Basic Info

Title

Longread Sequencing - Evaluating Integrity of Consensus Sequence Generation from Reads Binned by Unique Molecular Identifiers (UMIs)

Team Members

- Caleb Cranney
    - Email: caleb.cranney@utah.edu
    - uID: u0704188
- Alexander Millar
    - Email: alexander.millar@utah.edu
    - uID: u0740821

Project Repository

https://github.com/illato/conseq

Background and Motivation

Both teammates are in the Biomedical Informatics (BMI) master's program at the University of Utah. In considering possible projects, it was agreed that we wanted to focus on a project with bioinformatics applications. Our proposed project focuses on long-read nanopore sequencing, a relatively new topic pertinent to the future of genetic sequencing. In addition to being a generally exciting topic with numerous visualization possibilities, we both felt this would be beneficial to our future careers.

Teammate Caleb Cranney currently works in the Biochemistry lab of Justin English (see here). Among the various current projects of the lab is VEGAS, a protocol for synthetically designing biomolecules via directed evolution. In essence, viruses with a specific protein are placed under evolutionary pressure to improve the efficacy of that protein. By the end of the protocol, we have the genetic blueprint for a "better" protein. We are interested in studying the changes that occurred in this process to improve the protein. In order to do so, the English Lab is developing a protocol for extracting and sequencing viruses throughout this process, allowing lab members to compare genetic changes over time from the original protein state to the final protein state. The end goal is to effectively create a phylogenetic tree and track major, widespread changes in the gene of interest. However, the sequencing process has several major obstacles, which Caleb was originally hired to help mitigate and that will be the focus of this project.

Nanopore sequencing is the ideal sequencing method for VEGAS. In addition to being less costly and more convenient, it can sequence a single entire virion genome in one pass. This is an ideal situation where we anticipate extracting viruses in various states of genetic modification. Illumina short-read sequencing would require stitching together portions of these genomes after the fact to find an original consensus, and as such the subtle differences we are trying to identify would be lost. Nanopore sequencing, while ideal, is also notoriously inaccurate. Several protocols exist to mitigate this problem. Generally, these methods involve duplicating the reads before sequencing them, then comparing the duplicates computationally to generate the "original," accurate consensus sequence. Most methodologies rely on a reference genome or sequence to compile the consensus. However, relying on the reference genome risks overlooking small changes that may have been introduced since the virus diverged, and is therefore not viable for the VEGAS strategy. Thus, the English lab has been developing a protocol and software for *de novo* consensus sequence generation (see this paper for the protocol their current approach is based on). This includes marking individual virion genomes with Unique Molecular Identifier (UMI) tags, in addition to duplicating and sequencing them. Caleb has written software for binning sequences by UMI tag in his work, as well as a candidate consensus sequencing algorithm, and is now in the process of evaluating its applicability to VEGAS. Alexander and Caleb both believe the application of interactive data visualization will be substantially beneficial in the evaluation, refinement, and furthering of this work.

Our proposed project focuses on evaluating the validity of consensus sequencing algorithms. See Project Objectives for more details.

Project Objectives

As stated previously, nanopore sequencing is notoriously inaccurate. This is augmented by the reality that these errors are often systematic – specific sequences are more likely than others to have specific sequencing errors. Thus when a consensus sequence is generated it would be hugely beneficial to look at common discrepancies between 1) each of the “binned” sequences that were used to generate the final consensus and 2) the final consensus itself. For example, for a consensus sequence that was generated by comparing 100 binned sequences, what is the most common discrepancy between these sequences and the final result? If 30 of those sequences had a C->A mutation at base pair 67, then we may want to look at that location more carefully in future experiments that rely on the same genome.

In general, we are hoping to create a visual that explores differences between the binned sequences and their generated consensus sequence. Some questions that could be answered are:

- What are the most common differences between the binned sequences and the consensus?
- Are specific error types more likely than others (insertion, deletion, mutation)?
- Are there specific base pairs that are more prone to errors?
- Are there entire areas of the consensus sequence that have a higher density of errors than others?
- Are specific error types concentrated in these base pairs/areas?

All of the above questions focus on the generation of a consensus from a single collection of binned sequences. However, in a real experiment, we would generate multiple consensus sequences, one for each of numerous bins of sequences. In a VEGAS experiment, these consensus sequences would be descended from the same original viral genome. So, in addition to being able to view individual bins, there would be benefit to seeing how these errors exist across multiple bins. Some questions that could be answered are:

- Are there any consensus sequences that have a unique concentration of errors in a given area?
- Are there any areas where the consensus sequences tend to have similar error occurrence patterns?
- Are there any areas where the consensus sequences tend to have dissimilar error occurrence patterns?

Data

Data for this project is generated by members of the English Lab and is located on their Lab computer. The consensus sequence algorithm element of the project is still in development, so we are not performing directed evolution experiments as of yet. Test data, however, is still generated from virion genomes as would apply in a VEGAS directed evolution experiment. These genomes each have a unique barcode inserted at a specific locus. This was largely done to verify the binning process succeeded, however, given that this locus is expected to vary substantially from consensus to consensus sequence it also benefits any attempts to visualize areas of frequent disagreement.

