International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395-0056
Volume: 10 Issue: 07 | July 2023 www.irjet.net p-ISSN: 2395-0072

A MATLAB Computational Investigation of the Jordan Canonical Form of a Class of Zero-One Matrices

Raouth R. Ghabbour[1], Magdy Tawfik Hanna[2]

[1] *Demonstrator, Department of Engineering Mathematics and Physics, Fayoum University, Fayoum, Egypt,*
[2] *Professor, Department of Engineering Mathematics and Physics, Fayoum University, Fayoum, Egypt,*

Abstract – *The difficult task of creating the Jordan canonical form is handled only for a class of zero-one square matrices with the additional property that each column has at most one nonzero element. The method is based on the construction and analysis of the adjacency matrix of directed graph. The computational study focuses primarily on the creation of the Jordan canonical form and modal matrix which contains eigenvectors and generalized eigenvectors.*
The computation's accuracy is measured by calculating the difference between the provided matrix and that created by combining the Jordan form and the modal matrix. Both the maximum element in absolute value and the Frobenius norm of the difference matrix - defined as the difference between the given and computed matrices - are used as error measures. The computational investigation is carried out by creating a set of MATLAB functions that can be seen as a novel contributed toolbox. Fortunately, it was discovered that this toolbox outperformed the built-in MATLAB function "jordan" because the contributed toolbox successfully processed square matrices of order up to 1000 while the built-in function of the MATLAB lingered for matrices of order between 40 to 60.

Key Words: **Directed graphs, Jordan canonical form, zero-one matrices.**

1.INTRODUCTION

Jordan canonical form is a specific type of upper triangular matrix's representation of a linear transformation over a finite dimensional vector space. Every such linear transformation has a distinct Jordan canonical form with the nice properties. It is simple to explain and well-suited for calculation. Every square matrix is similar to a unique matrix in Jordan canonical form, because similar matrices correspond to representations of the same linear transformation with respect to different bases, according to the change of basis theorem. Jordan canonical form is a generalisation of diagonalizability to arbitrary linear transformations (or matrices); in fact, the Jordan canonical form of a diagonalizable linear transformation (or matrix) is a diagonal matrix also. To compute Jordan canonical form, most researchers in matrix theory and most users of its methods are aware of the importance of graphs in linear algebra. This can be seen from the large number of papers in which graph theoretic methods for solving problems in linear algebra are used. The use of combinatorial and graph theoretic methods for understanding the Jordan canonical form has a long history. In 1837, Jacobi showed that a square matrix of order n is similar to an upper triangular matrix. Many proofs of the Jordan form rely on this result.
Directed graphs may help in dealing with several problems in neural networks and learning systems, network science and engineering, and automatic control[1]–[8]. Previously, Cardon and Tuckfield [9] proposed an algorithm for finding the Jordan canonical form of a class of non-diagonalizable zero-one square matrices - with the property that each column has at most one nonzero element - using the directed graph (digraph) tool. The Jordan form was obtained for the adjacency matrix describing the directed graph.

It is possible to date the beginning of graph theory to 1735, when the Swiss mathematician Leonhard Euler found an answer to the Königsberg bridge puzzle [10], [11]. The Königsberg bridge problem was an old puzzle that involved trying to find a way over each of the seven bridges that span a branched river that flows by an island without using any of them more than once. As shown in Figure 1, there were seven bridges in the city crossing a waterway between two banks and two islands. The question was whether it was practical to cross every bridge just once and return to the starting point of the journey. The response was no, it is impossible to cross every bridge precisely once. It should be mentioned that the Königsberg bridge problem is not a simple graph example because there are multiple edges connecting the same two nodes, which is acceptable in a multigraph. In a multigraph with four vertices and seven edges, the issue mathematically comes down to finding an Eulerian cycle [11].

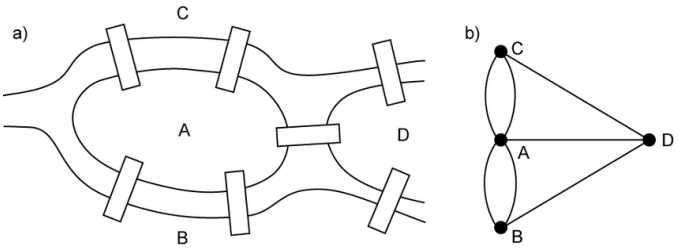


Figure 1: Königsberg bridge puzzle (a) Real seven bridge problem, (b) Bridge puzzle graph representation **[12]**.

Data charts like line graphs and bar graphs are not considered graphs in the context that the term is used in graph theory. Instead, it describes a collection of edges (or lines) that connect the vertices, which are points or nodes.

Graphs are widely used models of both biological and artificial systems. They can be used to simulate a wide range of relationships and processes in computer science[13]–[15] such as artificial intelligence [16], physical, biological such as the human brain networks and functions[17]–[19], and social systems. Graphs can be used to represent a wide range of practical issues. In general, graph theory has numerous applications in a variety of disciplines.

Almost every biological research begins with an analysis of interactions in biological systems. Graph theory, as a mathematical formal language, seeks to describe relationships and interactions between objects as an essential tool [20], assisting in the analysis of biological interaction networks at various scales. Many applications of graph theory to biology have been suggested and developed over the last few decades [21], [22]

A network can be used to address a wide range of issues [23], [24] such as a system of roads [25], study natural phenomena [26], in industry fields such as fabric design [27],study the connectivity and dysconnectivity of networks [28], subnetworks [29], modelling nano scale networks [30], [31], electromagnetic systems [32], railroads, or electric lines and circuits, etc. [40]– [58].

In software development graph theory is used to visualize software component [52], [53] and in mathematical science graph theory is a strong tool to solve problems [54], [55].

In chemical field, graph theory is used to visualize the structure of molecules [44], [47]. These systems build networks that can be formulated using a graph as a model. This can be thought of as an abstract representation of a collection of items that are linked together in pairs.

It can be used to simply model the topology of space, which is the connectivity between different components of space. But in order to properly understand why and what graphs exist, one must go deeply into graph theory. Graph theory is the field of discrete mathematics which studies networks of points that are linked by lines.

Graphs and matrices are inextricably linked. A matrix is a collection of numbers that are arranged in rows and columns to form a rectangular array. Some matrices can provide useful information about graphs, such as how many vertices are connected, how many paths exist between two vertices, and so on [56],[57].

An adjacency matrix is a matrix of zeros and ones that determines whether two vertices have an edge between them. If two vertices are connected, the number 1 is inserted at the corresponding matrix entry [58].

Jordan Canonical Form (JCF). In 1987, Richard A. Brualdi proved that the Jordan canonical form has the greatest number of off-diagonal zero entries among all matrices that are similar to a given square complex matrix. He also described the matrices that achieve this highest number [59].

The reason for developing a MATLAB toolbox for calculating the JCF is that the built-in function in MATLAB called "jordan" fails to calculate the JCF of square matrices of order higher than 60. This is due to the sensitivity of computing the JCF. Consider the following matrix A:

$$A=\begin{pmatrix}1 & 1\\ x & 1\end{pmatrix}$$

Matrix A is non-diagonalizable. However, the matrix is a Jordan block if $x=0$. This means $A=J$. Hence, the first column of the identity matrix **I** is an eigenvector and the second column is a generalized eigenvector.

$$A=\begin{pmatrix}1 & 1\\ x & 1\end{pmatrix}_{x=0} \Rightarrow J=A=\begin{pmatrix}1 & 1\\ 0 & 1\end{pmatrix}$$

On the other hand, if $x \neq 0$, even if x is very small number, matrix A becomes diagonalizable, and $J \neq A$. The diagonal matrix D of A is

$$D=\begin{pmatrix}1+\sqrt{x} & 0\\ 0 & 1-\sqrt{x}\end{pmatrix}$$

As a result, developing an accurate numerical algorithm for Jordan form computation is difficult and sensitive. For this reason, the Jordan Form is typically avoided in numerical representation.

