**Identification of Single-Nucleotide Variants from Mitochondrial DNA Sequencing Data of Single Cells**

Sean-Wyn Ng

Advisor: Dr. Ben Raphael

**Abstract**

*Single-cell nucleic DNA sequencing has exceedingly sparse coverage throughout the genome [4]. However, as there exist hundreds to thousands of mitochondria per cell [5], one of the goals of this project was to confirm the high coverage of mitochondrial sequencing DNA, using the results to identify single-nucleotide variants within real breast tissue datasets. By filtering genomic sequences from the dataset, this project was successful in obtaining a high number of reads in bulk and per cell within the mitochondrial sequences. Afterwards, single-point variants were successfully detected and analyzed per cell throughout the mitochondrial genome.*

**Introduction**

DNA sequencing has been imperative to the acceleration of biological and cancer research over the past half-century, as the ordering of nucleic acids within DNA molecules “ultimately contains the information for the hereditary and biochemical properties of terrestrial life” [1]. Traditionally, researchers have used bulk approaches to sequencing DNA, where mutations are analyzed over an entire cellular dataset without distinguishing between individual mutations.

**A screenshot of a cell phone

Description automatically generated**

**Figure 1: Bulk vs. Single-Cell Sequencing [4]**

However, the analysis of single-cell sequencing data has additional applications in medicine, as the output of bulk sequencing data “masks differences between individual disease-associated cells.” [2] For example, the development of cancer usually stems from somatic mutations occurring within one or more single cells, which begin to divide uncontrollably in the denigration of cell cycle checkpoint markers that trigger apoptosis, or cell death. As a result, daughter cells of the original mutated cells are generated, which could accumulate additional mutations. This effectively forms heterogeneous cancerous cell samples, which consist of subpopulations of cells that each contain different mutations in their respective DNA sequences. Therefore, through single-cell sequencing, researchers can further their “understanding of the composition, interactions, dynamics, and operating principles of tumor ecosystems.” [2] With this information, researchers can potentially trace the evolutionary lineage of tumorous cell subpopulations, and consequently, develop a better understanding of “the tumor microenvironment [and its components’] roles during cancer progression, metastasis, immune evasion, and the development of therapeutic resistance.” [2] Additionally, the ability to infer subpopulations of tumorous cells has significant implications in fields such as personalized medicine, as the currently exorbitant costs of drug development and ineffectiveness of treatments are often due to cellular heterogeneity that exists between cancer patients, where “treatment failure may result if a tumor contains many malignant subsets, of which only some respond to treatment [3].”

But due to existing biological and technological limitations, single-cell sequencing typically has an extremely low genome coverage rate (0.03x per cell) [4], where coverage is defined as the total length of reads divided by the length of the genome. Therefore, it is difficult to consistently identify mutations throughout the DNA sequence with single-cell sequencing.

A screenshot of a cell phone

Description automatically generated

**Figure 2: Sparse coverage of the genome per individual cell in sc-RNA seq. [4]**

However, human cells contain many mitochondria, which are present at rates in the low hundreds to thousands per cell [5]. These mitochondria which may contain anywhere from 1-15 copies of mitochondrial DNA (~16 kb) each [6]. It could be possible to circumvent low read coverage rates for single-cell sequencing data, as many more sequencing reads may be obtained from mitochondrial as opposed to nucleic DNA (of which there exist only two copies per cell). Therefore, the goal of this project was to use sequenced mitochondrial DNA to identify single point mutations within cancerous tumor samples.

**2. Background and Related Work**

**a. Whole Genome Amplification**

Whole genome amplification, which is “the use of biochemical methods to produce multiple copies of the entire genome” [7] can increase low initial coverage rates. However, the technological noise that accompanies whole genome amplification can introduce difficulties into the identification of single nucleotide variants within cells. According to Xianwan Ren of Peking University, “notable allelic dropouts (i.e., amplification and sequencing of only one allele of a particular gene in a diploid/multiploidy cell) and non-uniform genome coverage [2]” contribute to the current limitations of single-cell sequencing in detecting various types of mutations. For example, loss of coverage over certain positions can make it difficult to obtain a sufficient amount of data on lowly-expressed genes with significant functional impact. Additionally, allelic dropout from whole genome sequencing could result in false negatives for the detection of single nucleotide variants, whereas sequencing and amplification errors in certain areas of the genome would contribute towards a larger false positive rate for those positions [7].

It is worth noting that reductions in the detection of mutations during single-cell sequencing can also be caused by low tumor purity, which “is defined as the fraction of cells in the sample that are cancerous” [8]. The presence of normal cells in a tumor sample will cause the proportion of reads with somatic mutations to decline, so the number of mutations per position in the sample will decrease. Algorithms for detecting single point mutations “require[e] multiple reads with the same single-letter substitution to be aligned at the position” [8] in order to prevent sequencing errors from being classified as mutations. Therefore, with low tumor purity, less positions will be identified as containing mutations, even though a significant amount of mutations may be present at those positions.

**b.** **Mitochondrial DNA Sequencing**

WGA does not entirely address the problems of low coverage, due to its uneven amplification of certain areas of the genome across individual cells. Low tumor purity can also cause less mutations to reach the threshold for being identified as mutations. Consequently, some researchers have already turned to mitochondrial DNA for larger depth coverage amongst cells. For one, “mtDNA is not chromatinized and is therefore highly accessible” [4], so it is physically easier in some cases to sequence mitochondrial rather than nucleic DNA. Additionally, mitochondrial DNA has a ten-fold higher mutation rate compared to the nuclear genome [4], so more positions within the mitochondrial chromosome would be classified as containing mutations by various algorithms for somatic variant detection.

