DS-GA 1014 - Homework 6

Eric Niblock

October 10th, 2020

1. **(2 points). Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix. Show that A is orthogonal if and only if its eigenvalues all have absolute value 1 (i.e. are either +1 or -1).**

The implication must be shown in both directions. We first set out to show that if A is an orthogonal symmetric matrix in $\mathbb{R}^{n \times n}$, then all of its eigenvalues are ± 1. We show the adapted results of a previous homework question here.

We know that A is an orthogonal matrix, and hence preserves the length of any vector is acts upon, since,

$$||A\vec{\mathbf{v}}||^2 = \langle A\vec{\mathbf{v}}, A\vec{\mathbf{v}}\rangle = \vec{\mathbf{v}}^T A^T A\vec{\mathbf{v}} = \vec{\mathbf{v}}^T\vec{\mathbf{v}} = \langle \vec{\mathbf{v}}, \vec{\mathbf{v}}\rangle = ||\vec{\mathbf{v}}||^2 \tag{1}$$

Taking $\vec{\mathbf{v}} \in \mathbb{R}^n$ to be some eigenvector of A. We also have,

$$A\vec{\mathbf{v}} = \lambda\vec{\mathbf{v}} \tag{2}$$

$$\langle A\vec{\mathbf{v}}, A\vec{\mathbf{v}}\rangle = \langle \lambda\vec{\mathbf{v}}, \lambda\vec{\mathbf{v}}\rangle = \lambda^2\langle \vec{\mathbf{v}}, \vec{\mathbf{v}}\rangle = \lambda^2||\vec{\mathbf{v}}||^2 \tag{3}$$

This implies that,

$$||\vec{\mathbf{v}}||^2 = \lambda^2||\vec{\mathbf{v}}||^2 \tag{4}$$

$$\lambda = \pm 1 \tag{5}$$

Which holds for any eigenvector $\vec{\mathbf{v}}$, meaning that we have shown for any A, an orthogonal symmetric matrix in $\mathbb{R}^{n \times n}$, we have that all of it's eigenvalues are ± 1. We now must show that if we have some matrix A with all eigenvalues being ± 1, that it must be the case that A is a symmetric orthogonal matrix.

Take that $\vec{\mathbf{v}} \in \mathrm{I\!R}^{\mathrm{n}}$ represents some eigenvector of the matrix A. Since we have assumed that the eigenvalues of A are all ± 1, then we have that,

$$A\vec{\mathbf{v}} = (\pm 1)\vec{\mathbf{v}} \tag{6}$$

Taking the magnitude of both sides yields,

$$||A\vec{\mathbf{v}}|| = ||(\pm 1)\vec{\mathbf{v}}|| \tag{7}$$
$$\langle A\vec{\mathbf{v}}, A\vec{\mathbf{v}}\rangle = \langle (\pm 1)\vec{\mathbf{v}}, (\pm 1)\vec{\mathbf{v}}\rangle \tag{8}$$
$$\vec{\mathbf{v}}^T A^T A\vec{\mathbf{v}} = \vec{\mathbf{v}}^T\vec{\mathbf{v}} \tag{9}$$

It therefore must be the case that $A^T A = I$, since $\vec{\mathbf{v}}^T A^T A = \vec{\mathbf{v}}^T$, then we would have,

$$\vec{\mathbf{v}}^T Id_n\vec{\mathbf{v}} = \vec{\mathbf{v}}^T\vec{\mathbf{v}} \tag{10}$$
$$\vec{\mathbf{v}}^T\vec{\mathbf{v}} = \vec{\mathbf{v}}^T\vec{\mathbf{v}} \tag{11}$$

Now, given the following proposition,

> *Let $A \in \mathrm{I\!R}^{\mathrm{n\times n}}$. Then if A is an orthogonal matrix, this is equivalent to saying $AA^T = Id_n = A^T A$ [Prop. 1].*

And the fact that $A^T A = I$, it becomes clear that A must be an orthogonal matrix. Having shown the implication in both directions, it becomes clear that A is orthogonal if and only if its eigenvalues all have absolute value 1.

2. (3 points). We say that a symmetric matrix $M \in \mathrm{I\!R}^{\mathrm{n\times n}}$ is positive semi-definite if for all $\vec{\mathbf{x}} \in \mathrm{I\!R}^{\mathrm{n}}$,

$$\vec{\mathbf{x}}^T M\vec{\mathbf{x}} \geq 0$$

(a) Let $A \in \mathbb{R}^{n \times k}$. Show that AA^T is symmetric, positive semi-definite.

First, we know that a matrix A is symmetric if $A = A^T$. Therefore, AA^T is symmetric since,

$$(AA^T)^T = (A^T)^T A^T = AA^T \tag{12}$$

So since we have $AA^T = (AA^T)^T$ we know that AA^T is a symmetric matrix. Now we show that AA^T is positive semi-definite for all $\overrightarrow{\mathbf{x}} \in \mathbb{R}^n$. If this is the case, then,

$$\overrightarrow{\mathbf{x}}^T AA^T \overrightarrow{\mathbf{x}} \geq 0 \tag{13}$$

Which we know to be true, since,

$$\overrightarrow{\mathbf{x}}^T AA^T \overrightarrow{\mathbf{x}} \geq 0 \tag{14}$$

$$(A^T \overrightarrow{\mathbf{x}})^T A^T \overrightarrow{\mathbf{x}} \geq 0 \tag{15}$$

$$\langle A^T \overrightarrow{\mathbf{x}}, A^T \overrightarrow{\mathbf{x}} \rangle \geq 0 \tag{16}$$

$$||A^T \overrightarrow{\mathbf{x}}||^2 \geq 0 \tag{17}$$

And this is clearly the case, since the length of any vector will be a positive quantity or zero. Thus we have shown that $\overrightarrow{\mathbf{x}}^T AA^T \overrightarrow{\mathbf{x}} \geq 0$, and therefore that AA^T is symmetric, positive semi-definite.

