The relation between runs and wins

Daniel J. Eck

Background

This lecture is meant to supplement Chapter 4 in your textbook.

We already performed an analysis on the relation of runs and components of baseball.

Now we look at the relation between runs and wins.

Example: the relation beween runs and wins

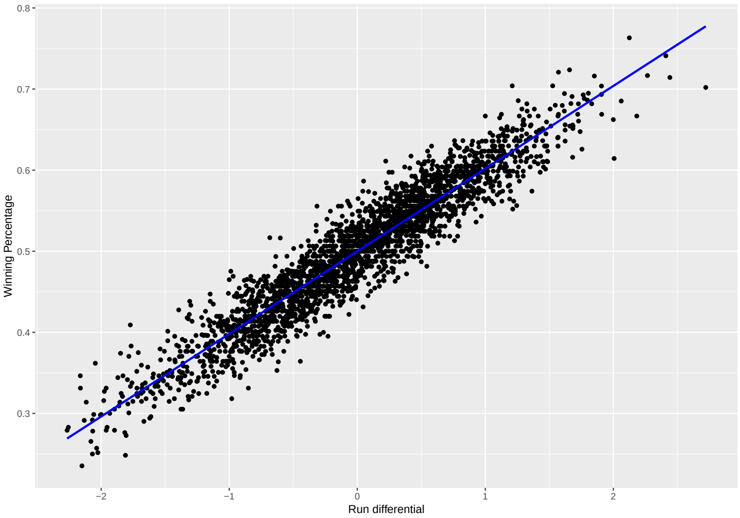
We will consider the relationship between runs and wins since 1900.

```r
library(tidyverse)
library(Lahman)
dat = Teams %>%
    filter(yearID >= 1900) %>%
    select(teamID, yearID, lgID, W, L, G, R, RA) %>%
    mutate(RD = (R - RA)/(W + L), Wpct = W / (W + L))
head(dat, 5)
```

```
##   teamID yearID lgID  W  L   G   R  RA         RD      Wpct
## 1    BRO   1900   NL 82 54 141 816 722  0.6911765 0.6029412
## 2    BSN   1900   NL 66 72 142 778 739  0.2826087 0.4782609
## 3    CHN   1900   NL 65 75 146 635 751 -0.8285714 0.4642857
## 4    CIN   1900   NL 62 77 144 703 745 -0.3021583 0.4460432
## 5    NY1   1900   NL 60 78 141 713 823 -0.7971014 0.4347826
```

Here the scaling by outcomes is for interpretability, it will not affect inference.

```r
ggplot(dat, aes(x = RD, y = Wpct)) +  geom_point() +
    scale_x_continuous("Run differential") +
    scale_y_continuous("Winning Percentage") +
        geom_smooth(method = "lm", se = FALSE, color = "blue")
```



Winning percentage is well explained by run differential

```r
m = lm(Wpct ~ RD, data = dat)
summary(m)
```

```
##
## Call:
## lm(formula = Wpct ~ RD, data = dat)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.090509 -0.018436  0.000292  0.017966  0.089745
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.4999985  0.0005184   964.6   <2e-16 ***
## RD          0.1019067  0.0006801   149.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02663 on 2638 degrees of freedom
## Multiple R-squared:  0.8948, Adjusted R-squared:  0.8948
## F-statistic: 2.245e+04 on 1 and 2638 DF,  p-value: < 2.2e-16
```

The intercept is basically equal to 0.5. This isn't surprising because the line of best fit includes the point $(\bar{x}, \bar{y})$.

