2023

Partially Filled Latin Squares

Mariam Abu-Adas
Scripps College

Partially Filled Latin Squares

Mariam Abu-Adas

Shahriar Shahriari, Advisor

Christopher Towse, Reader

Submitted to Scripps College in Partial Fulfillment
of the Degree of Bachelor of Arts

March, 2023

Department of Mathematics

Abstract

In this thesis, we analyze various types of Latin squares, their solvability and embeddings. We examine the results by M. Hall, P. Hall, Ryser and Evans first, and apply our understandings to develop an algorithm that the determines the minimum possible embedding of an unsolvable Latin square. We also study Latin squares with missing diagonals in detail.

Contents

Abstract **iii**

Acknowledgments **xi**

1 Introduction **1**
1.1 The Topic 1
1.2 Literature Review 2
1.3 Guide to the Thesis 3

2 Bipartites and Permutation Matrices **5**
2.1 Properties of Bipartite Graphs 5
2.2 Permutation Matrices 6

3 Completing Empty Rows **13**

4 Ryser's Conditions **17**
4.1 What is the Theorem? 17
4.2 Proving the Theorem 20
4.3 Applying Ryser's condition 24

5 Evans Paper **27**

6 Minimum Embedding **31**
6.1 The Algorithm 32
6.2 Examples 34
6.3 Where the Algorithm Might Fail 36

7 Pageant Squares **41**
7.1 Extending to $n+1$ 41
7.2 Extending to $2n$? 46

7.3 Extending to $2n - 1$ 46

8 In the Next Episode... 51

A Source Code 53
A.1 Python Code 53

Bibliography 55

List of Figures

1.1 A 2x2 Latin square that cannot be completed. 2
1.2 Extension of the unsolvable 2x2 to a solvable 3x3 2

2.1 Example of a 3×6 permutation matrix. 6
2.2 A $0-1$ matrix A and its bipartite representation. 7
2.3 The original graph $G(X, Y)$ and its extension to a 2-regular graph $G'(X', Y)$ (with the added vertices and edges in red). . 10
2.4 Two perfect matchings of $G'(X', Y)$ in blue and green, and the corresponding matchings in $G(X, Y)$. 10
2.5 The matchings in G and their corresponding permutation matrices. 11
2.6 A written as the sum of permutation matrices. 11
2.7 A bipartite graph $G(X, Y)$, and one selected matching such that no more matchings can be made. 12
2.8 The graph $G(X, Y)$ from figure 2.7 extended to a regular bipartite graph $G'(X', Y)$, with the 3 perfect matchings in red, blue, and green 12
2.9 The 3 matchings of size $|X|$ from figure 2.7, extracted from the extended graph $G'(X', Y)$ from figure 2.8 12

3.1 A partially filled Latin square with complete and empty rows 13
3.2 The bipartite graph $G(X, Y)$ representing figure 3.1 and its two perfect matchings that represent the added rows. 14
3.3 The matchings of figure 3.2 as complete rows added to the incomplete square in figure 3.1 14
3.4 Partially filled Latin square L. 15
3.5 A bipartite representation of L 15
3.6 A perfect matching found in the graph of L 15
3.7 The next matching in the graph of L 15

3.8 Latin square L with added rows from perfect matchings . . . 16

4.1 Can a filled 2x2 with elements from 1 to 3 be embedded and solved in a 3x3? . . . 17
4.2 $n = 4 \quad t = 6$ . . . 18
4.3 $n = 4, \quad t = 7$ . . . 19
4.4 Attempting to fill T with the element 7. . . . 19
4.5 $n = 4 \quad t = 8$ . . . 20
4.6 A Latin rectangle A which we wish to extend to T of order 6. 24
4.7 The 0-1 matrix A_{01} of A. . . . 24
4.8 The $0-1$ matrix of A and its permutation matrix decomposition. 24
4.9 The complete Latin rectangle that can now be solved . . . 25
4.10 The Latin rectangle embedded and solved in a Latin square. 25

5.1 Filling L with elements of new construction S. . . . 28
5.2 Examples of Evan's constructions with $n = 4$ and $n = 7$. . . . 28

6.1 In incomplete Latin square L of order 4 that can be embedded into a T of order less than 8 . . . 32
6.2 A maximal Latin square L of order 3 . . . 34
6.3 The Latin square L embedded in a square of order 5 . . . 35
6.4 A maximal Latin square of order 4 (Evans' construction) . . 36
6.5 Two Latin squares with the order of the rows permuted. These could be considered equivalent. . . . 37

7.1 Pageant squares of order 4 and 6 that need to be extended to $t = n + 1$. . . . 42
7.2 Pageant squares of order 4 and 6 embedded in squares of order 5 and 7 . . . 44
7.3 The construction in formula 7.1 on a grids of order 4 and 5, with missing values in orange. . . . 45
7.4 An odd case ($n = 5$) that works, but we can't explain why, nor can we generalize it. . . . 45
7.5 The element 1 being added 3 or 2 times to a grid of order 5. x represents where 1 can no longer be added. . . . 47
7.6 Visual representation of the cases in equation 7.2 Light orange: $j \geq i + \left\lfloor \frac{n}{2} \right\rfloor$ Purple: $i < j < i + \left\lfloor \frac{n}{2} \right\rfloor$ Orange: $i - \left\lfloor \frac{n}{2} \right\rfloor \leq j < i$ Light purple: $j < i - \left\lfloor \frac{n}{2} \right\rfloor$ Red: $(i, j) = (\left\lfloor \frac{n}{2} \right\rfloor, \left\lceil \frac{n}{2} \right\rceil$ for odd n . 48
7.7 The construction of equation 7.2 on grids of orders 6 and 9 . 48

7.8 The construction of equation 7.2 on grids of orders 6 and 9 with missing values added in orange. 49
7.9 The Latin square L of order 6 extended to T of order $2n-1=11$. 50

Acknowledgments

I believe my thanks should start with my parents because I would not be here today without them, so thank you for having me. My mother in particular, as my first, and best math teacher, my life coach, my rock, my everything. My father, as my benefactor, my hype-man, my supporter in every endeavor I undertake. I thank you.

Professor Shahriari, thank you for taking me on as a thesis student despite the fact that your schedule for the semester was already at 150% capacity, I hope I did you proud, I could not have imagined doing this thesis with anyone else.

Professor Towse, you should already be well aware of how grateful I am to have had you as an academic advisor, a teacher, and lastly as a friend, for that, I thank you, sir.

Special shout out to Ruby, Mica and Becca for dealing with me throughout this process, I hope you at least got some laughs seeing me spiral into insanity.

Chapter 1

Introduction

1.1 The Topic

Sudoku, a puzzle invented in the year 1979, is a 9x9 grid divided into nine 3x3 sections [Meng and Lu (2011)]. The goal of the game is to fill the grid such that every row, column, and 3x3 section contains every number between 1 and 9 exactly once. In a classic Sudoku game, solutions have both properties of existence and uniqueness. However, it is not obvious how we may guarantee that a partially filled Sudoku grid can be uniquely solved. In order to understand properties of solvability and uniqueness, we address Sudoku's parent grid, the Latin square.

A Latin square of order n is an n by n grid filled with entries from the element list $[1, 2, \ldots n]$ such that each element from 1 through n appears in every column and every row exactly once. Thus, every Sudoku grid is a Latin square of order 9. However, not every Latin square of order 9 is a Sudoku grid. That is because, unlike Sudoku, Latin squares do not have the extra sectioning of the grid into 3x3 squares. So one might say a Latin square is an arbitrarily sized, simplified Sudoku. Or, if we consider the timeline of which grid existed first, Sudoku is more complicated, size-restricted Latin square.

Given a partially filled Latin square, the grid may either be solvable, or not.

1	
	2

Figure 1.1 A 2x2 Latin square that cannot be completed.

For example, if we consider the simple 2x2 Latin square in figure 1.1, we see that, while it is currently adhering to the properties of Latin squares, where no number between 1 and 2 is repeated in a row or column, there is no way to continue filling the grid while still adhering to these rules. So the square in figure 1.1 is *not* solvable. We define partially filled Latin squares that are not solvable as *incomplete Latin squares*.

However, if we simply extend the 2x2 grid into a 3x3, by adding a column to the right and a row on the bottom, we would need to fill this new grid with elements from 1 to 3. In the new extended grid, we ask ourselves again whether or not the grid can be solved. As shown in figure 1.2, the Latin square does indeed become solvable when extended to a 3x3.

1		
	2	

1	3	2
3	2	1
2	1	3

Figure 1.2 Extension of the unsolvable 2x2 to a solvable 3x3

So we have seen that a partially filled Latin square may or may not be solvable. This raises the question, which partially filled Latin squares can be solved? For those that cannot be solved, how much must we extend the Latin square by in order for it to be solvable? Or in other words, if we have an unsolvable Latin square L of order n, can we embed L into a larger grid T of order t such that it becomes solvable? If so what is the minimum value t can be?

Our goal for this thesis is to understand already proven conditions for a Latin square to be embedded and completed, and to develop our own conditions for specific outcomes of the grid.

1.2 Literature Review

Key work conducted on understanding partially filled Latin Squares was primarily conducted between mid 20th century until the beginning of the

21st century. Before delving into Latin squares, we first address P. Hall's Marriage theorem [Hall (1935)] as an important theorem used in future results. Then, M. Hall wrote a paper on the completion of Latin squares made up of complete rows and empty rows, which we will address and prove in chapter 3 [Hall (1945)]. In 1951, Ryser then used the results of both Halls to develop a criterion necessary and sufficient to guarantee that a rectangular grid filled with elements from 1 to t can successfully be embedded in a Latin square of order t [Ryser (1951)]. Following Ryser's work, Evans continued the line of thinking regarding partially filled Latin squares. Evans proved that for an arbitrary partially filled Latin square of order n, the minimum embedding needed to guarantee completion is to a grid of order exactly $2n$ [Evans (1960)]. Of course there exist Latin squares that can be embedded into squares of order less than $2n$. However, what Evans proves is that $2n$ will always be sufficiently large enough, and that there do indeed exist Latin squares that *need* to be extended to an order of $2n$, in other words, the upper bound of $2n$ is sharp. Following this paper, Evans made a conjecture that any partially filled Latin square of order n with at most $n-1$ elements in it will always be solvable. This conjecture was approached by various mathematicians, each working to improve the results of those before them until the conjecture was finally proved by both Andersen and Hilton [Andersen and Hilton (1983)] as well as by B. Smetaniuk [Smetaniuk (1981)]. These are the primary works and results we will be analysing and building on in this thesis.

1.3 Guide to the Thesis

We begin the thesis with a preliminary chapter that goes over properties of bipartite graphs that will prove useful to us when attempting to decipher the proofs of Ryser and Evans works. We also look at permutation matrices and a way we can utilize them to decompose other matrices. We continue by considering a very specific case of Latin squares where each row is either entirely filled or completely empty, and we consider the solvability of such grids according to M. Hall. These types of grids will come up again in future chapters, and so we choose to address them earlier on to be easily referenced later. Once we have developed a complete understanding of these chapters we can move on to stating and proving Ryser's theorem, which gives us the conditions needed to solve grids with specific properties. While Ryser's conditions do not directly answer the question of which partially filled Latin

squares can be solved, his work does offer a very strong foundation on which many other mathematicians have built, and on which we intend to develop our own work. Following Ryser, we will study Evans' work which is a direct application of Ryser's results. That will conclude our examination of other mathematician's works in our own words and we will continue by presenting our own findings. Since Evans merely gives an upper bound for embedding partially filled Latin squares, we created an algorithm to determine the smallest embedding needed to solve a given Latin square and develop a code to implement said algorithm. Following that, we studied a specific case of Latin squares where only the diagonal is missing and considered the range of embedding sized such Latin squares would need.

