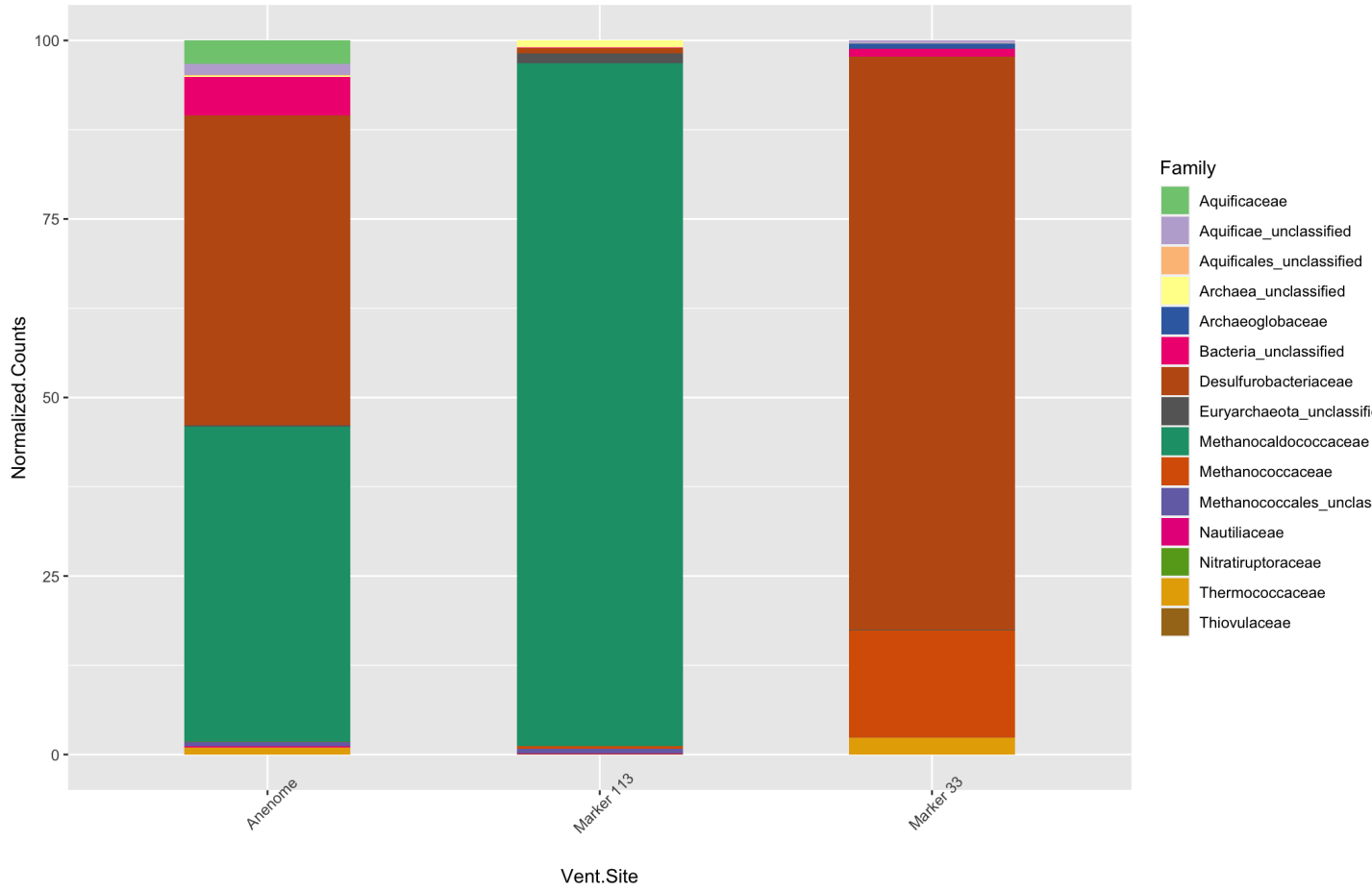


#by family

```
#ggplot code - 80C 2013
library(tidyverse)
library(ggplot2)
library(RColorBrewer)
display.brewer.all()

#read in the count table
SIP_80_2013_tax_table <-
  read.delim("/Users/sabrinaelkassas/Desktop/output-tax-table_80C_2013_SIP_normalized.txt",
  header = TRUE, sep = "\t")

#converting to a formatted .csv, so I can impose cutoffs
write.csv(SIP_80_2013_tax_table,
          "/Users/sabrinaelkassas/Desktop/output-tax-table_80C_2013_SIP.csv")

#selecting enough colors for the plot
n <- 15
qual_col_pals = brewer.pal.info[brewer.pal.info$category == 'qual',]
col_vector = unlist(mapply(brewer.pal, qual_col_pals$maxcolors, rownames(qual_col_pals)))
pie(rep(1,n), col=sample(col_vector, n))

#ggplot code
ggplot(SIP_80_2013_tax_table, aes(x = Vent.Site, y = Normalized.Counts, fill = Genus)) +
    geom_bar(stat = "identity", width = 0.5) +
  theme(axis.text.x.bottom = element_text(angle = 45)) +
  scale_fill_manual(values=col_vector)

#optional title code
ggtitle("Axial Seamount, 80C Year 2013") +
  theme(plot.title = element_text(face = "bold", colour = "black", size = 13))
```

