Department of Artificial Intelligence

22MAT220: Maths for computing-3

Project Report

Sparse inverse covariance estimation using graphical lasso

Team Members:

Kiran Kishore V : CB.SC.U4AIE23307
K.Koushik :CB.SC.U4AIE23312
Mhokesh P : CB.SC.U4AIE23316
Nithin Pranav S S : CB.SC.U4AIE23326

Under Guidance of

Mr.Neelesh Ashok

Department of Artificial Intelligence
Amrita Vishwa Vidyapeetham, Coimbatore

AMRITA SCHOOL OF ARTIFICIAL INTELLIGENCE
AMRITA VISHWA VIDYAPEETHAM
COIMBATORE – 641 112 (INDIA)

DECLARATION

We, hereby declare that the project entitled is the record of the "Sparse inverse covariance estimation using graphical lasso" the work done by our team under the guidance of Mr.Neelesh Ashok, Amrita School of Artificial Intelligence, Coimbatore. To the best of our knowledge, this work has not formed the basis for the award of any degree/ diploma/ associate ship/ fellowship/ or a similar award to any candidate in any University.

Project Team:

Section - D

Name	Roll Numbers
1. *Kiran Kishore V*	*CB.SC.U4AIE23307*
2. *K.Koushik*	*CB.SC.U4AIE23312*
3. *Mhokesh P*	*CB.SC.U4AIE23316*
4. *Nithin Pranav S S*	*CB.SC.U4AIE23326*

Signature of the Faculty

[Mr.Neelesh Ashok]

Department of Artificial Intelligence

Amrita Vishwa Vidyapeetham, Coimbatore

Place: Coimbatore

Date: 15.11.2024

ACKNOWLEDGEMENT

We express our sincere gratitude to Dr. K.P. Soman, the Dean of our department, for providing us with the opportunity to undertake this valuable project on the topic of “Sparse inverse covariance estimation using graphical lasso” using the python programming language. He encouragement and support have been instrumental in our journey.

We extend our heartfelt thanks to our project guide, Mr.Neelesh Ashok, for his unwavering patience, dedication, and invaluable guidance throughout the project. He expertise and insights have played a significant role in shaping our understanding and implementation of Maths concepts in python programming concepts.

We are immensely grateful to our team members for their collaborative efforts and dedication, without which this project would not have been possible. Each team member's contribution has been invaluable in achieving our objectives.

TABLE OF CONTENT

ABSTRACT

"Sparse Inverse Covariance Matrix Estimation" focuses on developing a robust methodology for identifying direct dependencies among variables using the framework of Gaussian Graphical Models (GGM). The primary objective is to estimate a sparse precision (inverse covariance) matrix that captures meaningful relationships between variables, such as stocks in a financial dataset. This is achieved through the application of the Graphical Lasso method, which imposes an $\ell 1$-penalty to encourage sparsity, effectively filtering out weaker, indirect dependencies. This approach has significant applications in fields such as financial analysis, where understanding direct relationships between assets is critical for portfolio optimization and risk management. By focusing on sparsity, the model achieves interpretability and reduces complexity, allowing for a clearer understanding of the underlying structure of the data. The project's results demonstrate the effectiveness of sparse inverse covariance matrix estimation in uncovering direct, meaningful connections in high-dimensional datasets.

1.INTRODUCTION

The project, titled "Sparse Inverse Covariance Matrix Estimation," aims to identify and analyze direct relationships among stocks. By estimating a sparse precision (inverse covariance) matrix, we focus only on the most important connections between stocks, reducing complexity and highlighting meaningful patterns. This approach uses the *Graphical Lasso* technique, which combines statistics and optimization to create a matrix that captures dependencies while eliminating weaker connections.

We represent these relationships as a network where each stock is a node, and edges show strong dependencies. To understand group behaviors, we then apply a community detection algorithm, *Louvain*, which clusters stocks with similar patterns. This simplified network helps us interpret stock relationships, valuable for financial analysis and risk assessment.

2.LITERATURE REVIEW

Hsieh et al. [1] proposed a novel algorithm for estimating a sparse inverse covariance matrix using a quadratic approximation method. The algorithm focuses on the L1-regularized Gaussian maximum likelihood estimator, leveraging Newton's method instead of traditional first-order gradient approaches. By utilizing the problem's structural properties, it achieves superlinear convergence. Their experimental results on synthetic and real data demonstrate a significant improvement in performance compared to state-of-the-art methods.

