

Minimum rank problems

Leslie Hogben*

May 26, 2009

Abstract

A graph describes the zero-nonzero pattern of a family of matrices, with the type of graph (undirected or directed, simple or allowing loops) determining what type of matrices (symmetric or not necessarily symmetric, diagonal entries free or constrained) are described by the graph. The minimum rank problem of the graph is to determine the minimum among the ranks of the matrices in this family; the determination of maximum nullity is equivalent. This problem has been solved for simple trees [11, 9], trees allowing loops [5], and directed trees allowing loops [2]. We survey these results from a unified perspective and solve the minimum rank problem for simple directed trees.

Keywords minimum rank, maximum nullity, symmetric minimum rank, asymmetric minimum rank, rank, symmetric matrix, matrix, path cover, path cover number, zero forcing set, zero forcing number, tree, ditree, directed tree, graph.

AMS Classification 05C05, 15A03, 15A18

1 Introduction

A graph describes the zero-nonzero pattern of a family of matrices, with the type of graph (undirected or directed, simple or allowing loops) determining what type of matrices (symmetric or not necessarily symmetric, diagonal entries free or constrained) are described by the graph. The minimum rank problem of the graph is to determine the minimum among the ranks of the matrices in this family; the determination of maximum nullity is equivalent. This problem has been studied extensively for various types of graphs.

Let G be an undirected (respectively, directed) graph. The *vertex set* of G , denoted by $V(G)$, is a nonempty set. The *edge (arc) set* of G , denoted by $E(G)$, is a set of two element multisets (ordered pairs) of vertices; multiple edges (multiple arcs in the same direction) between one particular pair of vertices are not permitted. The *order* of G , denoted $|G|$, is the number of vertices of G .

We examine four types of graphs. When describing a specific type of graph, we always use one of the terms *simple* or *loop* and one of the terms *graph* or *digraph*. Thus a *simple digraph* is a directed graph that does not allow loops, and a *loop graph* is an undirected graph that allows loops. The term *simple (di)graph* means a simple graph or simple digraph, and analogously for *loop (di)graph*. The term *undirected graph* (respectively, *directed graph*) means a simple graph or a loop graph (simple digraph or loop digraph). We do however, use the unqualified term *graph* more broadly, as in a *graph of any type*, which means one of a simple graph, a loop graph, a simple digraph, or a loop digraph.

All of the matrices we discuss are real and square, although the minimum rank/maximum nullity problem has been studied over fields other than the real numbers. Each type of graph describes a set of matrices. For a simple graph G , the *qualitative class* of G , i.e., the family of matrices described by G , is

*Department of Mathematics, Iowa State University, Ames, IA 50011, USA (lhogben@iastate.edu) and American Institute of Mathematics, 360 Portage Ave, Palo Alto, CA 94306 (hogben@aimath.org).

$$\mathcal{Q}(G) = \{A \in \mathbb{R}^{|G| \times |G|} : A^T = A \text{ and for } i \neq j, a_{ij} \neq 0 \Leftrightarrow \{i, j\} \in E(G)\}.$$

For a simple digraph G ,

$$\mathcal{Q}(G) = \{A \in \mathbb{R}^{|G| \times |G|} : \text{for } i \neq j, a_{ij} \neq 0 \Leftrightarrow (i, j) \in E(G)\}.$$

For a loop graph G ,

$$\mathcal{Q}(G) = \{A \in \mathbb{R}^{|G| \times |G|} : A^T = A \text{ and } a_{ij} \neq 0 \Leftrightarrow \{i, j\} \in E(G)\}.$$

For a loop digraph G ,

$$\mathcal{Q}(G) = \{A \in \mathbb{R}^{|G| \times |G|} : a_{ij} \neq 0 \Leftrightarrow (i, j) \in E(G)\}.$$

Note that in the family of matrices described by a simple (di)graph, the diagonal entries of the matrix are free, whereas in the family of matrices described by a loop (di)graph, the zero-nonzero pattern of the diagonal is constrained by the absence or presence of loops. The matrices described by an undirected graph are symmetric, whereas those described by a directed graph are not required to be symmetric. In the literature, the family of matrices described by an undirected graph G is usually denoted by $\mathcal{S}(G)$ rather than $\mathcal{Q}(G)$, but the latter notation facilitates our unified treatment of graphs of any type.

For a graph G of any type,

$$\text{mr}(G) = \min\{\text{rank}(A) : A \in \mathcal{Q}(G)\} \quad \text{and} \quad \text{M}(G) = \max\{\text{null}(A) : A \in \mathcal{Q}(G)\}.$$

Clearly $\text{mr}(G) + \text{M}(G) = |G|$.

One source of interest in the problem of determining the minimum rank/maximum nullity of the real symmetric matrices described by a simple graph is the connection with possible eigenvalues of the same family of matrices, because maximum nullity is the same as maximum multiplicity of an eigenvalue for this family of matrices. The problem of determining the minimum rank of a simple graph has received considerable attention recently (see [6] for a survey with extensive bibliography). The Jordan structure for the zero eigenvalue of matrices described by a loop digraph was studied earlier in a series of papers by Hershkowitz and Schneider, summarized in [7].

Trees are an important class of graphs for which the minimum rank/maximum nullity problem has been solved, and the one that was studied first for simple graphs. To define the various types of tree, we need some terminology. Let G be an undirected (respectively, directed) graph. A *path* in G is an ordered set of distinct vertices (v_1, \dots, v_k) of G such that for $i = 1, \dots, k - 1$, the edges $\{v_i, v_{i+1}\}$ (respectively, arcs (v_i, v_{i+1})) are contained in $E(G)$. The vertex v_1 is the *initial point* of P and v_k is the *final point* of P . A *cycle* in G is an ordered set of vertices (v_1, \dots, v_k) of G such that v_1, \dots, v_{k-1} are distinct, $v_k = v_1$, and for $i = 1, \dots, k - 1$, the edges $\{v_i, v_{i+1}\}$ (respectively, arcs (v_i, v_{i+1})) are contained in $E(G)$. The *length* of a cycle is the number of distinct vertices.

