University of Liège
School of Engineering and Computer Science

Master's thesis carried out to obtain
a Master of Science in Computer Science Engineering

Ranking Algorithms: Robustness and Accuracy

Thomas LUCAS

Supervised by
Professor Quentin Louveaux

2021-2022

Acknowledgement

The completion of this master's thesis could not have been possible without the help of the people surrounding me.

I would like to thank my supervisor Professor Quentin Louveaux for the time he has putted in this research to make sure it was going in the right direction as well as the suggestions he shared with me in terms of algorithmic solution and writing.

I also would like to thanks the readers for the time they gave to the reading of the entire work.

Words can not express how grateful I am to my girlfriend for the motivation and support she gave me all along the course of my studies and especially during this research.

Finally, I would like to thanks my parents and family for the education, the support and the help they provided me with in order to undertake these studies.

Contents

Chapter 1

Introduction

1.1 Objectives

This research aims at analyzing different ranking systems according to sports data sets and discuss the benefits and issues coming from the use of these algorithms. Multiple ranking techniques have been developed in order to extract, from a list of pairwise results, an ordering of participants reflecting their skills in the tournament they are competing in. However, no system has been proven to be the best in all cases, extracting a ranking from the complex directed graph of the pairwise results is a difficult task and might be biased towards certain participants. In this regard, this research tries to analyse each type of ranking method with respect to data bias, errors and lack of data and show results on the robustness and accuracy of these ranking methods.

To estimate the results of such algorithms, they are tested on different data sets such as:

- Men's international football results,
- ATP Tour Tennis results,
- and Premier League results.

These data sets have been chosen because of the difference in number of matches per participants, with the Premier League having matches between each pair of teams, ATP Tour that has more matches played by the players who get further in tournaments and FIFA results having matches mainly between countries belonging to the same continent. Each ranking system is tested on each data set to ensure the robustness, speed and accuracy of those algorithms. The used metrics are developed during the following sections.

The different systems all have issues computing a ranking which is as good as possible. All of them have advantages and disadvantages. As of today, the variety in the ranking system is high with some that are relatively stable such as the sum of points and other that inflate over time such as the ELO system. The main differences between these systems are not actually in the underlying algorithms but rather in the coefficients used in order to produce a rating for each participant.

This work contributes in different ways towards the elaboration of the ranking algorithm that outputs the best possible ranking from a data set of matches. It presents a survey of different ranking systems, the proposal of new ranking algorithms and their testing in various situations according to multiple metrics. The thesis presents multiple ways of extracting a ranking from a list of results.

1.2 Background

The two most common ranking algorithms nowadays are the ELO system which is used in FIFA ranking, Chess, Go, Baseball, Table Tennis, Esports, etc. and also the sum of the points such as the one used in Premier League, Tennis, Basketball, Ice Hockey, etc. It is often said that the ranking of a championship, such as the Premier League, with matches for each teams or players against each other is very reliable. However, even this kind of system does not take into account the level of the teams they won or lost against which makes a ranking even more independent on the actual skill level of the participants. The point addition system in tennis is an even better example as tennis players are seeded in the tournaments. The best ranked players actually play early on against the worst ranked players resulting in easier point gain for the players that were ranked higher than the others. The bias towards the first players results in a ranking which is slow to adjust towards the actual skill level of the participants.

In other words, the systems used today to rank teams at a professional level might not be the most suitable ones and other methods are proven to be better at predicting the order of the teams or players in a championship or tournament. A common example that has been criticized for decades and changed many times is the FIFA ranking. Belgium has been, for years, the number one team even though it has never won a major tournament in the time frame. The only podium of the team during the last two decades in a major tournament is a third place at the 2018 World Cup in Russia. It has never reached a podium in UEFA European Championship since 1980.

1.3 Organisation

As the research shows a wide range of algorithms and many mathematical formulas, it is divided into three distinct parts in order to follow the course of the study and show the results at the end of the thesis.

- The first part presents the definition and the needs for any algorithm to be able to rank participants in a competition such as the data needed and its structure as well as the measures of robustness and accuracy. This section also shows the issues to produce the optimal ranking out of a data set.
- The second part is the mathematical and computational explanations of the different algorithms found in the literature with more details towards their own implementations.
- The third section presents the suggested ranking methods that showed potential to be used for ranking complex competitions with their explanation and implementation.
- The final part is the presentation of the data sets as well as the achieved results by all the algorithms exploiting these data bases followed by their analysis. The results are presented with regard to computing time, accuracy and robustness. Graphs and tables are provided in order to better visualize the measures computed during the implemented testing. The presented results are quite interesting with different algorithms being better than others according to the sparsity and randomness of the data. The implemented code uploaded to GitHub is provided in the appendices within the bibliography.[12]

Chapter 2

Definitions

2.1 Ranking

In the literature, several different notations are used. We propose to first fix the main definitions and notations used in this thesis. Designing a ranking algorithm requires data where the comparison between different teams or players can be made such as football results. This data either contains the result of the match such as a win, draw or loss, or it contains the actual score at the end of the match.

The ranking system can either output a rating r for each participant or an ordering of the different participants. The rating systems give each team a rating with the highest one being given to the best team. The other systems, that outputs an ordering with the best team as the first one in the ranked list do not present a rating for the participants. This research presents a variety of ranking systems with some that outputs a rating and others that outputs only an ordering. Of course, from the order of the team, a rating can be produced such as: for each participant the rating equals the total of participant subtracted by the ranking of the participant but in this case it does not really represent the actual skill of this participant.

2.2 Adjacency matrix

The data is a succession of matches that can result in a win, loss or a draw. As this data can be represented as a graph with the edges replacing the matches and the nodes replacing the participants, an adjacency matrix can be produced which is also called encounter matrix. Thanks to this extracted data, the adjacency matrix of the graph can be constructed through the following rule:

$$AMatrix_{ij} = k \tag{2.1}$$

when the team i won k times against the team j.

The results within the matches is not taken into account even though it could be interesting especially in the football case. The implementation would be probably more complex in the tennis case due to the fact that a player can win a match with less points or games than the opponent. However, the number of sets can be taken into account the same way as the football results. The draw matrix $DMatrix$ is built upon the same rule but with the draws represented this time leading to the matrix being symmetrical given the fact that the number of draws of a against b is the same than the number of draws of b against a.

Regarding the date of the matches in the data, one can easily state that a match played in 1950 should not influence the rankings of a team today. However, should a 2018, 2014 or 2010 result influence that ranking? The skill level of a team is less dependent on older results, meaning that an importance coefficient could be implemented in order to simulate a penalty on the older results to change the number in the $AMatrix$. This study does not implement or test such a coefficient. The time penalty is implemented as a time window where the matches older than 4 years in the FIFA ranking are not considered. This allows to always consider a world cup when computing a ranking. This time window is reduced to one year in the case of ATP Tour and Premier League as there are more matches played per participants per year with the most important ones in the time frame.

This thesis presents an importance coefficient added towards important matches such as the world cup matches, as the FIFA ranking system already implements[4]. The addition of such a bias towards more important matches should lead to better estimates of the actual level of each participant. The testing of these coefficient is presented in the testing chapter inspired from an article found in "The New-York Times".[6]

2.3 Measures from a ranking

Across the literature, several criteria are used to compare the rankings, each of them tries to compare their proposed algorithms to others already proven to be quite accurate on the ranking problem. Different methods are explained in the following section with a link to a paper that describes them in further details. A first focus on the way they will be evaluated is presented here.

There are two different approaches when evaluating a ranking algorithm. When the optimal ranking is available, in the case of synthetic data where the data is based on a defined ranking, the access to this optimal ranking allows us to compare it to the one computed by the algorithms. The measure of the performance is done through three different scoring methods:

- The Kendall distance defined as the number of pairs that are in a different order between the two rankings written as:

  $$\{(i,j) : i < j, [\tau_1(i) < \tau_1(j) \wedge \tau_2(i) > \tau_2(j)] \vee [\tau_1(i) > \tau_1(j) \wedge \tau_2(i) < \tau_2(j)]\}$$

  With $\tau_1(i)$ and $\tau_2(i)$ are the rankings of the elements i in τ_1 and τ_2 respectively. This measure only takes the ordering in the ranking to produce the score. In order to illustrate it, let us take the rankings $[A, B, C, D]$ and $[C, A, B, D]$. A table can be produced that states for each pair the order of the ranking in the case of the two different rankings and counts the number of times the order is different as:

Pair	order 1	order 2	Count
(A, B)	$A > B$	$A > B$	0
(A, C)	$A > C$	$A < C$	1
(A, D)	$A > D$	$A > D$	0
(B, C)	$B > C$	$B < C$	1
(B, D)	$B > D$	$B > D$	0
(C, D)	$C > D$	$C < D$	1

Table 2.1: Table of orderings in two different rankings in order to compute the Kendall distance

The result is that the Kendall distance between the two rankings is equal to 3 which is the sum of the last column. When the Kendall distance is high, the rankings are considered as far apart for one another.

- The second one is the Pearson correlation which is the covariance of the two outputted rankings divided by the two standard deviations multiplied by each other, mathematically written as:

  $$Correlation = \frac{cov(X,Y)}{\sigma(X)\sigma(Y)} \tag{2.2}$$

  with X and Y the two rankings. This measure takes the ratings of the participants into account. In order to illustrate it, let us take the rankings $[A, B, C, D]$ with the ratings being $[0.6, 0.2, 0.1, 0.1]$ for the first ranking and $[0.3, 0.3, 0.2, 0.2]$ for the other. The correlation is then the covariance divides by the product of the two standard deviations which leads to $\frac{0.03}{0.1*0.412} = 0.7276$. When the Pearson correlation is high in absolute value the two rankings are considered as close to each other.

- The third metric is the root mean square error defined as the square root of the sum or the square of the differences divided by the number of teams with T the number of teams:

  $$RMSE = \sqrt{\frac{\sum_{t=1}^{T}(X_t - Y_t)^2}{T}} \tag{2.3}$$

  where X and Y are the two rankings. This method is probably the most common one and takes into account the ratings of the participants. By taking the same ratings as for the previous example, the total value of the mean square root is:

  $$\sqrt{\frac{(0.6-0.3)^2 + (0.2-0.3)^2 + (0.1-0.2)^2 + (0.1-0.2)^2}{4}} = 0.1732$$

  When the root mean square error is high, the rankings are further apart.

2.4 Measures from matches

The "real" ranking from which the synthetic data is obtained is not always the actual optimal ranking for the generated data, as a random factor should be added in order to make the ranking less trivial to produce and there could be multiple solutions to the problem of ranking participants in a competition.

One of the main measure used is the number of upsets which measures the number of matches where the winner is ranked below the loser which means the number of matches wrongly predicted by the ranking. This measure represents better the actual accuracy of a ranking algorithm on match data compared to the previously presented ones. It is computed as:

$$Upsets = \sum_{i,j}^{N} AMatrix_{ij}, \text{ if } rank_j \geq rank_i \tag{2.4}$$

where $AMatrix_{ij}$ is the number of matches won by team i against team j and N is the number of participants. The issue with such a measure is that it is discrete and very strict as the loss against a team ranked one place behind or a hundred places behind is computed taken into account the same way.

Example 1 *In order to illustrate this measure, let us introduce a simple example given by the data: teams* $= [a, b, c, d]$

- *a beats b and d*
- *b beats c and d*
- *c beats a*
- *d beats c*

which gives:

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

If the ranking to be evaluated is $[d, c, b, a]$ then the sum is going to be:

$$AMatrix_{01} + AMatrix_{02} + AMatrix_{03} + AMatrix_{12} + AMatrix_{13} + AMatrix_{23} = 4$$

This simple example is also used in all the sections describing an algorithm to better visualize the presented method.

2.5 Robustness

The data is not perfect because a result of a match is not the perfect representation of the difference of skills between the two participants. Thus, some randomness is present in the data and some matches can be considered as outliers. Furthermore, as many competitions do not provide matches of each participants against all the other opponents, the data is not complete meaning that, when an algorithm ranks the participants, reducing the amount of matches available should not change the produced ranking too much. Thus, the algorithms need to be robust to ensure that the results they provide is a good representation of the skill of the participants.

Chapter 3

Literature algorithms

3.1 Colley matrix

The first method explored is one found in the literature and the search for ranking methods. Proposed by Colley in his paper [13], the algorithm shows a lot of advantages towards the creation of a ranking. The author claims that his algorithm:

- is easily reproducible,
- adjusts for strength of the whole schedule,
- and produces common sense results.

The time efficiency is not claimed by the authors but compared to some ranking algorithms the method is quite fast to produce its ranking. It is inspired from the Laplace's method which consists in computing the rating by:

$$r = \frac{1 + n_w}{2 + n_{tot}} \tag{3.1}$$

where n_w being the number of wins and n_{tot} the total number of matches. This method has the starting values of the rating at $1/2$ for each team participating and keeps this value as the average rating after the computation of the ratings. By taking the simple example of 2 teams meeting once and the first team winning the game, it can be noticed that the rating becomes:

$$r_1 = \frac{1+1}{2+1} = \frac{2}{3}$$

and

$$r_2 = \frac{1+0}{2+1} = \frac{1}{3}$$

Leading to the winning team being considered twice as good as the second team.

However, this method does not take into account the rating of the encountered teams. Thus, the author proposes an iterative method computing adjustments using the computed ratings and add this adjustment an infinite amount of time. The correction formula is:

$$n_{w,i}^{eff} = (n_{w,i} - n_{l,i})/2 + \sum_{j=1}^{n_{tot,i}} r_j^i \tag{3.2}$$

where $n_{w,i}^{eff}$ is the correction term, $n_{w,i}$ the number of wins of team i, $n_{l,i}$ the number of losses of team i and $n_{tot,i}$ the number of matches played by team i. From the previous simple example, we can compute iterations of the correction and show that the corrections vary in sign while shrinking in magnitude leading to the convergence of the algorithm (shown in the paper). The (3.1) and (3.2) equations can be rearranged in the form:

$$(2 + n_{tot,i})r_i - \sum_{j=1}^{n_{tot,i}} r_j^i = 1 + (n_{w,i} - n_{l,i})/2 \tag{3.3}$$

which is a system of N linear equations with N variables:

$$C\vec{r} = \vec{b} \tag{3.4}$$

where r is the column vector of the ratings, b the column vector of the right hand side of the equation and C the matrix defined as:

$$C_{ii} = 2 + n_{tot,i}, C_{ij} = -n_{j,i} \tag{3.5}$$

The ranking is obtained by solving this system and taking the vector of ratings. One main issue with the method presented in the paper is that it does not work with winning percentages. The Colley matrix must not be a singular matrix which would make the resolution of the system impossible.

Example 2 *In order to illustrate the algorithms, let us introduce a simple example given by the data presented in the section 2.4 where the adjacency matrix is:*

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

From this data, the Colley matrix ranking system creates the C matrix as follows:

$$\begin{bmatrix} 5 & -1 & -1 & -1 \\ -1 & 5 & -1 & -1 \\ -1 & -1 & 5 & -1 \\ -1 & -1 & -1 & 5 \end{bmatrix}$$

And the b vector $= \begin{bmatrix} 1.5 & 1.5 & 0.5 & 0.5 \end{bmatrix}$

By solving (3.4), *the algorithm outputs the vector of ratings:*

$$r = \begin{bmatrix} 0.58333333 & 0.58333333 & 0.41666667 & 0.41666667 \end{bmatrix}$$

The outputted ranking is: $[b, a, d, c]$ but the two first teams, a and b, have the same rating and the two last teams, c and d, as well.

3.2 Eigenvector

A ranking system based on eigenvalues is proposed by James P. Keener [5]. The author bases his ranking on the fact that "the assigned score should depend on both the outcome of the interaction and the strength of its opponents". From a vector r, with positive components r_j indicating the strength of participant j, the score of a participant is then computed as:

$$s_i = \frac{1}{n_i} \sum_{j=1}^{N} AMatrix_{ij} r_j \tag{3.6}$$

Where $AMatrix_{ij}$ is the outcome of the match between the participant i and the participant j, N is the total number of participants in the competition and n_i is the number of games played by participant i.

From this definition the score of participants should also be dependent on their skill level. The problem becomes a matrix equation:

$$A\vec{r} = \lambda\vec{r} \tag{3.7}$$

where A is the matrix with the terms $A_{ij} = \frac{a_{ij}}{n_i}$ meaning that the ranking vector r is a positive eigenvector of the positive matrix A associated with the eigen value λ. In other words, it extracts the eigenvector from the encounter matrix to produce the vector of scores. The solution exists thanks to the Perron-Frobenius theorem which says that: "If the matrix A has non-negative entries, then there exists an eigenvector r with non-negative entries, corresponding to a positive eigenvalue λ. Furthermore, if the matrix A is irreducible, the eigenvector r has strictly positive entries, is unique and simple, an the corresponding eigenvalue is the largest eigen value of A in absolute value."

This method shows a high bias towards the number of matches played and is extremely weak on small data sets. The algorithm based on the eigenvectors has three of the four advantages that the Colley matrix has such as:

- is easily reproducible,
- adjusts for strength of the whole schedule,
- and is time efficient.

However, regarding the disadvantages, the results, even though they seem quite good, are actually less relevant than the ones given by Colley's matrix because of a bias towards the number of matches played.

Example 3 *From the previously presented simple example, with the adjacency matrix:*

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

The algorithm takes the eigen values of the matrix as well as its corresponding eigen vectors. The corresponding eigen vectors of the eigen values are the columns of the matrix of eigen vectors.

$$eigenvalues = \begin{bmatrix} 1.4 & -0.46 + 1.14j & -0.46 - 1.14j & -0.47 \end{bmatrix}$$

$$eigenvectors = \begin{bmatrix} 0.63 & 0.62 & 0.62 & 0.14 \\ 0.55 & 0.01 + 0.42j & 0.01 - 0.42j & -0.7 \\ 0.32 & -0.3 + 0.29j & -0.3 - 0.29j & 0.63 \\ 0.45 & -0.19 - 0.47j & -0.19 + 0.47j & -0.3 \end{bmatrix}$$

By taking the maximum positive eigen value of A, the corresponding eigen vector is real and is equal to:

$$matrixvalues = \begin{bmatrix} 0.62562957 & 0.55162833 & 0.32133575 & 0.44837167] \end{bmatrix}$$

By ordering the teams with their respective ratings, the obtained ranking is $[a, b, d, c]$.

3.3 Singular Value Decomposition

This algorithm is inspired from [1] which states that the applicability of the SVD-Rank approach stems from the observation that the noiseless matrix of rank offsets $C = (C_{ij})_{1 \leq i,j \leq n}$, is a skew-symmetric matrix of even rank 2 since $R = re^T - er^T$, where e denotes the all-ones column vector. In the noisy case, C is a random perturbation of a rank-2 matrix.

Thus, the algorithm uses the two first left singular vectors of *Amatrix* and computes which ones is the best with regard to the number of upsets of the ranking. It uses the singular vectors from the singular value decomposition of the matrix $AMatrix$ which is expressed as:

$$AMatrix = U\Sigma V^* \tag{3.8}$$

where $AMatrix$ is the adjacency matrix, U the matrix of the right-singular vectors, Σ the diagonal matrix of singular values of the $AMatrix$ and V the matrix of the left-singular vectors. It has been noted that the first two vertical vectors of U can be used as classifications thanks to the absolute value of these vectors.

This method can be compared to the eigen value method and both did not obtain extremely convincing results while still getting some rankings close to the real one. The SVD ranking method is extremely bad for large data sets but great when used on complete and small data sets.

The results obtained are sometimes quite far from the reality while staying believable. Another disadvantages of the algorithm is the computation time which is quite large as the function tests the different possibilities when associating multiple singular vectors.

Regarding the advantages the ranking through SVD:

- is easily reproducible,
- and adjusts for strength of the whole match schedule.

Example 4 *From the previously presented simple example, with the adjacency matrix:*

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

The algorithm computes the matrix of the the right-singular vectors U, whose first two vectors are in absolute value:

$$u_1 = \begin{bmatrix} 0.59100905 & 0.73697623 & 0.32798528 & 0 \end{bmatrix}$$

$$u_2 = \begin{bmatrix} 0.73697623 & 0.32798528 & 0.59100905 & 0 \end{bmatrix}$$

Leading to the rankings being: $r_1 = [a, b, d, c]$ and $r_2 = [b, a, c, d]$. As the two rankings have the same number of upsets being 2, the first one is taken.

3.4 Bradley-Terry

Bradley-Terry's method is based on the maximum likelihood and is presented in the paper written by Bradley and Terry [9]. This algorithm computes the maximum likelihood to win matches against the other participants to predict the outcome of pairwise comparisons. The maximum likelihood estimate is expressed as:

$$L(p) = \sum_{i}^{n} \sum_{j}^{n} [w_{ij} * \ln(p_i) - w_{ij} * \ln(p_i + p_j)] \tag{3.9}$$

With the correction term being computed as:

$$p_i = \frac{W_i}{\sum_{j \neq i} \frac{w_{ij} + w_{ji}}{p_i + p_j}} \tag{3.10}$$

where p_i is the rating of the team i, W_i is the number of wins of team i, w_{ij} is the number of wins of team i against team j. The scoring of each team is recorded in an array. The recursion of the algorithm improves the log-likelihood every iteration and converges towards a unique maximum with corrections being smaller and smaller in size. Concerning the results, they are often quite close to Colley's method and even better on small data sets which puts it as one of the best algorithm found in the literature. The two main characteristics of the algorithm that differentiate from the closest other ranking system are:

- Bradley-Terry can also be run on a rating scale meaning that the matrix of encounters created with the win rates of each teams can be used for this algorithm.

- Bradley-Terry scores are on a ratio scale which means that when the team i has k times the score of the team j it means it is k times "better", or at least that the algorithm estimates the probability of winning for team i over team j as k times higher.

Regarding the advantages of the algorithm they are the same as for the Colley matrix meaning it is one of the best presented while being able to deal with singular matrices. The Bradley-Terry ranking system:

- is easily reproducible,
- adjusts for strength of the whole schedule,
- is time efficient,
- and produces common sense results.

