5 Pythagorean expectation and MLB

17 July 2020

Pythagorean expectation as predictor in Major League Baseball

One of the main reasons that people are interested in sports analytics is that they want to predict the outcome of events that have not yet occurred. Thus we want to go beyond "explanation" - finding the model that best fits the data (i.e. history) and to use our model to forecast the outcome of games in the future.

Pythagorean Expectation can be thought of as a forecast. At any point in the season, it can be calculated based on the games already played. Using it as a forecast would amount to saying that from that point onward the win percentage of the team would equal the Pythagorean Expectation to date.

In this notebook, we're going to see if it is a good forecasting model in the context of the MLB data we examined earlier. Specifically, we will take the Pythagorean expectation based on games already played up to the All-Star Game (which takes place roughly in the middle of the season) and then see how well it correlates with win percentage in the second half of the season. We also have a natural benchmark against which to evaluate this forecast. The simplest forecast of all is to assume that win percentage will stay the same. Hence we will compare Pythagorean Expectation before the All-Star Game to win percentage before the All-Star Game as forecast of win percentage in the second half of the season.

To implement this test we initially follow the same procedures as we used in the previous MLB notebook to measure team performance. But then we split the data at the All-Star Game and compare statistics for each half of the season.

```
# Import packages
library("readxl",quietly = TRUE)
library("tidyverse",quietly = TRUE)
```

```
# Read in the data
MLB <- read_excel("Retrosheet MLB game log 2018.xlsx")
```

```
## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
## sheet, : Expecting logical in CH2431 / R2431C86: got 'reybd901'
```

```
## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
## sheet, : Expecting logical in CJ2431 / R2431C88: got 'hamaa901'
```

```
## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
## sheet, : Expecting logical in CH2432 / R2432C86: got 'rackd901'
```

```
## Warning in read_fun(path = enc2native(normalizePath(path)), sheet_i =
## sheet, : Expecting logical in CJ2432 / R2432C88: got 'wolcq901'
```

```
names(MLB)
```

```
##   [1] "Date"                    "DoubleHeader"
##   [3] "DayOfWeek"               "VisitingTeam"
##   [5] "VisitingTeamLeague"      "VisitingTeamGameNumber"
##   [7] "HomeTeam"                "HomeTeamLeague"
##   [9] "HomeTeamGameNumber"      "VisitorRunsScored"
##  [11] "HomeRunsScore"           "LengthInOuts"
##  [13] "DayNight"                "CompletionInfo"
##  [15] "ForfeitInfo"             "ProtestInfo"
##  [17] "ParkID"                  "Attendence"
##  [19] "Duration"                "VisitorLineScore"
##  [21] "HomeLineScore"           "VisitorAB"
##  [23] "VisitorH"                "VisitorD"
##  [25] "VisitorT"                "VisitorHR"
##  [27] "VisitorRBI"              "VisitorSH"
##  [29] "VisitorSF"               "VisitorHBP"
##  [31] "VisitorBB"               "VisitorIBB"
##  [33] "VisitorK"                "VisitorSB"
##  [35] "VisitorCS"               "VisitorGDP"
##  [37] "VisitorCI"               "VisitorLOB"
##  [39] "VisitorPitchers"         "VisitorER"
##  [41] "VisitorTER"              "VisitorWP"
##  [43] "VisitorBalks"            "VisitorPO"
##  [45] "VisitorA"                "VisitorE"
##  [47] "VisitorPassed"           "VisitorDB"
##  [49] "VisitorTP"               "HomeAB"
##  [51] "HomeH"                   "HomeD"
##  [53] "HomeT"                   "HomeHR"
##  [55] "HomeRBI"                 "HomeSH"
##  [57] "HomeSF"                  "HomeHBP"
##  [59] "HomeBB"                  "HomeIBB"
##  [61] "HomeK"                   "HomeSB"
##  [63] "HomeCS"                  "HomeGDP"
##  [65] "HomeCI"                  "HomeLOB"
##  [67] "HomePitchers"            "HomeER"
##  [69] "HomeTER"                 "HomeWP"
##  [71] "HomeBalks"               "HomePO"
##  [73] "HomeA"                   "HomeE"
##  [75] "HomePassed"              "HomeDB"
##  [77] "HomeTP"                  "UmpireHID"
##  [79] "UmpireHName"             "Umpire1BID"
```

```