(b) Show that a symmetric matrix $M \in \mathbb{R}^{n \times n}$ is positive semi-definite if and only if all its eigenvalues are non-negative.

We must show the implication in both directions. We first show that if a symmetric matrix $M \in \mathbb{R}^{n \times n}$ is positive semi-definite, this implies that the eigenvalues of M are non-negative.

Take $\overrightarrow{\mathbf{x}} \in \mathbb{R}^n$ to be any eigenvector of M. If this is the case then,

$$\overrightarrow{\mathbf{x}}^T M \overrightarrow{\mathbf{x}} = \overrightarrow{\mathbf{x}}^T \lambda \overrightarrow{\mathbf{x}} = \lambda \overrightarrow{\mathbf{x}}^T \overrightarrow{\mathbf{x}} = \lambda ||\overrightarrow{\mathbf{x}}||^2 \geq 0 \tag{18}$$

Since we know that $\lambda||\vec{\mathbf{x}}||^2 \geq 0$, and we also know that $||\vec{\mathbf{x}}||^2$ is positive (it should be noted that the zero-vector cannot, by definition, be an eigenvector), this implies that λ must be positive for any eigenvector associated to M. So, if a symmetric matrix $M \in \mathbb{R}^{n \times n}$ is positive semi-definite, this implies that the eigenvalues of M are non-negative.

Now we must show that if all of the eigenvalues of a symmetric matrix $M \in \mathbb{R}^{n \times n}$ are non-negative, than M is a positive semi-definite matrix.

Proof by contradiction. Assume that symmetric matrix M is not a positive semi-definite matrix, and that it possesses all positive eigenvalues. We have the following proposition,

> *Let $A \in \mathbb{R}^{n \times n}$ be a symmetric matrix. Then there is a orthonormal basis of $\mathbb{R}^n$ composed of eigenvectors of A. [Prop. 2].*

Since we assume that M is not a positive semi-definite matrix, we have at least one vector, call it $\vec{\mathbf{x}}$, such that,

$$\vec{\mathbf{x}}^T M \vec{\mathbf{x}} < 0 \tag{19}$$

However by *Proposition 2*, we have that $\vec{\mathbf{x}}$ can be composed of a linear combination of the eigenvectors of M, call them $\{\vec{\mathbf{v}}_1, ..., \vec{\mathbf{v}}_n\}$, so $\vec{\mathbf{x}} = c_1\vec{\mathbf{v}}_1 + ... + c_n\vec{\mathbf{v}}_n$, and,

$$(c_1\vec{\mathbf{v}}_1 + ... + c_n\vec{\mathbf{v}}_n)^T M (c_1\vec{\mathbf{v}}_1 + ... + c_n\vec{\mathbf{v}}_n) < 0 \tag{20}$$

$$(c_1\vec{\mathbf{v}}_1^T + ... + c_n\vec{\mathbf{v}}_n^T)(c_1\lambda_1\vec{\mathbf{v}}_1 + ... + c_n\lambda_n\vec{\mathbf{v}}_n) < 0 \tag{21}$$

Now, note that when we perform this multiplication, if $i \neq j$, we have, $\vec{\mathbf{v}}_i^T\vec{\mathbf{v}}_j = 0$ since for all $i \neq j$, we have, $\vec{\mathbf{v}}_i \perp \vec{\mathbf{v}}_j$. This too follows from *Proposition 2*, which notes the that the set of eigenvectors of A forms an orthonormal basis. So, the remaining terms from the multiplication are,

$$c_1^2\lambda_1\vec{\mathbf{v}}_1^T\vec{\mathbf{v}}_1 + ... + c_n^2\lambda_n\vec{\mathbf{v}}_n^T\vec{\mathbf{v}}_n < 0 \tag{22}$$

$$c_1^2\lambda_1||\vec{\mathbf{v}}_1|| + ... + c_n^2\lambda_n||\vec{\mathbf{v}}_n)|| < 0 \tag{23}$$

However, this leads to a contraction because we know that $\lambda_i > 0$, $c_i^2 > 0$ and $||\vec{\mathbf{v}}_i||^2 > 0$ for all i. Therefore, we have shown that if all of the eigenvalues of a symmetric matrix $M \in \mathbb{R}^{\mathrm{n\times n}}$ are non-negative, than M is a positive semi-definite matrix.

Having shown the implication in both direction, we have shown that a symmetric matrix $M \in \mathbb{R}^{\mathrm{n\times n}}$ is positive semi-definite if and only if all its eigenvalues are non-negative.

(c) Let $M \in \mathbb{R}^{\mathrm{n\times n}}$ be a (symmetric) positive semi-definite matrix. Let $r = rank(M)$. Show that there exists $A \in \mathbb{R}^{\mathrm{n\times r}}$ such that $M = AA^T$.