Data Processing

We do not expect substantial data cleanup. The consensus sequence algorithm Caleb developed relies on tracking differences between the consensus sequence and an alignment of the binned sequences. While running this program, it would be relatively trivial to write these differences out to a CSV-delimited file.

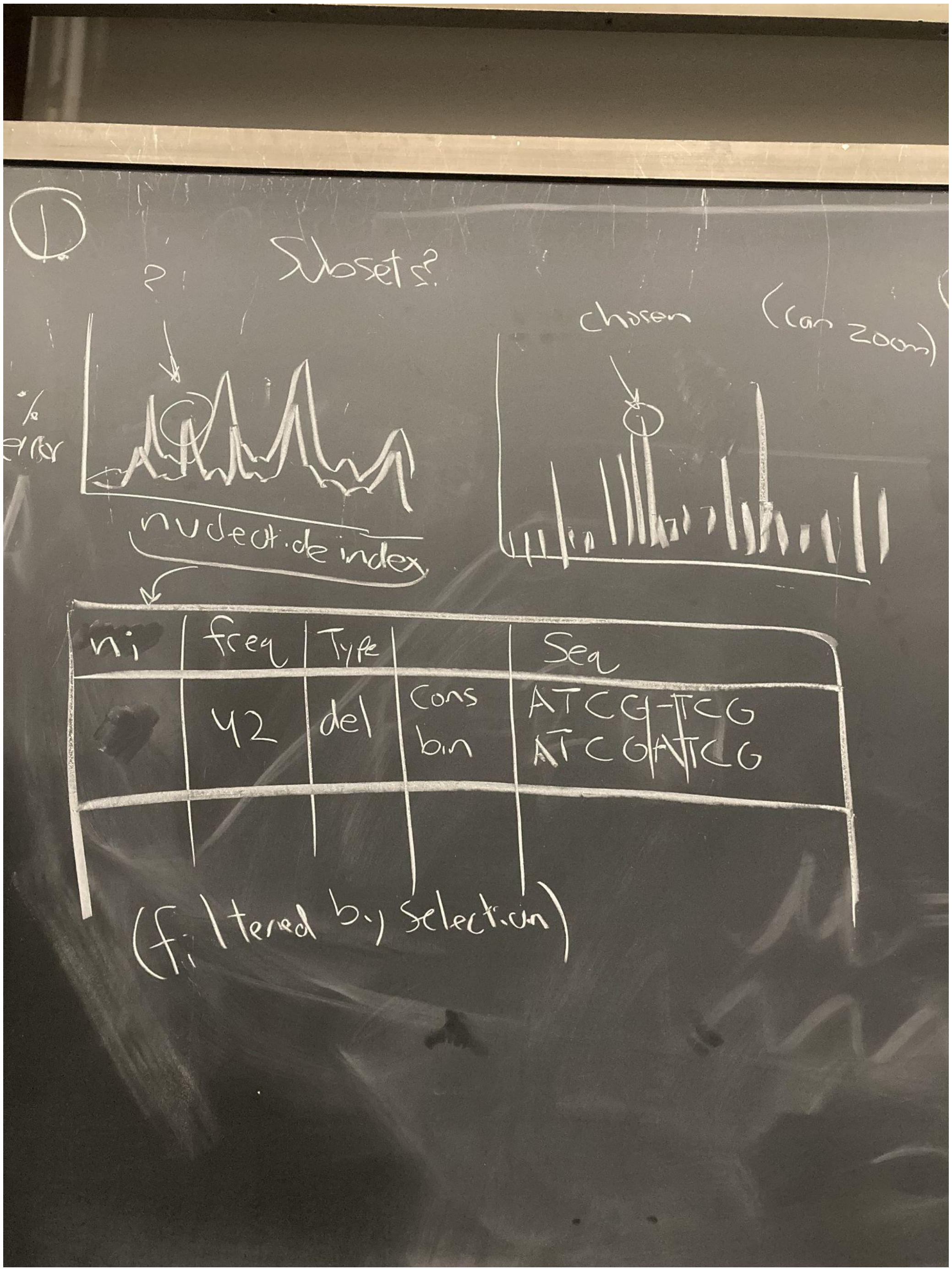
The current plan for the data format is to use a table with the following columns:

- ‘start’: The index of the consensus sequence where the aligned binned sequence begins to diverge.
- ‘end’: The index of the consensus sequence where the aligned binned sequence ends diverging.
- ‘insert’: The portion of the binned sequence that, when inserted between the ‘start’ and ‘end’ indices, would make the consensus sequence and binned sequence identical at this locus.
- ‘binNum’: An identifier for the specific consensus sequence. Lower bin numbers indicate the consensus sequence was generated from a larger bin of sequences.
- ‘seqID’: An identifier for the specific bin sequence. This is in case we come up with a useful idea for visualizing individual bin sequences against their consensus sequence, though we may get rid of this column if we decide it is a waste of memory. See Optional Features.