##  [81] "Umpire1BName"               "Umpire2BID"
##  [83] "Umpire2BName"               "Umpire3BID"
##  [85] "Umpire3BName"               "UmpireLFID"
##  [87] "UmpireLFName"               "UmpireRFID"
##  [89] "UmpireRFName"               "VisitorManagerID"
##  [91] "VisitorManagerName"         "HomeManagerID"
##  [93] "HomeManagerName"            "WinningPitcherID"
##  [95] "WinningPitcherName"         "LosingPitcherID"
##  [97] "LosingPitcherNAme"          "SavingPitcherID"
##  [99] "SavingPitcherName"          "GameWinningRBIID"
## [101] "GameWinningRBIName"         "VisitorStartingPitcherID"
## [103] "VisitorStartingPitcherName" "HomeStartingPitcherID"
## [105] "HomeStartingPitcherName"    "VisitorBatting1PlayerID"
## [107] "VisitorBatting1Name"        "VisitorBatting1Position"
## [109] "VisitorBatting2PlayerID"    "VisitorBatting2Name"
## [111] "VisitorBatting2Position"    "VisitorBatting3PlayerID"
## [113] "VisitorBatting3Name"        "VisitorBatting3Position"
## [115] "VisitorBatting4PlayerID"    "VisitorBatting4Name"
## [117] "VisitorBatting4Position"    "VisitorBatting5PlayerID"
## [119] "VisitorBatting5Name"        "VisitorBatting5Position"
## [121] "VisitorBatting6PlayerID"    "VisitorBatting6Name"
## [123] "VisitorBatting6Position"    "VisitorBatting7PlayerID"
## [125] "VisitorBatting7Name"        "VisitorBatting7Position"
## [127] "VisitorBatting8PlayerID"    "VisitorBatting8Name"
## [129] "VisitorBatting8Position"    "VisitorBatting9PlayerID"
## [131] "VisitorBatting9Name"        "VisitorBatting9Position"
## [133] "HomeBatting1PlayerID"       "HomeBatting1Name"
## [135] "HomeBatting1Position"       "HomeBatting2PlayerID"
## [137] "HomeBatting2Name"           "HomeBatting2Position"
## [139] "HomeBatting3PlayerID"       "HomeBatting3Name"
## [141] "HomeBatting3Position"       "HomeBatting4PlayerID"
## [143] "HomeBatting4Name"           "HomeBatting4Position"
## [145] "HomeBatting5PlayerID"       "HomeBatting5Name"
## [147] "HomeBatting5Position"       "HomeBatting6PlayerID"
## [149] "HomeBatting6Name"           "HomeBatting6Position"
## [151] "HomeBatting7PlayerID"       "HomeBatting7Name"
## [153] "HomeBatting7Position"       "HomeBatting8PlayerID"
## [155] "HomeBatting8Name"           "HomeBatting8Position"
## [157] "HomeBatting9PlayerID"       "HomeBatting9Name"
## [159] "HomeBatting9Position"       "AdditionalInfo"
## [161] "AcquisitionInfo"
```

```
# Create df containing only the variables we need
# Create a counter
```

```
MLB18 <- MLB %>% select(VisitingTeam,HomeTeam,VisitorRunsScored,HomeRunsScore,Date) %>%
                 rename(VisR = VisitorRunsScored, HomR = HomeRunsScore)%>%
                 mutate(count = 1)
head(MLB18)
```

```
## # A tibble: 6 x 6
##   VisitingTeam HomeTeam  VisR  HomR     Date count
##   <chr>        <chr>    <dbl> <dbl>    <dbl> <dbl>
## 1 COL          ARI          2     8 20180329     1
## 2 PHI          ATL          5     8 20180329     1
## 3 SFN          LAN          1     0 20180329     1
## 4 CHN          MIA          8     4 20180329     1
## 5 SLN          NYN          4     9 20180329     1
## 6 MIL          SDN          2     1 20180329     1
```

```
tail(MLB18)
```

```
## # A tibble: 6 x 6
##   VisitingTeam HomeTeam  VisR  HomR     Date count
##   <chr>        <chr>    <dbl> <dbl>    <dbl> <dbl>
## 1 CLE          KCA          2     1 20180930     1
## 2 CHA          MIN          4     5 20180930     1
## 3 TEX          SEA          1     3 20180930     1
## 4 TOR          TBA          4     9 20180930     1
## 5 MIL          CHN          3     1 20181001     1
## 6 COL          LAN          2     5 20181001     1
```

```
# Create df recording team performance as home team
# We create an additional column 'home' which here
# has a value 1 to designate that these were home team games
MLBhome  <- MLB18 %>% select(HomeTeam,HomR,VisR,count,Date) %>%
                 rename(team = HomeTeam, RA = VisR , R = HomR)%>%
                 mutate(home = 1)
```

```
# Create df recording team performance as visiting team
# As above, we create an additional column 'home',
# which now has a value 0 to designate that these were away team games
MLBaway  <- MLB18 %>% select(VisitingTeam,VisR,HomR,count,Date) %>%
                 rename(team = VisitingTeam, R = VisR , RA = HomR)%>%
                 mutate(home = 0)
```

