

Math Madness: An analytical approach to the seeding of NCAA Men's Basketball Tournaments

Brice Anderson, Ryan McElvain, and Adam Warren
Section 003

University of Colorado, Boulder

Abstract. The authors explore three distinct methods for seeding men's college basketball teams. Within these methods, scheduling and scoring are heavily taken into account, compared to the traditional win/loss-based methodology used in typical team seeding. They compare their results with existing seedings and draw conclusions based on the findings.

Keywords: Perron–Frobenius theorem · Least Squares.

1 Attribution

This project was a collective effort of all three authors. Sections 3.1 and 5 were the main work of Brice Anderson. Sections 2 and 5 were the main work of Ryan McElvain. Sections 3.2-3, 4.1, and 6 were the main work of Adam Warren.

2 Introduction

In sports, it is very common that not all teams face each other during the regular season. This can create a situation where seeding teams based solely on their win-loss record becomes one-dimensional. When teams do not all play against each other, their records can be misleading. For example, a team with a strong record might have faced weaker opponents, while another team could have a weak record simply because they've played against stronger teams.

In our project, we plan to use the Perron-Frobenius Theorem to seed men's college basketball teams by using three methods that all take scheduling into account: The first is a more straightforward method that takes both head-to-head matchups and strength of schedule into account. The second method is very similar to the first, the only difference is that it takes the scores of the head-to-head matchups into account, rather than just using a win-loss based strategy. The third method uses a least-squares approach that determines the seeding by approximating the probability of winning for each team based on previous scores.

For our procedure, we will select all 16 teams from the SEC conference in the 2024-2025 season, and use the data from their head-to-head matchups from their 18 regular-season games. Using our three methods, we will form a preference

matrix, which is square and contains information regarding the regular season games. From this matrix, we will find a positive, real eigenvector that represents the strength of the teams. The product of the results matrix and the strength vector will yield a score vector \mathbf{s} seeding all teams. The highest value in vector \mathbf{s} will correspond to the 1 seed, and the lowest value will correspond to the 16 seed. We will then compare our new sets of seeds to the original conference seeds and analyze the similarities and differences between the original seeds and our new sets.

3 Mathematical Formulation

Our goal is to seed the 16 basketball teams in the SEC conference in the 2024-25 regular season. Each team played 18 games against other teams within the conference. Teams will be organized alphabetically, so Alabama will be team 1 and Vanderbilt will be team 16.

3.1 Simple Preference Matrix

One way we can achieve this is by using a linear method. We will create a "preference" matrix A containing all our data regarding head-to-head matchups. The element a_{ij} represents the number of wins that team i had against team j . The element a_{ii} is meaningless in reality, as team i cannot play itself. Several options were considered for this element, such as having it equal 0 for all i , 0.5 for all i , 0 for all 1, having it equal the average of the entries in the i th column and the i th row [2]. In the end, we decided to have the diagonal entries equal 0, so as not to influence our seedings. After we create matrix A , we will diagonalize the matrix, so that it is in the form of

$$A = SAS^{-1} \quad (1)$$

where S is a matrix whose columns are the eigenvectors of A , and Λ is a diagonal matrix, where its entries are the eigenvalues of A . The seed vector, \mathbf{s} will be determined using the formula below:

$$\mathbf{s} = A\mathbf{r} \quad (2)$$

where A is the results matrix and \mathbf{r} is the positive real eigenvector of A . The entry s_i is the ranking of team i , where the highest ranking represents the best team. The basis of this equation comes from the Perron-Frobenius Theorem, which states:

If the (nontrivial) matrix A has nonnegative entries, then there exists an eigenvector \mathbf{r} with nonnegative entries, corresponding to a positive eigenvalue λ . Furthermore, if the matrix A is irreducible, the eigenvector \mathbf{r} has strictly positive entries, is unique and simple, and the corresponding eigenvalue is the largest

eigenvalue of A in absolute value (i.e., is equal to the spectral radius of A). [1]

Since all teams played the same number of games in the dataset, this equation was simplified from the equation below.

$$s_i = \frac{1}{n_i} \sum_{j=1}^N a_{ij} r_j \quad (3)$$

where s_i is the score of team i , n_i is the number of games played by team i , N is the total number of teams a_{ij} is the number of victories team i has against team j , and r_j is the real, positive eigenvector rank of team j , where $r_j > 0$. [1]

Since we only need the largest real positive eigenvector, we can use the formula:

$$\lim_{n \rightarrow \infty} \frac{A^n \mathbf{r}_0}{\|A^n \mathbf{r}_0\|} = \mathbf{r} \quad (4)$$

