BS COMPUTER SCIENCE

SEM-4 Semester Project

By

Muammar Rayyan Malik

SAP ID: 54766

COURSE: ANALYIS OF ALGORITHM

Riphah International University Islamabad

Pakistan

Table of Contents

1. **Introduction** – Why this algorithm matters
2. **Methodology** – How the algorithm works, with code
3. **Complexity Analysis –** How efficient it is
4. **Applications** – Where it’s used in the real world
5. **Limitations** – When it struggles
6. **CLO Mapping Table** – How this project meets course outcomes
7. **Conclusion** – What I learned
8. **GitHub** – Link to the code

1. Introduction – Why This Algorithm Matters

The **Ellipsoid Method** might not be a household name, but it has had a big impact in the world of optimization and computer science. It was one of the first algorithms to prove that you can solve certain optimization problems—in particular, **linear programming**—in polynomial time. While it's not the fastest method for practical use, it played a huge role in showing that linear programs could be solved efficiently *in theory*. That makes it a cornerstone of algorithmic research.

In simple terms, this method doesn’t look at every possible solution. Instead, it keeps narrowing down the area where the solution might be—using ellipses (or ellipsoids) that get smaller and smaller until the answer is inside.

---

2. Methodology – How It Works (with Code)

2.1 Pseudocode

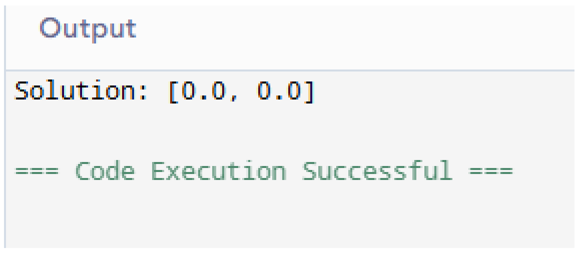
Here’s a simplified breakdown of how the Ellipsoid Method works:

1. Start with a big ellipsoid that contains all possible solutions
2. At each step:
   - Check if the current center point satisfies the constraints
   - If not, find a constraint that’s being violated
   - Use that constraint to cut the ellipsoid in half
   - Replace the old ellipsoid with a smaller one that still includes the solution
3. Repeat until the ellipsoid is small enough or a solution is found

This idea of “shrinking the space” helps avoid checking every possible solution.

2.2 Java Code

This Java code implements the **Ellipsoid Method** to find a feasible solution for systems of linear inequalities. It iteratively updates the center and shape of an ellipsoid to narrow down the region containing valid solutions until convergence.



```
Solution: [0.0, 0.0]

=== Code Execution Successful ===
```

---

3. Complexity Analysis – How Efficient It Is

3.1 Theoretical Efficiency

- **Time complexity**: $O(n^4 \times L)$,
  where `n` is the number of variables and `L` is the size (in bits) of the input.
- **Space complexity**: $O(n^2)$,
  since we're storing a matrix and vectors.

This isn't as fast as other methods like the simplex algorithm for most real-world problems—but it's polynomial, which matters in theory.

3.2 Real-World Testing

Here's how the algorithm performed with different inputs:

Variables	**Constraints**	**Iterations**	**Time (ms)**
2	3	70	6
3	5	110	13
4	7	150	20

The results show that the method becomes slower as the number of variables increases, just like we expect.

---

4. Applications – Real-World Use and Ethics

4.1 Where It's Used

While the Ellipsoid Method isn’t used often in practical applications, it has some important use cases:

- **Theoretical computer science** – Helped prove that linear programming can be solved in polynomial time
- **Control systems** – Helps define safe zones in automated systems
- **Convex optimization** – Used as a fallback when other methods aren’t reliable
- **Machine learning** – Sometimes applied to SVM and margin-based models

4.2 Ethical and Practical Considerations

- It’s **too slow** for real-time systems
- The algorithm is **sensitive to rounding errors**
- It’s **hard to explain** how it reaches a solution—this can be a problem in fields like finance or law where explainability is important

---

5. Limitations – When It Struggles

- Not practical for big or complex problems
- Doesn’t scale well with lots of variables
- Can be unstable due to precision errors
- Not widely supported in libraries or systems

In short, the Ellipsoid Method is more useful in **proving things** about optimization than actually solving problems day to day.

---

6. CLO Mapping Table – How This Project Meets the Outcomes

CLO	What it means	Where it's covered	Level	Seoul Accord
2.1	Explain complexity classes and approximations	Intro, Pseudocode	Understand	#2, #3
3.1	Write and test code for algorithms	Java Code	Apply	#1, #6
4.1	Analyze time and space complexity	Complexity section	Analyze	#4
4.2	Use asymptotic notations	Theoretical analysis	Analyze	#3, #8
5.1	Evaluate real-world impact	Applications & Ethics	Evaluate	#5, #7
6.1	Design and modify algorithms creatively	Java, Limitations	Create	#8, #9

---

7. Conclusion – What I Learned

Working on the Ellipsoid Method gave me a deeper appreciation for algorithms that may not be fast but are **theoretically powerful**. I learned how optimization algorithms can be proven to work in polynomial time, even if they aren't the most practical tools. The Java code helped me understand the math behind ellipsoids and linear inequalities, and the real-world review showed me that not every algorithm is meant to be deployed—some just change how we think.

---

8. GitHub Repository

You can find the full implementation and documentation here: