# NUMERIC SOLUTION OF SPARSE LINEAR SYSTEMS
# WITH KACHMARZ AND CIMMINO METHODS FOR 2D and 3D Matrixes

**Project Goal:** The project aims to analyze various numerical methods for solving linear systems of equations and understand their performance in terms of solution accuracy, computational efficiency, and memory usage.

**Used Information:** "A NOVEL PARTITIONING METHOD FOR ACCELERATING THE BLOCK CIMMINO ALGORITHM" F. SUKRU TORUN , MURAT MANGUOGLU‡ , AND CEVDET AYKANAT

**Project Describing: The project involves three main parts:**

1. **Methods Time and Residuals:**
    - **Various numerical methods are applied to solve linear systems, and their computational times and residual errors are recorded.**
    - **Issues such as SparseEfficiencyWarning are addressed, and the impact of matrix format on computation efficiency is discussed.**
    - **Differences between methods like LU decomposition, Bicgstab, and preconditioned Bicgstab are highlighted, emphasizing their respective advantages and drawbacks.**
2. **Cimmino and Kaczmarz Methods:**
    - **Performance of Cimmino and Kaczmarz methods are evaluated in both 2D and 3D matrices.**
    - **Comparisons are drawn with the Least Square method regarding residuals, computational times, and memory usage.**
    - **Estimations for different values of K are analyzed to assess the accuracy of the methods in predicting solution times.**
3. **Steps 6-9 with Cage 10 Matrix (Cimmino with Cage 10):**
    - **Cimmino method is applied to solve the Cage 10 matrix, and the residuals and times for different values of K are recorded.**
    - **Comparisons are made with the Least Square method to evaluate solution accuracy and efficiency.**
    - **Estimations for K values are assessed to determine the accuracy of predicted solution times.**

**PART 1 METHODS TIME AND RESIDUALS**

| **Methods** | **Time** | **Residual** |
|---|---|---|
| **NATURAL** | 685.9105277061462 | 164.39867941770635 |
| **MMD_ATA** | 10.67905569076538 | 164.3986794177065 |
| **COLAMD** | 9.641236066818237 | 164.39867941770652 |
| **QR Decomposition** | 247.41031527519226 | 6.087204528437828e-15 |
| **Bicg_stab** | 0.17081189155578613 | 164.39868365633762 |
| **Preconditioning ILU Bicg_stab** | 0.7809216976165771 | 164.39868307832162 |

In this part, we got a warning called SparseEfficiencyWarning. This warning indicates that the function requires the matrix A to be in Compressed Sparse Row (CSR) or Compressed Sparse Column (CSC) format for efficient computation.

SciPy's spsolve function is designed to solve sparse linear systems, which are represented using sparse matrices. Sparse matrices store only non-zero elements, making them more memory-efficient for large matrices that are mostly comprised of zeros.

And it says that matrix A is not in these formats and we should convert it to achieve more efficient results.

Another warning that we got is in Preconditioning ILU Bicg_stab. (SparseEfficiencyWarning: spilu converted its input to CSC format warn('spilu converted its input to CSC format' ) This warning says that the given A matrix is not in CSC format. The spilu has converted the matrix instead of us.

In summary, the magnitudes of time and residual depend on the specific system being solved, the chosen method, and its implementation details.

LU decomposition is a method used to separate a matrix into lower and upper triangular matrices. However, during this process, new elements are added between the non-zero elements of the matrix and filled.

While MMD_ATA and COLAMD edits are used to reduce the amount of padding, the NATURAL edit is used to increase the amount of padding.

Reducing the amount of padding often helps optimize solution times and memory usage. This can reduce solution times and parsing cost, especially in large and sparse matrices. Increasing the amount of padding can create a denser matrix, which can increase solution times and memory usage.

The main difference between Bicgstab and Bicgstab preconditioning ILU is the application of the ilu preconditioning method in the solution of the matrix in the operations performed with bicgstab with ilu. This can be especially effective when working with large and sparse matrices, since preconditioning can speed up the solution process and reduce the number of iterations.

In all three cases, the residual is calculated by dividing the norm of the error vector created by subtracting the predicted results from the actual results by the norm of the actual results.
This normalization is used to obtain the residual value.
This measurement shows the accuracy of the solution and the magnitude of the error.

**PART 2 CIMMINO AND KACZMARZ METHODS : TIME AND RESIDUALS**

| K VALUE | CIMMINO METHOD | | KACZMARZ METHOD | |
|---|---|---|---|---|
| | RESIDUAL | TIME | RESIDUAL | TIME |
| K = 2 | 15.05285269194509 | 0.00105237960815429 | 17.919306052086778 | 7.066655397415161 |
| K = 4 | 9.396899357558423 | 0.00609517097473144 | 10.793642087987081 | 4.390268087387085 |
| K = 6 | 7.6110242325844775 | 0.0049860477447509 | 8.432083081122826 | 4.445668697357178 |
| K = 8 | 6.501163556371901 | 0.00401735305786132 | 7.092980312488246 | 4.4624645709991455 |

```
Least Square Method   Residual:201.64995049494215   Time:0.01311182975769043
Estimated time for K=6: 7.611024232584473
Estimated time for K=16: 4.568478961493248
Actual time for K=16: 0.002025127410888672
```

This table for 2D matrix because in 3D matrix our Kaczmarz method did not work , it fills up the RAM. That's why the program closed and did not give any results. Therefore, we could not reach a result of the Kachmarz method in 3D.

| K VALUE | CIMMINO METHOD | | KACZMARZ METHOD |
|---|---|---|---|
| | TIME | RESIDUAL | This method did not give any results in 3D matrix |
| K = 2 | 241.25090646743774 | 42.8581369750403 | |
| K = 4 | 59.222782611846924 | 33.55640314402865 | |
| K = 6 | 23.73327660560608 | 29.570712477188895 | |
| K = 8 | 13.836348056793213 | 25.63616995588912 | |

```
Least Square Method   Residual:83.89253827339788   Time:1.1713500022888184
Estimated time for K=6: 29.570712477187968
Estimated time for K=16: 18.478156633341424
Actual time for K=16: 3.092082977294922
```

This table and our Least Square method result for 3D matrix.