We have the following proposition, also associated with Spectral Decomposition,

> *Let $A \in \mathbb{R}^{\mathrm{n\times n}}$ be a symmetric matrix. Then there exists an orthogonal matrix P and a diagonal matrix D of sizes $n \times n$, such that,*
>
> $$A = PDP^T$$
>
> *[Prop. 3]*

Then, it follows that we can write $M = PDP^T$, where D is formed from the eigenvalues of M and P is formed from the corresponding eigenvectors of M, associated to the eigenvalues in D. Now, since D is formed of non-negative eigenvalues, since M is a symmetric positive semi-definite matrix, it follows that the we can take the square root of each diagonal element of D, and call this new matrix $D^{1/2}$. Furthermore, it is clear that,

$$D = D^{1/2} D^{1/2} \tag{24}$$

Since the elements the i-th row of $D^{1/2}$ is only populated in the i-th position, and the i-th column of $D^{1/2}$ is only populated in the i-th position. Therefore, the multiplication of any i-th row onto any column only produces a non-zero result when considering the corresponding i-th column. So, we write,

$$M = PD^{1/2} D^{1/2} P^T \tag{25}$$

Furthermore, it is clear that $D^{1/2} = (D^{1/2})^T$, since every diagonal matrix is also a symmetric matrix, and symmetric matrices are equal to their transposes. So it is evident that,

$$M = PD^{1/2}D^{1/2}P^T = PD^{1/2}(PD^{1/2})^T = AA^T \tag{26}$$

If we take $A = PD^{1/2}$. So for any $M \in \mathbb{R}^{\mathrm{n}\times\mathrm{n}}$ which is a symmetric positive semi-definite matrix, regardless of rank, we have that there exists $A \in \mathbb{R}^{\mathrm{n}\times\mathrm{n}}$ such that $M = AA^T$. However, to show that we can produce an $A \in \mathbb{R}^{\mathrm{n}\times\mathrm{r}}$, all that remains to be done is note that eigenvalues of zero are related to $\mathcal{N}(M)$, which is evident from,

$$M\vec{\mathbf{v}} = \vec{\mathbf{0}} \cdot \vec{\mathbf{v}} = \vec{\mathbf{0}} \tag{27}$$

Therefore, by the Rank-Nullity Theorem, we know then that there can only be r nonzero eigenvalues and eigenvectors. If we remove the columns associated with zero-valued eigenvalues from matrix D, we then arrive at a matrix which is $r \times r$. Furthermore, we remove the corresponding eigenvectors (columns) from P. So, the size of A becomes $n \times r$ as expected.

3. (5 points). Download the Jupyter notebook *tennis rank.ipynb* and the two files *atp.csv* and *wta.csv*. These two files contain the outcome of all the tennis games on the professional circuit of the last two decades. Follow the instructions and questions on the notebook to find out who are the best players!

Please observe the attached PDF containing the Python file used for this problem.

In [2]:

```
%matplotlib inline
import matplotlib.pyplot as plot
import csv
import numpy as np
plot.rc('font',family='serif')
plot.rc('xtick',labelsize=14)
```

In [3]:

```
# The database contains the results of all tennis games
# in the pro men (ATP, from 2000 to end 2019) and women (WTA, from 2007 to end 2019) 't

# This codes reads the data
# Select the category: 'wta' for women, 'atp' for men
# For the questions of the homework, make sure to select the WTA dataset
# But you can use the ATP dataset to have fun !!
tour = 'wta'

# a setting to read the CSV files
if tour == 'atp':
    i_loser = 30
    i_winner = -2
else:
    i_loser = 21
    i_winner = -2

N=0                     # Total number of players (will be incremented when reading the fi
player_ID = dict()      # Given a 'name', player_ID[name] gives the ID of the player
player_name=[]          # Given an 'id', player_name[id] gives the name of the player

# This reads the CSV file to construct N, player_ID and player_name
with open(tour+'.txt') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    next(reader)
    for row in reader:
        loser = row[i_loser].rstrip()
        winner = row[i_winner].rstrip()

        for player in [winner,loser]:
            if not player in player_ID:
                player_ID[player]=N
                player_name.append(player)
                N +=1

# Matrix of the game records: R[i,j] will contain the number of time i beat j
R=np.zeros(shape=(N,N))

# This constructs R
with open(tour+'.txt') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    next(reader)

    for row in reader:
        # each row corresponds to a game
        loser = player_ID[row[i_loser].rstrip()]   # ID of the loser
        winner = player_ID[row[i_winner].rstrip()] # ID of the winner
        R[winner,loser] += 1  # count +1 victory for the winner
```

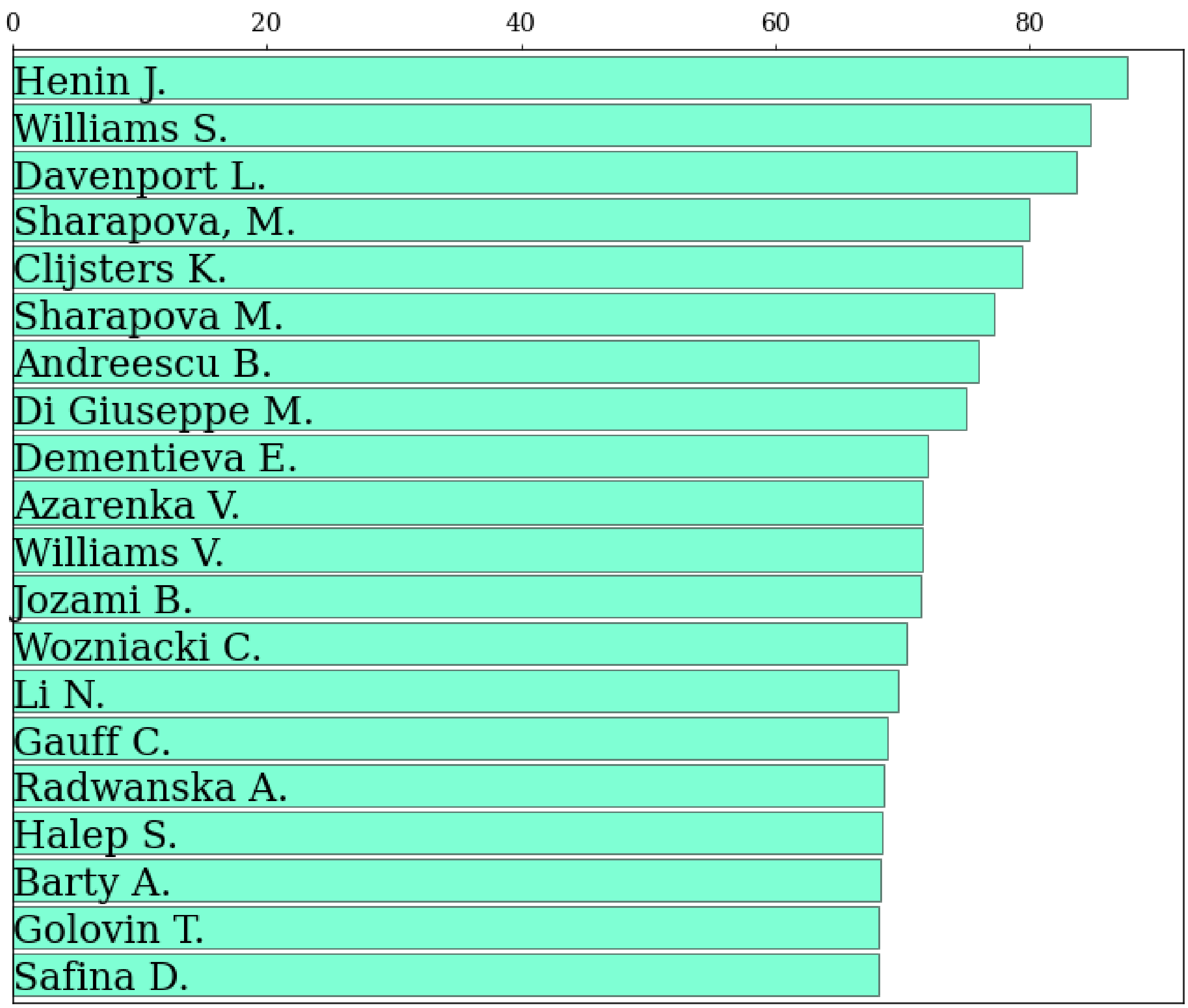
In [4]:

```
wins = np.sum(R,axis=1)    # total number of victories
losses = np.sum(R,axis=0)  # total number of losses
N_games = wins + losses    # total number of games
```

