

### 3-equitable prime cordial labeling : MATLAB techniques for analyzing cycle twin chord graphs

Salim Said Al Hudaify \*

Mohammad Al Natour †

*Department of Mathematics and Information Technology*

*Centre for Preparatory Studies*

*Sultan Qaboos University*

*Sultanate of Oman*

---

#### Abstract

Graph labeling methods are necessary in addressing complex matters in a Mobile Ad-Hoc Network (MANET) such as connectivity, scaling, routing, and modeling the network. We can make measurements reasonably confident by structuring networks in the form of graphs, by representing them using matrices and by considering mathematical graph theory as the foundational analytical basis. The management of the network will be automated by algorithms, whereas simulating the networks are basic elements of networks, node density, node mobility, and link formation as well as packet routing in relation to the random graph theory. The research of congestion analysis of MANETs can be analyzed using various algorithms, all of which initially dwell on the principles of the graph theory. In this paper, certain programming of MATLAB will be introduced to investigate how 3-equity prime cordial labeling acts in cycle graphs with twin chords, and hence it may clarify the intricate dynamics of a MANET.

---

**Subject Classification (2010):** 05C78.

**Keywords:** Mobile ad-hoc networks (MANETs), Twin chord cycle, Graph labelling, Prime labelling, 3-equitable graph, 3-equitable prime cordial labeling.

#### I. Introduction

Graph theory is a fundamental area of discrete mathematics that deals with the study of graphs consisting of vertices and edges. The basic concepts, terminology, and conventions adopted in this work follow the standard treatment presented by Gross and Yellen [6]. A graph provides a flexible mathematical framework for representing relationships and interactions among objects and has found extensive applications in various fields, including coding theory, X-ray

---

\* E-mail: [salimsaid@squ.edu.om](mailto:salimsaid@squ.edu.om) (Corresponding Author)

† E-mail: [alnatour@squ.edu.om](mailto:alnatour@squ.edu.om)

crystallography, radar, astronomy, circuit design, communication networks, and database management.

In the present study, we consider finite, connected, simple, and undirected graphs. The fundamental terminology and notation used throughout the investigation are based on the framework given by Gross and Yellen [6] and Harary [13]. The development of graph theory has led to numerous specialized areas of research, including graph labeling, graph energy, and numbered graphs. Bloom and Golomb [2] investigated applications of numbered undirected graphs, demonstrating the relevance of graph structures in practical applications. Balakrishnan [3] studied the energy of graphs and established important results concerning graph energy, while Balakrishnan and Ranganathan [4] provided a comprehensive treatment of the fundamental concepts of graph theory. Gallian [5] presented a detailed dynamic survey of graph labeling, highlighting the extensive development of different labeling techniques and their applications.

With the advancement of computational techniques, MATLAB has become a useful tool for constructing graphs, implementing graph algorithms, verifying mathematical results, and visualizing graph structures. Abhirami and Vikramaprasad [1] developed a MATLAB program for determining the energy of even sum cordial graphs, demonstrating the usefulness of computational methods in graph-theoretic investigations. Gupta and Srivastav [7] proposed MATLAB algorithms for obtaining new results concerning the energy of graphs. Their work demonstrates how computational techniques can complement theoretical investigations by providing efficient procedures for constructing and analyzing graph structures.

Graph labeling is another important and extensively studied area of graph theory. A graph labeling generally involves assigning integers or other mathematical objects to the vertices, edges, or both, subject to certain prescribed conditions. Lourdasamy and Patrick [14] investigated sum divisor cordial labeling for star- and ladder-related graphs, contributing to the development of labeling techniques for specific graph families. Srivastav and Gupta [15] further studied sum divisor cordial labeling of graphs and obtained additional results for different graph structures. These investigations illustrate the continuing development of graph labeling and its application to various classes of graphs.

Computational approaches have also been incorporated into the study of graph labeling. Gupta and Srivastav [12] investigated the middle graph of a cycle and its related graphs in the context of sum divisor cordial labeling and presented MATLAB-based computational results. Their work provides an effective foundation for implementing graph-labeling concepts computationally. Similarly, Gupta and Srivastav [7] demonstrated the use of MATLAB algorithms for investigating graph energy, emphasizing the role of computational tools in supporting theoretical graph research.

