INTEGRATING METAGENOMICS AND COMPUTATIONAL BIOLOGY TO UNCOVER MURBURN MECHANISMS

OUR TEAM

- Hemesh Yeturu (CB.SC.U4AIE23166)
- Joel John (CB.SC.U4AIE23131)
- Abhishek S (CB.SC.U4AIE23107)
- Adarsh Pradeep (CB.SC.U4AIE23109)

ABSTRACT

This project integrates metagenomics and computational biology to investigate murburn mechanisms, with a focus on reactive oxygen species (ROS) in microbial communities. We aim to identify ROS-related genes in bacterial genomes by leveraging advanced bioinformatics tools such as BLAST, sequence alignment, and homology detection. Further, we employ graph-based algorithms, network flow models, and random walk simulations to analyze ROS pathways and simulate their interactions within microbial ecosystems. By combining biological insights with efficient computational techniques, we enable pathway analysis, diffusion modeling, and visualization of ROS-driven processes. This interdisciplinary approach provides new perspectives on microbial metabolism, ecological dynamics, and potential applications in biotechnology and medicine.

INTRODUCTION

Aspect	Metagenomics	Murburn Concept
Definition	Study of genetic material from entire microbial communities.	Focuses on reactive oxygen species (ROS) and their role in cellular respiration and oxidative stress.
Primary Focus	Microbial functions, diversity, and interactions.	ROS dynamics in metabolism and cellular processes.
Key Objectives	- Identify microbial genes. - Analyze metabolic pathways. - Simulate microbial interactions.	- Identify ROS-related genes. - Study oxidative stress mechanisms. - Explore energy metabolism.
Computational Role	Uses advanced bioinformatics tools for genetic analysis.	Employs modeling and simulations for ROS behavior in cells.
Interdisciplinary Approach	Combines biology, computational analysis, and environmental studies.	Integrates chemistry, biophysics, and computational biology.
Significance	Helps understand microbial ecology, evolution, and disease links.	Offers new perspectives on metabolic regulation and oxidative stress.

OBJECTIVE

INTEGRATING BIOLOGY AND ALGORITHM

1. Meta Genomic Data Processing of Bacterial Genes (Using De Bruijn graph)
2. Cross-species analysis of ROS genes (human vs bacteria)
3. Network Flow Algorithms (for Pathway Analysis)
4. Random Walk Model (for ROS Diffusion Simulation)

METHODOLOGY

1. Data Collection:

- Obtain metagenomic datasets from public repositories like NCBI , SRA and MG-RAST.

2. Data Preprocessing:

- Data retrieval from NCBI using entrez
- Quality filtering (Extracting only valid Nucleotide sequences)

3. K-mer extraction --> Graph Construction (De Bruijn graph)

- The function extract_limited_kmers() extracts short overlapping substrings (k-mers) of size k (default k=10).

The function build_de_bruijn_graph() constructs a directed graph where:

- Nodes represent (k-1)-mers (prefix & suffix of k-mers).
- Edges represent k-mer transitions between nodes.

METHODOLOGY

4. Gene Annotation and Pathway Analysis:

- Bacterial genes similar to human ROS genes, helping in comparative genomics.(**Using BLASTp**)
- This comparative approach links bacterial ROS defence mechanisms to human systems, aiding in evolutionary and biomedical insights.

5. Network flow algorithm:

- We use **Edmonds-Karp** algorithm to determine maximum flow between human ROS genes and bacterial homologs.
- The flow capacity represents gene similarity strength, allowing us to quantify gene conservation & functional transfer.

6. Random Walk model:

- Steady-state probabilities (genes most affected by ROS diffusion).
-Mean first passage time (MFPT) (how quickly ROS reaches different bacterial genes).

7. Integration:

- Combine metagenomic analysis with computational modeling to explore murburn mechanisms and their ecological significance.

What do you find in Metagenomic Data processing ?

- **Metagenomic Data Processing:** Focuses on efficient assembly, classification, and analysis of bacterial genomes from environmental samples.
- **Sequence Assembly**: Reconstructs original bacterial genomes from short DNA sequences (reads) obtained through sequencing technologies.
- **Classification & Analysis**: Helps in understanding microbial diversity, functions, and interactions within ecosystems.

Why De Bruijn Graphs?

De Bruijn Graphs (**DBGs**) are preferred for assembling short-read sequencing data because:

- **Reduced Computational Complexity**: Traditional methods like overlap graphs require comparing all reads, which is computationally expensive. DBGs convert reads into fixed-length k-mers, simplifying the process.
- **Error Correction**: DBGs detect and remove low-frequency k-mers, improving accuracy.
- **Handling Repeats**: Many bacterial genomes contain repetitive sequences, and DBGs efficiently resolve ambiguities in such cases.
- **Efficient Representation**: The adjacency-based structure of DBGs avoids explicit pairwise read comparisons, making them suitable for large metagenomic datasets.

What Do De Bruijn Graphs Do?

Step 1: K-mer Splitting

- The input sequencing reads are broken into overlapping k-mers (substrings of length k).
- These k-mers serve as the building blocks for graph construction.

