Solution to Lab 1

Due on 02/03 at 11:59 pm

Instructions: This lab report needs to be professional. Only report relevant and finalized code. Your writing should be concise and void of spelling errors. Use code chunk options to hide unnecessary messages/warnings. Your report should be reproducible. Reports that involve simulations need to have the random seed specified so that simulation results are reproducible. You are allowed to work on this lab assignment in groups of 2-3. You still need to submit an individual lab report if you do work in a group, and you need to list your collaborators.

Question 1 In lecture it was demonstrated that baseball is a game of offense, pitching, and defense with a regression model that considered expected run differential as a function of explanatory variables OPS, WHIP, and FP. Do the following:

- Fit a similar regression model with runs as the response variable. Report problems with this model. Investigate problematic residuals to discover what went wrong. Fix the problem with this model by adding categorical variable(s) to the list of explanatory variables. Briefly explain what went wrong.

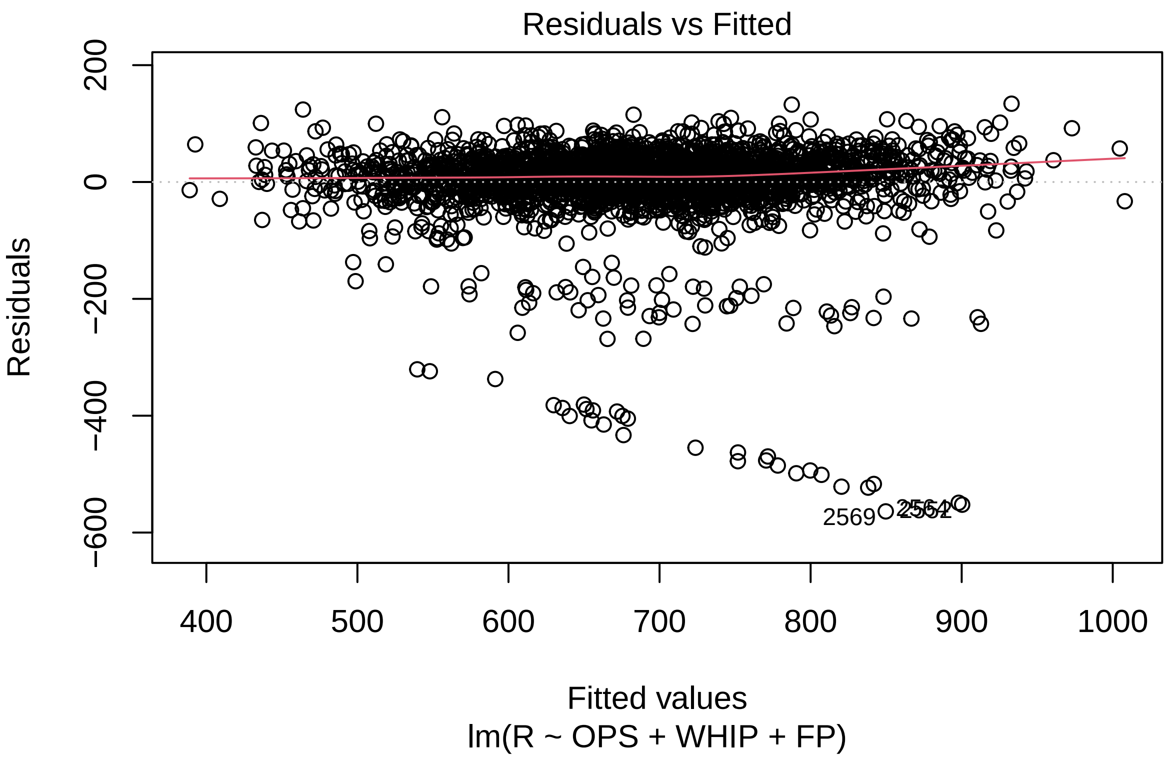
Solution

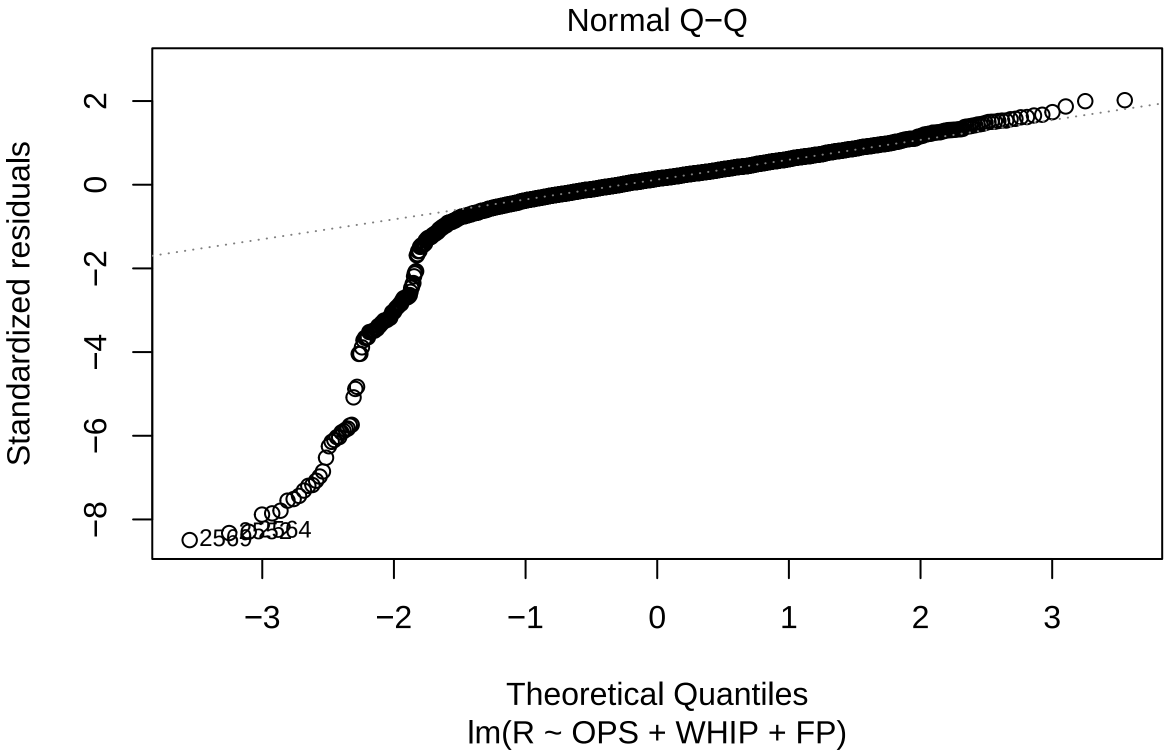
```r
library(Lahman)
library(tidyverse)
dat <- Teams %>%
    select(yearID, franchID, W, L, AB, H, X2B, X3B, HR, BB, HBP, SF,
                 HA, HRA, BBA, SOA, IPouts, FP, R, RA, G) %>%
    filter(yearID >= 1900) %>%
    replace_na(list(HBP = 0, SF = 0)) %>%
    mutate(RD = (R - RA) / (W + L), X1B = H - (X2B + X3B + HR)) %>%
    mutate(OBP = (H + BB + HBP)/(AB + BB + HBP + SF)) %>%
    mutate(SLG = (X1B + 2*X2B + 3*X3B + 4*HR)/AB) %>%
    mutate(OPS = OBP + SLG) %>%
    mutate(WHIP = 3*(HA + BBA)/IPouts) %>%
    mutate(FIP = 3*(13*HRA + 3*BBA - 2*SOA)/IPouts)

mod_1a <- lm(R ~ OPS + WHIP + FP, data = dat)
summary(mod_1a)
```

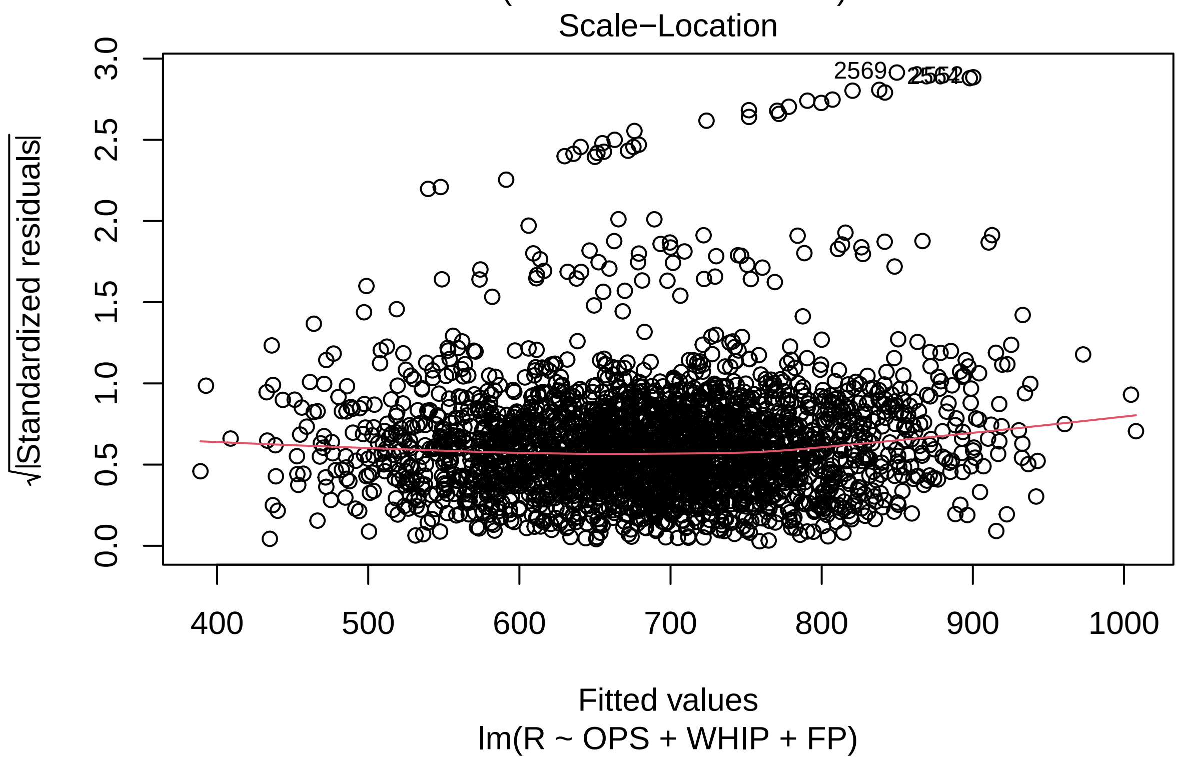
```
##
## Call:
## lm(formula = R ~ OPS + WHIP + FP, data = dat)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -563.74  -13.16    9.62   29.36  133.98
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1187.52     150.14   7.909 3.79e-15 ***
## OPS          1958.31      29.83  65.659  < 2e-16 ***
```