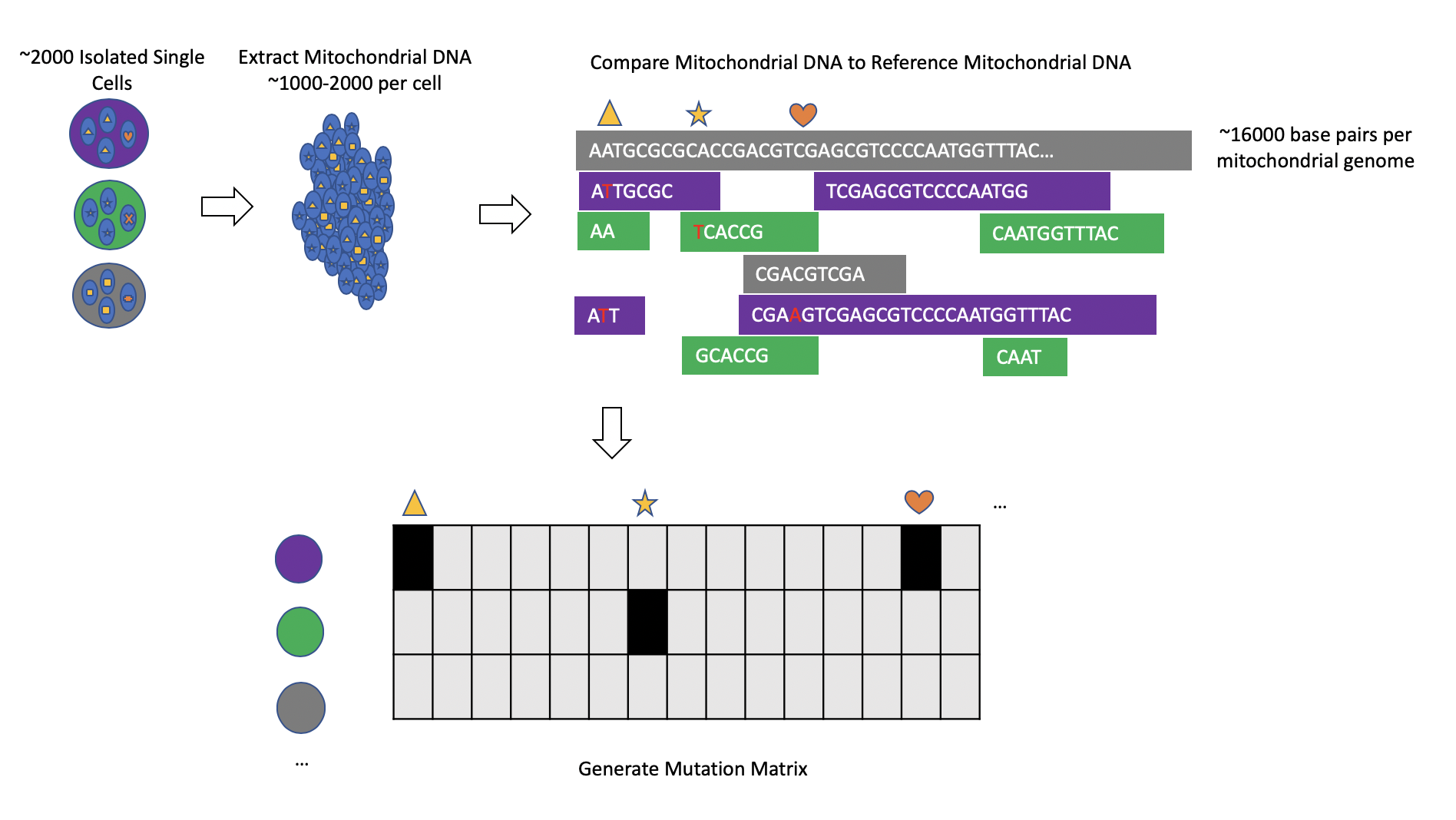
There have been attempts to analyze mitochondrial DNA in the context of single cells. Laurent Chatre and Miria Ricchetti of the Institut Pasteur used fluorescent single-cell imaging to reveal the heterogeneity of physical internal mitochondrial structures within single cells [9]. A study by Diane Lehmann of the University of Ulm examined the “clear correlation between the level of mtDNA deletion and extent of respiratory chain deficiency within a single cell,” inspired by evidence of mtDNA deletions in muscle fibre from single-molecule PCR [10].

Researchers at the Broad institute of MIT and Harvard recently used mtDNA mutations within single cells to try to derive clonal evolutionary relationships [10]. They performed a “proof-of-principle experiment, where [they] derived and propagated sub-clones of the hematopoietic TF1 cell line,” [11] essentially creating their own population of subsets containing unique cell sequences. The mitochondrial genome had a coverage of about 3,380-fold, and the researchers were able to detect high-confidence mitochondrial mutations, ultimately concluding that using somatic mtDNA mutations could potentially allow “clonal tracking at a 1000-fold greater scale than with nuclear genome sequencing.” [11]

**3. Approach**

As previous studies have demonstrated success in analyzing mitochondrial DNA through single-cell sequencing, this project aimed to show high coverage per mitochondrial genome within single cells, explicitly comparing these results with typical expected coverages of single cells across the nuclear genome. An additional goal was to use an actual breast tissue dataset to identify and analyze single-point mutations occurring within the cellular samples.

