

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 between runs and wins

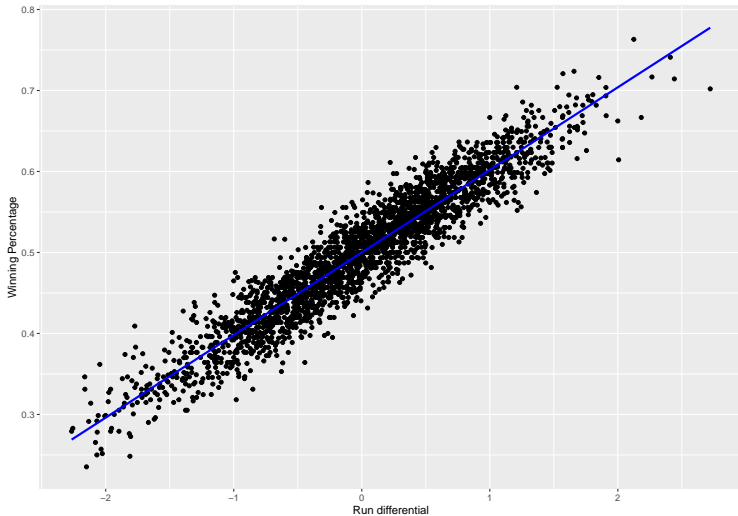
We will consider the relationship between runs and wins since 1900.

```
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.

```
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")
```

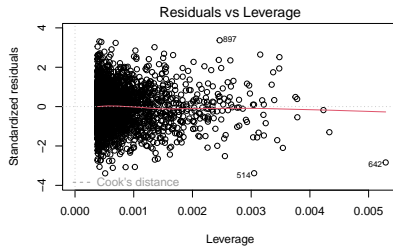
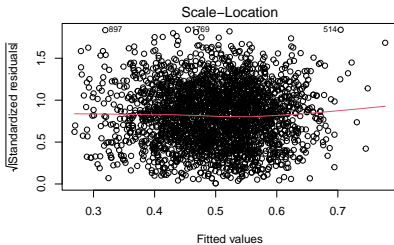
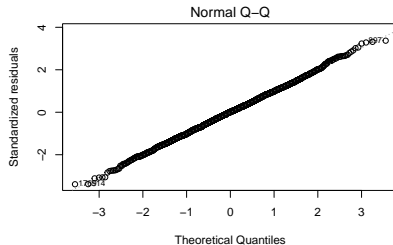
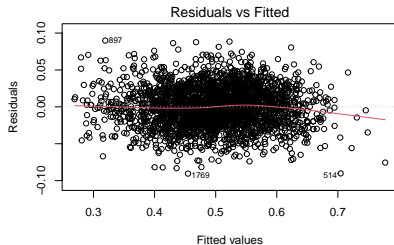


Winning percentage is well explained by run differential

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

```
##
## Call:
## lm(formula = Wpct ~ RD, data = dat)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.090485 -0.018487  0.000281  0.017993  0.089843
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 0.4999985   0.0005225   956.8  <2e-16 ***
## RD          0.1019617   0.0006870   148.4  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.0267 on 2608 degrees of freedom
## Multiple R-squared:  0.8941, Adjusted R-squared:  0.8941
## F-statistic: 2.203e+04 on 1 and 2608 DF, p-value: < 2.2e-16
```

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



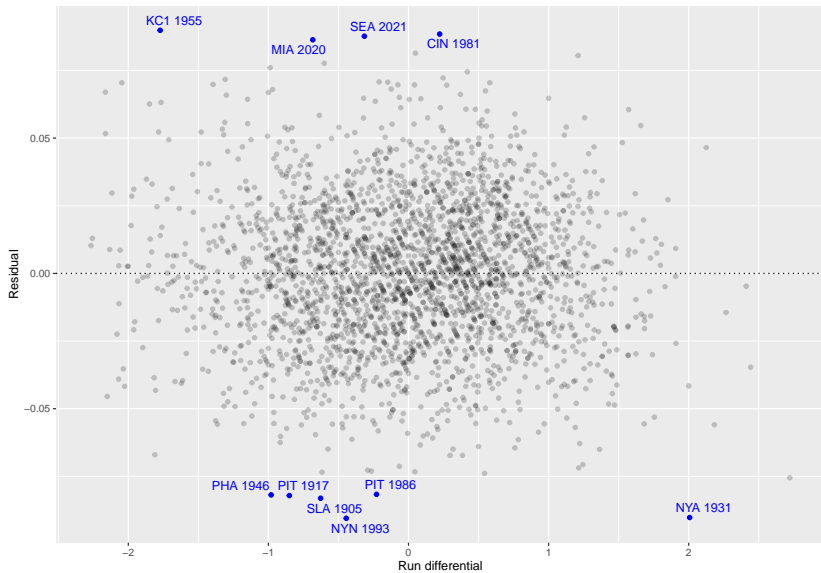
We can see that some teams exhibited deviations from the trend