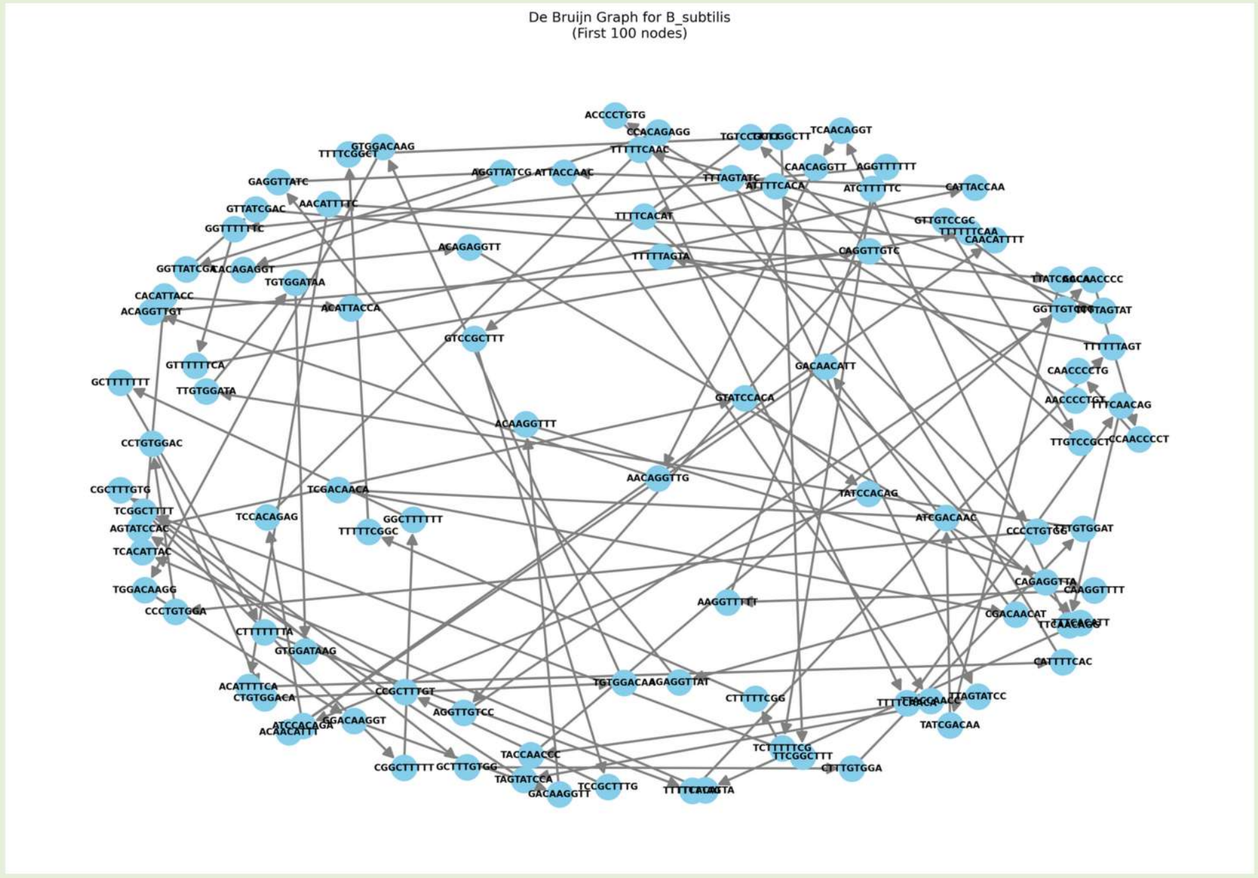
Step 2: Graph Construction

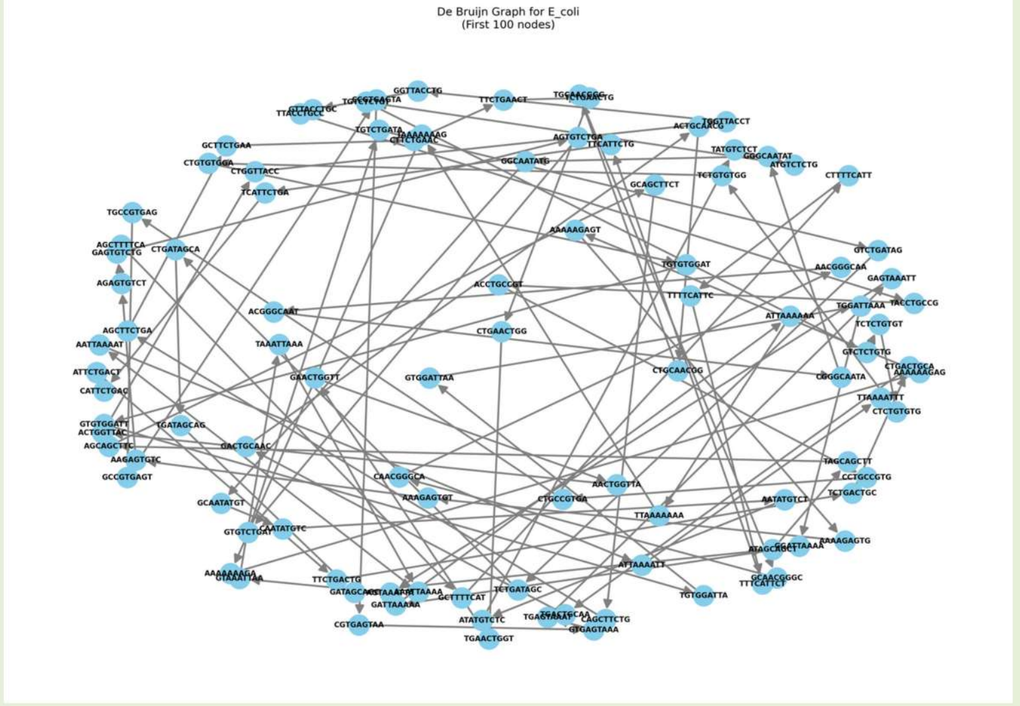
- Each node in the graph represents a (k-1)-mer, which is a k-mer excluding its last base.
- Directed edges connect consecutive k-mers, linking nodes where the suffix of one k-mer matches the prefix of another.

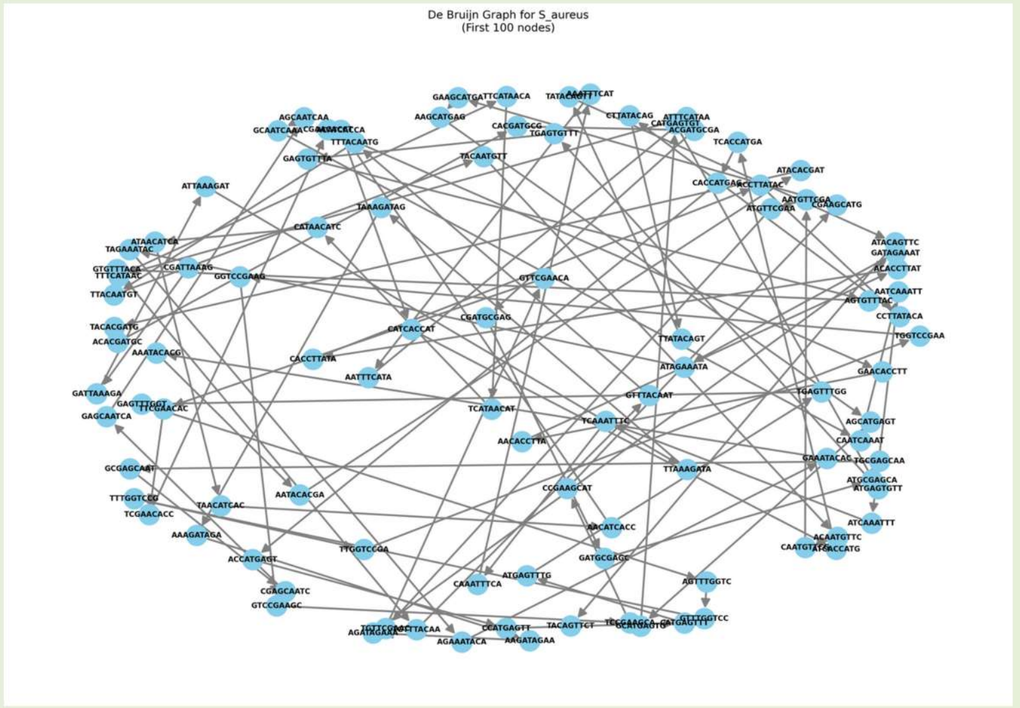
Step 3: Path Traversal (Genome Assembly)

- The assembled genome is reconstructed by finding an Eulerian path, a path that visits every edge exactly once.
- This approach efficiently reconstructs the original sequence while handling sequencing errors and repeated regions.

