Max Cheng

Work Sample Portfolio

Bioinformatics and Genome Analysis

Phone: 408-908-0395

Email: mcheng30@jh.edu

LinkedIn: www.linkedin.com/in/maxcheng98

GitHub: https://github.com/mbcheng88

Table of Contents

[Introduction 4](#_Toc105955057)

[Genomic Records 5](#_Toc105955058)

[ENSEMBL/BioMart/BioMaRt 6](#_Toc105955059)

[Ensembl 6](#_Toc105955060)

[BioMart 7](#_Toc105955061)

[BioMaRt 7](#_Toc105955062)

[Gene Annotation and Prediction 9](#_Toc105955063)

[Prokaryotic Gene Annotation Pipeline 9](#_Toc105955064)

[GLIMMER 10](#_Toc105955065)

[FGENESB 11](#_Toc105955066)

[Eukaryotic Gene Prediction 12](#_Toc105955067)

[Splign 13](#_Toc105955068)

[Genomic BLAST 14](#_Toc105955069)

[BLAT 14](#_Toc105955070)

[Genomic File Types 15](#_Toc105955071)

[BED (browser extension data) files 15](#_Toc105955072)

[WIG (Wiggle Format) files: 16](#_Toc105955073)

[SAM (Sequence Alignment Map) files 17](#_Toc105955074)

[BAM files 17](#_Toc105955075)

[Genome Browsers 18](#_Toc105955076)

[UCSC Genome Browser 18](#_Toc105955077)

[IGV (Integrated Genome Viewer) 20](#_Toc105955078)

[NCBI Variation Viewer 20](#_Toc105955079)

[Galaxy 22](#_Toc105955080)

[BEDtools 22](#_Toc105955081)

[NGS/Genome Assembly/Variant Calling Pipeline 24](#_Toc105955082)

[ChIP-seq and RNA-seq Pipeline 26](#_Toc105955083)

[SNP Research Paper 30](#_Toc105955084)

[Computational Programs and Databases 48](#_Toc105955085)

[SQL 48](#_Toc105955086)

[Python 49](#_Toc105955087)

[Circle “Program” 49](#_Toc105955088)

[DNA Length and Start Codon Program 49](#_Toc105955089)

[Motif Counter 50](#_Toc105955090)

[FASTA Sequence Counter 50](#_Toc105955091)

[Parsing Program 51](#_Toc105955092)

[Genome Analysis Program 51](#_Toc105955093)

[Angle Converter 52](#_Toc105955094)

[DNA Reverse Complement Program 53](#_Toc105955095)

[GFF Feature Exporter 54](#_Toc105955096)

[FASTQ Analysis Tool 56](#_Toc105955097)

[Docker/Biocontainers 64](#_Toc105955098)

[Cromwell and WDL 64](#_Toc105955099)

[Jupityr Notebooks 64](#_Toc105955100)

[Acknowledgements 65](#_Toc105955101)

# Introduction

Hi, my name is Max Cheng, and I am a master’s student in Biotechnology with a concentration in Bioinformatics at Johns Hopkins University. My main background is in the biological field, and I recently graduated with a M.S. in NPB (Neurobiology, Physiology, and Behavior) at UC Davis. However, most of my work experience is focused along data analysis and using programs. I’ve been entering financial data at my family-owned accounting firm since I was young, and during university I was a data analysis research assistant at UC Davis MIND Institute’s AIR (Attention, Impulsivity, and Response) ADHD lab. Accounting involved a lot of copying data from one source to another, so I always wanted to automate that process to expedite the work. At the AIR lab, I worked with python code that took fMRI data and prepared it for data analysis and well as designing a VR classroom on the HTC VIVE using Unity. So, to mesh my academic and work experiences, I sought to work towards the goal of becoming a bioinformatics scientist, where I can use my extensive background on biology and translate them into tools and pipelines used by modern biotechnology companies. Thus, I’ve been slowly transitioning to writing programs and performing data analysis on biological data such as genome sequencing. Through both academics and extracurriculars, I’ve learned how to write programs with Python, gather and analyze data with R, use the Unix terminal to read and transfer files, and database using SQL.

Now that my university experience is wrapping up, I hope to retain much of what I have learned as a student, so that I can perpetuate these ideas and concepts out into the workforce. Hence, I have created a sample work portfolio that goes through some of my most recent projects at Johns Hopkins, including tools I’ve used and the programs I’ve written to analyze genomic data. I hope that this compendium will not only serve as a useful reference to build upon, but an aggregation of what I am capable of as a bioinformatics scientist.

# Genomic Records

INSDC (International Nucleotide Sequence Database Collection)

There are 3 main databases with a centralized annotation system:

* + Genbank (by NCBI)
  + ENA (European Nucleotide Archive)
  + DDBJ (DNA Data Bank of Japan)

*Example record from NCBI*

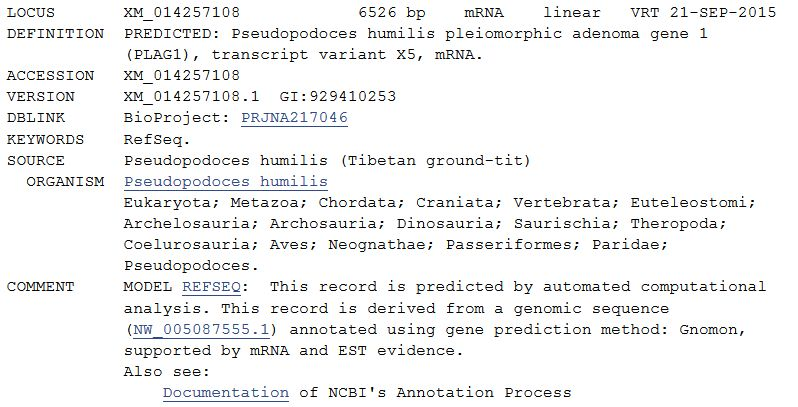


image from http://www.ncbi.nlm.nih.gov/nuccore/XM\_014257108.1

These databases will serve as a basis for most of the genomic data being analyzed. Additional databases will be used as a resource for specific organism genomes such as <http://www.wormbase.org> for nematodes and <https://yeastmine.yeastgenome.org> for fission yeast genomes.

## ENSEMBL/BioMart/BioMaRt

### Ensembl

A genome browser for chordates

* You can search specific genes and species-specific indexes
* Ensembl’s data is stored MySQL relational databases allowing for easy access through SQL queries though a terminal or other related program

Timeline

Description automatically generated with medium confidence

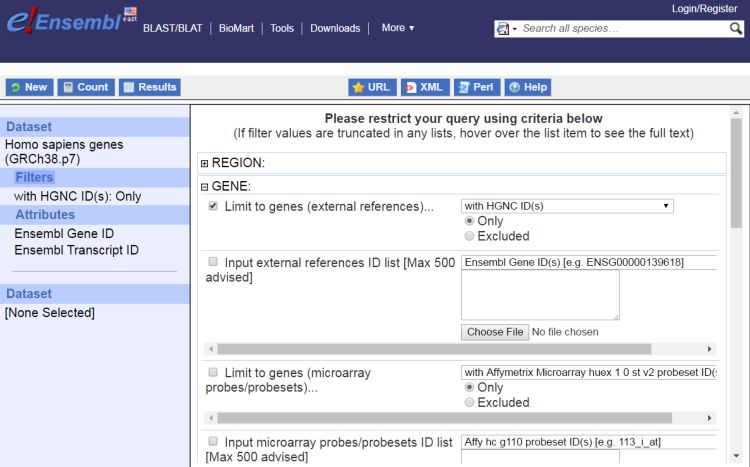
*Figure above shows the Gene Summary for the CCL4 gene*

### BioMart

Allows users to download datasets from ENSEMBL

* Many filters can be applied such as Region, Gene, Phenotype, Gene Ontology, Multi Species Comparisons, Protein Domains and Family, and Variant
* Additionally, you can filter further by Attributes: Gene, External, and Protein Domains and Family

Sample query from the browser version of Ensembl:



https://m.ensembl.org/info/data/biomart/index.html

### BioMaRt

A Bioconductor package that integrates bioinformatics tools into R

* Does the same thing as BioMart but can be used in a terminal or R to view datasets from ENSEMBL

*Code in R I used to pull a specific query from the ENSEMBL database using BioMaRt shown below. The identifiers were pulled from OMIM by searching at OMIM.org for Huntington’s disease. Then these values: 603218, 604802, 143100, 606438 are used in the getBM command to find matching ENSEMBL gene ID’s:*

A screenshot of a computer screen

Description automatically generated with medium confidence

*The results of the R query:*

A computer screen capture

Description automatically generated with low confidence

# Gene Annotation and Prediction

## Prokaryotic Gene Annotation Pipeline

**There are a couple of tools that can help predict bacterial genes:**

**ORF Finder**

Finds open reading frames in a DNA sequence (along with its corresponding protein sequence) in FASTA format. Usually used to find the CDSs (longest ORFs) in a prokaryotic genome. This SMART, BLAST, or regular BLASTP can now be run to verify the predicted protein.