In [5]:
```
# naive ranking: rank players by percentage of victories
ratio = wins/N_games
naive_ranking = ratio.argsort()[::-1]
naive_scores = np.sort(ratio)[::-1]
```

In [6]:
```
# Function that plots rankings
def plot_ranking(ranking,scores,n):
    y=-np.array(range(n))
    plot.figure(figsize=(12,n/2),frameon=False)
    plot.barh(y,100*scores[:n],color='aquamarine', height=0.9, edgecolor = 'black',line
    for i in range(n):
        plot.text(0.0922,y[i]-0.35,player_name[ranking[i]],fontsize=22)
    t=plot.yticks([],[])
    l=plot.ylim(-n+ 0.4,0.6)
    ax = plot.gca()
    ax.xaxis.tick_top()
    #plot.savefig("ranking.pdf",bbox_inches='tight',transparent=True)
```

In [7]:
```
# Plot the 'naive' (ie in terms of percentage of victories) ranking of the top 20 playe
plot_ranking(naive_ranking,naive_scores,20)
```



(a) Compute the transition matrix P as in the notes, then construct the matrix

$$M = \alpha P + \frac{1-\alpha}{N} J$$

where J is the all-one matrix, and $\alpha = 0.99$.

In [8]:
```
P = np.zeros((N,N))

for i in range(N):
    for j in range(N):
        V_j = np.sum(R[j,:])
        G_j = np.sum(R[:,j] + R[j,:])
        if i == j:
            P[i,j] = V_j/G_j
        else:
            P[i,j] = R[i,j]/G_j
```

In [9]:
```
J = np.ones((N,N))
alpha = 0.99

M = alpha*P + (1-alpha)*J/N
```

(b) Compute the stationary distribution of the Markov chain of transition matrix M.