Results for De Bruijn Graph







Cross-Species Analysis of ROS Genes (Human vs. Bacteria)

Comparing ROS(Reactive Oxygen Species) Defence Mechanisms

- Humans and bacteria both produce ROS-scavenging enzymes (e.g., catalase, superoxide dismutase), but their mechanisms differ.
- This analysis reveals how bacterial ROS defenses help pathogens survive in oxidative environments (e.g., immune system attack).

Identifying Evolutionarily Conserved ROS Genes

- Some ROS-related genes are highly conserved across species.
- Example: Superoxide dismutase (SOD) is found in both humans and bacteria, showing its fundamental role in oxidative stress protection.

We use BLASTp . What is BLASTp ?

Basic Local Alignment Search Tool for Proteins ---> BLASTp

Finds Similar Proteins → Compares a query protein sequence to a protein database (e.g., NCBI, UniProt).

Identifies Evolutionary Relationships → Detects homologous proteins (orthologs, paralogs).

Predicts Function of Unknown Proteins → If a query protein is similar to a known protein, it likely has a similar function.

What does it do ?

Input Query Protein Sequence

Breaks Query into Small Words (k-mers) → Default k=3 (3 amino acids at a time).

Searches for Similar Segments in the Database.

Extends Matches to find high-scoring alignments.

Computes Statistical Significance → Assigns E-value, % Identity, and Bit Score.

Results found in BLASTp

```
🔍 Running BLAST for SOD1 against Bacillus subtilis...
Running BLASTp for NP_000454.1 against Bacillus subtilis proteome... (This may take time)
No significant homologs found in Bacillus subtilis.

🔍 Running BLAST for SOD1 against Staphylococcus aureus...
Running BLASTp for NP_000454.1 against Staphylococcus aureus proteome... (This may take time)
No significant homologs found in Staphylococcus aureus.

Processing GPX1 (NP_000572.2)...
Fetching sequence NP_000572.2 from NCBI...

🔍 Running BLAST for GPX1 against Escherichia coli...
Running BLASTp for NP_000572.2 against Escherichia coli proteome... (This may take time)
🧬 Homologs found in Escherichia coli:
   - bifunctional thioredoxin/glutathione peroxidase [Escherichia coli] (E-value: 4.29627e-29, Identity: 69/157)
   - bifunctional thioredoxin/glutathione peroxidase [Escherichia coli] (E-value: 5.2136e-29, Identity: 71/158)
   - bifunctional thioredoxin/glutathione peroxidase [Escherichia coli] (E-value: 5.68185e-29, Identity: 69/157)
   - bifunctional thioredoxin/glutathione peroxidase [Escherichia coli] (E-value: 6.96937e-29, Identity: 70/158)
   - bifunctional thioredoxin/glutathione peroxidase [Escherichia coli] (E-value: 9.93661e-29, Identity: 68/157)

🔍 Running BLAST for GPX1 against Bacillus subtilis...
Running BLASTp for NP_000572.2 against Bacillus subtilis proteome... (This may take time)
🧬 Homologs found in Bacillus subtilis:
   - glutathione peroxidase [Bacillus subtilis] (E-value: 2.75208e-23, Identity: 65/184)
   - glutathione peroxidase [Bacillus subtilis] (E-value: 2.81152e-23, Identity: 64/184)
   - glutathione peroxidase [Bacillus subtilis] (E-value: 6.39841e-23, Identity: 65/184)
   - glutathione peroxidase [Bacillus subtilis] (E-value: 1.10279e-22, Identity: 64/184)
   - MULTISPECIES: glutathione peroxidase [Bacillaceae] >ref|NP_390073.1| bacillithiol peroxidase [Bacillus subtilis subsp. subtilis str. 168] (E-value: 1.6542e-22, Identity: 64/184)

🔍 Running BLAST for GPX1 against Staphylococcus aureus...
Running BLASTp for NP_000572.2 against Staphylococcus aureus proteome... (This may take time)
/usr/local/lib/python3.11/dist-packages/Bio/Blast/NCBIWWW.py:275: BiopythonWarning: BLAST request WJES1C7A013 is taking longer than 10 minutes, consider re-issuing it
  warnings.warn(
🧬 Homologs found in Staphylococcus aureus:
   - glutathione peroxidase [Staphylococcus aureus] (E-value: 2.08957e-21, Identity: 59/170)
   - glutathione peroxidase [Staphylococcus aureus] (E-value: 1.26271e-19, Identity: 56/183)
   - glutathione peroxidase [Staphylococcus aureus] (E-value: 5.76241e-19, Identity: 55/183)
   - glutathione peroxidase [Staphylococcus aureus] (E-value: 6.33966e-19, Identity: 55/183)
   - glutathione peroxidase [Staphylococcus aureus] (E-value: 7.59226e-19, Identity: 55/183)
```

Network Flow Algorithms (for Pathway Analysis)

Network Flow Algorithms are used in computational biology, particularly for pathway analysis, to model and analyze the flow of biological entities (such as molecules, proteins, or metabolites)

These algorithms help in understanding how substances move and interact within cellular networks, such as metabolic, signaling, or genetic pathways.

What does it do ?

Network flow algorithms work by modeling biological systems as graphs, where:

Nodes represent biological entities (like genes, proteins, metabolites, or enzymes).

Edges represent the interactions or relationships between those entities (like biochemical reactions, protein-protein interactions, etc.).