The study of graph parameters extends beyond graph energy and labeling to other important structural properties. Sweetly and Joseph [8] investigated the radio number of  $(W_n:2)$  graphs, while Sweetly and Joseph [9] studied the radio number of graphs of diameter three. These works demonstrate the application of

distance-based labeling concepts to different graph structures. In addition, Leel et al. [10] investigated the domination number in the context of some new graphs. Gupta, Gupta, and Srivastav [11] further studied the domination number for STAR-related graphs, demonstrating the importance of domination parameters in the investigation of newly constructed graph families.

The classical foundations of graph theory provided by Harary [13] and the systematic treatment of graph theory and its applications by Gross and Yellen [6] form the theoretical basis for the concepts considered in this study. These foundational works, together with the contributions of Bloom and Golomb [2], Balakrishnan [3], Balakrishnan and Ranganathan [4], and Gallian [5], provide the necessary mathematical background for investigating graph structures, graph parameters, and labeling techniques.

MATLAB programming provides an efficient computational environment for implementing the mathematical concepts discussed above. Chapman [16] provides a comprehensive treatment of MATLAB programming for engineering and scientific applications, which serves as a useful computational reference for developing algorithms and programs. In graph-theoretic research, MATLAB can be used to generate adjacency matrices, construct graph structures, calculate graph parameters, implement labeling algorithms, verify prescribed labeling conditions, and visualize the resulting graphs.

In the present work, the concepts and results associated with graph labeling are further explored through a MATLAB-based computational approach. In particular, the work of Lourdusamy and Patrick [14] and Srivastav and Gupta [15] provides the theoretical background for sum divisor cordial labeling, while the computational methodology presented by Gupta and Srivastav [12] motivates the development of MATLAB algorithms for implementing the labeling process. The studies of Abhirami and Vikramaprasad [1] and Gupta and Srivastav [7] further support the use of MATLAB as an effective computational tool in graph-theoretic investigations.

Thus, the present study combines the theoretical foundations of graph theory with computational techniques to investigate graph labeling and related graph parameters. The MATLAB-based approach facilitates the construction of graph structures, assignment of labels, verification of labeling conditions, and visualization of results. This integration of mathematical theory and computational implementation provides an efficient framework for studying new classes of graphs and obtaining further results in graph labeling.

### Preliminaries

**Definition 1.1:** Graph labeling is the way we assign values to vertices based on certain rules. It involves three main things:

1. Rules for giving values to edges.
2. Choosing numbers to label vertices.
3. Making sure the assigned values meet specific requirements.

**Definition 1.2:** A ternary vertex labeling of a graph  $G$  is a vertex labelling such that  $|a_g(j) - a_g(k)| \leq 1$  and  $|b_g(j) - b_g(k)| \leq 1$  for all  $0 \leq j, k \leq 2$ . A prime cordial labeling graph is referred to as a ternary graph.

**Definition 1.3:** A cycle  $C_n$  has a valid edge between two non-adjacent vertices, which is known as a chord of  $C_n$  ( $n \geq 4$ ).

**Definition 1.4:** If two chords of a cycle  $C_n$  ( $n \geq 5$ ) form a triangle with an edge of  $C_n$  these chords will be called twin chords. A cycle  $C_n$ ,  $r$  will be referred to as  $C_{n-r}$  when it has twin chords under which chords create cycles  $C_r$ ,  $C_3$  and  $C_{n+1-r}$  and which do not have chords with the edges of  $C_n$  given positive integers  $m$  and  $r$  with  $5 \leq r + 2 \leq n$ .

**Definition 1.5:** A bijection  $g: V(H) \rightarrow \{1, 2, \dots, m\}$  is a 3-equity prime cordial labelling of a graph  $H$  whose vertex set is  $V(H)$  and defined by

$$g(e = pq) = \begin{cases} 1, & \text{if } \gcd(g(p), g(q)) = 1 \text{ and} \\ & \gcd(g(p) + g(q), g(p) - g(q)) = 1 \\ 2, & \text{if } \gcd(g(p), g(q)) = 1 \text{ and} \\ & \gcd(g(p) + g(q), g(p) - g(q)) = 2 \\ 0, & \text{otherwise} \end{cases}$$

and  $|b_g(j) - b_g(k)| \leq 1$  for all  $0 \leq j, k \leq 2$ . An equitable Graph is a graph may possess equitable prime cordial labelling.

This paper has applied MATLAB programming to examine the behavior of 3-equitable prime cordial labelling on the graph  $H$  cycle on the graph of one chord.