Chapter 2

Bipartites and Permutation Matrices

We use this chapter as a preliminary one to state important theorems and lemmas that will be relevant in our work moving forward. The statements in this chapter are not explicitly about Latin squares, but rather about bipartite graphs and permutation matrices, which is why we are including them here in the preliminary chapter, rather than introducing them in the chapters where they will be used. We will only provide proofs for some statements, but the reader can feel free to refer to the citations if they wish to explore any particular statement further.

2.1 Properties of Bipartite Graphs

In this section, we state the Marriage theorem along with an important corollary of the theorem. We begin by first addressing some definitions and notations. First, we notate a bipartite graph G as $G = (X, \Delta, Y)$ where X and Y are the two sets of vertices that make up a bipartite graph, and Δ is the edges set. The use of absolute values, such as $|X|$ denotes the size of a set, in this case, $|X|$ is the number of vertices in the set X. Next we also have the notation $N(S)$, where if S is a subset of vertices in X, then $N(S)$ is the set of vertices in Y that are adjacent to the vertices in S. A *matching* is defined as an independent edge set, or in other words, an edge set where no two edges are adjacent. A matching can be thought of as a one-to-one pairing of some vertices. A matching is *perfect* if all the vertices in G are incident to the edge set in the matching (so it is a one-to-one pairing of all the vertices in

the bipartite graph). Lastly, a *regular* bipartite graph is a graph where all the vertices have the same degree. Similarly, a graph is k-regular if every vertex in the graph has degree exactly k.

Theorem 2.1. [*The Marriage Theorem*] [*Hall* (1935)] *Let $G = (X, \Delta, Y)$ be a bipartite graph with $|X| \leq |Y|$. Then G has a matching of size $|X|$ if and only if, for every $S \subseteq X$, we have $|N(S)| \geq |S|$.*

Corollary 2.1. *A regular bipartite graph always has a perfect matching.*

The reader may find a proof of the corollary in Shahriari (2022) on page 596 under problem 11.3.13

2.2 Permutation Matrices

In this section, we will state and prove a lemma that is used in the proof of Ryser's theorem in chapter 4. The reader can feel free to skip this section for now, and return to it once they reach Ryser's theorem, which is when this lemma becomes relevant.

Before we get into the lemma, we first set up some definitions and notations.

We define a 0-1 matrix to be any $m \times n$ matrix with entries of only 0 or 1. A permutation matrix, much like its name suggests, is a 0-1 matrix that acts on a matrix A by permuting its rows or columns (depending on whether it is being left or right multiplied). A matrix L is said to be a permutation matrix if it satisfies the condition that $LL^T = I_m$.

We note that a permutation matrix must have exactly one non-zero entry in every row and no more than one non-zero element in any column. So for a permutation matrix of size $r \times c$ (for $r \leq c$), the matrix L would have exactly r 1s, one in each row, and r of the columns have a 1 in them. Consider figure 2.1 as an example of a permutation matrix.

0	0	0	0	0	1
0	0	0	1	0	0
0	1	0	0	0	0

Figure 2.1 Example of a 3×6 permutation matrix.

We now move on to the lemma itself, which states the conditions necessary to write a $0 - 1$ matrix as the sum of permutation matrices.

Lemma 2.1. [*Decomposition Lemma*] *Consider a* $0-1$ *matrix* A *with dimensions* $r \times c$ *with* $1 \leq r < c$. *Let there be exactly* k 1s *in every row. Let* $M(i)$ *be the number of* 1s *in the* i^{th} *column of* A. *Then if for each* $i \in 1, 2, \ldots c$, *if we have the condition that*

$$k - (c - r) \leq M(i) \leq k,$$

then we we can write A *as the sum of* k *permutation matrices* $L_1 + L_2 + \ldots L_k$.

We wish to show that we can write our matrix A in the lemma as the sum of k permutation matrices. We do that by first representing A as a bipartite graph, and then letting each permutation matrix L_i be a matching in our graph. We claim that that is the equivalent to the desired result of writing A as $L_1 + L_2 + \ldots L_k$.

Proof. First we represent A as a bipartite graph:
Let $G(X, Y)$ be a bipartite graph where X represents the rows of A and Y represents the columns of A. We note that $|X| = r$ and $|Y| = c$, as that is the number of rows and columns in A. We denote the elements of X and Y as $x_1, x_2, \ldots x_r$ for the r rows and $y_1, y_2, \ldots y_c$ for the c columns of A. Then for every non-zero entry $a_{i,j}$ in the matrix A, we add an edge between the vertices x_i and y_j. So for example if we have a matrix A with $a_{2,4} = 1$, then we would add an edge between the vertices of x_2 and y_4. The number of edges added is therefore equal to the number of 1s in the matrix. Subsequently, the degree of a vertex, denoted $d(x_i)$ or $d(y_j)$, is equal to the number of 1s in that row or column.

1	0	0	0	0	1
0	0	1	1	0	0
0	1	0	0	1	0

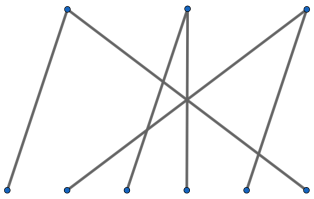


Figure 2.2 A $0-1$ matrix A and its bipartite representation.

Given the conditions of this lemma, we know the following bits of

information:

$$\begin{aligned} 1 \leq r < c \quad &\Rightarrow \quad 1 \leq |X| < |Y| \\ k \text{ 1s in a row} \quad &\Rightarrow \quad d(x_i) = k \\ k - (c - r) \leq M(i) \leq k \quad &\Rightarrow \quad k - (c - r) \leq d(c_i) \leq k \end{aligned}$$

Our goal is to find k disjoint matchings of size $|X|$, where each matching will represent one of the k permutation matrices L_i.

Strategy
First: We will add $c - r$ vertices to X, call them $z_1, z_2, \ldots z_{c-r}$ so that we have $|X| = |Y|$.
Second: We will add k edges to each of these vertices in a manner to make our graph G becomes a regular graph. We will denote the extended graph with the additional edges and vertices as $G'(X', Y)$. We will need to prove that we can actually create $G'(X', Y)$ so that it is regular.
Third: Once we have a regular bipartite graph $G'(X', Y)$, we use corollary 2.1 which allows us to find k perfect matchings.
Fourth: We then get rid of the new vertices that we added ($z_1, \ldots z_{c-r}$), and their respective edges. We now have k matchings from $|X|$ to $|Y|$ which we will use to represent the k permutation matrices $L_1, L_2 \ldots L_k$.

The only thing we need to prove is step 2. We need to show that we can add $c - r$ vertices to X and $(c - r)k$ edges to said vertices in such a way that we can turn our graph G into a $k-$regular graph G'.

The goal is to add one vertex at a time, and add k edges while maintaining the property that $k - (c - r) \leq d(y_i) \leq k$. We first show that we can add k edges without exceeding the upper bound $d(c_i) \leq k$. Second, we note that when we add a vertex, we are really increasing the value of r by one (as a vertex in X represents a row in A, and r is the number of rows in A). Thus, we need to check that all the vertices in Y have degrees that meet the new lower bound $k - (c - (r + 1))$.

Claim 2.1. *We can add a vertex in X and k edges to said vertex in such a way that we maintain the upper bound $d(y_i) \leq k$.*

Proof. Let a represent the number of vertices in Y with degree less than k. We recall that $|Y| = c$, so there are $c - a$ vertices in Y with the maximum degree k. Counting the number of edges we have from X, we know that

each x_i has degree k, so there are $k * |X| = k * r$ edges coming out of X. Counting the number of edges coming out of Y, we know there are $(c - a)$ vertices of degree exactly k, and a vertices with degree at least $k - (c - r)$. So the number of edges coming out of Y is at least $(c - a)(k) + (a)(k - (c - r))$. Therefore we have the following inequality:

$$\begin{aligned} kr &\geq (c - a)(k) + (a)(k - (c - r)) \\ kr &\geq kc - ka + ka - a(c - r) \\ a(c - r) &\geq k(c - r) \end{aligned}$$

And since we know that $r < c$, that tells us that $c - r > 0$ and we can conclude that

$$a \geq k.$$

So the number of vertices in Y with incomplete degree (a degree less than k) is greater than the number of edges we are adding to our new vertex. That means that we can add the k edges without exceeding the maximum degree $d(y_i) \leq k$. □

Now we check that the lower bound still holds. Once we have added the new vertex, the value of r increased by one, and so our lower bound on $d(y_i)$ has changed from $k - (c - r)$ to $k - (c - (r + 1))$, and we need to check that we can add the edges while maintaining this new lower bound.

Claim 2.2. *We can add a vertex in X and k edges to it in such a way that we maintain the new lower bound $k - (c - (r + 1))$.*

Proof. Let b denote the number of vertices in Y with minimum degree $k - (c - r)$. Then $c - b$ is the number of vertices with degree greater than the minimum. We again count the number of edges coming out of X and then from Y. From X we have kr edges again, and from Y we have b vertices with minimum degree $k - (c - r)$, and the remaining $c - b$ vertices have degree less than or equal to k. So the number of edges total must be less than or equal to $(b)(k - (c - r)) + (c - b)(k)$. We again set up an inequality:

$$\begin{aligned} kr &\leq (b)(k - (c - r)) + (c - b)(k) \\ kr &\leq bk - b(c - r) + ck - bk \\ b(c - r) &\leq k(c - r) \\ b &\leq k. \end{aligned}$$

So the number of vertices with minimum degree is less than or equal to the number of edges we add, so we know that we can add our k edges and increase the degree of all the vertices that had a minimum degree $k-(c-r)$, so that their degree is now at least $k-(c-(r+1))$. □

We can now extend our bipartite graph G such that $|X| = r+1$ with $d(x_i) = k$ and $k-(c-(r+1)) \leq d(y_i) \leq k$. By repeating this process until $|X| = |Y|$, we successfully extend G into a $k-$regular graph G', and that completes step 2. The extension of G to a regular bipartite graph G' can be seen in figure 2.3

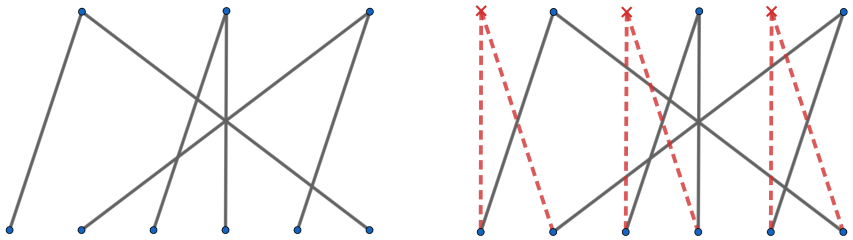


Figure 2.3 The original graph $G(X,Y)$ and its extension to a 2-regular graph $G'(X',Y)$ (with the added vertices and edges in red).

We know that k-regular graphs have perfect matchings by corollary 2.1, therefore we have exactly k distinct perfect matchings in G'. This is step 3, and can be seen in figure 2.4 on the left. after finding the matchings in G', we consider the matchings restricted to only the elements in our original graphs G. This is step 4, and can be seen in the figure on the right in figure 2.4.

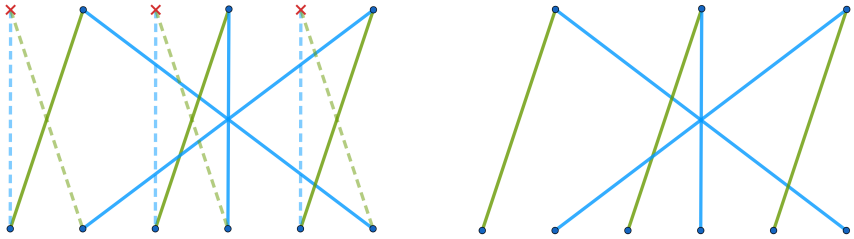
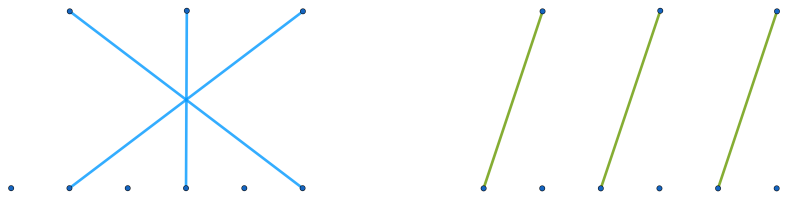


Figure 2.4 Two perfect matchings of $G'(X',Y)$ in blue and green, and the corresponding matchings in $G(X,Y)$.