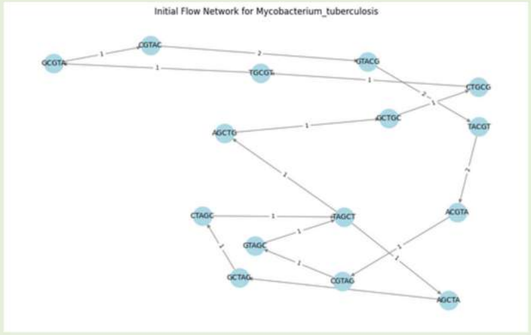
The goal is to analyze how "flow" (e.g., energy, metabolites, signals) travels through the network from one point to another, often to optimize or find bottlenecks, understand dynamics, or identify critical components within a pathway.

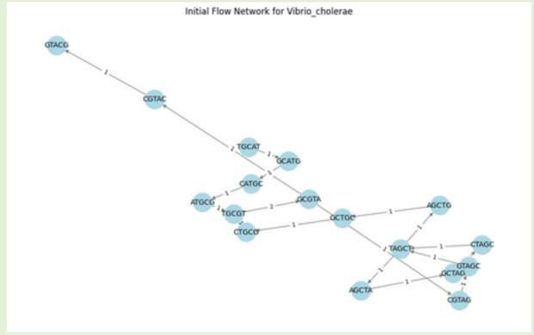
Max-Flow Algorithms (Edmonds-Karp)

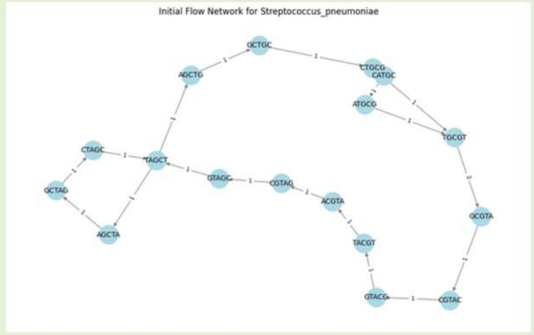
Edmonds-Karp finds the maximum flow in a flow network. This is the largest amount of flow that can be sent from a source node to a sink node in the network while respecting the capacities of the edges

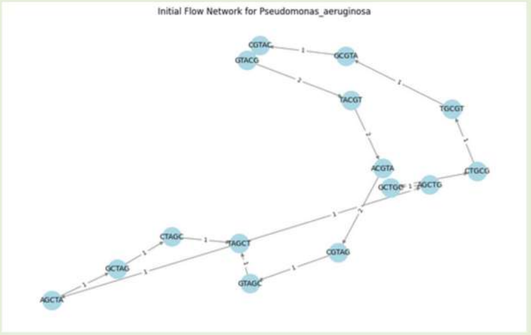
- A network flow is a directed graph where each edge has a capacity (the maximum amount of flow it can carry).
- The flow must satisfy the constraints:
- The amount of flow entering a node (except for the source and sink) must equal the amount of flow leaving the node (this is the conservation of flow).
- The flow through an edge cannot exceed its capacity.

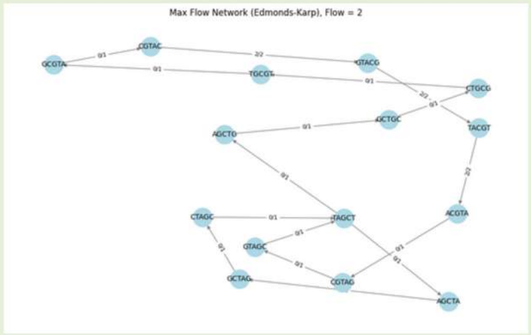
Results from Max flow algorithm

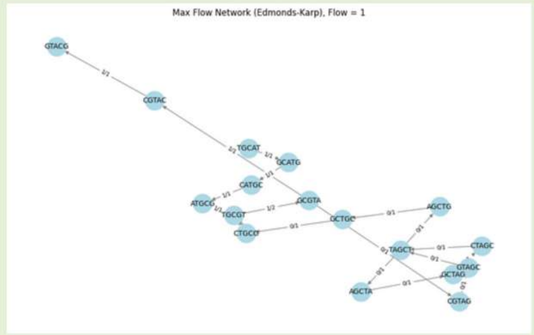


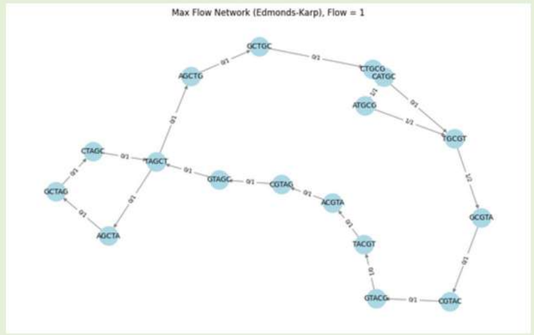


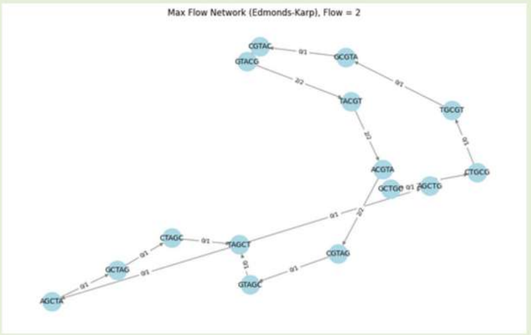












Concluding results on this

```
=== Maximum Flow Results ===
Salmonella_enterica: Max Flow = 1
Mycobacterium_tuberculosis: Max Flow = 2
Vibrio_cholerae: Max Flow = 1
Pseudomonas_aeruginosa: Max Flow = 2
Streptococcus_pneumoniae: Max Flow = 1
```

1. **Salmonella_enterica:**
   - **Max Flow = 1**
   - **Indicates that the flow network allows only one unit of flow from source to sink.**
2. **Mycobacterium_tuberculosis:**
   - **Max Flow = 2**
   - **Suggests that two independent paths or an increased capacity in the network enables a higher flow.**
3. **Vibrio_cholerae:**
   - **Max Flow = 1**
   - **Shows that the sequence structure supports one effective flow path.**
4. **Pseudomonas_aeruginosa:**
   - **Max Flow = 2**
   - **Similar to Mycobacterium_tuberculosis, the network supports two units of flow, indicating a well-connected structure.**
5. **Streptococcus_pneumoniae:**
   - **Max Flow = 1**
   - **Demonstrates that one unit of flow reaches the sink, with a structure supporting a single path.**

Random Walk Model (for ROS Diffusion Simulation)