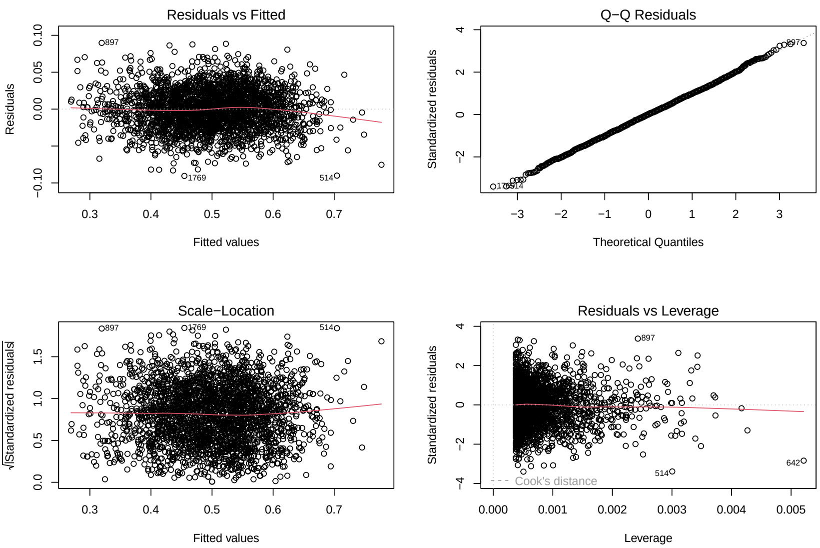
Now, in one season, we have

$$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} \frac{\mathrm{R}_i - \mathrm{RA}_i}{W + L} = 0$$

and, with each team having the same number of decisions, we have

$$\bar{y} = \frac{1}{n}\sum_{i=1}^{n} \mathrm{WP}_i = \frac{1}{n}\sum_{i=1}^{n} \frac{W_i}{W_i + L_i} = \frac{\bar{W}}{G} = 0.5.$$

```r
par(mfrow = c(2,2))
plot(m)
```



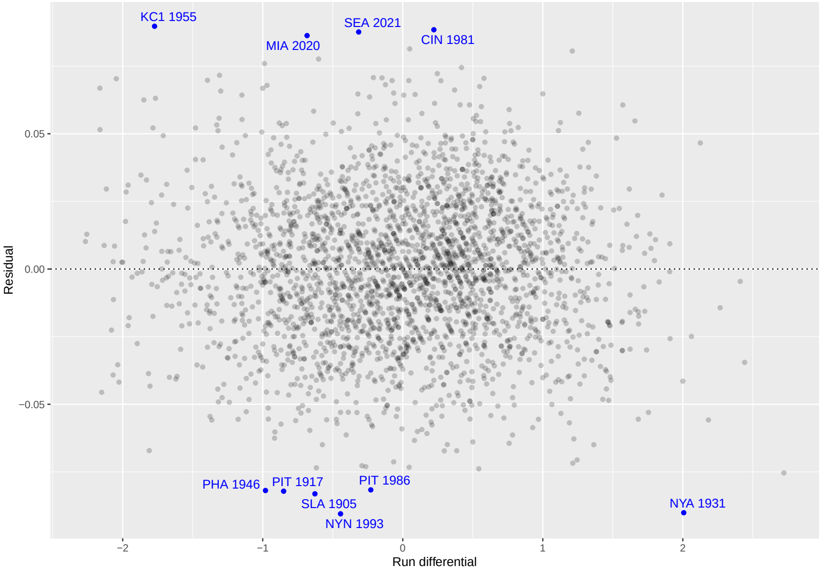
We can see that some teams exhibited deviations from the trend

```r
library(broom)
library(ggrepel)
dat_aug = augment(m, data = dat)

base_plot = ggplot(dat_aug, aes(x = RD, y = .resid)) +
        geom_point(alpha = 0.2) +
        geom_hline(yintercept = 0, linetype = 3) +
        xlab("Run differential") + ylab("Residual")

highlight = dat_aug %>%
        arrange(desc(abs(.resid))) %>%
        head(10)

base_plot +
  geom_point(data = highlight, color = "blue") +
  geom_text_repel(data = highlight, color = "blue",
                  aes(label = paste(teamID, yearID)))
```


KC1 1955
MIA 2020
SEA 2021
CIN 1981
PHA 1946
PIT 1917
SLA 1905
NYN 1993
PIT 1986
NYA 1931
0.05
0.00
−0.05
Residual
−2
−1
0
1
2
Run differential