Example 5 *From the previously presented simple example, with the adjacency matrix:*

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

By initializing the 4 entries in the parameter vector p arbitrarily, such as $p = [1, 1, 1, 1]$*, the algorithm can start recursively computing the update of the vector as:*

$$p_1 = \frac{1+0+1}{\frac{1+0}{1+1} + \frac{0+1}{1+1} + \frac{1+0}{1+1}} = 1.333$$

$$p_2 = \frac{0+1+1}{\frac{0+1}{1+1} + \frac{1+0}{1+1} + \frac{1+0}{1+1}} = 1.333$$

$$p_3 = \frac{1+0+0}{\frac{1+0}{1+1} + \frac{0+1}{1+1} + \frac{0+1}{1+1}} = 0.667$$

$$p_4 = \frac{0+1+0}{\frac{0+1}{1+1} + \frac{1+0}{1+1} + \frac{0+1}{1+1}} = 0.667$$

By normalizing p, we obtain:$p = \begin{bmatrix} 0.667 & 0.667 & 0.333 & 0.333 \end{bmatrix}$

The formula (3.10) *is then applied again on the normalized vector.*

$$p_1 = \frac{1+0+1}{\frac{1+0}{0.7+0.7} + \frac{0+1}{0.7+0.3} + \frac{1+0}{0.7+0.3}} = 0.727$$

$$p_2 = \frac{0+1+1}{\frac{0+1}{0.7+0.7} + \frac{1+0}{0.7+0.3} + \frac{1+0}{0.7+0.3}} = 0.727$$

$$p_3 = \frac{1+0+0}{\frac{1+0}{0.3+0.7} + \frac{0+1}{0.3+0.7} + \frac{0+1}{0.3+0.3}} = 0.286$$

$$p_4 = \frac{0+1+0}{\frac{0+1}{0.3+0.7} + \frac{1+0}{0.3+0.7} + \frac{0+1}{0.3+0.3}} = 0.286$$

normalized again the vector p is then: $p = \begin{bmatrix} 0.69 & 0.69 & 0.271 & 0.271 \end{bmatrix}$

After multiple iterations the algorithm converges to $p = \begin{bmatrix} 0.671 & 0.671 & 0.224 & 0.224 \end{bmatrix}$

The algorithm classifies the teams a and b as the two best teams and the teams c and d as the last teams with the same ratings for each pair of teams as the Colley matrix has done.

3.5 ELO

The purpose of this section is to briefly introduce the ELO ranking system which is one of the mostly used ranking algorithm world wide and some of the following algorithms take some inspiration from it.

The ELO ranking system is based upon the computation of the expected score of both players against each other as:

$$E_A = \frac{1}{1 + 10^{(r_B - r_A)/400}} \tag{3.11}$$

where r_B and r_A are the ratings of both team A and team B. The expected score is then used to compute the gain or loss of points thanks to the comparison with the actual result of the game:

$$r'_A = r_A + K \cdot (S_A - E_A) \tag{3.12}$$

where S_A is the result of the match, K is a factor often given the value 32 and r'_A is the new rating of the team A. Here the order of the matches is important because the rating of the opponent influences the point gain or loss of the participant.

Example 6 *From the previously presented simple example, with the adjacency matrix:*

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

However, as the order of the matches is important, let us assume that the matches are played in the following order:

- *a plays against b*
- *c plays against d*
- *a plays against c*
- *b plays against d*
- *a plays against d*
- *b plays against c*

In this order the ratings starting from 1000 rating points for each participants evolves as:

- $r_A = 1000 + 32 * (1 - \frac{1}{1+10^0}) = 1016$
- $r_B = 1000 - 32 * (1 - \frac{1}{1+10^0}) = 984$
- $r_C = 1000 - 32 * (1 - \frac{1}{1+10^0}) = 984$
- $r_D = 1000 + 32 * (1 - \frac{1}{1+10^0}) = 1016$
- $r_A = 1016 - 32 * (1 - \frac{1}{1+10^{(1016-984)/400}}) = 998.53$
- $r_C = 984 + 32 * (1 - \frac{1}{1+10^{(1016-984)/400}}) = 1001.47$
- $r_B = 984 + 32 * (1 - \frac{1}{1+10^{(1016-984)/400}}) = 1001.47$
- $r_D = 1016 - 32 * (1 - \frac{1}{1+10^{(1016-984)/400}}) = 998.53$
- $r_A = 998.53 + 32 * (1 - \frac{1}{1+10^{(998.53-998.53)/400}}) = 1014.53$
- $r_D = 998.53 - 32 * (1 - \frac{1}{1+10^{(998.53-998.53)/400}}) = 982.53$
- $r_B = 1001.47 + 32 * (1 - \frac{1}{1+10^{(1001.47-1001.47)/400}}) = 1017.47$
- $r_C = 1001.47 - 32 * (1 - \frac{1}{1+10^{(1001.47-1001.47)/400}}) = 985.47$

Leading to the vector of ratings: $r = [1014.53, 1017.47, 985.47, 982.53]$ *and the produced ranking being [b, a, c, d].*

Chapter 4

Suggested Algorithms

4.1 Recursive ELO

As all the previous algorithms are coming from the literature, the following systems are the ones that are proposed in this thesis. One of the issues faced when utilizing the famous ELO system is the need for a score which represents the actual skill of a participant. In the case of chess, considering the number of matches recorded between the players, their ELO score represents their potential of winning against another player. In the case of FIFA ranking, the number of matches is quite small with the starting score for each team that was decided by the FIFA organisation in 2018. Can these rating be assumed to represent the actual skill of the different teams?

In order to reduce the issue of the skill not being represented by their score. This thesis proposes a recursive ELO (RELO) system which computes multiple times the rating of the participants using the ELO system but with a lower coefficient on the point gain or loss after a match. This change results in the ratings getting longer to increase. By going multiple times over the data, the rating can be assumed to be closer to the actual skill of the participant enabling the comparison between the different teams. The algorithm converges towards a ranking which is more reliable that the ELO ranking implemented by the FIFA organisation.

Example 7 *From the previously presented simple example, with the adjacency matrix:*

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

The algorithm initializes the ratings of all teams as 1000 *to keep the same scale as the ELO ranking system. And it computes for each match the gain and the loss of points. When team a beats team b it gets one point and team b losses one:*

$$\begin{bmatrix} 1001 & 999 & 1000 & 1000 \end{bmatrix}$$

The same goes for each of the matrix's entries that are greater than one, leading to the array of ratings evolving as follows:

$$\begin{bmatrix} 1002 & 999 & 999 & 1000 \end{bmatrix}$$

$$\begin{bmatrix} 1002 & 1000 & 998 & 1000 \end{bmatrix}$$

$$\begin{bmatrix} 1002 & 1001 & 998 & 999 \end{bmatrix}$$

$$\begin{bmatrix} 1002 & 1001 & 999 & 998 \end{bmatrix}$$

$$\begin{bmatrix} 1001 & 1001 & 999 & 999 \end{bmatrix}$$

A second run in the ELO algorithm leads to the array of ratings:

$$\begin{bmatrix} 1001.988 & 1001.988 & 998.012 & 998.012 \end{bmatrix}$$

After 100 computations, as the ratings keep diverging from the mean rating of 1000*, the algorithm outputs the array of ratings:*

$$\begin{bmatrix} 1060.275 & 1060.275 & 939.725 & 939.725 \end{bmatrix}$$

Leading to teams a and b being ranked equal for first place and the same goes for the teams c and d for last place.

4.2 Optimization through QAP

The idea of optimizing the number of upsets comes from the measurement method which computes the number of wrongly predicted matches. By optimizing this measure the ranking would be the best possible. However, the problem to be solved is a quadratic assignment problem with multiple solutions which was first proven to be extremely difficult to solve.[11] A similar approach has been presented in the paper written by Cassady, Maillart, and Salman.[10]

The representation of the ranking problem can be expressed as a matrix of boolean where: $Matrix_{ij} = 1$ when the team i is ranked at the j^{th} rank and $Matrix_{ij} = 0$ otherwise. The idea is to minimize the function of the number of badly predicted matches which was presented in the section about the measures as the number of upsets. In this representation, the number of upsets is computed differently as the ranking matrix is different. It is expressed as:

$$upsets = \sum_{ijkl}^{N} (r_{ik} * r_{jl}) * AMatrix_{ij} \tag{4.1}$$

The 2 constraints that are part of the ranking problem in this case are:

$$\forall i, \sum_{j}^{N} Matrix_{ij} = 1 \tag{4.2}$$

which represents the need for each team to have one and only one ranking. But also one ranking can be attributed to only one team which is expressed as:

$$\forall j, \sum_{i}^{N} Matrix_{ij} = 1 \tag{4.3}$$

However, the size of the search tree grows extremely fast when the optimum is not close to zero and makes the machine stops because of the lack of memory. Thus ranking clusters and merging them in order to obtain a global ranking is a solution to cut off the size of the search tree. The clustering can be done in multiple ways such as clustering in continents or even divide in groups of a certain length according to the alphabetical ordering of their names or any kind of differentiation that can be applied on the participants.

However, as the merging is quite difficult to accomplish, one way to cluster the teams is to already take off the participants that have no win or no loss and in the end, put them as last or first respectively in the computed ranking. Then, if this solution does not provide better computational times (does not remove any team in the premier league for example), a clustering algorithm is implemented in order to find two clusters which, when merged with one entirely in front of the other, provides the least amount of upsets. The optimization approach is then applied on each cluster to output a ranking on the entirety of the participants.

QAP algorithm is the ranking system that produces the ranking that shows the least amount of wrongly predicted matches. However, considering that QAP is an NP-hard problem, the system has no known algorithm that can solve it in polynomial time leading to the time consuming algorithm it is. It is not as easy to reproduce as the algorithm needs a lot of memory and time to output a ranking. On the other hand, the algorithm has still some of the advantages shown by the other algorithm and has the best potential of all to reach the best possible ranking. The assets of the system are:

- it adjusts for strength of the whole match schedule,
- and produces common sense results.

Example 8 *From the previously presented simple example, with the adjacency matrix:*

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

The algorithm creates the sum of the upsets as described in the section 2.4 with the equation (2.4)*. In this case it leads to the minimization of:*

$$\begin{aligned} &r_{01}*r_{10}+r_{02}*r_{10}+r_{02}*r_{11}+r_{03}*r_{10}+r_{03}*r_{11}+r_{03}*r_{12}+r_{01}*r_{20}+r_{02}*r_{20}+ \\ &r_{02}*r_{21}+r_{03}*r_{20}+r_{03}*r_{21}+r_{03}*r_{22}+r_{11}*r_{20}+r_{12}*r_{20}+r_{12}*r_{21}+r_{13}*r_{20}+ \\ &r_{13}*r_{21}+r_{13}*r_{22}+r_{11}*r_{30}+r_{12}*r_{30}+r_{12}*r_{31}+r_{13}*r_{30}+r_{13}*r_{31}+r_{13}*r_{32}+ \\ &r_{21}*r_{00}+r_{22}*r_{00}+r_{22}*r_{01}+r_{23}*r_{00}+r_{23}*r_{01}+r_{23}*r_{02}+r_{31}*r_{20}+r_{32}*r_{20}+ \\ &r_{32}*r_{21}+r_{33}*r_{20}+r_{33}*r_{21}+r_{33}*r_{22} \end{aligned} \tag{4.4}$$

Leading to the entries of the matrix $[0,0]$*,* $[1,1]$*,* $[2,3]$ *and* $[3,2]$ *being the ones that are equal to one where the others are null. This creates a ranking ["a", "b", "d", "c"] which has a number of upsets of one.*

4.3 Ranking Through Clustering

The idea of Ranking Through Clustering (RTC) system is that the optimization with the QAP problem is not actually doable for large data sets. As a result, the aim is to divide the data set in multiple clusters. With the merging being quite difficult to implement, the idea is to already group in a way that each of the clusters are already ranked between each other with all the best teams in the same cluster.