2. JORDAN CANONICAL FORM

Let $f:\square \to \square$ be any function, where $\square$ is natural number. For each $n \in \mathcal{N}$, define the square matrix $\mathbf{A_n}=\left(a_{i,j}\right)$ of order n by:

$$a_{i,j}=\begin{cases}1 & if\ i=f(j), \quad i,j\in\square\\ 0 & otherwise\end{cases} \quad \textbf{(1)}$$

Matrix $\mathbf{A_n}$ can be regarded as the adjacency matrix of a digraph (directed graph) Γ_n with vertices labeled $1,\cdots,n$ having a directed edge from vertex j to vertex i if and only if $i = f(j)$.

Remark: The out-degree of any vertex must be less than or equal to one.

We consider a partition of the weighted directed graph Γ_n, into chains and cycles. The Jordan decomposition of the weighted adjacency matrix A will be related to the lengths of these chains and cycles. Some definitions related to Eq. (1) are next presented.

Merge points: the points which have an in-degree greater than one.

Terminal points: the vertices having a zero outdegree and the vertices having a function value larger than the order of the adjacency matrix A. For example, for a matrix A of order 5, if $f(3) = 500$ then vertex 3 is a terminal point.

Fortunately, MATLAB has two built-in functions called "indegree" and "outdegree" that give the merge points and the terminal points respectively.

Cycle: A set of vertices such that each vertex is a function of the previous vertex. For example, the first vertex continues to be a function of the last vertex of the cycle. Moreover, when a cycle is extracted, it should be written according to the sequence of its vertices no matter which vertex is the starting one.

In MATLAB, a function called "allcycles" is modified in order to obtain the cycles $Z = \{Z_1,\ldots, Z_r\}$ of a directed graph Γ_n. The result is in a form of a cell array containing the extracted cycles.

Chain: a set of vertices that is connected in an open looped sequence. The last vertex of this set sequence can be either a terminal point, or its function (the following point not included in the set) is a merge. However, that merge point can be located on either another chain or a cycle.

Remark: a single point can be considered a cycle if the function of the point is the point itself. Alternatively, it can be considered a chain when the function of this point is unidentifiable.

If the graph does not contain cycles $(Z = [\ ])$, the MATLAB routine "findchains" considers the graph as group of chains only. Then, it starts to extract chains sorted by length in a cell array. If the graph includes cycles, the found cycles Z are excluded from the graph using the code "findchains2", hence the remaining edges represent chains $C = \{C_1,\ldots, C_S\}$ of the graph Γ_n. The resulting array of chains is sorted in descending order of the lengths of the chains. For instance, length (C_1) is the largest, and length (C_S) is the smallest.

The combination of cycles and chains represents the proper partition P of graph Γ_n, where $P = \{Z_1,..,Z_r,C_1,...,C_S\}$ according to definition 3 in [9].

Remark: the proper partition is not unique i.e., if two proper partitions exist

$P = \{Z_1,\ldots,Z_r,C_1,...,C_S\}$ and $P^* = \{Z_1^*,...,Z_r^*,C_1^*,\ldots, C_S^*\}$, then the two groups of cycles $\{Z_1,\ldots, Z_r\}$ and $\{Z_1^*,\ldots, Z_r^*\}$ are the same up to reordering and the two groups of the chains are not the identical but have the same lengths of each chain i.e., $Len(C_i) = Len(C_i^*)$ for $1 \le i \le s$. For example, the following graph has proper partition:

$Z_1 = \{1,4,2\}$, $Z_2 = \{5,7\}$, $C_1 = \{8,6,3\}$, $C_2 = \{11,10\}$, $C_3 = \{9\}$

Another one,

$Z_1 = \{1,4,2\}$, $Z_2 = \{5,7\}$, $C_1 = \{11,10,3\}$, $C_2 = \{8,6\}$, $C_3 = \{9\}$

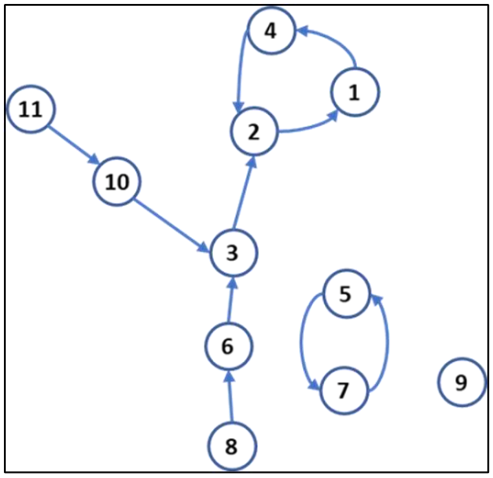


Figure 2: Example 1 digraph.

$$A = \begin{bmatrix} 0&1&0&0&0&0&0&0&0&0&0 \\ 0&0&1&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&1&0 \\ 1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1 \\ 0&0&0&0&0&0&0&0&0&0&0 \end{bmatrix}$$

Definition 1. Let A be an adjacency matrix corresponding to a digraph, and let v be a generalized eigenvector corresponding to eigenvalue λ. Let p be the smallest positive integer such that $(A - \lambda I)^p (v) = 0$, where p is the length of the chain of the generalized eigenvector, though the generalized eigenvectors of matrix A in order are

$\left\{(A-\lambda I)^{p-1}(v),(A-\lambda I)^{p-2}(v),\ldots,(A-\lambda I)(v),v\right\}$ which is called a chain of generalized eigenvectors of A corresponding to λ. The first term in the ordered set is an eigenvector.

The eigenvalues, eigenvectors and generalized eigenvectors can be computed by MATLAB through two different approaches: the zero-one matrix and the weighted matrix. These approaches are discussed briefly in the next two sections. In order to proceed to the calculation of the eigenvalues and generalized eigenvectors, one resorts to theorem 6 in [9].

2.1 Eigenvalues

Each vertex in the graph has an associated eigenvalue. First, eigenvalues associated with cycles composed of one vertex are set to 1, hence its Jordan block illustrated in definition.1 is $J_1(1)$. When the cycle has two or more vertices, one computes the primitive ℓ_j root of unity $\omega_j = exp\left(\frac{2\pi i}{\ell_j}\right)$ where ℓ_j is the length of the cycle. Consequently, the eigenvalues pertaining to the vertices starting from the first vertex are ω, ω^2,…, ω^h, where h is the number of vertices in the cycle. This gives the Jordan blocks $J_1(\omega), J_1(\omega^2),\ldots,J_1(\omega^h)$. For example, let the cycles of a proper partition be $Z=\{Z_1,\ldots,Z_r\}$ where $Z_j=\{z_1,\ldots,z_{\ell_j}\}$ is any cycle in Z of length ℓ_j and $1\le j\le r$. Finally, in order to get the eigenvalue pertaining to the k[th] vertex in a cycle of length ℓ_j, one applies:

$$\lambda_j^k = exp\left(\frac{2\pi i}{\ell_j}\right)^k, \quad k=1,\cdots,\ell_j \qquad (2)$$

The subscript j refers to the cycle number in a proper partition since a value of ω^2 in a cycle of three vertices is definitely different from the value of ω^2 in another cycle of a different number of vertices.

The eigenvalues associated with the vertices of the chains are set to be zero. Let the chains of a proper partition be $C=\{C_1, C_2,\ldots C_s\}$ where $C_u=\{c_1, c_2, \ldots, c_{\ell_u}\}$ and length $(C_u)=\ell_u$ and $1\le u\le s$. Hence, the Jordan decomposition has $J_{\ell_u}(0)$ block associated to chain C_u and the order of the Jordan block is equal to the length of the chain. Therefore, the eigenvalues are independent of the values of the vertices although they depend on the location of the vertices whether they are on a cycle or a chain.

Example.1 the proper partition P for a graph Γ in **Figure 2** is $Z_1=\{1,4,2\}$, $Z_2=\{5,7\}$, $C_1=\{8,6,3\}$, $C_2=\{11,10\}$, $C_3=\{9\}$.

The eigenvalues of example 1 are listed in Table 1.