Pythagorean formula for winning percentage

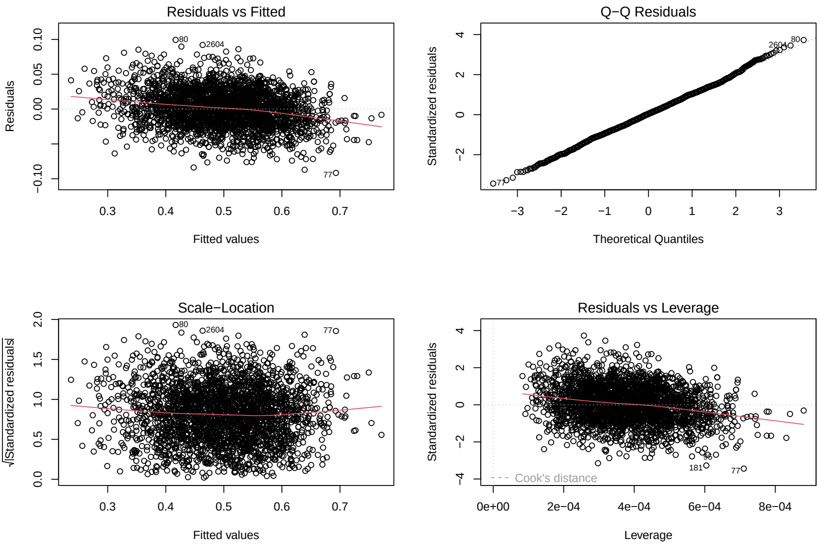
Bill James empirically derived the following non-linear formula to estimate winning percentage, called the Pythagorean expectation

$$\mathrm{W}pct = \frac{R^2}{R^2 + RA^2}$$

```r
dat_aug = dat_aug %>%
        mutate(Wpct_pyt = R^2 / (R^2 + RA^2))
m2 = lm(Wpct ~ 0 +  Wpct_pyt, data = dat_aug)
summary(m2)
```

```
##
## Call:
## lm(formula = Wpct ~ 0 + Wpct_pyt, data = dat_aug)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.091696 -0.016853  0.001062  0.018817  0.099341
##
## Coefficients:
##          Estimate Std. Error t value Pr(>|t|)
## Wpct_pyt 0.997322   0.001022   975.8   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02664 on 2639 degrees of freedom
## Multiple R-squared:  0.9972, Adjusted R-squared:  0.9972
## F-statistic: 9.522e+05 on 1 and 2639 DF,  p-value: < 2.2e-16
```

```r
par(mfrow = c(2,2))
plot(m2)
```



The formula is very powerful and it explains expected wins very well!

```
dat_aug %>%
  summarise(rmse = sqrt((mean(.resid^2))))
```

```
## # A tibble: 1 x 1
##     rmse
##    <dbl>
## 1 0.0266
```

```
sqrt(mean(resid(m2)^2))
```

```
## [1] 0.02663617
```

```
dat_aug = dat_aug %>% mutate(residuals_pyt = Wpct - Wpct_pyt)
dat_aug %>%
  summarise(rmse_pyt = sqrt((mean(residuals_pyt^2))))
```

```
## # A tibble: 1 x 1
##   rmse_pyt
##      <dbl>
## 1   0.0267
```

The nonlinear nature of the equation allow for more realistic prediction in the extremes:

```r
# RD = 5
predict(m, newdata = data.frame(RD = 5))
```

```
##        1
## 1.009532
```

```r
# RD = 5; team1 scores 6 and allows 1, team2 scores 10 and allows 5
cand = c((6*162)^2 / ((6*162)^2 + 162^2), 1620^2 / (1620^2 + 810^2))
predict(m2, newdata = data.frame(Wpct_pyt = cand))
```

```
##         1         2
## 0.9703669 0.7978573
```

```r
# theoretical limits for teams with RD >= 0
predict(m2, newdata = data.frame(Wpct_pyt = c(1,1/2)))
```

```
##         1         2
## 0.9973216 0.4986608
```

Obtain optimal exponent

Start with the Pythagorean formula with an unknown exponent

$$\mathrm{W}pct = \frac{W}{W + L} = \frac{R^k}{R^k + RA^k}$$

A bit of algebra yields

$$\frac{W}{L} = \frac{R^k}{RA^k}$$

Taking logarithms yields

$$\log\left(\frac{W}{L}\right) = k\log\left(\frac{R}{RA}\right)$$

```r
dat_aug = dat_aug %>%
  mutate(logWratio = log(W / L),
         logRratio = log(R / RA))

pyFit = lm(logWratio ~ 0 + logRratio, data = dat_aug)
pyFit
```

```
##
## Call:
## lm(formula = logWratio ~ 0 + logRratio, data = dat_aug)
##
## Coefficients:
## logRratio
##     1.851
```