```
# Here is where the approach differs from the previous notebooks.
# Instead of taking sums and averages, we first
# concatenate, meaning that we stack performances as home team and away team.
# This creates a list of games played
# by each team across the season. The list is 4,862 rows long, which is twice the numb
# season.
```

```r
MLB18 = rbind(MLBhome,MLBaway)
head(MLB18)
```

```
## # A tibble: 6 x 6
##   team      R    RA count     Date  home
##   <chr> <dbl> <dbl> <dbl>    <dbl> <dbl>
## 1 ARI       8     2     1 20180329     1
## 2 ATL       8     5     1 20180329     1
## 3 LAN       0     1     1 20180329     1
## 4 MIA       4     8     1 20180329     1
## 5 NYN       9     4     1 20180329     1
## 6 SDN       1     2     1 20180329     1
```

```r
tail(MLB18)
```

```
## # A tibble: 6 x 6
##   team      R    RA count     Date  home
##   <chr> <dbl> <dbl> <dbl>    <dbl> <dbl>
## 1 CLE       2     1     1 20180930     0
## 2 CHA       4     5     1 20180930     0
## 3 TEX       1     3     1 20180930     0
## 4 TOR       4     9     1 20180930     0
## 5 MIL       3     1     1 20181001     0
## 6 COL       2     5     1 20181001     0
```

```r
# We define a win

MLB18[,'win'] = ifelse(MLB18$R > MLB18$RA,1,0)
head(MLB18)
```

```
## # A tibble: 6 x 7
##   team      R    RA count     Date  home   win
##   <chr> <dbl> <dbl> <dbl>    <dbl> <dbl> <dbl>
## 1 ARI       8     2     1 20180329     1     1
## 2 ATL       8     5     1 20180329     1     1
## 3 LAN       0     1     1 20180329     1     0
## 4 MIA       4     8     1 20180329     1     0
## 5 NYN       9     4     1 20180329     1     1
## 6 SDN       1     2     1 20180329     1     0
```

```r
tail(MLB18)
```

```
## # A tibble: 6 x 7
##   team      R    RA count     Date  home   win
##   <chr> <dbl> <dbl> <dbl>    <dbl> <dbl> <dbl>
## 1 CLE       2     1     1 20180930     0     1
## 2 CHA       4     5     1 20180930     0     0
```

```
## 3 TEX       1     3     1 20180930     0     0
## 4 TOR       4     9     1 20180930     0     0
## 5 MIL       3     1     1 20181001     0     1
## 6 COL       2     5     1 20181001     0     0
```