Table 1. Example 1 eigenvalues.

No.	Eigenvalue's numbering associated to vertex number (λ)	Substitution in Eq. (2)	The eigenvalue	Notes
1	λ_1	$\omega_1^1 = exp\left(\frac{2\pi*1i}{3}\right)$	$\omega=-\frac{1}{2}+\frac{\sqrt{3}}{2}i$	The 3 roots of unity
2	λ_4	$\omega_1^2 = exp\left(\frac{2\pi*2i}{3}\right)$	$\omega^2=-\frac{1}{2}-\frac{\sqrt{3}}{2}i$	
3	λ_2	$\omega_1^3 = exp\left(\frac{2\pi*3i}{3}\right)$	$\omega^3=1$	
4	λ_5	$\omega_2^1 = exp\left(\frac{2\pi*1i}{2}\right)$	$\omega=-1$	The 2 roots of unity
5	λ_7	$\omega_2^2 = exp\left(\frac{2\pi*2i}{2}\right)$	$\omega^2=1$	
6	λ_8	N/A	0	These are vertices associated to chains
7	λ_6	N/A	0	
8	λ_3	N/A	0	
9	λ_{11}	N/A	0	
10	λ_{10}	N/A	0	
11	λ_9	N/A	0	

2.1 Eigenvectors

The eigenvectors (generalized eigenvectors) are computed according to theorem (6) of [9]. The eigenvectors associated to the vertices of one cycle are

$$v_k = \sum_{j=1}^{\ell}\omega^{-kj}e_{z_j} \quad (3)$$

In order to relate the above equation to the eigenvalues pertaining to the cycle, re-express it as:

$$v_k = \sum_{h=1}^{\ell_j}\left(\frac{1}{\lambda_k}\right)^h e_{z_h} \quad (4)$$

where j is associated to the cycle containing vertex v_k, and h is the vertices counter of the cycle as mentioned in Eq. (2).

Next generate the eigenvectors associated to chains. For chain $C=\{c_1,...,c_\ell\}$ two cases can arise:

1. c_ℓ is a terminal point of the graph i.e., $f(c_\ell)=0$ or $f(c_\ell)>n$: here the chain is flipped such that each vertex c_k is attached to generalized eigenvector equals e_{c_k}, $1\le k\le l$ i.e. $\{e_{c_l},\ldots,e_{c_1}\}$ is a chain of generalized eigenvectors of A_n pertaining to an eigenvalue of 0.
2. $f(c_l)$ is a merge point on a cycle or a chain: let z be the vertex on a cycle or a chain containing $f(c_l)$ such that $f^m(z)=f(c_l)$. Then $\{e_{c_l}-e_{f^{m-1}(z)},\ \ldots,\ e_{c_3}-e_{f^2(z)},\ e_{c_2}-e_{f(z)},e_{c_1}-e_z\}$ is a chain of generalized eigenvector attached to an eigenvalue of 0.

Note: By convention, the first element of a chain of generalized eigenvectors is the eigenvector and the remaining ones are generalized eigenvectors, but the eigenvector corresponds to the last element of the chain of vertices $\{c_1,...,\ c_m\}$ in Lemma 10 of [9]. Hence, the order of indices in the subscripts is reversed. The eigenvectors are listed in Table 2.

In MATLAB, the developed codes "jeig" and "jeigv" compute the eigenvalues and eigenvectors respectively. The generalized eigenvectors (including the eigenvectors) form the columns of a matrix called the "Modal matrix (Q)".

Table 2. Example 1 eigenvectors.

Vertex	Eigen-vectors (v_k)	Substitution in Eq. (4)	The corresponding eigenvalue	Notes
1	v_1	$v_1=\left(\frac{1}{\omega_1^1}\right)^1e_1+\left(\frac{1}{\omega_1^1}\right)^2e_4+\left(\frac{1}{\omega_1^1}\right)^3e_2$ $=\left(\omega_1^1\right)^2e_1+\omega_1^1e_4+e_2$ $v_1=\left(-\frac{1}{2}-\frac{\sqrt{3}}{2}i\right)e_1+\left(-\frac{1}{2}+\frac{\sqrt{3}}{2}i\right)e_4+e_2$	$\omega_1^1=-\frac{1}{2}+\frac{\sqrt{3}}{2}i$	Those are the 3 roots of unity
4	v_2	$v_2=\left(\frac{1}{\omega_1^2}\right)^1e_1+\left(\frac{1}{\omega_1^2}\right)^2e_4+\left(\frac{1}{\omega_1^2}\right)^3e_2$ $=\omega_1^1e_1+\left(\omega_1^1\right)^2e_4+e_2.$ $v_2=\left(-\frac{1}{2}+\frac{\sqrt{3}}{2}i\right)e_1+\left(-\frac{1}{2}-\frac{\sqrt{3}}{2}i\right)e_4+e_2$	$\omega_1^2=-\frac{1}{2}-\frac{\sqrt{3}}{2}i$	
2	v_3	$v_3=\left(\frac{1}{\omega_1^3}\right)^1e_1+\left(\frac{1}{\omega_1^3}\right)^2e_4+\left(\frac{1}{\omega_1^3}\right)^3e_2$ $=e_1+e_4+e_2$	$\omega_1^3=1.$	
5	v_4	$v_4=\left(\frac{1}{\omega_2^1}\right)^1e_5+\left(\frac{1}{\omega_2^1}\right)^2e_7=-e_5+e_7$	$\omega_2^1=-1$	Those are the 2 roots of unity
7	v_5	$v_5=\left(\frac{1}{\omega_2^2}\right)^1e_5+\left(\frac{1}{\omega_2^2}\right)^2e_7$ $=e_5+e_7$	$\omega_2^2=1$	
3	v_6	$v_6=e_3-e_4$	0	These are vertices associated to chains
10	v_7	$v_7=e_{10}-e_1$	0	
11	v_8	$v_8=e_{11}-e_2$	0	
6	v_9	$v_9=e_6-e_{10}$	0	
8	v_{10}	$v_{10}=e_8-e_{11}$	0	
9	v_{11}	$v_{11}=e_9$	0	

$$Q=\begin{bmatrix} -\frac{1}{2}-\frac{\sqrt{3}}{2} & -\frac{1}{2}+\frac{\sqrt{3}}{2} & 1 & 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 \\ 1 & 1 & 1 & 0 & 0 & 0 & 0 & -1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ -\frac{1}{2}+\frac{\sqrt{3}}{2} & -\frac{1}{2}-\frac{\sqrt{3}}{2} & 1 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & -1 & 0 \end{bmatrix}$$

Finally, the Jordan form can be calculated from:

$$J=Q^{-1}AQ=\begin{bmatrix} -\frac{1}{2}+\frac{\sqrt{3}}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & -\frac{1}{2}-\frac{\sqrt{3}}{2} & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

2. COMPUTATIONAL ANALYSIS

An extensive work is conducted expanding that reported in [60]. A set of MATLAB functions - forming an experimental toolbox - is developed in order to obtain the components of the digraph associated with the zero-one matrix under investigation.

Vectors s and t initiation. First, the random generation of starting and ending vertices of each edge in the graph, s and t respectively, is done by the built-in MATLAB function "randi". This random selection considers the creation of two vectors, s and t, of length less than or equal the current matrix size in the investigation scope. For example, the investigation is carried out for matrices of size 10 to 1000 with increment of 10. If the current loop counter is 3 then n=30 and the length of s and t vectors is less than or equal 30. The code initiates a vector s that is of length equal 30. Next, the role of function "randi" starts to select less than or equal 20% of the starting points in s to be removed. Hence, in the case of n = 30: 6, 5, ...,1 or 0 starting points with random vertices numbers are removed. This procedure ensures that the starting points have no duplicates in order to get vertices with outdegree equal 0 or 1; however, the indegree of any vertex can be greater than or equal to 1.