Graphical user interface, application

Description automatically generated

https://www.ncbi.nlm.nih.gov/orffinder/

### GLIMMER

Unix-based program or finding genes in microbial DNA. Has ability to find long ORFs and can be used to build a training set for genome analysis.

Unix commands used to generate an ORF .predict file using a full genome (sheli.fasta) to train the dataset, and predicts the ORF of the partial genome (sheliprt.fasta):

[mcheng30@bfx3 FASTAS]$ long-orfs -n -t 1.15 sheli.fasta sheli.longorfs

[mcheng30@bfx3 FASTAS]$ extract -t sheli.fasta sheli.longorfs > sheli.train

[mcheng30@bfx3 FASTAS]$ build-icm -r sheli.icm < sheli.train

[mcheng30@bfx3 FASTAS]$ glimmer3 -o50 -g110 -t30 sheliprt.fasta sheli.icm sheliprt

[mcheng30@bfx3 FASTAS]$ extract -t sheliprt.fasta sheliprt.predict >sheliprt.glimmer

Predict file with the ORFs:

>Spiroplasma helicoides strain TABS-2, partial sequence

orf00001 635 991 +2 2.77

orf00003 1154 1312 +2 0.42

orf00004 1334 1978 +2 7.53

orf00005 2242 2463 +1 5.70

orf00007 2585 4003 +2 11.22

orf00008 4034 4678 +2 7.77

orf00009 4880 5143 +2 7.72

### FGENESB

Annotates of bacterial genomes using Genbank based on Markov Chain Models that is very useful in finding operon and predicting genes in bacteria.

Same dataset (sheliprt.fasta) as the GLIMMER example but run through FGENESB/softberry instead. *Results of the query are shown below.*

Text

Description automatically generated

From: http://www.softberry.com/berry.phtml?topic=fgenesb&group=programs&subgroup=gfindb

## Eukaryotic Gene Prediction

Expression-based predictions align cDNA to define CDS' which is very accurate but with limitations such as cDNA obtainability

* Content sensors: prediction algorithms that depend on coding sequence features such as nucleotides.
  + Hexamer frequency most used as a content sensor using HMM training sets and probability to predict coding regions in unknown DNA
* Signal Sensors: finds genes based on specific sequence associated with a gene such as promotors, splice sites, stop codons
  + Uses the PSSM (weight matrix) to find conserved positions. Neural networks can also be used through AI training sets.

cDNA's can be aligned to the genome revealing the location of the gene and exon pattern

Programs that use this technique include:

### Splign

Hosted by NCBI to align cDNA and genomic DNA and supports cross species alignments with cDNA input. *Results of an alignment where CDS and mRNA locations can be predicted in the figures below:*

Table

Description automatically generated

Text

Description automatically generated

https://www.ncbi.nlm.nih.gov/sutils/splign/splign.cgi?textpage=online&level=form

### Genomic BLAST

When cDNA is not available so uses EST seq (partial cDNA seq). Main search menu shown below.

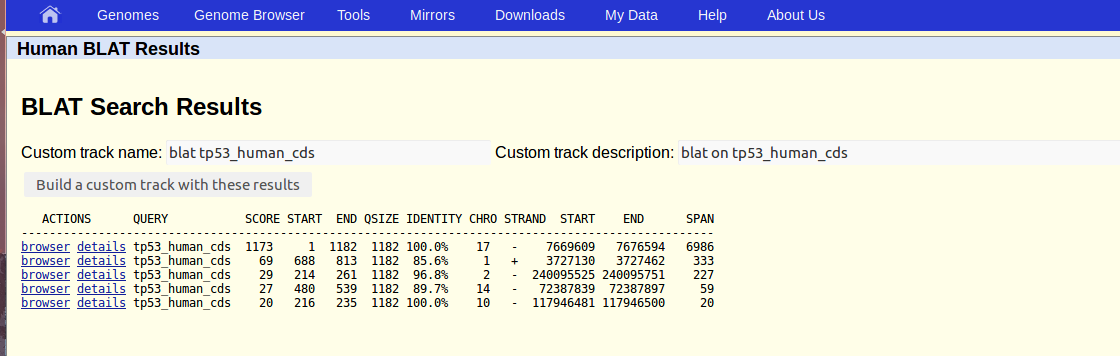
Graphical user interface, text, application, email

Description automatically generated

https://blast.ncbi.nlm.nih.gov/Blast.cgi?PROGRAM=blastp&PAGE\_TYPE=BlastSearch&LINK\_LOC=blasthome

### BLAT

Aligns mRNA sequence to a genome without the uses of genomic DNA. *Figure below shows the human TP53 mRNA aligned with UCSC BLAT.*



https://genome.ucsc.edu/cgi-bin/hgBlat

# Genomic File Types

BED, WIG, SAM, BAM file formats

## BED (browser extension data) files

Stores info on genomic intervals such as exons which have an "interval" from the start to end position

BED3: 3 column table with chromosome number, start, and end positions

Can be zero based position like UCSB genome browser where zero based systems have nucleotides between numbers so "start" is one off "true

BED 6: chrom, start, end, name, score, strand

chr1 23401234 23401496 promoter 0 +

BED 12: chrom, start, end, name, score, strand, thickStart, thickEnd, itemRgb, blockCount, blockSizes, blockStarts

chr1 11873 14409 gene\_1 0 + 11873 11873 0 3 354,109,1189, 0,739,1347,

Below is a BED6 file I created showing the transcription factor DdlR binding to the promoter region of the ddlR-ddl operon in Brevibacillus brevis.

Graphical user interface, application

Description automatically generated

*And this is the same BED file loaded into NCBI Genome Browser as a user custom track showing the promoter, 5' UTR region, and CDS of ddlR:*

Timeline

Description automatically generated with medium confidence

https://www.ncbi.nlm.nih.gov/variation/view

## WIG (Wiggle Format) files:

Stores continuous data across large genomic locations.

Declaration line followed by chromosome position then data value

variableStep chrom=chr2

300701 12.5

300702 12.5

300703 12.5

300704 12.5

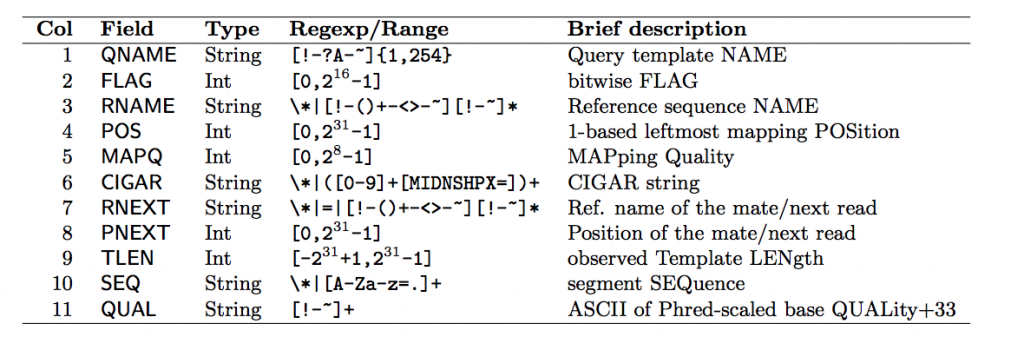
300705 12.5

BED and WIG files can be imbedded to the genome browser of your choosing such as NCBI variation viewer, UCSB genome browser, IGV, etc.