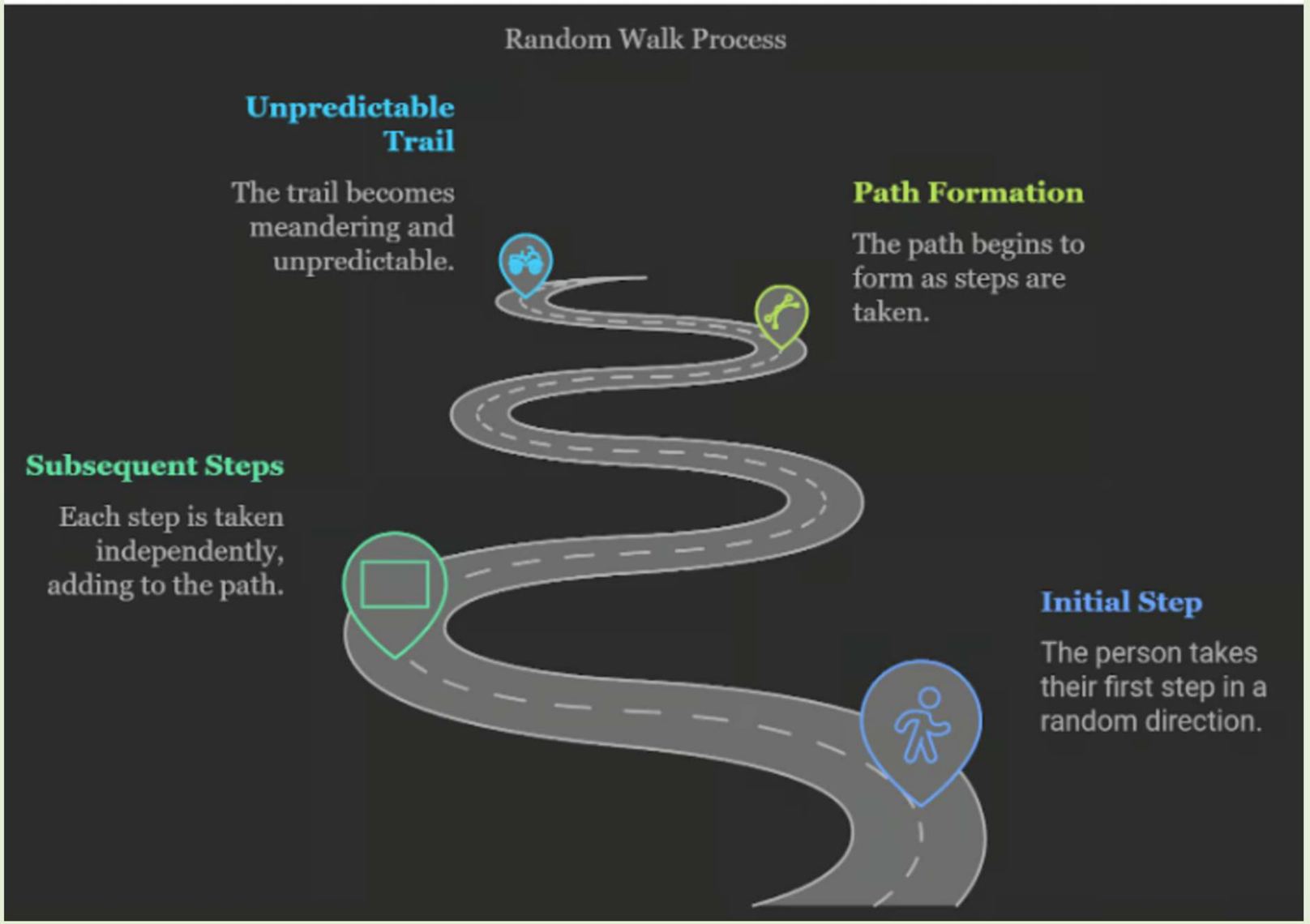
Random Walk Model for ROS (Reactive Oxygen Species) diffusion is used to simulate how ROS molecules, generated either internally (by metabolic processes) or externally (due to environmental stress or immune responses), diffuse through bacterial cells or colonies

Understanding how ROS spreads inside bacterial cells or through bacterial communities (e.g., biofilms) is important for studying the bacterial response to oxidative stress and for designing antimicrobial strategies.

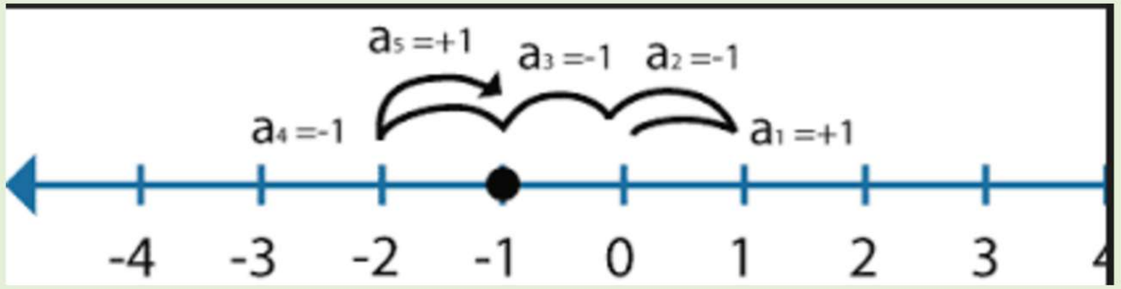
Use if you have a complex system where you want to estimate the behavior of the system using random sampling to account for both random movement and complex interactions (like ROS-induced damage in bacteria or complex biochemical processes).

Random Walk Model (for ROS Diffusion Simulation)