Construction of digraph. Each vertex must have an outdegree of 1 or 0 due to the properties of the directed graphs. The final step in creating a graph is implementing the MATLAB built-in function "digraph" by using the two aforementioned vectors s and t. The output of "digraph" is a graph struct (structure) that is called G in Figure 3. The graph struct is a workspace output that includes all information required about the graph as node list, edge list, in and out degrees of each node, etc.

Finding the proper partition. Next, the proper partition of the digraph is obtained by two MATLAB functions called "allcycles" and "findchains". The "allcycles" function works on finding and saving all the closed paths in the graph structure by scanning the s and t vectors where the cycle is considered a chain of edges that starts and ends with the same vertex. Then, the "findchains" function starts from the output of "allcycles" function which are the cycles of the proper partition (CY). After that, the cycles are trimmed out of the graph leaving the remaining edges that form the chains. The extraction of the chains is done by getting all vertices that have indegree = 0. Then scanning the following vertices of each vertex with indegree = 0, if the outdegree of any of the following vertices is 0, this means a chain is located that include the starting vertex with indegree =0 and the located vertex with outdegree = 0. The contributed function "findchains" can distinguish the longest chain when chains intersect at a merge point by means of a function called "setdiff" with option 'stable' that trims the parts of chains that are repeated in a longer chain. Finally, the chains (CH) are sorted in descending order of their lengths by "sort" function.

Finding the eigenvalues. The cycles and chains are forwarded to a new function called "jeig" that calculates the eigenvalues. The eigenvalues of the vertices associated to cycles are calculated for each cycle one by one. If the cycle has only one vertex, the eigenvalue of that vertex becomes 1. Otherwise, if the cycle has two or more vertices, one computes the primitive ℓ_j root of unity is $\omega_j=\exp\left(\frac{2\pi i}{\ell_j}\right)$ where ℓ_j is the length of the cycle. Consequently, the eigenvalues associated to the vertices starting from the first vertex are $\omega,\omega^2,.....,\omega^{\ell}$ where ℓ is the number of vertices in the cycle of a proper partition of the graph [9]. Hence, the eigenvalues are:

$$\lambda_j^h=\exp\left(\frac{2\pi i}{\ell_j}h\right)=\omega_j^h \quad ,1\le h\le \ell_j \quad \textbf{(5)}$$

Finally, the eigenvalues of the vertices associated to chains are set to 0.

Finding the eigenvectors and generalized eigenvectors. The obtained eigenvalues are forwarded to the eigenvectors function called "jeigv" in order to calculate the modal matrix "Q". In this stage, four cases of eigenvectors are considered. The first case is that of eigenvectors associated with cycles vertices V_{CY}. The eigenvectors of cycles vertices are calculated according to Eq. (4). For the first cycle of the graph, the number of vertices in the cycle is ℓ. Hence, the procedure followed in the MATLAB function "jeigv" is to reserve ℓ columns in the empty modal matrix. Also, each column has ℓ row locations associated with the cycle vertices. For example, a cycle c = {1, 2, 4}, ℓ =3, the first three columns in "Q" are reserved for this cycle and the row locations 1, 2 and 4 in each column are addressed. The MATLAB code "jeigv" addresses the vertices of cycles first; hence, the modal matrix is filled with eigenvectors of cycles vertices first, and the eigenvectors of chains vertices second. Before elaborating on the chains' cases, the code runs chains in descending order in order to classify them according to the following three categories by employing conditional checks using "ismember" condition of MATLAB. The modal matrix is filled according to the order of the chains obtained in the proper partition.

1. The second case (first case in chains) is that of eigenvectors of chains ending with terminal points V_{CH1}. If the last vertex is a terminal point (Check1=ismember(last vertex, terminal points)=1), the chain is classified as ending with terminal. Assuming the length of this chain is ℓ, the modal matrix reserves ℓ columns for the eigenvectors of this chain. In each column, the location number of each vertex is given to the row of this column. For example, chain ch={11, 10, 3}, length ℓ = 3, the row locations in the three columns are 3, 10 and 11 as the chain is flipped. Finally, the eigenvectors are equal 1 at these locations.
2. The third case (second chain case) is the chain ending with merge point on a chain V_{CH2}. If the second check (Check2=ismember(last vertex, Merge points)=1), and the sub-check (Check21=ismember(last vertex, any chain)=1), then the chain is classified as a chain ending with a merge point on a chain. The procedure of Section 2.2 is followed. Therefore, if the length of the chain is ℓ, the modal matrix will have a reservation for ℓ columns. Each column has two values (1, -1) as illustrated in Section 2.2.
3. The last case (third chain case) is the chain ending with merge point on a cycle V_{CH3}. If the second check (Check2=ismember(last vertex, Merge points)=1), and the sub-check (Check22=ismember(last vertex, any cycle)=1), then the chain is classified as a chain ending with a merge point on a cycle. In this case, two sub-cases are introduced. The first subcase is that the length of the chain ℓ is less than the length of the merge cycle. Then, the cycle is considered as another chain, and the eigenvectors are calculated as illustrated in Section 2 Also, the modal matrix will have new reservation for ℓ columns. Each column has two values (1, -1) associated to the calculated eigenvectors. The second subcase is the length of the chain ℓ is equal or greater than the length of the merge cycle. The merge cycle is considered as another chain that repeats until reaching the length of the chain ℓ. For example, if ℓ =5, and the merge cycle c = {3,6,10}, the merge cycle is modified to be a chain that equal {3,6,10,3,6} in order to calculate the eigenvectors as explained in Section 2.

Figure 3 illustrates the flowchart of the contributed MATLAB toolbox in detail. The inputs to the toolbox are the starting and ending points of each edge of the digraph, s and t respectively. The outputs of the toolbox are the modal matrix "Q" and the Jordan form "J". The "jeigv" code treats all cycles' and chains' cases according to the explanation in [9]. The modal matrix is obtained at the bottom after the "jeigv" block. The following eigenvectors are computed: V_{CY} (the eigenvectors pertaining to cycles), V_{CH1} (the generalized eigenvectors pertaining to chains ending with a terminal point), V_{CH2} (the generalized eigenvectors pertaining to chains having an end point on another chain) and V_{CH3} (the generalized eigenvectors pertaining to chains having an end point on a cycle). Hence V_{CH1}, V_{CH2} and V_{CH3} are the generalized eigenvectors pertaining to chains ending on a merge point. Finally, the modal matrix Q is generated and the Jordan blocks are obtained by the block before J in Figure 3. where adjacency(G) is the built-in MATLAB function that extracts the graph edge list from the graph struct in order to create an adjacency matrix of zeros and ones according to the relationship between all vertices in the extracted edge list table.

The reason for developing such toolbox is that the built-in MATLAB code "jordan" lingers for square matrices of order between 40 and 60. The contributed toolbox and the MATLAB built-in function "jordan" are tested in a loop of matrix order [10:10:1000] by generating random digraphs. The random generation is done by the MATLAB function "randi". It is found that the "jordan" function halts for matrix order of 40, 50 or 60 every time the test is performed.