## Results

**Theorem:** Cycle  $C_m$  with twin chord is 3-equitable prime cordial graph  $m \equiv 4 \pmod{6}$ .

**Proof:** Let  $H$  be the cycle  $C_m$  with twin chord. Let  $p_1, p_2, p_3, \dots, p_m$  be the vertices of cycle  $C_m$  and let  $e_1 = p_1 p_3$  and  $e_2 = p_1 p_4$  be the chord of cycle  $C_m$ . We define the labeling  $g: V(H) \rightarrow \{1, 2, \dots, m\}$  as follows:

**Case 1:**  $m \equiv 0 \pmod{6}$

$$g(p_j) = \begin{cases} j + 1; & j = 1 \pmod{6} \\ j + 2; & j = 2 \pmod{6} \\ j + 3; & j = 3 \pmod{6} \\ j - 3; & j = 4 \pmod{6} \\ j - 2; & j = 5 \pmod{6} \\ j - 1; & j = 0 \pmod{6} \end{cases}$$

**Case 2:**  $m \equiv 1 \pmod{6}$

$$g(p_j) = \begin{cases} j+1; & j = 1(\text{mod}6) \\ j+2; & j = 2(\text{mod}6) \\ j+3; & j = 3(\text{mod}6) \\ j-3; & j = 4(\text{mod}6) \\ j-2; & j = 5(\text{mod}6) \\ j-1; & j = 0(\text{mod}6) \end{cases}$$

$$g(p_m) = m$$

**Case 3:**  $m \equiv 2 \pmod{6}$

$$g(p_j) = \begin{cases} j; & j = 1(\text{mod}6) \\ j+1; & j = 2(\text{mod}6) \\ j+2; & j = 3(\text{mod}6) \\ j-2; & j = 4(\text{mod}6) \\ j-1; & j = 5(\text{mod}6) \\ j; & j = 0(\text{mod}6) \end{cases}$$

$$g(p_{m-1}) = m$$

$$g(p_m) = m - 1$$

**Case 4:**  $m \equiv 3 \pmod{6}$

$$g(p_j) = \begin{cases} j+1; & j = 1(\text{mod}6) \\ j+2; & j = 2(\text{mod}6) \\ j+3; & j = 3(\text{mod}6) \\ j-3; & j = 4(\text{mod}6) \\ j-2; & j = 5(\text{mod}6) \\ j-1; & j = 0(\text{mod}6) \end{cases}$$

$$g(p_m) = m - 1, \quad g(p_{m-1}) = m, \quad g(p_{m-2}) = m - 2$$

**Case 5:**  $m \equiv 5 \pmod{6}$

$$g(p_j) = \begin{cases} j+1; & j = 1(\text{mod}6) \\ j+2; & j = 2(\text{mod}6) \\ j+3; & j = 3(\text{mod}6) \\ j-3; & j = 4(\text{mod}6) \\ j-2; & j = 5(\text{mod}6) \\ j-1; & j = 0(\text{mod}6) \end{cases}$$

$$g(p_m) = m, \quad g(p_{m-4}) = m - 3, \quad g(p_{m-3}) = m - 1, \quad g(p_{m-2}) = m - 4,$$

$$g(p_{m-1}) = m - 2$$

| Cases             | Edge labelling             |
|-------------------|----------------------------|
| 0,1,2,3,4,5(mod6) | $ e_g(0) - e_g(1)  \leq 1$ |

As shown by the above labelling pattern that meets the required conditions  $|e_g(0) - e_g(1)| \leq 1$ . The cycle  $C_m$  with twin chord has been designated as a 3-equitable prime cordial graph.

