Mito-AssemblyVis: Mitochondrial Genome Assembly Assessment Visualization

Armaghan Sarvar (armaghansarvar@gmail.com), Cecilia Yang (ceciliayang1276@gmail.com)

October 21, 2022

1 Introduction

Mitochondrial genomes (mitogenomes) are short and circular DNAs in cellular organelles called mitochondria. Mitogenomes are significantly more abundant than nuclear genomes per cell and are less prone to degradation thereby allowing forensic and environmental scientists as well as anthropologists to obtain DNAs from samples that consist of little biological material [1]. In the past, the high sequencing cost and the inaccessibility of DNA sequencing technologies led to a shortage of complete mitogenomes of some culturally and environmentally important animal species in public databases such as the National Center for Biotechnology Information (NCBI). Nowadays, whole genome sequencing technologies have become much cheaper and more easily accessible, making it possible to assemble mitogenomes on a large scale. To ensure the accuracy of the assembled mitochondrial DNA sequences, choosing effective visualization methods for the quality assessment of mitogenome assemblies is very crucial in the field.

During the past year of Cecilia's graduate studies, she has focused on creating a mitogenome assembly pipeline that takes in short sequenced DNA reads as input and generates a reconstructed mitogenome as output. Visualization tools can be applied to provide insights into the accuracy of the reconstructed mitogenome which will eventually allow us to assess the performance of her pipeline compared to other published mitogenome assembly pipelines.

The key attributes we intend to visualize include sequence length, coverage, quality metrics (e.g., number of gaps and misassemblies), contiguity statistics (e.g., NG50), mapped regions between two or more sequences, and genome annotation results. Mitogenomes are approximately 16,000 base-pair long, so sequence length will help us evaluate the completeness of the assembled sequence. Coverage indicates the number of DNA reads mapped to a given nucleotide in a reconstructed sequence [2]. Higher genome coverage is often associated with better genome assemblies, meaning the resulting reconstructed genome is close to its original form. Contiguity statistics indicate whether the reads are joined correctly to provide a contiguous representation of the genome. Quality metrics reflect the overall accuracy of the assembled genome. Genome annotation is a method that determines the specific gene locations. Potential genomic rearrangements and the completeness of assembled mitogenomes can be investigated by visualizing the mapped regions between the reconstructed and a reference genome and comparing their annotation results.

2 Related Work

This work will propose a visualization toolbox for assessing the output of mitogenome assembly pipelines. Hence, here, an overview of some of the visualization tools used in genomics is provided.

Gosling [3] is a grammar for interactive genomics data visualization accompanied by the JavaScript toolkit Gosling.js that balances expressiveness for multi-scale genomics visualizations and provides scalable rendering. Gosling.js is built on an existing platform for web-based genomics data visualization to simplify the visualizations of various genomics data formats.

Considering related tools that look into contigs or chromosomes in a fine-grained manner, we could mention the Integrative Genomics Viewer (IGV) [4], which is an interactive linear genome browser tool that supports the integration of the common types of genomic data, investigator-generated or from publicly available sources. In the context of our

project, it is particularly useful when the user needs to investigate the reads aligned to an assembly, to derive information such as the coverage. An improvement over IGV is the Ribbon tool [5] that provides better support for visualizing long-read sequencing data, showing how alignments are positioned within both the reference and read contexts. This gives an intuitive view that enables a better understanding of structural variants and the read evidence supporting them.

Icarus [6] is a visualization tool that looks more specifically into genome assemblies. Icarus is an open-source software integrated with QUAST (A quality assessment tool for genome assemblies) [7] and complements its output with interactive visualizations of assembly alignments to the reference genome, and various features detected by QUAST (e.g. misassemblies).

Considering looking at the assembly aligned to the reference genome, there is also Xmatchview [8] that compares pairs of genomes to each other. Although it has been used to compare chloroplast assemblies to an NCBI reference, it does not handle multi-fasta sequences; hence, one could use it to compare two scaffolds in the same genomic region, but not multi-chromosome.