This method, called the power method, uses the repeated multiplication of some random initial vector \mathbf{r}_0 by the matrix A . We can represent this vector as a linear combination of eigenvectors that make up A , $\tilde{\mathbf{r}}$. At first this seems to not help much, but because $A \sim \Lambda$ (by equation 1), the multiplication of this new $\tilde{\mathbf{r}}$ by A becomes multiplication by the diagonal matrix and the eigenvector with the largest eigenvalue becomes the dominant entry of $\tilde{\mathbf{r}}$. We divide by the norm to make sure that the series converges. The new $\tilde{\mathbf{r}}$ becomes all zeros except for the component corresponding to the largest eigenvector. After changing back into the standard basis, \mathbf{r} becomes the largest eigenvector of the matrix. See §6.2 for a justification.

3.2 Score Based Preference Matrix

A more robust approach takes the score into consideration for each game. By only assigning a value of 1 or 0 given the outcome of the game, large score differentials are not taken into account. We can adjust our preference matrix A , taking into account close games and big wins. Given a game with score S_{ij} of team i winning against S_{ji} points of team j , we can assign the winning team "points" for each game given by

$$a_{ij} = \frac{1}{N} \sum_{k=1}^N \left(\frac{S_{ij}}{S_{ij} + S_{ji}} \right)_k \quad (5)$$

where N is the number of games between teams i and j . We divide by the total points scored in the game to normalize the data into a score quotient for each game k . Due to the high-scoring nature of basketball, this leads many games to award these points very close to 0.5. We can contradict this by using a non-linear function to map these scores. Using a function like a shifted sigmoid function (6) makes the preference matrix reflect the wins better. We used the equations

$$h(x) = \frac{1}{1 + e^{-\alpha(x-0.5)}}, \quad (6)$$

$$a_{ij} = \frac{1}{N} \sum_{k=1}^N h \left[\left(\frac{S_{ij}}{S_{ij} + S_{ji}} \right)_k \right] \quad (7)$$

where α is chosen based to fit most of the score quotients between 0 and 1. For basketball, many games have these quotients on the order of 0.5 to 0.6, so a fairly aggressive $\alpha = 50$ was chosen such that $h(0.6) \approx 0.99$.

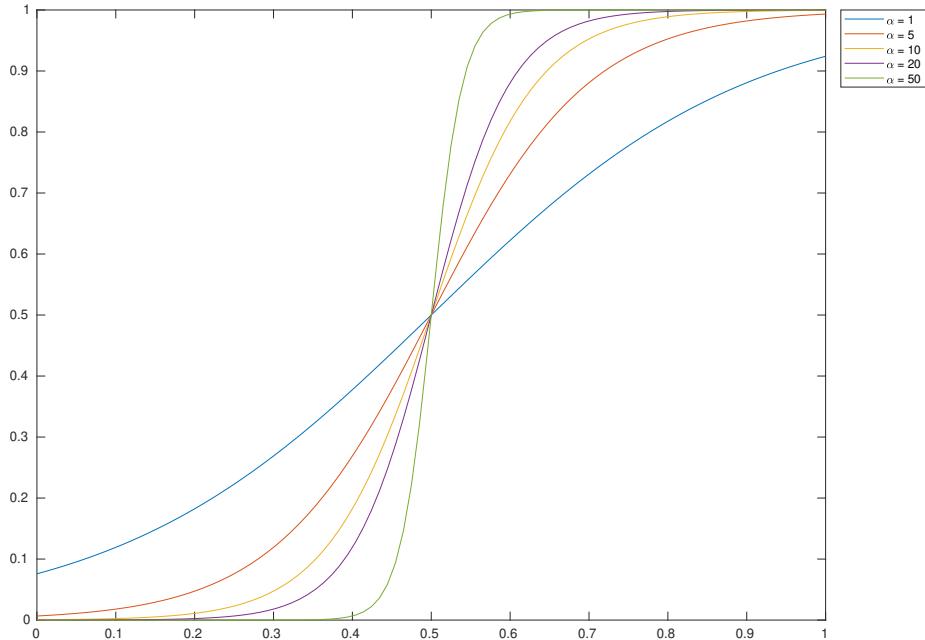


Fig. 1. Various $h(x)$ with different α parameters. Notice as α increases the reward increases.

After computing the new preference matrix, the Perron-Frobenius theorem still proves the existence of a positive eigenvalue with a corresponding eigenvector with nonnegative entries. This can be found again using the method described above (4).