This algorithm is thus based on the Min-Cut algorithm where the goal is to find the cut between two clusters that minimizes the number of upsets between the two created clusters creating a collection of teams that should be ranked in the top half and a second one of teams that should be ranked in the bottom half. Using this system in order to rank teams or players is not the most obvious classification method. However, the idea seems quite simple as the partition in two classes and, recursively, partition each of the created clusters in subgroups resembles to the merge sort algorithm meaning it ends with groups of only one team ordered in a ranking way. The partition is chosen where the minimum of matches contradicts the ordering.

In this case, the function to be minimized by the minimum cut is the number of contradiction in the matches between the two clusters ranked as r_1 is the upper cluster and r_0 the lower one:

$$Upsets = \sum_{i=1}^{n} \sum_{j=1}^{n} m_{ij} * (r_j - r_i) \tag{4.5}$$

where n is the number of participants in the competition. If implemented with no condition, the partition would always output a ranking with all the teams in the same cluster. The added condition to have the lowest computing time while getting reasonable results is to divide the group in two equally long clusters.

Such a method does not output a scoring for the teams but only an ordering, making it impossible to use the metric from converging gradient ranking to evaluate the results it provides. The algorithm has been implemented through pyomo and the choice for the cut is done through a linear problem solved by the cplex optimizer.

The time complexity of this system is higher than the algorithms that are not recursive but lower than the recursive ones. This is coming for the fact that it only takes into account the matches from the teams in the same cluster and not the ones that are not in it. Regarding the complexity of the algorithm as the resolution of the problem relies mainly on the complexity of the min cut algorithm which is $O(n^5)$ where n is the number of participants. This is used on the encounter matrix and then on the 2 matrices of both created clusters and recursively on each encounter matrix of a cluster. This means that the complexity of the entirety of the computation can be expressed as:

$$C = \sum_{i=1}^{log_2(n)-1} \left(\frac{n}{2^i}\right)^5 2^i \tag{4.6}$$

Which means that the complexity is still of order $O(n^5)$. The Ranking though clustering method shows many advantages such as:

- is easily reproducible,
- adjusts for strength of parts of the match schedule,
- and is time efficient.

However, the results obtained are not as good as it may seem. The algorithm is biased towards the "only winners". The teams that only won in the competition, even with only one win, can be considered as the best teams even though it does not seem like they should be. All of the algorithms shown outputs a rating for the teams except for the quadratic assignment problem and the RTC algorithm where the output is an ordering of the teams.

Example 9 *From the previously presented simple example, with the adjacency matrix:*

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

The algorithm computes the minimum of matches wrongly predicted between the two created clusters. Which in this case is one and the two clusters are ["a"] and ["b", "c", "d"].

Then it gets the clusters ["a"], ["b"] and ["c", "d"] with zero as a minimum of wrongly predicted matches between the two last clusters and the same goes for the last iteration to obtain the ranking:

$$\left["a", "b", "c", "d" \right]$$

4.4 Converging Gradient Ranking

From the minimization of the upset measure, which has been presented above, another scoring method could be implemented in order to better represent the likelihood of a team to win against another one. Such a scoring needs to be continuous in order to reflect this likelihood of a win in the ratings of the teams. Thus a scoring method is designed as a sum of sigmoid and Gaussian functions.

A win or a loss is represented as a sigmoid where the x axis is the difference of ranking between the two rankings. This measure method allows the evaluation of the ranking system from a more continuous function and with better scoring when the winner of a match is further apart from the loser. This measure must have a derivative close to zero when the two teams are far apart, leading to the choice of a sigmoid function. The same thing goes for the Gauss function taken in the case of a draw where the function should be high when the two teams are rated close to each other. This leads to the design of the following function:

$$s_i = \sum_{j}^{N} AMatrix_{ij} * \sigma(r_i - r_j) + Gauss(r_i - r_j) * (DMatrix_{ij}) \tag{4.7}$$

where σ is the sigmoid function, $Gauss$ is the Gaussian function, r_i the rating of team i, $AMatrix$ the adjacency matrix and $DMatrix$ the draw matrix. When the function needs to be a scalar, the Score of the ranking overall is simply denoted by:

$$s = \sum_{i=1}^{n} s_i \tag{4.8}$$

Such measure are presented in the section 2.4 which presents the upsets measure which should be minimized and the score from match prediction which should be maximized. The three following ranking methods are built to output a ranking as optimized as possible regarding the different measures.

An idea, to find an optimum or at least a local optimum for the scoring measure, is to use the gradient descent algorithm on the function of scoring present in the section 2.4 in order to approximate the ranking of a competition by converging towards the best possible ranking. Such a method does not find a global minimum but only a local minimum. A scoring method whose derivative is easily computable is needed to find the gradient of the scoring function such as the above-mentioned scoring method which is based on the sum of the sigmoid in the case of a win or a loss and Gaussian functions in the case of a draw.

The gradient descent algorithm relies on computing a gradient and updating the rating of each team towards the gradient's direction in the case of maximization. One of the issue when using a gradient descent algorithm is when the function which is dealt with is non convex leading to the algorithm reaching local maximums and not the global maximum. Another research about the error analysis of gradient descent ranking system has been presented but shows a different implementation and different test data bases.[2]

Such a method has some resemblance with the famous ELO system with a function which outputs the score added or subtracted to the rating of the winning or losing team and converges towards a ranking. The two algorithms are similar in their construction but different in the way they use the data. ELO uses the data as a sequence and starts from the first match played to end with the last one while the converging gradient ranking goes through the data twice per iteration in order to compute the score and its derivative.

Regarding the advantages of this algorithm, it can achieve results that none of the other algorithms are able to keep up with. The accuracy of this system is not matched when it converges correctly. Only the converging gradient ranking algorithm proposed takes the draws into account when generating the ranking which makes it more dependant on the actual skill level of the participants when draws can be extracted, i.e. in football.

However, the inconveniences are numerous. The robustness seems worse than the one of other algorithms unless computing multiple times on the same data set with different starting points and take the mean of the results to have some kind of ensemble method which would take a long time to compute. It takes quite some time before the algorithm converges towards a local minimum compared to the other presented algorithms that are not recursive because CGR goes through the data multiple times. The algorithm on paper and in the results seems to be the best one. The algorithm is time consuming as the computation of the gradient and the function are quite costly in time. The number of operations needed for the algorithm to converge is quite large compared to the others.

The method of the converging gradient ranking has been implemented with the addition of a step regulator which is based on the Wolfe's conditions to regulate the step's magnitude. The first Wolfe condition is:

$$f(x^k + td) \leq f(x^k) + t\beta_1 \nabla f(x^k)^T d \tag{4.9}$$

Where x^k is the vector of rankings of the teams, t is the step magnitude, d is the direction of the gradient, $f(x^k)$ the sum of all the scores presented above and β_1 is the correction factor. And the second Wolfe condition is:

$$frac\nabla f(x^k + td)^T d \nabla f(x^k)^T d \leq \beta_2 \tag{4.10}$$

Where β_2 is the correction factor and $0 < \beta_1 < \beta_2 < 1$. These two conditions allows for an update of the step towards one that is neither too small nor too large. This makes the algorithm always converge at a reasonable speed. The parameters b1 and b2 both make the condition stricter when they increase.[3]

Example 10 *From the previously presented simple example, with the adjacency matrix:*

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

Initializing randomly the ratings such as:

$$ratings = \begin{bmatrix} 0 & 0.33 & 0.67 & 1 \end{bmatrix}$$

The computed gradient is: $\begin{bmatrix} 0.27 & 0.22 & -0.22 & -0.27 \end{bmatrix}$

The computed optimal step is 8.05 *which leads to the array of rankings being* $\begin{bmatrix} 2.18 & 2.14 & -1.14 & -1.18 \end{bmatrix}$

After subtracting the minimum of the array and normalizing the array, it becomes: $\begin{bmatrix} 1 & 0.988 & 0.0123 & 0 \end{bmatrix}$.

With this rating the gradient becomes: $\begin{bmatrix} 0.251 & 0.147 & -0.147 & -0.251 \end{bmatrix}$

By repeating the procedure 100 *times the array converges towards the ratings of* $\begin{bmatrix} 1 & 0.967 & 0.033 & 0 \end{bmatrix}$. *Which ends up with the ranking [a, b, c, d].*

Chapter 5

Testing and results

5.1 Data sets

Different data bases have been used in order to test the different methods with extremely different encounter matrix especially regarding their densities. The two sports that are looked at are: Football (or soccer) and tennis. These are one of the most known sports in the world, but the ranking systems are really not the same even in the same sport.

Regarding the football rankings, the FIFA ranking method uses the ELO system which is often regarded as one of the best system of ranking. However, international football teams often play the same teams because of the geographic differences: European teams play against European teams, etc. One of the issues of the ELO ranking, as seen in the chess world, is that it does not have a limit. It inflates as time passes by, as matches are played, giving a score to the players today that could not have been reached before. This means that a continent which have more matches played will get higher rankings for its best teams than the other continents and comparison towards other continent might not be accurate. The data base used for the testing of the ranking methods is coming from Kaggle and is provided by Mart Jürisoo[8]. The data base has a lot more matches in the same continent than between continents as shown by the following table:

Continent	EU[1]	SA[2]	NA[3]	AFR[4]	ASIA[5]	OCE[6]
EU	1025	30	24	36	42	1
SA	12	143	29	5	10	0
NA	14	26	350	3	9	2
AFR	3	6	1	699	17	2
ASIA	20	27	18	36	544	8
OCE	0	0	0	2	0	27

Table 5.1: Distribution of matches between the continents in the FIFA international football league.

[1] European countries in the UEFA football association

[2] South American countries in the CONMEBOL football association

[3] North American countries in the CONCACAF football association

[4] African countries in the CAF football association

[5] Asian countries in the AFC football association

[6] Oceanic countries in the OFC football association

It should be also noted that CONMEBOL and OFC associations have a lot less countries than the other associations. Regarding the number of countries in each association: UEFA 55, CONMEBOL 10, CONCACAF 33, CAF 54, AFC 46, OFC 10. Leading to averages of matches played by each teams in the same order: 19, 14, 11, 13, 12, 3.

On the other hand, the championship rankings, used by national leagues such as the Premier League, seems to be better on paper. However, this kind of system does not take into account the level of the team played. In the case of a championship, it seems to be the most suitable given that every team has played each other but not at all when the encounter matrix is quite sparse.