```
# Now we define the season up to the All-Star Game (which was on July 17, 2018)
# as the first half of the season
# We use .describe() to show the summary statistics. You can see this includes
# 2,886 rows and therefore the results of 1,443 games.

Half1 <- MLB18 %>% filter(Date < 20180717)
Half1 %>% summary()
```

```
##      team                R                RA             count
##  Length:2886        Min.   : 0.000   Min.   : 0.000   Min.   :1
##  Class :character   1st Qu.: 2.000   1st Qu.: 2.000   1st Qu.:1
##  Mode  :character   Median : 4.000   Median : 4.000   Median :1
##                     Mean   : 4.418   Mean   : 4.418   Mean   :1
##                     3rd Qu.: 6.000   3rd Qu.: 6.000   3rd Qu.:1
##                     Max.   :20.000   Max.   :20.000   Max.   :1
##       Date                home             win
##  Min.   :20180329   Min.   :0.0   Min.   :0.0
##  1st Qu.:20180426   1st Qu.:0.0   1st Qu.:0.0
##  Median :20180523   Median :0.5   Median :0.5
##  Mean   :20180540   Mean   :0.5   Mean   :0.5
##  3rd Qu.:20180619   3rd Qu.:1.0   3rd Qu.:1.0
##  Max.   :20180715   Max.   :1.0   Max.   :1.0
```

```
# Now we define the season after the All-Star Game
# (which was on July 17, 2018) as the second half of the season
# We use .describe() to show the summary statistics.
# You can see this includes 1,976 rows and therefore the results of
# 988 games.


Half2 = MLB18%>% filter(Date > 20180717)
Half2 %>% summary()
```

```
##      team                R                RA             count
##  Length:1976        Min.   : 0.000   Min.   : 0.000   Min.   :1
##  Class :character   1st Qu.: 2.000   1st Qu.: 2.000   1st Qu.:1
##  Mode  :character   Median : 4.000   Median : 4.000   Median :1
##                     Mean   : 4.494   Mean   : 4.494   Mean   :1
##                     3rd Qu.: 6.000   3rd Qu.: 6.000   3rd Qu.:1
##                     Max.   :25.000   Max.   :25.000   Max.   :1
##       Date                home             win
```

```
##  Min.   :20180719   Min.   :0.0   Min.   :0.0
##  1st Qu.:20180807   1st Qu.:0.0   1st Qu.:0.0
##  Median :20180825   Median :0.5   Median :0.5
##  Mean   :20180842   Mean   :0.5   Mean   :0.5
##  3rd Qu.:20180912   3rd Qu.:1.0   3rd Qu.:1.0
##  Max.   :20181001   Max.   :1.0   Max.   :1.0
```

```
# We now use .groupby to sum the number of games, wins, runs
# and runs against for the first half of the season.

Half1perf  <-  Half1 %>% group_by(team)%>%
            dplyr::summarise(count = sum(count),
                             win= sum(win),
                              R = sum(R),
                             RA = sum(RA)
                            )%>%
                            ungroup()%>%
                            rename(count1 = count ,
                                   win1= win,
                               R1 = R,
                              RA1 = RA)
head(Half1perf)
```

```
## # A tibble: 6 x 5
##   team  count1  win1    R1   RA1
##   <chr>  <dbl> <dbl> <dbl> <dbl>
## 1 ANA       97    49   425   401
## 2 ARI       97    53   421   378
## 3 ATL       94    52   456   388
## 4 BAL       97    28   345   504
## 5 BOS       98    68   530   367
## 6 CHA       95    33   384   510
```

```
tail(Half1perf)
```

```
## # A tibble: 6 x 5
##   team  count1  win1    R1   RA1
##   <chr>  <dbl> <dbl> <dbl> <dbl>
## 1 SFN       98    50   399   425
## 2 SLN       94    48   413   402
## 3 TBA       96    49   401   381
## 4 TEX       97    41   424   487
## 5 TOR       95    43   423   467
## 6 WAS       96    48   417   387
```

```
# From these statistics we calculate win percentage
# and Pythagorean Expectation for the first half of the season.
```

```
Half1perf['wpc1'] = Half1perf['win1']/Half1perf['count1']
Half1perf['pyth1'] = Half1perf['R1']**2/(Half1perf['R1']**2 + Half1perf['RA1']**2)
head(Half1perf)
```

```
## # A tibble: 6 x 7
##   team  count1  win1    R1   RA1  wpc1 pyth1
##   <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 ANA       97    49   425   401 0.505 0.529
## 2 ARI       97    53   421   378 0.546 0.554
## 3 ATL       94    52   456   388 0.553 0.580
## 4 BAL       97    28   345   504 0.289 0.319
## 5 BOS       98    68   530   367 0.694 0.676
## 6 CHA       95    33   384   510 0.347 0.362
```

```
tail(Half1perf)
```

```
## # A tibble: 6 x 7
##   team  count1  win1    R1   RA1  wpc1 pyth1
##   <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 SFN       98    50   399   425 0.510 0.468
## 2 SLN       94    48   413   402 0.511 0.513
## 3 TBA       96    49   401   381 0.510 0.526
## 4 TEX       97    41   424   487 0.423 0.431
## 5 TOR       95    43   423   467 0.453 0.451
## 6 WAS       96    48   417   387 0.5   0.537
```