Recently, many algorithms trying to detect large genomic rearrangements using Circos [9] for their visualization step. Circos, is a Perl-based visualization software that requires setting up numerous configuration files with a large number of parameters. R packages like RCircos [10] or ggbio [11] provide functions to display circular plots for genomic data. However, these tools are very general and do not filter specific genomic inputs.

Hiplot [12] is a python package that facilitates multi-dimensional data visualization, this tool was initially designed for deep-learning hyperparameter tune-ups. We proposed an alternative implementation for this tool - displaying the assembly results generated by different sets of assembly parameters which help potential users tune up their assembly pipelines.

Although the visualization tools discussed above have one or more features suitable for mitogenome assembly evaluation visualization, there is currently a lack of visualization tools that are specifically designed for mitochondrial genome assembly evaluations.

Since one of the project goals is to compare the performance of our novel assembly pipeline with the previously published ones, some of the published mitogenome assembly algorithms are summarised here: GetOrganelle [13] and MitoZ [14] are two state-of-the-art mitochondrial genome assembly tools that can assemble mitogenomes from whole genome sequencing data. Since they also assemble complete mitochondrial genomes and produce all possible configurations of the circular genomes, we will utilize them to assemble mitogenomes using the DNA reads of the species we are interested in. MitoZ also has a visualization component that allows users to view the final assembly products, but only a single sequence can be visualized at once. To compare the sequences by different assembly tools, we would need to include a Circos-based plot in our final visualization.

3 Data and Task Abstraction

3.1 Data Abstraction

Three culturally and environmentally important animal species in Canada were selected including grizzly bear, sea otter, and canada goose. Their reference sequences and DNA read data are downloadable on NCBI. Each species sample will be assembled using three mitogenome assembly pipelines - a novel pipeline developed in our lab, MitoZ, and GetOrganelle, resulting in three datasets for visualization. The key attributes of the datasets are shown in Tables 1 and 2.

3.2 Task Abstraction

Assessment of mitogenome assemblies gauges both the completeness and contiguity of an assembly and helps provide confidence in downstream biological insights. In order to compare quality across multiple assemblies, a set of standard metrics are typically calculated and then compared to one or more gold-standard reference genomes. While several tools

Table 1: Key Attributes in the Dataset Generated from the Novel Assembly Pipeline

Variable	Type	Description	Possible Values
k	Categorical	K-mer size used in assembly	k=64, 80, 96
kc	Categorical	k-mer minimum coverage multiplicity cutoff	kc=2, 3, 4
Gene Start Position	Spatial	Starting position for an annotated gene	24
Gene End Position	Spatial	End position for an annotated gene	100
Sequence Length	Quantitative	Length of assembled sequence	16000bp (base-pair)
Number of Gaps	Quantitative	Type of assembly quality metric	1
Number of Misassemblies	Quantitative	Type of assembly quality metric	1
Number of Sequences	Quantitative	Type of assembly quality metric	1
Coverage	Quantitative	Type of assembly quality metric	100x

Table 2: Key Attributes in the Datasets Generated from MitoZ and GetOrganelle Pipelines

Variable	Type	Description	Possible Values
k	Categorical	K-mer size used in assembly	k=64, 80, 96
Gene Start Position	Spatial	Starting position for an annotated gene	24
Gene End Position	Spatial	End position for an annotated gene	100
Sequence Length	Quantitative	Length of assembled sequence	16000bp (base-pair)
Number of Gaps	Quantitative	Type of assembly quality metric	1
Number of Misassemblies	Quantitative	Type of assembly quality metric	1
Number of Sequences	Quantitative	Type of assembly quality metric	1
Coverage	Quantitative	Type of assembly quality metric	100x

exist for calculating individual metrics, applications showing comprehensive evaluations of multiple assembly features are, perhaps surprisingly, lacking. Our visualization tool will allow a bioinformatician to assess a genome assembly algorithm and compare its output with other assembly tools. The levels of this assessment are as follows:

At the assembly level, bioinformaticians often need to fine-tune the assembly toolbox parameters to get the best-assembled genomic sequence based on their pre-determined evaluation criteria. Hence, a systematic visualization solution for fine-tuning the hyper-parameters would be advantageous.