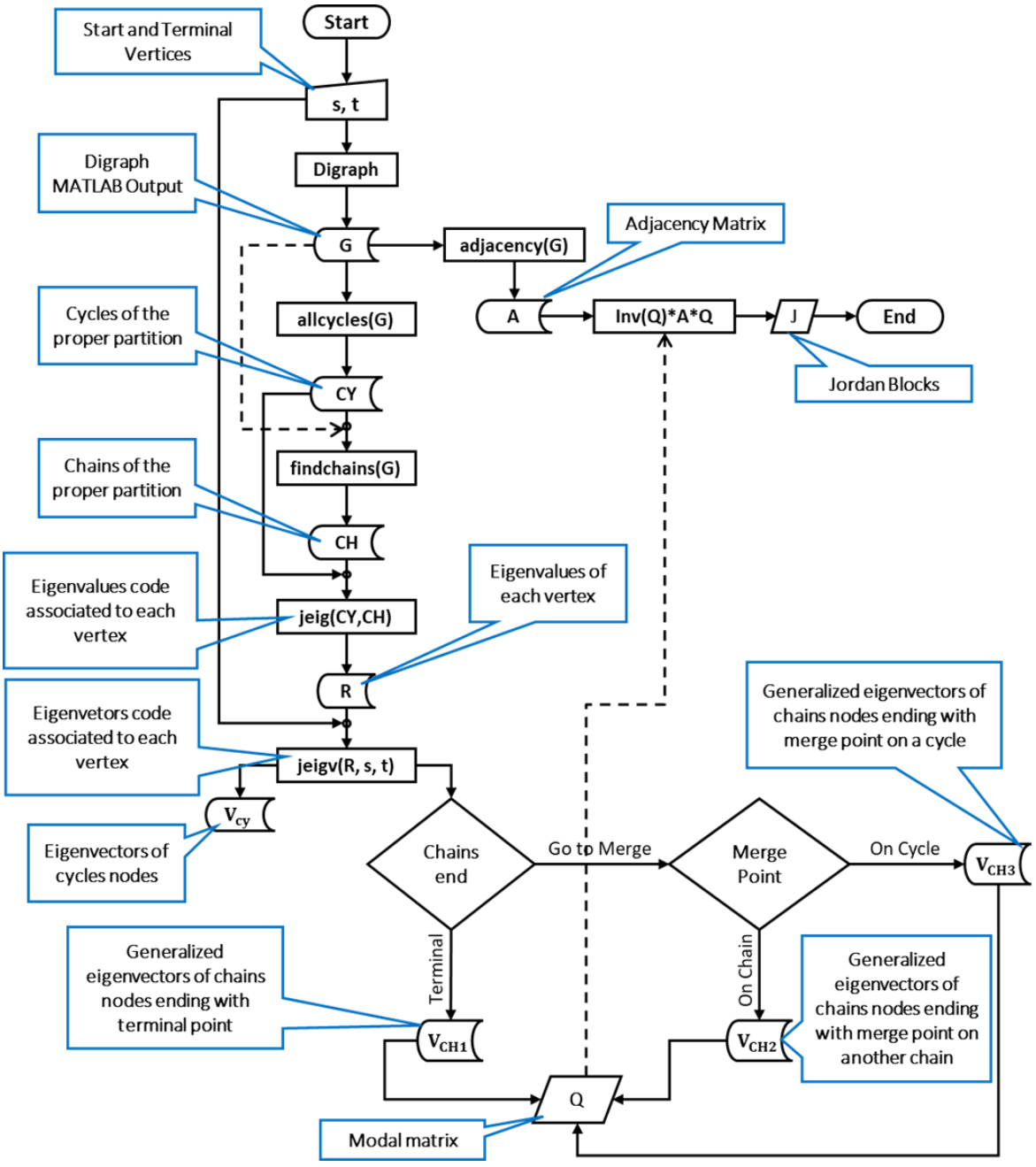


Figure 3: Flowchart.

In order to assess the accuracy of the computational procedure, the adjacency matrix is computed according to Eq.(6). The absolute value of the difference between the computed adjacency matrix and the actual one is calculated elementwise by Eq.(7). Two measures of the error matrix $\mathbf{E}$ are computed; namely, the maximum error (defined as the maximum element) of Eq. code) **(8)** and the Frobenius error (defined as the Frobenius matrix norm) of Eq.(9).

$$\mathbf{A}_{computed} = \mathbf{QJQ}^{-1} \quad \textbf{(6)}$$

$$\mathbf{E} = \left| \mathbf{A}_{actual} - \mathbf{A}_{computed} \right| \quad \textbf{(7)}$$

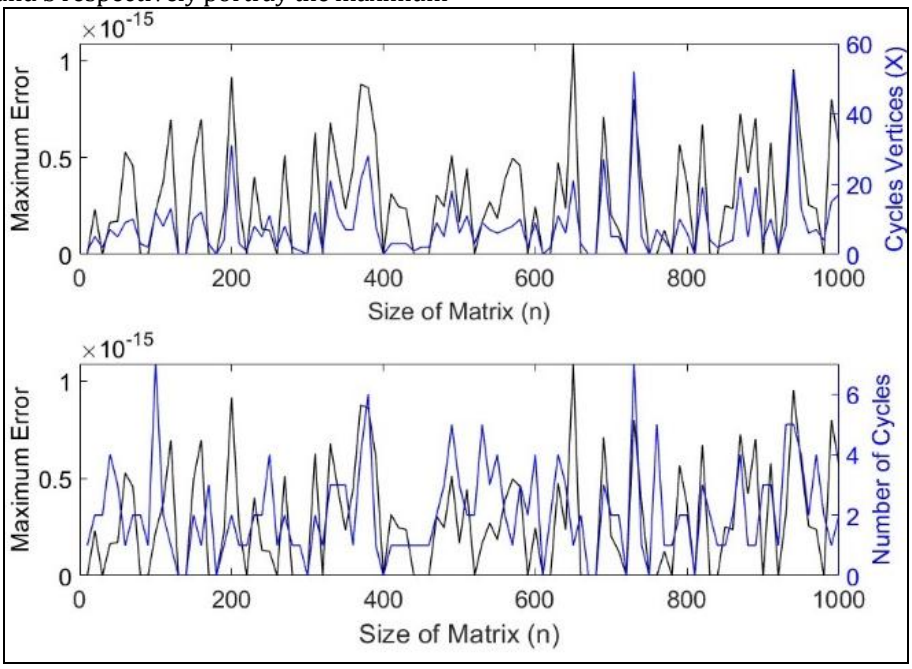
$$E_{\max} = \max(\max(\mathbf{E})) \text{ (a MATLAB code)} \quad \textbf{(8)}$$

$$E_{fro} = \sqrt{\sum_{i=1}^{m}\sum_{j=1}^{n}\left|E_{ij}\right|^2} \quad \textbf{(9)}$$

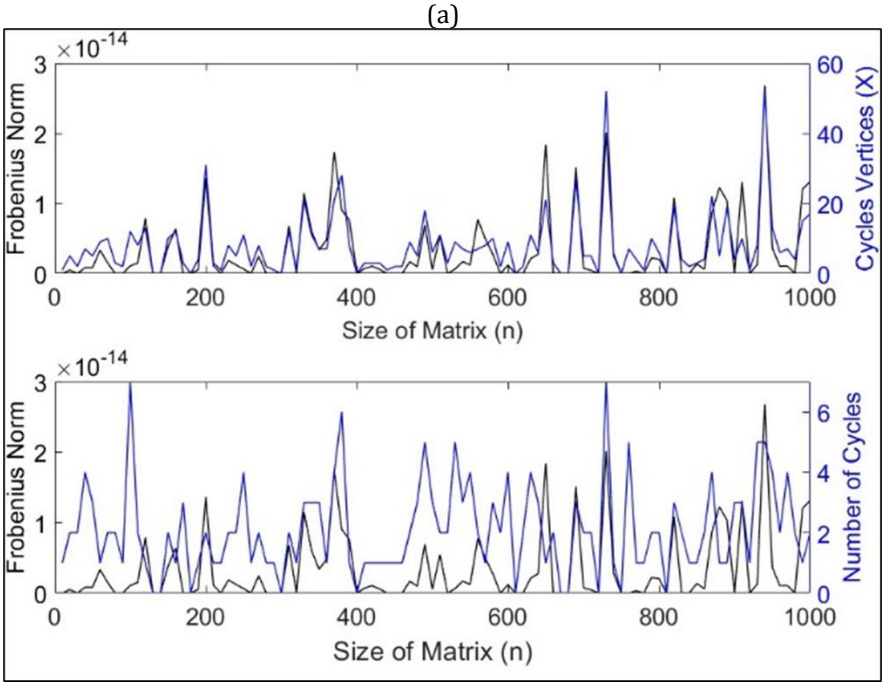
3. Computational Results

Aiming at finding a relation between the error measures given by code) **(8)** and (9) and some characterizing features of the graph, one starts by defining the cycles vertices (X) as

the number of vertices involved in all the cycles of the graph. Since a graph can be characterized by both X and the number of its cycles, one looks for any correlation between the error measures on one hand and X and the number of cycles in the graph on the other hand. Figure 4 depicts the results of two tests where parts a and b respectively portray the maximum and Frobenius error measures. A quick inspection shows that the correlation between the error measures and X is more pronounced than that between the error measures and the number of cycles. Moreover, the Frobenius error shows a stronger correlation with X than the maximum error does.