## SAM (Sequence Alignment Map) files

Contains info on how sequences align to reference genome mainly results of NGS

There are 11 mandatory fields that represent a single alignment event:



https://learn.gencore.bio.nyu.edu/ngs-file-formats/sambam-format/

## BAM files

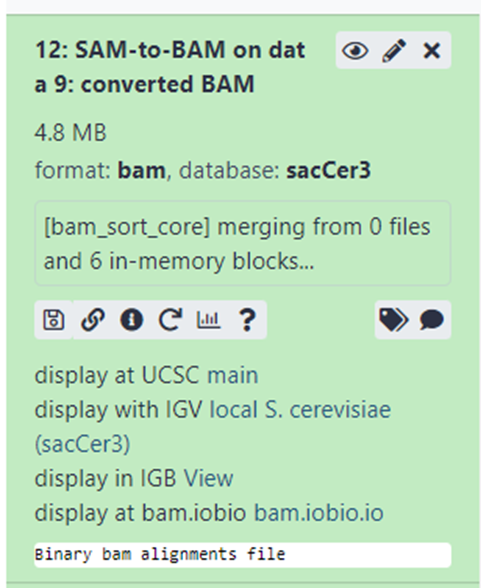
A compressed binary version of a SAM file that is used to store aligned sequences up to 128MB

Text

Description automatically generated

*https://support.illumina.com/help/BS\_App\_RNASeq\_Alignment\_OLH\_1000000006112/Content/Source/Informatics/BAM-Format.htm*

SAM and WIG files can be converted and processed using SAMtools and BEDtools from Galaxy:



*Saccharomyces cerevisiae reference genome aligned with a SAM file and converted from a SAM to a BAM file via the SAM-to-BAM tool in Galaxy.*

# Genome Browsers

One of the most important tools for visualizing data from both databases and user created datasets.

## UCSC Genome Browser

Primarily an online eukaryotic genome browser that includes assembly data, precomputed comparative genomic data, mRNA, EST, and RefSeq gene alignments, and links to NCBI Map Viewer and Ensembl

Tracks

* UCSC genes: Gene prediction set based on RefSeq, Genbank, and UniProt
* Conservation: View conservation levels between 100 species
* PhastCons: Genome wide alignments between human, mouse, dog, zebrafish, fly using blastz

I personally used this browser a lot because it includes a table browser that allows you to directly input datasets and set search parameters at https://genome.ucsc.edu/cgi-bin/hgTables:

Graphical user interface, text, application, email

Description automatically generated

*When investigating human TUBB3 gene and looking for what histone modifications appear near the transcription start site of the TUBB3 gene, the UCSC Genome Browser was a useful tool to assist with data visualization shown the figure below:*

Timeline

Description automatically generated with medium confidence

## IGV (Integrated Genome Viewer)

Genome browser that is available as local download outside of web browser that allows you to input your own datasets such as BED and BEDGRAPHS as user inputted tracks or downloaded datasets of websites (URL) and servers.

Graphical user interface, application

Description automatically generated

*The figure above shows the EPHX1 gene in homo sapiens on the hg19 genome*

## NCBI Variation Viewer

A genome browser that focuses on the human genome that is especially useful to view, search, and navigate variations housed in dbSNP, dbVar, and ClinVar in genomic context.

When researching the CCL4 gene and creating a BED file to find the transcription start site, TATA box, and a DNA location of the palindromic sequence, I used the Variation Viewer to align my used inputted BED files as tracks to compare with known dbSNPs.

Graphical user interface

Description automatically generated with low confidence

*Locations of the transcription start site and TATA box in relation to known SNPs*

Timeline

Description automatically generated

*Palindromic sequence in the CCL4 gene and the known SNPs shown on the track below*

# Galaxy

Galaxy is one of the ultimate genomic tools that allows you to design, execute and visualize an entire genomic pipeline. It is a publicly available web service and package that allows for data analysis with over 100 tools.

* Allows for auto detection of files and has innate conversion capabilities
* Galaxy has data libraries where you can publish and share data/results
* Has a collaborative input (usually user files or from UCSC table browser) and output system (IGV, IGB, UCSC genome browser, ENSEMBL, vcf.iobio) and much more

## BEDtools

One very useful tool that I have used is the BEDtools intersect feature which allows you to take two datasets and find intersects.

For example, when I was researching SNPs in the CFL2 gene (in cows), I used Galaxy’s BEDtool to intersect the SNPs (in the form of a user created BED file) with the coding exons after they have matched with a BED file containing all the exons in CFL2

*This workflow is shown in the Galaxy history below along with the results of the intersection:*

Graphical user interface, text, application, chat or text message

Description automatically generated

*It was shown that 3 SNP’s intersect with all exons, and 2 SNP’s intersect with coding exons*

Similarly, to Biomart, BEDtools can also be run from the command line:

*Here is example of the code I used to intersect two BED files on from a H3K4me3 cell line versus a reference genome:*

$bedtools intersect -u -a hs\_chr20\_H3K4me3\_for\_linux.bed -b hs\_chr20\_refseq.bed

$bedtools intersect -u -a hs\_chr20\_H3K4me3\_for\_linux.bed -b hs\_chr20\_refseq.bed | wc -l

$bedtools intersect -v -a hs\_chr20\_H3K4me3\_for\_linux.bed -b hs\_chr20\_refseq.bed

$bedtools intersect -v -a hs\_chr20\_H3K4me3\_for\_linux.bed -b hs\_chr20\_refseq.bed | wc -l

## NGS/Genome Assembly/Variant Calling Pipeline

**NGS Technologies**

NGS data can be used to explore metagenomics (the study of genetic materials from environmental samples).

The main technologies that are used are Illumina, ABI Solid, Ion torrent, Pacific, and Oxford

**Genome Sequence Assembly**

De novo Based assembly

Overlapping fragments assembled from scratch typical of new genomes

* There is usually a read length and error rate tradeoff for example PacBio has a long lead but high error rate. Vice versa for Illumina.
* Some examples of de novo assemblers are graphical approaches and Velvet

**Reference Based Assembly**

Useful for assembling human/cancer genomes based of the human hg38 genome. Alignment of trimmed data to FASTQ data to a reference genome is required.

* Some examples of aligners include: Bowtie, BWA, and HISAT

**Variant Calling**

Aligns NGS reads to a reference genome

* Sequencing errors, Heterozygous SNP, Homozygous SNP, and PCR amplification errors can cause mismatches
* Better coverage in sequencing helps variant calling easier to tell if this is a sequencing error
* The output is in the VCF format that shows the chromosomal position, the expected reference nucleotide, and the actual sequenced nucleotide, among many other statistics

Examples include: Freebayes, SAM tools, GATK, Platypus

**Full Pipeline**

Galaxy can execute this entire workflow with an input FASTQ file and produces a VCF file. The full pipeline is as follows:

FASTQ Groomer: Reformat the FASTQ file(s) to Sanger/Illumina 1.9 encoding.

FASTQ Trimmer or Trimmomatic: Trim low quality regions from FASTQ file(s).

Bowtie2/BWA/HISAT: Align reads in FASTQ file to a reference genome.

SAM-to-BAM: Convert output SAM file(s) to BAM file(s) if necessary.

Filter SAM or BAM (optional): Limit BAM file to a chromosome or a genomic region.

Sort BAM (optional): Sorts BAM file(s) in order based on chromosomal positions; may not be needed as some tools automatically sort BAM.

FreeBayes: Use aligned BAM file and reference genome to call variants; produces VCF file

This pipeline will be featured in my SNP paper later in this portfolio to find a specific SNP from a FASTQ file from the 1000 genomes database.

## ChIP-seq and RNA-seq Pipeline

**ChIP-seq**

Stands for Chromatin immunoprecipitation sequencing

* ChIP experiment is followed by next-gen sequencing, aligned to a reference genome, and regions with man reads are histone associated regions
* Data reported in the FASTQ format and are deposited in the Sequence Read Archive (SRA) and/or the Gene Expression Omnibus (GEO)
* BWA and Bowtie allows for the alignment to the reference genome out puts a SAM file
* MACS2 and Slicer now do peak calling which indicates which regions of DNA are associated with the protein that was immunoprecipitated. Uses bam file so SAM file may need to be converted.