If G is a simple (respectively, loop) digraph, the associated simple (loop) graph \widehat{G} is the graph having the same vertex set, and $\{i, j\} \in E(\widehat{G})$ exactly when at least one of $(i, j), (j, i) \in E(G)$. An undirected (respectively, directed) graph G is *connected* if there is path in G (respectively, \widehat{G}) from any vertex of G to any other vertex of G . A directed graph G is *strongly connected* if there is path in G from any vertex of G to any other vertex of G .

A *forest* is an undirected (respectively, directed) graph G such that G (respectively, \widehat{G}) has no cycles of length greater than two; a tree is a connected forest. The terms *simple tree* or *loop tree* refer to undirected trees whereas *simple ditree* or *loop ditree* refer to directed trees; analogous terminology is used for forests.

The minimum rank/maximum nullity problem has been solved for simple trees [11, 9], loop trees [5], and loop ditrees [2] (of course solving the minimum rank/maximum nullity problem for a particular type of tree also solves the problem for the same type of forest). In this context, ‘solved’ means showing that minimum rank or maximum nullity is equal to a graph parameter, thereby reducing the problem of finding a minimum (or maximum) over an infinite set of matrices to finding the optimum over a finite set of vertices – see Section 4 for more information on methods of computation.

This paper surveys the results on minimum rank/maximum nullity of all types of trees from a unified perspective and solves the minimum rank problem for simple ditrees (Section 3). In Section 2 we first

examine two parameters, zero forcing number and path cover number, which are used to solve the minimum rank problem for trees, for any type of graph.

Some additional terminology is needed. In an undirected graph G , vertex v is a *neighbor* of u if $\{u, v\} \in E(G)$. In an directed graph G , vertex v is an *out-neighbor* (respectively, *in-neighbor*) of u if $(u, v) \in E(G)$ ($(v, u) \in E(G)$). Note that if there is a loop (edge $\{u, u\}$ or arc (u, u)), u is an neighbor or out- and in-neighbor of itself.

To *reverse* the arc (u, v) in a directed graph means to replace it by the arc (v, u) . A directed graph G is *symmetric* if the result of reversing every arc of G leaves G unchanged (this means the same arc set, not just an arc set that gives an isomorphic graph).

Finally, it should be noted that a symmetric (simple or loop) digraph G and its associated (simple or loop) graph describe different families of matrices, even though the positions of nonzero entries are identical, and this difference can produce a difference in minimum rank/maximum nullity, as in the next (well-known) example.

Example 1.1. Let $K_{3,3,3}$ be the symmetric simple digraph whose associated simple graph $\widehat{K_{3,3,3}}$ is shown in Figure 1 (to obtain $K_{3,3,3}$ from $\widehat{K_{3,3,3}}$, replace each edge $\{i, j\}$ by the pair of arcs $(i, j), (j, i)$).

It is known [4] that $\text{mr}(\widehat{K_{3,3,3}}) = 3$. For the block matrix $A = \begin{bmatrix} 0 & J & -J \\ J & 0 & J \\ J & J & 0 \end{bmatrix}$ where $J = \begin{bmatrix} 1 & 1 & 1 \\ 1 & 1 & 1 \\ 1 & 1 & 1 \end{bmatrix}$,

$A \in \mathcal{Q}(K_{3,3,3})$ and $\text{rank}(A) = 2$. Thus $\text{mr}(K_{3,3,3}) < \text{mr}(\widehat{K_{3,3,3}})$ and $M(K_{3,3,3}) > M(\widehat{K_{3,3,3}})$.

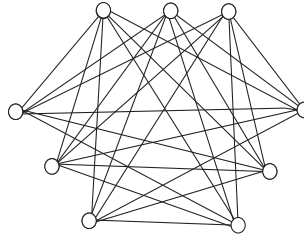


Figure 1: The simple graph $\widehat{K_{3,3,3}}$ for Example 1.1.

2 Relationships between zero forcing number, path cover number, and maximum nullity for graphs

In this section we discuss for graphs of any type the graph parameters zero forcing number, which bounds maximum nullity, and path cover number. These parameters are most useful for trees (see Section 3).

The zero forcing number was introduced in [1] for simple graphs, and extended to loop digraphs in [2]. In fact, the definition is identical for graphs of any type except for the color-change rule, which varies with the type of graph.

Definition 2.1. Let G be a graph of any type, with each vertex colored either white or black.

- The *color-change rule* depends on the type of graph.
 - Let G be a a simple graph. If u is a black vertex and exactly one neighbor v of u is white, then change the color of v to black.
 - Let G be a a simple digraph. If u is a black vertex and exactly one out-neighbor v of u is white, then change the color of v to black.

- Let G be a loop graph. If exactly one neighbor v of u is white, then change the color of v to black (the possibility that $u = v$ is permitted).
- Let G be a loop digraph. If exactly one out-neighbor v of u is white, then change the color of v to black (the possibility that $u = v$ is permitted).
- When the color-change rule is applied to u to change the color of v , we say u forces v , and write $u \rightarrow v$.
- Given a coloring of G , the *derived set* is the set of black vertices obtained by applying the color-change rule until no more changes are possible.
- A *zero forcing set* for G is a subset of vertices Z such that if initially the vertices in Z are colored black and the remaining vertices are colored white, then the derived set is all the vertices of G .
- For a given zero forcing set, construct the derived set, listing the forces in the order in which they were performed. This list is a *chronological list of forces*.
- The *zero forcing number* $Z(G)$ is the minimum of $|Z|$ over all zero forcing sets $Z \subseteq V(G)$.

The derived set (of a specific coloring of a graph) is in fact unique, since any vertex that turns black under one sequence of applications of the color change rule can always be turned black regardless of the order of color changes. Since for our purposes the uniqueness of the derived set is not necessary, we do not supply the details.

Example 2.2. Recall that for a simple digraph, in order for u to force v , u must be black and v must be the unique white out-neighbor of u . For the simple digraph G_2 shown in Figure 2, $\{1, 4, 9\}$ is a zero forcing set, with a chronological list of forces

$$9 \rightarrow 7, \quad 7 \rightarrow 8, \quad 1 \rightarrow 3, \quad 4 \rightarrow 5, \quad 3 \rightarrow 2, \quad 5 \rightarrow 6. \quad (1)$$

There are other chronological lists of forces for this zero forcing set, e.g.,

$$1 \rightarrow 3, \quad 4 \rightarrow 5, \quad 3 \rightarrow 2, \quad 5 \rightarrow 6, \quad 6 \rightarrow 7, \quad 7 \rightarrow 8. \quad (2)$$

It is shown in Example 2.17 below that $Z(G_2) = 3$.