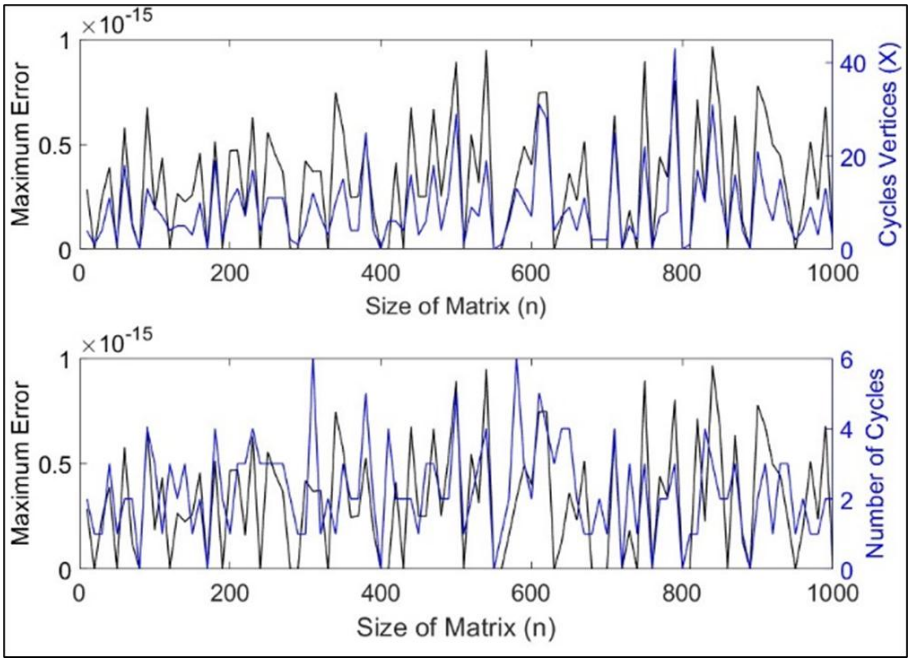


(a)

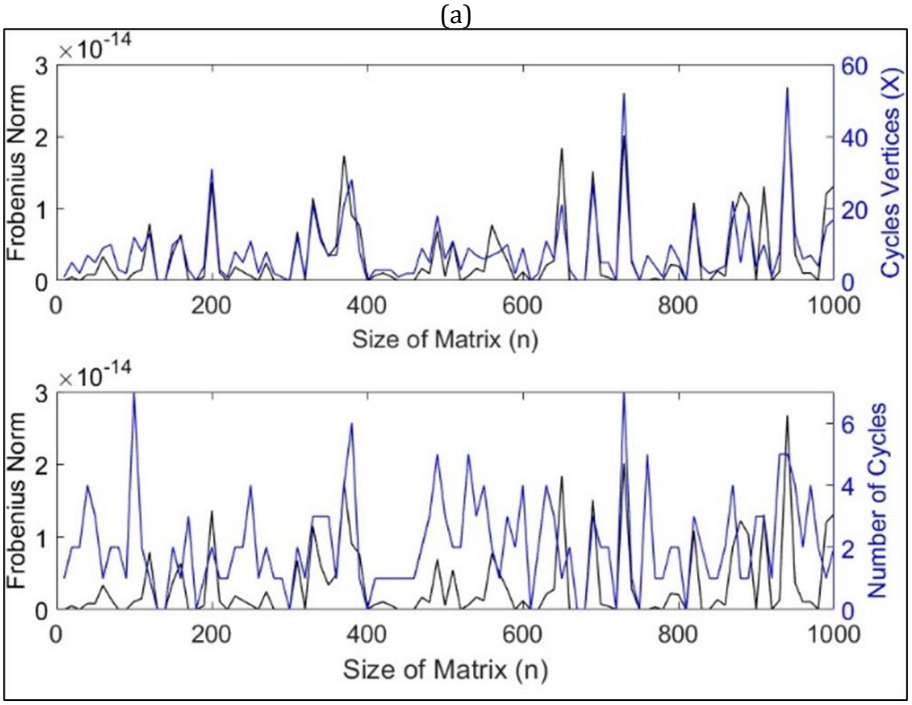


(b)

Figure 4. The error measures, the X and number of cycles versus matrix size (n) in test 1 (a) maximum error, (b) Frobenius error.



(a)



(b)

Figure 5. The error measures, the X and number of cycles versus matrix size (n) in test 2 (a) maximum error, (b) Frobenius error.

During performing the simulation, the following observations were recorded:

1. X is not totally the main factor affecting the error measures.
2. The longer the cycles, especially as an odd number, the greater the error.
3. When the number of cycles is large while their lengths are very short (particularly 1, 2, 4 and 8), the error shrinks almost to zero.
4. The presence of large imaginary parts in the modal matrix leads to large error.

4. CONCLUSIONS

Based on Cardon's approach[9], an experimental MATLAB toolbox is developed to construct the Jordan canonical form for a class of zero-one square matrices of order up to 1000, with the additional property that each column has at most

one nonzero element. The method is based on the creation and examination of a directed graph with the matrix under discussion as its adjacency matrix. each column has at most one nonzero element based on Cardon's algorithm [9]. This investigation focuses primarily on computing the eigenvalues and generalized eigenvectors to construct the Jordan canonical form and modal matrix. The computation's accuracy is measured by calculating the difference between the given matrix and the developed one obtained by combining the Jordan form and the modal matrix.

DECLARATION

This work has been supported by Science, Technology & Innovation Funding Authority (STIFA), Egypt under Grant Number 37084 Basic and Applied Research

REFERENCES

[1] X. Gong, Y. Cui, T. Wang, J. Shen, and T. Huang, "Distributed Prescribed-Time Consensus Observer for High-Order Integrator Multi-Agent Systems on Directed Graphs," *IEEE Trans. Circuits Syst. II*, vol. 69, no. 4, pp. 2216–2220, Apr. 2022, doi: 10.1109/TCSII.2021.3127358.

[2] J. Yang, G. Cao, Z. Sun, and W. Zhang, "A New Method to Design Distributed Consensus Controller for Linear Multi-Agent Systems With Directed Graphs," *IEEE Trans. Circuits Syst. II*, vol. 69, no. 3, pp. 1492–1496, Mar. 2022, doi: 10.1109/TCSII.2021.3120869.

[3] I. Ahmed, M. Rehan, and N. Iqbal, "A Novel Exponential Approach for Dynamic Event-Triggered Leaderless Consensus of Nonlinear Multi-Agent Systems Over Directed Graphs," *IEEE Trans. Circuits Syst. II*, vol. 69, no. 3, pp. 1782–1786, Mar. 2022, doi: 10.1109/TCSII.2021.3120791.

[4] M. Yu and X. Cheng, "An algorithm for the Jordan canonical form and the transition matrix," in *2016 IEEE International Conference of Online Analysis and Computing Science (ICOACS)*, Chongqing, China: IEEE, May 2016, pp. 235–238. doi: 10.1109/ICOACS.2016.7563086.

[5] T. Suzuki and T. Suzuki, "Computing the Jordan canonical form in finite precision arithmetic," in *12th GAMM - IMACS International Symposium on Scientific Computing, Computer Arithmetic and Validated Numerics (SCAN 2006)*, Duisburg, Germany: IEEE, Sep. 2006, pp. 39–39. doi: 10.1109/SCAN.2006.13.

[6] T. Xu, G. Lv, Z. Duan, Z. Sun, and J. Yu, "Distributed Fixed-Time Triggering-Based Containment Control for Networked Nonlinear Agents Under Directed Graphs," *IEEE Trans. Circuits Syst. I*, vol. 67, no. 10, pp. 3541–3552, Oct. 2020, doi: 10.1109/TCSI.2020.2991101.