1. **Overview**



**Figure 3: Project Schematic**

I obtained datasets of normal and tumor breast tissue from the 10x genomics website. I used library packages such as SAMtools in order to extract the mitochondrial portions of their genomes. Using my data, I used mutation callers in order to identify mutations across all of the distinct individual cells in my dataset. I analyzed the mutations detected across my dataset and constructed a mutation matrix detailing the occurrence of mutations per position per cell.

1. **Mutation Calling**

I used the mutation caller VarScan2 in order to identify mutations within my sample, as the tool “allows a user to provide an estimate of tumor purity in order to calibrate the expected number of reads containing a somatic mutation at a single locus” [8]. The normal and tumor purities of my sample were known, at 0.9 and 0.82 respectively.

1. **Implementation**
   1. **Datasets**

I obtained a normal and tumor breast tissue dataset, as well as a human reference dataset, from 10x genomics, a website supplying information about single-cell sequencing DNA. The samples were in a preformatted BAM file, which is a compressed version of a SAMtools file. This would make it easier for me to format the data using SAMtools later on. Using *wget,* I downloaded the datasets into a personal folder on Princeton’s remote cycles server. On Princeton’s remote cycles server, I would often run *screen* sessions in order to perform very long, computationally intensive tasks while logged out of cycles.

Although the tumor sample itself would naturally contain differences in positions with regards to the reference DNA, the normal sample was necessary to specifically identify somatic mutations—germline mutations present in both the normal and tumor samples were not counted towards my final results.

* 1. **Data Formatting**

I prepared my starting data for use with the mutation callers on Princeton’s remote cycles server. First, I used the SAMtools *view* command in order to extract the mitochondrial portion of the chromosome from both the normal and tumor cell samples into new BAM files. Afterwards, I used the SAMtools *sort* command in order to properly align the normal and tumor samples with the ordering of the reference file. Then, I used the SAMtools *index* command in order to generate new indices for the generated BAM files.

* 1. **Coverage Calculation and Verification**

I installed a tool, MOSDepth, to calculate the depth of reads in my normal and tumor samples within the mitochondrial genome. MOSDepth runs faster than other common depth coverage calculation tools (such as BEDtools), and it automatically outputs and formats a list of coverage per chromosome as well. I transferred the results to my personal computer with FileZilla and used Jupyter Notebook to sum over this column to obtain the total read length of the mitochondrial genome sequences.

A screenshot of a cell phone

Description automatically generated

**Figure 4: Number of reads per position (normal sample)**

MOSDepth also output average coverage per chromosome over the entire genome. Then, to double check that the coverage was being calculated correctly, I also used the SAMtools *stats* command on my normal and tumor mitochondrial files. I obtained statistics for the average length of reads per cell, as well as total read counts under various conditions (reads mapped, reads unmapped, reads properly paired, etc.). For the normal sample, the number of properly paired reads multiplied by the average read length was 259 million, and the total read count calculated by MOSDepth was 222 million. The percent difference between MOSDepth and SAMtools depth calculations was about 15.3%, probably due to slightly differing parameters and thresholds for calculating coverage. To calculate the number of reads per individual cell, I used SAMtools *view* and *grep* to select all unique barcodes within the original normal and tumor mitochondrial BAM files. Afterwards, I used a standard UNIX *sort* command to filter out the top two hundred cells with the highest number of reads and place them into a new file, as I wanted to ensure minimal noise in my dataset.

* 1. **Mutation Calling and CHISEL**

I downloaded VarScan into a separate software folder within my personal cycles account folder using *wget*. Then, I ran VarScan on my normal, tumor, and reference samples with multiprocessing, setting an option to use 30 jobs. However, VarScan called 0 somatic mutations afterwards, as the default minimum variant allele frequency (the proportion of reads at a position containing a mutation) was set too high. My final run of VarScan set the minimum variant allele frequency to 0.005, as well as setting the normal cell purity rate to 0.9 and the tumor cell purity rate to 0.82. VarScan output a VCF file containing detected somatic mutations.

I ran the VCF file through CHISEL, a program developed by Dr. Simone Zaccaria in Dr. Benjamin Raphael’s lab group at Princeton. The program output a TSV file of barcodes per position in the mitochondrial genome, if the position contained at least one mutation. The depth of coverage (COV), number of mutations (MUTCOV), and specific letter mutation per position per cell was also listed.

A close up of a logo

Description automatically generated

**Figure 5: Output of CHISEL when used on a VCF file.**

* 1. **Data Cleaning and Processing**

I transferred the TSV file to my home computer using FileZilla and used Jupyter notebook to clean and process my data. As the TSV file was extremely large and very computationally intensive to work on, I opened the original TSV file once and selected only the rows that contained the barcodes of the top 200 cells. I wrote these rows to a new CSV file within my personal computer. Then, I parsed the data from the new CSV file into various python dictionaries whose key corresponded to a cell barcode, and value to an array with the length of the mitochondrial genome. The four dictionaries, and their four corresponding arrays kept track of MUTCOV, COV, BINARY\_MUTCOV, and per position respectively. A value within BINARY\_MUTCOV was set to 1 if MUTCOV was greater than or equal to one; otherwise it was set to 0. For example, to add to my COV dictionary in the first row of Figure , my program would have accessed the “Cell” parameter (AAACCTGA…) and indexed into the dictionary with COV[cell ID]. Then, the program would have added the COV value (1) into the corresponding array at index POS (65).