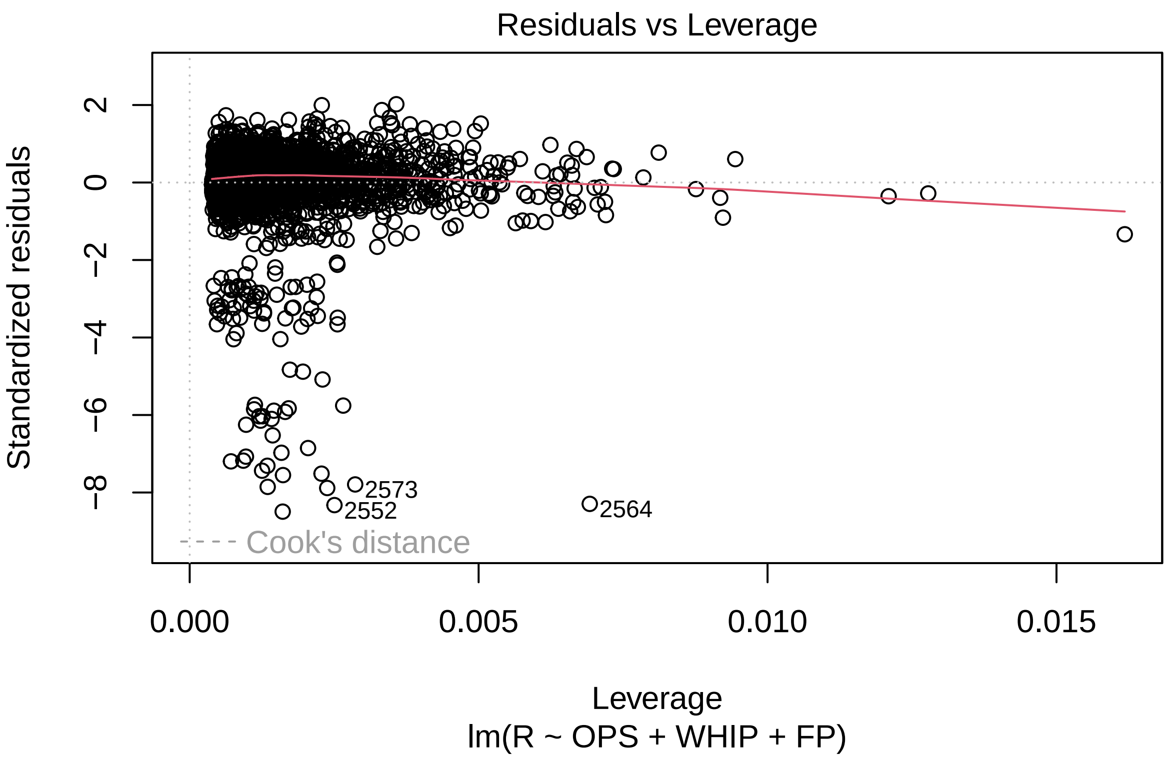
```
## WHIP          -45.19      12.32  -3.668  0.00025 ***
## FP          -1882.77     159.93 -11.772  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 66.43 on 2606 degrees of freedom
## Multiple R-squared:  0.6697, Adjusted R-squared:  0.6693
## F-statistic:  1761 on 3 and 2606 DF,  p-value: < 2.2e-16
```

```
## Q-Q plot seems weird
plot(mod_1a)
```