[7] T. Xu, Y. Hao, and Z. Duan, "Fully Distributed Containment Control for Multiple Euler-Lagrange Systems Over Directed Graphs: An Event-Triggered Approach," *IEEE Trans. Circuits Syst. I*, vol. 67, no. 6, pp. 2078–2090, Jun. 2020, doi: 10.1109/TCSI.2020.2971037.

[8] H. Su, B. Cheng, and Z. Li, "Fully Distributed Event-Based Protocols for Lur'e Systems Over Directed Graphs," *IEEE Trans. Circuits Syst. II*, vol. 69, no. 3, pp. 1812–1816, Mar. 2022, doi: 10.1109/TCSII.2021.3128740.

[9] D. A. Cardon and B. Tuckfield, "The Jordan canonical form for a class of zero–one matrices," *Linear Algebra and its Applications*, vol. 435, no. 11, pp. 2942–2954, Dec. 2011, doi: 10.1016/j.laa.2011.05.022.

[10] R. J. Wilson, *Introduction to graph theory*, 3rd ed., Repr. with minor corrections. Burnt Mill, Harlow, Essex, England : New York: Longman Scientific & Technical ; Wiley, 1985.

[11] E. W. Weisstein, "Königsberg Bridge Problem," *MathWorld-A Wolfram Web Resource*. [Online]. Available: https://mathworld.wolfram.com/KoenigsbergBridgeProblem.html

[12] P. Boguslawski, "Modelling and analysing 3D building interiors with the dual half-edge data structure (Thesis)." Jan. 2011. [Online]. Available: https://www.researchgate.net/publication/265219734_Modelling_and_analysing_3D_building_interiors_with_the_dual_half-edge_data_structure

[13] A. Ruan, W. Li, C. Xiang, J. Song, S. Kang, and Y. Liao, "Graph theory for FPGA minimum configurations," *J. Semicond.*, vol. 32, no. 11, p. 115018, Nov. 2011, doi: 10.1088/1674-4926/32/11/115018.

[14] A. Broder *et al.*, "Graph structure in the Web," *Computer Networks*, vol. 33, no. 1–6, pp. 309–320, Jun. 2000, doi: 10.1016/S1389-1286(00)00083-9.

[15] F. Riaz and K. M. Ali, "Applications of Graph Theory in Computer Science," in *2011 Third International Conference on Computational Intelligence, Communication Systems and Networks*, Bali, Indonesia: IEEE, Jul. 2011, pp. 142–145. doi: 10.1109/CICSyN.2011.40.

[16] G. Narayanan Kannaiyan, B. Pappula, and R. Veerubommu, "RETRACTED: A Review on Graph Theory in Network and Artificial Intelligence," *J.*

Phys.: Conf. Ser., vol. 1831, no. 1, p. 012002, Mar. 2021, doi: 10.1088/1742-6596/1831/1/012002.

[17] P. Verma, S. Nagarajan, and A. Raj, "Spectral graph theory of brain oscillations—-Revisited and improved," *NeuroImage*, vol. 249, p. 118919, Apr. 2022, doi: 10.1016/j.neuroimage.2022.118919.

[18] H. Shokouh Alaei, M. Ghoshuni, and I. Vosough, "Directed brain network analysis in anxious and non-anxious depression based on EEG source reconstruction and graph theory," *Biomedical Signal Processing and Control*, vol. 83, p. 104666, May 2023, doi: 10.1016/j.bspc.2023.104666.

[19] C. J. Stam and J. C. Reijneveld, "Graph theoretical analysis of complex networks in the brain," *Nonlinear Biomed Phys*, vol. 1, no. 1, p. 3, Dec. 2007, doi: 10.1186/1753-4631-1-3.

[20] F. Abdul Razak, F. Ahmad Shahabuddin, and N. Sarah Nik Zamri, "Analyzing research collaborations within the School of Mathematical Sciences, UKM using Graph Theory," *J. Phys.: Conf. Ser.*, vol. 1212, p. 012033, Apr. 2019, doi: 10.1088/1742-6596/1212/1/012033.

[21] K. F. Kantelis *et al.*, "Graph theory-based simulation tools for protein structure networks," *Simulation Modelling Practice and Theory*, vol. 121, p. 102640, Dec. 2022, doi: 10.1016/j.simpat.2022.102640.

[22] J. delEtoile and H. Adeli, "Graph Theory and Brain Connectivity in Alzheimer's Disease," *Neuroscientist*, vol. 23, no. 6, pp. 616–626, Dec. 2017, doi: 10.1177/1073858417702621.

[23] D. J. Sanderson, D. C. P. Peacock, C. W. Nixon, and A. Rotevatn, "Graph theory and the analysis of fracture networks," *Journal of Structural Geology*, vol. 125, pp. 155–165, Aug. 2019, doi: 10.1016/j.jsg.2018.04.011.

[24] E. Durand-Cartagena, J. Soria, and P. Tradacete, "Doubling constants and spectral theory on graphs," 2021, doi: 10.48550/ARXIV.2111.09199.

[25] S. Heidari Masteali, P. Bettinger, M. Bayat, B. Jabbarian Amiri, and H. Umair Masood Awan, "Comparison between graph theory connectivity indices and landscape connectivity metrics for modeling river water quality in the southern Caspian sea basin," *Journal of Environmental Management*, vol. 328, p. 116965, Feb. 2023, doi: 10.1016/j.jenvman.2022.116965.

[26] S. Yang, Z. Zhang, Y. Wang, R. Liang, Y. Wan, and K. Li, "An improved 3D fish habitat assessment model based on the graph theory algorithm," *Ecological Indicators*, vol. 148, p. 110022, Apr. 2023, doi: 10.1016/j.ecolind.2023.110022.

[27] A. Ru, "AN APPLICATION OF GRAPH THEORY IN FABRIC DESIGN," *Textile and Apparel*, vol. 5, no. 2, pp. 41–45, May 2001, doi: 10.1108/RJTA-05-02-2001-B004.

[28] C. Hatlestad-Hall *et al.*, "Reliable evaluation of functional connectivity and graph theory measures in source-level EEG: How many electrodes are enough?," *Clinical Neurophysiology*, vol. 150, pp. 1–16, Jun. 2023, doi: 10.1016/j.clinph.2023.03.002.

[29] L. Nabulsi *et al.*, "Aberrant Subnetwork and Hub Dysconnectivity in Adult Bipolar Disorder: A Multicenter Graph Theory Analysis," *Cerebral Cortex*, vol. 32, no. 10, pp. 2254–2264, May 2022, doi: 10.1093/cercor/bhab356.

[30] D. A. Vecchio, S. H. Mahler, M. D. Hammig, and N. A. Kotov, "Structural Analysis of Nanoscale Network Materials Using Graph Theory," *ACS Nano*, vol. 15, no. 8, pp. 12847–12859, Aug. 2021, doi: 10.1021/acsnano.1c04711.

[31] W. R. Khalifa and T. H. Jasim, "Some new concepts of nano topological space via graph theory," presented at the INTERNATIONAL CONFERENCE OF COMPUTATIONAL METHODS IN SCIENCES AND ENGINEERING ICCMSE 2021, Heraklion, Greece, 2023, p. 040058. doi: 10.1063/5.0134716.

[32] Z. Saad-Saoud and A. C. Williamson, "Simulation of electromagnetic drive systems using graph theory," *IEE Proc., Electr. Power Appl.*, vol. 145, no. 4, p. 393, 1998, doi: 10.1049/ip-epa:19981837.

[33] L. Toscano, S. Stella, and E. Milotti, "Using graph theory for automated electric circuit solving," *Eur. J. Phys.*, vol. 36, no. 3, p. 035015, May 2015, doi: 10.1088/0143-0807/36/3/035015.