While not expecting too much cleanup, we will be considering ways to compress this data for use in the visualization. There are many individual differences between each binned sequence and their consensus, and having a CSV file to the order of millions of rows would not be out of scope for some projects. One example for how we may do that is to group all identical errors (same ‘start’, ’end’, and ‘insert’ values) and add a column that lists bin numbers where that error was identified, substantially shortening the table. One potential problem with this approach is the computational expense of getting all the errors for a particular bin. As we implement features that view the data from different angles, we will continue to evaluate the implications on size/performance balance. At scale, a database (e.g., SQLite) may be a more practical solution, but this is outside the scope of this project – the primary focus being visualization.

Visualization Design

Prototype Visual 1:



ni	freq	Type		Seq
	42	del	cons bin	ATCG-TCG ATCGATCG

The first prototype was designed primarily with function in mind - searching for and obtaining specific data in a format scientists are familiar with. There are three components to this prototype visual, a line graph (top left), bar chart (top right), and a table (bottom).

a) The line graph is a high-level representation of all consensus sequences and the frequency of errors in their subsequent binned sequences. By showing all error frequencies together and normalizing their portrayal by using percentages, one could identify consensus sequences that had areas with an unusual concentration of errors, or sections where consensus sequences frequently did/did not have errors.
    i) There would be a line for each consensus.
        1) Interactively, one could select a line, which would then populate the bar plot and table with all data pertaining to that consensus and its binned sequences.
    ii) The x-axis would be the sequence index where an error could be found (index of a specific nucleotide).
    iii) The y-axis is the percentage of binned sequences with an error at the given index (number of sequences with a discrepancy at that index / the total number of sequences). This would likely need to be a percentage because the specific number of errors would be strongly impacted by the number of sequences in that bin, requiring a normalizing approach like percentage.

b) The bar graph drills down into a specific consensus and the discrepancies between it and the bin of sequences used to create it.
    i) Each bar represents errors at a specific index of the consensus.
        1) Interactively, selecting a bar would filter the table to only show errors found at that index. Selecting the bar would cause it to change color to show it was selected. You could probably select more than one bar, and the table would show them together.
        2) Optional: We could color these bars based on the specific nucleotide at that index, one each for adenosine, thymine, cytosine, and guanine (ATCG).
    ii) The x-axis would be the sequence index where an error could be found (index of a specific nucleotide).
    iii) The y-axis is the count of errors at the given index. This could alternatively be shown as a percentage as in a)iii), or a second y-axis could be added to the right side of the histogram.
    iv) Optional: Filtering errors by error type (insertion, mutation, or deletion), shortening the bars and possibly changing their color.
    v) Optional: In line with iv), we could possibly do a stacked bar chart with three possible colors for each error type.

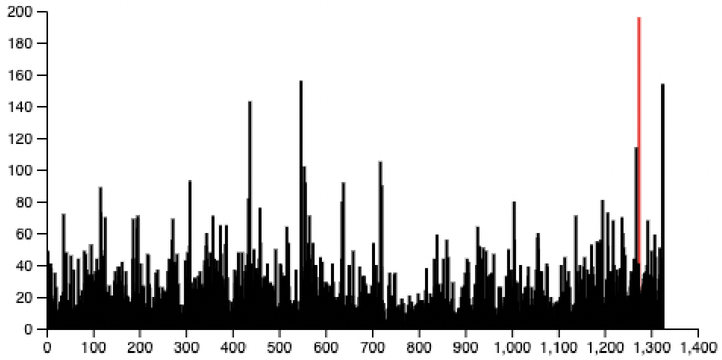
c) The table
    i) Each row would refer to a specific error at a specific index of the consensus sequence where there is a discrepancy with one or more sequences in the corresponding bin. There would likely need to be five columns. The first three would need to be sortable.

1) Nucleotide Index: The index of the error. When a specific bar is selected, all values in this column would reflect the selected index.
2) Error Type: Indicates if the error is an insertion, mutation, or deletion. For clarity, these would each have a specific color font that would match the changes in the sequence comparison later.
3) Frequency: Indicates the frequency of the specific error.
4) Placeholder column: include text to differentiate what is found in the consensus sequence and what is in the binned sequence for this error.
5) Sequence Comparison: Showing an alignment of the consensus and binned sequence to show what the error looks like.
    (a) This is a must-have for the project - somewhere, we need to be able to drill down and clearly show the error in question.
    (b) To highlight the difference, a color change for the differential area would likely be required.

ii) Because this table would be pretty long, we'd likely need a cutoff of how many were presented at any given time. Say, it shows 20 rows, then provides the option of showing the next 20, etc.