I used Python dictionaries because they are very fluid and easy to manipulate; adding or removing items from the dictionaries is not difficult. I could easily view COV and MUTCOV statistics over the entire length of the mitochondrial genome per single cell, which was mapped to a nunmpy array. I also chose this representation because it’s relatively straightforward to filter values from numpy arrays, and perform calculations and visualizations on them via Python libraries such as matplotlib, seaborn, and statistics.

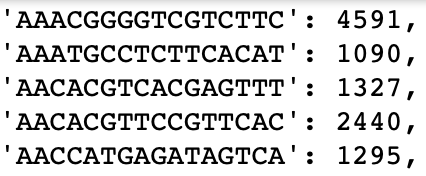
I also summed over the entire COV column directly row-by-row, and compared this value with the sum of all values across all COV arrays to ensure that my data parsing worked correctly. I ran the same test with the MUTCOV dictionary as well. I also checked that the average number mutations per position was consistent with the average number of mutations per cell. Additionally, when adding values to my dictionary, which kept track of variant allele frequency per position, I added “0” to a position in case the coverage at a position within one cell was 0.

To calculate statistics on COV, MUTCOV, etc. per mitochondrial position without regards to individual cells, I allocated a single numpy array with the length of the mitochondrial genome. Then, I iterated through the relevant dictionary and summed the arrays within the dictionary together, assigning the resulting contents to the previously allocated numpy array. I used this method with the MUTCOV dictionary to identify and sort the top fifty positions with the highest mutation coverage. I saved the sorted top fifty positions into a separate array for later use. Afterwards, I modified the dictionaries to have values of arrays with length fifty instead of 16,559, so that they could solely contain information concerning the top fifty positions. A close up of a piece of paper

Description automatically generated

**Figure 6: Contents of COV dictionary over top 50 positions**

I also kept arrays corresponding to each of my dictionaries. One array would store the sums of its corresponding dictionaries’ arrays for each cell, as illustrated below:



**Figure 7: An array contained sums over each cell from its corresponding dictionary. The right column is the array for COV from before positions filtering.**

* 1. **Visual and Statistical Data Analysis**

The purpose of the arrays such as those in Figure 7 was to use them with the statistics and seaborn packages, as they take in array arguments to model frequency distributions and calculate parameters like medians and averages of parameters per single cell, such as total MUTCOV, the total number of mutations at a position per cell.

I used the seaborn distplot function in order to generate my histograms. To generate clustermaps, I used the seaborn clustermap function. The function takes in an *m x n* array, so I wrote a function in order to generate a multi-dimensional array from my dictionaries (the BINARY\_MUTCOV dictionary in particular, for the mutation matrix). The function took in a dictionary argument, and initialized an empty array of length fifty. I sorted the dictionary alphabetically by cell name, then iterated through it. Upon each iteration, I called the numpy vstack function to add one of the dictionary arrays to the final multi-dimensional array, essentially transforming the dimensions of the final array from *m x n* to (*m +* 1) *x n* per loop.

A close up of a logo

Description automatically generated

**Figure 8: Multi-dimensional array for COV over each cell. Rows are individual cells, columns are top 50 positions.**

I also stored the sorted names of the cells in a separate array, in order to label the cells on the clustermap (this would prove impossible because there were 200 cells). I did not want the clustermap to cluster different positions, because I wanted them to remain in order. As I only wanted to cluster cells, I set a parameter in seaborn’s clustermap function to prohibit the rearrangement of the array along the vertical axis.

**5.** **Evaluation**

**5.1 Coverage Analysis**

According to the SAMTools *stat* calculations, the reads across both the normal and tumor cell populations had an average length of 92. The normal cell population consisted of 2,822,786 properly paired mitochondrial reads, whereas there were 2,084,056,878 total properly paired across the entire chromosome. On the other hand, the tumor cell mitochondrial chromosome had 3,024,434 properly paired reads. Normal and tumorous mitochondrial reads consequently took up about 0.0015% of the total number of reads, even though approximately 0.0000053% of the human genome (~3.1 billion base pairs) consists of mitochondrial DNA (~16500 base pairs). Additionally, the total coverage of the mitochondrial chromosome, calculated by *N* x *L* / *R*, where *N* is the number of reads, *L* the average length of the reads, and *R* the length of the genome, was 15683 over the normal sample, and 16083 for the mitochondrial sample. For the entire normal cell sample genome, the total coverage was about 61.85.

The MOSDepth calculations of coverage was more conservative—for normal mitochondrial cells, the total length of reads (*N* x *L*) was reported to be 221,765,560, with a mean coverage of 13384.37, whereas for the tumorous mitochondrial cells, the total length of reads was 228,644,249, with a mean coverage of 13799.52. Furthermore, MOSDepth reported coverages for all other chromosomes in the normal sample, which were all significantly lower than the coverage for the mitochondrial genome (almost all had an average coverage of 40-60). For the tumor sample, the range of coverage for the nucleic chromosomes ranged from 45 to 92.