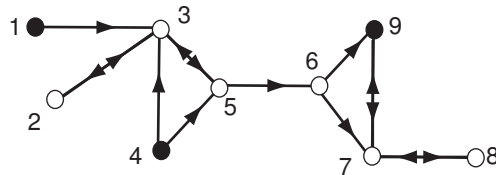


Figure 2: The simple digraph G_2 for Example 2.2 with a zero forcing set colored black.

Example 2.3. Recall that for a loop graph, u can force v if v is the unique white neighbor of u (regardless of whether u is black or white). For the loop graph G_3 shown in Figure 3, $\{3\}$ is a zero forcing set with a chronological list of forces

$$1 \rightarrow 1, \quad 2 \rightarrow 2, \quad 8 \rightarrow 7, \quad 4 \rightarrow 5, \quad 3 \rightarrow 4, \quad 6 \rightarrow 9, \quad 5 \rightarrow 6, \quad 7 \rightarrow 8. \quad (3)$$

Another zero forcing set is $\{5\}$ with chronological lists of forces

$$4 \rightarrow 3, \quad 1 \rightarrow 1, \quad 2 \rightarrow 2, \quad 3 \rightarrow 4, \quad 8 \rightarrow 7, \quad 6 \rightarrow 9, \quad 9 \rightarrow 6, \quad 7 \rightarrow 8. \quad (4)$$

Since the derived set of the empty set is $\{7\} \neq V(G_3)$, $Z(G_3) = 1$.

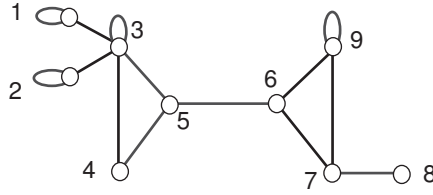


Figure 3: The loop graph G_3 for Example 2.3.

Zero forcing number is of interest in the study of maximum nullity and minimum rank because zero forcing number bounds maximum nullity from above, and, in fact, that was the motivation for defining zero forcing number. The idea is to show that if a vector in the null space has zeros in the coordinates associated with a zero forcing set, then the vector must be zero. The next result then gives the bound (by a proof similar to that of [1, Proposition 2.4]).

Proposition 2.4. [1, Prop. 2.2] *If A is a matrix and $\text{null}(A) > k$, then there is a nonzero vector $\mathbf{x} \in \ker(A)$ vanishing at any k specified positions.*

Theorem 2.5. *If G is any type of graph, then $M(G) \leq Z(G)$.*

Proof. Assume that $M(G) > Z(G)$. Let Z be a zero forcing set such that $|Z| = Z(G)$. Construct the derived set of Z and the chronological list of forces $u_j \rightarrow v_j, j = 1, \dots, t$. Let $A \in \mathcal{Q}(G)$ be such that $\text{null}(A) = M(G)$. By Proposition 2.4, there exists a nonzero vector $\mathbf{x} = [x_i] \in \ker(A)$ such that $x_z = 0$ for all $z \in Z$.

Proceeding in chronological order of forces, we show that if (in addition to the coordinates associated with the zero forcing set) $x_{v_j} = 0$ for $j = 1, \dots, k - 1$, then $x_{v_k} = 0$. Since Z is a zero forcing set, the derived set is all the vertices of G , so this implies $\mathbf{x} = 0$, contradicting the choice of \mathbf{x} . Assume $x_z = 0$ for all $z \in Z$ and $x_{v_j} = 0$ for $j = 1, \dots, k - 1$. Since $\mathbf{x} \in \ker(A)$, $(A\mathbf{x})_{u_k} = 0$. The color change rule and the force $u_k \rightarrow v_k$ reduce this equation to $a_{u_k, v_k} x_{v_k} = 0$. Since $a_{u_k, v_k} \neq 0$, $x_{v_k} = 0$. \square

The proof of Theorem 2.5 explains why out-neighbors were used in the definition of zero forcing number rather than in-neighbors for directed graphs. One could define an in-zero forcing number, and show that this also bounds maximum nullity, e.g., by reversing all the arcs, which corresponds to transposing the matrices described by the directed graph. However, this does not add additional information: From a relationship between zero forcing number and the additional parameter triangle number [2, Theorem 4.13], it follows that reversing all the arcs does not change the zero forcing number.

The idea of a forcing chain was introduced in [2] for loop digraphs to connect the zero forcing number to the path cover number (defined formally in Definition 2.10 below). We extend the forcing chain idea to all types of graphs.

Definition 2.6. Let G be any type of graph. For a fixed chronological list of forces of a zero forcing set Z of G ,

- A *forcing chain* is an ordered set of vertices (v_1, v_2, \dots, v_k) such that $k > 1$ and $v_i \rightarrow v_{i+1}$ for all $i = 1, \dots, k - 1$ (and no force is repeated), or $k = 1$ and $v_1 \in Z$.
- A *maximal forcing chain* is a forcing chain that is not a proper subset of another forcing chain.
- The *chain set* is the set of all maximal forcing chains (of the given chronological list of forces of a zero forcing set).

Example 2.7. For the simple digraph G_2 in Example 2.2, zero forcing set $\{1, 4, 9\}$, and chronological list of forces (1), the chain set is

$$(1, 3, 2), \quad (4, 5, 6) \quad (9, 7, 8).$$

For G_2 , $\{1, 4, 9\}$, and chronological list of forces (2), the chain set is

$$(1, 3, 2), \quad (4, 5, 6, 7, 8) \quad (9).$$

For the loop graph G_3 in Example 2.3, zero forcing set $\{3\}$, and chronological list of forces (3), the chain set is

$$(1, 1), \quad (2, 2), \quad (3, 4, 5, 6, 9) \quad (8, 7, 8).$$

For G_3 , $\{5\}$, and chronological list of forces (4), the chain set is

$$(1, 1), \quad (2, 2), \quad (3, 4, 3) \quad (5) \quad (6, 9, 6) \quad (8, 7, 8).$$

There are two types of maximal forcing chains in Example 2.7, paths and cycles, and this is true in general.