```
# As above we use .groupby to sum the number of games, wins,
# runs and runs against for the second half of the season.



Half2perf  <-  Half2 %>% group_by(team)%>%
            dplyr::summarise(count = sum(count),
                             win= sum(win),
                              R = sum(R),
                             RA = sum(RA)
                            )%>%
                            ungroup()%>%
                            rename(count2 = count ,
                                   win2 = win,
                                R2 = R,
                                RA2 = RA)

head(Half2perf)
```

```
## # A tibble: 6 x 5
##   team  count2  win2    R2   RA2
##   <chr>  <dbl> <dbl> <dbl> <dbl>
## 1 ANA       65    31   296   321
## 2 ARI       65    29   272   266
## 3 ATL       68    38   303   269
## 4 BAL       65    19   277   388
## 5 BOS       64    40   346   280
## 6 CHA       67    29   272   338
```

```
tail(Half2perf)
```

```
## # A tibble: 6 x 5
##   team  count2  win2    R2   RA2
##   <chr>  <dbl> <dbl> <dbl> <dbl>
## 1 SFN       64    23   204   274
## 2 SLN       68    40   346   289
## 3 TBA       66    41   315   265
## 4 TEX       65    26   313   361
## 5 TOR       67    30   286   365
## 6 WAS       66    34   354   295
```

```
# From these statistics we calculate win percentage
# and Pythagorean Expectation for the second half of the season.

Half2perf[,'wpc2'] = Half2perf[,'win2']/Half2perf[,'count2']
Half2perf[,'pyth2'] = Half2perf[,'R2']**2/(Half2perf[,'R2']**2 + Half2perf[,'RA2']**2)
Half2perf
```

```