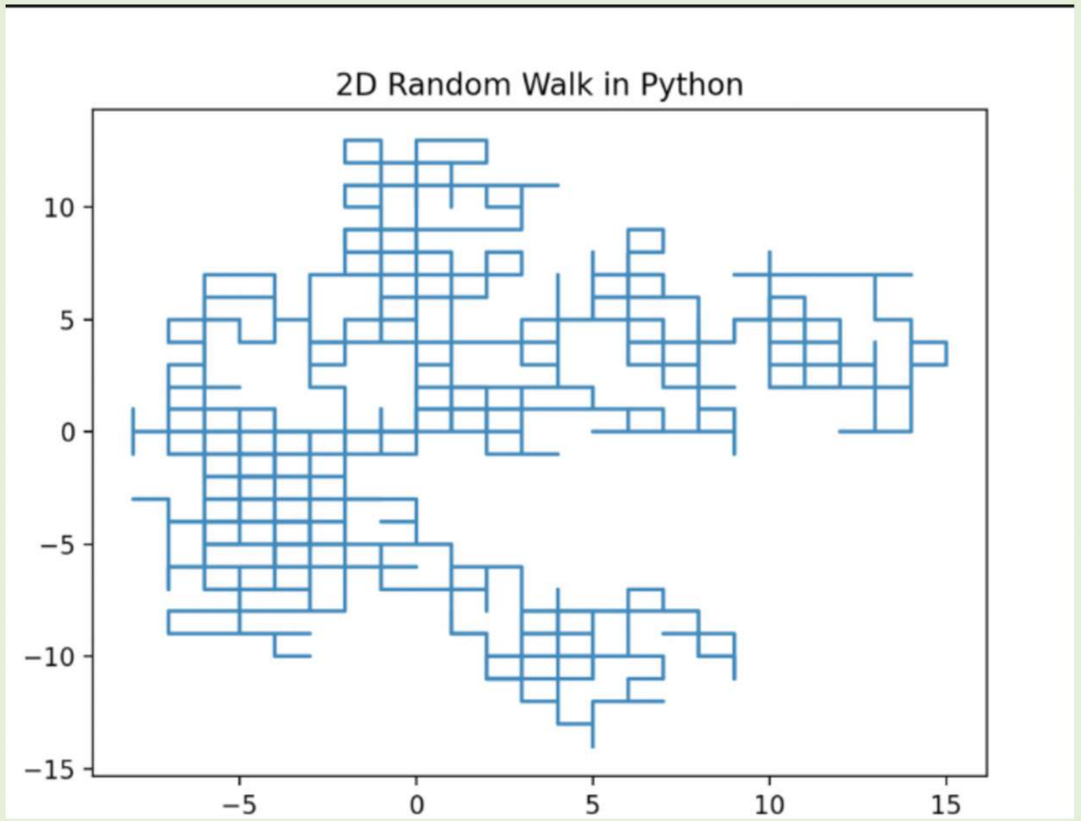
- Initialize the starting position (e.g., starting at the center of a 2D grid).
- Choose a random direction for the first step (e.g., up, down, left, or right).
- Move in that direction by updating the current position.
- Repeat the random movement process for a set number of steps or until a termination condition is met (e.g., after 100 steps or when the ROS reaches the cell boundary).
- Optionally, track the positions over time or collect statistical data about the walk (e.g., average distance traveled).


Random Walk Process
Unpredictable Trail
The trail becomes meandering and unpredictable.
Path Formation
The path begins to form as steps are taken.
Subsequent Steps
Each step is taken independently, adding to the path.
Initial Step
The person takes their first step in a random direction.

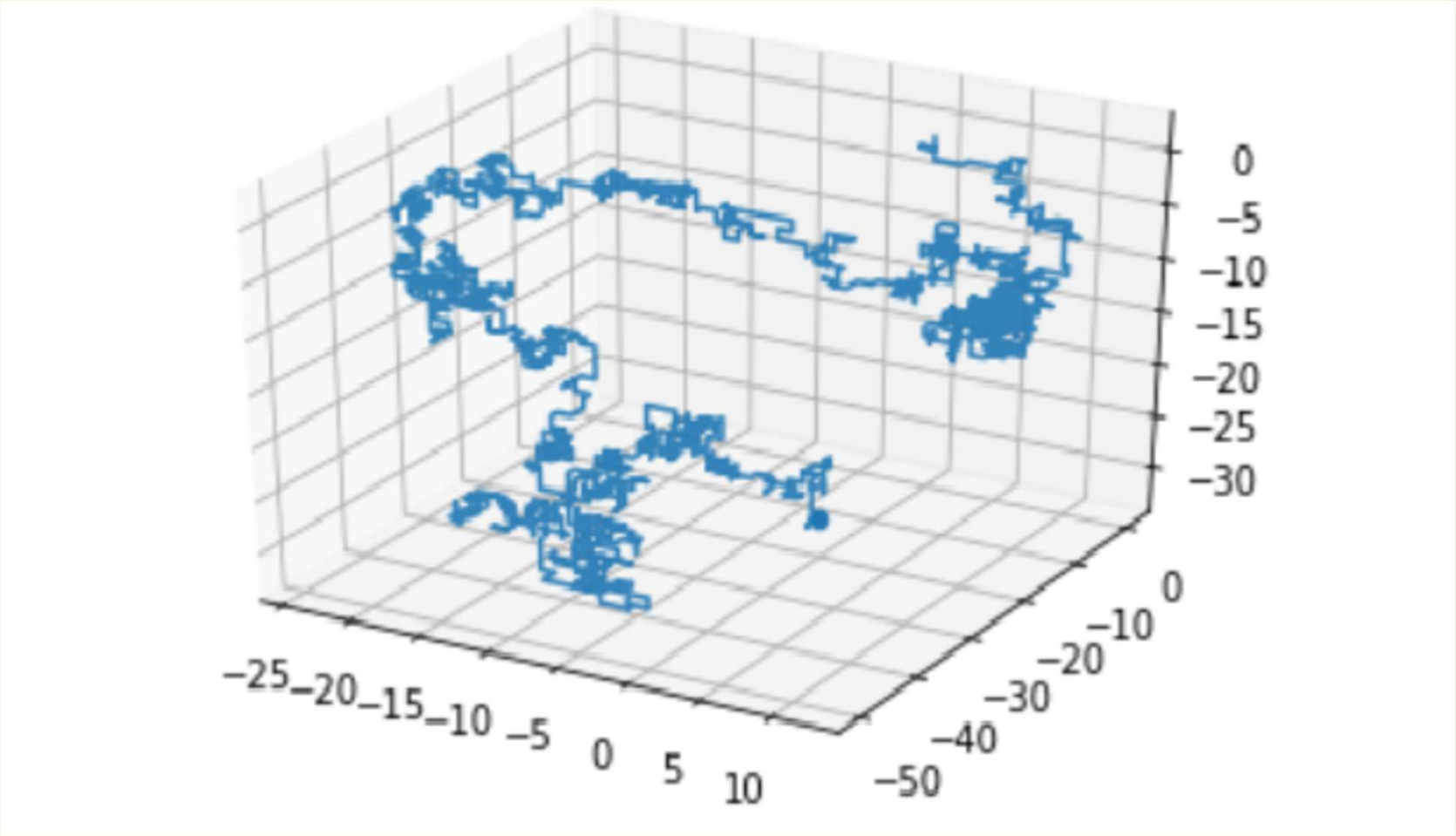
1-D random walk



2-D random walk



3-D random walk



pseudocode

Python

```
function random_walk(start_position, num_steps):
    current_position = start_position
    for i in range(num_steps):
        # Generate a random direction (e.g., -1 or 1 for a 1D walk)
        direction = random.choice([-1, 1])

        # Update current position based on the random direction
        current_position += direction

    return current_position
```

Note:

For 1D random walk and 2D random walk there is a high probability of returning to the starting point whereas in case of 3D or higher dimensions the probability decreases.