**RNA-seq**

Also known as massively parallel cDNA sequencing used to defines transcription start sites, reveal undiscovered small noncoding RNAs and long noncoding RNAs

* Used to study transcriptomes of organisms with a complete or nearly complete reference genome. One tool is called the TopHat suite which has a bowtie aligner
* Cufflinks: probabilistic model that uses assembled reads and outputs complete transcripts.
* Stringtie: alternative to Cufflinks that performs transcript assembly and expression estimates in one step.

Galaxy can also run ChIP-seq and RNA-seq pipelines which is very similar to the variant calling pipeline. The Ch-IP pipeline is mostly the same except you run a MAC2 callpeak on the aligned BAM file instead of running Freebayes. This results in a MACS2 callpeak graph that can be visualized and compared with reference genes in the genome browser of your choice (IGV, UCSC, etc).

*Figure below shows a mouse ChIP-seq reads in the FASTQ format ran through the ChIP pipeline in Galaxy. This produces a MACS2 Bedgraph (on the top track) that can be compared with the mm9 reference genome ChIP-seq genes (on the bottom track)*

A picture containing chart

Description automatically generated

The RNA-seq pipeline is a little more complicated but shares the same steps as the previous pipelines except the reads are aligned and trimmed (ex. Trimmomatic) you run: HISAT2🡪Stringtie🡪Gffcompare🡪feature counts🡪bam coverage

*Full Galaxy workflow is in the diagram below which was used to run the analysis of a mouse (mm10) genome with the G1E and Megakaryocyte cell lines. There are 4 of each cell line with forward and reverse reads giving a total of 8 input reads:*

Graphical user interface, table

Description automatically generated

*Loading the 8 BIGWIG (BAM) files generated from the bamcoverage steps of the pipeline into IGV gives the peaks shown in the top tracks. Now we can compare these peaks with the mm10 genome (Refseq genes) and the Refseq reference gens (gtf files on the bottom tracks), to find similar gene expression values between the two cell lines. This gives us a good idea of the upregulation/downregulation of these cell lines. Bottom chart shows the output of the feature counts command comparing the two cell lines in a phylogenetic distance plot*

Graphical user interface

Description automatically generated

Chart, box and whisker chart

Description automatically generated

# SNP Research Paper

An Investigation of 4 SNPs in the APOE Gene in a Subset of Mexican-American Individuals.

Bioinformatics: Tools for Genome Analysis

Final Paper Group 2

Johns Hopkins University

Christopher Gerth, Max Cheng, Sarah Weisberg

**Abstract**

Parkinson’s Disease is a deadly disease of the brain and nervous system, and through statistical analysis, has been associated with various SNPS. In this investigation, four SNPs found in the promoter and genome region of the APOE protein are examined: rs449647, rs405509 (both found in the promoter region of the APOE gene) and rs7412, rs429358 (both found in exon 4 of the APOE gene). The population for study consisted of three randomly chosen male individuals of Mexican-American ethnicity, geographically located in California, U.S.A.. Male Mexican-American ethnicity in California, has been shown through other studies, to be at a higher risk of Parkinson’s disease. Our hypothesis was that the SNPs we were looking for would be in the genotype of one or more of these individuals. Our Results show that even though male, Californian, Mexican-American ethnicity is associated with a higher prevalence of Parkinson’s disease, none of these four SNPs were detected in our three individuals.

**Introduction**

APOE mainly functions in lipid transport between organs via the plasma and interstitial fluids playing a critical role in plasma and tissues lipid homeostasis. APOE is also a core component in the plasma lipoproteins and is involved in their production, conversion and clearance [18]. APOE’s lipid transport role in the central nervous system depends on correct structure and concentration if it is to effectively interact with these lipoproteins.

APOE’s protein “portion” is mostly transcribed by astrocytes and microglia and is lipidated by the ABCA transporter to form lipoprotein particles [11]. This lipoprotein is a vital part of the (Amyloid beta) Aβ metabolism where APOE is able to efficiently package soluble Aβ into a lipid-protein complex through a variety of cell surface receptors including LRP1, LDLR, and HSPG. This process allows for the transport of cholesterol from the astrocytes to the neuron and the breakdown of the amyloid compounds via lysosomal degradation. Failure to clear Aβ results in the buildup of Aβ oligomers and amyloid plaques causing neurofibrillary tangles or cerebral amyloid angiopathy (CAA) in nearby blood vessels. These conditions are indicative of neurodegenerative disorders such as Alzheimer's Disease (AD) and Parkinson’s Disease (PD). Figure 1 summarizes these processes.

The APOE is also involved in lipoprotein transport and metabolism as such it's no surprise that this gene's transcription gets activated by lipoprotein related receptors. Some of those receptors include liver X receptor, peroxisome proliferator-activated receptor γ and nuclear receptors. The ligands that bind the liver X receptor include oxysterols which are the oxygenated derivatives of cholesterol [2]. This receptor can also be found in other genes and proteins such as ABC, CETP, FAS, CYP7A1, LPL and ChREBP [2]. This receptor plays a key role in cholesterol metabolism. Peroxisome proliferator-activated receptor γ is a nuclear receptor protein that acts as a transcription factor [6]. This receptor also plays a key role in carbohydrate, lipid and protein metabolism.

**Diagram

Description automatically generated**

**Fig 1. A Diagram showing Alipoprotein E and amyloid-β metabolism in the brain. [7]**

Apolipoprotein (APOE) is a 299 AA long protein that contains multiple amphipathic α-helices bundled in groups of 4 at the non-polar N-terminus [14]. The C-terminus also contains α-helices along with a low-density lipoprotein receptor (LDLR)-binding site. The transcribed APOE protein can be divided by its C and N terminal domains (fig 2) [20]. The C terminal mainly binds lipids while the N terminal mainly contains receptors for its regulation. The main classes of receptors are low-density lipoprotein receptor (LDLR), very-low-density lipoprotein receptor (VLDLR) and apoE receptor 2 (apoER2) [20].

Genetically speaking, the SNPs of APOE pose a big risk factor for the development of AD and PD. For example, APOE can form 4 different isoforms by a combination of allele types from the SNPs rs429358 and rs7412 (found in exon 3 of the coding region and which are designated by episolons, see Table 1). It has been found that:” APOE e2 …reduces the risk of Alzheimer's; APOE e4…increases the risk of Alzheimer's and is associated with getting the disease at an earlier age. APOE e3…the most common [allele]... doesn't seem to affect the risk of Alzheimer's.” [24]. These isoform types also affect the predisposition to PD in the same way. Further, in a study done with individuals aged 50-59 years old, there was a 40.7 percent chance of developing AD in carriers (epsilon 4) compared to 8.2 percent in non-carriers [10]. In addition, APOE4 also has been found to contribute to multiple brain homeostatic pathways including synaptic integrity and plasticity, glucose metabolism pathways, and cerebrovascular pathways [21]. Further, in a previous study, it was found that the SNPs rs449647-A, rs405509-G were overrepresented in female PD patients from Eastern India. [25]

**Timeline

Description automatically generated with low confidence**

**Fig 2. Demonstrates the C and N terminal of the transcribed APOE gene [20].**

The APOE gene is located on chr19:44905796-44909393 (hg 38) and has a Coding Region at

Chr19:44,906,625-44,909,250 with a Coding Exon Count of 3. A promoter region is found at

Chr19:44905553-44905892. [17] (see Fig. 3).