The tennis ranking system does not take into account the level of the opponents. It gives points to each player according to the depth of the tournament where they lost giving the score to the player earned at the end of the tournament and subtracting the score he or she earned during the last year's tournament. This allows for a year long scoring system that evolves with the calendar and does not inflate except when new tournaments are added to the calendar or more points are granted to the players for a tournament. The data set comes from github sustained by Jeff Sackmann and is updated with new results every now and then [7].

In the next section the algorithms are tested and ranked according to the number of wrongly predicted matches as well as a computed score and even according to the speed of the algorithms to fully understand the pros and cons of each of them. Some testing on the robustness of the algorithms when removing or adding synthetic data are also shown in the results.

One last thing to be noted is that the QAP algorithm can not be tested on large data sets (more than 20 teams) due to its immense time and space complexity. As a result, it is tested with the implementation of the RTC algorithm that creates clusters of about 10 to 20 teams which are then passed to the QAP to be ranked inside their own clusters.

These systems have been tested on 2 different data sets in the international football with the matches from the January of 2018 until December of 2021 for the "2018" data set and from the January of 2020 until December of 2021 for the "2020" data set. The Tennis data set is the ATP Tour matches of the year 2020. The real ranking has been extracted from the FIFA's website with the ranking set on the 23rd of December for each data set.

5.2 Computing time

System	2018	2020	Tennis
Colley	0.068	0.053	0.14
Eigen	0.041	0.012	0.047
SVD	0.45	0.31	1.11
BT	10.82	8.29	24.02
RELO	8.43	5.91	18.86
CGR	37.39	21.64	93.10
RTC	13.68	5.71	9.29
QAP	719.66	8.57	24.1

Table 5.2: Table of the computing time for each algorithm in seconds.

In order to better visualize this table, the median computational times of each algorithm is presented in ascending order in the following bar graph:

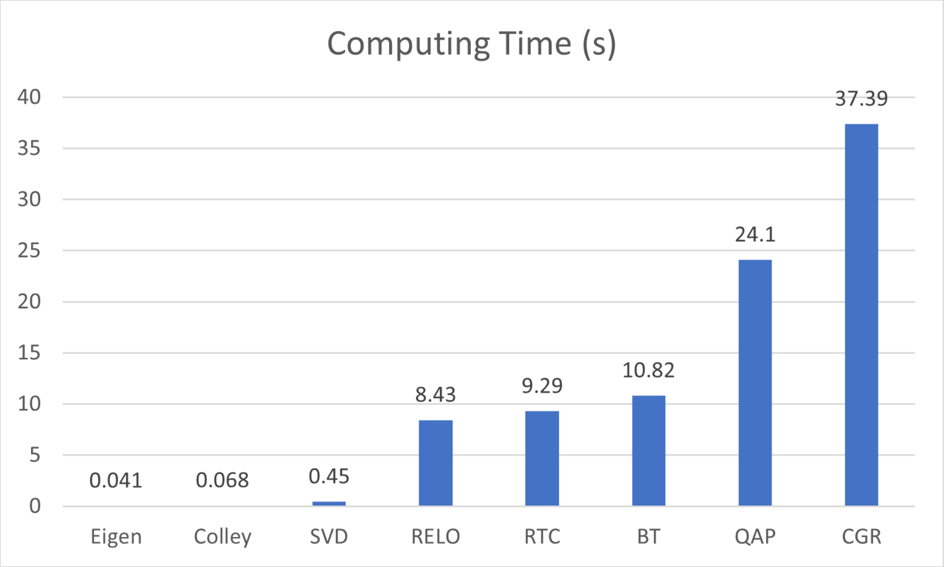

Figure 5.1: Median computing time for each algorithm on the three different data sets.

The tests have been done on multiple data sets. In order to have a better vision of it, the following table contains the compositions of each data set:

Data set	Number of matches	Number of participants
2018	3171	207
2020	1326	188
Tennis	1462	345

Table 5.3: Table containing the composition of each data set

From these results, the proposed algorithms often take a longer time than the algorithm coming from the literature to produce a ranking. However, each algorithm has its own strengths and weaknesses and the following measure are more indicative of the quality of the ranking produced.

The Colley matrix algorithm and the eigen value algorithm outperform every other ranking system regarding the computing time with the only other one that finishes computing in under a second: the SVD ranking system.

5.3 Accuracy

This section presents the measures on the different methods thanks to graphs and tables based on different metrics such as the number of upsets, the time of computation and the learning of some algorithms. As some algorithm can not be measured in some cases, such as the RTC or the QAP which do not output a score for each team only the ones that can be measured are shown in the tables and graphs. The table below shows the results of each algorithm regarding the number of wrongly predicted matches, also called the number of upsets.

Algorithm	2018	2020	Tennis
Real ranking	511	186	443
Colley	468	136	376
Eigen	529	231	477
SVD	538	209	460
BT	456	122	367
CGR	378	122	347
RELO	477	144	387
RTC	471	192	348
QAP	406	149	282
Total	3171	1326	1462

Table 5.4: Table of the number of upsets for each algorithm and for each FIFA data base

Bradley-Terry, Converging gradient ranking and the quadratic assignment problem are the three algorithms, that are definitely better. On the other hand, the SVD and eigen vector ranking system do not achieve great accuracy. The following bar plots help to better visualize the table and the results it presents by ordering the algorithm for each data set.

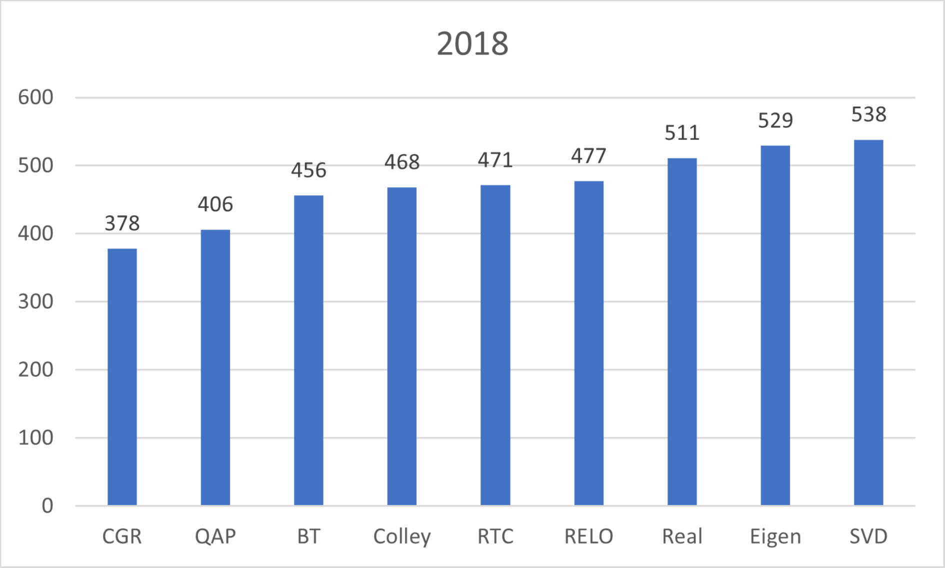

Figure 5.2: Number of upsets measured on the rankings produced by each algorithm on the 2018 FIFA data set.

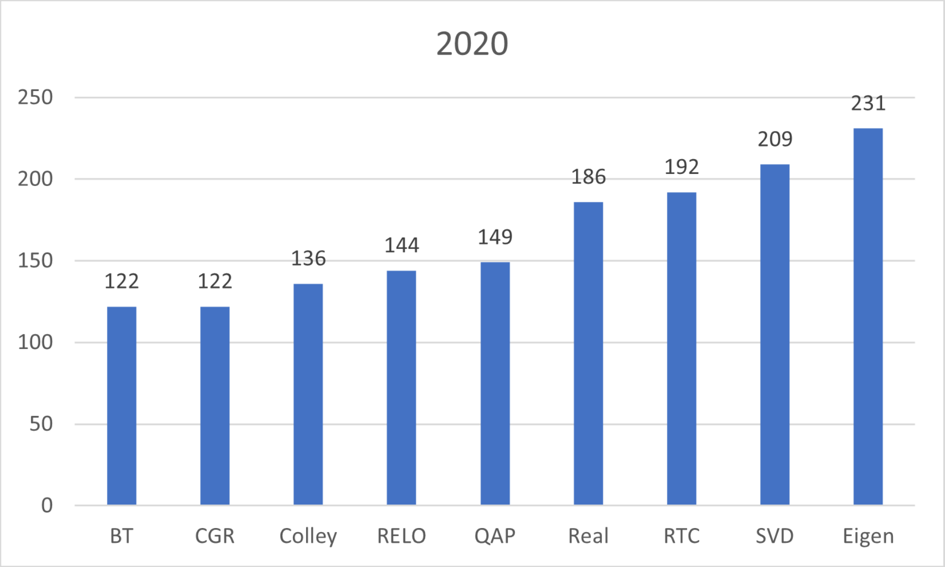

Figure 5.3: Number of upsets measured on the rankings produced by each algorithm on the 2020 FIFA data set.

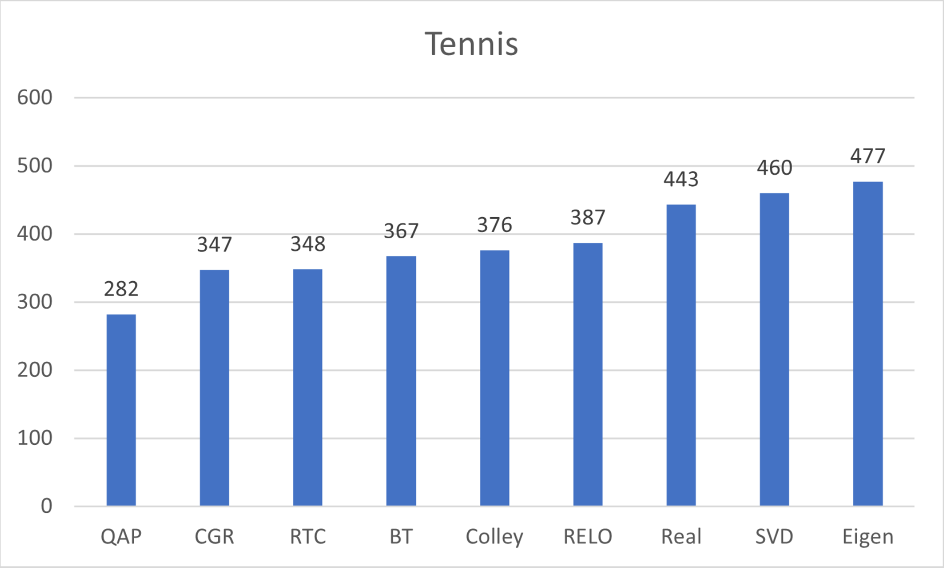

Figure 5.4: Number of upsets measured on the rankings produced by each algorithm on the 2020 Tennis data set.