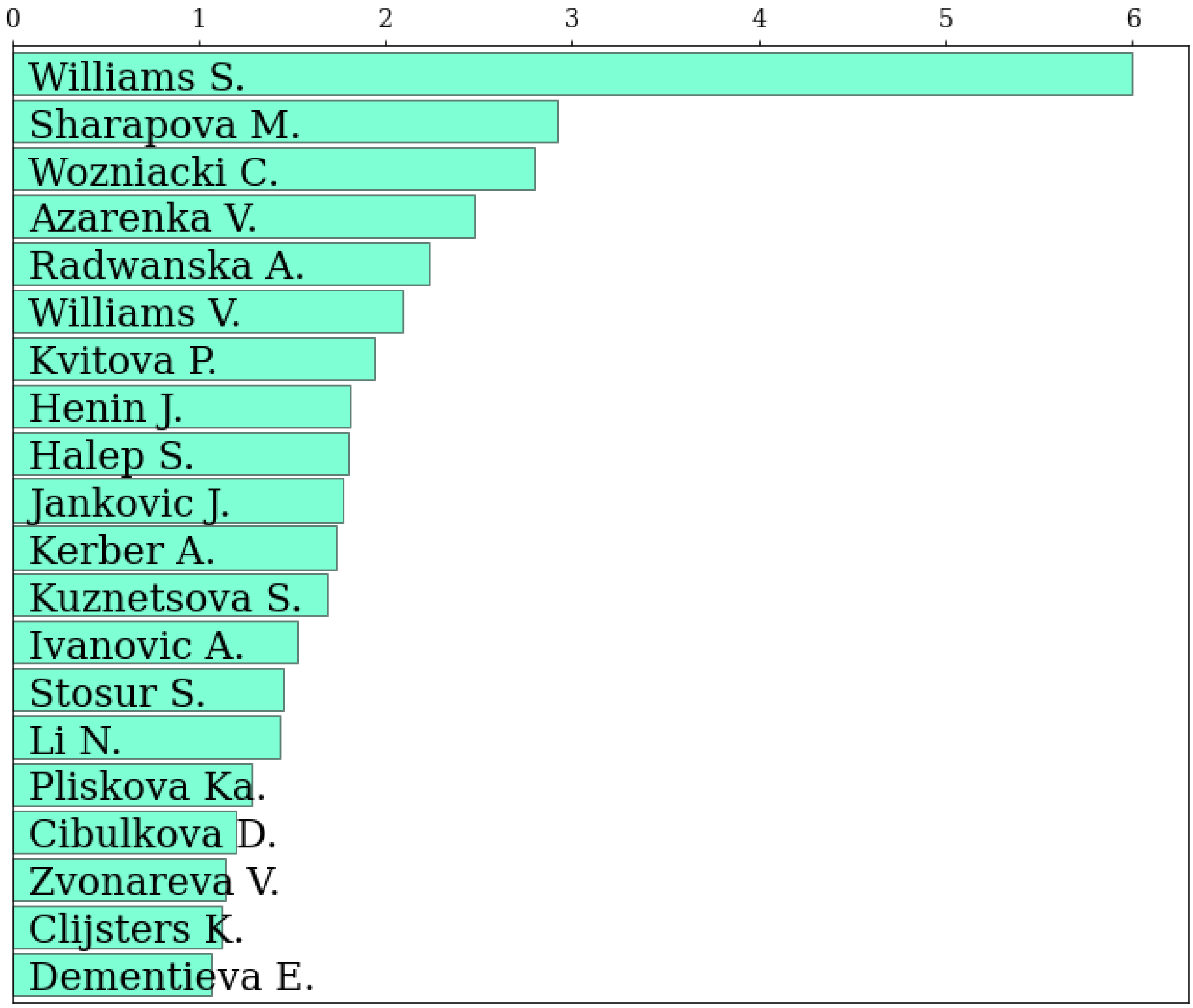
In [21]:
```
state = np.random.randint(100, size=N)
state = state/(sum(state))

while np.linalg.norm(state - np.matmul(M, state)) > 10**(-10):
    state = np.matmul(M, state)
```

(c) Use the stationary distribution to rank the players, and plot the ranking of the best 20 players.

In [11]:
```
sort_mu, players_ranked = zip(*sorted(zip(state, list(range(len(state))))))
sort_mu = list(sort_mu)[::-1]
players_ranked = list(players_ranked)[::-1]

plot_ranking(np.array(players_ranked),np.array(sort_mu),20)
```



(d) Open-ended question. For this question, no particular answer is awaited. Investigate the data (and maybe the wikipedia pages of the players, even though you do not need to know their careers by heart!), do some plots, other rankings, to find possible explanations to the following observations (for the women rankings):

- How would you explain that Henin, who is N1 in term of percentage of victories is way behind in page-rankings?
- Recompute the ranking, but now with $\alpha = 0.9$. How do you explain that Wozniacki is now ranked before Sharapova?

The most reasonable explanation as to why Henin dropped in ranking can be observed in the plot below, which simply looks at the raw number of games won by the players. Henin only managed to win 114 games, placing her outside of the top 100 in terms of raw games won. The naive ranking by percentage does not take into account the fact that players like Williams S. have won close to 500 games, which is obviously a greater achievement than winning only 114. With that being said, page-ranking takes into account both of these philosophies regarding ranking - the overall percentage record, as well as the pervasiveness of winning. Williams S. passes Henin even though she had a worse percentage of wins because she had a much larger impact on the league overall, winning far more games than Henin. The column of the transition matrix associated to Williams S. is clearly more populated, because she has played more games, presumably against a wider array of people, and has won more as well.

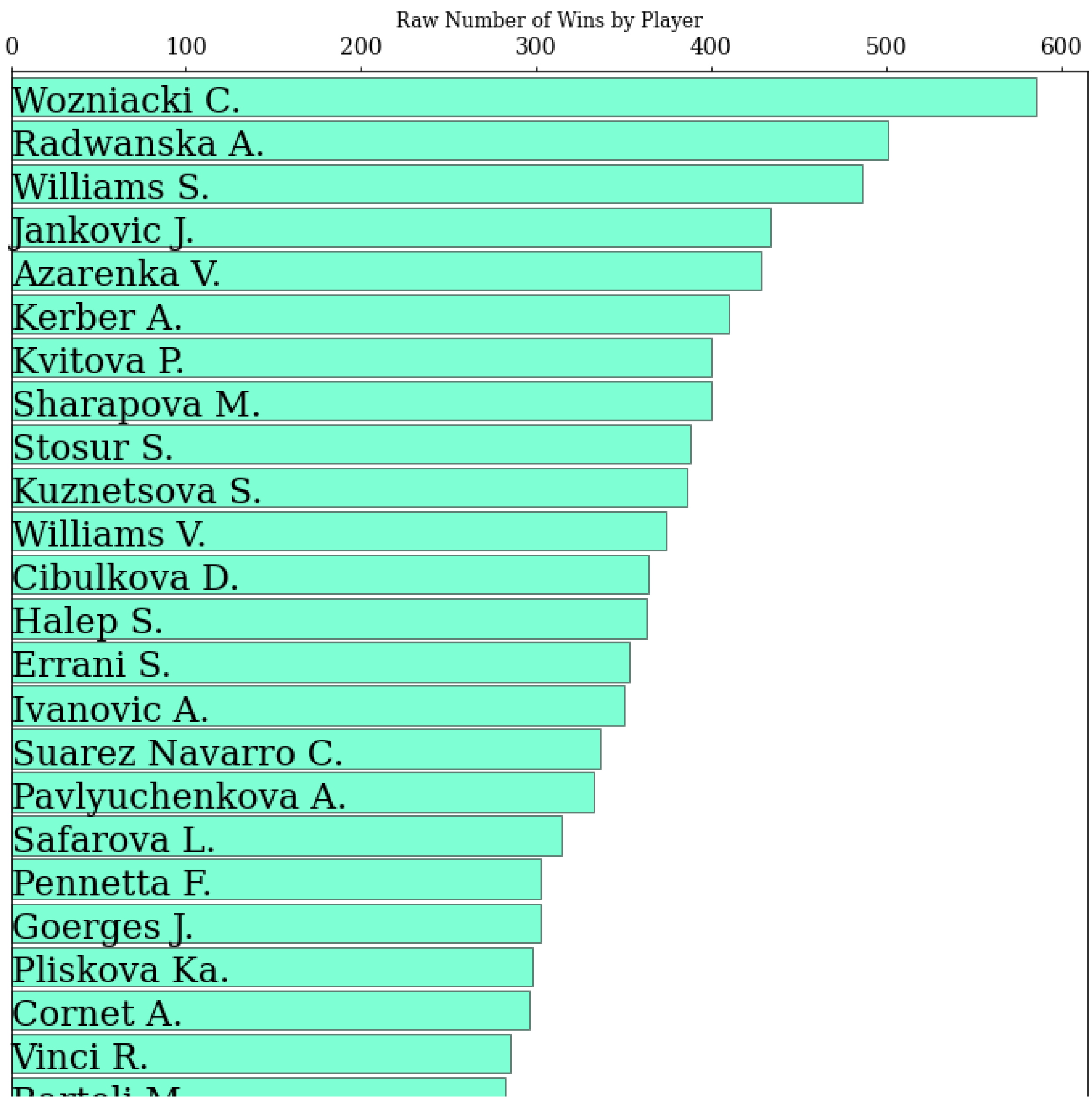
In [12]:

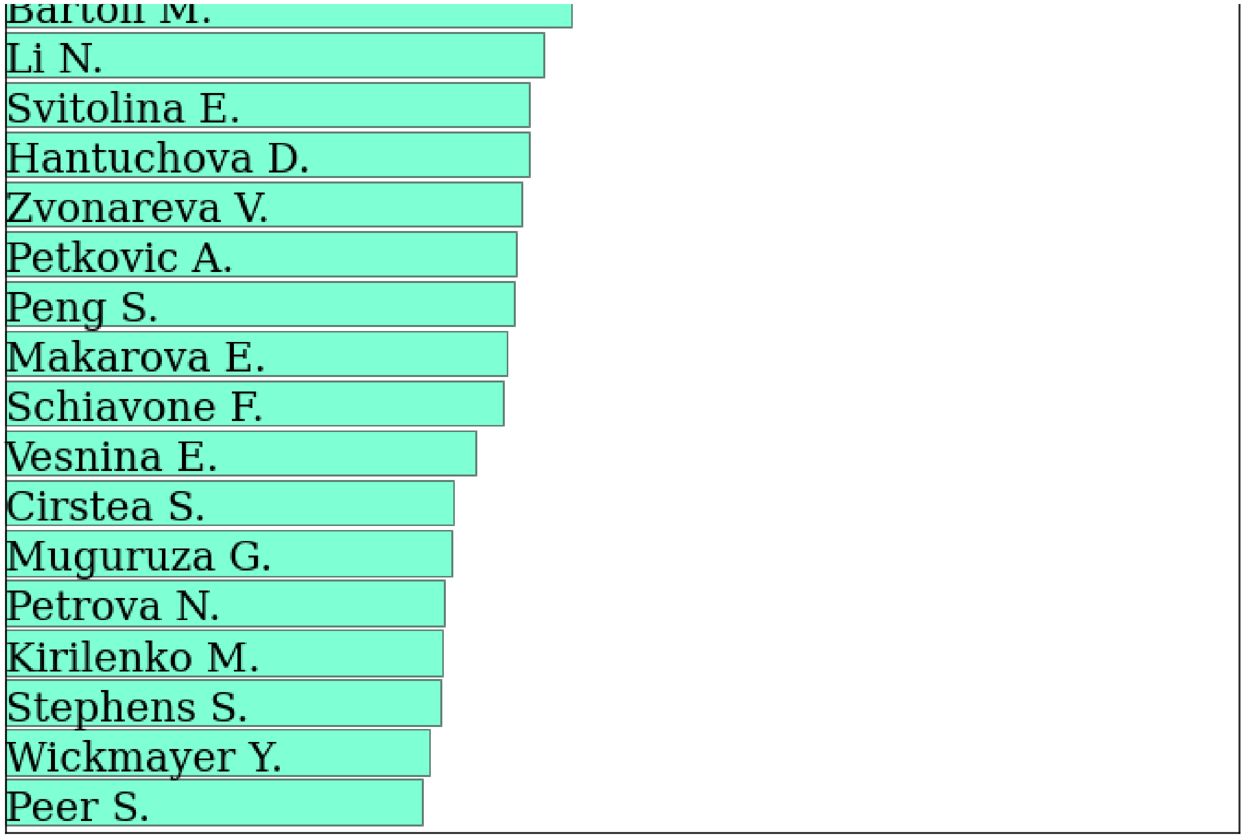
```
raw_ranking = wins.argsort()[::-1]
raw_scores = np.sort(wins)[::-1]
```

In [13]:

```
def plot_normal(ranking,scores,n):
    y=-np.array(range(n))
    plot.figure(figsize=(12,n/2),frameon=False)
    plot.barh(y,scores[:n],color='aquamarine', height=0.9, edgecolor = 'black',linewidt
    for i in range(n):
        plot.text(0.0922,y[i]-0.35,player_name[ranking[i]],fontsize=22)
    t=plot.yticks([],[])
    l=plot.ylim(-n+ 0.4,0.6)
    ax = plot.gca()
    ax.xaxis.tick_top()
    plot.title('Raw Number of Wins by Player')
    #plot.savefig("ranking.pdf",bbox_inches='tight',transparent=True)

plot_normal(raw_ranking,raw_scores,40)
```