**Chart, box and whisker chart

Description automatically generated**

Fig 3. The APOE gene, its associated control regions, SNPS and the exon/intron structure. The promoter region is shown in red, rs405509 can be seen; rs429358 and rs7412 can be seen in the exon region of APOE and lead to the various isoforms. The position of rs449647 can also be seen, this SNP is also associated with an increased risk of PD. [17]

|  |  |  |
| --- | --- | --- |
| **rs429358** | **rs7412** | **Epsilon designation** |
| **C** | **T** | **ε1** |
| **T** | **T** | **ε2** |
| **T** | **C** | **ε3** |
| **C** | **C** | **ε4** |

**Table 1. The different alleles of APOE4, specified as epsilons, [15].**

A particular SNP of interest and based on our initial SNP selection, is SNP rs405509 found in the promoter region of the APOE gene. The SNP rs405509 is considered a non-Mendelian factor along with several other polymorphisms related to PD. The rs405509 SNP causes a change from G-T at position -219. People who have the -219T allele have been found to be more susceptible to myocardial infarction. Moreover, people with this SNP have been found to have lower APOE plasma concentration, associated with an increased PD risk in both males and female subjects. Another study found that the rs405509 SNP is associated with increasing numbers of diseased vessels (see Fig. 3). Given the above information about other SNPs found in the APOE gene, this study seeks to probe the expression of not only the main SNP found in the promoter region, but four SNPs in a population of high susceptibility to PD. These SNPs are: rs449647, rs405509, rs7412 and rs429358.

**Diagram, schematic

Description automatically generated**

**Fig 4. Rs405509 T-allele reduces APOE expression when the T allele is present. Graphs A and B show promoter activity with T to G and G to T substitutions in rs405509 respectively and the resulting promoter activity. Graphs C-J show the expression of APOE of different allelic combinations in different regions of the brain [2].**

Rs405509 is part of the promoter region of APOE gene which plays a major role in lipid transport and management especially in the brain. Neurodegenerative diseases such as Parkinson’s and Alzheimers can both be attributed to the APOE gene and causes an inefficiency in packaging soluble Aβ into a lipid-protein complex. This leads to the build up of amyloid plaques which are characteristics for both risk and progression of the aforementioned neurodegenerative disorders. Studies in Alzheimer's patients have found that those with the T- allele tend to have a younger onset and more severe cortical atrophy than those with G-allele [2] for this specific substitution. A reporter gene assay showed that there was a significant difference in APOE expression with the SNP t allele causing reduced activity.

The APOE gene contains many SNPs some of which have been found to be restricted to certain populations. Based on a Medicare study from 2010, it was found that Americans of Mexican heritage residing in and around Los Angeles, CA. area had a higher disposition to PD [19]. The SNP which we are studying (rs405509) has been found to be mostly associated with the elderly population [22], however there are differences between Mexican groups [23]. The closest approximation that could be found is the map represented below (Fig. 5).

Diagram, map

Description automatically generated

**Fig. 5. Demonstrates the variability of the rs405509 SNP among the Mexican population. (I) Durango. (2) Mexico City. (3) Guanajuato, (4) Jalisco. (5) Michoacdn. (6) Morclos, (7) Nayarit. (8) Oaxaca, (9) Yucatan, (10) Zacatecas [23].**

**Table

Description automatically generated**

**Table 2. APOE Genotype Frequencies in Western and Northwestern Mexican population [23].**

**Hypothesis**

Based on the above data, we believe that individuals of Mexican-American dissent, who are prone to a higher incidence of PD should have an associated increase in the alleles associated with PD. Therefore, we are looking at the following SNPs to detect this: rs449647, rs405509, rs7412, rs429358.

|  |  |  |
| --- | --- | --- |
| SNP and change | Location (hg 38) | Significance |
| rs449647  G->A | chr19:44905307 | SNP in the enhancer region. |
| rs405509  T->G | chr19:44905579 | our main SNP in the promoter region. |
| rs7412  *(see table 1)* | chr19: 44908822 | One of the SNPs located in exon 4 involved in producing the APOE4 protein |
| rs429358  (*see table 1*) | chr19:44908684 | The second SNP located in exon 4 involved in producing the APOE4 protein |

**Table 3: Description of the 3 SNP we were looking for in our 4 individuals and the associated alleles**

**Methods**

We sampled 3 male individuals of Mexican American dissent from 1000 genomes as clinical ethnicity studies have shown that Mexican-Americans have a higher susceptibility to this condition (data found above). Since we cannot determine whether these patients have Parkinson’s disease due to clinical/HIPAA restrictions on genome analysis, we believe that surveying an ethnicity group with the highest chance for PD was our best option. Thus, the incidence rate of the SNPs that we researched should be higher in this population.

Three randomly chosen whole genome paired sequenced files of males of Mexican-American ethnicity living in California were chosen for investigation from the 1000 genomes web site. The sample numbers and associated FASTQ files were:

1. NA19798

ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR793/SRR793155/SRR793155\_1.fastq.gz ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR793/SRR793155/SRR793155\_2.fastq.gz

2. NA19661

ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034567/SRR034567\_1.fastq.gz ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034567/SRR034567\_2.fastq.gz

3. NA19685

ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034810/SRR034810\_1.fastq.gz ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034810/SRR034810\_2.fastq.gz

The data were chosen from two sections of the 1000 genomes database: 30X GRCH 38 and Phase 3. 30XGRCh38 uses the hg 38 assembly and Phase 3 uses the hg 37 assembly as a reference genome. These two different databases were used because NA19798 could only be found in the 30XGRCh38 and not the higher quality Phase 3 database.

Once the samples were chosen they were imported into Galaxy. Each paired-end file was initially examined using the FASTQC tool and was trimmed according to an average base score of 30 using a sliding window of 4 via the trimmomatic tool. Each sample was then aligned to the appropriate reference genome, hg38 or hg37 using the BowTie program with the paired-end reads option chosen. SNP calling was done by using the FreeBayes tool producing a VCF file. VCF filtering was done to a read depth of 10 or greater, but the data was not filtered for heterozygosity due to the nature of the APOE4 genotype.

To determine if our particular SNPs were present two bed files were imported from the UCSC Genome Table Browser, the hg38 using theAll SNPs 147 database and hg37 using the All SNPs 147 database. The SNP coordinates and names were compared against the two bed files to determine if the SNPs of interest were present.

**Graphical user interface, application, table, Excel

Description automatically generated**

**Fig. 6 Galaxy Workflow diagram showing the complete SNP pipeline process including the input files and the VCF output file**

**Results**

**Table

Description automatically generated**

**Fig 7. VCF annotation of Mexican American Male from NA19798 and FASTQ files ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR793/SRR793155/SRR793155\_1.fastq.gz ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR793/SRR793155/SRR793155\_2.fastq.gz Paired-end files used on hg37 genome from UCSC Table browser in the region 44,900,000 - 45,000,000 (where the APOE gene and our SNP are).**

**Graphical user interface

Description automatically generated with low confidence**

**Fig 8. VCF annotation of Mexican American Male from NA19661 and FASTQ files ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034567/SRR034567\_1.fastq.gz ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034567/SRR034567\_2.fastq.gz Paired-end files used on hg37 genome from UCSC Table browser in the region 44,900,000 - 45,000,000 (where the APOE gene and our SNP are).**

**Graphical user interface, application, table

Description automatically generated**

**Fig 9. VCF annotation of Mexican American Male from NA19685 and FASTQ files ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034810/SRR034810\_1.fastq.gz ftp://ftp.sra.ebi.ac.uk/vol1/fastq/SRR034/SRR034810/SRR034810\_2.fastq.gz Paired-end files used on hg37 genome from UCSC Table browser in the region 44,900,000 - 45,000,000 (where the APOE gene and our SNP are).**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Sample IDs | rs449647  (promoter) | rs405509  (promoter) | rs7412 (APOE4) | rs429358 (APOE4) |
| NA19798 | No | No | No | No |
| NA19661 | No | No | No | No |
| NA19685 | No | No | No | No |

**Table 4: Table of samples and SNPs located.**

All the genomes we searched did not have any of the SNPs we looked for, nor did they have our SNP, rs405509, which was the main subject of our paper [8]. This is not surprising since these allele frequencies are very small.