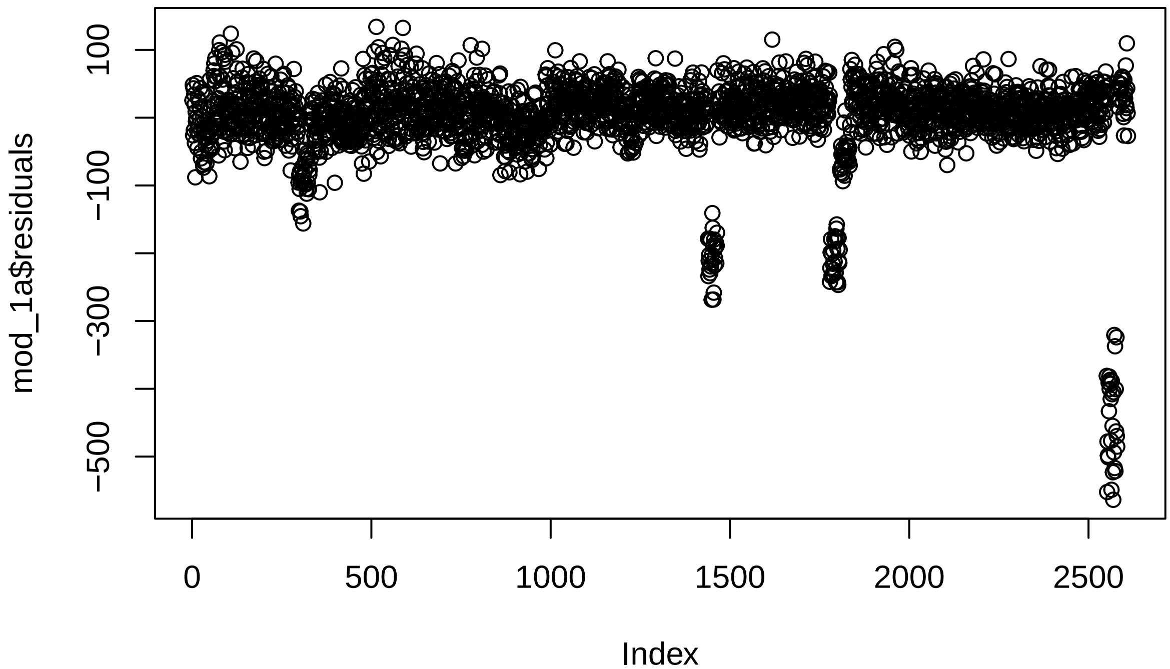

Normal Q−Q
Standardized residuals
2
0
−2
−4
−6
−8
−3
−2
−1
0
1
2
3
Theoretical Quantiles
lm(R ~ OPS + WHIP + FP)


Scale−Location
√|Standardized residuals|
3.0
2.5
2.0
1.5
1.0
0.5
0.0
2569
400
500
600
700
800
900
1000
Fitted values
lm(R ~ OPS + WHIP + FP)



The plots appear to show normality except for several extreme negative residuals. The Q-Q plot shows the many observations drop at the left tail.

```
plot(mod_1a$residuals)
```



Several observations have large residuals

```
unique(dat[which(abs(mod_1a$residuals) > 150),]$yearID)
```

```
## [1] 1918 1981 1994 2020
```

Find the seasons that involve large residuals

```
dat %>% filter(yearID %in% c(1918, 1981, 1994, 2020)) %>% select(G)
```

```
##       G
## 1   126
## 2   126
## 3   124
## 4   124
## 5   131
## 6   129
## 7   129
## 8   128
## 9   124
## 10  126
## 11  130
## 12  125
## 13  126
## 14  123
## 15  131
## 16  130
## 17  107
## 18  105
## 19  108
## 20  110
## 21  106
## 22  106
## 23  108
## 24  103
## 25  109
## 26  110
## 27  103
## 28  110
## 29  110
## 30  109
## 31  108
## 32  107
## 33  105
## 34  109
## 35  107
## 36  103
## 37  110
## 38  110
## 39  111
## 40  103
## 41  105
## 42  106
## 43  114
## 44  112
## 45  115
## 46  115
## 47  113
## 48  113
## 49  115
## 50  113
```

```
## 51   117
## 52   115
## 53   115
## 54   115
## 55   115
## 56   114
## 57   113
## 58   115
## 59   114
## 60   113
## 61   113
## 62   114
## 63   115
## 64   114
## 65   117
## 66   112
## 67   115
## 68   115
## 69   114
## 70   115
## 71    60
## 72    60
## 73    60
## 74    60
## 75    60
## 76    60
## 77    60
## 78    60
## 79    60
## 80    58
## 81    60
## 82    60
## 83    60
## 84    60
## 85    60
## 86    60
## 87    60
## 88    60
## 89    60
## 90    60
## 91    60
## 92    60
## 93    60
## 94    60
## 95    60
## 96    58
## 97    60
## 98    60
## 99    60
## 100   60
```

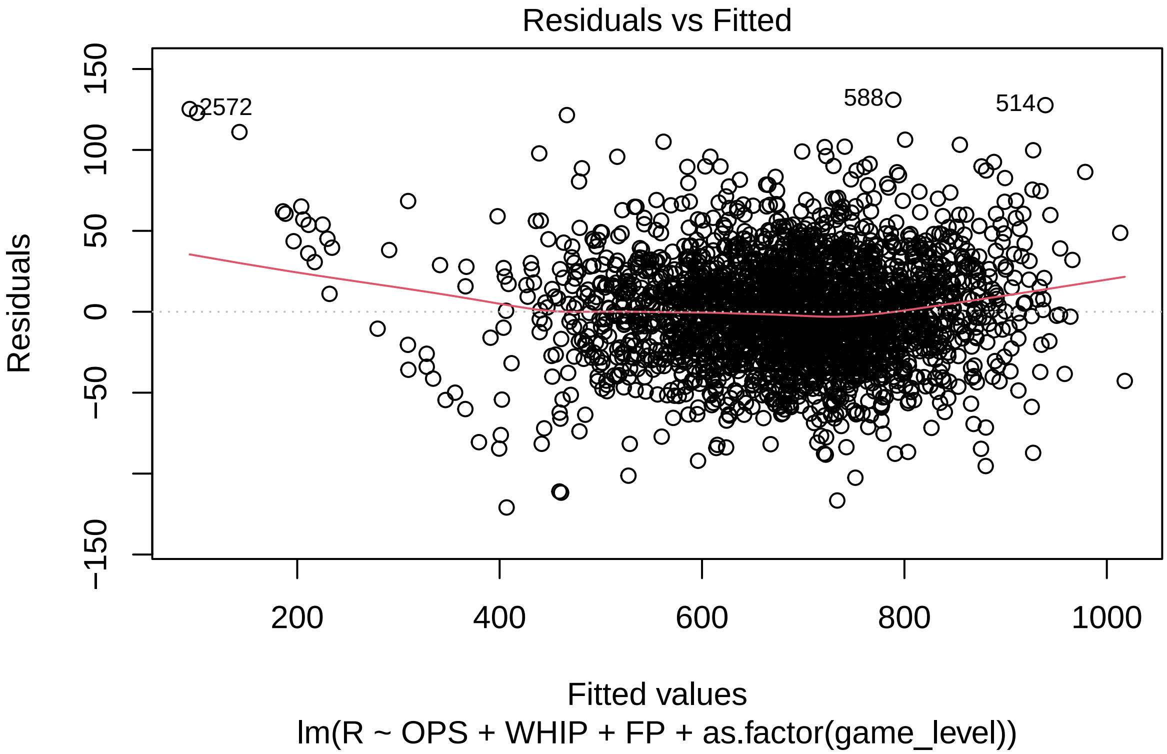
These seasons were all short seasons.

```
dat_game <- dat %>%
  mutate(game_level = ifelse(G <= 60, 1,
```