Scheinberg et al. [2] introduced a first-order method for sparse inverse covariance matrix estimation based on alternating linearization techniques. Their approach, applied to a convex maximum likelihood problem with an L1-regularization term, exploits the special structure of the problem, yielding subproblems with closed-form solutions. The proposed method achieves an ϵ-optimal solution in $O(1/\epsilon)$ iterations, with numerical experiments showing its superiority over competitive algorithms, particularly in gene association networks.

Friedman et al. [3] developed the graphical lasso, an efficient algorithm for estimating sparse graphs using an L1 penalty on the inverse covariance matrix. The method employs a coordinate descent procedure, solving large-scale problems quickly (e.g., a 1000-node problem in under a minute). The graphical lasso significantly outperforms earlier methods and provides a link between exact solutions and the approximation technique by Meinshausen and Bühlmann. Their experiments on proteomics data illustrate the practical effectiveness of the approach.

3.PROBLEM STATEMENT

In high-dimensional datasets, such as financial data involving numerous stocks, understanding the direct relationships between variables is challenging due to the complexity and potential indirect dependencies. Traditional covariance analysis often captures both direct and indirect correlations, making it difficult to identify meaningful, interpretable connections.

The challenge lies in estimating a model that reveals only the most significant direct dependencies, while discarding weaker, indirect relationships. This is particularly relevant in fields like finance, where accurately identifying clusters of stocks with direct relationships can inform portfolio optimization, risk assessment, and market segmentation.

The goal of this project is to develop a methodology for Sparse Inverse Covariance Matrix Estimation using Gaussian Graphical Models (GGM) and the Graphical Lasso algorithm. By leveraging these techniques, the project aims to:

Estimate a precision matrix that highlights direct relationships between variables in a sparse manner.
Construct a network graph based on these direct dependencies, enabling clearer visualization and analysis of inter-variable connections.
Detect community structures within the graph using the Louvain algorithm, providing insights into clusters of variables that share strong dependencies.

4.METHODOLOGY

Step 1: Data Collection and Preparation :

Objective: The aim of this initial step is to gather historical stock price data and prepare it for further analysis. This includes computing daily returns for each stock, which allows us to study relative price changes over time. Normalization of these returns is done to make the data comparable across different stocks, as raw prices vary significantly.

Process:

1.Data Source and Stock Selection:

We choose a set of stocks and collect historical price data, typically using a financial data provider like Yahoo Finance. The list of stocks, often referred to by their ticker symbols (e.g., AAPL for Apple, MSFT for Microsoft), represents the financial entities we wish to analyze.

The data includes prices for each trading day within a specified time frame (e.g., the last 3 years).

2.Data Fetching:

The stock data is programmatically fetched using libraries like yfinance in Python, which allows us to easily download stock prices.

Each stock's data typically includes the adjusted closing price—the price after accounting for events like dividends and stock splits

For each stock, we compute the daily returns to observe how the stock's value changes relative to the previous day.

3.Calculation of Daily Returns:

For each stock, we compute the daily returns to observe how the stock's value changes relative to the previous day.

$$r_t = \frac{P_t - P_{t-1}}{P_{t-1}}$$

where P_t is the price on day t and P_{t-1} is the price on the previous day.

This calculation results in a time series of returns for each stock, which normalizes different stocks to a common scale and reflects percentage changes rather than absolute values.

4.Normalization:

Once the daily returns are calculated, we normalize them to ensure consistency across different stocks, as some may have more volatile returns than others.

Standardization: This involves scaling each return series to have a mean of 0 and a standard deviation of 1. Standardization makes each stock's return series comparable by adjusting for differences in volatility.

$$z_i = \frac{r_i - \mu}{\sigma}$$

where r_i is the return, μ is the mean, and σ is the standard deviation

Output:

After this step, we have a DataFrame of standardized daily returns for each stock. These returns are now on a comparable scale and are ready for further processing, where we will analyze the dependencies between stocks.