**Discussion**

Out of the three Mexican-American individuals that we studied through the SNP pipeline, none of them had any of the SNPs we were looking for. This result was expected as finding 4 specific SNPs in 3 random individuals (who may or may not have PD), was astronomically low even in an ethnicity that is more prone to PD. In fact, if there was an individual with one of the SNPs we studied, it would either be a false positive and we would still not be able to make any meaningful conclusions. Thus, our hypothesis that Mexican-Americans are more prone to have these SNPs due to their higher incidence of PD still remains unclear. While we did not find these 4 SNPs in these individuals, we cannot prove or disprove our hypothesis due to both sample size and access to genomic data from actual patients who have PD. If we actually had a reasonably sized genomic sample of PD patients, we may be able to find a correlation that gives us a better idea of the prevalence of the SNPs in the PD patients. If anything, this study was meant to present a technique that allows for the study of specific SNPs in relation to their respective clinical conditions. The SNP pipeline that we used can be applied to any type of medical condition and research the SNP prevalence in accordance. Further analysis of these SNPs in Parkinson’s patients is required.

**Conclusion**

We choose to investigate four SNPs located in the promoter of the APOE protein which were found to be highly correlated with Parkinson’s Disease. We hypothesized that our SNPs of interest would be found in a larger percentage in the Mexican-American population as they are at a higher risk for PD. However we did not detect any of the SNPs in any of the individuals we tested. Admittedly our methods contained some limitations; the 1000 genomes database that was used in this experiment was limited in the information it provided; while ethnicity, gender and geographical location were provided, age and phenotype of the individual were not provided. Further, the sample sizes for a particular geographical location and ethnicity were not particularly large, in our case only six samples were available for testing of which we chose three. Obviously a larger database with phenotype data would be more useful in examining this type of statistical association for a Californian, Mexican-American population. Other databases such as Data collection, Public genetic data by Personal Genome Project and TCGA's cancer sequencing data portal provide this type of data and a program could be written to scan them for the alleles of interest and the associated phenotypes, an approach we had considered, but several of these databases need the type of permission usually associated with study conducted under the auspices of a grant: P.I., grant number, associated investigating institute, etc, which we did not have access to.

**References**

**1.** 1000 Genomes Project Consortium, Auton, A., Brooks, L. D., Durbin, R. M., Garrison, E. P., Kang, H. M., Korbel, J. O., Marchini, J. L., McCarthy, S., McVean, G. A., & Abecasis, G. R. (2015). A global reference for human genetic variation. Nature, 526(7571), 68–74. https://doi.org/10.1038/nature15393

**2.** Choi KY, Lee JJ, Gunasekaran TI, Kang S, Lee W, Jeong J, Lim HJ, Zhang X, Zhu C, Won S-Y, Choi YY, Seo EH, Lee SC, Gim J, Chung JY, Chong A, Byun MS, Seo S, Ko P-W, Han J-W, McLean C, Farrell J, Lunetta KL, Miyashita A, Hara N, Won S, Choi S-M, Ha J-M, Jeong JH, Kuwano R, Song MK, An SSA, Lee YM, Park KW, Lee H-W, Choi SH, Rhee S, Song WK, Lee JS, Mayeux R, Haines JL, Pericak-Vance MA, Choo IH, Nho K, Kim K-W, Lee DY, Kim S, Kim BC, Kim H, Jun GR, Schellenberg GD, Ikeuchi T, Farrer LA, Lee KH, Neuroimaging Initiative AD. APOE Promoter Polymorphism-219T/G is an Effect Modifier of the Influence of APOE ε4 on Alzheimer’s Disease Risk in a Multiracial Sample. Journal of Clinical Medicine. 2019; 8(8):1236. https://doi.org/10.3390/jcm8081236

**3.** Enis Afgan, Dannon Baker, Bérénice Batut, Marius van den Beek, Dave Bouvier, Martin Čech, John Chilton, Dave Clements, Nate Coraor, Björn Grüning, Aysam Guerler, Jennifer Hillman-Jackson, Vahid Jalili, Helena Rasche, Nicola Soranzo, Jeremy Goecks, James Taylor, Anton Nekrutenko, and Daniel Blankenberg. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update, Nucleic Acids Research, Volume 46, Issue W1, 2 July 2018, Pages W537–W544, doi:10.1093/nar/gky379

**4.** ENSEMBL. Summary - homo\_sapiens - ensembl genome browser 104. (n.d.). Retrieved October 4, 2021, from http://uswest.ensembl.org/Homo\_sapiens/Transcript/Summary?db=core%3Bg.

**5.** Galaxy Enis Afgan, Dannon Baker, Bérénice Batut, Marius van den Beek, Dave Bouvier, Martin Čech, John Chilton, Dave Clements, Nate Coraor, Björn Grüning, Aysam Guerler, Jennifer Hillman-Jackson, Vahid Jalili, Helena Rasche, Nicola Soranzo, Jeremy Goecks, James Taylor, Anton Nekrutenko, and Daniel Blankenberg. The Galaxy platform for accessible, reproducible and collaborative biomedical analyses: 2018 update, Nucleic Acids Research, Volume 46, Issue W1, 2 July 2018, Pages W537–W544, doi:10.1093/nar/gky379

**6.** https://en.wikipedia.org/wiki/Apolipoprotein\_E

**7.** https://www.jillcarnahan.com/2017/03/26/will-get-alzheimers-disease-apoe/ retrieved 10/03/2021

**8.** Huang, M., Wang, Y., Wang, L., Chen, B., Wang, X., & Hu, Y. (2020). APOE rs405509 polymorphism and Parkinson's disease risk in the Chinese population. Neuroscience letters, 736, 135256. https://doi.org/10.1016/j.neulet.2020.135256

**9.** James T. Robinson, Helga Thorvaldsdóttir, Douglass Turner, Jill P. Mesirov. igv.js: an embeddable JavaScript implementation of the Integrative Genomics Viewer (IGV). bioRxiv 2020.05.03075499.

**10.** Kok E, et al. Apolipoprotein E–dependent accumulation of Alzheimer disease–related lesions begins in middle age. Ann Neurol. 2009;65:650–657

**11.** Liu, C. C., Liu, C. C., Kanekiyo, T., Xu, H., & Bu, G. (2013). Apolipoprotein E and Alzheimer disease: risk, mechanisms and therapy. Nature reviews. Neurology, 9(2), 106–118. https://doi.org/10.1038/nrneurol.2012.263

**12.** National Center for Biotechnology Information; [1988] – [cited 2021 Oct 02]. Available from: https://www.ncbi.nlm.nih.gov/

**13.** Papaioannou, Ioannis & Simons, J & Owen, James. (2012). “Targeted In Situ Gene Correction of Dysfunctional APOE Alleles to Produce Atheroprotective Plasma ApoE3 Protein. Cardiology research and practice. 2012. 148796. 10.1155/2012/148796”.

**14.** Phillips MC (September 2014). "Apolipoprotein E isoforms and lipoprotein metabolism". IUBMB Life. 66 (9): 616–23. doi:10.1002/iub.1314. PMID 25328986. S2CID 6159310.

**15.** SNPedia. https://www.snpedia.com/index.php/APOE#:~:text=Variations%20in%20ApoE%2 0are%20also%20associated%20with%20altered,these%20genes%20are%20call ed%20ApoE2%2C%20ApoE3%2 C%20and%20ApoE4. Retrieved 11/1/2021.

**16.** Solovieff, N., Hartley, S. W., Baldwin, C. T., Perls, T. T., Steinberg, M. H., & Sebastiani, P. (2010). Clustering by genetic ancestry using genome-wide SNP data. BMC genetics, 11, 108. https://doi.org/10.1186/1471-2156-11-108

**17.** UCSC Table Browser: Karolchik D, Hinrichs AS, Furey TS, Roskin KM, Sugnet CW, Haussler D, Kent WJ. The UCSC Table Browser data retrieval tool. Nucleic Acids Res. 2004 Jan 1;32(Database issue):D493-6.