3.3 Least Squares

Another approach to ranking is to assess the probabilities between two teams. The probability that team i beats team j denoted p_{ij} , can be found by taking the unknown rank of each team into account in the equation:

$$P(\text{team } i \text{ wins over team } j) = p_{ij} = \frac{r_i}{r_i + r_j} \quad (8)$$

As the game between opponents is distributed as a Bernoulli random variable, we know $p_{ij} + p_{ji} = 1$. We can also say:

$$\frac{r_j}{r_j + r_i} r_i = \frac{r_i}{r_j + r_i} r_j \quad (9)$$

$$p_{ji} r_i = p_{ij} r_j$$

in order to find the rankings given the probability of winning. However, since we do not know p_{ij} , we cannot do this just yet. We can estimate p_{ij} as a random variable. We consider p_{ij} as a normally distributed quantity (motivated by [3]). Estimating the probability of a win between the teams can be done using the scores between the two teams, given by:

$$p_{ij} = \frac{S'_{ij}}{S'_{ij} + S'_{ji}} \quad (10)$$

where S'_{ij} is the sum of all points team i scored on team j . Equation (8) becomes:

$$S'_{ji} r_i - S'_{ij} r_j = 0 \quad (11)$$

This gives an equation in the form $S\mathbf{r} = \mathbf{0}$ which has a trivial solution of $\mathbf{r} = \mathbf{0}$. Instead, we minimize the squared residual:

$$\underset{\|\mathbf{r}\|=1}{\operatorname{argmin}} \sum_{i,j} (S_{ij} r_i - S_{ji} r_j)^2 \quad (12)$$

It is shown in the appendix that this sum can be rewritten as the squared norm induced by the inner product:

$$\langle \mathbf{r}, \mathbf{r} \rangle_B = \mathbf{r}^T B \mathbf{r} \quad (13)$$

with B formed by the entries:

$$\begin{aligned} b_{ii} &= \sum_k S'_{ik}^2 & i = j \\ b_{ij} &= -S'_{ij} S'_{ji} & i \neq j \end{aligned} \quad (14)$$

Solving this least squares problem simplifies down to the following equation, which is different than the way we solved it in class. This method uses Lagrangian multipliers whose eigenvalues can once again be proven using the Perron-Frobenius theorem.

$$B\mathbf{r} = \mu \mathbf{r} \quad (15)$$

To ensure that our eigenvalues are positive, we can shift the matrix to be diagonally dominant. $B' = B + \lambda_0 I$. This new matrix is inverted to ensure the eigenvectors are positive and the limit [1]

$$\lim_{n \rightarrow \infty} \frac{(B + \lambda_0 I)^{-n} \mathbf{r}_0}{\|(B + \lambda_0 I)^{-n} \mathbf{r}_0\|} = \mathbf{r}$$

As Keener [1] suggests, taking the inverse of B' is not advisable computationally; however, because our matrix is much smaller (16×16), our condition number doesn't dramatically change, and we can compute B'^{-1} directly. The results in relation to our SEC data are shown in §4.

4 Examples/Numerical Results

4.1 Computation

To obtain our results for our dataset, we used MATLAB to compute our rankings. As a simplified explanation of the code (located in Appendix §6.3), we break down our program into a few steps, with the code used for generating the tables omitted for clarity.

We first read the data from an Excel spreadsheet with all games played in the regular season of the SEC conference 2025 (data were obtained from [5]). The data is then used in the creation of the preference matrices used for methods 1 and 2. This simple algorithm starts with the 0 matrix and computes the sums shown in equations (3) and (7) using a for loop. These methods also require us to find the eigenvalue of the preference matrices generated, so instead of writing our own method, we used the built-in function *eigs* as it is more numerically stable. This is done on the preference matrix for methods 1 and 2.

For method 3, a more robust algorithm was written to obtain the combination of features needed in the B' matrix. We first generated an intermediary matrix that holds the total scores $a_{ij} = S'_{ij}$ and the total points scored in the season on the diagonals. Then, negating the Hadamard product of this matrix with itself, the matrix B described by (14) is obtained after negating the diagonal entries. We then use an algorithm to find all of the Gershgorin disk radii and find the correct factor, λ_0 , to offset the matrix by. After this B' matrix is obtained, we compute the inverse directly and find the eigenvector of that inverse. As noted above, this direct computation of the inverse can affect the numerical integrity of the results. However, because the matrix is of a small dimension, its effect on the condition number was negligible.

4.2 Scores

The tables below show how teams rank purely on conference win-loss record, we rated and seeded teams using the three methods stated above, and the disparity between the rankings.