Lemma 2.8. *Let G be any type of graph. Every forcing chain of G is a path or a cycle. For a chain set constructed from a zero forcing set Z , the maximal forcing chains partition the vertices of G , and the elements of Z are in one-to-one correspondence with the maximal forcing chains that are paths.*

Proof. From the definition of forcing, a vertex can force at most one vertex and can be forced by at most one vertex. Thus a forcing chain does not contain any repeated vertex except possibly the first and the last, so every forcing chain is a path or a cycle, and two maximal zero forcing chains have disjoint vertices. If we start with a zero forcing set and produce a chain set, then every vertex is in this chain set, and the vertices in Z are exactly the vertices that have never been forced by any other vertex (i.e., exactly the initial vertices of the paths). \square

In [9], a path cover is defined for a simple tree T as a set of vertex disjoint induced paths that cover all the vertices of T , and the path cover number $P(T)$ is the minimum number of paths in a path cover of T . With these definitions, it is shown that $P(T) = M(T)$ for a simple tree T .

What is an appropriate extension of the definition of path cover for other types of trees, or for any type of graph? The superficially obvious interpretation of this definition applied to a loop (di)graph (literally the same definition) fails to retain the property $P(T) = M(T)$ for loop (di)trees.

Example 2.9. Let T_4 be the loop tree shown in Figure 4. Since any matrix $A \in \mathcal{Q}(T_4)$ is of the form $A = \begin{bmatrix} 0 & a \\ a & 0 \end{bmatrix}$, it is clear that A is nonsingular and thus $M(T_4) = 0$. But any path cover that covers all vertices contains at least one path.



Figure 4: The loop tree T_4 for Example 2.9.

It is hardly surprising that the literal definition fails to retain $P(T) = M(T)$, since some loop (di)trees, such as T_4 , have maximum nullity equal to zero. We say a loop (di)graph G *requires nonsingularity* if $M(G) = 0$, i.e., $A \in \mathcal{Q}(G)$ implies A is nonsingular; otherwise G allows singularity.

The fact that every simple graph allows singularity, which is immediate by considering $A - \lambda I$ where $A \in \mathcal{Q}(G)$ and λ is an eigenvalue of A , plays a crucial role in the proof that $P(T) = M(T)$ for simple

trees. In [2, Definition 4.19] the definition of path cover number was generalized to loop digraphs (and implicitly also to loop graphs) in a manner that retains the property $P(T) = M(T)$ for a loop ditree. A key idea was to ignore components that require nonsingularity (such components cannot exist in a simple graph). This definition also extends naturally to simple digraphs, and in fact the same definition is valid for all types.

Definition 2.10. Let G be a graph of any type. A *path cover* of G is a set of vertex disjoint paths whose deletion from G leaves a graph that requires nonsingularity (or the empty set). The *path cover number* $P(G)$ is the minimum number of paths in a path cover.

There is an important connection between zero forcing sets and path covers.

Definition 2.11. Let G be a graph of any type. A *ZFS path cover* of G is the set of maximal zero forcing chains that are paths in a chain set of G .

Example 2.12. The ZFS path covers for G_2 produced by the chronological lists of forces (1) and (2) in Example 2.2 are the chain sets given in Example 2.7, i.e., $\{(1, 3, 2), (4, 5, 6), (9, 7, 8)\}$ and $\{(1, 3, 2), (4, 5, 6, 7, 8), (9)\}$, respectively.

The ZFS path covers for G_3 produced by the chronological lists of forces (3) for zero forcing set $\{3\}$ and (4) for $\{5\}$ in Example 2.3 are $\{(3, 4, 5, 6, 9)\}$ and $\{(5)\}$, respectively.

Note that in each case in Example 2.12, the ZFS path cover was a path cover. This is true in general, and was established for loop digraphs in [2]. The proof is valid for all types of graphs.

Theorem 2.13. *For any type of graph G , every ZFS path cover is a path cover of G . Thus $P(G) \leq Z(G)$.*

Proof. Let Z be a zero forcing set of order $Z(G)$. Construct a chronological list of forces and its chain set, and let P be its ZFS path cover. Delete the vertices in P from G . Then the rest of the graph, $G - P$, can force itself, i.e., $Z(G - P) = 0$. Since $M(G - P) \leq Z(G - P)$, $G - P$ requires nonsingularity. By Lemma 2.8, the number of paths in P is equal to $|Z|$, so $P(G) \leq Z(G)$. \square

The existence of components (cycles) that are not paths is one of the obvious differences between the chain sets of graphs G_2 and G_3 (cf. Example 2.7), but there are others. The paths in the chain sets of G_2 were all induced (see Definition 2.14 below), whereas the path $(3, 4, 5, 6, 9)$ in the chain set for G_3 for zero forcing set $\{3\}$ is not induced.

This presents a bit of a dilemma in defining a path cover – should a path be required to be induced? Recall that the original definition in [9] used the term *induced path*. Of course, this is irrelevant for any type of tree, where every path is induced, so Definition 2.10 is a valid extension of the definition of path cover given in [9], and was chosen in [2] to facilitate the connection between path covers and zero forcing sets. Here we examine further the issue of requiring paths to be induced.

Definition 2.14. Let G be any type of graph. A path (v_1, \dots, v_k) in G is *induced* if $E(G)$ does not contain any edge (arc) of the form $\{v_i, v_j\}$ with $j > i + 1$ ((v_i, v_j) with $j > i + 1$ or $i > j + 1$).

if G is a directed graph, a path (v_1, \dots, v_k) in G is *Hessenberg* if $E(G)$ does not contain any arc of the form (v_i, v_j) with $j > i + 1$; in an undirected graph a Hessenberg path is the same as an induced path.

An *induced (Hessenberg) path cover* is a set of vertex disjoint induced (Hessenberg) paths whose deletion from G leaves a graph that requires nonsingularity (or the empty set).