Step 2: Sparse Inverse Covariance Estimation via Graphical Lasso :

Objective: The goal of this step is to estimate a sparse inverse covariance matrix, also known as the precision matrix, which captures direct dependencies between stocks. By enforcing sparsity, we limit the number of connections, making it easier to interpret which stocks are directly related. This method uses Graphical Lasso, a technique that combines statistical estimation with an optimization-based approach to create a precision matrix with zero entries for weak relationships.

Background and Mathematical Foundation:

1.Covariance Matrix:

The covariance matrix, often denoted as S, provides a measure of the pairwise relationships between stocks based on their returns. However, covariance alone cannot distinguish between direct and indirect dependencies, which can make the relationships hard to interpret.

For a matrix S with elements s(i,j), where each entry represents the covariance between stocks i and j, a high value implies that the stocks often move together.

2.Precision Matrix:

The precision matrix, denoted Θ, is the inverse of the covariance matrix, Θ=*S*^-1. It captures direct dependencies—when two stocks have a non-zero precision matrix entry, they are directly connected.

Sparsity in the precision matrix implies that many entries are zero, which makes the graph less dense and reveals only the strongest, most direct relationships.

3.Graphical Lasso (GLasso):

Graphical Lasso is a regularization method that estimates a sparse precision matrix by optimizing a penalized log-likelihood function. The optimization problem is:

$$\text{minimize} \quad -\log\det\Theta + \text{trace}(S\Theta) + \alpha||\Theta||_1$$

det denotes the determinant of Θ, and −logdetΘ ensures that the matrix is positive definite.

trace(SΘ) aligns Θ with the sample covariance matrix S, capturing direct dependencies.

α||Θ|| is an ℓ1-norm penalty term. This penalty term enforces sparsity by shrinking weak dependencies to zero.

4.Interpretation of α:

The regularization parameter α controls the level of sparsity in the precision matrix. A higher α results in more zero entries in Θ, reducing the number of connections. This provides a trade-off between accuracy (fitting the data) and interpretability (limiting connections to the most important ones).

Cross-validation is often used to select the optimal α value that balances sparsity with the precision of the model.

Output:

The result is a sparse precision matrix where each non-zero entry represents a direct dependency between two stocks. This matrix will be used to construct a network in the next step, where each stock is represented by a node and each non-zero matrix entry by an edge.

Step 3: **Network Construction and Thresholding**

Objective: Using the sparse precision matrix estimated in the previous step, we construct a network where each node represents a stock, and edges represent direct dependencies between them. To simplify the network further and emphasize only the strongest connections, we apply a threshold to remove weaker links.

Process:

1.Understanding the Sparse Precision Matrix:

In the precision matrix Θ, each non-zero entry θ(i,j) indicates a direct dependency between stocks i and j. The magnitude of θ(i,j) shows the strength of this dependency; a higher value suggests a stronger connection.

2. Thresholding the Precision Matrix:

Although the Graphical Lasso already promotes sparsity, we can further refine the network by

thresholding the precision matrix to retain only the strongest relationships.

We define a threshold τ and set any entry $\theta(i,j)$ with $|\theta(i,j)| < \tau$ to zero. This process removes weaker connections, leaving only the most significant relationships.

3. Network Representation:

We represent the sparse precision matrix as a graph G, where:

Nodes: Each node corresponds to a stock in the dataset.

Edges: An edge is drawn between two nodes i and j if $\theta(i,j) \neq 0$ after thresholding. The edge weight can be the magnitude $|\theta(i,j)|$, indicating the strength of the dependency.

This graph structure, often known as a financial network, captures significant relationships between stocks while minimizing complexity.

4. Graph Construction using NetworkX:

The networkx library in Python allows for creating and manipulating graphs. Each stock is added as a node, and edges are added for every non-zero, thresholded entry in the precision matrix.

Choosing the Threshold:

The threshold τ can be selected based on domain knowledge or by analyzing the distribution of values in the precision matrix. For instance, retaining only the top 10% of values might highlight the strongest dependencies.

Output:

The result is a sparse network graph G, where each edge represents a significant relationship between two stocks. This graph simplifies the complexity of the stock dependencies, focusing only on the most impactful relationships.

Step 4: Community Detection Using the Louvain Algorithm

Objective: With the network graph constructed, the next step is to identify clusters or communities of stocks that are strongly interconnected. The Louvain Algorithm is used here to detect these communities by maximizing a measure called modularity. This helps group stocks that have similar behavior or direct dependencies, providing insights into clusters of stocks that may behave similarly.