Table 1. Simple Preference Seeding compared with Win-Loss Record Seeding

Team	SEC Rank	Method 1	Ranking Difference
Auburn	1	1	0
Florida	2	2	0
Alabama	3	3	0
Tennessee	4	4	0
Texas A&M	5	7	-2
Kentucky	6	5	1
Missouri	7	6	1
Mississippi	8	8	0
Arkansas	9	10	-1
Mississippi St.	10	12	-2
Georgia	11	11	0
Vanderbilt	12	9	3
Texas	13	13	0
Oklahoma	14	14	0
LSU	15	15	0
South Carolina	16	16	0

Table 2. Score Based Preference Seeding compared with Win-Loss Record Seeding

Team	SEC Rank	Method 2	Ranking Difference
Auburn	1	1	0
Florida	2	2	0
Alabama	3	3	0
Tennessee	4	4	0
Texas A&M	5	7	-2
Kentucky	6	5	1
Missouri	7	6	1
Mississippi	8	8	0
Arkansas	9	10	-1
Mississippi St.	10	12	-2
Georgia	11	9	2
Vanderbilt	12	11	1
Texas	13	13	0
Oklahoma	14	14	0
LSU	15	16	-1
South Carolina	16	15	1

Table 3. Least Squares Seeding compared with Win-Loss Record Seeding

Team	SEC Rank	Method 3	Ranking Difference
Auburn	1	3	-2
Florida	2	10	-8
Alabama	3	1	2
Tennessee	4	12	-8
Texas A&M	5	11	-6
Kentucky	6	2	4
Missouri	7	4	3
Mississippi	8	7	1
Arkansas	9	9	0
Mississippi St.	10	13	-3
Georgia	11	14	-3
Vanderbilt	12	5	7
Texas	13	8	5
Oklahoma	14	6	8
LSU	15	15	0
South Carolina	16	16	0

Table 4. All results combined together

Team	SEC Rank	Method 1	Method 2	Method 3
Auburn	1	1	1	3
Florida	2	2	2	10
Alabama	3	3	3	1
Tennessee	4	4	4	12
Texas A&M	5	7	7	11
Kentucky	6	5	5	2
Missouri	7	6	6	4
Mississippi	8	8	8	7
Arkansas	9	10	10	9
Mississippi St.	10	12	12	13
Georgia	11	11	9	14
Vanderbilt	12	9	11	5
Texas	13	13	13	8
Oklahoma	14	14	14	6
LSU	15	15	16	15
South Carolina	16	16	15	16

5 Discussion and Conclusion

5.1 Discussion of Results

The tables provide valuable insight into the strength of teams that their records do not show. This can be well illustrated using the seeding difference metric. A positive difference means that the team was ranked higher in our seeding compared to the conference standings, a negative difference means that the team was ranked lower in our seeding compared to the conference standings, and a zero difference means that the team was ranked equally in our seeding and in the standings. The seeding difference provides valuable insight into these teams' strength of schedule. Specifically, a positive difference between method 1 and the standings indicates an above-average level of competition a team played, a negative difference indicates a team played an easier schedule than average, and a zero difference indicates a team played an average schedule. Additionally, a positive difference between methods 1 and 2 (seed in method 2 is better/lower than in method 1) indicates that a team won more blowouts and lost more close games, a zero difference indicates that a team won and lost an average number of blowouts and close games, and a negative difference indicates that a team lost more blowouts than average and won more close games than average.

The combined results of our three methods demonstrate a strong similarity between method 1, method 2, and the standings, but a stark contrast with method 3. Method 1, method 2, and the standings are relatively uniform, with small differences found primarily in the middle of the standings. This is not especially surprising since the middle of the standings featured many tied records. Furthermore, since method 1 and method 2 utilized similar strategies, it stands to reason that their results would be similar.

Method 1 and the standings' similarity indicates that most teams played similar strengths of schedules. Vanderbilt is an exception, which seems to have played an exceptionally difficult schedule. In contrast, Texas A&M and Mississippi State seem to have played an exceptionally easy schedule.