A close up of text on a black background

Description automatically generated

**Figure 9: MOSDepth, Normal Sample. Total read length (column 3) and coverage per chromosome (column 4).**

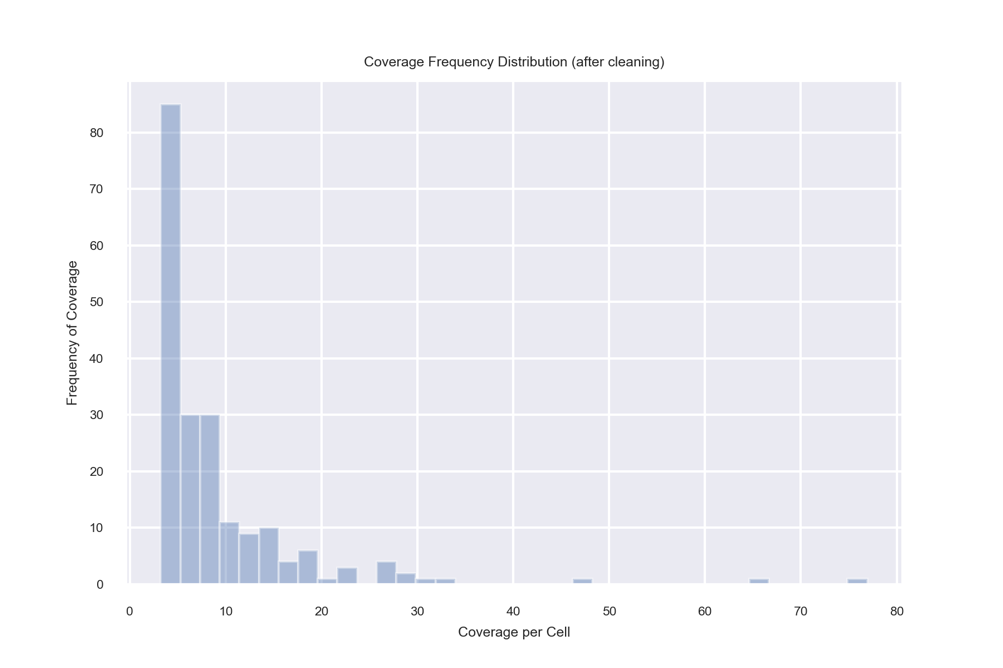
As for individual cell coverage, I found that there were over 100,000 unique barcodes within the original normal and tumor mitochondrial BAM files. However, most of the barcodes contained very few reads per cell, indicating that many of them were noise.

A screenshot of a cell phone

Description automatically generated

**Figure 10: (Tumor Sample) Initial Coverage Frequency Distribution per Cell**

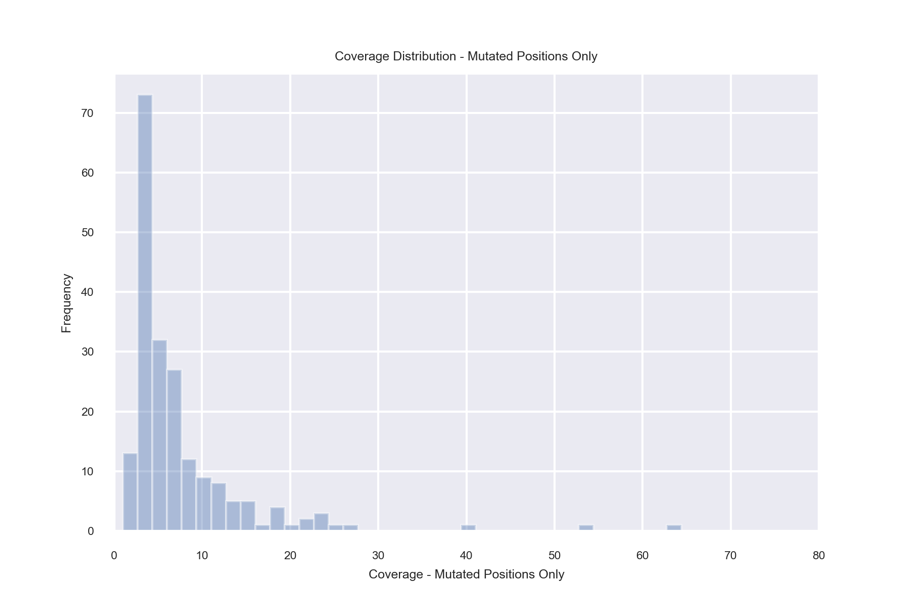
To account for the noise, I selected the top two hundred cells with the highest cumulative coverage. The median read count per cell amongst the top two hundred in the normal sample was 1110, and as the average read length was 92, this meant there was a median coverage of 6.2. The average read length was 1342, and standard deviation was 707. For the top two hundred cells of the tumor sample, the median read count was 1128.5, and median coverage consequently 6.3. The average read count was 1654, and the standard deviation was 1637, as the distribution had some extreme outliers on the high end. Compared to the original sequencing coverage of 0.03, this is about a 200-fold increase, suggesting that the number of mitochondria per cell within some cells in the dataset may be in the low hundreds.