Relevance of Random Walk in Our Project

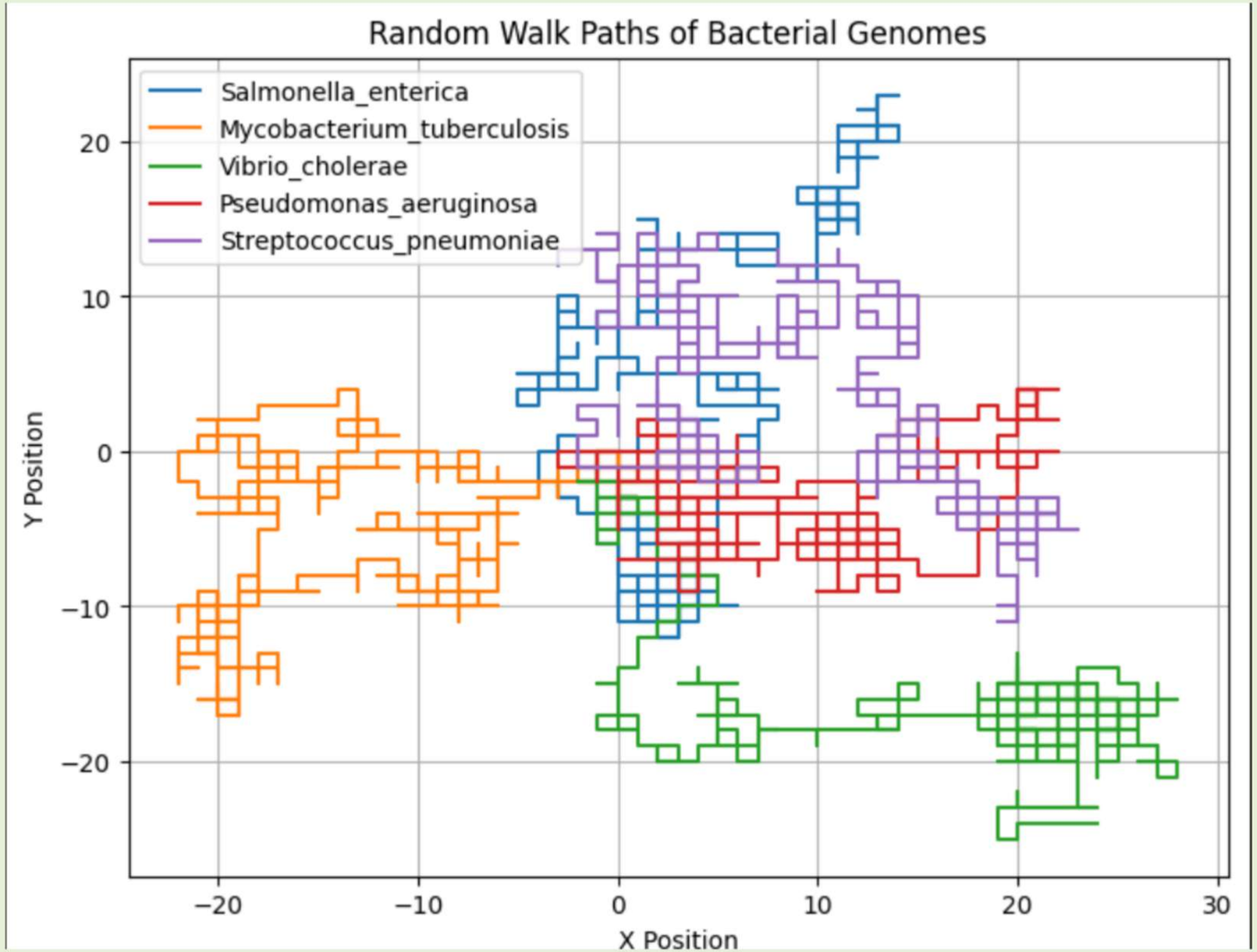
Unpredictable Movement

- **ROS molecules do not follow a fixed path due to random collisions with cellular structures and other molecules.**
- **Their motion resembles Brownian motion, where particles move unpredictably due to kinetic energy.**