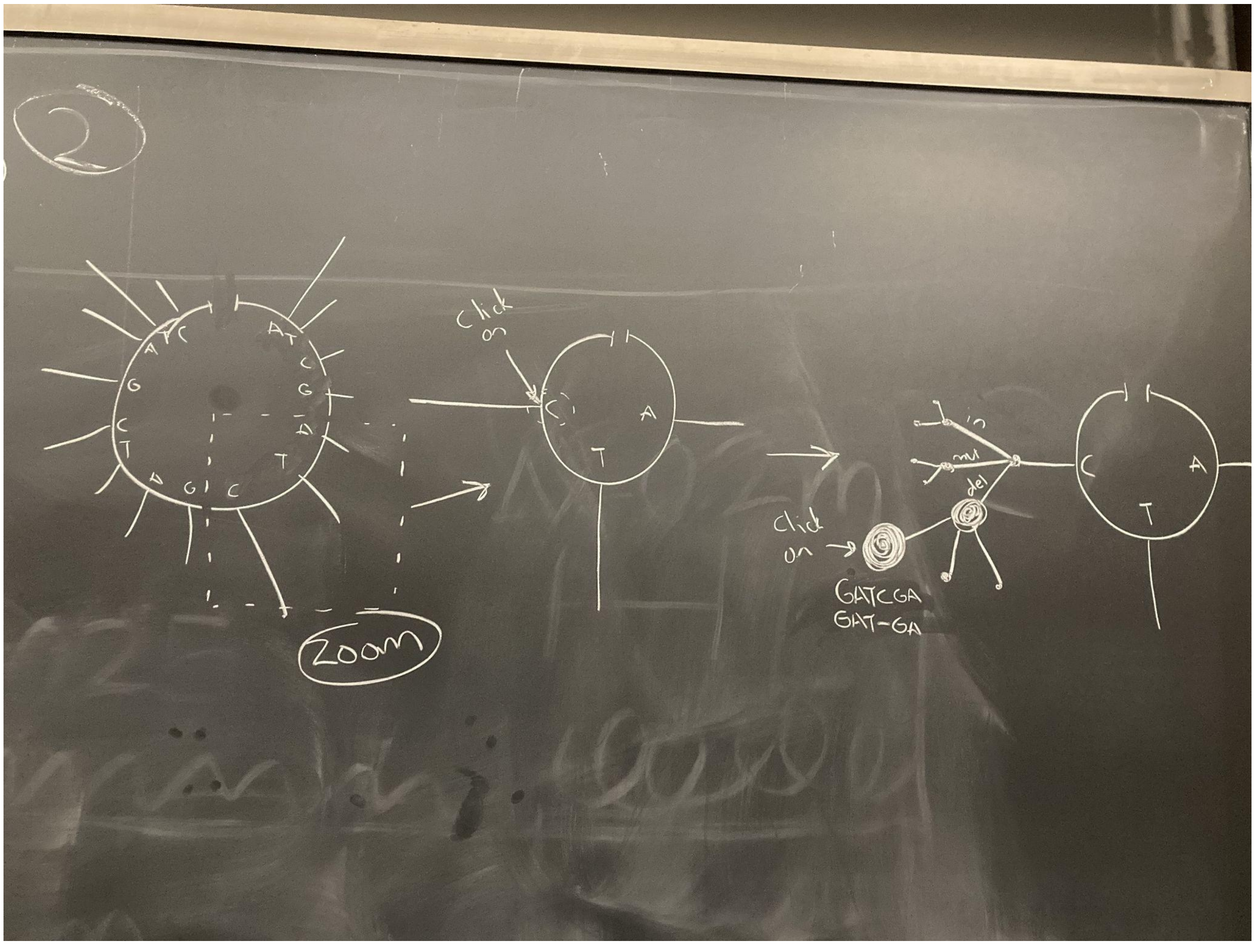
We started tinkering with what this may look like and created the following graphic for the bar chart (b) and the table (c), with the bar at index 1271 of the bar chart selected.

Bar Charts



Nucleotide Index	Error Type	Frequency	Sequence Comparison
1271	deletion	174	Consensus Sequence: CCTCCCCCGTGCCTT Bin Sequence: CCTCCCC-GTGCCTT
1271	mutation	12	Consensus Sequence: CCTCCCCCGTGCCTT Bin Sequence: CCTCCCCTGTGCCTT
1271	insertion	3	Consensus Sequence: CCTCCCC-CGTGCCT Bin Sequence: CCTCCCCGCGTGCCT
1271	insertion	2	Consensus Sequence: CCTCCCC-CGTGCCT Bin Sequence: CCTCCCCACGTGCCT
1271	mutation	2	Consensus Sequence: CTCCCCCGTGCCTTC Bin Sequence: CTCCCCA-TGCCTTC
1271	deletion	1	Consensus Sequence: CTCCCCCGTGCCTTC Bin Sequence: CTCCCC--TGCCTTC