**18.** UniProt ConsortiumEuropean Bioinformatics InstituteProtein Information ResourceSIB Swiss Institute of Bioinformatics. (2021, June 2). Apolipoprotein E. UniProt

**19.** Wright Willis A, Evanoff BA, Lian M, Criswell SR, Racette BA. Geographic and ethnic variation in Parkinson disease: a population-based study of US Medicare beneficiaries. Neuroepidemiology. 2010;34(3):143-151. doi:10.1159/000275491

**20.** Zhao, N., Liu, C. C., Qiao, W., & Bu, G. (2018). Apolipoprotein E, Receptors, and Modulation of Alzheimer's Disease. Biological psychiatry, 83(4), 347–357. https://doi.org/10.1016/j.biopsych.2017.03.003

**21.** Yamazaki, Y., Zhao, N., Caulfield, T. R., Liu, C. C., & Bu, G. (2019). Apolipoprotein E and Alzheimer disease: pathobiology and targeting strategies. Nature reviews. Neurology, 15(9), 501–518.

https://doi.org/10.1038/s41582-019-0228-7

**22.** Abondio, P.; Sazzini, M.; Garagnani, P.; Boattini, A.; Monti, D.; Franceschi, C.; Luiselli, D.; Giuliani, C. The Genetic Variability of APOE in Different Human Populations and Its Implications for Longevity. Genes 2019, 10, 222. https://doi.org/10.3390/genes10030222

**23.** Aceves, D., Ruiz, B., Nuño, P., Roman, S., Zepeda, E., & Panduro, A. (2006). Heterogeneity of apolipoprotein E polymorphism in different Mexican populations. *Human biology*, *78*(1), 65–75. https://doi.org/10.1353/hub.2006.002[1](https://doi.org/10.1353/hub.2006.0021)

**24.** Alzhemiers Genes are You at Risk? https://www.mayoclinic.org/diseases-conditions/alzheimers-disease/in-depth/alzheimers-genes/art-20046552. Retrieved 12/20/2021

**25.** Pal, P., Sadhukhan, T., Chakraborty, S. *et al.* Role of Apolipoprotein E, Cathepsin D, and Brain-Derived Neurotrophic Factor in Parkinson’s Disease: A Study from Eastern India. *Neuromol Med* **21,** 287–294 (2019). https://doi.org/10.1007/s12017-019-08548-4.

# Computational Programs and Databases

## SQL

Graphical user interface, text, application, email

Description automatically generated

With the sample database of genes, I can use SQL in the terminal to search for:

Just the thrB gene:

SELECT Gene FROM genes WHERE Product=Homoserine Kinase

All three genes which starts with 'thr'

SELECT Gene FROM genes WHERE Gene=’thr%’

All genes whose start site falls between base 2500 and 5000.

SELECT Gene FROM genes WHERE Start BETWEEN 2500 AND 5000

I can create the entire gene table above from scratch with the code:

CREATE TABLE genes (Protein\_ID VARCHAR(12) NOT NULL,Product VARCHAR(255) NOT NULL, Gene VARCHAR(10) NOT NULL, Start Int(8) NOT NULL, Stop Int(8) NOT NULL);

I can also add data to the table by using the insert function to add the second row of the table

INSERT INTO genes (ProteinID, Product, Gene, Start, Stop)

VALUES (ACB01208.1, Homoserine kinase, thrB, 2801, 3733)

## Python

The scripts I’ve written are all also on Github but I want to showcase some of my scripts here so that you can get a better understanding of the types of programs I can write, and my progression as a programmer.

### Circle “Program”

A picture containing graphical user interface

Description automatically generated

*First program (can you even call it that if it is just a math equation) I’ve ever written!*

Everyone must start somewhere I guess, and this was mine. Input your radius and I can get it to output the circumference. But you will have to manually go into my code and change the values to get an output, lame. But with no knowledge of definitions/functions and inputs, there was a lot to improve on.

### DNA Length and Start Codon Program

Graphical user interface, text

Description automatically generated

Output: The length of your sequence is= 28 The first codon is= ATG

Learned how to use the len function as well has how to print elements of a string. This allowed me to write a program that can count how long the input DNA sequences are as well as the first codon.

### Motif Counter

Text

Description automatically generated

Output: There are 6 instances of the motif 'T'

First real function I’ve written that takes a motif input and a DNA string and counts how many instances of that motif are in the string. Still no input on the command line yet but I can write functions now

### FASTA Sequence Counter

A screenshot of a computer

Description automatically generated with medium confidence

Output: There were 4126 sequences within the file.

My programs are getting more practical now and this one allows you to enter the path to a FASTA file as an argument and return the number of sequence entries within it. Very basic, but very useful program if you have many FASTA files to analyze. This code can also be modified for other filetypes as well.

### Parsing Program

Text

Description automatically generated

Output: The brown, quick fox jumped over some other non-descript animal

My first introduction into the power of Regex and parsing. This program allows you to input a sentence with two adjectives separated by a comma and reverses them. Parsing with regex allows me to utilize pattern recognition in text ranging from simple sentences to complex genomic data files.

### Genome Analysis Program

A picture containing text

Description automatically generated

Output: L: 139002 (10.7%)

A: 123885 (9.6%)

G: 95475 (7.4%)

V: 91683 (7.1%)

I: 77836 (6.0%)

A more complex program used to generate an amino acid usage report from one FASTA files with multiple polypeptide sequences. Through this project I learned how to use dictionaries to store and manipulate data.

### Angle Converter

Text

Description automatically generated

Output: Enter starting units(D or R):D

Enter starting angle: 120

120.0 degrees is 0.7 radians

Building upon the simple mathematic function between degrees and radians, I wrote a converter that allows users to input their desired units and angle and the program will spit out the corresponding angle. I also began annotating my code to elucidate what I am doing.

### DNA Reverse Complement Program

Text

Description automatically generated

Output: Enter nucleotide sequence:ATGCAGTACGATTTC

GAAATCGTACTGCAT

Enter nucleotide sequence:AATGAGGADTCWD

invalid character found

My next project is a reverse complementary base program that gives you the reverse complement of a user inputted string of DNA. Here I learned that you could add user-friendly, fidelity checks to your code as well as utilizing multiple dictionaries and logic statements.

### GFF Feature Exporter

Text

Description automatically generated

Text

Description automatically generated

The input and output of the code is a comment at the top of the program

My programing magnum opus for my first semester of coding is a GFF3 feature exporter. This is the longest code I’ve ever written that ties everything I’ve learned from the previous programs. From the command line, users can enter a specific gene from the GFF3 file format for any organism and the parser will match the gene with its corresponding nucleotide sequence via the FASTA sequences located at the bottom of the GFF3 file.

### FASTQ Analysis Tool

After another semester, my new magnus opus is a website that I designed which runs a tool that parses user inputted FASTQ sequencing files for values such as the number of entries, read length, total bases, GC percentages, and quality scores. This we basically a full stack project that put together many of the topics I learned throughout the semester such as HTML, CSS, mySQL, and Python. Together, the website sends a server request with the information from the form to the server and is parsed by a python script. This resulting data is stored in a simple SQL database and is sent back to the webpage in the “results” section and the past runs can be found in the “previous runs” section where users can see the time and information of their previous runs. As always, the code can be found on my GitHub but is listed below for convenience.

Webpage:

A picture containing graphical user interface

Description automatically generated

HTML:

<!DOCTYPE html>

<html lang="en">

<head>

    <script type="text/javascript" src="https://ajax.googleapis.com/ajax/libs/jquery/1.6.2/jquery.min.js"></script>

    <script src="https://code.jquery.com/jquery-3.6.0.js"></script>

    <script src="https://code.jquery.com/ui/1.13.1/jquery-ui.js"></script>

    <script type="text/javascript" src="js/FASTQSTAT.JS"></script>

<meta charset="utf-8">

<meta name="viewport" content="width=device-width, initial-scale=1">

<style>

\* {

  box-sizing: border-box;

}

body {

  font-family: Arial, Helvetica, sans-serif;

}

.header {

  padding: 5px;

  text-align: left;

  font-size: 30px;

}

/\* subHeader \*/

.subheader {

  background-color: #7FFFD4;

  padding: 5px;

  text-align: left;

  font-size: 15px;

}

/\* Flexbox Container \*/

.row {

  display: -webkit-flex;

  display: flex;

}

.searchbox {

    padding: 5px;

    width: 200px;

    height: 400px

    }

.fileform {

    padding: 5px;

    width: 100px;

    height: 100px

}

/\* Flexbox columns\*/

.column {

  -webkit-flex: 1;