5.4 Robustness

The robustness is tested thanks to an importance given to the different matches. The importance is implemented as two variables and, in the case of football, represents the importance coefficient between the friendly matches and the continental cups as well as the world cup qualifiers. The second factor represents the coefficient between the world cup and the continental and World cup qualifiers. In the case of tennis the factor either represents the tournament round such as final, semi-final, ... or the importance of the tournament such as Grand Slam and Masters 1000, 500, ... The ranking of each team is stored as a function of the two importance factors

The following graphs presents FIFA's top ten ranked countries at the end of 2021. These graphs show the impact of the importance given to certain matches reflecting the robustness of each algorithm. The x axis represents the importance coefficient between the continental matches and the friendly ones while the y axis represents the coefficient between the world cup matches and the continental matches. The value on each point is the ranking outputted by the algorithm.

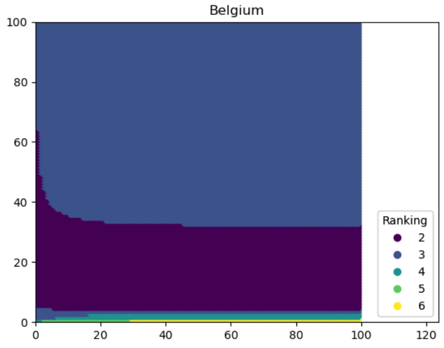

Figure 5.5: Colley ranking system's robustness graph of Belgium

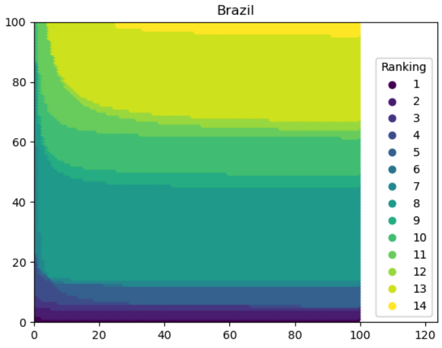

Figure 5.8: Colley ranking system's robustness graph of Brazil

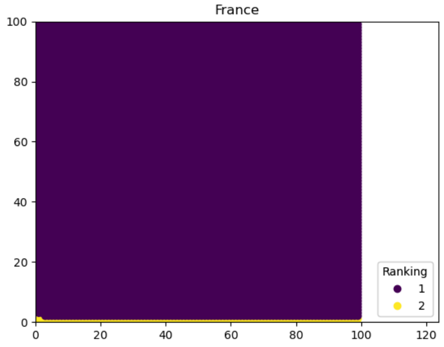

Figure 5.6: Colley ranking system's robustness graph of France

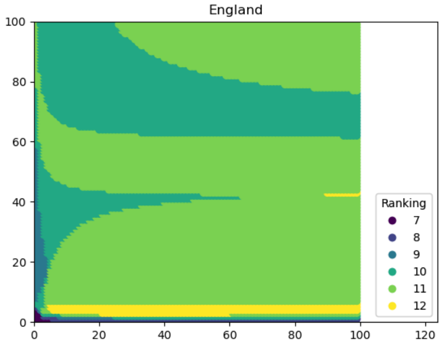

Figure 5.9: Colley ranking system's robustness graph of England

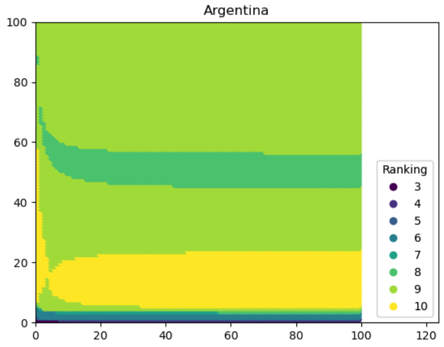

Figure 5.7: Colley ranking system's robustness graph of Argentina

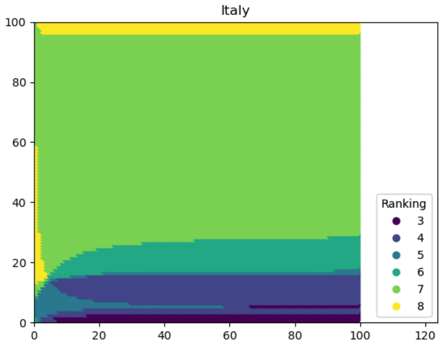

Figure 5.10: Colley ranking system's robustness graph of Italy

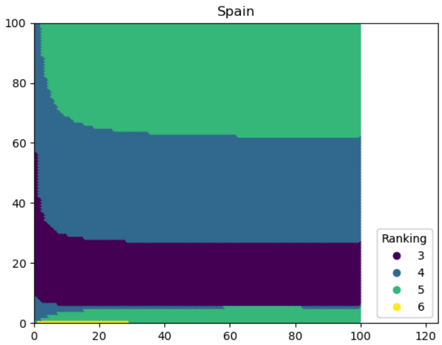

Figure 5.11: Colley ranking system's robustness graph of Spain

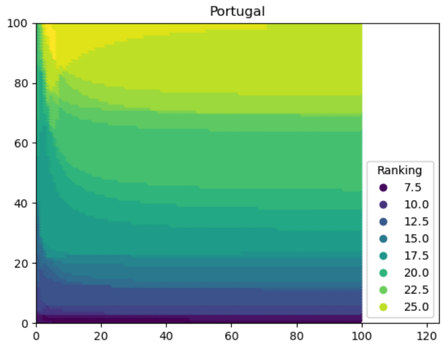

Figure 5.14: Colley ranking system's robustness graph of Portugal

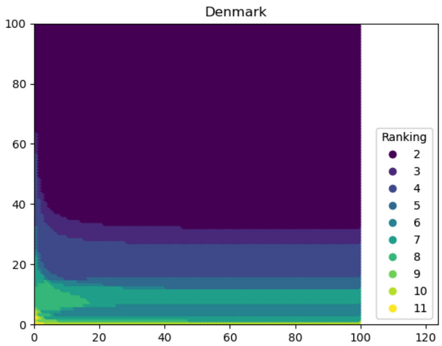

Figure 5.12: Colley ranking system's robustness graph of Denmark

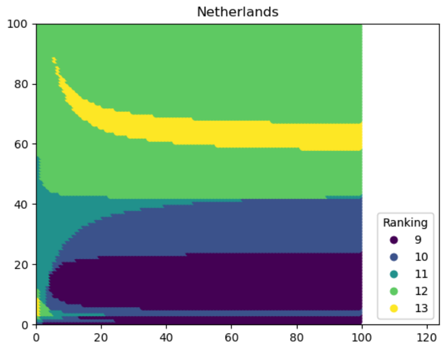

Figure 5.15: Colley ranking system's robustness graph of Netherlands

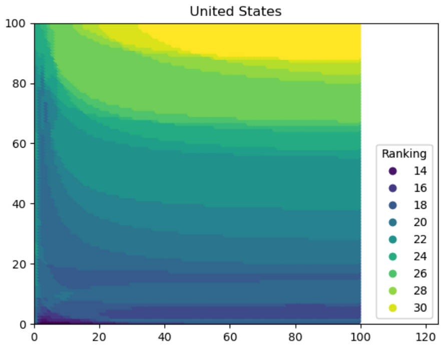

Figure 5.13: Colley ranking system's robustness graph of United States

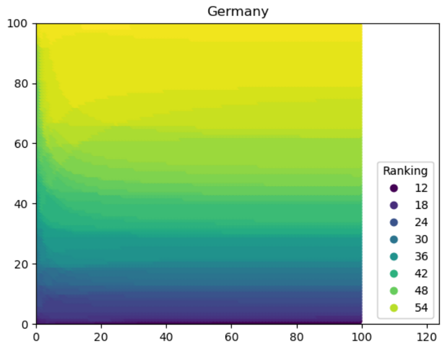

Figure 5.16: Colley ranking system's robustness graph of Germany

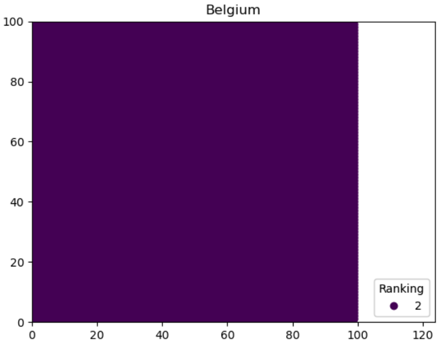

Figure 5.17: Eigenvector ranking's robustness graph of Belgium

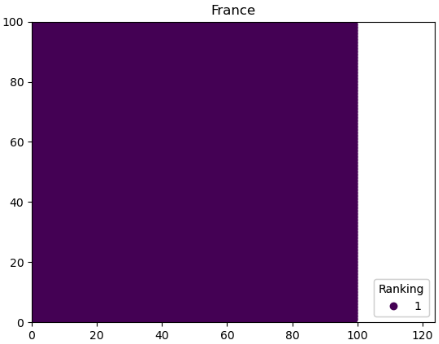

Figure 5.18: Eigenvector ranking's robustness graph of France

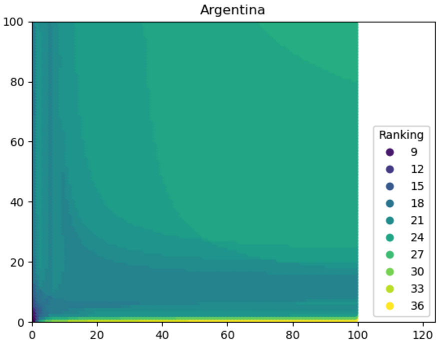

Figure 5.19: Eigenvector ranking's robustness graph of Argentina

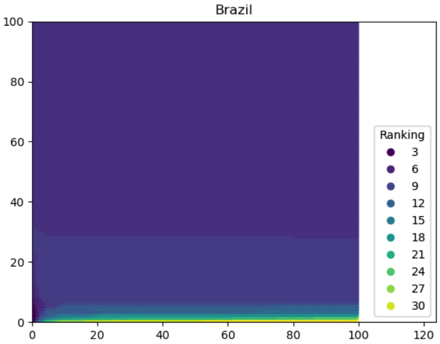

Figure 5.20: Eigenvector ranking's robustness graph of Brazil

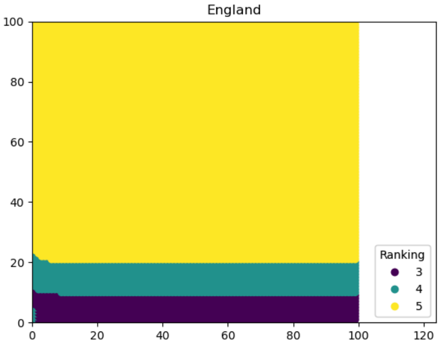

Figure 5.21: Eigenvector ranking's robustness graph of England

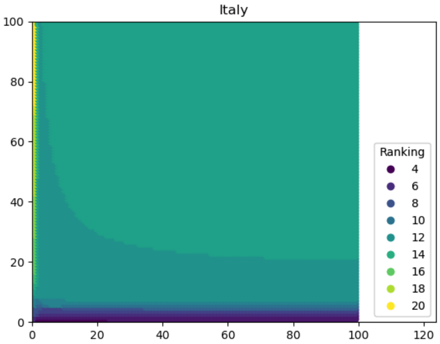

Figure 5.22: Eigenvector ranking's robustness graph of Italy

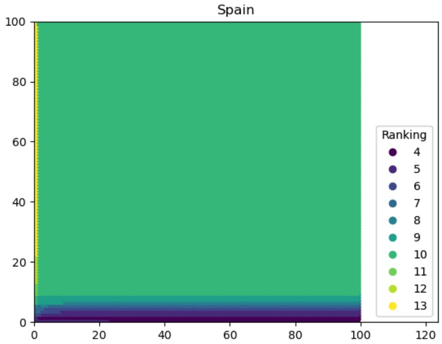

Figure 5.23: Eigenvector ranking's robustness graph of Spain

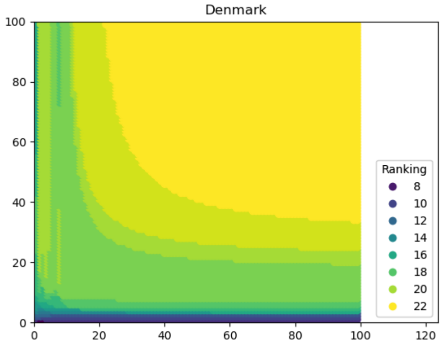

Figure 5.24: Eigenvector ranking's robustness graph of Denmark

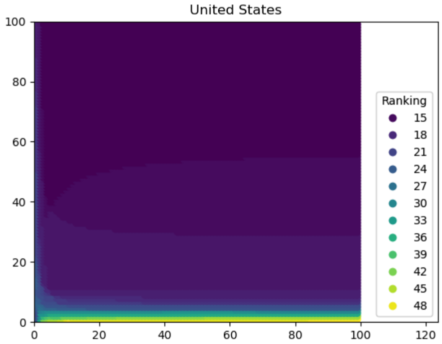

Figure 5.25: Eigenvector ranking's robustness graph of United States

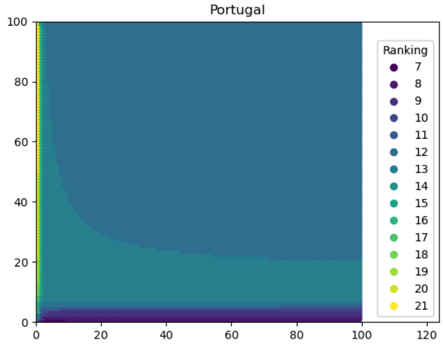

Figure 5.26: Eigenvector ranking's robustness graph of Portugal

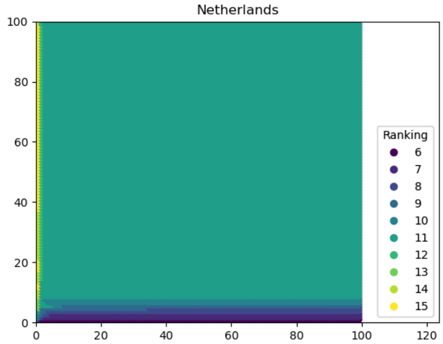

Figure 5.27: Eigenvector ranking's robustness graph of Netherlands

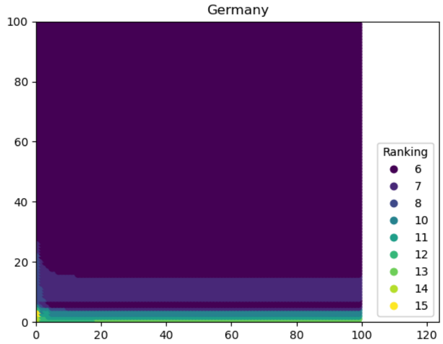

Figure 5.28: Eigenvector ranking's robustness graph of Germany

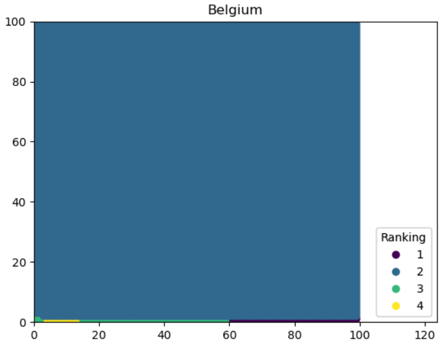

Figure 5.29: Recursive ELO ranking's robustness graph of Belgium