Prototype Visual 2



This visual was designed for portraying a single bin of sequences/consensus in a way that encapsulated all the data of b) and c) of prototype visual 1. There are three new representations introduced, namely a circular representation of the consensus sequence (left), a dynamic method for zooming in on particular subsections of that sequence (middle), and a tree representation of specific errors and their frequency (right).

a) The circle represents a kind of circular representation of the bar chart in 1.b. The circle itself is made of the nucleotides of the consensus sequences, with bars extending from the outside of the circle indicating frequency of errors (y-axis of the bar chart).
    i) Optional: We could color these bars based on the specific nucleotide at that index, one each for adenosine, thymine, cytosine, and guanine (ATCG).
b) The diagram in the middle of the above drawing would be formed from the dotted-line square drawn in a) above. Interactively drawing this square, all nucleotides outside of the

square are filtered out, and the selected portions would expand to fill the circle. The nucleotide font size would likely increase to fill the circle as best as possible.

c) The diagram on the right was a method we invented for portraying the frequency of specific errors at a given index. After clicking on a nucleotide (‘C’ was selected above), the bar previously used to indicate error frequency would be replaced by a tree. We may constrain the number of nucleotide indices that can have a tree expanded at a given time, possibly even to a single index. In this case, clicking on another index’s bar would collapse any open tree. A tree would have the following characteristics:

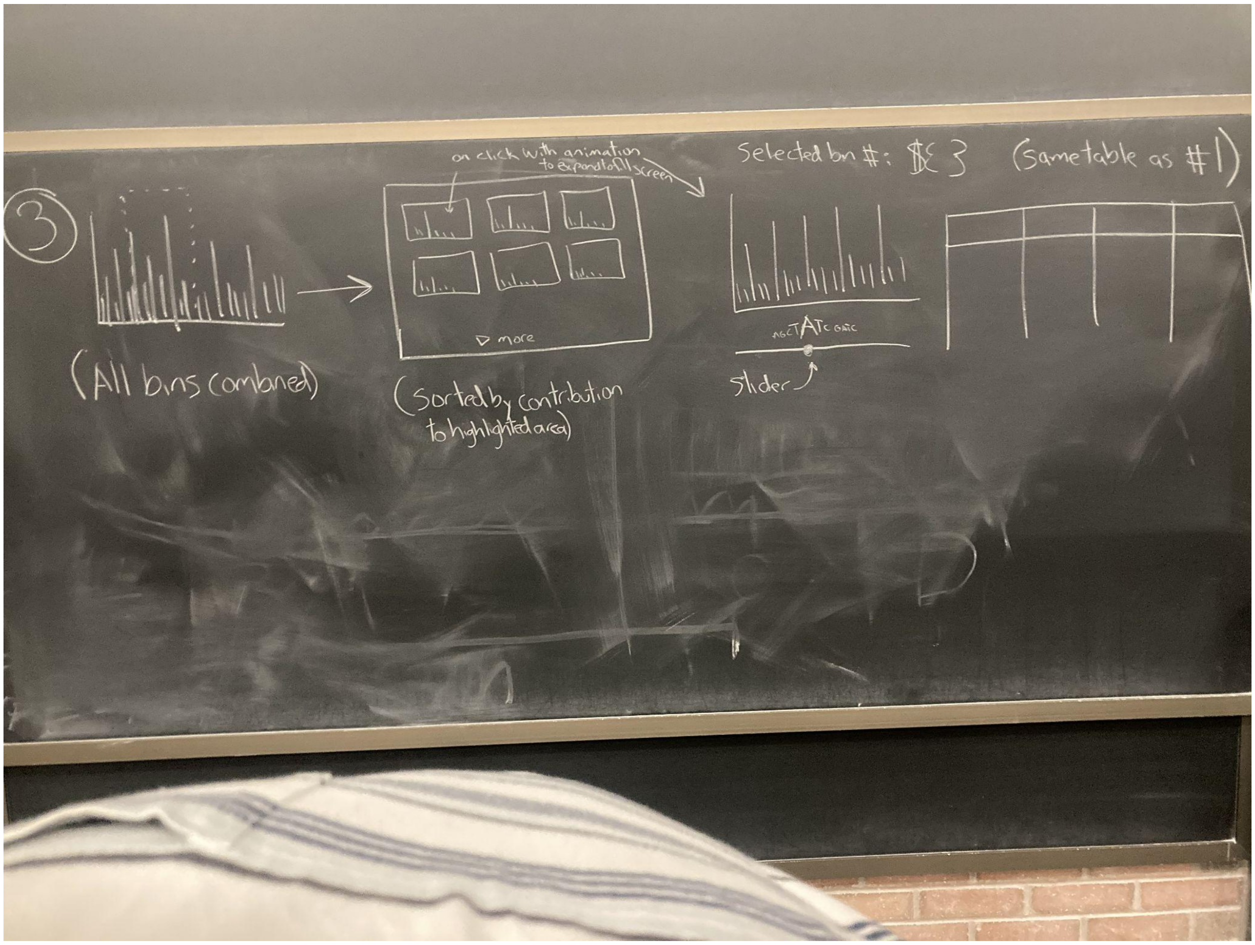
    i) The area of each node is used to express the frequency of all collective errors of the nodes in child branches. Therefore later nodes are either the same size as the parent or smaller.

    ii) A single ‘trunk’ from the circle to the first node. This merely extends it away from the remainder of the circle.

    iii) Up to three branches separating at the first node. These are for separating into error types of insertions, mutations, and deletions, respectively. The three nodes at the end of this branch and all children nodes would be colored by error type. For an example of coloring please see our final visual drawing.

    iv) A single branch is then created for each specific error identified at that nucleotide, ending with a node that represents the frequency of that specific error.