**Algorithm:**

The MATLAB program that will generate the vector of edge labels and vertex labels in cycle  $C_m$  with twin chord is a 3-equitable prime cordial graph.

```
function [el,vl] = cyclewithtwinchord(m)
% CYCLEWITHTWINCHORD Construct and plot the "cycle with twin
chord".
% Returns:
% el - edge-label vector (original indexing behaviour preserved)
% vl - vertex-label vector (order used to build the graph)
%
% Supports rem(m,6) in {0,1,2,3,5}. Otherwise prints the original message.

% initialize
el = [];
vl = [];

b = [];
y = 1;
r = rem(m,6);

if (r == 0)
    for j = 1:m
        switch rem(j,6)
            case 1, b(y,:) = j+1;
            case 2, b(y,:) = j+2;
            case 3, b(y,:) = j+3;
            case 4, b(y,:) = j-3;
            case 5, b(y,:) = j-2;
            case 0, b(y,:) = j-1;
        end
        y = y + 1;
    end

    vl = reshape(b.',1,[]);
    [q,ed] = pair_from_vl(vl, 1, 2, m);
    H = build_graph_and_chain(q,ed);

    H = addedge(H, vl(1), vl(m));
    H = addedge(H, vl(1), vl(3));
    H = addedge(H, vl(1), vl(4));
```

```

elseif (r == 1)
    for j = 1:(m-1)
        switch rem(j,6)
            case 1, b(y,:) = j+1;
            case 2, b(y,:) = j+2;
            case 3, b(y,:) = j+3;
            case 4, b(y,:) = j-3;
            case 5, b(y,:) = j-2;
            case 0, b(y,:) = j-1;
        end
        y = y + 1;
    end
    b(y,:) = m; y = y + 1;

    vl = reshape(b.',1,[]);
    [q,ed] = pair_from_vl(vl, 1, 2, m-1);
    H = build_graph_and_chain(q,ed);

    H = addedge(H, vl(m-1), vl(m));
    H = addedge(H, vl(1), vl(m));
    H = addedge(H, vl(1), vl(3));
    H = addedge(H, vl(1), vl(4));

elseif (r == 2)
    for j = 1:(m-2)
        switch rem(j,6)
            case 1, b(y,:) = j;
            case 2, b(y,:) = j+1;
            case 3, b(y,:) = j+2;
            case 4, b(y,:) = j-2;
            case 5, b(y,:) = j-1;
            case 0, b(y,:) = j;
        end
        y = y + 1;
    end
    b(y,:) = (m-1) + 1; y = y + 1; % append m
    b(y,:) = m - 1; y = y + 1; % append m-1

    vl = reshape(b.',1,[]);
    [q,ed] = pair_from_vl(vl, 1, 2, m);
    H = build_graph_and_chain(q,ed);

    H = addedge(H, vl(1), vl(4));
    H = addedge(H, vl(1), vl(m));
    H = addedge(H, vl(1), vl(3));

```

```

elseif (r == 3)
    for j = 1:(m-3)
        switch rem(j,6)
            case 1, b(y,:) = j+1;
            case 2, b(y,:) = j+2;
            case 3, b(y,:) = j+3;
            case 4, b(y,:) = j-3;
            case 5, b(y,:) = j-2;
            case 0, b(y,:) = j-1;
        end
        y = y + 1;
    end
    b(y,:) = m-2; y = y + 1;
    b(y,:) = m-1; y = y + 1;
    b(y,:) = m;   y = y + 1;

    vl = reshape(b.',1,[]);
    [q,ed] = pair_from_vl(vl, 1, 2, m-1);
    H = build_graph_and_chain(q,ed);

    H = addedge(H, vl(m-1), vl(m));
    H = addedge(H, vl(1), vl(m));
    H = addedge(H, vl(1), vl(3));
    H = addedge(H, vl(1), vl(4));

elseif (r == 5)
    % --- important: use original per-j conditionals for the final block ---
    for j = 1:(m-5)
        switch rem(j,6)
            case 1, b(y,:) = j+1;
            case 2, b(y,:) = j+2;
            case 3, b(y,:) = j+3;
            case 4, b(y,:) = j-3;
            case 5, b(y,:) = j-2;
            case 0, b(y,:) = j-1;
        end
        y = y + 1;
    end

    % original code appended for j = (m-4):m using conditionals:
    for j = (m-4):m
        if j == m
            b(y,:) = j; y = y + 1;
        end
    end

```

```

        if j == m-4
            b(y,:) = j+1; y = y + 1;
        end
        if j == m-3
            b(y,:) = j+2; y = y + 1;
        end
        if j == m-2
            b(y,:) = j-2; y = y + 1;
        end
        if j == m-1
            b(y,:) = j-1; y = y + 1;
        end
    end
end

vl = reshape(b.',1,[]);
[q,ed] = pair_from_vl(vl, 1, 2, m-1);
H = build_graph_and_chain(q,ed);

H = addedge(H, vl(m-1), vl(m));
H = addedge(H, vl(1), vl(m));
H = addedge(H, vl(1), vl(3));
H = addedge(H, vl(1), vl(4));

else
    disp('Cycle with twin chord is not 3-equitable prime cordial at n=4(mod
6) ')
    return;
end

% compute edge labels following original gcd logic
el = zeros(1, m+2);

for j = 2:m
    el(j-1) = edge_label(vl(j-1), vl(j));
end

el(m) = edge_label(vl(1), vl(m));
el(m+1) = edge_label(vl(1), vl(3));
el(m+2) = edge_label(vl(1), vl(4));

% plot
h = plot(H);
camroll(90)
title([' The cycle with twin chord for m = ' num2str(m)])