## # A tibble: 30 x 7
##    team  count2  win2    R2   RA2  wpc2 pyth2
##    <chr>  <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
##  1 ANA       65    31   296   321 0.477 0.460
##  2 ARI       65    29   272   266 0.446 0.511
##  3 ATL       68    38   303   269 0.559 0.559
##  4 BAL       65    19   277   388 0.292 0.338
##  5 BOS       64    40   346   280 0.625 0.604
##  6 CHA       67    29   272   338 0.433 0.393
##  7 CHN       70    40   285   283 0.571 0.504
##  8 CIN       66    24   235   326 0.364 0.342
##  9 CLE       67    39   331   243 0.582 0.650
## 10 COL       67    40   313   277 0.597 0.561
## # ... with 20 more rows
```

```
# Now we merge the two dfs
Half2predictor<- merge(x = Half1perf,y= Half2perf,by=c('team'))
head(Half2predictor)
```

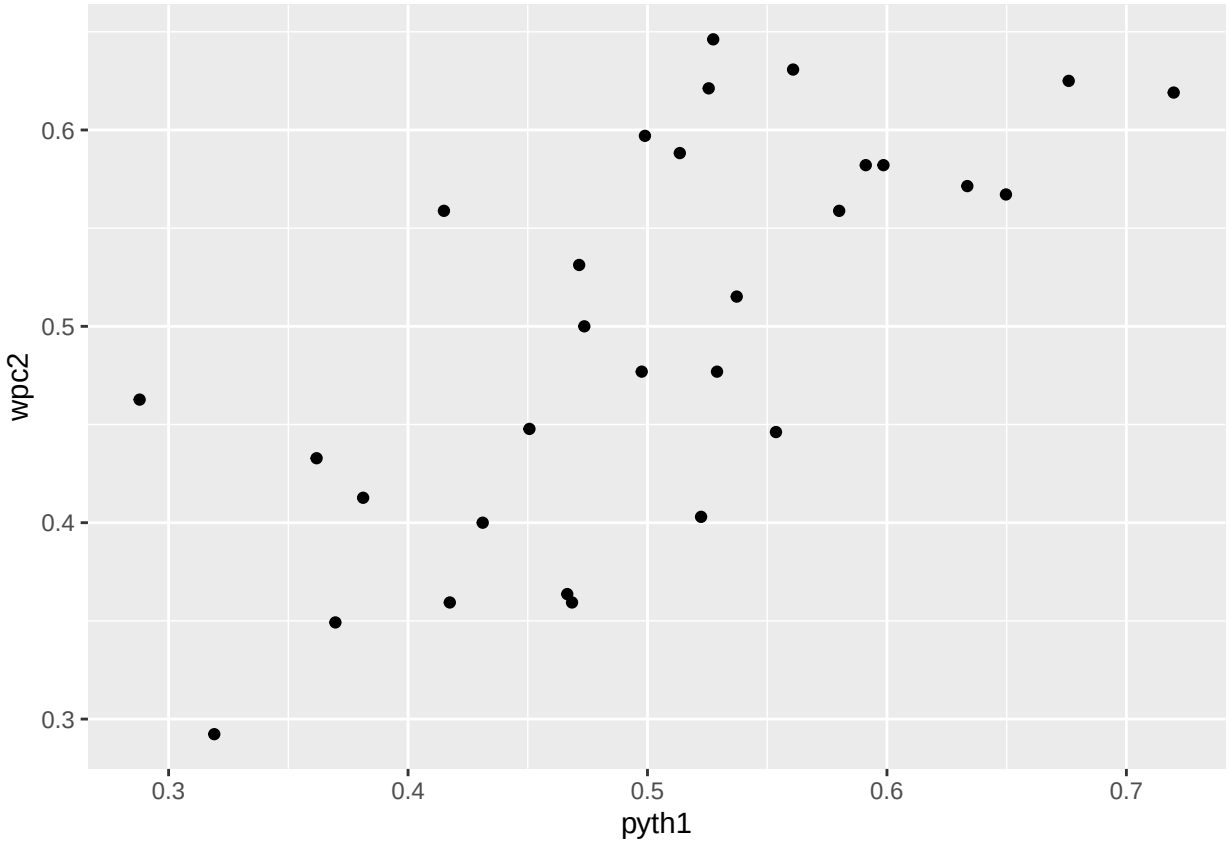
```
##   team count1 win1  R1 RA1      wpc1     pyth1 count2 win2  R2 RA2
## 1  ANA     97   49 425 401 0.5051546 0.5290312     65   31 296 321
## 2  ARI     97   53 421 378 0.5463918 0.5536619     65   29 272 266
## 3  ATL     94   52 456 388 0.5531915 0.5800491     68   38 303 269
## 4  BAL     97   28 345 504 0.2886598 0.3190668     65   19 277 388
## 5  BOS     98   68 530 367 0.6938776 0.6759082     64   40 346 280
## 6  CHA     95   33 384 510 0.3473684 0.3618055     67   29 272 338
##        wpc2     pyth2
## 1 0.4769231 0.4595478
## 2 0.4461538 0.5111510
## 3 0.5588235 0.5592313
## 4 0.2923077 0.3376072
## 5 0.6250000 0.6042722
## 6 0.4328358 0.3930552
```

```
tail(Half2predictor)
```

```