[34] D. Krleza and K. Fertalj, "Graph Matching Using Hierarchical Fuzzy Graph Neural Networks," *IEEE Trans. Fuzzy Syst.*, vol. 25, no. 4, pp. 892–904, Aug. 2017, doi: 10.1109/TFUZZ.2016.2586962.

[35] A. Said, M. Shabbir, B. Broll, W. Abbas, P. Völgyesi, and X. Koutsoukos, "Circuit design completion using graph neural networks," *Neural Comput & Applic*, Feb. 2023, doi: 10.1007/s00521-023-08346-x.

[36] D. E. Nurvazly and K. A. Sugeng, "Graceful Labelling of Edge Amalgamation of Cycle Graph," *J. Phys.: Conf. Ser.*, vol. 1108, p. 012047, Nov. 2018, doi: 10.1088/1742-6596/1108/1/012047.

[37] P. Lalitha and L. Tamilselvi, "Analysis of Psychology of Students by Graphical Representation," *J. Phys.:*

Conf. Ser., vol. 1377, no. 1, p. 012029, Nov. 2019, doi: 10.1088/1742-6596/1377/1/012029.

[38] G. Uma Maheswari, M. S. Umamaheswari, and R. Muthusamy, "A Method of Descriptions for Graph Labelings," *J. Phys.: Conf. Ser.*, vol. 1706, no. 1, p. 012046, Dec. 2020, doi: 10.1088/1742-6596/1706/1/012046.

[39] E. McCafferty, "Graph Theory and the Passivity of Binary Alloys," *J. Electrochem. Soc.*, vol. 151, no. 2, p. B82, 2004, doi: 10.1149/1.1637899.

[40] F. P. Kalaganis, N. A. Laskaris, V. P. Oikonomou, S. Nikopolopoulos, and I. Kompatsiaris, "Revisiting Riemannian geometry-based EEG decoding through approximate joint diagonalization," *J. Neural Eng.*, vol. 19, no. 6, p. 066030, Dec. 2022, doi: 10.1088/1741-2552/aca4fc.

[41] S. Hosouli, J. Elvins, J. Searle, S. Boudjabeur, J. Bowyer, and E. Jewell, "A Multi-Criteria decision making (MCDM) methodology for high temperature thermochemical storage material selection using graph theory and matrix approach," *Materials & Design*, vol. 227, p. 111685, Mar. 2023, doi: 10.1016/j.matdes.2023.111685.

[42] J. Lanzone *et al.*, "Vagal nerve stimulation cycles alter EEG connectivity in drug-resistant epileptic patients: A study with graph theory metrics," *Clinical Neurophysiology*, vol. 142, pp. 59–67, Oct. 2022, doi: 10.1016/j.clinph.2022.07.503.

[43] N. Mokrov, M. Panov, B. A. Gutman, J. I. Faskowitz, N. Jahanshad, and P. M. Thompson, "Simultaneous Matrix Diagonalization for Structural Brain Networks Classification," in *Complex Networks & Their Applications VI*, C. Cherifi, H. Cherifi, M. Karsai, and M. Musolesi, Eds., in Studies in Computational Intelligence, vol. 689. Cham: Springer International Publishing, 2018, pp. 1261–1270. doi: 10.1007/978-3-319-72150-7_102.

[44] C. Chen, X. Zhang, H. Zhang, Y. Cai, and S. Wang, "Managing water-energy-carbon nexus in integrated regional water network planning through graph theory-based bi-level programming," *Applied Energy*, vol. 328, p. 120178, Dec. 2022, doi: 10.1016/j.apenergy.2022.120178.

[45] J.-H. Choi, H. Lee, H. R. Choi, and M. Cho, "Graph Theory and Ion and Molecular Aggregation in Aqueous Solutions," *Annu. Rev. Phys. Chem.*, vol. 69, no. 1, pp. 125–149, Apr. 2018, doi: 10.1146/annurev-physchem-050317-020915.

[46] M. C. Litwińczuk, N. Muhlert, N. Trujillo-Barreto, and A. Woollams, "Using graph theory as a common language to combine neural structure and function in models of healthy cognitive performance," *Human Brain Mapping*, p. hbm.26258, Mar. 2023, doi: 10.1002/hbm.26258.

[47] J. Ji *et al.*, "Revealing Density Thresholds of Carbon Nanotube Cross-Links for Load Transfer: A Graph Theory Strategy," *ACS Nano*, vol. 16, no. 4, pp. 6929–6936, Apr. 2022, doi: 10.1021/acsnano.2c03003.

[48] L. W. Beineke and R. J. Wilson, Eds., *Topics in algebraic graph theory*. in Encyclopedia of mathematics and its applications, no. v. 102. Cambridge, UK ; New York: Cambridge University Press, 2004.

[49] J. Lee, Y. Kim, W. Y. Kim, and H. B. Oh, "Graph theory-based reaction pathway searches and DFT calculations for the mechanism studies of free radical-initiated peptide sequencing mass spectrometry (FRIPS MS): a model gas-phase reaction of GGR tri-peptide," *Phys. Chem. Chem. Phys.*, vol. 22, no. 9, pp. 5057–5069, 2020, doi: 10.1039/C9CP05433B.

[50] A. Suyitno, H. Suyitno, Rochmad, and Dwijanto, "Graph theory as a tool to track the growth of student's mathematical creativity," *J. Phys.: Conf. Ser.*, vol. 1321, no. 3, p. 032119, Oct. 2019, doi: 10.1088/1742-6596/1321/3/032119.

[51] I. Q. Abduljaleel, S. A. Abdul-Ghani, and H. Z. Naji, "An Image of Encryption Algorithm Using Graph Theory and Speech Signal Key Generation," *J. Phys.: Conf. Ser.*, vol. 1804, no. 1, p. 012005, Feb. 2021, doi: 10.1088/1742-6596/1804/1/012005.

[52] Y. Ding *et al.*, "A graph-theory-based dynamic programming planning method for distributed energy system planning: Campus area as a case study," *Applied Energy*, vol. 329, p. 120258, Jan. 2023, doi: 10.1016/j.apenergy.2022.120258.

[53] M. Franz *et al.*, "Cytoscape.js 2023 update: a graph theory library for visualization and analysis," *Bioinformatics*, vol. 39, no. 1, p. btad031, Jan. 2023, doi: 10.1093/bioinformatics/btad031.

[54] P. Sembiring, U. Sinulingga, M. Situmorang, and S. Sembiring, "Representative Model the Graph Theory in Calculations Kendall Correlation Coefficient," *J. Phys.: Conf. Ser.*, vol. 930, p. 012040, Dec. 2017, doi: 10.1088/1742-6596/930/1/012040.

[55] Y. Pan *et al.*, "Abnormal network properties and fiber connections of DMN across major mental disorders: a probability tracing and graph theory

study," *Cerebral Cortex*, vol. 32, no. 15, pp. 3127–3136, Jul. 2022, doi: 10.1093/cercor/bhab405.

[56] R. B. Bapat, *Graphs and Matrices*. London: Springer London, 2010. doi: 10.1007/978-1-84882-981-7.

[57] J. L. Gross, J. Yellen, and M. Anderson, *Graph Theory and Its Applications*, 3rd ed. Chapman and Hall/CRC, 2018. doi: 10.1201/9780429425134.

[58] P. Paramadevan and S. Sotheeswaran, "Properties of adjacency matrix of a graph and it's construction," *J. Sc.*, vol. 12, no. 1, p. 13, Aug. 2021, doi: 10.4038/jsc.v12i1.29.

[59] R. A. Brualdi, "The Jordan Canonical Form: An Old Proof," *The American Mathematical Monthly*, vol. 94, no. 3, p. 257, Mar. 1987, doi: 10.2307/2323392.

[60] R. R. Ghabbour, I. H. Abdelgaliel, and M. T. Hanna, "A Directed Graph and MATLAB Generation of the Jordan Canonical Form for a Class of Zero-One Matrices," in *2022 18th International Computer Engineering Conference (ICENCO)*, Cairo, Egypt: IEEE, Dec. 2022, pp. 86–91. doi: 10.1109/ICENCO55801.2022.10032513.