ace03_annotate_DESeq2_full

August 17, 2018

In [1]:
```
# import Biopython SeqIO package
from Bio import SeqIO
```

In [2]:
```
# opening files
in_genbank = "./data/b_burgdoreferi_B31_genome.gbk"
in_deseq_res = "./data/sb01_ph68C_ph76C_diffs.txt"
out_anno_res = open("./data/sb01_ph68C_ph76C_diffs_anno.txt", 'w')
```

In [ ]:
```
# write header for outfile
out_anno_res.write("locus_tag\tgene\tproduct\tbaseMean\tlog2FoldChange\tlfcSE\tstat\tpv
```

In [ ]:
```
# extract gene names and gene products, populate dicts
gene_dict = {}
product_dict = {}

# parse genbank file, populate dicts with gene names and products
with open(in_genbank):
    full_record = SeqIO.parse(in_genbank, "genbank")
    for record in full_record:
        # investigate record object, tab autocompletion
        #print(record)
        #print(record.description)

        # loop though each feature  of each record
        for feature in record.features:
            # investigate feature object
            #print(feature)

            # only look at coding sequences
            if "CDS" in feature.type:
                # extract locus_tags , products
                locus_tag = feature.qualifiers.get('locus_tag')[0]
                product = feature.qualifiers.get('product')[0]

                # extract protein sequence for specific gene
                if locus_tag == "BB_0002":
                    print(feature)
```

```
                            # grab coordinates
                            start = feature.location.start
                            end = feature.location.end

                            # extract nucleotides
                            BB_0002_nt = record.seq[start:end]

                            # translate
                            #print(BB_0002_nt.translate())
                            #print(BB_0002_nt.reverse_complement())
                            #print(BB_0002_nt.reverse_complement().translate())

                            # methods for extraction
                            BB_0002_nt = feature.extract(record).seq
                            BB_0002_aa = feature.qualifiers.get('translation')[0]

                            print(">{}_nucleotide\n{}\n".format(locus_tag, BB_0002_nt))
                            print(">{}_amino_acid\n{}\n".format(locus_tag, BB_0002_aa))


                        # populate dictionaries
                        if feature.qualifiers.get('gene'):
                            gene = feature.qualifiers.get('gene')[0]
                        else:
                            gene = "NA"
                        product_dict[locus_tag] = product
                        gene_dict[locus_tag] = gene
```

In [5]:
```
# add gene name and gene product to results
with open(in_deseq_res) as f:
    # skip the header
    next(f)
    for line in f:
        line = line.rstrip("\n")
        locus_tag, baseMean, log2FoldChange, lfcSE, stat, pvalue, padj = line.split("\t
        if locus_tag in product_dict.keys():
            out_anno_res.write("{0}\t{1}\t{2}\t{3}\t{4}\t{5}\t{6}\t{7}\t{8}\n".format(l
        else:
            out_anno_res.write("{0}\tNA\tNA\t{1}\t{2}\t{3}\t{4}\t{5}\t{6}\n".format(loc

out_anno_res.close()
```