**Figure 11: (Tumor Sample) Coverage Frequency Distribution per Cell**

**5.2 VarScan Analysis**

According to VarScan, there were 335 positions within the mitochondrial chromosome that contained at least one mutation for at least one of the cells in the sample. Therefore, about 2% of the mitochondrial positions contained mutations. The median cumulative COV value per cell, or, the total number of reads across the 355 mutated positions, was 1760.5. Therefore, the average depth would have been 5.2 over the mutated positions for the median cell.

****

**Figure 12: Coverage Frequency Distribution per Cell – Mutated Positions *Only***

This is lower than the median tumor depth calculated by SAMtools and MOSDepth earlier, but according to the VarScan website, “SAMtools and IGV show and count all bases at a given position, regardless of base quality. In contrast, VarScan requires that bases meet the minimum Phred quality score (default 15 for most commands) to count them for things like read counts.” [12]. This explains the slightly lower read coverage, and the shapes of the graphs of Figures 11 and 12 are still fairly similar.

Additionally, I calculated the distribution of the individual coverage frequency. This distribution records every individual value of coverage at every position across every cell:

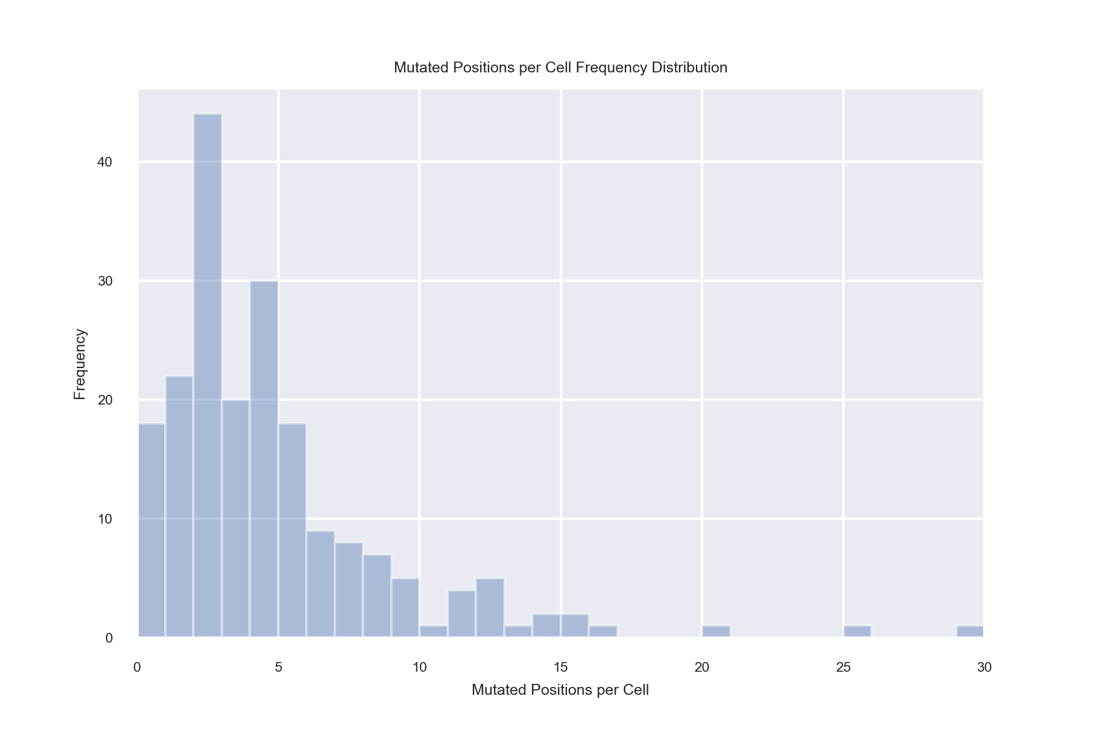
A picture containing white, kitchen

Description automatically generated

**Figure 13: Individual Coverage Frequency.**

This distribution has a median of 5, average of 7.9, and standard deviation of 9.5, which is consistent with the depth of 5.2 calculated from the previous graph. There is high variance for individual coverage per position per cell, as it is heavily skewed—there are only a handful individual coverages with high as compared to low read counts at any position within any cell. I confirmed that after filtering for the top fifty positions to account for noise, the shape of the graph of figure 13 did not change much—the median value only increased to 6, and the average to 8.4.