An induced path cover for an undirected (directed) graph contains only the edges (arcs) in the path and possibly some loops (and in a directed graph, possibly some additional arcs that can be obtained by reversing arcs in the path).

For a forcing chain (v_1, \dots, v_k) , we say that the forces are *done in chronological order* if for all $i = 1, \dots, k - 2$, $v_i \rightarrow v_{i+1}$ precedes $v_{i+1} \rightarrow v_{i+2}$ in the chronological list of forces. For the paths in the

chain sets of G_2 , the forces in each path were done in chronological order (although forces in different paths were interspersed chronologically); this is not true for the path $(3, 4, 5, 6, 9)$ in the chain set of G_3 for zero forcing set $\{3\}$, where the force $4 \rightarrow 5$ precedes $3 \rightarrow 4$.

Lemma 2.15. *Let G be a graph of any type. Fix a particular chronological list of forces of a zero forcing set. Let (v_1, \dots, v_k) be a path in its ZFS path cover. If the forces in (v_1, \dots, v_k) are done in chronological order, then (v_1, \dots, v_k) is a Hessenberg path of G .*

Proof. If the forces are done in chronological order, then when $v_i \rightarrow v_{i+1}$ occurs, for $j > i + 1$, v_j has not yet been forced and is therefore white. Since v_i forces v_{i+1} , v_j is not an (out) neighbor of v_i . Therefore (v_1, \dots, v_k) is a Hessenberg path of G . \square

For simple graphs or simple digraphs, because the color change rule requires that a vertex be black to force, the forces in a forcing chain must be done in chronological order, which implies that a zero forcing chain is an Hessenberg path.

Corollary 2.16. *For a simple graph (simple digraph) G , every ZFS path cover is an induced (Hessenberg) path cover of G .*

It is easy to verify that, as required by Corollary 2.16, both the ZFS path covers of the simple digraph G_2 produced in Example 2.12 are Hessenberg path covers. In the next example we show these are not minimum path covers, and in fact G_2 does not have a Hessenberg minimum path cover.

Example 2.17. Note that $\{(1, 3, 2), (4, 5, 6, 9, 7, 8)\}$ is a path cover, but is clearly not Hessenberg. Since any path cover must have 1 and 4 as initial points of (separate) paths, $P(G_2) = 2$. But any path cover consisting of two paths is not Hessenberg: Since 1 and 4 must be on separate paths, to cover G_2 with only two paths, 6, 7, 8, 9 must all be in the same path, which is therefore not Hessenberg. Since every ZFS path cover is a Hessenberg path cover, $Z(G_2) \geq 3$, and since Example 2.2 exhibited a zero forcing set of three vertices, $Z(G_2) = 3$.

It is not always the case for a simple digraph that there exists an induced minimal path cover, even when a ZFS path cover is a minimum path cover (and is necessarily a Hessenberg path cover).

Example 2.18. The set $\{1\}$ is a zero forcing set for the simple digraph G_5 shown in Figure 5, so $P(G_5) = M(G_5) = Z(G_5) = 1$. Any induced path cover must have at least two paths, because all vertices must be covered and G_5 is not itself a path. Thus Theorem 2.13 would be false if the definition of path cover required induced paths. Note that the path $(1, 2, 3, 4)$ in a ZFS path cover produced by $\{1\}$ is Hessenberg.

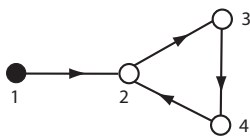


Figure 5: The simple digraph G_5 for Example 2.18.

The next example shows that for loop digraphs, it is not always the case that the forces can be done in chronological order and if the paths in a minimal path cover were required to be Hessenberg, then $P(G) \leq Z(G)$ would not be true for some graphs.

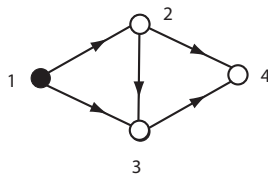


Figure 6: The loop digraph G_6 for Example 2.19.

Example 2.19. [2, Example 4.17] Let G_6 be the loop digraph shown in Figure 6. The set $\{1\}$ is a zero forcing set of G_6 , $3 \rightarrow 4, 2 \rightarrow 3, 1 \rightarrow 2$ is a chronological list of forces that produces the ZFS path cover $\{(1, 2, 3, 4)\}$, so $P(G_6) = M(G_6) = Z(G_6) = 1$. Any Hessenberg path cover must have at least two paths, because the graph has no cycles, so any subgraph (with nonempty vertex set) allows singularity, and the graph is not itself a Hessenberg path. Thus Theorem 2.13 would be false if the definition of path cover required Hessenberg paths.

Recall that for any type of graph G , $M(G) \leq Z(G)$ and $P(G) \leq Z(G)$. The relationship between $P(G)$ and $M(G)$ is less clear. It is known that for simple graphs path cover number and maximum nullity are incomparable. It is easy to find an example of a graph G having $P(G) < M(G)$, e.g., the complete graph (of any type, including all loops for a loop (di)graph) on four vertices has $M(K_4) = 3$ but $P(K_4) = 1$ (or 2 if the paths are required to be induced). It is less trivial to find an example of a simple (di)graph G such that $P(G) > M(G)$, but the first known example of such a simple graph, the pentasun, retains this inequality when viewed as a symmetric simple digraph:

Example 2.20. [3, Proposition 3.2] Let G_7 denote the pentasun (there called H_5), the simple graph shown in Figure 7. Then $P(G_7) = 3 > 2 = M(G_7)$.

Let H be the symmetric simple digraph that satisfies $\hat{H} = G_7$. Since a path can contain at most two of the five vertices of degree one, $P(H) = 3$. The proof given in [3] that $M(G_7) = 3$ relies only on the positions of the nonzero entries, not on symmetry of the matrices, and so $M(H) = 2$.

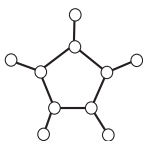


Figure 7: The pentasun G_7 for Example 2.20.

When loops are added to the pentasun to reflect the positions of nonzero diagonal entries in a matrix realizing $\text{mr}(G_7)$, the path cover number of this loop graph is then two. In fact, when the pentasun is used to create a loop graph with any choice of loops, either the path cover number goes down or the minimum rank goes up. In [2] it was asked whether there exists a loop digraph G for which $P(G) > M(G)$, and the analogous question can also be asked for loop graphs.