Background and Mathematical Foundation:

1.Community Detection and Modularity:

In network analysis, community detection aims to find groups of nodes (stocks, in this case) that are more densely connected to each other than to the rest of the network.

The modularity metric measures the quality of a given division of nodes into communities. It quantifies the density of edges within communities compared to edges between communities.

Modularity Q is defined as:

$$Q = \frac{1}{2m} \sum_{i,j} \left(A_{ij} - \frac{k_i k_j}{2m} \right) \delta(c_i, c_j)$$

A(i,j) is the adjacency matrix of the network, where A(i,j)=1 if there is an edge between nodes i and j.
k and j are the degrees of nodes i and j (number of connections each node has).
m is the total number of edges in the network.
$\delta(c,c)$ is 1 if nodes i and j are in the same community, and 0 otherwise.

2.The Louvain Algorithm:
The Louvain Algorithm is a popular, efficient method for community detection. It operates in two main phases:
Phase 1 (Local Moving Phase): Each node is moved to the community that maximizes modularity gain. This phase continues until no further gain is possible.
Phase 2 (Aggregation Phase): Communities are collapsed into single nodes, creating a new, smaller network where edges represent connections between communities from Phase 1. The algorithm repeats Phase 1 and Phase 2 iteratively until modularity can no longer be increased.

Output:
The output is a dictionary where each stock is mapped to a community label. Stocks with the same label belong to the same community, indicating they have significant relationships with each other.

Step 5: Visualization of the Network with Community Labels
Objective: The goal of this step is to create a visual representation of the stock network, with nodes colored by their community labels. This allows for an intuitive understanding of the clusters detected by the Louvain algorithm and helps highlight the strongest inter-stock relationships in an accessible format.

Process:

1.Graph Layout:
We use a layout algorithm to position the nodes in a way that reveals the structure of the network. Common layouts include spring layout and circular layout.

The spring layout (using nx.spring_layout in networkx) simulates nodes as physical objects connected by springs, so that nodes with stronger connections are drawn closer together. This layout typically reveals clusters effectively.

2.Node Colors by Community:

Each community detected in the previous step is assigned a unique color. This color-coding helps differentiate groups visually, allowing us to quickly see the clusters formed in the network.

The community information from the dictionary partition (output from the Louvain algorithm) is used to assign colors to nodes based on their community.

3.Edge Weight Visualization:

To further enhance the visualization, edges can be drawn with widths proportional to their weights (i.e., the strength of the relationship between nodes). Thicker edges indicate stronger relationships.

This helps emphasize the strongest connections in the network, making it easy to spot the most important relationships.

Output:

The output is a plot displaying the stock network, with nodes colored by community. Each color represents a different community, and thicker edges show stronger dependencies between nodes.

Interpretation:

In this visualization, clusters (or communities) stand out due to color differences. You can interpret these clusters as groups of stocks that have strong interdependencies, likely reflecting shared sector influences or similar market behaviors.

Thicker edges highlight the strongest dependencies, while node positioning further emphasizes relationships between clusters.