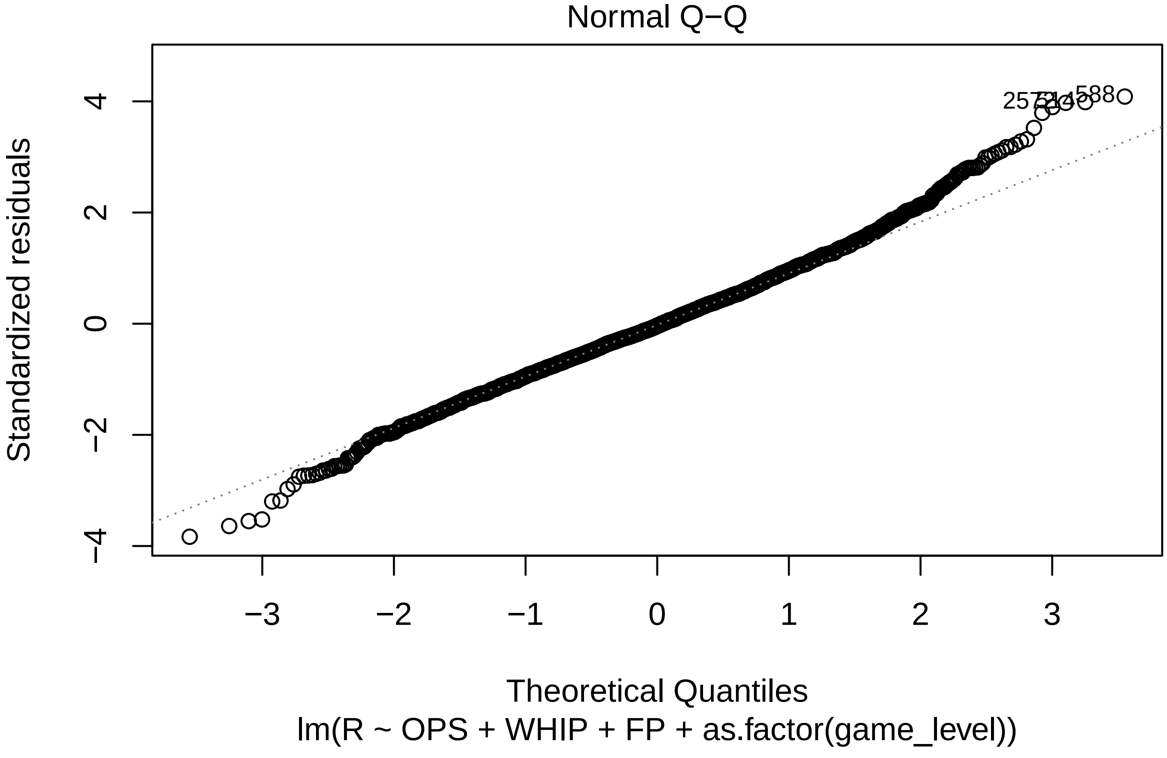
```r
                              ifelse(G <= 120, 2,
                                     ifelse(G <= 140, 3,
                                            ifelse(G <=160, 4, 5)))))
mod_1a2 <- lm(R ~ OPS + WHIP + FP + as.factor(game_level), data = dat_game)
summary(mod_1a2)
```

```
##
## Call:
## lm(formula = R ~ OPS + WHIP + FP + as.factor(game_level), data = dat_game)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -120.917  -20.719   -1.158   19.383  130.980
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)             1054.418    100.633   10.48   <2e-16 ***
## OPS                     1994.325     14.459  137.93   <2e-16 ***
## WHIP                     -56.385      5.992   -9.41   <2e-16 ***
## FP                     -2210.557    105.010  -21.05   <2e-16 ***
## as.factor(game_level)2   237.649      7.324   32.45   <2e-16 ***
## as.factor(game_level)3   384.817      7.615   50.53   <2e-16 ***
## as.factor(game_level)4   443.770      6.111   72.62   <2e-16 ***
## as.factor(game_level)5   460.707      5.930   77.70   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 32.1 on 2602 degrees of freedom
## Multiple R-squared:  0.923,  Adjusted R-squared:  0.9228
## F-statistic:  4456 on 7 and 2602 DF,  p-value: < 2.2e-16
```