Method 1 and method 2's similarity indicates that most teams won and lost a similar number of blowout games and close games. The positive difference in Georgia's seed between the two methods indicates that Georgia lost more close games than average and won more blowouts than average. Inversely, the negative difference in Vanderbilt's seed between the two methods indicates that Vanderbilt won more close games than average and lost more blowouts than average. Method 3, however, demonstrates a significant deviation in seedings between methods 1 and 2 and the SEC conference standings. The explanation for this discrepancy can be attributed to the complex optimization techniques utilized within the method. This method was created to seed American football teams, and although methods 1 and 2 translated nicely to basketball, method 3 appears to have a less fluid translation. It is clear that method 3 inaccurately seeds the teams through the two main examples in Table 4. Most notably, seed 2 in the SEC conference was re-seeded to seed 10, and seed 4 in the SEC conference was re-seeded to seed 12. In the SEC conference, and using methods 1 and 2,

seeds 1 through 4 all remain the same, so it is highly unlikely that they can be re-seeded so low in the standings without compatibility issues between the Least Squares method and men's college basketball statistics. Therefore, the seeds generated using method 3 should be ignored due to the major discrepancies between the seedings provided by the SEC and the other two methods, and between those of method 3.

5.2 Considerations

The dataset for this paper was a large point of contention. The games these teams played in the 2024-25 season can be split into 4 categories: out-of-conference regular-season games, conference regular-season games, conference championship games, and March Madness tournament games. Conference regular-season games were the basis of our motivation and thus were included in the dataset. Out-of-conference regular-season games were excluded from the dataset since the teams' opponents would have to be included in the score matrix, which was outside the scope of our paper.

Secondly, March Madness tournament games were excluded for a very similar reason. These games featured a mix of conference and out-of-conference opponents. We chose not to include the March Madness tournament games featuring conference games, since these games were played at a neutral court, unlike the conference games, and with a frantic schedule. We determined that these factors would affect the quality of play, and including these games would convolute the seeds.

Third, conference championship games were excluded. This was the point of greatest contention, since the games are all played between conference teams. We decided to exclude these games since they were played at a neutral location, unlike the conference regular season games, a factor that we determined might convolute the seeds. In conclusion, we only included conference regular season games to ensure homogeneous conditions and simplicity. It would be interesting to investigate the effects of including conference tournament games and in-conference March Madness games on our seedings.

5.3 Reflection and Conclusion

Teams' seedings depend on the factors you consider and how much you weigh them. The win-loss record will give you a general idea of the strength of a team, but it can miss context, if, for example, a team plays a very weak schedule or only beats weak teams. Accounting for the strength of schedule can help provide more context, but it is limited. It does not value close games and blowouts any differently, unlike most sports fans. However, accounting for the final score has its limitations. A team might have a massive blowout early in the game, but let off the gas, leading to the score not accurately reflecting the gulf in class. One could attempt to account for this, but ultimately, at a certain level, one might be attempting to add too much context, overcomplicate the seeding process, convolute the seeds, and lose sight of the bigger picture. Most of the seedings

presented in this paper have value and are useful in determining the strength of these teams. However, one ranking is not the end-all, be-all. We believe it is wiser to view these different rankings that should be utilized and weighed together to give sports fans a nuanced view of these teams.

References

1. The Perron-Frobenius theorem and the ranking of football teams, by JP Keener, SIAM Review (1993).
2. On the Singular Value Decomposition and Ranking, by Zizler, Peter, et al, Computational Methods in Science and Technology, (2020).
3. Remarks on the method of paired comparisons: I. The least squares solution assuming equal standard deviations and equal correlations, by Frederick Mosteller, Psychometrika, (1951).
4. ELENI ALOUPOGIANNI (2025). TABLE2LATEX (<https://github.com/foxelas/Matlab-assisting-functions/releases/tag/3.0>), GitHub. Retrieved May 1, 2025.
5. “2025-26 Projections -- Customizable College Basketball Tempo Free Stats - T-Rank.” T, barttorvik.com/trankpre.php. Accessed 30 Apr. 2025.

6 Appendix

6.1 Steps between equations 12 and 13

As an overview, we expand the square and sum, then reorder the sum on the r_j^2 term. Note we are able to turn the cross term sum index from i, j to $i \neq j$ as for $i = j$ we have $S_{ij} = 0$ and the term is equal to 0 in the sum.

$$\begin{aligned}
\sum_{i,j} (S_{jir_i} - S_{ijr_j})^2 &= \sum_{i,j} S_{ji}^2 r_i^2 - 2S_{ji} S_{ij} r_j r_i + S_{ij}^2 r_j^2 \\
&= \sum_i \left(\sum_j S_{ji}^2 \right) r_i^2 - 2 \sum_{i,j} S_{ji} S_{ij} r_j r_i + \sum_j \left(\sum_i S_{ij}^2 \right) r_j^2 \\
&= \sum_i \left(\sum_j S_{ji}^2 \right) r_i^2 - 2 \sum_{i,j} S_{ji} S_{ij} r_j r_i + \sum_i \left(\sum_j S_{ji}^2 \right) r_i^2 \\
&= 2 \left(\sum_i \left(\sum_j S_{ji}^2 \right) r_i^2 + \sum_{i \neq j} -S_{ji} S_{ij} r_j r_i \right)
\end{aligned}$$