**Question 1: Compare and analyze your results with the results of Least Square Method.**

Residuals: Both Cimmino and Kaczmarz methods achieved significantly lower residuals compared to the Least Square Method in both 2D and 3D matrices. This indicates that they produced more accurate solutions.

Time: The Least Square Method was generally faster than Cimmino and Kaczmarz methods, especially in the 2D case. However, the time differences were less pronounced in the 3D case.

RAM usage: Kaczmarz method encountered RAM limitations in the 3D case, preventing it from producing results.

**Question 2: How far is your estimation from the actual result of K=6?**

2D Matrix: The estimated time for K=6 using Cimmino method (7.611024232584473) was very far away to the actual time (0.0049860477447509).

3D Matrix: The estimated time for K=6 using Cimmino method (29.570712477188895) was reasonably close to the actual time(23.73327660560608).

**Question 3: How far is your estimation from the actual result of K=16?**

2D Matrix: The estimated time for K=16 (4.568478961493248) was significantly different from the actual time (0.002025127410888672) in Cimmino method. The estimation overestimated the time required.

3D Matrix: The estimated residual for K=16 using Cimmino method (18.478156633341424) was also relatively far from the actual time (3.092082977294922).

**PART 3 STEPS 6-9 WITH CAGE 10 MATRIX ( Cimmino with cage10)**

| K VALUE | RESIDUAL | TIME |
|---|---|---|
| K = 2 | 0.10255354591735928 | 62.50323987007141 |
| K = 4 | 0.1711938536241569 | 11.45804214477539 |
| K = 6 | 0.22513002609451394 | 4.206461668014526 |
| K = 8 | 0.25779557738977116 | 1.8342106342315674 |

Residual of Least Square Method: 0.2134712984921228
Estimated time for K=6: 0.22513002609465577
Estimated time for K=16: 0.3579749834913371
Actual time for K=16: 0.3919217586517334

In this part we found the cage 10 matrix solution using Cimmino method and we gave a random numbers to our f vector this means that every run our result and time changes with our random f vector.

**Question: Compare and analyze your results with the results of Least Square Method.**

Residuals: In the Cage 10 matrix the residuals of both Cimmino and Least Square method are relatively same. This means that each method found very similar and very accurate solutions

**Question: How far is your estimation from the actual result of K=6?**

Cimmino methods time and our estimation are relatively far away from each other.

**Question: How far is your estimation from the actual result of K=16?**

Our estimated time for k = 16 is very close to our real time. This means we correctly calculated the estimated time and our method fit this estimation.

1. **What are the advantages and disadvantages of direct methods for solving linear equations?**
   Direct methods have advantages such as exact solution, convergence, applicability and stability when solving linear equations. However, it also has disadvantages such as high computational complexity for large systems, memory requirements, difficulty in dealing with ill-conditioned systems, parallelization limitations, and unsuitability for large sparse systems.

2. **Give 3 examples of direct methods and compare their properties with one another.**
   a) Gauss Elimination Method
   b) LU Decomposition
   c) Cholesky Decomposition

   - These methods vary in application: Gaussian Elimination is general purpose, but not suitable for special matrix structures. LU Decomposition is more suitable for general matrices, while Cholesky Decomposition is more suitable for symmetric positive definite matrices.

   - If we look at it from the perspective of separation, Gaussian Elimination operates directly on the original matrix. LU Decomposition and Cholesky Decomposition involve decomposing the matrix into structured forms and provide advantages for iterative solutions.

   - Gaussian Elimination and LU Decomposition have similar O(n^3) complexity. Cholesky Decomposition has O(n^2) complexity. If we make a generalization, Cholesky Decomposition has a complexity of O(n^2) and requires less processing in large linear systems compared to the other two methods. It is said that Cholesky Decomposition can be faster and more efficient, especially when working with large matrices.

   - These methods differ in memory usage. If we look at it from the perspective of separation In general, LU Decomposition and Cholesky Decomposition require less memory than Gaussian Elimination.

3. **What are the advantages and disadvantages of iterative methods for solving linear equations?**
   **Advantages:**
   - Iterative methods are suitable for large sparse systems and can be less costly than direct methods.
   - They can operate with less memory usage, especially advantageous in cases where the matrix is sparse
   - They can effectively leverage parallel computing architectures.
   - They can adapt to special structures of the linear system, making them useful for different types of problems.
   - Fast solutions can be obtained by providing approximate solutions after a few iterations.
     **Disadvantages:**
   - Convergence depends on the quality of the initial estimate. Poor initial estimates can impact performance.
   - It has higher algorithmic complexity per iteration and the number of iterations required for convergence may vary.
   - They can be sensitive to ill-conditioned matrices, which can lead to slow convergence or non-convergence issues.
   - They do not guarantee a complete solution, the solution they provide is approximate and depends on the number of iterations.

- Performance depends on the properties of the specific linear system being solved. Selection and implementation of the preconditioning technique used to increase convergence can be difficult.