Nearly perfect relationship

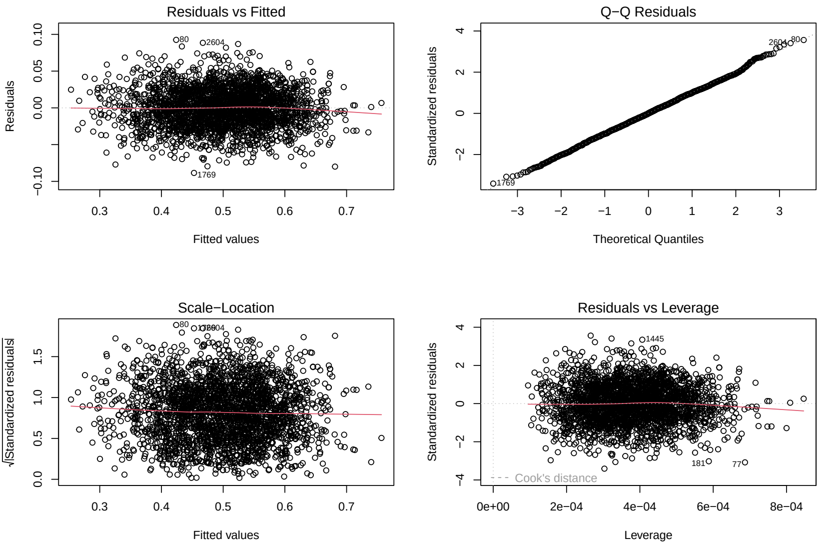
```
k = pyFit$coef
dat_aug = dat_aug %>%
  mutate(Wpct_pytk = R^k/(R^k + RA^k))
m3 = lm(Wpct ~ 0 + Wpct_pytk, data = dat_aug)
dat_aug = dat_aug %>% mutate(residuals_pytk = Wpct - Wpct_pytk)
summary(m3)
```

```
##
## Call:
## lm(formula = Wpct ~ 0 + Wpct_pytk, data = dat_aug)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.088545 -0.017524 -0.000001  0.017889  0.092493
##
## Coefficients:
##           Estimate Std. Error t value Pr(>|t|)
## Wpct_pytk 0.9993397  0.0009983    1001   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02597 on 2639 degrees of freedom
## Multiple R-squared:  0.9974, Adjusted R-squared:  0.9974
## F-statistic: 1.002e+06 on 1 and 2639 DF,  p-value: < 2.2e-16
```

```
sqrt(mean(resid(m3)^2))
```

```
## [1] 0.02596572
```

```r
par(mfrow = c(2,2))
plot(m3)
```



Example: 2011 predictions using the Pythagorean formula

The 2011 Boston Red Sox won 90 games and missed the playoffs. They were beaten out by the Rays who won 91 games. However, the Pythagorean formula and run differentials predicted more wins for the Red Sox.

```r
## Red Sox predictions
c(162 * predict(m, newdata = data.frame(RD = (875-737)/162)),
162 * predict(m2, newdata = data.frame(Wpct_pyt = 875^2/(875^2 + 737^2))),
162 * predict(m3, newdata = data.frame(Wpct_pytk = 875^k/(875^k + 737^k))))
```

```
##         1         1 logRratio
##  95.06287  94.51376  93.69687
```

```r
## Rays predictions
c(162 * predict(m, newdata = data.frame(RD = (707-614)/162)),
162 * predict(m2, newdata = data.frame(Wpct_pyt = 707^2/(707^2 + 614^2))),
162 * predict(m3, newdata = data.frame(Wpct_pytk = 707^k/(707^k + 614^k))))
```

```
##         1         1 logRratio
##  90.47707  92.10140  91.45204
```

The Red Sox had their victories decided by a larger margin than their losses.

```r
library(retrosheet)
library(skimr)
gl2011 = get_retrosheet("game", 2011)
BOS2011 = gl2011 %>%
  filter(HmTm == "BOS" | VisTm == "BOS") %>%
  select(VisTm, HmTm, VisRuns, HmRuns) %>%
  mutate(ScoreDiff = ifelse(HmTm == "BOS", HmRuns - VisRuns, VisRuns - HmRuns),
         W = ScoreDiff > 0)

(BOS2011 %>% group_by(W) %>% skim(ScoreDiff))[, 2:7]
```

```
## # A tibble: 2 x 6
##   skim_variable W     n_missing complete_rate numeric.mean numeric.sd
##   <chr>         <lgl>     <int>         <dbl>        <dbl>      <dbl>
## 1 ScoreDiff     FALSE         0             1        -3.46       2.56
## 2 ScoreDiff     TRUE          0             1         4.3        3.28
```