At the analysis level, the bioinformatician would want to know how identical the assembled output is compared to a ground-truth reference genomic sequence and whether a complete set of genes can be found on the reconstructed sequence, indicating how complete the results of different assembly pipelines are. Also, the user might need extra information, such as the coverage levels, which give more insight into how accurate the assembly is since higher coverage means one can be more confident with the assembly output in a specific region.

Once the user is able to get the optimal parameters for one assembly pipeline and analyze the assembly output quality, they may also want to compare sequences generated by different Mitogenome assembly pipelines. Moreover, since the genome assembly algorithm is often the critical step in the assembly pipelines, they may also want to solely focus on the performance of the built-in assemblers adopted by the pipelines.

4 Solution

4.1 Idiom 1

As mentioned in the related works, HiPlot (Figure 1) is an interactive visualization tool that, given experimental data points, helps us understand which parameters influence the metrics we want to optimize. However, this tool has never been used in the task of genome assembly parameter optimization. HiPlot can be used to illustrate the relationships between different pipeline parameters, such as the k-mer size, and the final assembly results (e.g., sequence length, number of scaffolds) using parallel coordinate plots. This multi-dimensional data visual will help bioinformaticians to tune up the mitogenome assembly pipeline to obtain optimal results using the best assembly parameters.

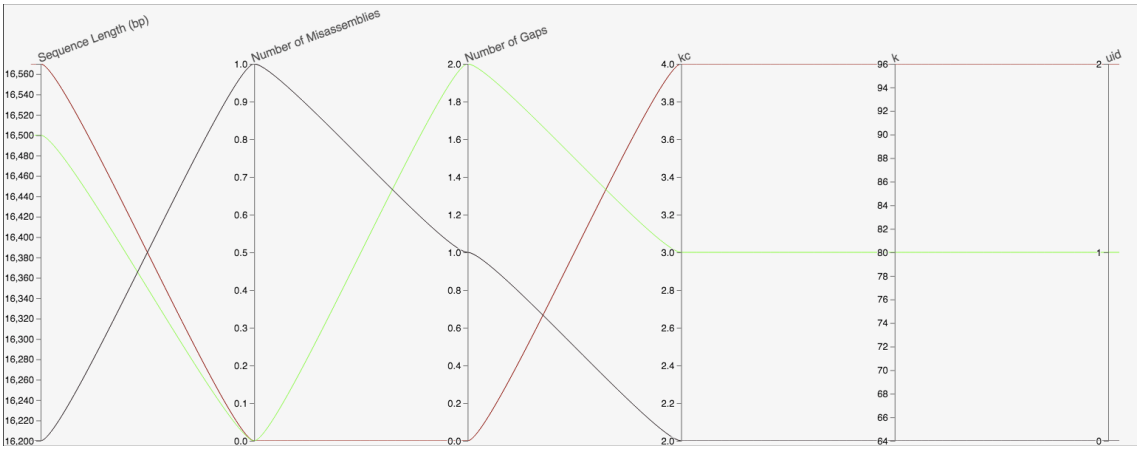

Figure 1: HiPlot [12]: Multi-dimensional Visualization of the Quality Metrics from the assemblies generated by various sets of k/kc parameters. k - k-mer size used in genome assembly; kc - cutoff for minimum k-mer coverage used in genome assembly; Sequence Length, Number of Misassemblies and Gaps are Quality Metrics that can be considered for parameter tuning

4.2 Idiom 2

Each of the three mitogenome assembly pipelines used in this project uses a different *de novo* genome assembler. The *de novo* genome assembly algorithm is often the critical step in the pipelines which ensures successful mitogenome assemblies. Icarus (Figure 2) is a visualization browser for contig-focused assembly evaluations, it can be used to display the output contigs generated by various assemblers and their corresponding assembly metrics (e.g., NG50). A sliding window can be applied to select a genomic region of interest, allowing in-depth assessment of assembly quality. Hence, here, the Icarus visualization tool will be used to help us evaluate performance of the built-in assemblers in three pipelines.

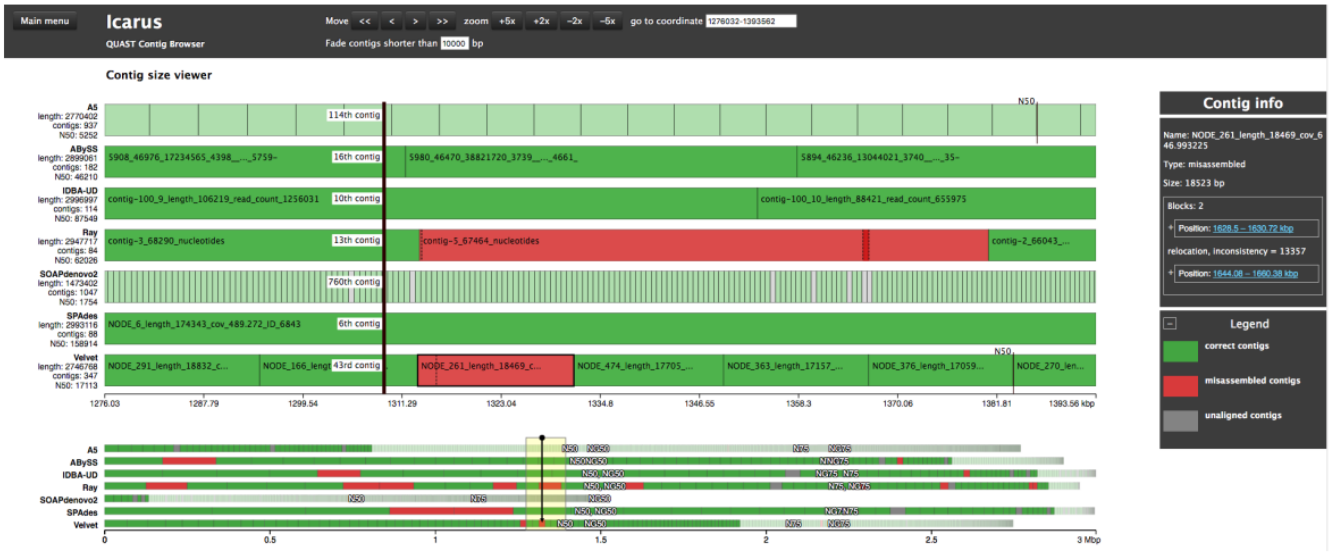

Figure 2: Icarus Contig Size Viewer [6]: A published visualizer for Assembler Software Evaluation. Contig sets (shown as rectangular blocks) generated by different assemblers (e.g., ABySS, SPAdes) are shown. The plot at the top is the zoomed-in version of the bottom plot which shows the genome contiguity metrics such as N50 and NG50. Three hues are used to differentiate contigs. Green, red, and grey blocks represent correct contigs, misassembled contigs, and unaligned contigs, respectively. This visualization facilitates the performance evaluation of the built-in assemblers in the three assembly pipelines of interest.