In [14]:

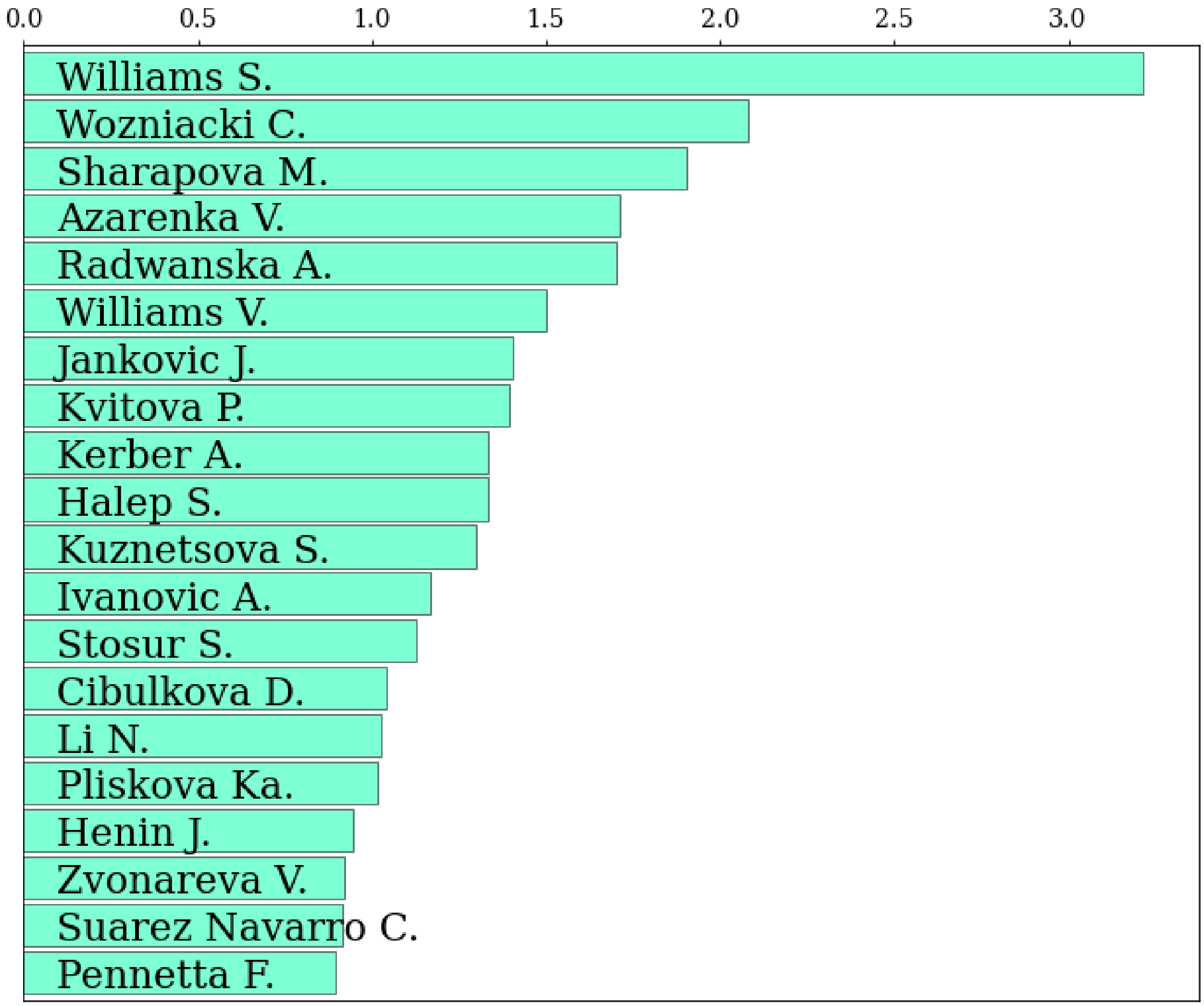
```
alpha = 0.9
M = alpha*P + (1-alpha)*J/N

state = np.random.randint(100, size=N)
state = state/(sum(state))

while np.linalg.norm(state - np.matmul(M, state)) > 10**(-10):
    state = np.matmul(M, state)

sort_mu, players_ranked = zip(*sorted(zip(state, list(range(len(state))))))
sort_mu = list(sort_mu)[::-1]
players_ranked = list(players_ranked)[::-1]

plot_ranking(np.array(players_ranked[:20]),np.array(sort_mu[:20]),20)
```



We could explain the fact that Wozniacki is now ahead of Sharapova because the parameter alpha describes an added amount of uniform randomness to the system, which allows the page-rank algorithm to escape from dead ends and loops within the system. Lowering the value of alpha corresponds to an increased level of random jumping, which perhaps draws the system further away from Sharapova, who is positioned within a dead end or a loop.

(e) Open-ended question. In fact, both CSV files contain the score (the number of sets won by each of the players) of each game. Propose a method based on PageRank, but with another transition matrix P, that takes the scores into account, in order to obtain more 'accurate' rankings and implement it. There is no particular method expected. Your are only suppose to propose something 'coherent' (for instance winning games by a large margin should improve rankings...)

Instead of composing the transition matrix P of fractions of games won (player-to-player), we can form P as the fraction of sets won (player-to-player). Doing so may yield insight into the more nuanced aspects of competition between two selected players.