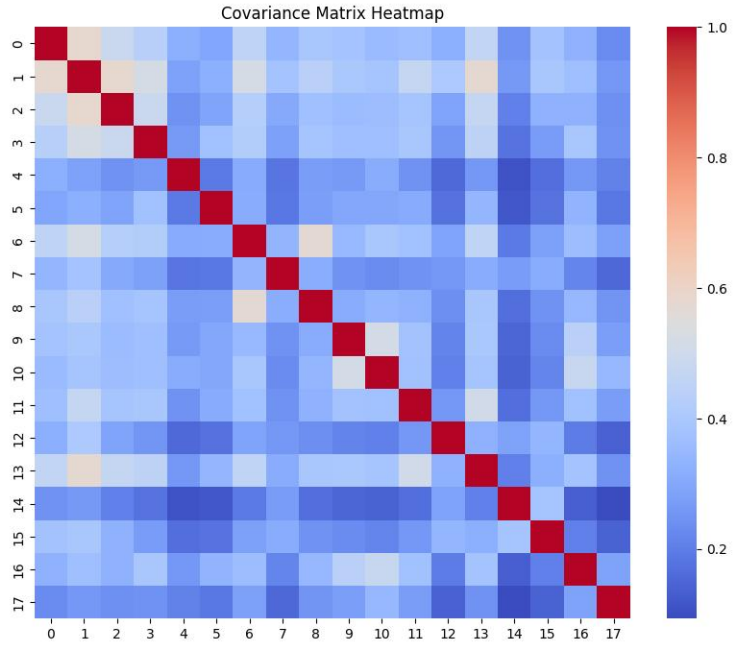
5.RESULTS

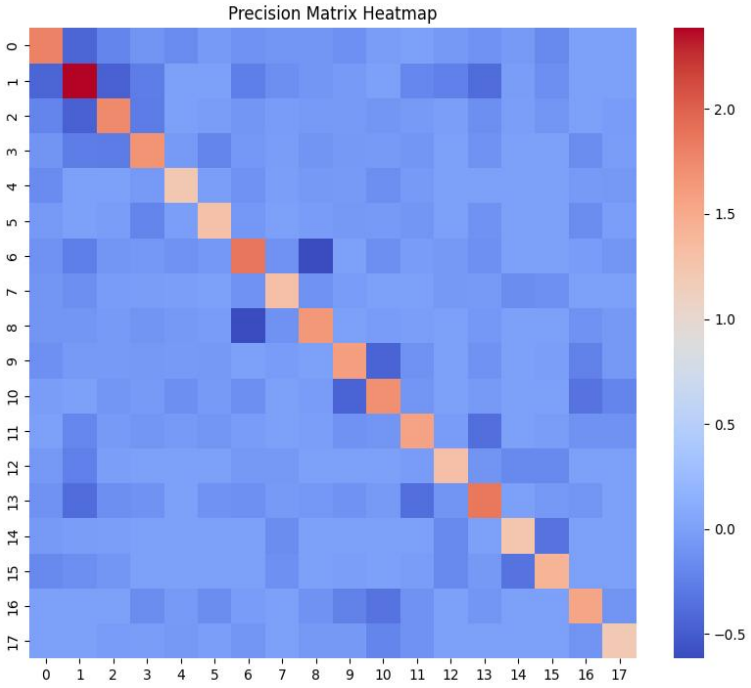
The Resultant Covariance and Precision Matrix :

```
Covariance matrix:
 [[1.         0.57979744 0.48459167 0.43057355 0.31742169 0.29257537
  0.45119862 0.34075539 0.39579345 0.38044411 0.35421499 0.36818449
  0.32019893 0.46189935 0.24599334 0.37790317 0.32559347 0.2282739 ]
 [0.57979744 1.         0.58196563 0.51842062 0.28398385 0.32316944
  0.52164272 0.38237265 0.44028169 0.39826249 0.38509025 0.47113341
  0.40457467 0.57581183 0.26132076 0.39555801 0.36691317 0.25659047]
 [0.48459167 0.58196563 1.         0.48381378 0.24646965 0.28938199
  0.42885776 0.30527288 0.37284519 0.35689815 0.36116856 0.38574948
  0.28910278 0.4692778  0.20582274 0.32500736 0.32668847 0.24081607]
 [0.43057355 0.51842062 0.48381378 1.         0.26558276 0.37372425
  0.4211755  0.28080491 0.38506868 0.36861551 0.36900886 0.39386603
  0.2561007  0.4505146  0.17543874 0.26738284 0.39232464 0.24535199]
 [0.31742169 0.28398385 0.24646965 0.26558276 1.         0.19081878
  0.30698557 0.18135327 0.27078382 0.26364314 0.31064164 0.24869586
  0.15421389 0.25771101 0.1097495  0.16557479 0.25942505 0.2078374 ]
 [0.29257537 0.32316944 0.28938199 0.37372425 0.19081878 1.
  0.31014053 0.18809319 0.27403244 0.29592295 0.29680459 0.30939525
  0.17182387 0.33890657 0.11666269 0.17810124 0.3331585  0.18718997]
 [0.45119862 0.52164272 0.42885776 0.4211755  0.30698557 0.31014053
  1.         0.33575842 0.57217951 0.3501111  0.39722886 0.37669946
  0.28858145 0.45801434 0.19081783 0.28065839 0.36107015 0.27963389]
 [0.34075539 0.38237265 0.30527288 0.28080491 0.18135327 0.18809319
  0.33575842 1.         0.31460023 0.24348884 0.22922564 0.2458577
  0.25735575 0.31202383 0.26769628 0.31272219 0.21664648 0.15297391]
 [0.39579345 0.44028169 0.37284519 0.38506868 0.27078382 0.27403244
  0.57217951 0.31460023 1.         0.30943867 0.34111022 0.32656992
  0.23677958 0.39504442 0.16700259 0.24571849 0.3462747  0.25157964]
 [0.38044411 0.39826249 0.35689815 0.36861551 0.26364314 0.29592295
  0.3501111  0.24348884 0.30943867 1.         0.51473294 0.37971264
 0.21250851 0.39887884 0.14917838 0.23071695 0.4374916  0.27705701]
[0.35421499 0.38509025 0.36116856 0.36900886 0.31064164 0.29680459
 0.39722886 0.22922564 0.34111022 0.51473294 1.         0.37595364
 0.20618155 0.38500531 0.14129166 0.21633045 0.47912614 0.34593173]
[0.36818449 0.47113341 0.38574948 0.39386603 0.24869586 0.30939525
 0.37669946 0.2458577  0.32656992 0.37971264 0.37595364 1.
 0.2579557  0.50492172 0.16435304 0.25684854 0.37394335 0.27351585]
[0.32019893 0.40457467 0.28910278 0.2561007  0.15421389 0.17182387
 0.28858145 0.25735575 0.23677958 0.21250851 0.20618155 0.2579557
 1.         0.32680348 0.28708317 0.33808974 0.19585862 0.13737126]
[0.46189935 0.57581183 0.4692778  0.4505146  0.25771101 0.33890657
 0.45801434 0.31202383 0.39504442 0.39887884 0.38500531 0.50492172
 0.32680348 1.         0.20619592 0.32068988 0.38084749 0.24792215]
[0.24599334 0.26132076 0.20582274 0.17543874 0.1097495  0.11666269
 0.19081783 0.26769628 0.16700259 0.14917838 0.14129166 0.16435304
 0.28708317 0.20619592 1.         0.38653932 0.13296463 0.0933965 ]
[0.37790317 0.39555801 0.32500736 0.26738284 0.16557479 0.17810124
 0.28065839 0.31272219 0.24571849 0.23071695 0.21633045 0.25684854
 0.33808974 0.32068988 0.38653932 1.         0.20329565 0.14243655]
[0.32559347 0.36691317 0.32668847 0.39232464 0.25942505 0.3331585
 0.36107015 0.21664648 0.3462747  0.4374916  0.47912614 0.37394335
 0.19585862 0.38084749 0.13296463 0.20329565 1.         0.28777832]
[0.2282739  0.25659047 0.24081607 0.24535199 0.2078374  0.18718997