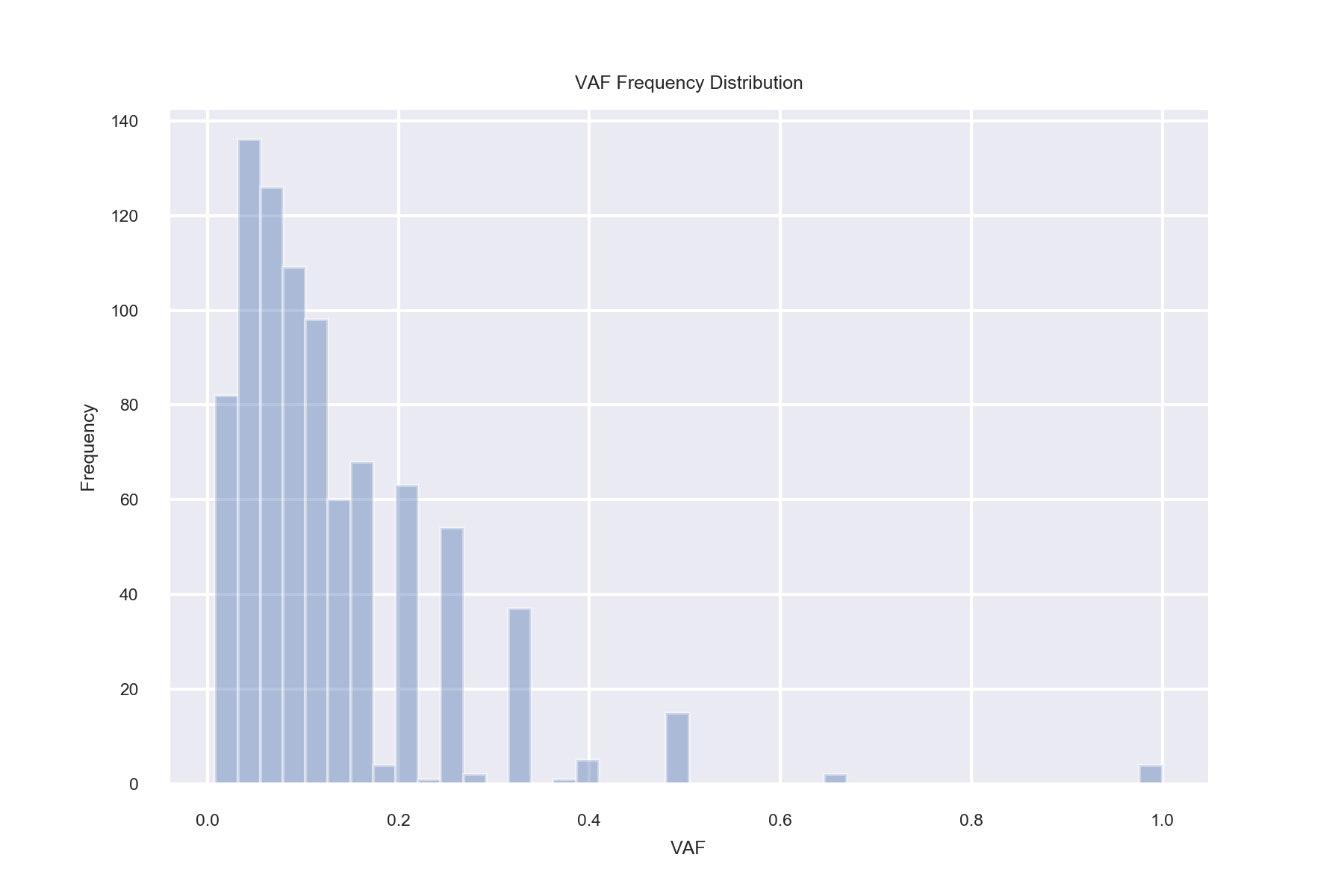
I used the highest cumulative mutation counts per position to determine the “top fifty positions,” calculating the number of mutated positions per cell afterwards:



**Figure 14: Mutated Positions per Cell Frequency Distribution**

The average number of mutated positions per cell is 4.335, the standard deviation 4.26, and the median 3, out of the top fifty positions. From the graph, it’s clear that some of the cells (eighteen, to be exact) did not have mutations in the top fifty positions even though they had exceedingly high coverage rates (~top 200 as calculated by SAMTools). It’s unlikely these cells descended from any of the others, otherwise they would have had mutations at the same positions. This indicates that those eighteen cells may be unrelated to the rest, or perhaps they could be normal cells. If the latter was the case, their mutations would be sequencing errors (and false positives on VarScan’s part, as the algorithm should have accounted for tumor purity). It’s also a possibility that these cells legitimately have mutations that exist outside the top fifty positions.

I also calculated the variant allele frequency over every position in every cell that had at least one mutation. The variant allele frequency is defined by MUTCOV divided by COV, and as a reminder, MUTCOV is the number of mutations at a position within a cell, while COV is the coverage at a position within a cell.



**Figure 15: Variant Allele Frequency Distribution**

The median variant allele frequency is 0.1, whereas the average is 0.13 and standard deviation 0.11. This shows that there is potential heteroplasmy across mitochondria within a cell. As the number of mutations is rarely as high as the coverage frequency at a position within a cell, that must mean that there is variation between the mitochondria internally to the cell—some contain mutations at certain positions, while others do not. Therefore, with an estimation of two hundred mitochondria per cell, I could make a rough approximation as to how many of them may have mutations at one of the top fifty positions in a typical case—200 x 0.1 = 20.

However, the reason that variant allele frequency reads were generally low (< 0.5) is because the vast majority of individual mutation coverages per position within cells are values of 1. There is rarely more than one mutation within a cell per position.

A screenshot of a cell phone

Description automatically generated

**Figure 16: Individual Mutation Coverage Distribution**

In fact, the median coverage value for positions that contain mutations is 11 (per cell), which is much higher than the original median coverage value of 5 from the distribution in Figure 13. This could explain why the variant allele frequency, MUTCOV divided by COV, is close to 0.1. Additionally, the correlation between MUTCOV and COV values is 0.49, indicating that there is an association between having higher coverages and detecting one mutation (or more, rarely).