```
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)))
```



Pythagorean formula for winning percentage

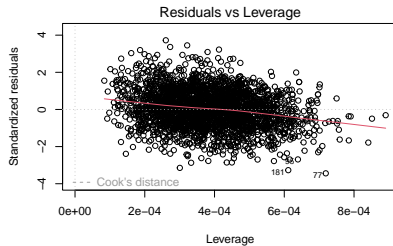
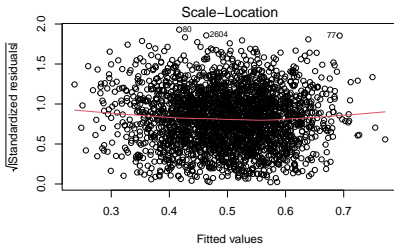
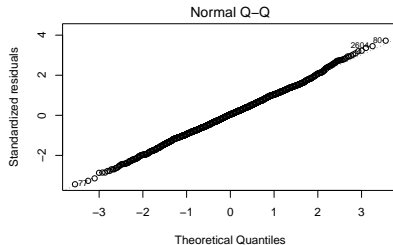
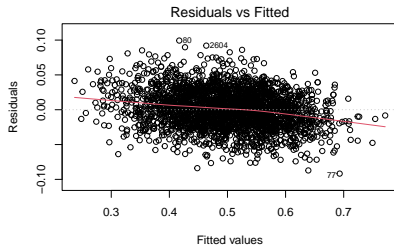
Bill James empirically derived the following non-linear formula to estimate winning percentage, called the Pythagorean expectation

$$Wpct = \frac{R^2}{R^2 + RA^2}$$

```
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   
## -0.091734 -0.016946  0.001017  0.018804  0.099318   
##  
## Coefficients:  
##              Estimate Std. Error t value Pr(>|t|)      
## Wpct_pyt  0.997376    0.001029     969  <2e-16 ***  
## ---  
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
##  
## Residual standard error: 0.02667 on 2609 degrees of freedom  
## Multiple R-squared:  0.9972, Adjusted R-squared:  0.9972   
## F-statistic: 9.389e+05 on 1 and 2609 DF,  p-value: < 2.2e-16
```

```
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 0.0267
```

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

```
## [1] 0.02666974
```

```
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:

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

```
##          1
## 1.009807
```

```
# 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
## 0.9704202 0.7979011
```

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

```
##          1          2
## 0.9973763 0.4986882
```

Obtain optimal exponent

Start with the Pythagorean formula with an unknown exponent

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

A bit of algebra yields

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

Taking logarithms yields

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

```
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  
##      1.853
```

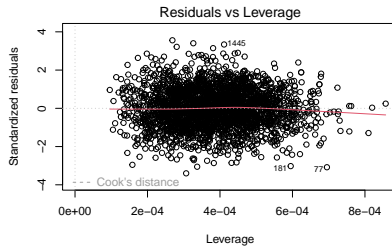
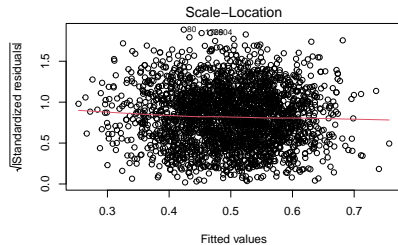
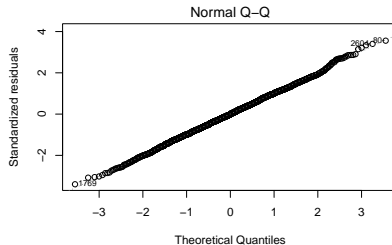
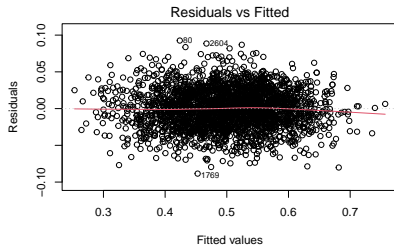
Nearly perfect relationship

```
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
## -0.088489 -0.017520 -0.000029  0.017899  0.092588
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## Wpct_pytk  0.999353    0.001006   993.2  <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.02603 on 2609 degrees of freedom
## Multiple R-squared:  0.9974, Adjusted R-squared:  0.9974
## F-statistic: 9.864e+05 on 1 and 2609 DF,  p-value: < 2.2e-16
sqrt(mean(resid(m3)^2))
```

```
## [1] 0.026022
```

```
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.

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_pyt_k = 875^k/(875^k + 737^k))))
```

```
##           1           1 logRatio  
## 95.07047 94.51895 93.71484
```

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_pyt_k = 707^k/(707^k + 614^k))))
```

```
##           1           1 logRatio  
## 90.48219 92.10646 91.46711
```

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

```
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
```

```

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

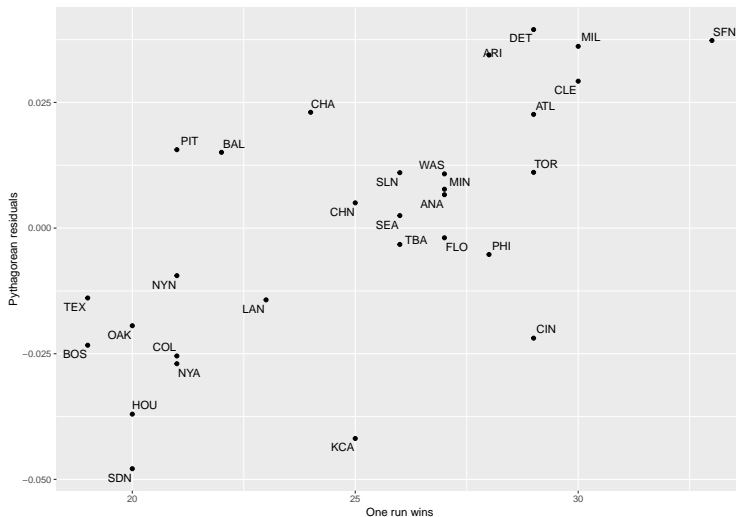
one_run_wins <- results %>%
  filter(diff == 1) %>%
  group_by(winner) %>%
  summarise(one_run_w = n())

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 $0.008640 * 162 = 1.40$ 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, by = c("teamID", "yearID")

##      Min.    1st Qu.      Median        Mean     3rd Qu.      Max.
## -0.047877 -0.007850   0.008125   0.008640   0.024134   0.081549
```