 0.27963389 0.15297391 0.25157964 0.27705701 0.34593173 0.27351585
 0.13737126 0.24792215 0.0933965  0.14243655 0.28777832 1.        ]]
```

```
Precision matrix:
 [[ 1.78899952e+00 -4.36811381e-01 -2.08474981e-01 -9.27940892e-02
  -1.66991297e-01 -4.55868718e-02 -1.23144164e-01 -8.89613554e-02
  -8.80349621e-02 -1.31559380e-01 -2.35259689e-02 -0.00000000e+00
  -5.46628837e-02 -1.09591013e-01 -4.91707164e-02 -1.82872787e-01
  -0.00000000e+00 -0.00000000e+00]
 [-4.36811381e-01  2.38526631e+00 -4.64618892e-01 -2.67576819e-01
  -0.00000000e+00 -2.18544692e-03 -2.56054129e-01 -1.37166978e-01
  -7.80735914e-02 -5.09196578e-02 -0.00000000e+00 -2.02592037e-01
  -2.49744462e-01 -3.86976264e-01 -2.86176052e-02 -1.38134622e-01
  -0.00000000e+00 -0.00000000e+00]
 [-2.08474981e-01 -4.64618892e-01  1.74811593e+00 -2.78879836e-01
  -0.00000000e+00 -1.93215484e-02 -7.91073815e-02 -3.83978104e-02
  -4.56672339e-02 -4.94303389e-02 -8.06347749e-02 -5.37058636e-02
  -1.03990789e-02 -1.39204290e-01 -8.09378040e-03 -8.28345431e-02
  -0.00000000e+00 -3.78842993e-02]
 [-9.27940892e-02 -2.67576819e-01 -2.78879836e-01  1.67242525e+00
  -5.03449113e-02 -2.15212162e-01 -7.07252966e-02 -2.84115411e-02
  -9.75332094e-02 -6.22108316e-02 -4.22366858e-02 -8.55501678e-02
  -0.00000000e+00 -1.09045468e-01 -0.00000000e+00 -0.00000000e+00
  -1.57902836e-01 -2.62076241e-02]
 [-1.66991297e-01 -0.00000000e+00 -0.00000000e+00 -5.03449113e-02
   1.21485049e+00 -1.28595041e-02 -1.05153706e-01 -1.61834715e-02
  -6.42491154e-02 -4.32120508e-02 -1.42996294e-01 -4.60629112e-02
  -0.00000000e+00 -0.00000000e+00 -0.00000000e+00 -0.00000000e+00
  -5.02462017e-02 -6.30755833e-02]
 [-4.55868718e-02 -2.18544692e-03 -1.93215484e-02 -2.15212162e-01
  -1.28595041e-02  1.28445033e+00 -7.00934440e-02 -5.66818378e-03
  -3.46725531e-02 -6.53221091e-02 -4.88893243e-02 -8.88270226e-02
  -0.00000000e+00 -1.06164394e-01 -0.00000000e+00 -0.00000000e+00
  -1.50804523e-01 -1.11257153e-02]
 [-1.23144164e-01 -2.56054129e-01 -7.91073815e-02 -7.07252966e-02
  -1.05153706e-01 -7.00934440e-02  1.87721151e+00 -1.02096390e-01
  -6.16807258e-01 -0.00000000e+00 -1.38015082e-01 -3.78251160e-02
  -5.63866464e-02 -1.35370672e-01 -0.00000000e+00 -0.00000000e+00
  -3.83777696e-02 -7.85123211e-02]
 [-8.89613554e-02 -1.37166978e-01 -3.83978104e-02 -2.84115411e-02
  -1.61834715e-02 -5.66818378e-03 -1.02096390e-01  1.30038884e+00
  -1.21566873e-01 -4.10026595e-02 -0.00000000e+00 -0.00000000e+00
  -5.85153594e-02 -4.41388125e-02 -1.51207854e-01 -1.34642975e-01
  -0.00000000e+00 -0.00000000e+00]
 [-8.80349621e-02 -7.80735914e-02 -4.56672339e-02 -9.75332094e-02
  -6.42491154e-02 -3.46725531e-02 -6.16807258e-01 -1.21566873e-01
   1.64125099e+00 -0.00000000e+00 -3.47079598e-02 -1.36542555e-02
  -0.00000000e+00 -6.80154454e-02 -0.00000000e+00 -5.56693144e-03
  -1.14352673e-01 -6.06289748e-02]
 [-1.31559380e-01 -5.09196578e-02 -4.94303389e-02 -6.22108316e-02
  -4.32120508e-02 -6.53221091e-02 -0.00000000e+00 -4.10026595e-02
  -0.00000000e+00  1.59202948e+00 -4.49570877e-01 -1.17109485e-01
  -0.00000000e+00 -1.07088337e-01 -0.00000000e+00 -1.57122906e-02
  -2.32081148e-01 -6.02189100e-02]
 [-2.35259689e-02 -0.00000000e+00 -8.06347749e-02 -4.22366858e-02
  -1.42996294e-01 -4.88893243e-02 -1.38015082e-01 -0.00000000e+00
  -3.47079598e-02 -4.49570877e-01  1.69647615e+00 -8.76023475e-02
  -0.00000000e+00 -5.13115447e-02 -0.00000000e+00 -0.00000000e+00
  -3.38502742e-01 -2.06865259e-01]
 [-0.00000000e+00 -2.02592037e-01 -5.37058636e-02 -8.55501678e-02
  -4.60629112e-02 -8.88270226e-02 -3.78251160e-02 -0.00000000e+00
  -1.36542555e-02 -1.17109485e-01 -8.76023475e-02  1.56208210e+00
  -3.57505318e-02 -3.70895799e-01 -0.00000000e+00 -2.68637619e-02
```