Rewriting the diagonal terms r_i^2 get multiplied by

$$b_{ii} = 2 \sum_k S_{ik}^2$$

and the off-diagonal terms as

$$b_{ij} = -2S_{ji}S_{ij} \quad i \neq j$$

We then divide the matrix by 2 which does not change the eigenvalues. This leaves us with the square of the inner product defined by

$$\begin{aligned} \langle \mathbf{r}, \mathbf{r} \rangle_B &= \mathbf{r}^T B \mathbf{r} \\ &= \|\mathbf{r}\|_B \end{aligned}$$

6.2 Justification of the Power Method

We start with a matrix $A \in \mathbb{R}^{n \times n}$ with a diagonal form $A = SDS^{-1}$, where D is diagonal with entries $\lambda_1, \lambda_2, \dots, \lambda_n$ and S has columns made of the eigenvectors $\mathbf{v}_1, \mathbf{v}_2, \dots, \mathbf{v}_n$. Given a random starting vector \mathbf{r}_0 we have the following. We set $\tilde{\mathbf{r}}_0$ equal to $S^{-1}\mathbf{r}_0$, and we have the largest eigenvalue, λ_1 .

$$\begin{aligned} \lim_{n \rightarrow \infty} \frac{A^n \mathbf{r}_0}{\|A^n \mathbf{r}_0\|} &= \lim_{n \rightarrow \infty} \frac{1}{\|(SDS^{-1})^n \mathbf{r}_0\|} (SDS^{-1})(SDS^{-1})(SDS^{-1})^{n-2} \mathbf{r}_0 \\ &= \lim_{n \rightarrow \infty} \frac{1}{\|(SDS^{-1})^n \mathbf{r}_0\|} (SD) S^{-1} S (D) S^{-1} S (DS^{-1})^{n-2} \mathbf{r}_0 \\ &= \lim_{n \rightarrow \infty} \frac{1}{\|SD^n S^{-1} \mathbf{r}_0\|} SD^n S^{-1} \mathbf{r}_0 \\ &= \lim_{n \rightarrow \infty} \frac{\lambda_1^n}{\|SD^n \tilde{\mathbf{r}}_0\|} S \frac{D^n}{\lambda_1^n} \tilde{\mathbf{r}}_0 \\ &= \lim_{n \rightarrow \infty} \frac{\lambda_1^n}{\|SD^n \tilde{\mathbf{r}}_0\|} S \frac{D^n}{\lambda_1^n} \tilde{\mathbf{r}}_0 \end{aligned}$$

Since all eigenvalues are smaller than λ_1 the diagonal matrix converges to

$$\frac{D^n}{\lambda_1^n} = \begin{bmatrix} 1 & \dots & 0 \\ \vdots & \ddots & \\ 0 & & 0 \end{bmatrix}$$

Although some steps are removed for clarity the limit converges to the matrix of eigenvectors times the elementary vector

$$\begin{aligned} &= S \mathbf{e}_1 \\ &= \mathbf{v}_1 \end{aligned}$$

For non-diagonalizable matrices, the Jordan-Canonical form is used where it can be shown the smaller blocks corresponding to $\lambda_2, \dots, \lambda_n$ also converge to the 0 matrix.

6.3 MATLAB Code

The project was coded in MATLAB with all results and tables generated by the script below.

```

1 % Read in the Data from Spreadsheet
2 % This code was auto-generated by matlab
3
4 % Set up the Import Options and import the data
5 opts = spreadsheetImportOptions("NumVariables", 24);
6
7 % Specify sheet and range
8 opts.Sheet = "SEC games";
9 opts.DataRange = "A2:X145";
10
11 % Specify column names and types
12 opts.VariableNames = ["x_", "Rk", "Date", "Type", " "
13     "Team1", "Team2", "Result", "Team1Score", " "
14     "Team2Score", "Var10", "Adj_D", "T", "EFF", "eFG_",
15     "TO_", "Reb_", "FTR", "EFF1", "eFG_1", "TO_1", " "
16     "Reb_1", "FTR1", "G_Sc", "x___"];
17 opts.VariableTypes = ["double", "double", "datetime", " "
18     "categorical", "categorical", "categorical", " "
19     "categorical", "double", "double", "string", "double"
20     ", "double", "double", "double", "double", "double"
21     ", "double", "double", "double", "double", "double"
22     ", "double", "double", "double"];
23
24 % Specify variable properties
25 opts = setvaropts(opts, "Var10", "WhitespaceRule", " "
26     "preserve");
27 opts = setvaropts(opts, ["Type", "Team1", "Team2", " "
28     "Result", "Var10"], "EmptyFieldRule", "auto");
29
30 % Import the data
31 APPM3310Data = readtable("/MATLAB Drive/Project/Data.
32     xlsx", opts, "UseExcel", false);
33
34 % Convert to output type
35 Team1 = APPM3310Data.Team1;
36 Team2 = APPM3310Data.Team2;
37 %Result = APPM3310Data.Result;
38 Team1Score = APPM3310Data.Team1Score;
39 Team2Score = APPM3310Data.Team2Score;
40
41 % Clear temporary variables
42 clear APPM3310Data