```

```

end

%% Helper functions

function [q,ed] = pair_from_vl(vl, startIdx, step, lastIdx)
    q = []; ed = [];
    k = 1;
    for j = startIdx:step:lastIdx
        q(k) = vl(j);
        ed(k) = vl(j+1);
        k = k + 1;
    end
end

function H = build_graph_and_chain(q,ed)
    H = graph(q,ed);
    z = length(q);
    for jj = 2:z
        H = addedge(H, q(jj), ed(jj-1));
    end
end

function lab = edge_label(a,b)
    if gcd(a,b) == 1 && gcd(a+b, a-b) == 1
        lab = 1;
    elseif gcd(a,b) == 1 && gcd(a+b, a-b) == 2
        lab = 2;
    else
        lab = 0;
    end
end
end

```

**Example 1:** Case 1:  $m \equiv 0 \pmod{6}$ , the output will come in Figure window as Figure 1.

```

>>[el,vl]= cyclewithtwinchord(12)
el =

    0    0    1    2    2    1    0    0    1    2    2    1    0    1

vl =

    2    4    6    1    3    5    8   10   12    7    9   11

```

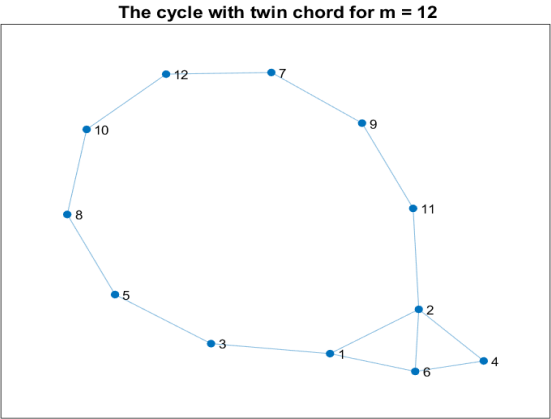


**Figure 1**  
**Cycle with twin chord  $C_{12}$**

**Example 2:** Case 2:  $m \equiv 1 \pmod{6}$ , the output will come in Figure window as Figure 2.

```
>>[el,vl]=cyclewithtwinchord(13)
```

```
el =
```

```
Columns 1 through 15
```

```
0 0 1 2 2 1 0 0 1 2 2 2 1 0 1
```

```
vl =
```

```
Columns 1 through 13
```

```
2 4 6 1 3 5 8 10 12 7 9 11 13
```

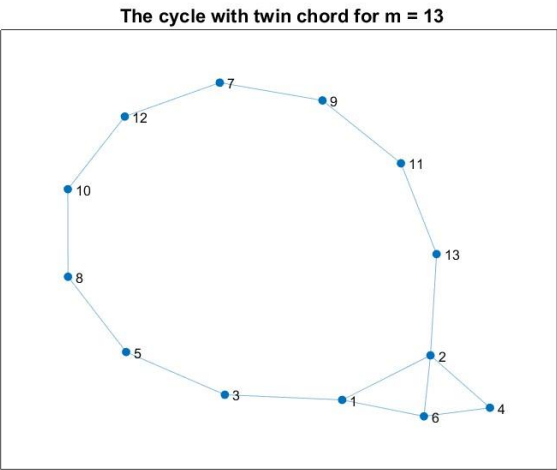


**Figure 2**  
**Cycle with twin chord  $C_{13}$**

**Example 3:** Case 3:  $m \equiv 2 \pmod{6}$ , the output will come in Figure window as Figure 3.

```
>> [el,vl]=cyclewithtwinchord(14)
```