We convert each matching into an $r \times c$ permutation matrix L_i. In a matching M_i, we have k edges connecting each of the r vertices in X to a

subset of size r of Y. For each edge in the matching connecting x_i to y_j, we add a 1 in the i, j entry of our permutation matrix, with the remaining entries being 0. We can see how this is done in figure 2.5, where the two matchings are in blue and green



0	0	0	0	0	1
0	0	0	1	0	0
0	1	0	0	0	0

1	0	0	0	0	0
0	0	1	0	0	0
0	0	0	0	1	0

Figure 2.5 The matchings in G and their corresponding permutation matrices.

Each permutation matrix then has exactly one 1 in each row, and at most one 1 in every column. The k matchings cover all the edges in our graph, therefore when reverting to matrix representation, the k permutation matrices add up to become our matrix A.

1	0	0	0	0	1
0	0	1	1	0	0
0	1	0	0	1	0

$=$

0	0	0	0	0	1
0	0	0	1	0	0
0	1	0	0	0	0

$+$

1	0	0	0	0	0
0	0	1	0	0	0
0	0	0	0	1	0

Figure 2.6 A written as the sum of permutation matrices.

Thus the proof of the lemma is complete. □

This lemma will serve useful in proving Ryser's condition which we discuss in chapter 4. But first we wish to take the time to elaborate on why we used the strategy that we did to prove this lemma. Given our bipartite graph $G(X, Y)$, one can check that we have all the conditions necessary to use the marriage theorem to find a matching of size $|X|$, so the question becomes, why did we bother adding new vertices to X and creating a regular graph in order to find the matchings? We ask the reader to consider the following example in figure 2.7.

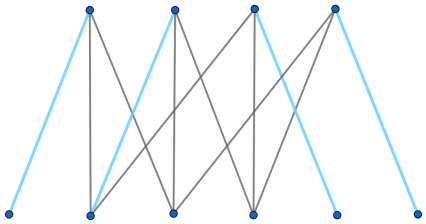


Figure 2.7 A bipartite graph $G(X, Y)$, and one selected matching such that no more matchings can be made.

The reason behind the extra steps taken in our proof is because if we jump straight into the marriage theorem, we are guaranteed to find one matching of size $|X| = 4$ (shown in blue in figure 2.7) But depending on the matching we select, we cannot guarantee being able to find another one, which ends up being the case in the above example. Of the remaining eight edges in grey, there is no way to select a matching of size $|X| = 4$. However, if we began by extending our graph to be a regular graph, we are forced into picking from a set of matchings that allows us to find all k matchings.

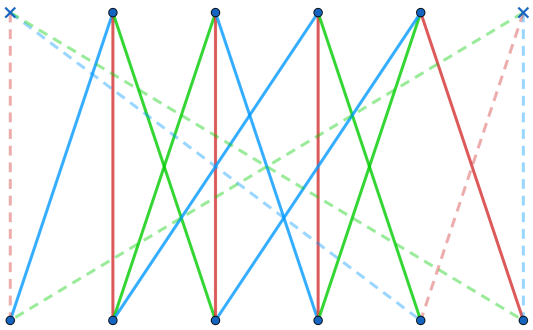


Figure 2.8 The graph $G(X, Y)$ from figure 2.7 extended to a regular bipartite graph $G'(X', Y)$, with the 3 perfect matchings in red, blue, and green

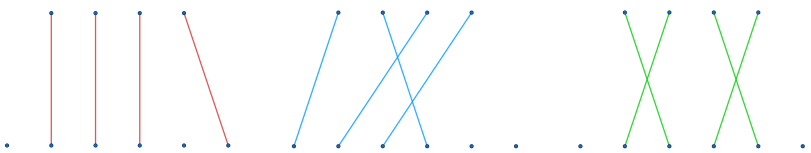


Figure 2.9 The 3 matchings of size $|X|$ from figure 2.7, extracted from the extended graph $G'(X', Y)$ from figure 2.8

Chapter 3

Completing Empty Rows

A simple case of a partially filled Latin squares is that in which our $n \times n$ grid has k complete rows and $n - k$ empty rows. Take for example the square in figure 3.1, which is a square of order 4 with 2 filled and 2 empty rows. The question this chapter aims to answer is: are grids of this form solvable? The results of this chapter will be used in the proof of Ryser's theorem in chapter 4.

1	3	4	2
3	2	1	4

Figure 3.1 A partially filled Latin square with complete and empty rows

Theorem 3.1. [*Row Completion Theorem*][*Hall* (1945)] *Given a Latin square L of order n with k rows completely filled in, and the remaining $n - k$ rows completely empty, we can always complete the Latin square.*

Proof. We create a bipartite graph $G(X, Y)$ that represents our Latin square L of order n. We let $|X| = |Y| = n$. Each element of X represents a column of L, and we denote the vertices in X as $c_1, c_2, \ldots c_n$. The elements of Y represent the values 1 through n, and we denote those vertices $p_1, p_2, \ldots p_n$. We add an edge e between the vertices c_i and p_j if p_j is *not* in the column c_i. So in the example of figure 3.1, the first column c_1 is missing values 2 and 4, thus we add an edge between c_1 and p_2, and another edge between c_1 and p_4. Because each column has exactly k filled rows and $n - k$ empty

rows, that means that every column is missing $n - k$ numbers, and hence the degree of each vertex in $|X|$ is equal to $n - k$. Similarly, each element from 1 through n is missing exactly $n - k$ times, so the degree of every $p_i \in Y$ is also exactly $n - k$. Thus G is an $(n - k)$-regular graph. Since our graph is regular, that implies that we have a perfect matching by the corollary of the marriage theorem [2.1]

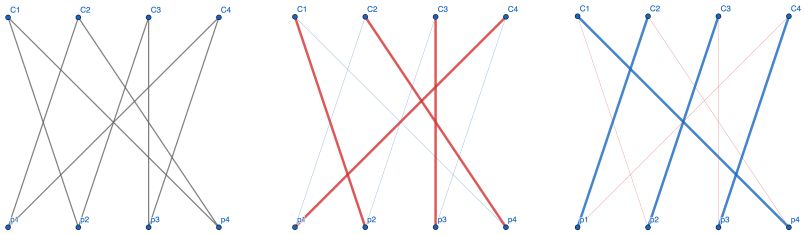


Figure 3.2 The bipartite graph $G(X, Y)$ representing figure 3.1 and its two perfect matchings that represent the added rows.

We claim that a perfect matching represents a row that can be added to the square. For every edge e connecting c_i to p_j, we add the element p_j to the i^{th} column of L. We know that we wont be adding the same element to the row twice because in a perfect matching all edges are non-adjacent. Therefore our perfect matching adds a whole new row to our square. In order to complete all the rows we simply remove the edges that were in the perfect matching (that represented our new row). The graph will still be regular because each vertex lost exactly one edge, and thus we can find another perfect matching in the remaining graph (by the corollary of the marriage theorem) and use it to add another row. The process continues until we find all the perfect matchings and fill in all the rows. This can be seen in figure 3.2 where the first matching is highlighted in red, and the second matching is highlighted in blue. Figure 3.3 below depicts the completed square using the matchings above. □

1	3	4	2
3	2	1	4
2	4	3	1
4	1	2	3

Figure 3.3 The matchings of figure 3.2 as complete rows added to the incomplete square in figure 3.1

Example 3.1. Consider the following partially filled Latin square L of order 7 in figure 3.4:

6	4	7	5	2	3	1
3	5	1	2	6	7	4
2	1	6	7	4	5	3
1	3	2	6	7	4	5
7	2	3	4	5	1	6

Figure 3.4 Partially filled Latin square L.

We represent the information in the Latin square with a bipartite graph $G(X, Y)$ where the c_i points represent the columns of L, and the points p_i represent the elements 1 through 7. This bipartite graph has a perfect matching shown in figure 3.6.

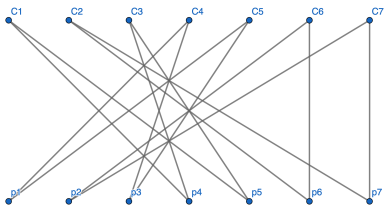


Figure 3.5 A bipartite representation of L

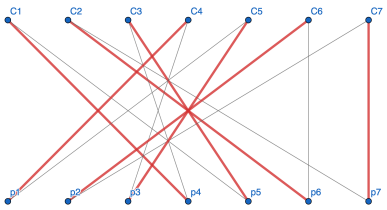


Figure 3.6 A perfect matching found in the graph of L

We use this perfect matching to add a new row onto L. For example, in the perfect matching, the column c_1 is matched to p_4, thus we add the value 4 into the first column. We continue in this manner and fill out the remaining row to give us the red row shown in figure 3.8

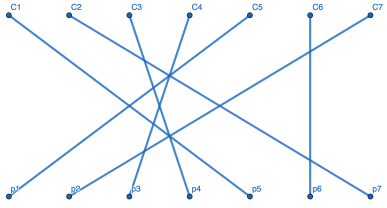


Figure 3.7 The next matching in the graph of L

We also note that if we consider the graph G with the edges of the perfect matching removed, there is another perfect matching to be found as shown in figure 3.7, which again represents another row to be added to the Latin square L, as depicted in blue in figure 3.8.

6	4	7	5	2	3	1
3	5	1	2	6	7	4
2	1	6	7	4	5	3
1	3	2	6	7	4	5
7	2	3	4	5	1	6
4	6	5	1	3	2	7
5	7	4	3	1	6	2

Figure 3.8 Latin square L with added rows from perfect matchings

Chapter 4

Ryser's Conditions

Consider a Latin square L of order n, and some value $t \geq n$. If we fill L entirely with elements from 1 to t under the assumption that we will eventually embed it in a square of order t, how can we guarantee that the embedding will be solvable? We recall the definition of solvability as the ability to complete a grid while adhering to the rules of Latin squares. Take figure 4.1 for example. Given the 2x2 square on the left filled with elements from 1 to 3, how do we know if the 3x3 square on the right can be solved?

1	3
2	1

1	3	
2	1	

Figure 4.1 Can a filled 2x2 with elements from 1 to 3 be embedded and solved in a 3x3?

4.1 What is the Theorem?

Ryser's theorem provides the condition necessary to embed a rectangle filled with the elements 1 to t into a complete Latin square of order t. He does this by utilizing the decomposition lemma 2.1. The goal of this chapter is to understand Ryser's condition and provide a proof of Ryser's theorem in terms of bipartite graphs and perfect matchings.

We begin by stating the theorem and some useful definitions and notations. An *incomplete* Latin rectangle is one is which no number is repeated in any row or column, however it cannot be solved in its current size (especially

since the number of rows and columns aren't equal!). An incomplete Latin rectangle needs to be embedded into a Latin square in order to be solvable.

We denote $N(i)$ as the number of times the element i appears in a Latin square A.