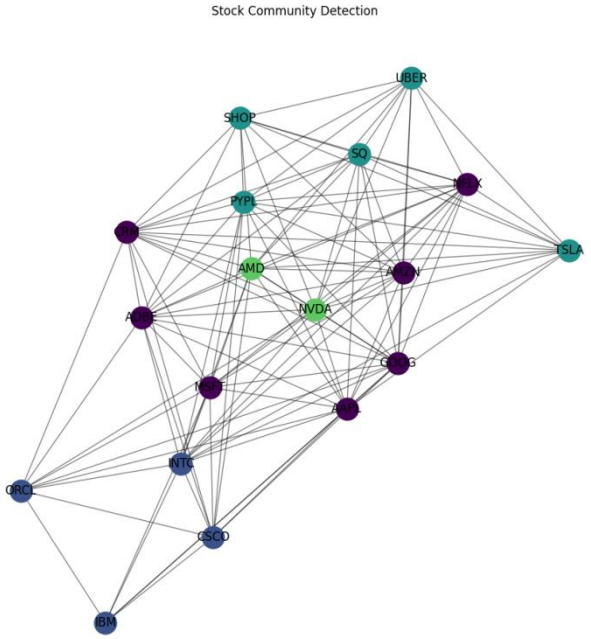
The Resultant Heatmap for Both Matrix :