4.3 Idiom 3

In a complete genome assembly pipeline, usually, genome annotation is conducted on the result of the assembly step, which is a process of inferring the structure and function of the assembled sequences. Here, we propose a circular genome annotation visualization such as a Chord diagram, where the results of the different assembly pipelines are shown in a circular representation to provide a quick and easy way to spot global patterns, features, or regions of interest based on the mitogenome annotation results. More specifically, as shown in Figure 3, information such as the mitochondria gene names, strands they are located on, and their starting and ending positions are visualized at the genomic scale to aid in understanding the organization of a genome or the similarities and differences across genomes resulting from different assembly algorithms. Other information, such as the coverage of the base pairs, will also be integrated into the visualization in order to help us compare the completeness of the results. This way, these algorithms can be easily compared to each other regarding the final gene positions.

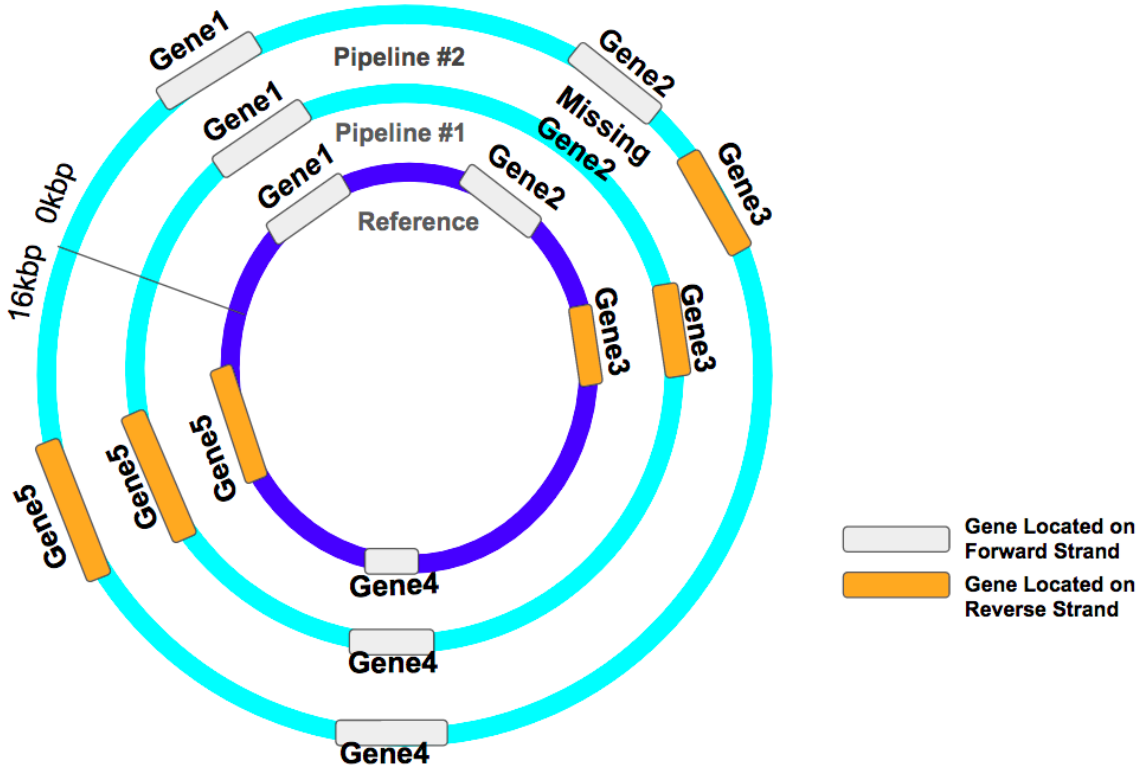

Figure 3: Multi-Layer Genome Annotation Plot: The innermost layer is the reference mitogenome and the other 2 layers are the genomes reconstructed by 2 pipelines. The size of the annotated genes and their locations relative to the reference genome are also shown. Genes found on different strands (forward, reverse) are indicated by two hues (grey, orange). By comparing the annotated genes on reconstructed genomes with the reference, missing genes or gene relocations can be observed.