Theorem 4.1. [*Ryser's Condition*] [*Ryser* (1951)] *An incomplete $r \times c$ Latin rectangle A filled with elements from 1 to t can be embedded in a Latin square of order t if and only if $N(i) \geq r + c - t$ for all $1 \leq i \leq t$.*

Ryser's condition rely on counting the number of times each element from 1 to t appears in a Latin square A. The theorem states that we can embed A into a square T of order t if and only if every element from 1 to t appears in A at least $2n - t$ times. Before we begin the proof of the theorem, we attempt to understand how it works by using it on some examples. Consider the following 4×4 square filled on the elements from 1 to 6 in figure 4.2. We wish to check if it can be embedded into a square of size 6, 7 and 8.

$A =$

4	2	5	3
1	3	6	2
6	1	3	4
5	4	2	1

$N(1) = 3$
$N(2) = 3$
$N(3) = 3$
$N(4) = 3$
$N(5) = 2$
$N(6) = 2$

$T =$

4	2	5	3		
1	3	6	2		
6	1	3	4		
5	4	2	1		

Figure 4.2 $n = 4$ $t = 6$

Ryser's theorem states that if every element from 1 to 6 appears in A at least $2n - t = 2 * 4 - 6 = 2$ times, then we can successfully embed A into T. Since we do indeed have $N(i) \geq 2$, by Ryser's theorem we can successfully do this embedding. The reader may feel free to try and complete the square to convince themselves that it is indeed solvable in a 6×6.

If, however, we alter A in figure 4.2 by replacing the 6 in the second row with a 4 instead, then we would have $N(6) = 1$, and by Ryser's condition, we would not be able to embed A into a 6×6.

Going back to the original example in figure 4.2 on the left, since we know we can put A in a 6×6, it seems intuitive to think we can therefore put A in a 7×7, since that is just a bigger square, however, that is not the case.

$A =$

4	2	5	3
1	3	6	2
6	1	3	4
5	4	2	1

$N(1) = 3$
$N(2) = 3$
$N(3) = 3$
$N(4) = 3$
$N(5) = 2$
$N(6) = 2$
$N(7) = 0$

$T =$

4	2	5	3			
1	3	6	2			
6	1	3	4			
5	4	2	1			

Figure 4.3 $n = 4, \qquad t = 7$

We note here that since we are attempting to embed A into a 7×7, Ryser requires that every element from 1 to 7 appears in A a certain number of times ($2 * 4 - 7 = 1$ time). However we see that the element 7 appears zero times in A, and thus by Ryser we cannot complete this embedding. Intuitively speaking, we need the element 7 to appear exactly 7 times, once in every row and every column, but once we start placing the 7s it is clear to see in figure 4.4 that we cannot add 7 more than 6 times without breaking the rules of Latin squares. By understanding why we can't embed A into a 7×7, we are also starting to develop an understanding of what exactly Ryser's condition is checking, and why it works.

4	2	5	3	7		
1	3	6	2		7	
6	1	3	4			7
5	4	2	1			
7						
	7					
		7				

Figure 4.4 Attempting to fill T with the element 7.

Now what if we attempt to embed A into an 8×8?
It may seem like we will run into the same problem as above, where the elements of 7 and 8 do not appear in A, however, if we refer back to Ryser's condition, he requires that all i between 1 and 8 appear at least $2n - t = 2 * 4 - 8 = 0$ times. So technically even if 7 and 8 dont appear here,

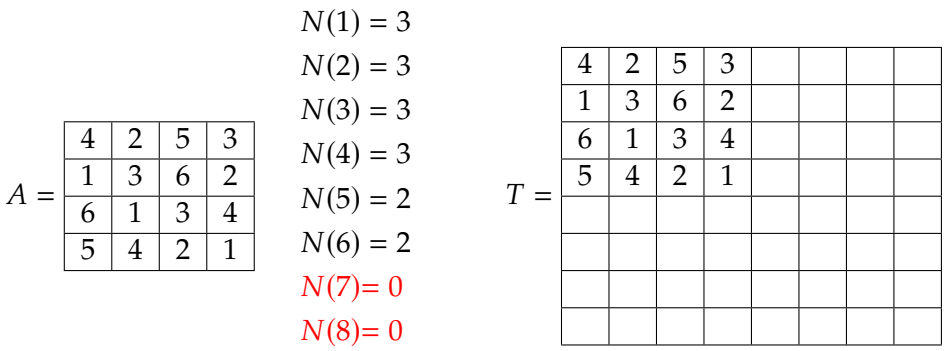


Figure 4.5 $n = 4 \qquad t = 8$

Ryser's condition hold and the embedding is possible. We encourage the reader to try and place the elements of 7 and 8 a total of eight times each in the remaining empty spaces in T. They will find that it is possible, and they will not run into the same issue that we ran into in figure 4.4. The reader may then challenge themselves to filling T completely, knowing that it is in fact possible.

4.2 Proving the Theorem

Now that we understand what Ryser's theorem says, we aim to prove it. In order to do this we must use the decomposition lemma 2.1. The lemma claims that we can write a 0-1 matrix that follows certain properties as the sum of permutation matrices. We restate both the lemma and the theorem here for the benefit of the reader, and then we tackle the proof.

Lemma 2.1 *[Decomposition Lemma] Consider a* $0-1$ *matrix* A *with dimensions* $r \times c$ *with* $1 \leq r < c$*. Let there be exactly* k 1s *in every row. Let* $M(i)$ *be the number of* 1s *in the* i^{th} *column of* A*. Then if for each* $i \in 1, 2, \ldots c$*, if we have the condition that*

$$k - (c - r) \leq M(i) \leq k,$$

then we we can write A *as the sum of* k *permutation matrices* $L_1 + L_2 + \ldots L_k$*.*

Theorem 4.1 *[Ryser's Condition] [Ryser (1951)] An incomplete* $r \times c$ *Latin rectangle*

A filled with elements from 1 *to* t *can be embedded in a Latin square of order* t *if and only if* $N(i) \geq r + c - t$ *for all* $1 \leq i \leq t$.

Strategy

Given an $r \times c$ incomplete Latin rectangle A made up of elements $1, 2, \ldots t$, we wish to show that we can embed A into a square matrix T of size $t \times t$ (such that T is a solvable Latin square) if and only if $N(i) \geq r + c - t$ for all i between 1 and t.

The general strategy for this proof is twofold. First, we extend our $r \times c$ rectangle to be $r \times t$ such that every row has every element from 1 to t. Second, we can add $t - r$ empty rows to our matrix to make it a square. We then have a partially filled Latin square with exactly $t - r$ empty rows, which we know can be completed by the row completion theorem 3.1.

Notation

We will provide some notation and follow it with an example Latin square A, along with examples of each notation being used.

$N(i) =$	the number of times i appears in A
$T_i =$	the set of elements that appear in the i^{th} row of A
$S_i = [t] \setminus T_i$	the set of elements that do not appear in the i^{th} row of A
$M(i) = r - N(i)$	the number of times i appears in the union of the S_i sets
$k = t - c$	the number of columns we need to extend A by

$$\text{A} = \begin{array}{|c|c|c|c|} \hline 4 & 2 & 5 & 3 \\ \hline 1 & 5 & 6 & 2 \\ \hline 6 & 1 & 3 & 4 \\ \hline \end{array}$$

In A, the number of rows is $r = 3$, and the number or columns is $c = 4$. We let $t = 6$, which means we want to extend A to be 6×6.

The number of times the element 1 appears in A is $N(1) = 2$. The set of elements that appear in the second row is $T_2 = \{1, 2, 5, 6\}$. The set of elements that *do not* appear in the second row is $S_2 = \{3, 4\}$. The number of times the element 1 appears in the union of the sets S_i is just $M(1) = 1$. This is also the number of times the number 1 must be added so that it appears once in every row once, so $M(1) = r - N(1) = 3 - 2 = 1$. Lastly, we let k be the difference between c and t, thus $k = 6 - 4 = 2$, which is the number of

columns we need to add to A to make it have dimensions $r \times t$.

Proof. ($\Rightarrow$)
We prove the first direction, that if A can be embedded into T, then the Ryser condition holds.

We note that if the embedding is possible, then $M(i)$ must be at most k, as if $M(i) \geq k$ then we need to add the element i more times than the number of columns we are extending A by. This would force us to add i to the same column more than once which breaks the rules of Latin squares. So we know that $M(i) \leq k$. We also know that $M(i) + N(i) = r$. We conduct the following computation:

$$\begin{aligned} N(i) &= r - M(i) \\ &\geq r - k \\ &= r - (t - c), \\ \text{hence, } N(i) &\geq r + c - t \end{aligned}$$

Thus we proved the first direction of the theorem, that if A can be embedded into a Latin square T of order t, then we have $N(i) \geq r + c - t$.

($\Leftarrow$)
We now prove the other direction of this theorem: if Ryser's condition holds for each i, then A can be embedded into T.

We construct a new matrix $A_{01} \in M_{r \times t}$ made up of 0s and 1s. The construction goes as follows: for a row i in A, we consider the set of numbers that do not appear in that row, which we denoted S_i. First we acknowledge that the size of S_i is equal to k, as a row i already has c distinct elements out of a total of t elements, and $k = t - c$. Say $S_i = \{s_1, s_2, \dots\ s_k\}$, then we add a 1 in the $s_1, s_2, \dots, s_k$ columns of the i^{th} row of A_{01}. So for example, say the 5th row of A is missing the numbers 2, 3 and 8, then we add the element 1 in the 2nd, 3rd, and 8th columns of the 5th row of A_{01}. We fill in the rest of the row with the element 0. We note two facts: the first is that there will be exactly k 1s in each row of A_{01}. The second fact is that the number of 1s that appear in the j^{th} column of A_{01} is equal to the number of times that the number j is missing from our matrix A. Hence the number of 1s in the j^{th} column is equal to $M(j)$. We preform one last counting argument and show that $k - (t - r) \leq M(i) \leq k$.
We prove the right side of this inequality.

We know that $N(i) = r - M(i)$ and by assumption $N(i) \geq r + c - t$

$$\begin{aligned} N(i) &\geq r + c - t \\ r - M(i) &\geq r + c - t \\ M(i) &\geq t - c \\ M(i) &\leq k \end{aligned}$$

Next we wish to prove that $k - (t - r) \leq M(i)$.
First we note that the number of times a number i appears in a row cannot be more than the number of columns in A. Thus $N(i) \leq c$. And again we use the fact that $N(i) = r - M(i)$.

$$\begin{aligned} N(i) &\leq c \\ r - M(i) &\leq c \\ r - M(i) &\leq t - (t - c) \\ r - M(i) &\leq t - k \\ M(i) &\geq k - (t - r). \end{aligned}$$

To summarize the conditions we have proven so far: we have a matrix A_{01} made up of 0s and 1s such that every row has exactly k 1s. We denoted $M(i)$ as the number of 1s that appear in the i^{th} column of A_{01} and we proved that $k - (t - r) \leq M(i) \leq k$. We therefore have all the conditions necessary to use the decomposition lemma 2.1. So we know that we can write A_{01} as the sum of k permutation matrices $A_{01} = L_1 + L_2 + \ldots L_k$. Each of these permutation matrices represent a column that can be adjoined to A. Consider L_1 which has exactly one 1 in every row, let us denote $\{b_1, b_2, \ldots b_r\}$ be the r columns in which the 1s occur in rows $1, 2, \ldots r$ respectively. Then we can add the numbers $b_1, b_2, \ldots b_r$ in that order as a new column of A. We do this again with the matrices $L_2, L_3, \ldots L_k$. Thus we can adjoin k columns to A to create a Latin rectangle of size $r \times t$ with every element in $[t]$ appearing exactly once in every row. Then as mentioned earlier, we can then easily extend this to a Latin square using our method from the previous chapter. □

4.3 Applying Ryser's condition

With a thorough understanding of the theorem and its uses, we present an examples of the proof's construction.

Example 4.1. Here is the example we have been teasing throughout this chapter in figure 4.6. We begin by first checking Ryser's conditions for $t = 6$. Since each $N(i) \geq 3 + 4 - 6 = 1$, by Ryser, the embedding is possible. We will show how he constructs the embedding.

$$A = \begin{array}{|c|c|c|c|}\hline 4 & 2 & 5 & 3 \\ \hline 1 & 5 & 6 & 2 \\ \hline 6 & 1 & 3 & 4 \\ \hline \end{array}$$

$$\begin{aligned} N(1) &= 2 \\ N(2) &= 2 \\ N(3) &= 2 \\ N(4) &= 2 \\ N(5) &= 2 \\ N(6) &= 2 \end{aligned}$$

Figure 4.6 A Latin rectangle A which we wish to extend to T of order 6.

We first create the $0-1$ matrix A_{01}, which is shown in figure 4.7.

$$A_{01} = \begin{array}{|c|c|c|c|c|c|}\hline 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Figure 4.7 The 0-1 matrix A_{01} of A.

We then use the decomposition lemma to write A_{01} as the sum of permutation matrices. We note that the $0-1$ matrix of A is in fact the one we used in the example of the decomposition lemma 2.1, and we see its decomposition in figure 4.8.

$$\begin{array}{|c|c|c|c|c|c|}\hline 1 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 1 & 1 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 1 & 0 \\ \hline \end{array} = \begin{array}{|c|c|c|c|c|c|}\hline 0 & 0 & 0 & 0 & 0 & 1 \\ \hline 0 & 0 & 0 & 1 & 0 & 0 \\ \hline 0 & 1 & 0 & 0 & 0 & 0 \\ \hline \end{array} + \begin{array}{|c|c|c|c|c|c|}\hline 1 & 0 & 0 & 0 & 0 & 0 \\ \hline 0 & 0 & 1 & 0 & 0 & 0 \\ \hline 0 & 0 & 0 & 0 & 1 & 0 \\ \hline \end{array}$$

Figure 4.8 The $0-1$ matrix of A and its permutation matrix decomposition.

So we have these two permutation matrices that correspond to the following two columns which we will adjoin to our matrix A.

4	2	5	3		
1	5	6	2		
6	1	3	4		

6
4
2

1
3
5

By adding these two columns we get the following

4	2	5	3	6	1
1	5	6	2	4	3
6	1	3	4	2	5

Figure 4.9 The complete Latin rectangle that can now be solved

And from this stage we can fill in the remaining rows using the row completion theorem 3.1.

4	2	5	3	6	1
1	5	6	2	4	3
6	1	3	4	2	5
2	3	4	1	5	6
3	6	2	5	1	4
5	4	1	6	3	2

Figure 4.10 The Latin rectangle embedded and solved in a Latin square.

Chapter 5

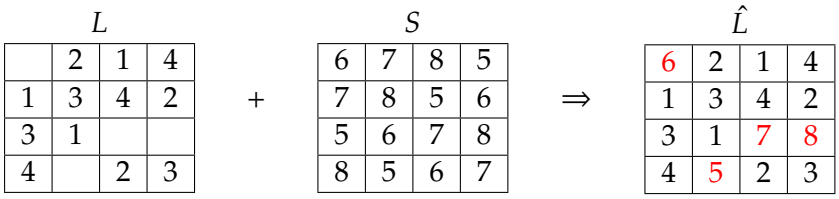
Evans Paper

Following Ryser, Evans works on further understanding the embeddings of incomplete Latin squares, whose definition we recall to be partially filled Latin squares that are not solvable. We emphasize that while Ryser worked with the more general Latin rectangle of size $r \times c$, Evans' work is in fact strictly for squares, where $r = c$. Evans proved that any incomplete Latin square of order n can be extended to a Latin square of any order $t \geq 2n$, and he also proved that that bound is sharp. We will go over his proof and its sharpness in this chapter.

Theorem 5.1 (Evans (1960))**.** *Given an incomplete Latin square L of order n, the square L can be embedded into a Latin square T of order t, given that $t \geq 2n$.*

Proof. Let L be an incomplete $n \times n$ Latin square on the elements of $[n]$. Construct S to be a Latin square of order $t - n$ with elements from the set $\{n + 1, n + 2, \ldots t\}$. We know that the order of S is at least n since $t \geq 2n$ so $t - n \geq n$. We can make the construction of S using our methods from chapter 3. Then for every blank space $L[i, j]$ in our matrix L, we fill that blank spot using the element in $S[i, j]$. In doing so we now filled in all the blank spaces in L with elements from 1 to t, and we call the newly filled Latin square $\hat{L}$. See an example of this in figure 5.1.

We can now apply Ryser's theorem, which states that we can embed our matrix L into a matrix T of order t if and only if $N(i) \geq n + n - t$. But since $t \geq 2n$, Ryser's condition say that the embedding is possible if and only if $N(i) \geq 0$ for all i, which is definitely true. Hence the proof is complete. □



	2	1	4
1	3	4	2
3	1		
4		2	3

6	7	8	5
7	8	5	6
5	6	7	8
8	5	6	7

6	2	1	4
1	3	4	2
3	1	7	8
4	5	2	3

Figure 5.1 Filling L with elements of new construction S.

Next we wish to prove that Evans' bound of $2n$ is sharp. That is not to say that an incomplete Latin square cannot be embedded in any square of order less than $2n$. Rather, if we are given an arbitrary Latin square L of order n, we cannot *guarantee* that L can be embedded in a square of order less that $2n$. Another way of saying this is that there exists a incomplete Latin square that *needs* to be extended to an order of $2n$.

Theorem 5.2 (Evans (1960))**.** *For any $n \geq 4$, there exists a Latin square L of order n that cannot be embedded in a Latin square T of order $t < 2n$.*

Proof. We prove by construction. Let L be a square matrix of order n. We let $L[n-1,2] = 1$ and we leave the spaces $L[n-1,3]$ and $L[n,2]$ as blank entries. Then for every other entry excluding the three that we have just prescribed, we let $L[i,j] = i + j - 1 \pmod{n}$. See examples of such constructions in figure 5.2 .

1	2	3	4
2	3	4	1
3	**1**		2
4		2	3

1	2	3	4	5	6	7
2	3	4	5	6	7	1
3	4	5	6	7	1	2
4	5	6	7	1	2	3
5	6	7	1	2	3	4
6	**1**		2	3	4	5
7		2	3	4	5	6

Figure 5.2 Examples of Evan's constructions with $n = 4$ and $n = 7$.

We claim that such constructions cannot be embedded in a Latin square of order less than $2n$. First we know that we cannot complete the Latin square without extending it, as the element missing from the n^{th} row is the number 1, but the unnocupied cell in the n^{th} row is in the second column, which already has a 1 in it. So the blank $L[n,2]$ cannot be filled with an element

in $[n]$. Similarly, the element missing from the 3^{rd} column is the number 1. But the blank space is in the $n-1$ row, which already has a 1 in it, so that space is also not fillable with any element from 1 to n. So we know that we need to extend L by at least 1, or in other words $t \geq n+1$. That implies that the element $n+1$ needs to be in our matrix $n+1$ times. However, we can only put that element in twice in our original matrix L, as we only have two blank spaces in L. So we still need the element $n+1$ to appear $n-1$ more times, and in order to do that without putting it in the same row or column twice, we need to extend L by at least $n-1$. Therefore we now have $t \geq 2n-1$. However we decide to fill the two unoccupied cells in L with the elements from $\{n+1, \ldots 2n-1\}$, we will still have elements from $[t]$ that never appears in L. Therefore there will be an $N(i) = 0$, and by Ryser, that implies that the minimum t can be is $2n$, and the proof is complete. □

Chapter 6

Minimum Embedding

We know that any partially filled Latin square L of order n can be embedded in a Latin square T of order $t \geq 2n$. However, the question now becomes, what is the minimum t needs to be in order to successfully embed and complete L?

The limitation of Ryser's condition is that he assumes the size of t before constructing L, and in order to apply his theorem, we need to begin with a filled square on the elements from 1 to t. However, if we begin with a partially filled square on only the elements in $[n]$, we have no way of using Ryser's condition unless we already have a value for t and fill in any unoccupied cells in L.

The goal of this chapter is to develop an algorithm that aims to find the minimum value t needs to be to embed a given a maximal Latin square L. A partially filled Latin square L of order n is *maximal* if it is filled with elements from 1 to n, and no new elements can be added to the blank spaces of the square without breaking the rules of Latin squares. In other words, all the unoccupied cells in L must be filled with elements from the set $\{n+1, \ldots t\}$. Take for example figure 6.1. The incomplete Latin square L on the left is indeed maximal (we ask the reader to check this if they would like). We know by Evans that we can embed L into any T of order greater than or equal to 8, but in this specific example we were able to embed L into a T of order 6. So extending L to an order of 8 is not necessary. And if we tried to extend L to T of order only 5, we will find that it fails. So L must be extended to at least order 6. The goal of this chapter is to develop an algorithm that tells us the smallest order a given L can be embedded into.

L

1		4	2
	3	2	
2		1	4
4	2	3	1

T

1	5	4	2	3	6
5	3	2	6	1	4
2	6	1	4	5	3
4	2	3	1	6	5
6	1	5	3	4	2
3	4	6	5	2	1

Figure 6.1 In incomplete Latin square L of order 4 that can be embedded into a T of order less than 8

6.1 The Algorithm

The idea behind the algorithm, which we will name the MWAA Minimizing Algorithm, is to find what $N(i)$ is for all i between 1 and n, and find the minimum occurrence. That will tell us the minimum t can be since by Ryser, we know that $N(i) \geq 2n - t$ or in other words $t \geq 2n - N(i)$. Once we know the minimum t can be, the goal is to add the new elements from $n + 1$ to t as many times as possible each into the remaining blank spaces in L. That is because in order for us to minimize t, we must maximize $N(i)$ for all $1 \leq i \leq t$. Thus the goal of the whole algorithm is based on maximizing $N(i)$.

Algorithm sketch:

1. Begin by letting $t = t_0 = n$.
2. Find the minimum occurrence of the elements in the set $[t]$ in L, denote that m_0. In other words, find $\min\{N(1), N(2), \ldots N(n)\}$
3. Determine the first lower bound that t must be using Ryser's condition, denote that t_1.
4. Consider the number of unoccupied cells in L, denote that d, and the minimum number of times an element from $\{n + 1, \ldots, t_1\}$ must appear in L, denote this m_1.
5. If $m_1 < m_0$ we return to step 1 and this time we let $t = t_1$. We keep going until $m_k \geq m_{k+1}$, in which case $t = t_k$.

Algorithm [The MWAA Minimizing Algorithm]:

Theorem 6.1. *Given a maximal Latin square L of order n, the MWAA minimizing algorithm provides a lower bound for what t needs to be in order to embed L into a Latin square of order t.*

Proof. We are assuming that we have maximized the number of times the elements $\{1, \ldots, n\}$ appear in L, so we consider the element that appears the minimum number of times. Let

$$m_0 = \min(N_L(1), N_L(2), \ldots, N_L(n)).$$

We rewrite Ryser's condition to be in terms of t. For every $i \in [t]$ we know that L can be embedded in a square T of order t, if and only if

$$t \geq 2n - N(i).$$

Ryser's condition applies to a grid that is entirely filled, however, we note that regardless of what happens in the unoccupied cells of L, we know that we have at least one element in $[t]$ that appears only m_0 times. Thus the minimum t can possibly be is $t \geq 2n - m_0$.