3 Equality of parameters for trees

Symmetric (simple or loop) ditrees are the strongly connected components of (simple or loop) ditrees and play an important role in the analysis of directed trees. We begin our discussion of trees with some

observations about parameter equality for a symmetric (simple or loop) ditree T and its associated (simple or loop) tree \widehat{T} .

Observation 3.1. *If G is a symmetric (simple or loop) digraph, then the zero forcing sets (and forcing chains, etc.) are exactly those of the associated (simple or loop) graph \widehat{G} , and $Z(G) = Z(\widehat{G})$.*

Note that zero forcing number is a graph parameter that does not involve matrices. For a symmetric directed graph G , the matrices in $\mathcal{Q}(G)$ are not required to be symmetric, whereas the matrices in $\mathcal{Q}(\widehat{G})$ are required to be symmetric, so $\mathcal{Q}(\widehat{G}) \subseteq \mathcal{Q}(G)$, and strict containment is possible. Thus it is possible to have $M(\widehat{G}) < M(G)$, as in Example 1.1. Path cover number is a graph parameter that does not involve matrices for a simple (di)graph but for loop (di)graphs, nonsingularity of components is an issue, and it is possible to have $P(\widehat{G}) < P(G)$.

Example 3.2. Let C_3 be a symmetric 3-cycle without loops viewed as a loop digraph. Let $A = \begin{bmatrix} 0 & 1 & -1 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{bmatrix}$. Then $A \in \mathcal{Q}(C_3)$ and $\det(A) = 0$, so $P(C_3) = 1$, but if $B \in \mathcal{Q}(\widehat{C}_3)$, then B has the form $B = \begin{bmatrix} 0 & a & b \\ a & 0 & c \\ b & c & 0 \end{bmatrix}$, and $\det(B) = 2abc \neq 0$. Thus \widehat{C}_3 requires nonsingularity, so $P(\widehat{C}_3) = 0$.

But for a symmetric (simple or loop) ditree T it is not possible to have $M(\widehat{T}) < M(T)$ or $P(\widehat{T}) < P(T)$, because it is well-known that given a (not necessarily symmetric) matrix $A \in \mathcal{Q}(T)$, there exist nonsingular diagonal matrices D_1, D_2 such that D_1AD_2 is symmetric, and so $D_1AD_2 \in \mathcal{Q}(\widehat{T})$.

Observation 3.3. *If T is a symmetric (simple or loop) ditree then $M(T) = M(\widehat{T})$.*

Observation 3.4. *If G is a symmetric simple digraph then the path covers are exactly those of the associated simple graph \widehat{G} , and $P(G) = P(\widehat{G})$. If T is a symmetric loop ditree then the path covers are exactly those of the associated loop graph \widehat{T} , and $P(T) = P(\widehat{T})$.*

The equality of path cover number and zero forcing number was established for simple trees in [1, Prop. 4.2]. This proof is easily adapted to simple ditrees.

Theorem 3.5. *For a simple ditree T , $P(T) = Z(T)$.*

Proof. We show by induction on $P(T)$ that the set Z consisting of the initial points of all the paths of a minimum path cover is a zero forcing set for T . This is clearly true for $P(T) = 1$. Assume true for all simple ditrees T' such that $P(T') < P(T)$. Choose a minimum path cover for T , let Z be the set of initial points of the paths in the minimum path cover, and choose a path P_1 in the minimum path cover that is joined to the rest of T by only one arc (u, v) (exactly one of u, v is a vertex of P_1 , and the other is a vertex of the simple ditree $T - P_1$). Apply the color-change rule repeatedly starting at the (black) initial point of P_1 until all vertices on P_1 through whichever of $u, v \in P_1$ are colored black. Now the path P_1 is irrelevant to the analysis of $T - P_1$, so by the induction hypothesis, the (black) initial points of the remaining paths are a zero forcing set for $T - P_1$, and all vertices not in P_1 , including whichever of $u, v \notin P_1$, can be colored black. Hence the remainder of path P_1 can also be colored black and Z is a zero forcing set for T . \square

Note that in the proof of Theorem 3.5, it is important that all vertices are covered by paths. It is possible to extend the method of proof in Theorem 3.5 to loop (di)trees, but since it introduces substantial complications due to the existence of components (requiring nonsingularity) that are not part of the path cover, and since $P(T) = Z(T)$ was established for loop ditrees in [2, Corollary 5.2], we do not provide such a proof here, and instead apply the loop ditree result to loop trees, by using Observations 3.1 and 3.4.

Corollary 3.6. *For any type of tree T , $P(T) = Z(T)$.*

We now turn our attention to connecting the results about the equality of zero forcing number and path cover number to maximum nullity. Results relating maximum nullity to other readily computable parameters such as path cover number have been established by several authors, including Johnson and Leal-Duarte; DeAlba, Hardy, Hentzel, Hogben and Wangsness; and Barioli, Fallat, Hall, Hershkowitz, Hogben, van der Holst, and Shader.

For (simple or loop) trees, an additional parameter is used. For a simple tree T , the parameter $\Delta(T)$ is defined in [9] to be the maximum over $p - q$ where the deletion of a set of q vertices from T leaves p paths. In [5], this is extended to a loop tree T by defining the parameter $\mathcal{C}_0(T)$ to be the maximum over $p - q$ where the deletion of a set of q vertices from T leaves p components that allow singularity.

Theorem 3.7. [9, Theorem] *For a simple tree, $M(T) = P(T) = \Delta(T)$.*

Corollary 3.8. *For a symmetric simple ditree, $M(T) = P(T)$.*

Theorem 3.9. [5, Theorem 2.1] *For a loop tree, $M(T) = \mathcal{C}_0(T)$.*

In [2], Algorithm 2.4 and Theorem 2.8 of [5] were used to establish the following result.

Theorem 3.10. [2, Theorem 5.6] *For a loop tree or symmetric loop ditree, $M(T) = Z(T) = P(T) = \mathcal{C}_0(T)$.*

For a directed tree that is not symmetric, a parameter involving deletion of vertices seems to be less useful, since information about arc direction is lost in the deletion process. However, results about symmetric directed trees can be used to prove a more general result.