Prototype Visual 3


3
on click with animation to expand to full screen
Selected bin #: ${ }
(Same table as #1)
▷ more
AGCTATC GATC
(All bins combined)
(Sorted by contribution to highlighted area)
Slider

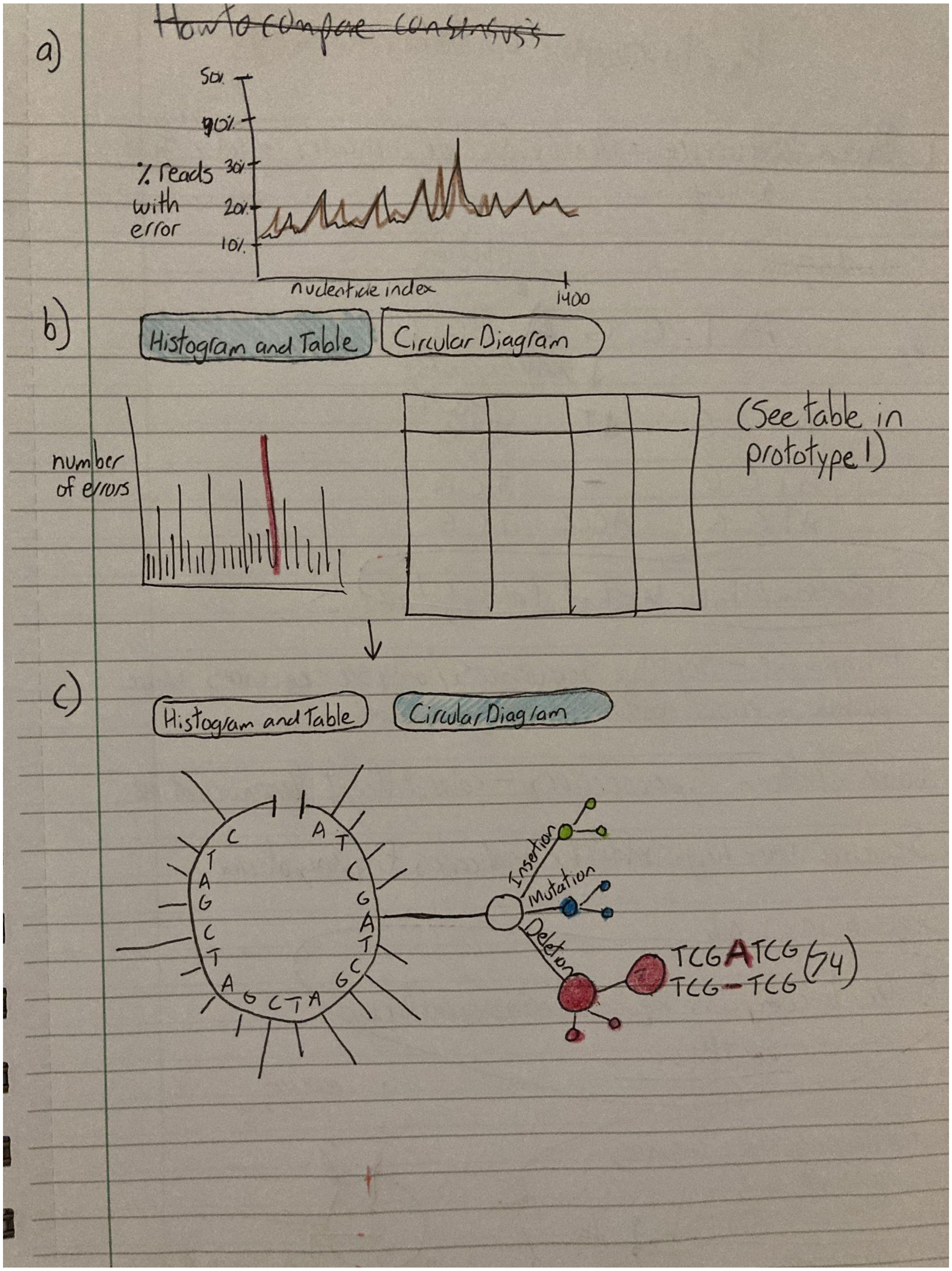
This prototype was developed to address two primary concerns. One, we felt we hadn't found a good way to visualize a consensus sequence in its entirety out of zooming options. With over a thousand base pairs or 'ticks' in a theoretical bar chart, these have not scaled well with our ideas. Second, we wonder if the line chart described in 1.a would become too cluttered to be meaningful. If there's a line for each consensus, and there are over 500 consensus sequences, there may be too much data for a line graph to reliably capture. So, this visual has the following characteristics:

a) The initial view will show all the errors from all the bins combined into one bar chart (left). These would share all characteristics of the bar chart in 1.b, but represent all bins instead of just one. This would reduce the clutter of a line graph. However, to inspect the individual contributions of specific bins, one can highlight a portion of the peaks to trigger the second view.
b) The second view is a grid of boxplots (middle left). After selecting a section of bars in the initial view, the program would determine which bins had on average the greatest percentage contribution of errors for the selected section. Each bin boxplot would be shown, starting with the top six 'contributing' bins.

    i) The selected portions would be highlighted in each boxplot of the grid, likely by changing their color.
    ii) There would be the option of revealing more boxplots after the initial six, either by expanding the bottom to reveal more than six boxplots or by being able to switch views to numbers seven through twelve, then thirteen through eighteen, etc.

c) The final view would be a barchart (middle right) and table (right), similar to what was described in prototype 1.b and 1.c. However, we altered the interactivity of the chart.
    i) Instead of interactively choosing a single bar in the bar chart, there would be a slider that ran along the bottom of the graph. Moving this slighter would highlight specific nucleotides by enlarging or magnifying them compared to their neighbors. The slider would only be able to highlight one nucleotide at a time, simultaneously magnifying the nucleotide and changing the color of the bar represented by that nucleotide (much like a mouseover event changing its color). When the slider is released, the table would filter according to the selected nucleotide index.

Final Visual



This visual summarizes several of our prototype ideas into a single graph. Note that we are allowing the user to choose between a “Histogram and Table” view as well as a “Circular Diagram” view, as described in 3.b and 3.c respectively.

a) We are currently settling on the line graph method (1.a) for showing the high-level consensus sequence comparisons, as that would allow us to select bins that have abnormal error patterns. This would be at the top of the visual.

b) This would be the histogram and table chart from 1.b and 1.c respectively. Some additions/alterations to the original idea described in prototype 1:

   i) Similar to prototype 2, we are going to implement a zooming function to the bar chart. Zooming will be performed using the mouse wheel or by brushing the overlay. Selection of bars in the histogram will be achieved by individual click or by brushing multiple bars, updating the table accordingly.

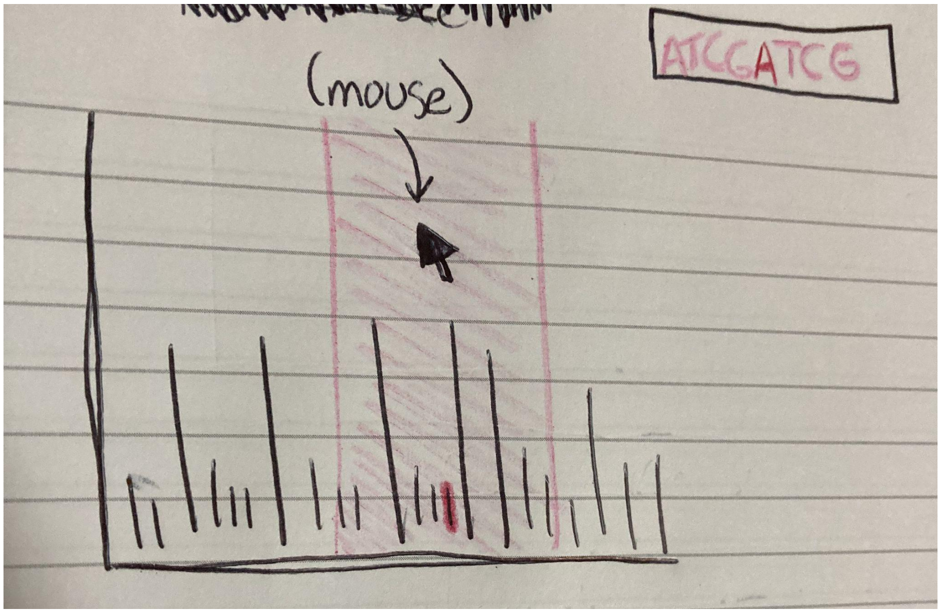
   ii) Instead of the slider described in prototype 3, we opted that a mouseover event that highlighted a subsection of 10-15 nucleotides would highlight sequences of interest more clearly. A box would appear over the 10-15 nucleotides around which the mouse was hovering, and those nucleotides would appear in a box over the histogram. See below image for a drawing.

c) This would be the circular graph described in prototype 2. The nucleotide highlight box described in 3.b.ii would apply here on mouse hover of the frequency bar.

   i) Note: We opted to not use the slider because it would not translate well to the circular diagram.

   ii) For all zoom/selection functions, there will be means to return to the initial state.