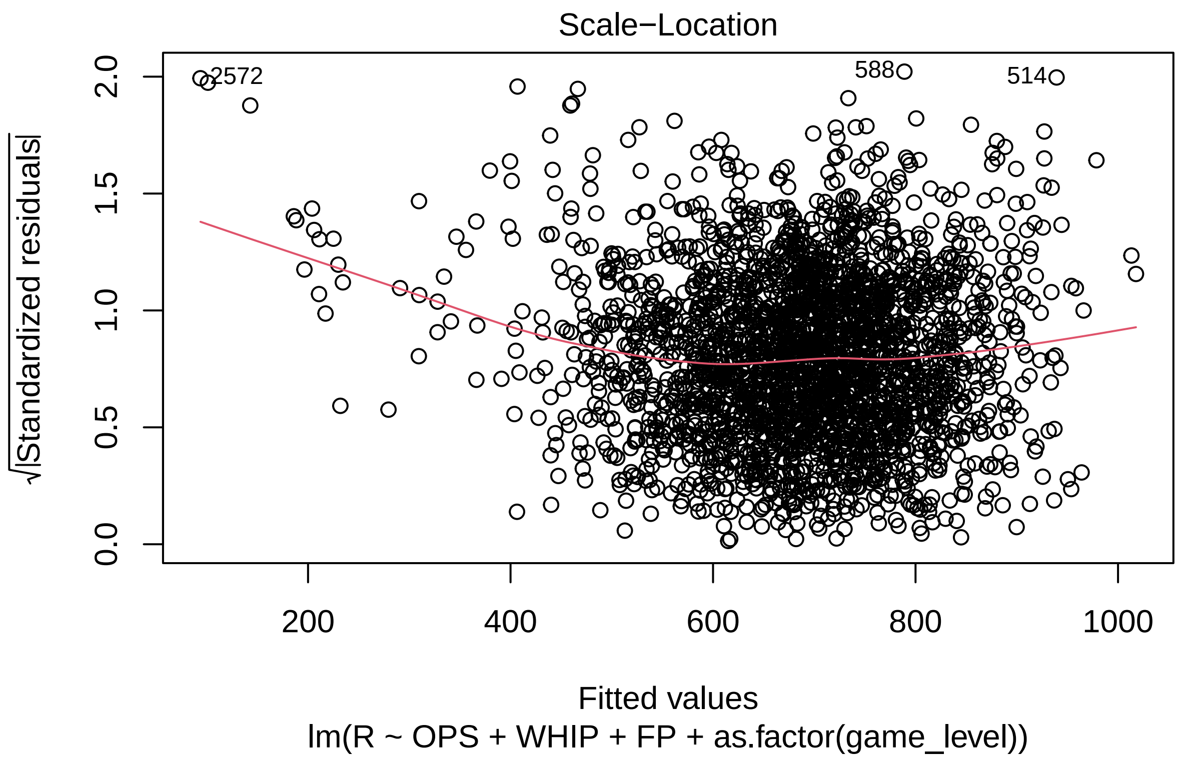
```r
plot(mod_1a2)
```

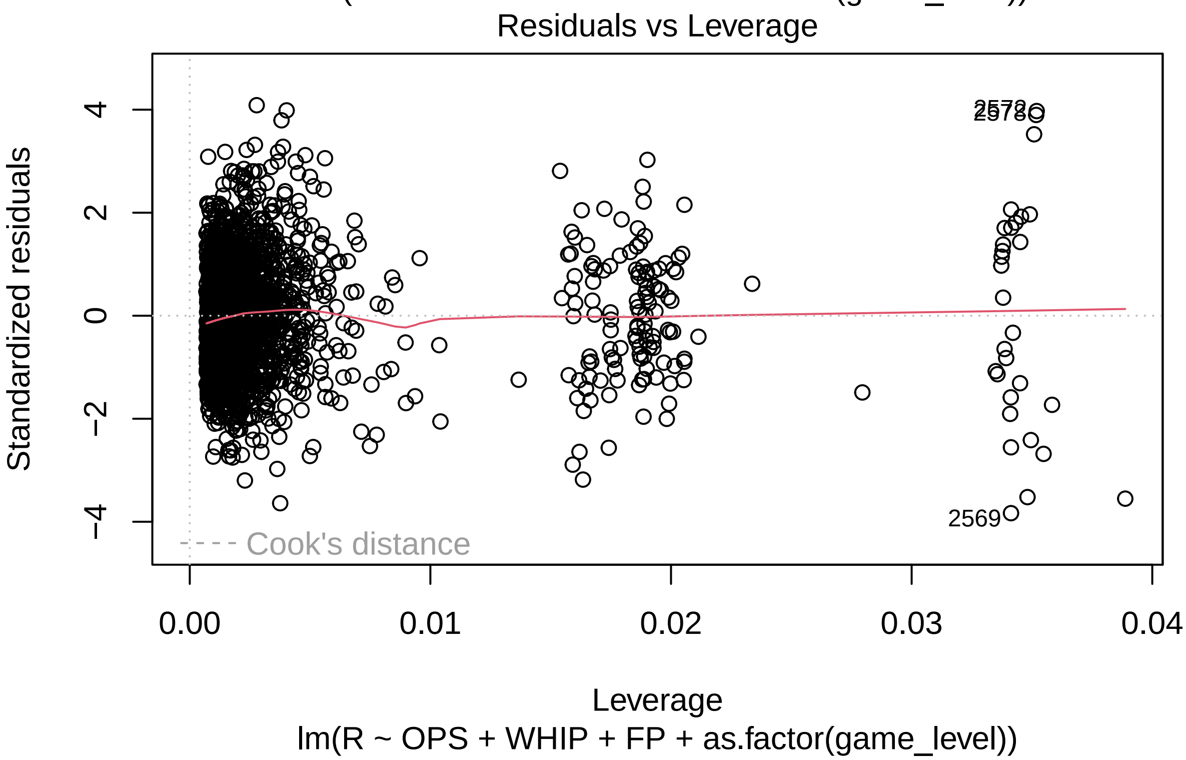


Residuals vs Fitted
Residuals
Fitted values
lm(R ~ OPS + WHIP + FP + as.factor(game_level))
2572
588
514
150
100
50
0
−50
−150
200
400
600
800
1000



Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(R ~ OPS + WHIP + FP + as.factor(game_level))
2572
514
588
4
2
0
−2
−4
−3
−2
−1
0
1
2
3





As the model shows, the addition of a categorical variable for games played significantly improved the residuals in the model.

WHIP and FP are defensive metrics and should have nothing to do with an offensive stat - Runs. WHIP has a negative coefficient, meaning that teams with weaker pitching will also score fewer runs, which checks out for the worst teams in the league. FP also has a negative coefficient, which means that teams who have a better defense will score fewer runs. This could illustrate the offense vs. defense aspect of constructing a lineup of position players.

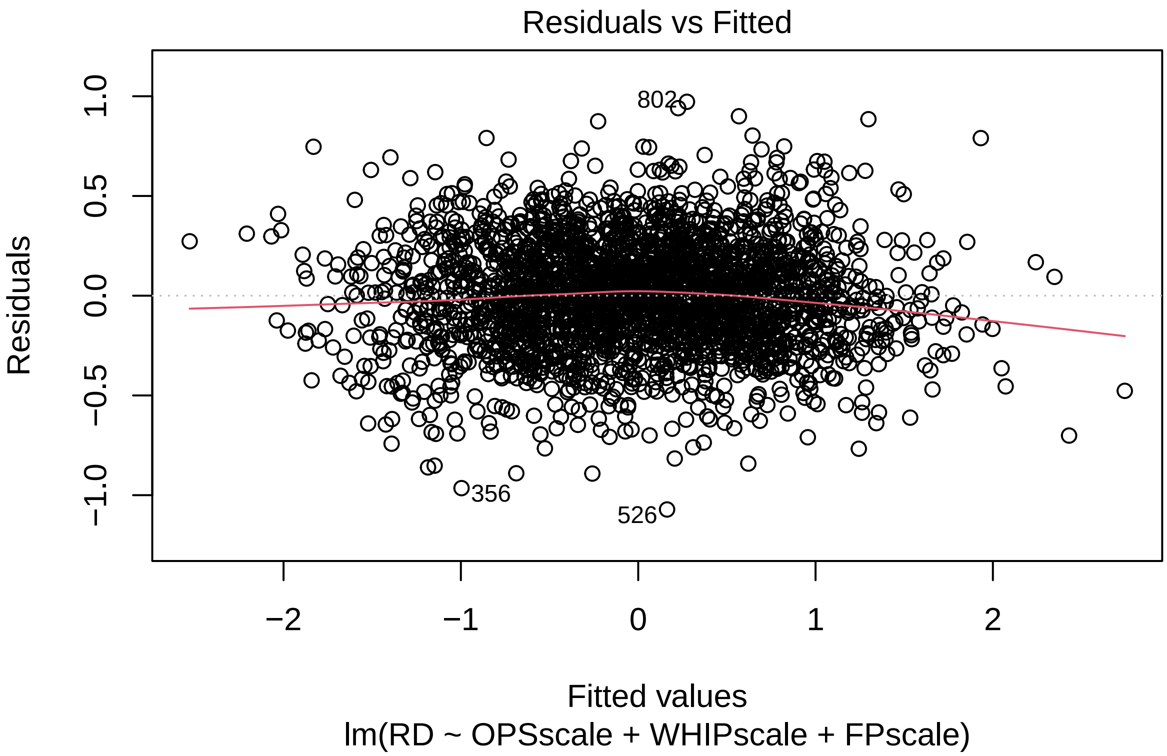
- We can significantly improve the regression model in the notes through a principled rescaling of OPS, WHIP, and FP. Split the Teams data frame by `yearID` and, for each year, create variables `OPSscale = OPS/avgOPS`, `WHIPscale = avgWHIP/WHIP`, and `FPscale = avgFP/FP` which require you to first create league average variables `avgOPS`, `avgWHIP`, and `avgFP`. Fit the linear regression model with runs differential as the response and explanatory variables `OPSscale`, `WHIPscale`, and `FPscale`, and report relevant output. Why does this model perform so much better than the model in the notes? Support your answer. Hint: functions `split`, `do.call`, and `lapply` are useful.