Let $t_1 = 2n - m_0$. We now know that we need to add at least $t_1 - n$ new elements to our square L. Again, the aim is to utilize Ryser's condition, so what we care about most is the number of times all these new elements can fit into L. Let d denote the number of empty entries in our square L. If we distribute the $t_1 - n$ new values $(n + 1, n + 2, \ldots t_1)$ as evenly as we can amongst the d blank spaces in L, then each of these elements appears $\frac{d}{t_1 - n}$ times. This fraction is not always an integer, and so we use the floor function to find the number of times each element can be added (assuming we are trying to add each element as many times as we can to maximize all the $N(i)$ values.) Then regardless of how we fill in the new elements of $n + 1$ through t_1 into L, we know that at least one of these elements can appear at most $\left\lfloor \frac{d}{t_1 - n} \right\rfloor$ times in L. We denote this value as the new minimum $m_1 = \left\lfloor \frac{d}{t_1 - n} \right\rfloor$. If $m_1 \geq m_0$, we terminate the algorithm and claim that L can be embedded in T of order $t = t_1$. If, however, $m_1 < m_0$, then we need to re-evaluate the smallest value t can be. So we repeat the process with the values of t_1 and m_1. We know that there exists an element i in the set $\{1, \ldots, t_1\}$ that appears at most m_1 times in L, and thus regardless of how the remaining empty spaces in L get filled, Ryser's condition give us a lower bound for t, which is $t \geq 2n - m_1$. So we let $t_2 = 2n - m_1$. We again check the minimum

occurrence of the new elements from $n+1$ to t_2, and we get a new minimum value $m_2 = \left\lfloor \frac{d}{t_2 - n} \right\rfloor$. We check if $m_2 \geq m_1$, in which case we are done, and we claim that $t = t_2$ is sufficient for the embedding. However, if $m_2 < m_1$, then we reiterate the algorithm. We repeat until we get $m_k \geq m_{k+1}$. We know that we can eventually get to such a case, and the algorithm will terminate, because as the value of t_k is strictly increasing, and soon as $t_k - n > d$, then we have $m_k = \left\lfloor \frac{d}{t_k - n} \right\rfloor = 0$ and in this case the minimum values fix at 0 and the algorithm terminates.

Now all that's left to prove is that when the algorithm terminates, we can in fact embed our square successfully. Intuitively speaking, if the algorithm terminates at t_k and m_k, that implies that every element in $[n]$ appears at least m_k times in L, as $m_0 \geq m_k$, and recall that m_0 was the minimum occurrence of the elements in $[n]$. Similarly, we have that $\left\lfloor \frac{d}{t_k - n} \right\rfloor = m_{k+1} \geq m_k$, which means that if we have enough blank spaces in L to add each element from $n+1$ to t_k at least m_k times each. Therefore, this algothirm provides us with a lower bound of what t must be in order to embed a given Latin square L. □

The only remaining concern however is that having enough space to add the new elements does not imply that we can indeed add these elements in a way that adheres to the rules of Latin squares. We will address this concern in section 6.3.

6.2 Examples

Example 6.1. We begin with an easier example where the algorithm terminates after merely one step. Consider the following example, in figure 6.2

1		3
	2	
3		1

Figure 6.2 A maximal Latin square L of order 3

We ask the reader to first confirm that this Latin square is indeed maximal before proceeding. Once we assure it is in fact maximal we begin

the algorithm. We know the following bits of information:

$$N(1) = 2 \qquad N(2) = 1 \qquad N(3) = 2.$$

Therefore the minimum occurrence value m_0 is equal to 1. According to Ryser, regardless of how the remaining 4 empty spaces are filled with elements from 1 to t, since we have one $N(i)$ value that is equal to 1, thus we know that the minimum t must be is $t \geq 2n - 1 = 5$. We set $t_1 = 5$.

This informs us that we need to add the values of both 4 and 5 to our grid. We know that we have 4 empty spaces and so if we distribute our new values of 4 and 5 to our empty spaces then we can fit each value at least $m_1 = \left\lfloor \frac{d}{t_1 - n} \right\rfloor = \left\lfloor \frac{4}{2} \right\rfloor = 2$ times. So we can fit both values 4 and 5 twice in our original grid L, and this is clear to see when just looking at the square. So assuming we put in the values of 4 and 5 twice each, we get $N(4) = N(5) = 2$ and since $m_1 = 2 \geq 1 = m_0$, our new minimum value is not less than our old minimum, and the t value we currently have, which is $t = 5$ will be sufficient to solve this Latin square. The reader may see how this is done in figure 6.3

1	**4**	3
5	2	**4**
3	**5**	1

1	4	3	2	5
5	2	4	3	1
3	5	1	4	2
2	3	5	1	4
4	1	2	5	3

Figure 6.3 The Latin square L embedded in a square of order 5

We emphasize that the Latin square on the left in figure 6.3 is one that we can apply Ryser's condition to, which is all we needed to know in order to guarantee that we can extend it to the 5×5 square on the right.

Example 6.2. The second example we will explore is one we saw in a previous chapter. We will test the algorithm on the Evans' construction of grids that need to be embedded into squares of order $2n$. Specifically, we will be working with the grid of order 4 in figure 5.2 which we will print again for the convenience of the reader in figure 6.4

1	2	3	4
2	3	4	1
3	1		2
4		2	3

Figure 6.4 A maximal Latin square of order 4 (Evans' construction)

We know before we begin that this square must be embedded into a square of order 8 as in accordance with Evans (detailed in chapter 5).

We begin by counting the occurrences of each element

$$N(1) = 3 \qquad N(2) = 4 \qquad N(3) = 4 \qquad N(4) = 3$$

So we have the minimum occurrence $m_0 = 3$. By Ryser the minimum t can be is $t \geq 2n - m_0 = 5$.

We have one new element to add, and since $d = 2$ we use $m_1 = \left\lfloor \frac{d}{t_1 - n} \right\rfloor = 2$, so the new element can be added at most twice. Therefore the new minimum is $m_1 = 2$ which is less than the old minimum, and we repeat the process. According to Ryser, the minimum t can be now is $t_2 = 2n - m_1 = 6$. That implies that there are two new element that can be added at most $\left\lfloor \frac{d}{t_2 - n} \right\rfloor = 1$ time each. This makes sense, as there are two remaining spaces and 2 new numbers to add, so one of them must appear at most once. Therefore our new minimum is $m_2 = 1$. We repeat, $t_3 = 2n - m_2 = 7$, and $m_3 = \left\lfloor \frac{d}{t_3 - n} \right\rfloor = 0$. And since $m_3 = 0 < 1 = m_2$, we see that our current t value is still not large enough. We try again, now we have $t_4 = 2n - m_3 = 8$, and we have $m_4 = \left\lfloor \frac{d}{t_4 - n} \right\rfloor = 0$. Now we have $m_4 = m_3 = 0$, and since the minimum occurrence of a value is larger than or equal to the previous minimum, that tells us that the t value of 8 will in fact suffice by Ryser's condition.

We developed a code to conduct this algorithm for any given Latin square L. The source code can be found in Appendix A in section A.

6.3 Where the Algorithm Might Fail

In order to better understand where the algorithm might fail, we attempt to construct counter-examples (emphasis on the word *attempt*). The simplest case of a counter-example is if the Latin square L of order n has only 2

unoccupied cells that are in the same row or column, and if the value of t_1 is $n + 1$. If we attempt to use the algorithm on such a scenario, we would be led to assume that the only new element we need to add (that is the element $n + 1$) can appear in our grid at least twice, and thus our new minimum has a value of 2. However, this would not be the case as if our two unoccupied cells are in the same row or column, then the new element $n + 1$ can only be added at most once in the original square L, and that would alter the value of the minimum occurrences that our algorithm is built on. In such a scenario our algorithm does not work as is intended. Upon further investigation we found that this specific scenario is never possible, which we can prove using straightforward combinatorial work.

Theorem 6.2. *There does not exist a maximal Latin square L of order n with exactly two empty cells in the same row.*

Before we dive into the proof, we address some generalizations we need to make in order to complete the proof. We note that the theorem specifies that there is no maximal Latin square with 2 blank spaces in the same row. The reader might ask, what if the two blank spaces were in the same column? To which we respond, that is the exact same question. If we could construct a maximal Latin square with two blank spaces in the same column, then its transpose will have two blank spaces in the same row. And the transpose of a Latin square is also a Latin square. Thus, it is not important for us to distinguish whether the property pertains to the row or the column, because one is merely a reflection of the other. Similarly, two Latin squares where the order of the rows or columns are permuted could be considered equivalent, and it does not benefit us to study each permutation independently. Consider figure 6.5 for an example of row order permutation.

1	2	3
2	3	1
3	1	2

1	2	3
3	1	2
2	3	1

Figure 6.5 Two Latin squares with the order of the rows permuted. These could be considered equivalent.

Thus, squares with permuted rows or columns orders are the same for all intensive purposes, and we will indicate this by utilizing the expression "without loss of generality". With this idea in mind, we continue.

We first attempt to construct a Latin square of small orders to see what goes wrong before attempting to generalize our claim to a Latin square of any order. And for the author's convenience, we will represent the Latin squares as matrices.

Consider a Latin square L of order 3. Without loss of generality, we let the 2 blank spaces in L be in the last row, in the first and last columns. We will highlight the spaces we intend to leave blank by placing a circle in its place.

$$\begin{bmatrix} & & \\ & & \\ \bigcirc & & \bigcirc \end{bmatrix}$$

In order for a space to be blank non trivially, the element missing from the column needs to be *different* from the element missing from the row. Without loss of generality, we place the elements 1 and 2 in the first column. Then in order for the blank space in the first column to be non-trivial, we would need the element 3 to be in the same row as the blank space as seen below.

$$\begin{bmatrix} 1 & & \\ 2 & & \\ \bigcirc & 3 & \bigcirc \end{bmatrix}$$

Then in order for the second blank space on the right to also be non-trivially empty, its column needs to have the elements of 1 and 2 as well.

$$\begin{bmatrix} 1 & & 2 \\ 2 & & 1 \\ \bigcirc & 3 & \bigcirc \end{bmatrix}$$

Once we get to this stage, it is clear to see that the two blank spaces in the second column cannot be filled without breaking the rules of Latin squares. Therefore we could not create a maximal Latin square of order 3 with exactly two empty spaces in the same row.

We continue by considering squares of order 4. We begin in the same manner, let the bottom two corners be the cells we wish to keep empty. Without loss of generality, we add the elements 3 and 4 to the last row as seen below

$$\begin{bmatrix} & & & \\ & & & \\ & & & \\ \bigcirc & 3 & 4 & \bigcirc \end{bmatrix}$$

In order for the cells to be non-trivially empty, we need to add the elements 1 and 2 to both the first and last columns. Without loss of generality, we add 1 and 2 into the first column in the first and second row.

$$\begin{bmatrix} 1 & & & \\ 2 & & & \\ & & & \\ \bigcirc & 3 & 4 & \bigcirc \end{bmatrix}$$

Then regardless of how we add the 1 and 2 to the last column, we find ourselves in a little bit of a pickle regarding how to proceed.

$$\begin{bmatrix} 1 & & & 2 \\ 2 & & & 1 \\ & & & \\ \bigcirc & 3 & 4 & \bigcirc \end{bmatrix} \qquad \begin{bmatrix} 1 & & & 2 \\ 2 & & & \\ & & & 1 \\ \bigcirc & 3 & 4 & \bigcirc \end{bmatrix} \qquad \begin{bmatrix} 1 & & & \\ 2 & & & 1 \\ & & & 2 \\ \bigcirc & 3 & 4 & \bigcirc \end{bmatrix}$$

In all three cases, we still need to add the values of 1 and 2 to the second and third columns in order for us to only have the two blank spaces in the bottom corners. It is very easy to see that there is no way to achieve that without breaking the rules of Latin squares. In summary, we need to add 1 to two different columns, however there is only one row left that does not already have a 1 in it! What a dilemma...

We claim that this particular dilemma will follow us around regardless of the size of the square, which is how we will prove our theorem.