Meanwhile, the Rays had their victories decided by a much closer margin than their losses.

```
TBA2011 = gl2011 %>%
  filter(HmTm == "TBA" | VisTm == "TBA") %>%
  select(VisTm, HmTm, VisRuns, HmRuns) %>%
  mutate(ScoreDiff = ifelse(HmTm == "TBA", HmRuns - VisRuns, VisRuns - HmRuns),
         W = ScoreDiff > 0)

(TBA2011 %>% group_by(W) %>% skim(ScoreDiff))[, 2:7]
```

```
## # A tibble: 2 x 6
##   skim_variable W     n_missing complete_rate numeric.mean numeric.sd
##   <chr>         <lgl>     <int>         <dbl>        <dbl>      <dbl>
## 1 ScoreDiff     FALSE         0             1        -3.27       2.16
## 2 ScoreDiff     TRUE          0             1         3.57       2.54
```

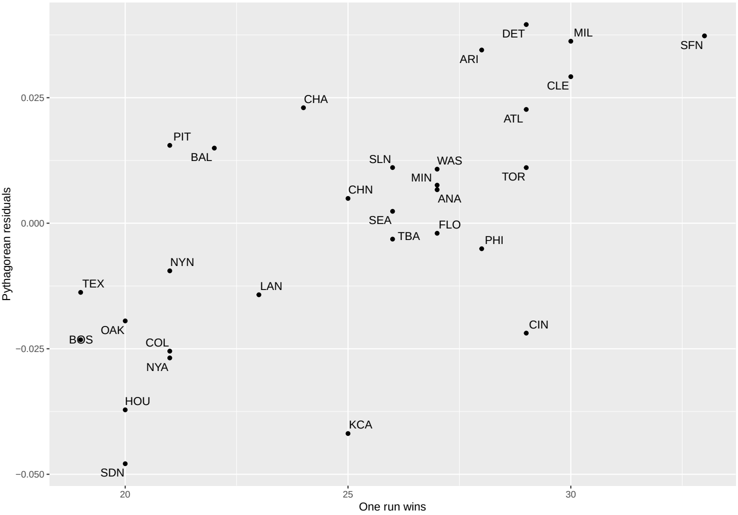
```r
results = gl2011 %>%
  select(VisTm, HmTm, VisRuns, HmRuns) %>%
  mutate(winner = ifelse(HmRuns > VisRuns, HmTm, VisTm),
         diff = abs(VisRuns - HmRuns))

one_run_wins = results %>%
  filter(diff == 1) %>%
  group_by(winner) %>%
  summarise(one_run_w = n())

dat2011 = dat_aug %>%
  filter(yearID == 2011) %>%
  mutate(teamID = ifelse(teamID == "LAA", "ANA", as.character(teamID))) %>%
  inner_join(one_run_wins, by = c("teamID" = "winner"))

ggplot(data = dat2011, aes(x = one_run_w, y = residuals_pyt)) +
  geom_point() +
  geom_text_repel(aes(label = teamID)) +
  xlab("One run wins") + ylab("Pythagorean residuals")
```

The figure shows that the Red Sox had a small number of one-run victories and a large negative Pythagorean residual.



Example: top closer

We can see that since 1990 teams with a top closer outpace their Pythagorean wins by 0.008764 * 162 = 1.42 wins on average.

```
top_closers = Pitching %>%
  filter(yearID >= 1990, GF >= 50 & ERA <= 2.50) %>%
  select(playerID, yearID, teamID)

dat_aug %>% inner_join(top_closers) %>%
  pull(residuals_pytk) %>%
        summary()
```

```
## Joining with `by = join_by(teamID, yearID)`

##      Min.   1st Qu.    Median      Mean   3rd Qu.      Max.
## -0.047895 -0.007345  0.008251  0.008764  0.024027  0.081556
```