Theorem 3.11. [2, Theorem 5.8] *For a loop ditree, $M(T) = Z(T) = P(T)$.*

In [2], the crucial step in deducing Theorem 5.8 from Theorem 5.6 was Lemma 5.7. The proof of Lemma 5.7 uses triangle number, which is closely related to zero forcing number for loop graphs and loop digraphs. This proof can be adapted to use the zero forcing number to establish the analogous result for simple ditrees.

Lemma 3.12. *Let H_1 and H_2 be disjoint simple digraphs, let $x_i \in H_i, i = 1, 2$, let H be the simple digraph having the two vertices x_1, x_2 and the single arc (x_2, x_1) , and let $G = H_1 \cup H_2 \cup H$. Let H'_1 be obtained from H_1 by deleting all arcs of the form $(v, x_1), v \in V(H_1)$, and H'_2 be obtained from H_2 by deleting all arcs of the form $(x_2, u), u \in V(H_2)$. If $M(H_i) = Z(H_i), i = 1, 2$ and $M(H'_i) = Z(H'_i), i = 1, 2$, then $M(G) = Z(G)$.*

Proof. Observe that

$$\text{mr}(H_1) + \text{mr}(H_2) \leq \text{mr}(G) \leq \text{mr}(H_1) + \text{mr}(H_2) + 1$$

or equivalently,

$$M(H_1) + M(H_2) - 1 \leq M(G) \leq M(H_1) + M(H_2)$$

If Z_i is a zero forcing set for H_i then $Z_1 \cup Z_2$ is a zero forcing set for G , so $Z(G) \leq Z(H_1) + Z(H_2)$. Thus if $M(G) = M(H_1) + M(H_2)$, then $M(G) = Z(G)$.

So we assume

$$M(G) = M(H_1) + M(H_2) - 1.$$

Assume that the vertices of H_1 are numbered $1, \dots, k$, $x_1 = k$, the vertices of H_2 are numbered $k + 1, \dots, n$, and $x_2 = k + 1$. For any matrix $A \in \mathcal{Q}(G)$, without loss of generality we may assume the nonzero entry associated with the arc $(k + 1, k)$ is 1, so

$$A = \begin{bmatrix} A_1 & O \\ E_{1k} & A_2 \end{bmatrix}$$

where $A_i \in \mathcal{Q}(H_i), i = 1, 2$. If $\text{null}(A_i) = M(H_i), i = 1, 2$, then $M(G) = \text{null}(A_1) + \text{null}(A_2) - 1$, so $\text{mr}(G) = \text{rank}(A_1) + \text{rank}(A_2) + 1$. If e_k^T is in the row space $\text{RS}(A_1)$ or e_1 is in the column space $\text{CS}(A_2)$, then we have the contradiction that $\text{rank}(A) = \text{rank}(A_1) + \text{rank}(A_2) < \text{mr}(G)$. Thus $e_k^T \notin \text{RS}(A_1)$ and $e_1 \notin \text{CS}(A_2)$. This implies that the last column of A_1 is in the span of the remaining columns of A_1 , and similarly the first row of A_2 is in the span of the remaining rows of A_2 .

We claim that $\text{rank}(A_1) = \text{mr}(H_1)$. If not, we can construct a matrix A of rank $\text{mr}(H_1)$ by starting with a minimum rank realization of H_1' and appending a (necessarily) independent column having nonzero entries exactly at the in-neighbors x_1 in H_1 . Such an A would have rank $\text{mr}(H_1)$, and yet its last column would not be in the span of the remaining columns of A_1 . Similarly, $\text{rank}(A_2) = \text{mr}(H_2)$.

Thus for $i = 1, 2$, $\text{mr}(H_i') = \text{mr}(H_i)$, so $M(H_i') = M(H_i)$, and thus $Z(H_i') = Z(H_i)$. Choose a zero forcing set Z_i for H_i' . By construction of H_2' , Z_2 is a zero forcing set in which x_2 does not perform a force. By construction of H_1' , Z_1 is a zero forcing set that does include x_1 . Then $Z = Z_2 \cup Z_1 \setminus \{x_1\}$ is a zero forcing set for G , because Z_2 can force all of the vertices of H_2 , then (in G) x_2 can force x_1 , and finally Z_1 can force the vertices of H_1 . Since $|Z| = Z(H_1) + Z(H_2) - 1$, $Z(G) \leq Z(H_1) + Z(H_2) - 1 = M(H_1) + M(H_2) - 1 = M(G)$. \square

As was done in [2], Lemma 3.12 could be stated for matrices over any field rather than over the real numbers.

Theorem 3.13. *If T is a simple ditree, then*

$$M(T) = Z(T) = P(T).$$

Proof. We prove $M(T) = Z(T)$ for simple diforests. Note first that the theorem is true for any symmetric simple diforest by Corollary 3.8, and for any simple diforest of order at most 2 by direct examination. Assume true for every simple diforest of order less than $|T|$. If T is symmetric we are done; if not T has a strongly connected component S that is joined to $T - S$ by exactly one arc (x_2, x_1) . Let T_1 be the component containing x_1 in $T - x_2$ and let T_2 be the component containing x_2 in $T - x_1$. Let T_1' be the simple diforest obtained from T_1 by deleting all in-neighbors of x_1 , and let T_2' be obtained from T_2 by deleting all out-neighbors of x_2 . Note that $|T_i| < |T|, i = 1, 2$, so by the induction hypothesis $M(T_i) = Z(T_i)$ and $M(T_i') = Z(T_i')$ for $i = 1, 2$. Apply Lemma 3.12 to conclude $M(T) = Z(T)$. \square

Corollary 3.14. *For a tree of any type, $M(T) = Z(T) = P(T)$.*

4 Computation of maximum nullity of trees

The equality $M(T) = Z(T)$ for any type of tree changes the problem of computing $M(T)$ from optimizing over an infinite set of matrices to optimizing over a finite number of subsets of the vertices of the graph. The zero forcing number can be computed for trees of all types by brute force computer programs, and such software is available, e.g., [8]. However, such programs run in reasonable time only for trees of modest size. For example, the (faster) Cython program for simple trees [8] usually runs almost instantly for trees of order less than 20 on an ordinary laptop computer, and usually runs in less than fifteen minutes for trees of order less than 30. Of course, the time required depends the zero forcing number as well as the order of the tree.