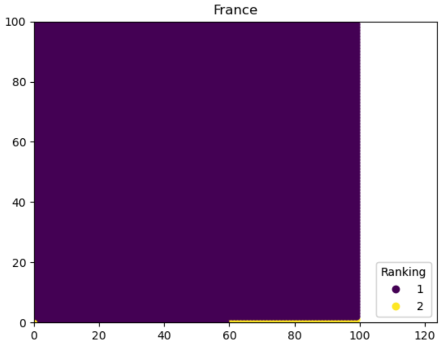

Figure 5.30: Recursive ELO ranking's robustness graph of France

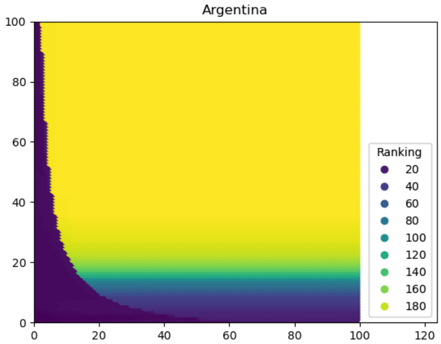

Figure 5.31: Recursive ELO ranking's robustness graph of Argentina

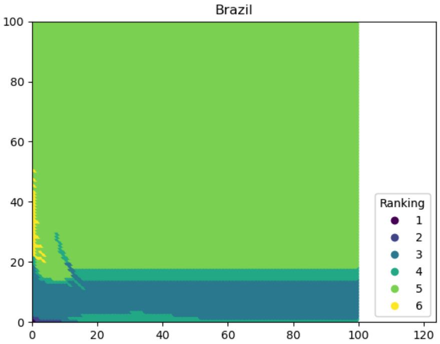

Figure 5.32: Recursive ELO ranking's robustness graph of Brazil

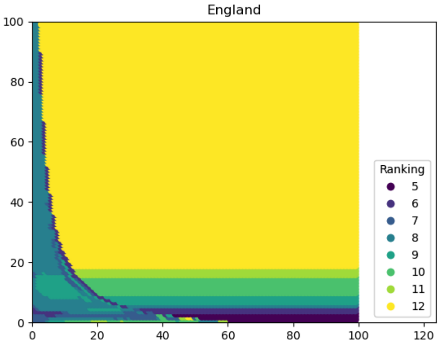

Figure 5.33: Recursive ELO ranking's robustness graph of England

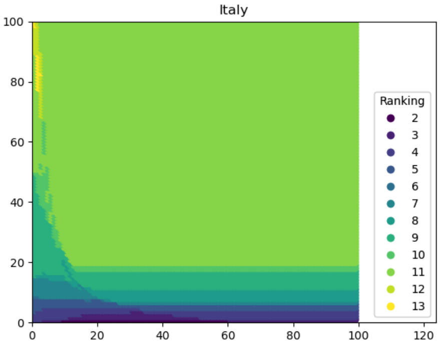

Figure 5.34: Recursive ELO ranking's robustness graph of Italy

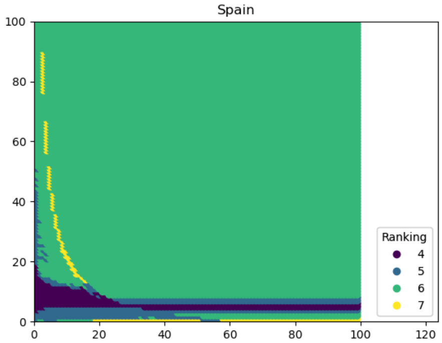

Figure 5.35: Recursive ELO ranking's robustness graph of Spain

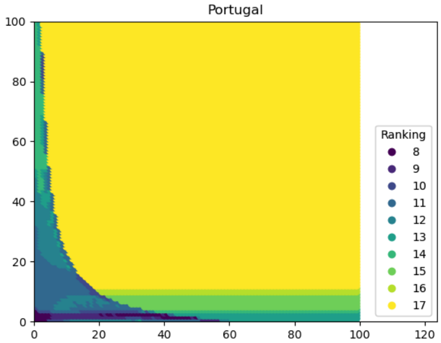

Figure 5.38: Recursive ELO ranking's robustness graph of Portugal

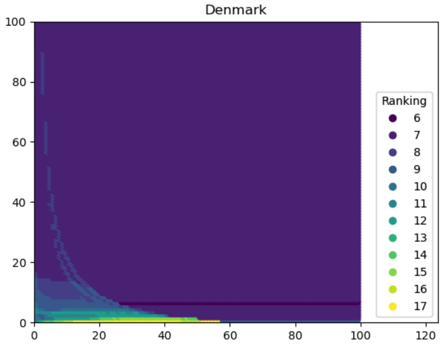

Figure 5.36: Recursive ELO ranking's robustness graph of Denmark

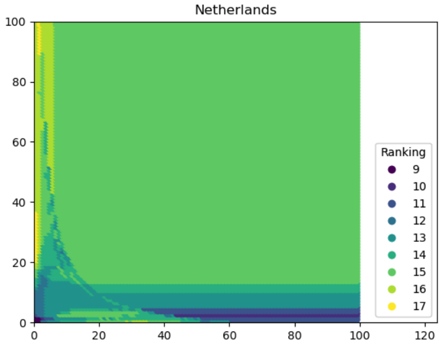

Figure 5.39: Recursive ELO ranking's robustness graph of Netherlands

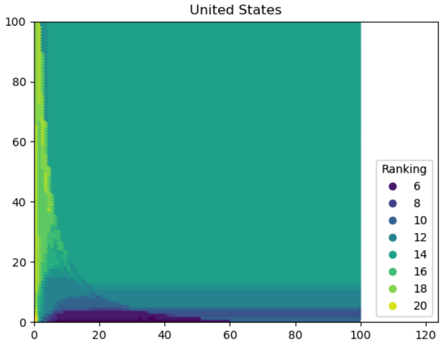

Figure 5.37: Recursive ELO ranking's robustness graph of United States

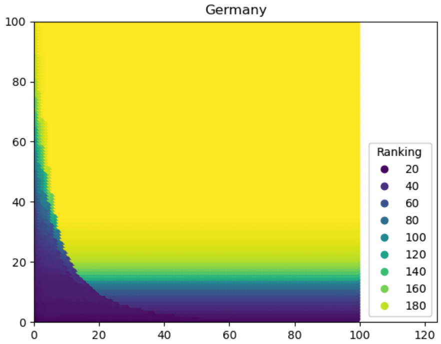

Figure 5.40: Recursive ELO ranking's robustness graph of Germany

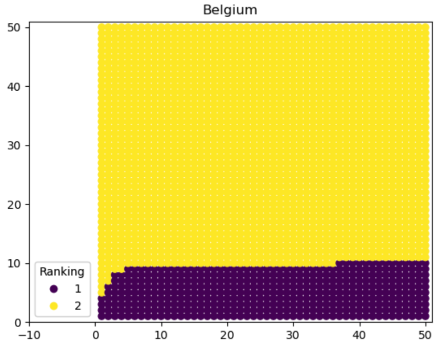

Figure 5.41: CGR ranking system's robustness graph of Belgium

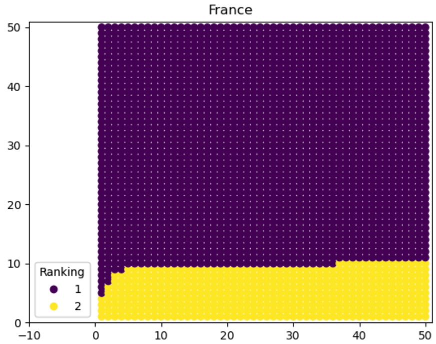

Figure 5.42: CGR ranking system's robustness graph of France

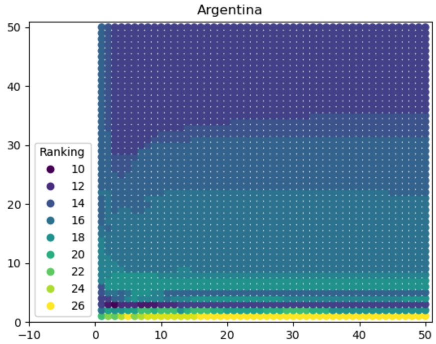

Figure 5.43: CGR ranking system's robustness graph of Argentina

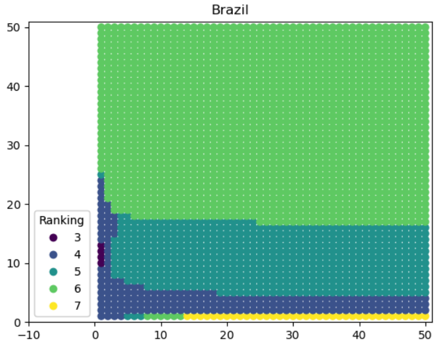

Figure 5.44: CGR ranking system's robustness graph of Brazil

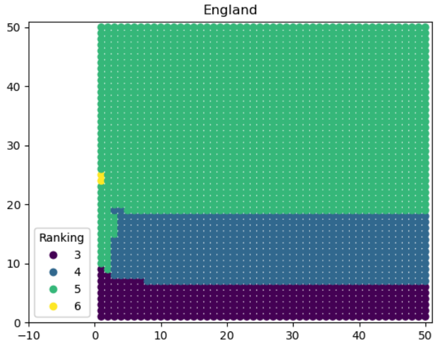

Figure 5.45: CGR ranking system's robustness graph of England

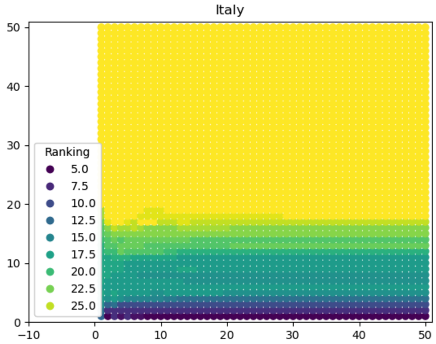

Figure 5.46: CGR ranking system's robustness graph of Italy

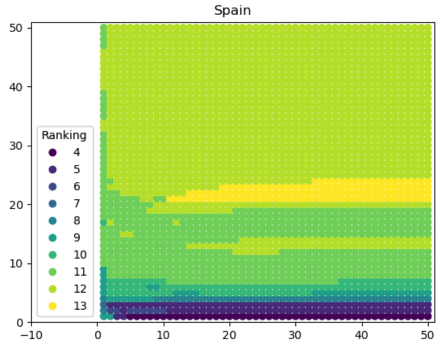

Figure 5.47: CGR ranking system's robustness graph of Spain

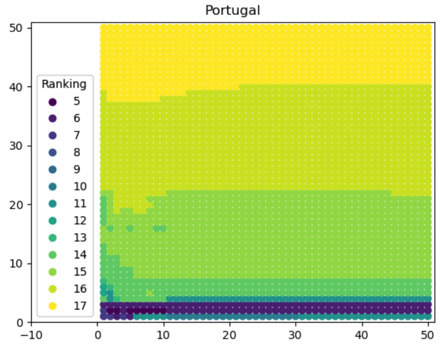

Figure 5.50: CGR ranking system's robustness graph of Portugal

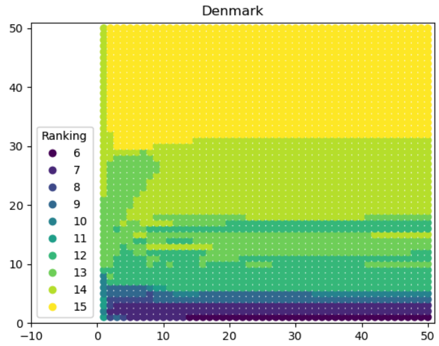

Figure 5.48: CGR ranking system's robustness graph of Denmark

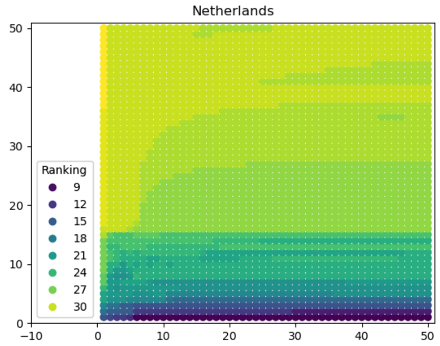

Figure 5.51: CGR ranking system's robustness graph of Netherlands

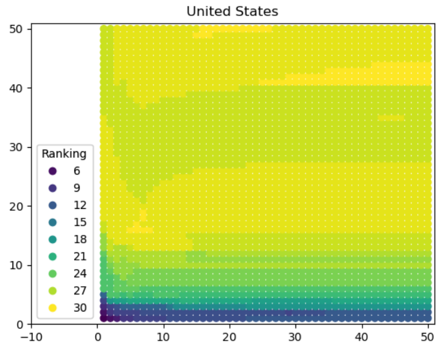

Figure 5.49: CGR ranking system's robustness graph of United States

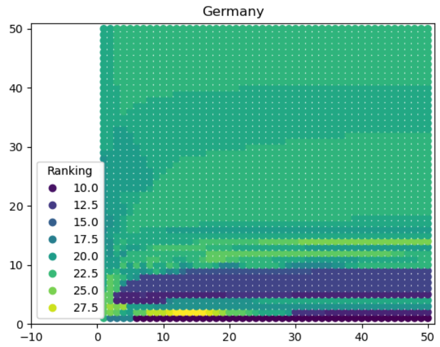

Figure 5.52: CGR ranking system's robustness graph of Germany

The one algorithm that appears as the least robust out of the ones tested is the RELO algorithm where the algorithm can have very different results given different importance coefficients. In the case of Argentine, as an example, the ranking method can assign the ranking of 20 as well as 180 to this country as well as having very large difference in rankings between two neighbouring points.

5.5 Conclusion and further improvements

The different algorithms come with a set of advantages and disadvantages. The three best ranking systems in terms of accuracy are Bradley-Terry, the quadratic assignment problem and the converging gradient ranking which are the three algorithms that take the longest to produce a ranking. On the other hand, the Eigen value algorithm is the fastest ranking method coming from the literature, but by offering this advantage, the method presents very unstable results due to a quite low robustness and one of the worst accuracy of them all.

Furthermore, the algorithms are better on some data sets and worst on others such as small data sets that are often better for algorithms that takes longer as they have some type of iteration. Thus, giving more time to compute to the algorithm allows for the production of a better ranking. Sparsity is also a factor that influence the results of algorithms. In our testing the tennis data set is a lot sparser than the football ones which makes the algorithms based on optimization, QAP, CGR and RTC better than the others due to the lack of information in the adjacency matrix. As a result, each data set has its own best algorithm but some are more often better than others.