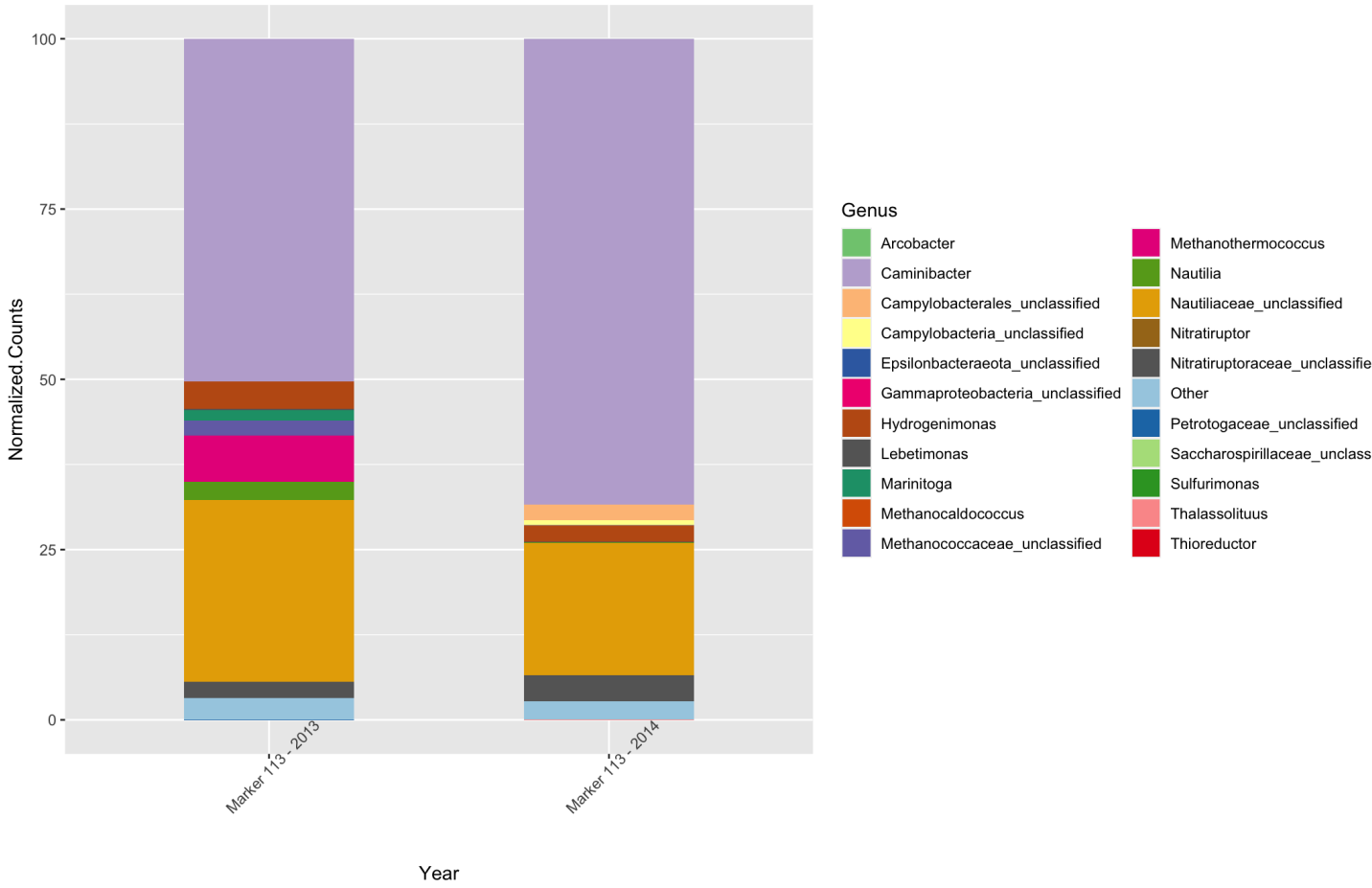


#by genus

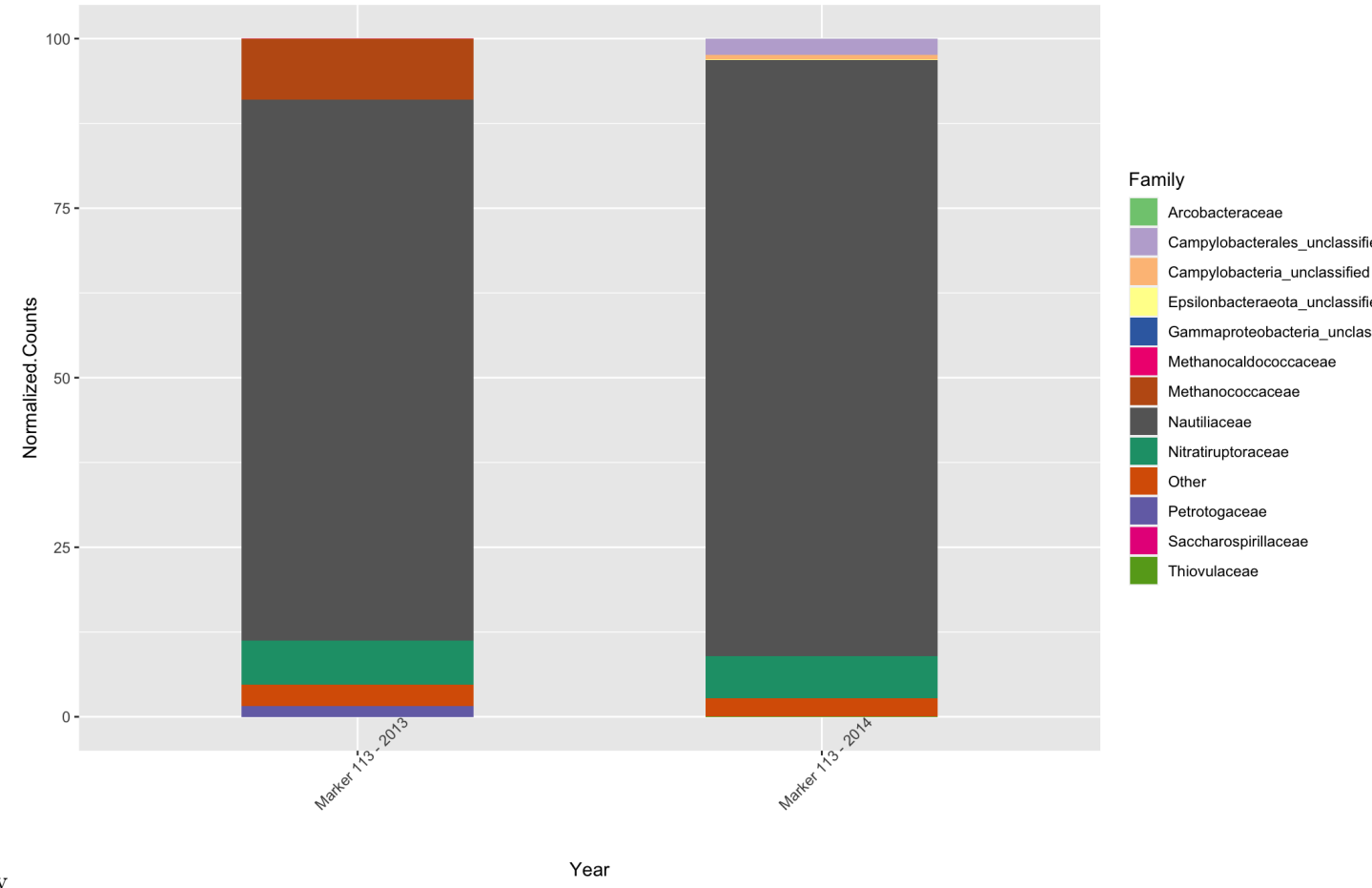


#by family

```
#ggplot code - Marker33 2013, 2014 55C

library(tidyverse)
library(ggplot2)
library(RColorBrewer)
display.brewer.all()

#read in the count table
SIP_M33_2013_2014_tax_table <-
  read.delim("/Users/sabrinaelkassas/Desktop/output-tax-table_Marker33_2013_2014_55_normaliz
             ed.txt", header = TRUE, sep = "\t")

#converting to a formatted .csv, so I can impose cutoffs
write.csv(SIP_M33_2013_2014_tax_table,
          "/Users/sabrinaelkassas/Desktop/output-tax-table_Marker33_2013_2014_55.csv")

#select enough colors for the plot
n <- 10
qual_col_pals = brewer.pal.info[brewer.pal.info$category == 'qual',]
col_vector = unlist(mapply(brewer.pal, qual_col_pals$maxcolors, rownames(qual_col_pals)))
pie(rep(1,n), col=sample(col_vector, n))

#ggplot code
ggplot(SIP_M33_2013_2014_tax_table, aes(x = Year, y = Normalized.Counts, fill = Family)) +
  geom_bar(stat = "identity", width = 0.5) +
  theme(axis.text.x.bottom = element_text(angle = 45)) +
  scale_fill_manual(values=col_vector)
```