el =

Columns 1 through Column 16

2 2 1 0 0 1 2 2 1 0 0 0 1 2 2 1

vl =

1 3 5 2 4 6 7 9 11 8 10 12 14 13

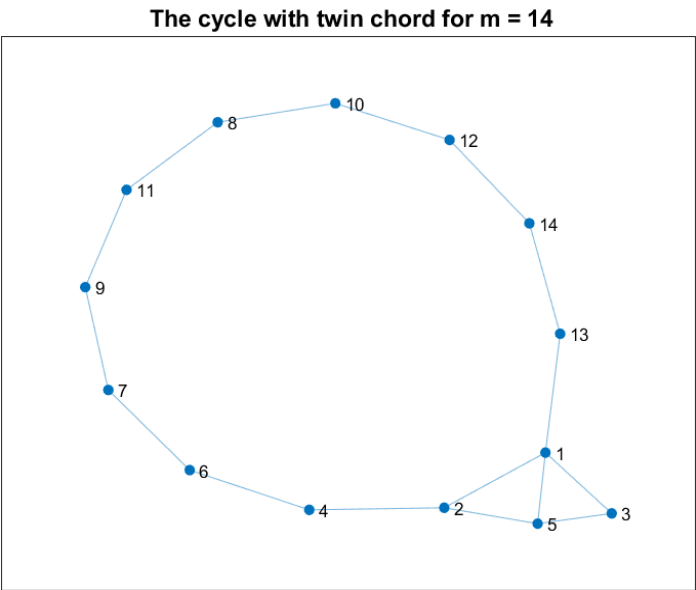


**Figure 3**  
Cycle with twin chord  $C_{14}$

**Example 4:** Case 4:  $m \equiv 3 \pmod{6}$ , the output will come in Figure window as Figure 4.

```
>> [el,vl]=cyclewithtwinchord(15)
```

el =

Columns 1 through Column 17

0 0 1 2 2 1 0 0 1 2 2 2 1 1 1 0 1

vl =  
Columns 1 through Column 15  
2 4 6 1 3 5 8 10 12 7 9 11 13 14 15

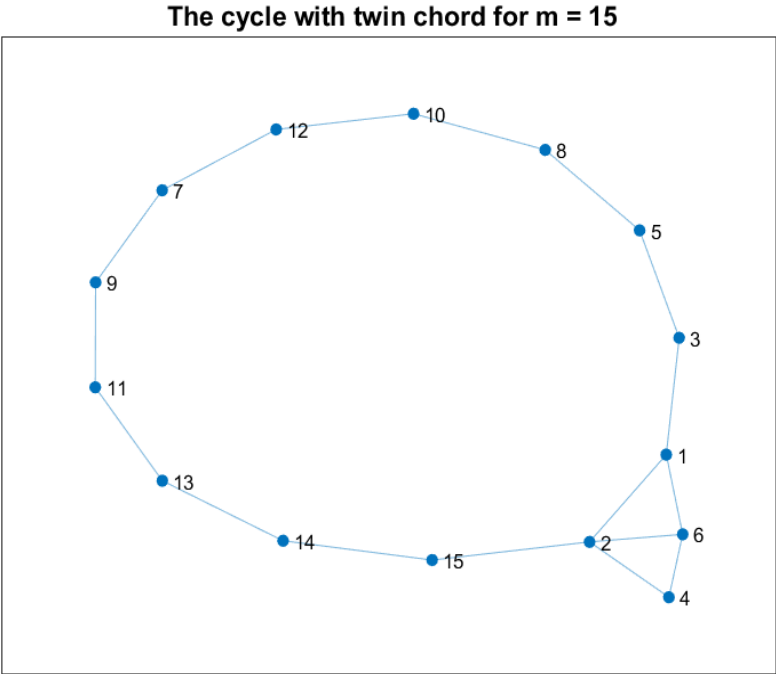


**Figure 4**  
Cycle with twin chord  $C_{15}$

**Example 5:** Case 5:  $m \equiv 5 \pmod{6}$ , the output will come in Figure window as Figure 5.

```
>>[el,vl]=cyclewithtwinchord(17)
el =
Columns 1 through 19

0 0 1 2 2 1 0 0 1 2 2 1 0 1 2 2 1 0 1

vl =
Columns 1 through 17

2 4 6 1 3 5 8 10 12 7 9 11 14 16 13 15 17
```