4.4 Idiom 4

We propose a pipeline for generating a Circos-based genome assembly consistency plot given a set of contigs relative to reference genome, as shown in Figure 4. This visualization is intended to visualize large scale translocations or missassemblies in draft assemblies, but it can also be useful when trying to show synteny between whole genomes. Given only a reference genome fasta file and an assembly scaffolds fasta file, this analysis is good for getting a quick qualitative view of the missassemblies in a genome assembly. One possible drawback of this method is that small misassemblies, possibly medidated by repeats, might not be visible.

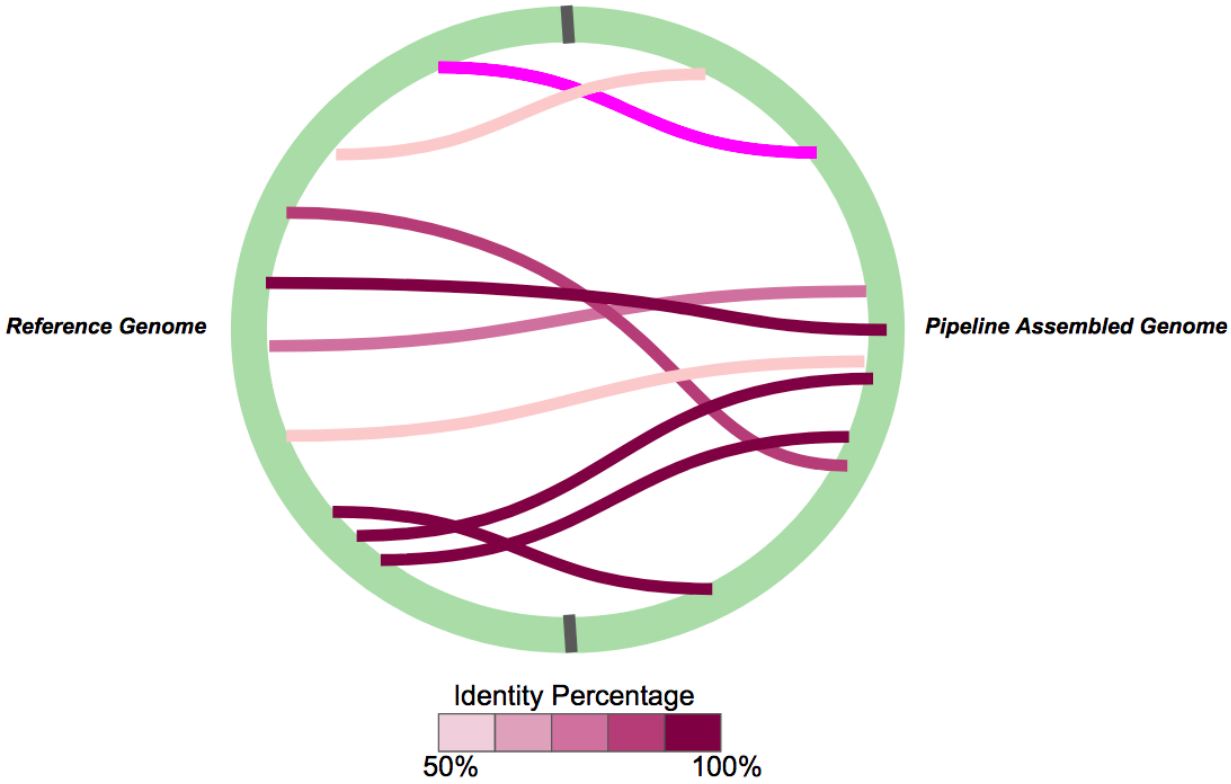

Figure 4: Circos-Based Genome Assembly Consistency Plot: Exploration of the relationships between reference genome sequence (left) and pipeline-reconstructed sequence (right). Regions with at least 50 percent similarity are highlighted by the lines within the circle. The color saturation reflects the similarity levels.