In the case of international football results, the CGR algorithm is proved to be extremely accurate with some robustness but a high computing time. This kind of algorithm could be implemented but the computing time can raise some doubts about the implementation for real sport rankings. The one that is winning overall against the other ranking systems is probably the Colley matrix algorithm which is the second best in time consumption, third or fourth in accuracy while having a very good robustness and which raises the question: Why is it not more common to rank sport teams with this algorithm? Especially when the real ranks are extremely often far behind the best one in accuracy.

The measure of robustness is based on making some matches more important than others in a competition. However, it can be understood as adding, in the data set, matches that have already been played. Thus, measuring it with the addition of matches that have not yet been played could be a better measure of the robustness. The choice made in this work was to represent graphically the robustness and led to the implementation of this measure which is quite representative of the stability of the algorithms.

Bibliography

[1] d'Aspremont A., Cucuringu M., and Tyagi H. *Ranking and synchronization from pairwise measurements via SVD.* URL: https://arxiv.org/pdf/1906.02746.pdf. (accessed: 26.10.2021).

[2] Hong Chen et al. "Error Analysis of Stochastic Gradient Descent Ranking". In: *IEEE TRANSACTIONS ON CYBERNETICS* 43.3 (2013), pp. 898–909. DOI: 10.1109/tsmcb.2012.2217957.

[3] Bonnans F. et al. *Numerical optimization.* Mathematics Subject Classification. Springer-Verlag Berlin Heidelberg New York, 2000. ISBN: 978-3540354451.

[4] FIFA. *Revision of the FIFA / Coca-Cola World Ranking.* URL: https://digitalhub.fifa.com/m/f99da4f73212220/original/edbm045h0udbwkqew35a-pdf.pdf. (accessed: 20.02.2022).

[5] Keener J-P. "Perron-Frobenius Theorem and the Ranking of Football Teams". In: *SIAM Review* 35.1 (1993), pp. 80–93. URL: http://www.jstor.org/stable/2132526.

[6] Katz J. *Tokyo Olympics: Who Leads the Medal Count?* URL: https://www.nytimes.com/interactive/2021/07/27/upshot/which-country-leads-in-the-olympic-medal-count.html. (accessed: 21.03.2022).

[7] Sackmann J. *ATP Tennis Rankings, Results, and Stats.* URL: https://github.com/JeffSackmann/tennis_atp. (accessed: 06.02.2021).

[8] Jürisoo M. *International football results from 1872 to 2021*. URL: `https://www.kaggle.com/martj42/international-football-results-from-1872-to-2017`. (accessed: 27.10.2021).

[9] Bradley R. and Terry M. "Rank Analysis of Incomplete Block Designs: I. The Method of Paired Comparisons". In: *Oxford University Press on behalf of Biometrika Trust Stable* 39.3/4 (1952), pp. 324–345. URL: `http://www.jstor.com/stable/2334029`.

[10] Cassady R., Maillart L., and Salman S. "Ranking Sports Teams: A Customizable Quadratic Assignment Approach". In: *Interfaces* 35.6 (2005), pp. 497–510. DOI: `10.1287/inte.1050.0171`.

[11] Koopmans T. and Beckmann M. "Assignment Problems and the Location of Economic Activities". In: *Econometrica* 25.1 (1957), pp. 53–76. DOI: `https://doi.org/10.2307/1907742`.

[12] Lucas T. *Ranking algorithms*. URL: `https://github.com/ThomasLucas99/RankingAlgorithms`. (accessed: 21.03.2022).

[13] Colley W. *Colley's Bias Free College Football Ranking Method: The Colley Matrix Explained*. URL: `https://www.colleyrankings.com/matrate.pdf`. (accessed: 20.11.2021).