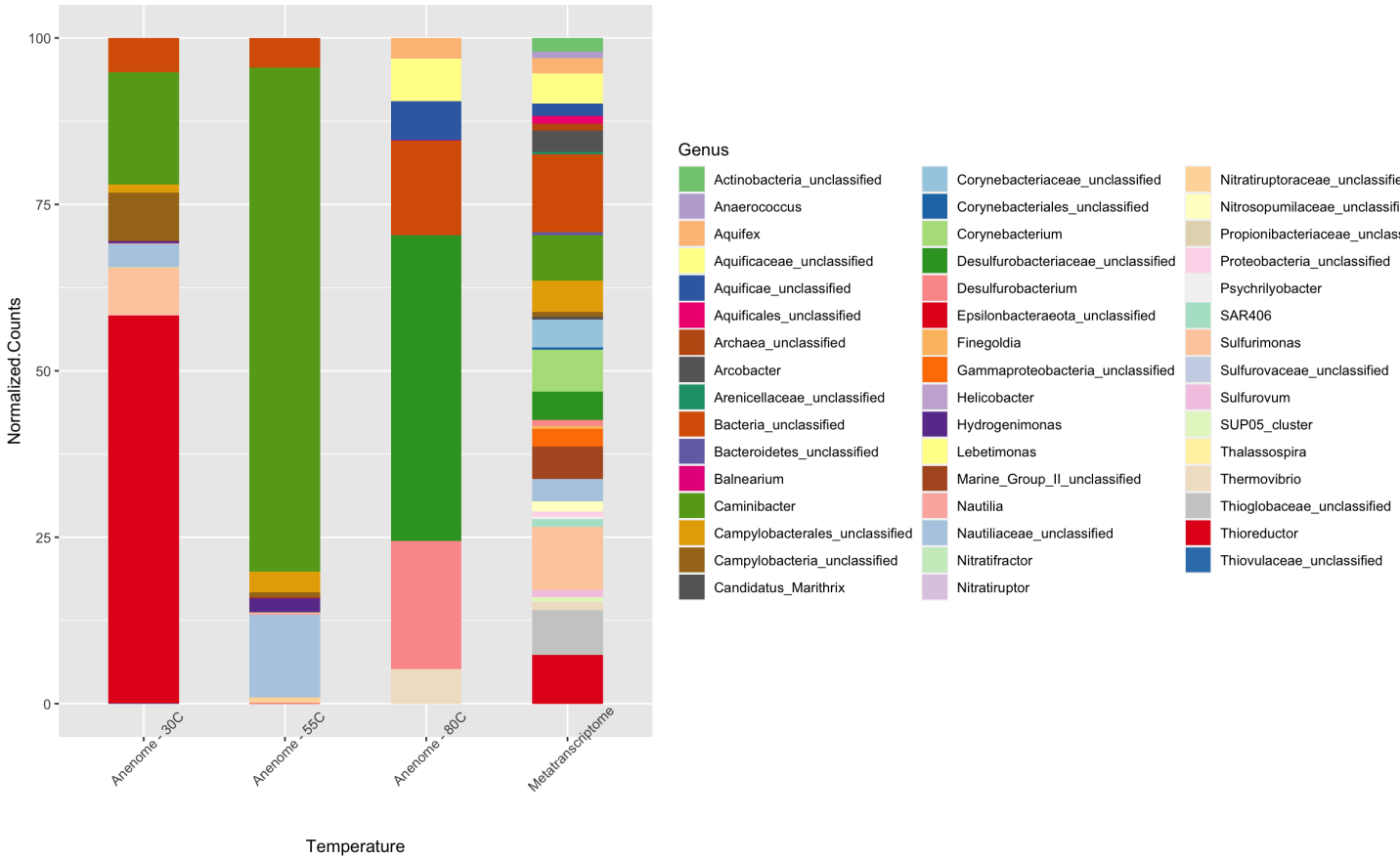


#by genus

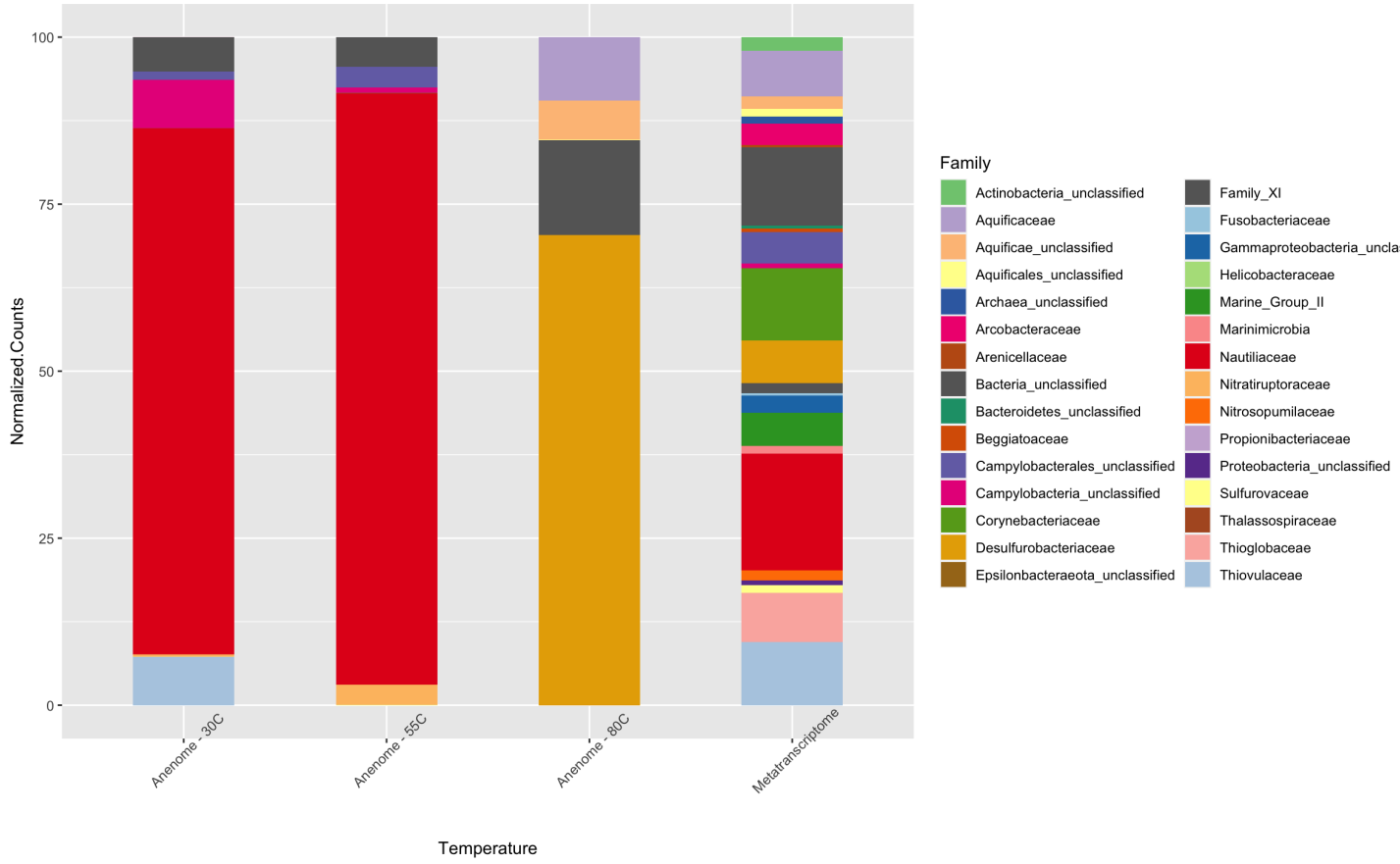


#by family

```
#ggplot code - Anemone All Temps

library(tidyverse)
library(ggplot2)
library(RColorBrewer)
display.brewer.all()

#read in the count table
SIP_Anemone_2014_tax_table <-
  read.delim("/Users/sabrinaelkassas/Desktop/output-tax-table_Anemone_all_temps_2014_normali
             zed.txt", header = TRUE, sep = "\t")

#converting to a formatted .csv, so I can impose cutoffs
write.csv(SIP_Anemone_2014_tax_table,
          "/Users/sabrinaelkassas/Desktop/output-tax-table_Anemone_all_temps_2014.csv")