For the 3.b.ii description:



Must-Have Features

- Global view
    - Line chart
        - Line selection
            - Transition to Single view
        - Crowded Selection
            - Brush and Zoom
- Single view
    - Navigation/State
        - Back (to Global view) button
        - Reset (state/selection) button
    - Histogram and Table view
        - Histogram
            - Bar selection
                - Single Selected Color
                - Populates table
        - Table
            - Sortable
                - Index
                - Type
                - Frequency
            - Sequence Comparison
                - Discrepancy colored
    - Hybrid view
        - Radial
            - Brushing (2D) Subset Selection
                - Selection Spreads to Fill Radial
            - Bar Selection
                - Populates Tree
        - Tree/Network
            - Hierarchical Expand/Collapse
            - Leaf Selection
                - Frequency
                - Sequence Comparison

Optional Features

- Global view
  - Line chart
    - Line selection
      - Transition to Single view
    - Crowded Selection
      - Brush and Zoom
      - Brush/Pan and Zoom
- Single view
  - Navigation/State
    - Back (to Global view) button
    - Reset (state/selection) button
  - Histogram and Table view
    - Histogram
      - Bar selection
        - Single Selected Color
        - Populates table
      - Brush and Zoom
      - Brushing Bar selection
        - 1D (x-axis)
        - 2D (x-axis & Y-axis)
        - Nucleotide-Specific Selected Color
        - Populates table
      - Histogram Filtering
        - Error Type(s) (Insertion/Deletion/Mutation)
      - Bar Hover
        - Peek Nucleotide Sequence
    - Table
      - Sortable
        - Index
        - Type
        - Frequency
      - Sequence Comparison
        - Discrepancy colored
    - Hybrid view
      - Pan/Zoom
      - Radial
        - Brushing (2D) Subset Selection
          - Selection Spreads to Fill Radial
        - Bar Selection
          - Populates Tree
        - Nucleotide Labels
          - Scale to Fit
        - Bar Hover
          - Peek Sequence
      - Tree/Network
        - Hierarchical Expand/Collapse
        - Leaf Selection
          - Frequency
          - Sequence Comparison
  - Consensus vs Selected Bin Sequence(s) view

Project Schedule

- Week 10 – **27 Oct** 2022 @ 23:59
    - Project Peer Feedback – **25 Oct 2022**
    - Goals
        - Histogram – **CC**
            - Bar Selection – **CC**
            - Crowded Selection - Brush and Zoom – **AM**
    - Stretch Goals
        - Line Chart – **AM**
            - Line Selection – **AM**
        - Table – **CC**
            - Sortable – **CC**
            - Sequence Comparison – **CC**

- Week 11 – **3 Nov** 2022 @ 23:59
    - Goals
        - Line Chart – **AM**
            - Line Selection – **AM**
            - Crowded Selection - Brush and Zoom – **AM**
        - Histogram
            - Bar Hover – **CC**
                - Peek Nucleotide Sequence – **CC**
            - 1D Brushing Selection – **CC**
    - Stretch Goals
        - Radial – **CC**
        - Navigation – **AM**

- Week 12 – **10 Nov** 2022 @ 23:59
    - Schedule time-slot with Staff Member – **11 Nov 2022**
        - Project Review Meeting – Following Week
    - Goals
        - Radial
            - 2D Brushing Subset Selection – **CC**
            - Bar Selection – **AM**
                - Populate Tree – **AM**
        - Tree – **AM**
    - Stretch Goals
        - Radial
            - Bar Hover – **CC**
                - Peek Nucleotide Sequence – **CC**
        - Tree
            - Expand/Collapse – **AM**

- Week 13 – **17 Nov** 2022 @ 23:59
    - Project Review Meeting with Mentor – 14-18* **Nov 2022**
    - Goals
        - Radial – **CC**
        - Tree
            - Node
                - Scaling – **CC**
                - Coloring – **CC**
            - Leaf Selection – **AM**
                - Frequency – **AM**
                - Sequence Comparison – **AM**
    - Stretch Goals
        - Hybrid
            - Pan and Zoom – **AM**
            - Radial
                - Scaling – **CC**
                - Coloring – **CC**

- Week 14 – **24 Nov** 2022 @ 23:59
    - Goals
        - Histogram
            - 2D Brushing Selection – **AM**
            - Error Type Filtering – **CC**
    - Stretch Goals
        - Tree
            - Leaf Selection
                - Consensus vs Bin Sequence(s)
                    - Line Chart – **AM**
    - **Thanksgiving Break**

- Week 15 – **1 Dec** 2022 @ 23:59
    - Final Project Due – **2 Dec 2022**
    - Goals
        - Transitions/Scaling/Navigation/Misc – **AM**/**CC**
    - Stretch Goals
        - Consensus vs Bin Sequence(s)
            - Line Chart – **AM**
                - Subset/Filtering/Selection – **CC**