For a simple tree T , the preferred method of computation for maximum nullity is to compute either $P(T)$ or $\Delta(T)$. Efficient algorithms are given in [10] and [6]. Such algorithms are very easy for a human being to implement - for one simple tree T having $|T| = 28, Z(T) = 12$, it took the author about two minutes to compute $\Delta(T)$ using Algorithm 2.5 in [6], whereas it took the Cython program [8] six minutes. Algorithm 2.3 in [6] for computing $\Delta(T)$, which uses the tree induced by the vertices of degree at

least three, may be more suitable for computer implementation than Algorithm 2.5, which uses pendent generalized stars and is highly visual. Given a set Q of vertices computed by Algorithm 2.5 whose deletion realizes $\Delta(T)$, it is immediate to construct a minimum path covering.

For a loop tree T or symmetric ditree, the preferred method for computation of maximum nullity is to compute $\mathcal{C}_0(T)$. Algorithm 2.4 in [5] computes a set of vertices Q realizing $\mathcal{C}_0(T)$ for a loop tree (Algorithm 5.4 in [2] is essentially the same algorithm stated for symmetric ditrees). This algorithm is related to Algorithm 2.5 in [6] that computes $\Delta(T)$ (working from the outside in), and can be implemented by human beings, but is considerably more challenging than the algorithm for $\Delta(T)$, as the algorithm for $\mathcal{C}_0(T)$ requires testing each component to determine whether it allows singularity. Given a set Q of vertices realizing $\mathcal{C}_0(T)$, a minimum path covering can be constructed in relatively straightforward manner.

For a (simple or loop) ditree, one can easily find a path cover from the minimum path covers for the strong components (which are symmetric ditrees) by joining to the extent possible the paths in the strong components. This produces an upper bound on $P(T)$, but not necessarily the exact value, since whether the paths can be joined to reduce the number depends on which paths are in the path cover, and unfortunately choices among path covers are involved.

It would be useful to have a more efficient algorithm for computation of path cover number of a (simple or loop) ditree. One possibility for developing such an algorithm would be to follow the methods of proof used in several of the proofs given here: Following Theorem 3.13, work from the outside in, picking off one strong component at a time, obtaining minimum zero forcing sets for the strong components efficiently by using the method of proof in Theorem 3.5 while giving preference when possible to those that as in Lemma 3.12 exclude x_2 and include x_1 (where (x_2, x_1) is the connecting arc).

Acknowledgments: Much of this paper is based on the author’s plenary lecture at the 15th ILAS Conference in Cancun, Mexico in June 2008 and summarizes the work of many people, who are mentioned in the text and/or listed in the references; the author thanks the organizers of the meeting for their invitation and support. The new results were inspired by the work done at the American Institute of Mathematics (AIM) SQuaRE “Minimum rank of matrices described by a graph;” the author thanks AIM and NSF for their support.

References

- [1] AIM Minimum Rank – Special Graphs Work Group (F. Barioli, W. Barrett, S. Butler, S. M. Cioabă, D. Cvetković, S. M. Fallat, C. Godsil, W. Haemers, L. Hogben, R. Mikkelsen, S. Narayan, O. Pryporova, I. Sciriha, W. So, D. Stevanović, H. van der Holst, K. Vander Meulen, A. Wangsness). Zero forcing sets and the minimum rank of graphs. *Linear Algebra and Its Applications*, 428: 1628–1648, 2008.
- [2] F. Barioli, S. Fallat, D. Hershkowitz, H. T. Hall, L. Hogben, H. van der Holst, B. Shader. On the minimum rank of not necessarily symmetric matrices: a preliminary study. *Electronic Journal of Linear Algebra*, 18: 126–145, 2009.
- [3] F. Barioli, S.M. Fallat, and L. Hogben. Computation of minimal rank and path cover number for graphs. *Linear Algebra and Its Applications*, 392: 289–303, 2004.
- [4] W. Barrett, H. van der Holst and R. Loewy. Graphs whose minimal rank is two. *Electronic Journal of Linear Algebra*, 11: 258–280, 2004.
- [5] L. M. DeAlba, T. L. Hardy, I. R. Hentzel, L. Hogben, A. Wangsness. Minimum Rank and Maximum Eigenvalue Multiplicity of Symmetric Tree Sign Patterns. *Linear Algebra and Its Applications*, 418: 389–415, 2006.

- [6] S. Fallat and L. Hogben. The minimum rank of symmetric matrices described by a graph: a survey. *Linear Algebra and Its Applications*, 426: 558–582, 2007.
- [7] D. Hershkowitz. The Combinatorial Structure of Generalized Eigenspaces - from nonnegative matrices to general matrices, *Linear Algebra and Its Applications*, 302/303: 173–191, 1999.
- [8] Iowa State University minimum rank software. Includes:
Program for calculating bounds on the minimum rank of a graph using *Sage* by L. DeLoss, J. Grout, T. McKay, J. Smith, G. Tims, available at <http://arxiv.org/abs/0812.1616> and at <http://www.aimath.org/pastworkshops/matrixspectrum.html>.
Faster cython *Sage* software for computation of the graph parameter Z , by J. Grout, available from the author of this paper.
- [9] C. R. Johnson and A. Leal Duarte. The maximum multiplicity of an eigenvalue in a matrix whose graph is a tree. *Linear and Multilinear Algebra* 46: 139–144, 1999.
- [10] C. R. Johnson, A. Leal Duarte, C. M. Saiago. Multiplicity Lists for the Eigenvalues of Symmetric Matrices with a Given Graph. In *Handbook of Linear Algebra*. L. Hogben, Editor. Chapman & Hall/CRC Press, Boca Raton, 2007.
- [11] P.M. Nylén, Minimum-rank matrices with prescribed graph, *Linear Algebra and Its Applications* 248: 303–316, 1996.