#choose enough colors for the plot
n <- 10
qual_col_pals = brewer.pal.info[brewer.pal.info$category == 'qual',]
col_vector = unlist(mapply(brewer.pal, qual_col_pals$maxcolors, rownames(qual_col_pals)))
pie(rep(1,n), col=sample(col_vector, n))

#ggplot code
ggplot(SIP_Anemone_2014_tax_table,
   aes(x = Temperature, y = Normalized.Counts, fill = Family)) + geom_bar(stat =
  "identity", width = 0.5) +
  theme(axis.text.x.bottom = element_text(angle = 45)) +
  scale_fill_manual(values=col_vector)
```



#by genus


Normalized.Counts
100
75
50
25
0
Anenome - 30C
Anenome - 55C
Anenome - 80C
Metatranscriptome
Temperature
Family
Actinobacteria_unclassified
Aquificaceae
Aquificae_unclassified
Aquificales_unclassified
Archaea_unclassified
Arcobacteraceae
Arenicellaceae
Bacteria_unclassified
Bacteroidetes_unclassified
Beggiatoaceae
Campylobacterales_unclassified
Campylobacteria_unclassified
Corynebacteriaceae
Desulfurobacteriaceae
Epsilonbacteraeota_unclassified
Family_XI
Fusobacteriaceae
Gammaproteobacteria_uncla
Helicobacteraceae
Marine_Group_II
Marinimicrobia
Nautiliaceae
Nitratiruptoraceae
Nitrosopumilaceae
Propionibacteriaceae
Proteobacteria_unclassified
Sulfurovaceae
Thalassospiraceae
Thioglobaceae
Thiovulaceae

#by family

“‘