Proof. Consider a Latin square L of order n. We again let the bottom 2 corners be the designated empty cells. Without loss of generality, we fill the remaining cells in the bottom row with the elements from 3 to n. We need the elements of 1 and 2 to be present in both the first and second columns in order for the spaces to be non-trivially empty.

$$\begin{bmatrix} 1 & & \cdots & & 2 \\ 2 & & \cdots & & 1 \\ \vdots & & \ddots & & \vdots \\ \bigcirc & 3 & \cdots & \mathrm{n} & \bigcirc \end{bmatrix}$$

However, in order to ensure there are no other blank spaces other than the two designated ones, the elements of 1 and 2 need to appear in every column. So 1 and 2 each need to appear n times in the n columns, but there are only $n - 1$ rows to put them in, so this construction is impossible.

We point out that in the above example of L with order n that we decided to put the 1 and 2 in the first two rows for both the first and last column, but we emphasize that this choice was arbitrary and that had the reader made different choices as to where to place the 1 and 2 in the first and last columns, they would run into the same dilemma regardless. With that our proof is complete.

□

We implore the reader to understand that we have not yet proven that there does not exist any case that breaks our algorithm, but rather that this specific attempt at a counter-example does not in fact exist. Upon even further investigation, we have yet to find a valid counter-example to this algorithm. Again, this does not suffice to prove that our algorithm always works, but it does suffice for the author of this thesis to leave it as a conjecture and move on with her life for the time being.

Conjecture 6.1. *The MWAA Minimizing Algorithm computes the minimum value t must be in order to embed a given Latin square L into a square T of order t.*

Chapter 7

Pageant Squares

Now that we have this algorithm, we get to play around with it. We wish to investigate partially filled maximal Latin squares L with only the elements of the main diagonal missing. The question then becomes, what is the minimum embedding for such constructions? Does that fact that only the diagonal is missing have any affect on the possible extensions needed? Can we construct squares that need to be extended to $2n$? How about just $n + 1$? The point of this chapter is simply to experiment with a specific form of maximal partially filled squares, and to test out our new and shiny algorithm.

For the fun of it, let us call these Latin squares with missing diagonals *pageant squares* (because they kind of look like they are wearing a pageant sash if you squint hard enough.)

Definition 7.1 (Pageant Square)**.** *A maximal incomplete Latin square L in which $L[i, j] \neq \emptyset$ for $i \neq j$, and $L[i, j] = \emptyset$ for all $i = j$.*

We claim that there exist pageant squares of order n that need to be embedded into a T of order $n + 1$. We also claim that there are pageant squares that need to be extended to at least $t = 2n - 1$. We prove both these things by construction.

7.1 Extending to $n + 1$

7.1.1 $n = 2k$

Theorem 7.1. *For every even number n, there exists a pageant square L of order n that can be embedded into a T of order $t = n + 1$*

Proof. We begin with the construction along with some examples of what the construction looks like.

$$L[i,j] = \begin{cases} j - i \pmod{n} & \text{for } j > i \\ j - (i-1) \pmod{n} & \text{for } j < i \\ \emptyset & \text{for } i = j \end{cases} \tag{7.1}$$

We provide some examples of this construction

	1	2	3
4		1	2
3	4		1
2	3	4	

	1	2	3	4	5
6		1	2	3	4
5	6		1	2	3
4	5	6		1	2
3	4	5	6		1
2	3	4	5	6	

Figure 7.1 Pageant squares of order 4 and 6 that need to be extended to $t = n + 1$.

It is clear to see from our construction that each element from 1 to n appears exactly $n - 1$ times in our grid. We first need to prove that this construction is in fact a pageant square by showing it is maximal. To do that we prove two things: first, that the element j is missing from the j^{th} column, second, that the element i appears in the i^{th} row for every i and j between 1 and n. Since the only unoccupied cells are of the form (i, i) our claims would show that we are missing i from the i^{th} column, but that i is present in the i^{th} row, which means that there is no way to add an element to the cell (i, i).

Claim 7.1. *For all $1 \leq j \leq n$, the element j is missing from the j^{th} column.*

Proof. We prove by contradiction:
When $j > i$, the value of the entries is equal to $j - i \pmod{n}$, and for an entry $j - i$ to equal j we would need i to equal 0 (which is equivalent to $i = n$ since we working in mod n). However, if $i = n$ then it must be the case that $i \geq j$ which is a contradiction.

Similarly, if $j < i$, then the value of the entries is $j - (i - 1) \pmod{n}$, and for this to equal j we would need $i - 1 = 0$, $i = 1$, however if $i = 1$ then it must be the case that $j \geq i$ which contradicts our starting conditions.

Therefore j cannot appear in the j^{th} column. □

Claim 7.2. *For all $1 \leq i \leq n$, the element i appears in the i^{th} row.*

Proof. First for $1 \leq i \leq \frac{n}{2}$, we claim that i appears in the i^{th} row in the upper triangle where $j > i$. The entry values are $j - i$ and we simply solve for $j - i = i$ and get $j = 2i$, so for i values between 1 and $n/2$, the entry $(i, 2i)$ has the element i in it, and since we are restricting i to be less than $n/2$, we know that the value of $2i$ will definitely be greater than i and we are still in the upper triangle where $j > i$.

Then for values of $\frac{n}{2} < i \leq n$, we claim that the element i appears in the i^{th} row in the lower triangle $j < i$. The value of the entries equals $j - (i - 1) \pmod{n}$ and if we set that equal to i we get $j = 2i - 1$. Since $i > \frac{n}{2}$ we know that the value of $2i - 1$ is greater than n, but since we are working in mod n, the value of $2i - 1 \pmod{n}$ is in fact less than i and we remain in the lower triangle of $j < i$. So the entry $(i, 2i - 1 \pmod{n})$ is equal to i, and we have thus shown that for every i, the element i appears in the i^{th} row. □

Therefore since the only unoccupied cells are in the form (i, i) and we know that the only element missing from the i^{th} column is the element i itself, but that i appears in the i^{th} row, this tells us that all the unoccupied cells are in fact non-trivially empty and that our Latin square is maximal.

Now that we have constructed the even case, we must show that this construction can be embedded into a square of order $n + 1$. Doing so is straightforward as we can utilize Ryser's conditions. We know that the empty diagonal cells cannot be filled with any element between 1 and n because L is maximal, so all the diagonal cells must be occupied by the new element $n + 1$. Then once we fill in the diagonal with the element $n + 1$, we now have a grid filled with elements from 1 to $n + 1$ and we can apply Ryser's conditions. Ryser states that if each element between 1 and $n + 1$ appear at least $2n - (n + 1) = n - 1$, times, then the grid can be successfully embedded. Since that is indeed the case now that we have added the element $n + 1$ into the n empty diagonal spaces, we know that we can in fact extend our grid by 1. In figure 7.2, we show how the constructions in figure 7.1 are extended to order $n + 1$. □

5	1	2	3	**4**
4	**5**	1	2	**3**
3	4	**5**	1	**2**
2	3	4	**5**	**1**
1	**2**	**3**	**4**	**5**

7	1	2	3	4	5	**6**
6	**7**	1	2	3	4	**5**
5	6	**7**	1	2	3	**4**
4	5	6	**7**	1	2	**3**
3	4	5	6	**7**	1	**2**
2	3	4	5	6	**7**	**1**
1	**2**	**3**	**4**	**5**	**6**	**7**

Figure 7.2 Pageant squares of order 4 and 6 embedded in squares of order 5 and 7

That completes the construction and embedding of the even case, we now consider the odd case.

7.1.2 $n = 2k + 1$

We first begin by understanding why our construction for the even case does not work in the odd case. In summary, if we use our construction on a Latin square L of odd order, the square will not be maximal. In the even construction, the values missing from columns $1, 2, 3, \ldots, n$ are the values $1, 2, 3, \ldots n$ respectively, however, the values missing from from rows $1, 2, 3, \ldots, n$ are the values $n, n-1, n-2, \ldots, 1$ respectively. So in an empty cell (i, i) the row will be missing $n - (i - 1)$, while the column is missing the value of i. So the row and the column will only ever be missing the same element if $i = n - (i - 1)$, which we can solve to find it only possible when $i = \frac{n+1}{2}$. If n is even, then this i value is not an integer, which implies there is no cell (i, i) where the row and column are missing the same value, and that implies that our grid is maximal. In the odd case however, for the entry $(\frac{n+1}{2}, \frac{n+1}{2})$, which is now an integer, the row and column are both missing the same value, and that blank space in the grid can be filled. Consider figure 7.3. We used the construction in formula 7.1 on grids of orders 4 and 5. In orange we added to the right side of the grid the values missing from each respective row, and below the grid, the values missing from each column. It is clear to see that in the even case, the row and the column are never missing the same value, however in the odd case, the 3^{rd} row and column are both missing the value 3, which means that we can add the element 3 to the position $(3, 3)$ in our grid without any contradictions to the Latin square, so our grid was not maximal. That is why we cannot use the same construction for the odd case.

	1	2	3	4
4		1	2	3
3	4		1	2
2	3	4		1
1	2	3	4	

	1	2	3	4	5
5		1	2	3	4
4	5		1	2	3
3	4	5		1	2
2	3	4	5		1
1	2	3	4	5	

Figure 7.3 The construction in formula 7.1 on a grids of order 4 and 5, with missing values in orange.

In our attempts to find pageant squares of odd degree that need to be embedded in $t = n + 1$, we found that it is in fact impossible to do so for $n = 3$. Proving this was simply a matter of considering all the cases and showing that each case was either not maximal, and thus not a pageant square, or it needed to be embedded into $t = 5$ instead of $t = 4$. The problem seemed to lie in the fact that the number 3 was simply just too small and there aren't enough permutations of 1, 2 and 3 to construct with.

Upon further investigation, we did find a construction for $n = 5$ and we used the minimum embedding algorithm to prove that it can indeed be embedded into a square of order 6. The construction of said square can be found in figure 7.4.

	1	2	3	4
3		4	5	2
4	5		1	3
2	4	5		1
5	3	1	2	

Figure 7.4 An odd case ($n = 5$) that works, but we can't explain why, nor can we generalize it.

The existence of the above pageant square leads us to believe that there may in fact be a construction for odd orders larger than 3, but we have yet to find it. We leave it as a conjecture.

Conjecture 7.1. *For every odd number $n > 3$, there exists a pageant square L of order n that can be embedded into a T of order $t = n + 1$*

7.2 Extending to $2n$?

Theorem 7.2. *A pageant square will always have a minimum embedding of order t less than $2n$.*

Proof. By Ryser's conditions, if L can only be embedded into T of order $t = 2n$ then there must exist some $i \in [t]$ such that $N(i) = 0$. We know that we have n unoccupied cells in L and thus we can place every element from $n + 1$ to t exactly once in L. Thus, if there should be an i such that $N(i) = 0$, it would need to be the case that $i \in [n]$. However, if i does not appear in L any times, then the empty diagonal is trivially unoccupied, because then you could simply add i to every diagonal entry and the square would be complete with no embedding necessary. So we cannot create a pageant square that needs to be extended to $2n$.

□

7.3 Extending to $2n - 1$

Theorem 7.3. *There exists a pageant square L of order n which cannot be embedded into any square T of order $t < 2n - 1$.*

Similar to the first section, we prove this by construction. We first begin with the intuition behind the construction. In the previous section, in order to ensure that our grid only needed to be extended to $n + 1$, we set it up so that every element between 1 and n appeared as many times as possible within our grid L. By doing so, we forced $N(i)$ to be greater than or equal to $n - 1$, for all i between 1 and n, which maximizes the minimum occurring value. If however, we want the minimum embedding to be as large as possible, the goal is to built our grid with a value appearing as few times as possible. Obviously the least number of times we can add an element to the grid is 0 times, however that would cause the missing diagonal entries to be trivially empty, and our grid would not be maximal. So we need to find the minimum number of times an element can appear in our grid while ensuring that the diagonal is not trivially empty.