In [15]:
```
# This code opens the game database
# it loops over all the games
# for each game it extracts the 'id' of the winner/loser
# and the number of sets won by each player

#Assume N is defined from before
```

```
R=np.zeros(shape=(N,N))

with open(tour+'.txt') as csvfile:
    reader = csv.reader(csvfile, delimiter=',')
    next(reader)

    for row in reader:
        # each row corresponds to a game
        loser = player_ID[row[i_loser].rstrip()]   # ID of the loser
        winner = player_ID[row[i_winner].rstrip()] # ID of the winner

        # check if the number of sets for each player is available
        if row[i_loser+1] != '' and row[i_winner+1] != '':
            loser_sets = int(float(row[i_loser+1]))
            winner_sets = int(float(row[i_winner+1]))
            if winner_sets == 0:
                # For some games (where one of the players retired because of injury...
                # The number of sets is 0. In that case we say that the winner won 2-0
                winner_sets = 2
                loser_sets = 0
        else:
            # if the number of sets are not available, we say that the winner won 2-0
            loser_sets = 0
            winner_sets = 2

        R[winner,loser] += winner_sets  # count +1 victory for the winner
```

In [16]:

```
P = np.zeros((N,N))

for i in range(N):
    for j in range(N):
        SV_j = np.sum(R[j,:])
        S_j = np.sum(R[:,j] + R[j,:])
        if i == j:
            P[i,j] = SV_j/S_j
        else:
            P[i,j] = R[i,j]/S_j
```

In [17]:

```
J = np.ones((N,N))
alpha = 0.99

M = alpha*P + (1-alpha)*J/N
```

In [18]:

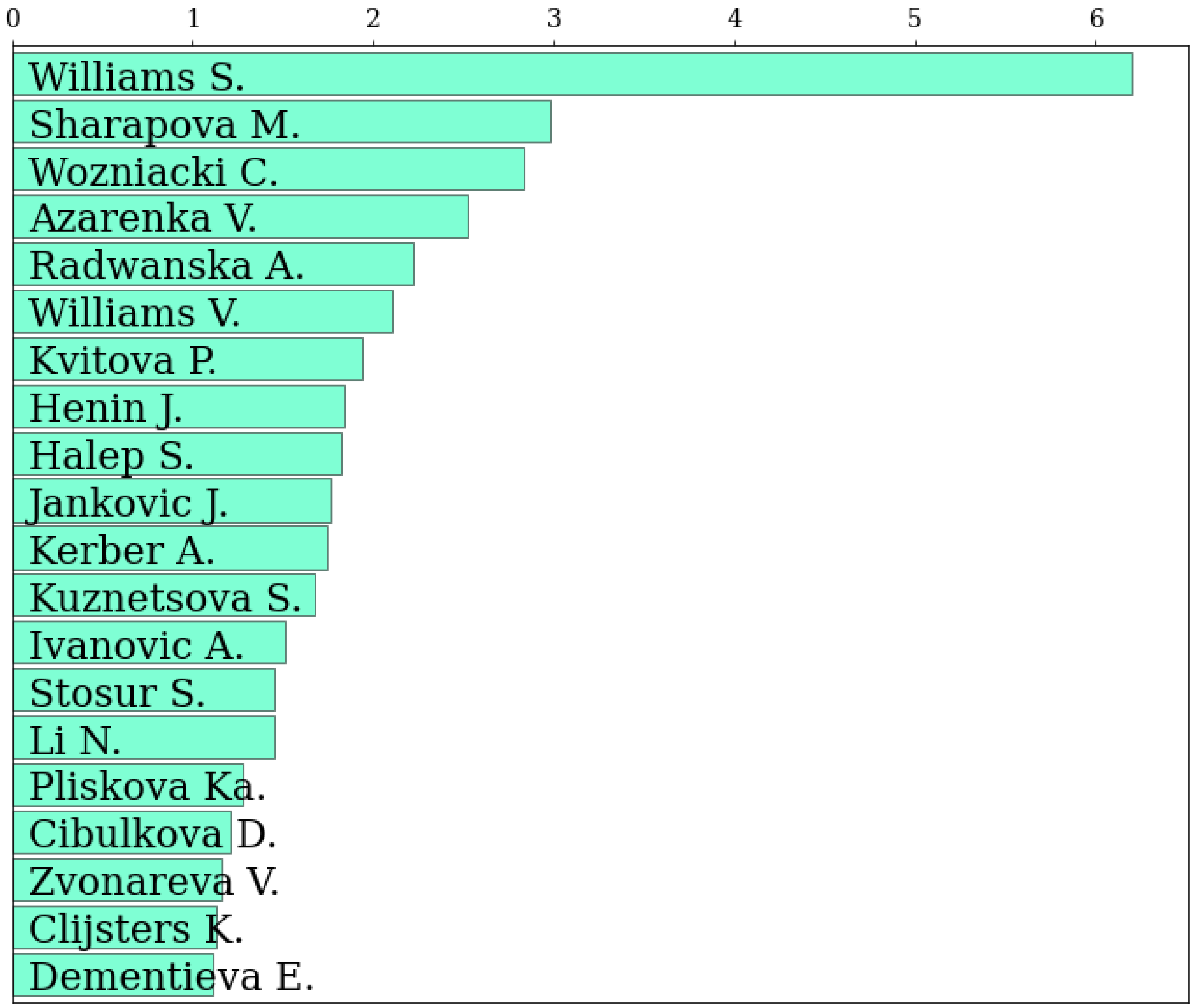
```
state = np.random.randint(100, size=N)
state = state/(sum(state))

while np.linalg.norm(state - np.matmul(M, state)) > 10**(-10):
    state = np.matmul(M, state)
```

In [19]:

```
sort_mu, players_ranked = zip(*sorted(zip(state, list(range(len(state))))))
sort_mu = list(sort_mu)[::-1]
players_ranked = list(players_ranked)[::-1]

plot_ranking(np.array(players_ranked),np.array(sort_mu),20)
```



It is interesting to note that this method barely changes the rankings, which is somewhat expected because the number of games won and the number of sets won are highly correlated.