Tutorial on the Least Squares Rating System

```
library(data.table)
```

This R notebook is an exercise and an illustration of the mathematical paper of Hal Stern at the journal Chance. His paper is entitled "Who's Number 1 in College Football? ... And How Might We Decide?"

Load and inspect the data

In this tutorial we are going to implement various algorithms related to the Least Squares Rating System. The data we need is as follows

- the schedule of season 1994.
- the ranking and ratings of the 107 Division 1-A teams.

The data with the match schedule of season 1994.

```
x = fread("./data/1994-season.csv")
x
```

```
##       Wk               Winner               Loser WS LS PD HA
##    1:  1             Nebraska       West Virginia 31  0 31  1
##    2:  1         Ohio State        Fresno State 34 10 24  1
##    3:  2              Arizona        Georgia Tech 19 14  5 -1
##    4:  2               Kansas             Houston 35 13 22 -1
##    5:  2 North Carolina State Bowling Green State 20 15  5  1
##   ---
## 632: 19        Florida State             Florida 23 17  6  0
## 633: 19           Penn State              Oregon 38 20 18  0
## 634: 19       South Carolina       West Virginia 24 21  3  0
## 635: 19  Southern California          Texas Tech 55 14 41  0
## 636: 19            Wisconsin                Duke 34 20 14  0
```

The ranking and ratings of the 107 Division 1-A teams

```
r = fread("./data/1994-rankings.csv")
r
```

```
##        Rk         School       Conf AP Rank  W  L T   OSRS   DSRS    SRS
##    1:   1     Penn State    Big Ten       2 12  0 0  20.15   5.87  26.02
##    2:   2        Florida SEC (East)       7 10  2 1  12.94   8.84  21.79
##    3:   3  Florida State        ACC       4 10  1 1  11.24   9.74  20.98
##    4:   4       Nebraska      Big 8       1 13  0 0   9.54  11.11  20.65
##    5:   5       Colorado      Big 8       3 11  1 0  13.05   5.35  18.40
##   ---
## 103: 103        Houston        SWC      NA  1 10 0 -11.34  -6.89 -18.23
## 104: 104 Arkansas State   Big West      NA  1 10 0 -15.89  -2.51 -18.40
## 105: 105     Kent State        MAC      NA  2  9 0 -17.94  -5.78 -23.72
## 106: 106           Ohio        MAC      NA  0 11 0 -23.51  -5.26 -28.78
## 107: 107          Akron        MAC      NA  1 10 0 -17.78 -13.39 -31.17
```

Let us separate the top teams from the lower-divisional teams.

```
teams = sort(unique(c(x$Winner, x$Loser)))
top_teams = sort(unique(r$School))
low_teams = setdiff(teams, top_teams)
```

Examples

Find the normal equations of the Least Squares Rating System with home advantage set to 3 points.

The solution is provided in the source file

```
source("lsq-functions.R")
```

```
l = lsq.system.ha(x, 3)
l$game[1:10, 1:10]
```

```
##       [,1] [,2] [,3] [,4] [,5] [,6] [,7] [,8] [,9] [,10]
##  [1,]   12    0    0    0    0    0    0    0    0    -1
##  [2,]    0   11    0    0    0    0    0    0    0     0
##  [3,]    0    0   13    0    0    0    0   -1    0     0
##  [4,]    0    0    0    1    0    0    0    0    0     0
##  [5,]    0    0    0    0    1    0    0    0    0     0
##  [6,]    0    0    0    0    0   12   -1    0    0     0
##  [7,]    0    0    0    0    0   -1   11    0    0     0
##  [8,]    0    0   -1    0    0    0    0   11    0     0
##  [9,]    0    0    0    0    0    0    0    0   11     0
## [10,]   -1    0    0    0    0    0    0    0    0    11
```

Solve the linear system and rank-order the teams according to rating

Define the coefficients and parameters

```
game = l$game; pd = l$pd; d = nrow(game)
```

Add a linear constraint to the system.

```
#all ratings sum to 0
game[d, ] = rep(1,d)
pd[d] = 0
#solve the linear system
sol = solve(game,pd)
```

Let us now rank-order the teams and display their ratings.

```
rorder = order(rank(sol[1:d]), decreasing = T)
rankings = data.table(team = teams[rorder], rating = round(sol[rorder],2))
rankings[1:10]
```

```
##                   team rating
##  1:          Penn State  39.00
##  2:             Florida  35.18
##  3:       Florida State  31.91
##  4:            Nebraska  30.40
##  5:            Colorado  27.09
##  6:          Miami (FL)  26.02
##  7:            Illinois  25.71
```

```
##  8:            Michigan  24.30
##  9:           Tennessee  24.25
## 10: Southern California  24.20
```

Write a function that implements all of the above steps

All these steps are implemented in `lsq.ratings.ha`.

```
lsq.ratings.ha(x, 3)[1:10]
```

```
##                    team rating
##  1:          Penn State  39.00
##  2:             Florida  35.18
##  3:       Florida State  31.91
##  4:            Nebraska  30.40
##  5:            Colorado  27.09
##  6:          Miami (FL)  26.02
##  7:            Illinois  25.71
##  8:            Michigan  24.30
##  9:           Tennessee  24.25
## 10: Southern California  24.20
```

Implement the Jackobi iteration method to solve a linear system and compare it with the built-in solver

We have already implemented the Jacobi iteration as an example, which we load from source file.

```
source("iter-methods.R")
```

We run the function `Jacobi`. It prints the number of iterations it makes in order to achieve the required tolerance level.

```
sol.ja = Jackobi(game, pd)
```

```
## [1] 99
```

In this case, the function made 99 iterations. Let us see the first five solutions

```
sol.ja[1:5]
```

```
## [1]  11.27988 -32.82019  19.97634 -61.39934 -14.06348
```

Let us compare them with the previous solution.

```
sol[1:5]
```

```
## [1]  11.27992 -32.82054  19.97645 -61.39932 -14.06338
```

Problems about the LSqRatings

Modify the above rating system so that the mean rating of all lower-divisional teams is zero

Modify the above rating system so that the rating of a particular team is zero

Compute the LSqRatings and the home advantage from the data

Compute the LSqRatings and the home advantage from the data with all lower-divisional teams having rating exactly 0

Decompose the ratings in offence and defence parts

Compute the LSqRatings and the home advantage from the data with given weights for each match

Problems about iterative numerical computation of linear systems

Implement the Jackobi algorithm and solve some of the above systems

Implement the Gauss-Seidel algorithm and solve some of the above systems