4.5 Implementation

The first phase of the project would be generating the input data: 1) Extracting species sequencing reads from possible databases 2) Generating the assemblies and 3) Parsing the resulting assembly outputs and the data annotation files. The Python programming language will be used for the above steps. As for the visualizations, we will also implement MitoAssemblyVis in Python using libraries such as Matplotlib [15] and Seaborn [16]. Other previously proposed libraries such as GGisy [17], pyCircos [18], and Mummer2Circos [19] will most probably be utilized as well. Finally, we intend to investigate using the D3.js [20] framework for generating a dashboard for user interaction and showing the final results.

4.6 Usage Scenario

Scenario 1:
Imagine a bioinformatician needs to examine the quality, and more specifically, the contiguity an assembled mitogenome resulted from a single assembly pipeline. If they also have access to the reference genome, using our tool, they can upload their assembly and reference files in the standard fasta format, and use Idiom 4, which helps them visually compare the two genomes.
Scenario 2:
Considering the above scenario, but this time, if the user needs to compare the related assembly quality metrics for different pipelines, and they have access to aligned sequencing read files in the bam format, they can upload them together with the two fasta files above and use Idiom 2, which provides the following viewer: drawn contigs ordered from longest to shortest, highlighting contiguity metrics N50, N75 (NG50, NG75) and long contigs larger than a user-specified threshold.
Scenario 3:
Another user might need to find out the optimal set of parameters used in either of the assembly pipelines that lead to the best result. Here, Idiom 1 can be utilized. After running their assembly pipelines with the hyper-parameters to be evaluated, the user will generate and upload a tabular dataset that summaries the parameters and the quality metrics. For example, considering mitogenome assemblies, most often, sequence length along with number of misassemblies or gaps are evaluated. For our novel assembly pipeline, hyper-parameters such as different k or kc values and help users to select the optimal assembly based on the above quality metrics. Depending on the assembly pipeline, the target hyper-parameters can change. For example, the GetOrganelle or mitoZ tools do not have the kc parameter; however, the user can still tune the k-mer size used in those pipelines.
Scenario 4:
Imagine a bioinformatician or clinical user has ran an assembly pipeline followed by an annotation step. Now, they have access to the gene positions and the corresponding strand orientations on the genome. If they are interested to compare the annotation results with those of other assembly pipelines, Idiom 3, which is our proposed circular multi-assembly annotation visualization can become handy. By uploading the output of the different annotation steps which need to be parsed into a csv format containing the name of the Gene, starting position, ending position, and the strand, they will be able to visually compare the extracted information for the target assembly tools of interest, or only one of them.

5 Milestones

The planned schedule and milestones for this project have been provided in the table below.

Table 3: Main Project Milestones

Deadline	Task to be completed	Assignments
September 28, 2022	Project Pitch	Armaghan and Cecilia
October 3, 2022	Preparing the Annotation Results of One Species Sample Assembly (Sea Otter)	Cecilia
October 7, 2022	Parsing the Sea Otter Annotation Results	Armaghan
October 12, 2022	Pre-Proposal Meeting	Armaghan and Cecilia
October 14, 2022	Running the MitoZ and GetOrganelle Assembly Tools	Armaghan and Cecilia
October 21, 2022	Completion of the Project Proposal	Armaghan and Cecilia
October 31, 2022	Completion of Different Assembly Data Generation and Having Installed the Needed Packages/Libraries	Armaghan and Cecilia
November 8, 2022	Having the Initial Version of Visualizations Ready	Armaghan and Cecilia
November 15, 2022	Written Update based on the Improvements and Needed Changes in the Project after Evaluation of the Progress	Armaghan and Cecilia
November 25, 2022	Peer Project Reviews	Armaghan and Cecilia
November 26, 2022	Post-Update Discussion	Armaghan and Cecilia
December 12, 2022	Finalize the Visualizations, Implementations and Analysis	Armaghan and Cecilia
December 14, 2022	Finalize the Project Presentation	Armaghan and Cecilia
December 16, 2022	Finish and Submit the Final Paper	Armaghan and Cecilia

References

[1] Mikhail Alexeyev et al. “The Maintenance of Mitochondrial DNA Integrity – Critical Analysis and Update”. In: *Cold Spring Harbour Perspectives in Biology* 5.5 (2013), a012641.