Finally, I generated a cluster map (or, mutation matrix) detailing the occurrence of mutations over positions per cell:

A picture containing nature, bird

Description automatically generated

**Figure 17: Binary Mutation Matrix. black = mutation, white = none.**

From this cluster map, it’s apparent that there is not a single position where the majority of individual cells contain a mutation. This indicates that they likely do not all have the same common ancestor—there is evidence of distinct subpopulations of cells within the tumor sample. In fact, the same cells that did not contain any mutations within the top fifty positions are clustered near the bottom. Evidence of potential clustering is visible in other areas of the graph as well—cells with similar reads across positions are grouped together at the bottom of the map, as well as in various locations within the top.

1. **Conclusions and Future Work**
2. **Summary**

Through analyzing single-cell sequencing data, the bulk coverage of the mitochondrial chromosome has been shown to be much higher than all of the other chromosomes. Additionally, when broken down per single cell, the median coverage of the mitochondrial chromosome was shown to be significantly higher than the 0.03x coverage rate expected from the nuclear chromosome. By using a mutation caller on real breast tissue datasets, I have been able to identify and analyze single-point mutations within the mitochondrial chromosome.

1. **Limitations and Future Work**

As to be expected, there is still exceedingly low coverage per individual mitochondrial DNA sequence within a cell. Since mitochondrial DNA has high heteroplasmy, information about the diversity of mitochondrial mutations within an individual cell would still be sparse if some individual mitochondria per cell are not covered. Additionally, I selected my top two hundred cells through highest cumulative coverage, or total read length alone, which may have been biased towards cells with uneven coverage over positions. Furthermore, mutation callers have widely varying performance [13], so a high consistency of results across various somatic mutation detection algorithms may not be guaranteed, even considering the high coverage of mitochondrial DNA.

Future work could involve tracing the evolutionary lineage of subgroups of tumorous cells within a sample, determining which cells may be in clonal subgroups. One could also potentially identify driver mutations within mitochondrial DNA chromosomes that are significant in the progression of cancer and various diseases. Additionally, one could analyze the prevalence of various other mutations within mitochondrial chromosomes beyond single nucleotide variants—such as deletions, insertions, and large-scale structural variations.

1. **Bibliography**
2. Heather, James M. Chain. Benjamin (2016). “The sequence of sequencers: The history of sequencing DNA.” *Genomics*, 107, 1-8.
3. Ren, Xianwen. Kang, Boxi. Zhang, Zemin. (2018) “Understanding tumor ecosystems by single-cell sequencing: promises and limitations,” *Genome Biology*, 1-14.
4. Shalek, Alex L. Benson, Mikael (2017). “Single-cell analyses to tailor treatment.” *Science Translational Medicine*, 9, 1-3.
5. Raphael, Ben. “Lecture3-Final-ComputationalGenomics-Fall2019.” September 2019, Princeton. COS IW Seminar 09.
6. Rogers, Kara. “Mitochondrion.” *Encyclopedia Britannica.* [https://www.britannica.com/science/mitochondrion. 1 January 2020](https://www.britannica.com/science/mitochondrion.%201%20January%202020).
7. Satoh, Masaya. (1991) “Organization of multiple nucleoids and DNA molecules in mitochondria of a human cell.” *Experimental Cell* Research, 196(1), 137-140.
8. Gawad, Charles. Koh, Winston. Quake, Stephen. (2016). “Single-cell genome sequencing: current state of the science.” *Nature*, 17, 175-188.
9. Raphael, Benjamin. Dobson, Jason R. Oesper, Layla. Vandin, Fabio. (2014). “Identifying driver mutations in sequenced cancer genomes: computational approaches to enable precision medicine.” *Genome Medicine*, 6(5), 1-17.
10. Chatre, Laurent. Ricchett, Miria. (2012) “Large heterogeneity of mitochondrial DNA transcription and initiation of replication exposed by single-cell imaging.” *Journal of Cell Science*, 914-926.
11. Lehmann, Diana. Tuppen, Helen. Campbell, Georgia. (31 May 2019) “Understanding mitochondrial DNA maintenance disorders at the single muscle fibre level.” *Nucleic Acids Research*, 47(14), 7430-7443.
12. Ludwig, Leif. Lareau, Caleb. Ulirisch, Jacob C. (2019). “Lineage Tracing in Humans Enabled by Mitochondrial Mutations and Single-Cell Genomics.” *Cell,* 176, 1-15.
13. Unknown. “Support FAQ.” *VarScan.* <http://varscan.sourceforge.net/support-faq.html>. 2013.
14. Roberts, Nicola. Kortschak, Daniel R. Parker, Wendy T. Schreiber, Andreas W. (2013). “A comparative analysis of algorithms for somatic SNV detection in cancer.”*Bioinformatics*, 29(18), 2223-2230.
15. **Acknowledgements**

I would like to thank Dr. Ben Raphael for his encouraging, thoughtful guidance during our seminars every week. Additionally, I would like to thank Dr. Simone Zaccaria for contributing so much of his time towards helping me navigate the unfamiliar environment of bioinformatics, from sending me guides on SAMTOOLS to giving advice on the generation of Seaborn clustermaps. This project would have been significantly more difficult to complete without his assistance. I’m also grateful towards my IW classmates—Claire, Kevin, and Yaw—for providing helpful feedback on my project throughout the seminar.

*This paper represents my own work in accordance with University Regulations – Sean-Wyn Ng*