Instead of aiming to minimize the occurrence of an arbitrary element, we decided to select the element 1 as the value whose occurrence we wish to minimize. Thus moving forward, our construction aims to minimize $N(1)$. We know that for each time we add 1 to a cell (i, j) (where $i \neq j$ because we are not adding entries to the diagonal), the entire column of j and row

i can no longer have the value 1 added to them, so the diagonal cells (i,i) and (j,j) cannot be filled with the element 1. Since each time we add a 1 we can eliminate at most 2 diagonal entries from being filled with 1, we would need at least $\lceil \frac{n}{2} \rceil$ cells to have 1 in them in order to ensure that none of the diagonals could be filled with the element 1.

x	x	1	x	x
x	x	x	1	x
x	x	x	x	1
		x	x	x
		x	x	x

x	x	x	1	x
x	x	x	x	1
		◯	x	x
			x	x
			x	x

Figure 7.5 The element 1 being added 3 or 2 times to a grid of order 5. x represents where 1 can no longer be added.

Take for example figure 7.5, to the grid on the left we added the element 1 in $\lceil \frac{5}{2} \rceil = 3$ cells, and showed that none of the diagonal entries can have a 1 in them. However, if we only add 1 twice, as shown on the grid on the right, then there is bound to be a diagonal entry that can have 1 added to it (in this case it is the cell with the red circle in it), so a grid with less than $\lceil \frac{n}{2} \rceil$ appearances of an element can never be maximal.
Now that we understand the foundation of the construction is ensuring that at least one element appears the minimum number of times, which is $\lceil \frac{n}{2} \rceil$, we can move on to the construction.

$$L[ij] = \begin{cases} j - i - (\lfloor \frac{n}{2} \rfloor - 1) \pmod{n} & \text{for } j \geq i + \lfloor \frac{n}{2} \rfloor \text{ and } i - \lfloor \frac{n}{2} \rfloor \leq j < i \\ j - i - \lfloor \frac{n}{2} \rfloor \pmod{n} & \text{for } j < i - \lfloor \frac{n}{2} \rfloor \text{ and } i < j < i + \lfloor \frac{n}{2} \rfloor \\ n & \text{ONLY if } n \text{ odd for } (i,j) = (\lfloor \frac{n}{2} \rfloor, \lceil \frac{n}{2} \rceil) \\ \emptyset & \text{for } i = j \end{cases} \tag{7.2}$$

In order to understand what the different cases are more clearly, consider figure 7.6, where the 5 different colors are the different cases. Both of the orange sections fall under the first case, the purple sections fall under the second case, and the red square is the 3rd case, unique to grids of odd order. Then consider figure 7.7 to see examples of the construction.

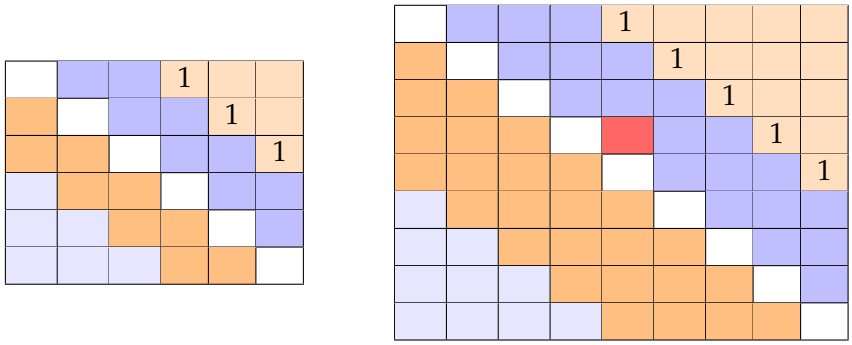


Figure 7.6 Visual representation of the cases in equation 7.2
Light orange: $j \geq i + \left\lfloor \frac{n}{2} \right\rfloor$
Purple: $i < j < i + \left\lfloor \frac{n}{2} \right\rfloor$
Orange: $i - \left\lfloor \frac{n}{2} \right\rfloor \leq j < i$
Light purple: $j < i - \left\lfloor \frac{n}{2} \right\rfloor$
Red: $(i, j) = \left(\left\lfloor \frac{n}{2} \right\rfloor, \left\lceil \frac{n}{2} \right\rceil\right)$ for odd n

	4	5	1	2	3
3		4	5	1	2
2	3		4	5	1
6	2	3		4	5
5	6	2	3		4
4	5	6	2	3	

	6	7	8	1	2	3	4	5
5		6	7	8	1	2	3	4
4	5		6	7	8	1	2	3
3	4	5		**9**	7	8	1	2
2	3	4	5		6	7	8	1
9	2	3	4	5		6	7	8
8	9	2	3	4	5		6	7
7	8	9	2	3	4	5		6
6	7	8	9	2	3	4	5	

Figure 7.7 The construction of equation 7.2 on grids of orders 6 and 9

Now that we have our construction, we can move on to proving this construction does indeed work and needs to be extended to $2n - 1$.

Proof. It is important to first check that the construction we made is maximal, as we did with the previous section. However, that just requires some straightforward computations. For an intuitive understanding of why the grid is maximal, please refer to figure 7.8 to see which values are missing in each row and column (printed in orange).

	4	5	1	2	3	6
3		4	5	1	2	6
2	3		4	5	1	6
6	2	3		4	5	1
5	6	2	3		4	1
4	5	6	2	3		1
1	1	1	6	6	6	

	6	7	8	1	2	3	4	5	9
5		6	7	8	1	2	3	4	9
4	5		6	7	8	1	2	3	9
3	4	5		**9**	7	8	1	2	6
2	3	4	5		6	7	8	1	9
9	2	3	4	5		6	7	8	1
8	9	2	3	4	5		6	7	1
7	8	9	2	3	4	5		6	1
6	7	8	9	2	3	4	5		1
1	1	1	1	6	9	9	9	9	

Figure 7.8 The construction of equation 7.2 on grids of orders 6 and 9 with missing values added in orange.

It is clear to see that a row and column will never be missing the same element. As for the actual rigorous proof that the construction is maximal, the author was kind enough to leave this exercises to her lovely readers, because she has better things to do with her time. Once the readers have convinced themselves that this construction is indeed maximal, either by checking or just having faith in the author, we can now use the algorithm we developed in the previous chapter.

We know by construction that our minimum value m_0 is equal to $\lceil \frac{n}{2} \rceil$, which means the minimum value t can be is $t \geq 2n - \lceil \frac{n}{2} \rceil = n + \lfloor \frac{n}{2} \rfloor$. Then since $d = n$ we find $m_1 = \left\lfloor \frac{n}{n+\lfloor \frac{n}{2} \rfloor - n} \right\rfloor \approx 2$. Since this minimum is smaller than m_0, we keep going. With this new minimum we get that t must be at least $2n-2$, then $m_2 = \lfloor \frac{n}{2n-2-n} \rfloor = 1$. Then since $m_2 < m_1$ we go again. With $m_2 = 1$ we find the minimum t must be is $t \geq 2n-1$ and $m_3 = \frac{n}{2n-1-n} = 1$. Now that we have $m_3 = m_2$ we can stop at $t = 2n-1$. So for this construction, the minimum t can be in order to embed our square is $2n-1$. The only outstanding question now is how do we guarantee that $2n-1$ is large enough? The algorithm just tells us that $2n-1$ is the minimum t can be, but is it possible for this construction needs to be embedded into a grid of order $2n$? No! Because we proved in the previous section that there is no construction of a pageant square that needs to be extended to an order of $2n$.

□

And for the sake of the reader, the author has provided a completed Latin square for this new construction for the square of order 6 that needs to be extended to order 11.

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

Figure 7.9 The Latin square L of order 6 extended to T of order $2n - 1 = 11$.

Chapter 8

In the Next Episode...

We begin with a little recap of what has happened so far in the show. First, we unpacked the works of M. Hall, P. Hall, Ryser and Evans, making sure to understand their works through the lens of bipartite graphs and matchings. Next, we used our understanding of their works to develop an algorithm that finds the minimum order an incomplete Latin square must be embedded into. Lastly, we defined matrices called pageant squares and found the minimum and maximum possible embeddings of these types of squares.

Moving forward, the first thing we would work on next is to try and complete the construction of the odd case for pageant squares that need to be embedded in a square of order $n + 1$.

After that, we would like to consider all the values between $n + 1$ and $2n - 1$ to see if there exist pageant squares that need to be embedded in squares of order t for all $n + 1 < t < 2n - 1$, or if there are certain values that cannot be achieved. Those two goals would be enough to quench our questions about pageant squares.

Our next goal would be to try and support our algorithm further. Our algorithm currently gives us a lower bound for what t must be for an embedding. We would like to show that the algorithm does in fact give the minimum embedding rather than just a bound. We would go about this by either attempting to find counter-examples, and finding ways to adjust our algorithm to account for such cases, or by proving that no such counter examples exist, and that the algorithm works perfectly as is.

Lastly we would like to go back to the questions that inspired this the-

sis paper to begin with, which is to understand uniqueness of the solutions to Latin squares, in addition to their solvability, which is what this paper aimed to address.

Appendix A

Source Code

A.1 Python Code

```
import numpy as np
from collections import Counter
from itertools import chain
from math import*
import copy

# The Latin square is defined by L
# we use 0 to denote empty cells

L = np.array([[1, 0, 3],
              [0, 2, 0],
              [3, 0, 1]])

L1 = np.array([[1, 2, 3, 4],
              [2, 3, 4, 1],
              [3, 1, 0, 2],
              [4, 0, 2, 3]])

#order of L
n = len(L[0])

#count of every element in the square
```

```
Num = Counter(chain(*L))

#if a number between 1 and n appears 0 times,
# we add it to the counter with a value of 0
for x in list(range(1, n+1)):
    if x not in Num.keys():
        Num.update({x:0})

#counter of every element not including 0 (empty cells)
Num2 = copy.deepcopy(Num)
del Num2[0]

#order counter by most occuring, extract the smallest value
m0 = Num2.most_common()[-1][1]

#number of empty cells in L
d = Num[0]

#The Algorithm
def PleaseWork(m):
    t1 = 2*n - m
    m1 = floor(d/(t1-n))

    if m1 >= m:
        print('We can embed L in a square of order', t1)
        return t1
    else:
        PleaseWork(m1)

PleaseWork(m0)
```

Bibliography

Andersen, Lars, Stacie Baumann, Anthony Hilton, and Chris Rodger. 2022. On Completing Partial Latin Squares with Prescribed Diagonals. *Electron J Combin* 29(3):Paper No. 3.47–. doi:10.37236/10705. URL https://doi.org/10.37236/10705.

Andersen, Lars Døvling, and Anthony JW Hilton. 1983. Thank Evans! *Proceedings of the London Mathematical Society* 3(3):507–522.

Donovan, Diane M. 2000. The completion of partial Latin squares. *Australas J Comb* 22:247–264.

Evans, Trevor. 1960. Embedding incomplete Latin squares. *Amer Math Monthly* 67:958–961. doi:10.2307/2309221. URL https://doi.org/10.2307/2309221.

Hall, Marshall. 1945. An existence theorem for Latin squares. *Bulletin of the American Mathematical Society* 51:387–388.

Hall, P. 1935. On representatives of subsets. *Journal of the London Mathematical Society* s1-10(1):26–30.

Meng, Jixian, and Xinzhong Lu. 2011. The design of the algorithm of creating sudoku puzzle. In *ICSI (2)*, 427–433.

Ryser, H. J. 1951. A combinatorial theorem with an application to Latin rectangles. *Proc Amer Math Soc* 2:550–552. doi:10.2307/2032005. URL https://doi.org/10.2307/2032005.

Shahriari, Shahriar. 2022. *An invitation to combinatorics*. Cambridge Mathematical Textbooks, Cambridge University Press, Cambridge.

Smetaniuk, Bohdan. 1981. EnglishA new construction on Latin squares. I: A proof of the Evans conjecture. *Ars Comb* 11:155–172.