[2] David Sims et al. “Sequencing depth and coverage: key considerations in genomic analyses”. In: *nature review genetics* 15 (2014), pp. 121–132.

[3] Sehi L’Yi et al. “Gosling: A grammar-based toolkit for scalable and interactive genomics data visualization”. In: *IEEE Transactions on Visualization and Computer Graphics* 28.1 (2021), pp. 140–150.

[4] Helga Thorvaldsdóttir, James T Robinson, and Jill P Mesirov. “Integrative Genomics Viewer (IGV): high-performance genomics data visualization and exploration”. In: *Briefings in bioinformatics* 14.2 (2013), pp. 178–192.

[5] Maria Nattestad et al. “Ribbon: intuitive visualization for complex genomic variation”. In: *Bioinformatics* 37.3 (2021), pp. 413–415.

[6] Alla Mikheenko et al. “Icarus: visualizer for de novo assembly evaluation”. In: *Bioinformatics* 32.21 (2016), pp. 3321–3323.

[7] Alexey Gurevich et al. “QUAST: quality assessment tool for genome assemblies”. In: *Bioinformatics* 29.8 (2013), pp. 1072–1075.

[8] René L Warren. “Visualizing genome synteny with xmatchview”. In: *bioRxiv* (2018), p. 238220.

[9] Martin Krzywinski et al. “Circos: an information aesthetic for comparative genomics”. In: *Genome research* 19.9 (2009), pp. 1639–1645.

[10] Hongen Zhang, Paul Meltzer, and Sean Davis. “RCircos: an R package for Circos 2D track plots”. In: *BMC bioinformatics* 14.1 (2013), pp. 1–5.

[11] Tengfei Yin, Dianne Cook, and Michael Lawrence. “ggbio: an R package for extending the grammar of graphics for genomic data”. In: *Genome biology* 13.8 (2012), pp. 1–14.

[12] Facebook Research. *HiPlot: High-dimensional interactive plots made easy*. `https://ai.facebook.com/blog/hiplot-high-dimensional-interactive-plots-made-easy/`. Accessed: 2022-10-18.

[13] Jian-Jun Jin et al. “GetOrganelle: a fast and versatile toolkit for accurate de novo assembly of organelle genomes”. In: *Genome biology* 21.1 (2020), pp. 1–31.

[14] Guanliang Meng et al. “MitoZ: a toolkit for animal mitochondrial genome assembly, annotation and visualization”. In: *Nucleic Acids Research* 47.11 (2019), e63.

[15] John D Hunter. “Matplotlib: A 2D graphics environment”. In: *Computing in science & engineering* 9.03 (2007), pp. 90–95.

[16] Michael L Waskom. “Seaborn: statistical data visualization”. In: *Journal of Open Source Software* 6.60 (2021), p. 3021.

[17] Sandro Valenzuela. *GGisy*. `https://github.com/Sanrrone/GGisy`. 2017.

[18] Daniel Bullock Hideto Mori Max Base. *pyCircos*. `https://github.com/ponnhide/pyCircos`. 2022.

[19] Oliver Schwengers Trestan Pillonel. *Mummer2Circos*. `https://github.com/metagenlab/mummer2circos`. 2022.

[20] Michael Bostock, Vadim Ogievetsky, and Jeffrey Heer. “D^3 data-driven documents”. In: *IEEE transactions on visualization and computer graphics* 17.12 (2011), pp. 2301–2309.