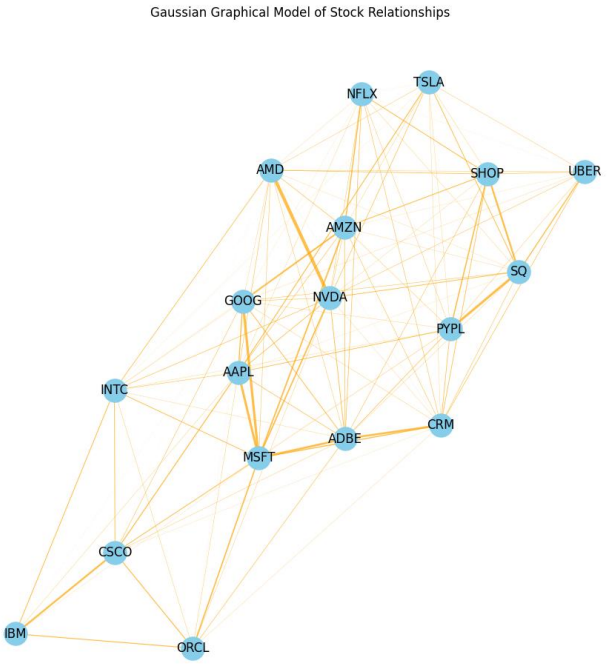


The Resultant Graph's :

1. Community Detection :



2. Gaussian Graph Model :



6. CONCLUSION

The project on Sparse Inverse Covariance Matrix Estimation successfully demonstrates the application of Gaussian Graphical Models (GGM) and the Graphical Lasso method to uncover direct dependencies between variables in high-dimensional datasets. By leveraging the Graphical Lasso algorithm, the project effectively estimates a sparse precision matrix, filtering out weaker, indirect correlations and highlighting only the most significant relationships. This sparse representation leads to a more interpretable and insightful understanding of the underlying data structure.

The construction of a network graph based on the estimated precision matrix enables clear visualization of the direct dependencies, providing an intuitive way to analyze the relationships between variables. Through community detection using the Louvain algorithm, the project identifies meaningful clusters of related variables, which can offer valuable insights, especially in the context of financial analysis, where identifying groups of interdependent stocks is crucial for portfolio management and risk mitigation.

`

7.REFERENCES

[1] Cho-jui Hsieh, Inderjit Dhillon, Pradeep Ravikumar, Mátyás Sustik, "Sparse Inverse Covariance Matrix Estimation Using Quadratic Approximation," 2011.

[2] Katya Scheinberg, Shiqian Ma, Donald Goldfarb, "Sparse Inverse Covariance Selection via Alternating Linearization Methods," 2010.

[3] Jerome Friedman, Trevor Hastie, Robert Tibshirani, "Sparse Inverse Covariance Estimation with the Graphical Lasso," 2007.