##    team count1 win1  R1 RA1      wpc1     pyth1 count2 win2  R2 RA2
## 25  SFN     98   50 399 425 0.5102041 0.4684780     64   23 204 274
## 26  SLN     94   48 413 402 0.5106383 0.5134945     68   40 346 289
## 27  TBA     96   49 401 381 0.5104167 0.5255587     66   41 315 265
## 28  TEX     97   41 424 487 0.4226804 0.4311744     65   26 313 361
## 29  TOR     95   43 423 467 0.4526316 0.4506823     67   30 286 365
## 30  WAS     96   48 417 387 0.5000000 0.5372616     66   34 354 295
##         wpc2     pyth2
## 25 0.3593750 0.3566311
## 26 0.5882353 0.5890463
## 27 0.6212121 0.5855710
## 28 0.4000000 0.4291428
## 29 0.4477612 0.3804094
## 30 0.5151515 0.5901639
```

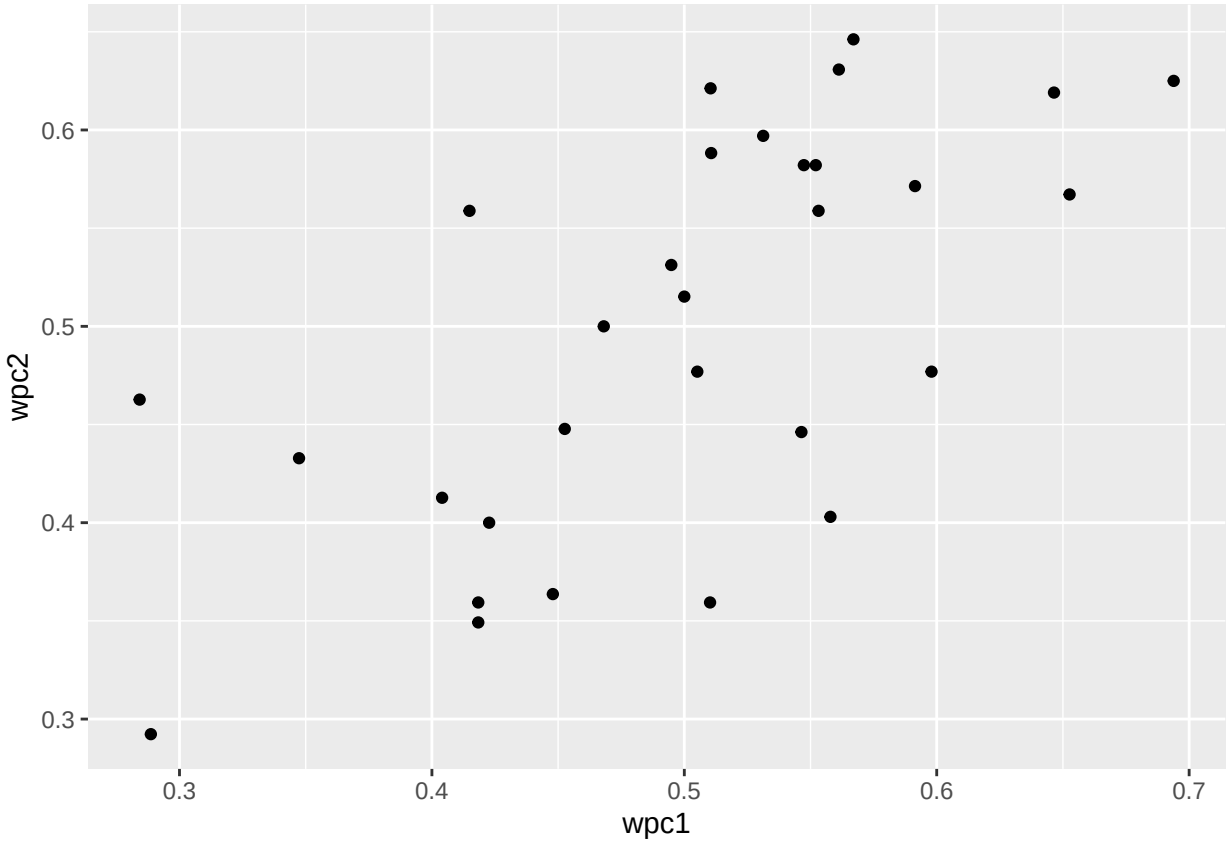
```
# First, plot Pythagorean Expectation against win percentage
# in the second half of the season

ggplot(data = Half2predictor,aes(x = pyth1,y = wpc2 )) + geom_point()
```




```
# Now, compare this with a plot of win percentage from
# the first half of the season against win percentage
# in the second half of the season

ggplot(data = Half2predictor,aes(x = wpc1,y = wpc2 )) + geom_point()
```




```
# The two plots look similar
# We can be more precise still if we compare the correlation coefficients.
# The first row of the table shows the
# correlation of win percentage in second half of the season against itself,
# win percentage in the first half of the season,
# Pythagorean Expectation in the first half of the season,
# and Pythagorean Expectation in the second half of the season.
# Our focus is on comparing the second and third columns.

keyvars <- Half2predictor[,c('team','wpc2','wpc1','pyth1','pyth2')]
str(keyvars)
```