```

```

31
32 % Clear temporary variables
33 clear opts
34
35 % Display results
36 %Team1, Team2, Result, Team1Score, Team2Score;
37 % Santize the Data
38
39 team12 = [Team1 Team2];
40 team12 = reshape(team12, [],1);
41 [g, teams] = grp2idx(team12);
42 team12 = reshape(g, [], 2);
43
44 teams1 = team12(:,1);
45 teams2 = team12(:,2);
46
47 clear g team12
48 clear Team1 Team2
49 % Preliminary Report Table
50
51 % the actual seeding of the teams in alphabetical order
52 tournament_seeds = [3 9 1 2 11 6 15 8 10 7 14 16 4 13 5
53 12];
54
55 % this is used to generate the tables given known data
56 % from the 25 season
57 report_base = table(teams, tournament_seeds');
58 report_base.Properties.VariableNames = ["teams", "SEC
      Rank"];
59 report_base = sortrows(report_base, 'SEC Rank', 'ascend
      ')
60 % Create the Preference Matrices
61
62 % get the number of teams and number of games
63 [nTeams, ~] = size(teams);
64 [nGames, ~] = size(teams1);
65
66 % The "direct-method" matrix
67 A1 = zeros(nTeams);
68 % The "better" method
69 A2 = zeros(nTeams);
70
71 % counts games for a team i
72 ni = zeros(nTeams,1);

```

```

72 % the "activation function used for method 2"
73 h = @(x) 1/(1+exp(-50*(x-0.5)));
74
75 % create the preference matrix given the game data
76 for i=1:nGames
77     t1 = teams1(i);
78     t2 = teams2(i);
79     t1s = Team1Score(i);
80     t2s = Team2Score(i);
81
82     % if team 1 wins record into the preference
83     % matrices accordingly
84     if t1s > t2s
85         A1(t1, t2) = A1(t1, t2) + 1;
86
87         x = (t1s)/(t1s + t2s);
88         A2(t1, t2) = A2(t1, t2) + h(x);
89     % if team 2 wins ...
90     else
91         A1(t2, t1) = A1(t1, t2) + 1;
92
93         x = (t2s)/(t2s + t1s);
94         A2(t2, t1) = A2(t2, t1) + h(x);
95     end
96
97     ni(t1) = ni(t1) + 1;
98     ni(t2) = ni(t2) + 1;
99 end
100 clear i t1 t2 t1s t2s x
101
102 A1;
103 A2;
104
105 nid = diag(1./ni);
106
107 % normalize team data by games played
108 A1 = A1*nid;
109 A2 = A2*nid;
110 %% Direct Method (Fully Linear)
111
112 % display first preference matrix for debugging
113 % purposes
114 A1

```

```

115 % Get the largest eigenvalue and eigenvector
116 [V,D] = eigs(A1,1); % produces negative eigenvalue (
117 % might consider using the method described in the
118 % paper)
119 s1 = V*D; % Compute \lambda*r to get the "score" of the
120 % team
121
122 % used to assign table data
123 order = 1:16;
124
125 report1 = join(report_base, table(teams, s1));
126 report1 = sortrows(report1, "s1", "descend");
127 report1 = addvars(report1, order');
128 report1.s1 = [];
129
130 diff = report1.Var4 - report1.("SEC Rank");
131 report1 = addvars(report1, diff);
132 report1.Properties.VariableNames = ["Team", "SEC Rank",
133 % Method 1", "Ranking Difference"];
134 report1 = sortrows(report1, "SEC Rank", "ascend");
135
136 report1
137
138 % generate first methods table report in table1.tex
139 % file
140 table2lateX(report1,'table1.tex')
141 % Accounting for Big Score Differences
142
143 A2
144
145 % find the eigenvector of the second preference matrix
146 [V,D] = eigs(A2, 1);
147 s2 = V*D;
148
149 % create the table for method 2
150 report2 = join(report_base, table(teams, s2));
151 report2 = sortrows(report2, "s2", "descend");
152 report2 = addvars(report2, order');
153 report2.s2 = [];
154
155 diff = report2.Var4 - report2.("SEC Rank");
156 report2 = addvars(report2, diff);
157 report2.Properties.VariableNames = ["Team", "SEC Rank",
158 % Method 2", "Ranking Difference"];
159 report2 = sortrows(report2, "SEC Rank", "ascend");