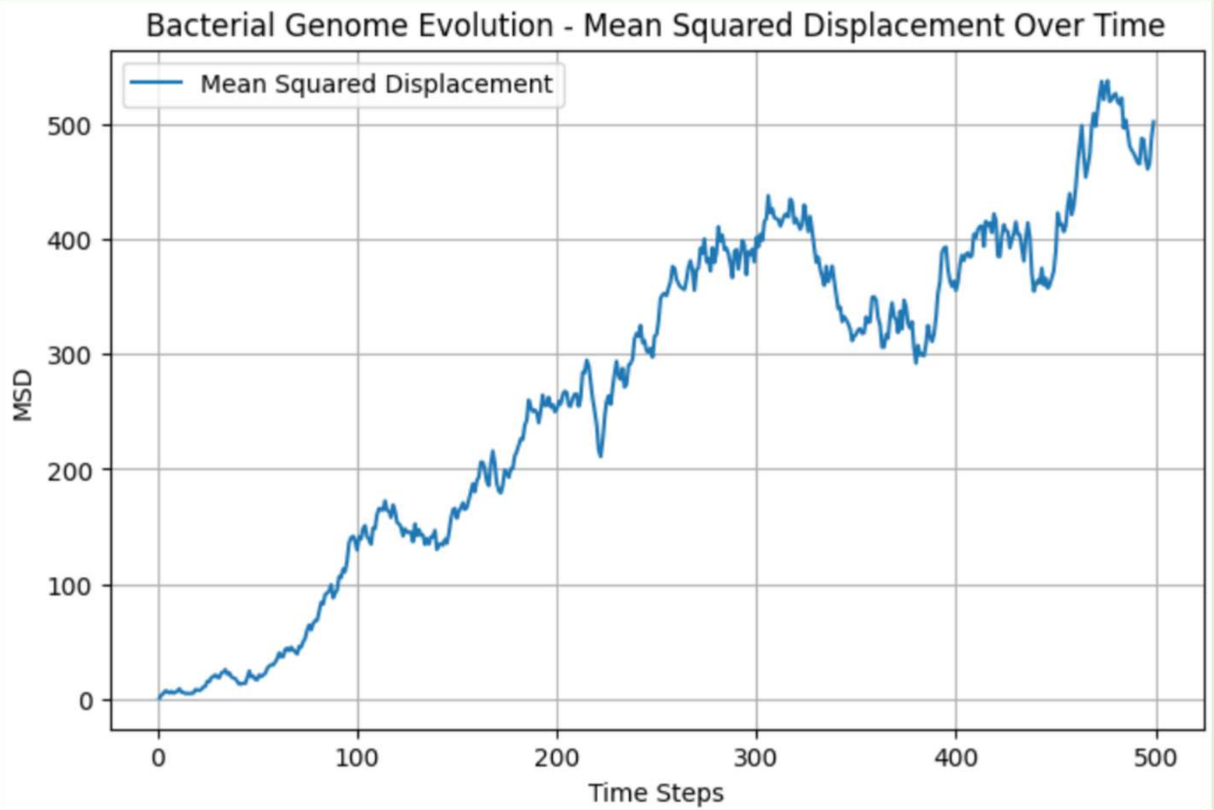
Models Diffusion Accurately

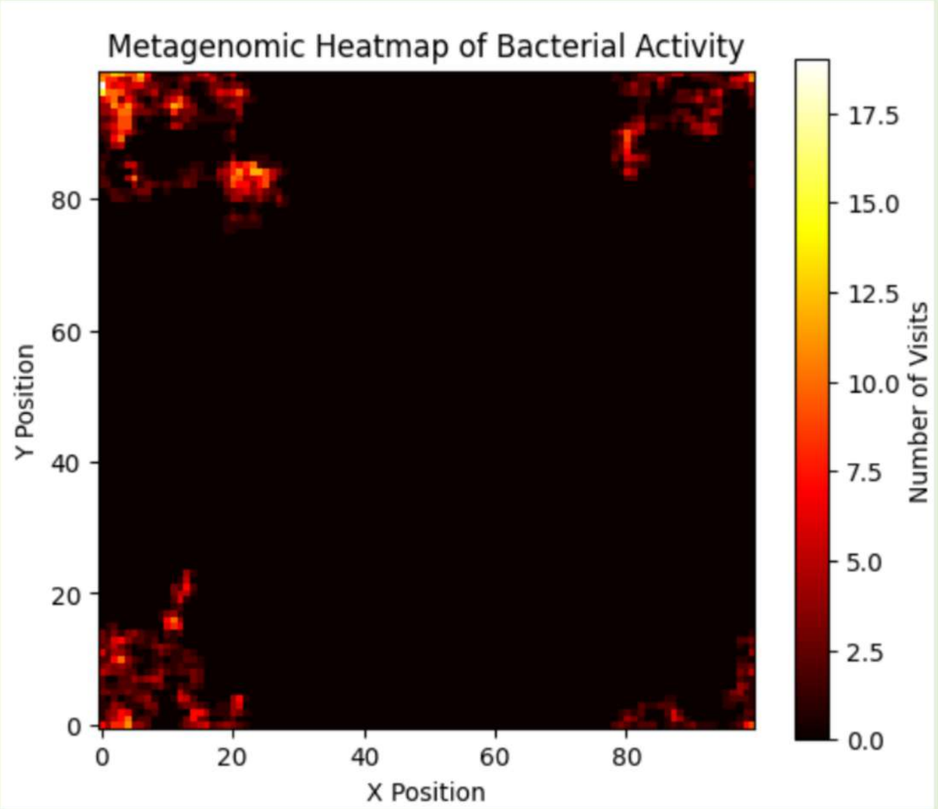
- **Diffusion governs how far and fast ROS spread within bacterial cells or biofilms.**
- **A random walk algorithm simulates this natural diffusion, helping predict ROS concentration gradients in different regions.**

Results


Random Walk Paths of Bacterial Genomes
Salmonella_enterica
Mycobacterium_tuberculosis
Vibrio_cholerae
Pseudomonas_aeruginosa
Streptococcus_pneumoniae
Y Position
X Position
20
10
0
−10
−20
−20
−10
0
10
20
30

Results


Bacterial Genome Evolution - Mean Squared Displacement Over Time
Mean Squared Displacement
MSD
Time Steps
0
100
200
300
400
500


Metagenomic Heatmap of Bacterial Activity
Y Position
X Position
Number of Visits
0
20
40
60
80
0.0
2.5
5.0
7.5
10.0
12.5
15.0
17.5

CONCLUSION & RESULTS EXPECTED

This project integrates metagenomics and computational biology to study reactive oxygen species (ROS) in microbial ecosystems. By combining gene identification, pathway analysis, and modeling, it provides insights into microbial functions and ecological roles.

Expected Results:

1. Identification of ROS-related genes.
2. Analysis of ROS pathways in microbial communities.
3. Simulations of ROS interactions.
4. Visualizations like heatmaps and networks.
5. Optimized algorithms for metagenomic analysis.

LITERATURE REVIEW

SR.NO	DESCRIPTION	REFERENCE
1	This study discusses the application of De Bruijn graphs in microbiome research, highlighting their role in assembling high-throughput sequencing data and their potential in comparative genomics.	Dufault-Thompson, K., & Jiang, X. (2022). Applications of de Bruijn graphs in microbiome research. iMeta, 1(1)
2	This paper revisits the E-value calculations in BLASTp, aiming to improve the accuracy of protein sequence alignments and the reliability of homology predictions.	Lu, Y. Y., Noble, W. S., & Keich, U. (2024). A BLAST from the past: revisiting blastp's E-value. Bioinformatics, 40(12), btae729.
3	This study provides an overview of graph models used in DNA sequencing by hybridization, focusing on the application of De Bruijn graphs in genome assembly. It discusses the theoretical framework and compares outcomes using SPAdes and Velvet assemblers.	Sarkar, B. (2024). A Study of Computational Genome Assembly by Graph Theory. Annals of West University of Timisoara: Mathematics and Computer Science, 60(1), 1–24.

LITERATURE REVIEW

SR.NO	DESCRIPTION	REFERENCE
4	This preprint serves as a beginner's guide to microbiome modeling, offering an interdisciplinary overview that starts with fundamental knowledge of microbiomes, metagenomics methods, and modeling approaches.	Lange, E., Kranert, L., Krüger, J., Benndorf, D., & Heyer, R. (2024). Microbiome Modeling: A Beginner’s Guide. Preprints, 2024010789
5	This dissertation explores advanced computational methods for analyzing metagenomic data, focusing on the application of De Bruijn graphs in genome assembly and the use of BLASTp for protein sequence alignment.	Smith, J. A. (2023). Advanced Computational Techniques in Metagenomics

THANK YOU