```
## 'data.frame':    30 obs. of  5 variables:
##  $ team : chr  "ANA" "ARI" "ATL" "BAL" ...
##  $ wpc2 : num  0.477 0.446 0.559 0.292 0.625 ...
##  $ wpc1 : num  0.505 0.546 0.553 0.289 0.694 ...
##  $ pyth1: num  0.529 0.554 0.58 0.319 0.676 ...
##  $ pyth2: num  0.46 0.511 0.559 0.338 0.604 ...
```

```
keyvars[,-1] %>% cor()
```

```
##            wpc2      wpc1     pyth1     pyth2
```

```
## wpc2  1.0000000 0.6525491 0.6907521 0.9244726
## wpc1  0.6525491 1.0000000 0.9410817 0.5778470
## pyth1 0.6907521 0.9410817 1.0000000 0.6595208
## pyth2 0.9244726 0.5778470 0.6595208 1.0000000
```

```
# We can also sort the variables to show for each club
# how close the relationships are between the first and second half
# of the season.

keyvars<- keyvars%>% arrange(desc(wpc2))
keyvars
```

```
##    team      wpc2      wpc1     pyth1     pyth2
## 1   OAK 0.6461538 0.5670103 0.5274393 0.6812239
## 2   MIL 0.6307692 0.5612245 0.5607948 0.5749834
## 3   BOS 0.6250000 0.6938776 0.6759082 0.6042722
## 4   TBA 0.6212121 0.5104167 0.5255587 0.5855710
## 5   HOU 0.6190476 0.6464646 0.7197476 0.6415527
## 6   COL 0.5970149 0.5312500 0.4989305 0.5607906
## 7   SLN 0.5882353 0.5106383 0.5134945 0.5890463
## 8   CLE 0.5820896 0.5473684 0.5911579 0.6497895
## 9   LAN 0.5820896 0.5520833 0.5985387 0.6832278
## 10  CHN 0.5714286 0.5913978 0.6335664 0.5035211
## 11  NYA 0.5671642 0.6526316 0.6497021 0.5762433
## 12  ATL 0.5588235 0.5531915 0.5800491 0.5592313
## 13  NYN 0.5588235 0.4148936 0.4149813 0.5625714
## 14  PIT 0.5312500 0.4948454 0.4714519 0.5469548
## 15  WAS 0.5151515 0.5000000 0.5372616 0.5901639
## 16  MIN 0.5000000 0.4680851 0.4736120 0.4782035
## 17  ANA 0.4769231 0.5051546 0.5290312 0.4595478
## 18  SEA 0.4769231 0.5979381 0.4975787 0.4432445
## 19  KCA 0.4626866 0.2842105 0.2879035 0.4966888
## 20  TOR 0.4477612 0.4526316 0.4506823 0.3804094
## 21  ARI 0.4461538 0.5463918 0.5536619 0.5111510
## 22  CHA 0.4328358 0.3473684 0.3618055 0.3930552
## 23  SDN 0.4126984 0.4040404 0.3812420 0.4104765
## 24  PHI 0.4029851 0.5578947 0.5223768 0.3866850
## 25  TEX 0.4000000 0.4226804 0.4311744 0.4291428
## 26  CIN 0.3636364 0.4479167 0.4664947 0.3419483
## 27  DET 0.3593750 0.4183673 0.4174353 0.3399415
## 28  SFN 0.3593750 0.5102041 0.4684780 0.3566311
## 29  MIA 0.3492063 0.4183673 0.3696520 0.3097008
## 30  BAL 0.2923077 0.2886598 0.3190668 0.3376072
```

Conclusion

We can see from the correlation matrix that win percentage in the second half of the season is correlated with win percentage in the first half of the season - the correlation coefficient is +0.653. It's not surprising that performance in the first half of the season is to an extent predictive of performance in the second half. But there are also clearly things that can change.

When we sort the teams from highest to lowest send half of season win percentage, we find a mixed picture. Some clubs perform with less than one percentage point difference in each half, e.g. The Brave (ATL), the Padres (SDN) or the Orioles (BAL), while others differed by more than ten percentage points, e.g. the Rays (TBA), the Mets (NYN) or the Mariners (SEA).

We could simply use first half win percentage as a predictor of second half win percentage, but when we look at the correlation matrix we can see that the Pythagorean Expectation is an even better forecast - the correlation coefficient is higher, at +0.691. To be sure, the difference is not large, but it is slightly better. This was, in fact, the initial impetus for Bill James when introducing the statistic. He argued that a win could ride on lucky hit and the difference of just one run, which made wins a less reliable predictor than the aggregate capacity to produce runs and limit conceding runs. As in many aspects of baseball analysis, our data show that James was quite right.