```

```

154
155 report2
156
157 table2latex(report2,'table2.tex')
158 %% Using the Least Squares Approximation
159 % $$b_{ii} = \sum_k{S_{ik}^2} \quad \backslash\!b_{ij} = -S_{ij}S_{-
160     ji}, \quad i \neq j$$
161 % must compute this matrix B
162 B = zeros(nTeams);
163
164 % put bij as sum of total match scores Sij
165 for i=1:nGames
166     t1 = teams1(i);
167     t2 = teams2(i);
168     t1s = Team1Score(i);
169     t2s = Team2Score(i);
170
171     % will square later in hadamard product
172     B(t1, t1) = B(t1, t1) + t1s;
173     B(t2, t2) = B(t2, t2) + t2s;
174
175     if t1s > t2s
176         B(t1, t2) = B(t1, t2) + t1s;
177         B(t2, t1) = B(t2, t1) + t2s;
178     else
179         B(t2, t1) = B(t2, t1) + t2s;
180         B(t1, t2) = B(t1, t2) + t1s;
181     end
182 end
183
184 clear i t1 t2 t1s t2s
185
186 % hadamard product
187 B = -B.*B';
188 % Ensuring the matrix is diagonally dominant
189
190 zRows = sum(B, 2);
191 zCols = reshape(sum(B, 1), [], 1);
192
193 % gershgorin radii
194 zr = -min(zRows, zCols);
195
196 clear zRows zCols
197

```

```

198 | for i=1:nTeams
199 | % remove diagonal entry from radius
200 | zr(i) = zr(i) - B(i, i);
201 | B(i, i) = -B(i, i);
202 | end
203 |
204 | clear i
205 |
206 | % debugging purposes
207 | B
208 |
209 | % force diagonal dominance
210 | b = diag(B);
211 |
212 | % use the minimum lambda to add to
213 | % decrease the effect on the condition number
214 | lambda0 = max(abs(b - zr));
215 |
216 | Bprime = B + lambda0*eye(nTeams); % create Bprime as
217 | % described in the paper
217 | Bprime = inv(Bprime); % could have issues with
218 | % condition number
218 |
219 | clear zr lambda0
220 | %%
221 | [V,D] = eigs(Bprime, 1); % produces negative eigenvalue
222 | % (might have to implement computation described in
223 | % paper)
222 | s3 = D*V;
223 |
224 | % create the report for the 3rd method
225 | report3 = join(report_base, table(teams, s3));
226 | report3 = sortrows(report3, "s3", "descend");
227 | report3 = addvars(report3, order');
228 | report3.s3 = [];
229 |
230 | diff = report3.Var4 - report3.("SEC Rank");
231 | report3 = addvars(report3, diff);
232 | report3.Properties.VariableNames = ["Team", "SEC Rank",
233 | % "Method 3", "Ranking Difference"];
233 | report3 = sortrows(report3, "SEC Rank", "ascend");
234 |
235 | report3
236 |
237 | table2latex(report3, 'table3.tex')

```

```
238 %%  
239 % Create the combined table for the final table report  
240 final_report = join(report_base, table(teams, s1, s2,  
    s3));  
241 final_report = sortrows(final_report, "s1", "descend");  
242 final_report = addvars(final_report, order');  
243 final_report.s1 = []  
244 final_report = sortrows(final_report, "s2", "descend");  
245 final_report = addvars(final_report, order');  
246 final_report.s2 = []  
247 final_report = sortrows(final_report, "s3", "descend");  
248 final_report = addvars(final_report, order');  
249 final_report.s3 = []  
250  
251 final_report.Properties.VariableNames = ["Team", "SEC  
    Rank", "Method 1", "Method 2", "Method 3"];  
252 final_report = sortrows(final_report, "SEC Rank", "  
    ascend");  
253  
254 final_report  
255  
256 table2latex(final_report, 'table4.tex')
```