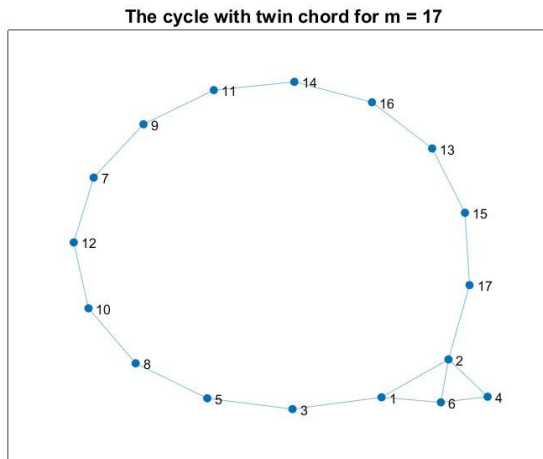


**Figure 5**  
Cycle with twin chord  $C_{15}$

### Conclusion

It is easy to use the given programs to find the vertex and edge labels of a cycle with a twin chord that supports 3-equitable prime cordial labeling, especially for larger values of  $m$ . It becomes cumbersome to try this manually. Research into similar programs is still ongoing, despite the fact that the edge label and vertex label are already known for many graphs.

### References

- [1] S. Abhirami and R. Vikramaprasad, "MATLAB program for energy of even sum cordial graphs," *Int. J. Mech. Eng. Technol.*, vol. 9, no. 2, pp. 601–614 (2018).
- [2] G. S. Bloom and S. W. Golomb, "Applications of numbered undirected graphs," *Proc. IEEE*, vol. 65, no. 4, pp. 562–570 (1977), doi: 10.1109/PROC.1977.10517.
- [3] R. Balakrishnan, "The energy of a graph," *Linear Algebra Appl.*, vol. 387, pp. 287–295 (2004), doi: 10.1016/j.laa.2004.02.038.
- [4] R. Balakrishnan and K. Ranganathan, *A Textbook of Graph Theory*, 1st ed. New York, USA: Springer (2000), doi: 10.1007/978-1-4419-8505-7.
- [5] J. A. Gallian, "A dynamic survey of graph labeling," *Electron. J. Combin.*, vol. 17, Dynamic Survey DS6 (2010).
- [6] J. L. Gross and J. Yellen, *Graph Theory and Its Applications*, 2nd ed. Boca Raton, FL, USA: Chapman & Hall/CRC (2006).

- [7] S. Gupta and S. Srivastav, "MATLAB algorithm for new results on energy of graphs," *Int. J. Comput. Sci. Inf. Secur.*, vol. 17, no. 3, pp. 14–17 (2019).
- [8] R. Sweetly and J. Paulraj Joseph, "The radio number of graphs," *J. Discrete Math. Sci. Cryptogr.*, vol. 12, no. 6, pp. 729–736 (2009), doi: 10.1080/09720529.2009.10698268.
- [9] R. Sweetly and J. Paulraj Joseph, "Radio number of graphs of diameter three," *J. Discrete Math. Sci. Cryptogr.*, vol. 10, no. 5, pp. 721–730 (2007), doi: 10.1080/09720529.2007.10698152.
- [10] S. Leel, S. Srivastav, S. Gupta, and G. Ganesan, "Domination number in the context of some new graphs," *Eng. Proc.*, vol. 62, no. 1, p. 14 (2024), doi: 10.3390/engproc2024062014.
- [11] P. Gupta, S. Gupta, and S. Srivastav, "Domination number for some STAR related graphs," *AIP Conf. Proc.*, vol. 3037, no. 1, Art. no. 020004 (2024), doi: 10.1063/5.0196045.
- [12] S. Gupta and S. Srivastav, "Results with MATLAB coding of middle graph of cycle and its related graphs in context of sum divisor cordial," *Int. J. Emerg. Trends Eng. Res.*, vol. 8, no. 2, pp. 398–401 (2020), doi: 10.30534/ijeter/2020/26822020.
- [13] F. Harary, *Graph Theory*, 3rd ed. Reading, MA, USA: Addison-Wesley (1972).
- [14] A. Lourdusamy and F. Patrick, "Sum divisor cordial labeling for star and ladder related graphs," *Proyecciones J. Math.*, vol. 35, no. 4, pp. 437–455 (2016), doi: 10.4067/S0716-09172016000400006.
- [15] S. Srivastav and S. Gupta, "Some more sum divisor cordial labelling of graphs," *Int. J. Comput. Sci. Inf. Secur.*, vol. 17, no. 4, pp. 143–145 (2019).
- [16] S. J. Chapman, *MATLAB Programming for Engineers*. Stamford, CT, USA: Cengage Learning (2008).

*Received January, 2026*