  -ms-flex: 1;

  flex: 1;

  padding: 10px;

  height: 400px;

}

/\* Footer Style \*/

.footer {

  background-color: #f1f1f1;

  padding: 10px;

  text-align: center;

}

/\* Responsive layout  \*/

@media (max-width: 600px) {

  .row {

    -webkit-flex-direction: column;

    flex-direction: column;

  }

}

</style>

</head>

<body>

<div class="header">

<h2>FASTQuant</h2>

<p>A fast and easy quantitative analysis tool for FASTQ files  </p>

<div class="subheader">

    <h2>FASTQ Entry</h2>

    <p>Click on the "Choose File" button to upload a file or enter it below:

    </p>

    <p>Note: For pastes with 10 or more entries please use the upload file method

    </p>

<div>

  <div class="searchbox">

    <textarea id="fastqbox" name="fastqbox" rows="22" cols="150" form="txtform"></textarea>

      <form id="txtform">

    <input type="submit" value="Submit" id="SubmitText">

  </form>

  </div>

<div class="fileform">

<form enctype="multipart/form-data" method="post">

<input type="file" id="myFile" name="myFile" value="">

<input type="submit" id="filesub">

</form>

</div>

</div>

<div class="row">

  <div class="column" style="background-color:#FF6347;">

  <strong>Quick Analysis Results</strong>

  <p>Number of entries:{{entry\_count}} </p>

  <p>Average read length:{{read\_length}} </p>

  <p>Total number of bases:{{total\_bases}}</p>

  <p>GC percentage:{{gc\_percent}}</p>

  <p>Average quality score:{{avg\_qual\_score}} </p>

</div>

  <div class="column" style="background-color:#2E8B57;">

  <strong>Previous Runs<strong>

  <table style="width:100%">

  <tr>

    <th>Date</th>

    <th>Number of entries</th>

    <th>Average read length</th>

    <th>Total number of bases</th>

    <th>Average GC percentage</th>

    <th>Average quality score</th>

  </tr>

 {% for entry in db %}

<tr>

   <td>{{entry.date}}</td>

   <td>{{entry.num\_entries}}</td>

   <td>{{entry.read\_length}}</td>

   <td>{{entry.base\_count}}</td>

   <td>{{entry.gc\_percentage}}</td>

   <td>{{entry.average\_qc}}</td>

</tr>

{% endfor %}

</table>

  </div>

</div>

<div class="footer">

  <p>For the full documentation visit my  <a href="https://github.com/mbcheng88/ACOMP\_Final\_Project">github!</a></p>

  <p>Special thanks to Prof. Orvis for helping this project along!</p>

</div>

</body>

</html>

Python Script:

#!/usr/local/bin/python3

import jinja2

import re

import argparse

import codecs

import gzip

import numpy

import os

import sys

import re

import mysql.connector

import cgi

templateLoader = jinja2.FileSystemLoader( searchpath="./templates" )

env = jinja2.Environment(loader=templateLoader)

template = env.get\_template('FASTQSTAT.html')

#database input/output connection

def dataload(num\_entries, read\_length, base\_count, gc\_percentage, average\_qc):

    conn = mysql.connector.connect(user='mcheng30', password='metagross88!',

                                   host='localhost', database='mcheng30')

    curs=conn.cursor()

    insval = "INSERT INTO FASTQTOOL (num\_entries, read\_length, base\_count, gc\_percentage, average\_qc) VALUES ('{0}','{1}', '{2}', '{3}', '{4}');".format(num\_entries, read\_length, base\_count, gc\_percentage, average\_qc)

    qry="SELECT\* FROM FASTQTOOL ORDER BY date DESC LIMIT 8"

    if num\_entries!="-":

        curs.execute(insval)

    curs.execute(qry)

    results=[]

    for row in curs:

        value={}

        value['date']=row[0]

        value['num\_entries']=row[1]

        value['read\_length']=row[2]

        value['base\_count']=row[3]

        value['gc\_percentage']=row[4]

        value['average\_qc']=row[5]

        results.append(value)

    conn.commit()

    conn.close()

    return results

#retrieves form data from the user inputted HTML form

def retrform():

    form = cgi.FieldStorage()

    txtbox = form.getvalue('fastqbox')

    if txtbox==None:

        txtbox=""

    else:

        return txtbox

    if 'myFile' in form:

        fileitem = form['myFile']

        if fileitem.file:

            textLines = ""

            while True:

                line = fileitem.file.readline()

                if not line: break

                textLines += line.decode('UTF-8')

            return textLines

    return ""

#parser function for file

def parser(txtdata):

    entry\_count = 0

    read\_length = 0

    gc\_count = 0

    total\_bases = 0

    qual\_score = 0

    avg\_entry\_length=0

    qual\_symbol=['!','"', '#', "$", "%", "&", "'", "(", ")", "\*", "+", ",", "-", ".", "/", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", ":", ";", "<", "=", ">", "?", "@", "A", "B", "C", "D", "E", "F", "G", "H", "I", "J", "K", "L", "M", "N", "O", "P", "Q", "R", "S", "T", "U", "V", "W", "X", "Y", "Z", "[", "\\", "]", "^", "\_", "`", "a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p", "q", "r", "s", "t", "u", "v", "w", "x", "y", "z", "{", "|", "}", "~"]

    if len(txtdata)>0:

        fh=txtdata

    else:

        return ["-","-","-","-","-"]

    line\_number=0

    for line in fh.splitlines():

        line = line.rstrip()

        line\_number += 1

        if line\_number % 4 == 1:

            if line.startswith('@'):

                entry\_count += 1

        elif line\_number % 4 == 2:

            total\_bases += len(line)

            gc\_count += line.count('G')

            gc\_count += line.count('C')

        elif line\_number % 4 == 0:

            for char in line:

                qual\_count= qual\_symbol.index(char)+1

                qual\_score+=qual\_count

    avg\_qual\_score = (qual\_score/entry\_count)

    avg\_entry\_length = total\_bases/entry\_count

    entry\_count=entry\_count

    read\_length=avg\_entry\_length

    total\_bases=total\_bases

    gc\_percent=round((gc\_count / total\_bases)\* 100,2)

    avg\_qual\_score=round(avg\_qual\_score,2)

    return [entry\_count, read\_length, total\_bases, gc\_percent, avg\_qual\_score]

#html data print

def maketemplate(entry\_count,read\_length,total\_bases,gc\_percent,avg\_qual\_score,data):

    print("Content-Type: text/html\n\n")

    print(template.render(entry\_count=entry\_count,read\_length=read\_length,total\_bases=total\_bases,gc\_percent=gc\_percent, avg\_qual\_score=avg\_qual\_score, db=data))

formdata=retrform()

results = parser(formdata)

data=dataload(results[0], results[1], results[2], results[3], results[4])

maketemplate(results[0], results[1], results[2], results[3], results[4], data)

Admittedly there is much to improve upon in both the code and purpose. For starters, a website can be quite useless as these types of tools are ran in the command line and through packages. So I began picking up how to develop packages and workflow management systems. Some of the programs I used include:

## Docker/Biocontainers

## Cromwell and WDL

## Jupityr Notebooks

# Acknowledgements

First, I want to thank my Tools for Genome Analysis Professors Sijung Yun & Sajung Yun for the time and effort spent on making sure that us students are prepared for the bioinformatics field with both their coursework and professional guidance. This showcase would not be possible without them. I also want to thank Professor Jarrett Morrow for the guidance in learning how to use Unix, SQL, and write code in Python.

Next, I appreciate the support of my peers who have answered any questions no matter how simple. Group projects and discussions allowed me to expand my scope of knowledge and provide me with different perspectives.

Furthermore, I am grateful for the resources provided by Johns Hopkins University such as the bioinformatics server when we can practice Unix and SLQ and test programs. The counselor guidance and seminars have been very useful.

Lastly, I want to thank the reader for taking the time out of their day to read this portfolio up to this point. I hope that you now have a better understanding of the content I have been working on and knowledge of the skills that I have developed. If you have any questions or concerns about my resume or my portfolio, please feel free to contact me by phone (408-908-0395) or email (mcheng30@jh.edu) as I would be happy to answer them.