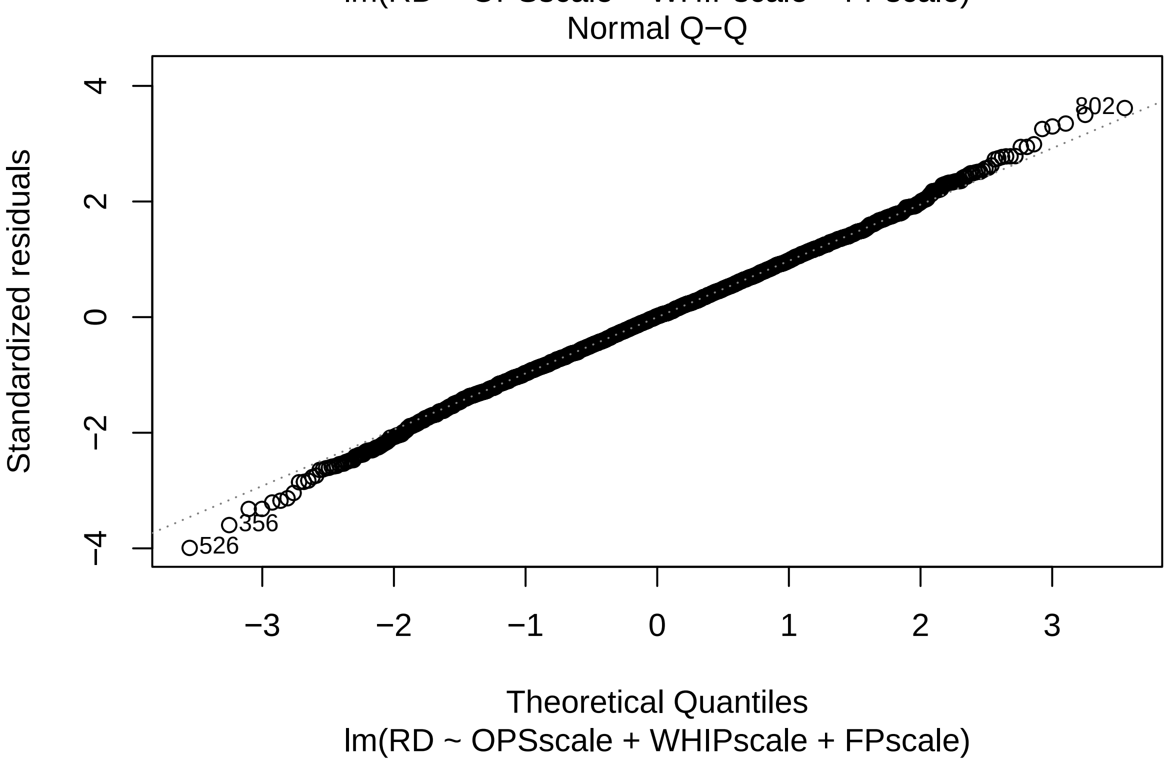
Solution

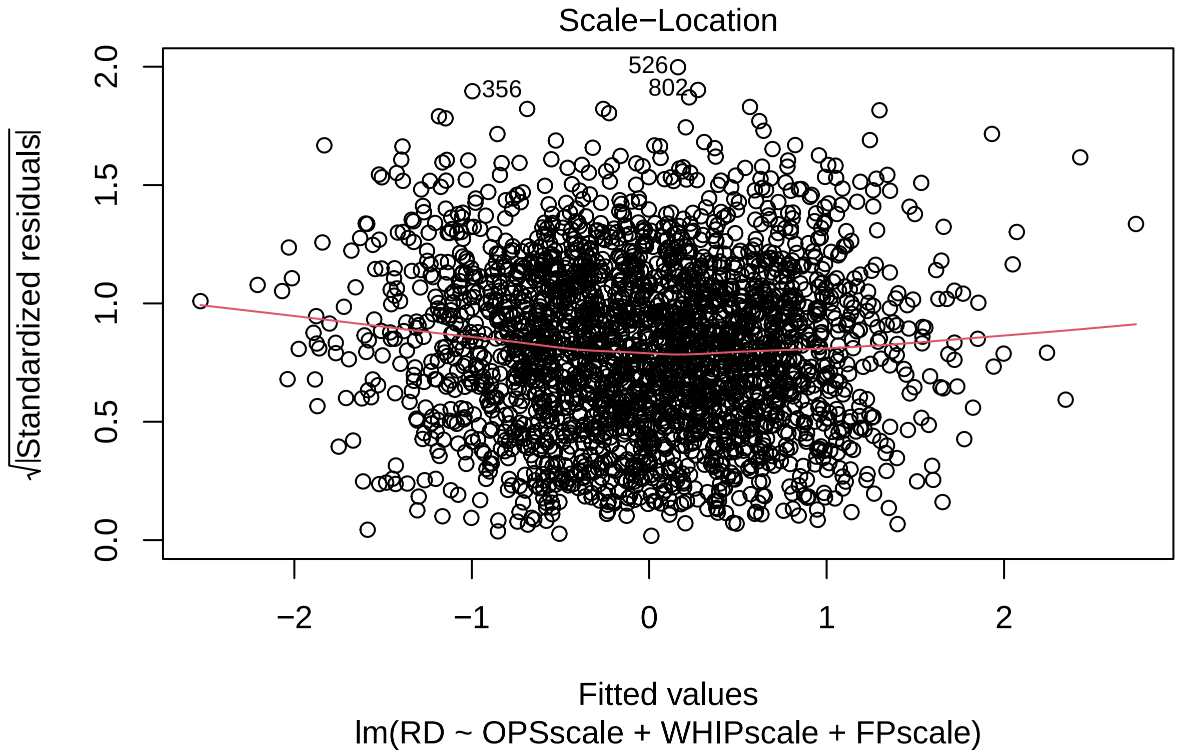
```
avg_data <- dat %>%
group_by(yearID) %>%
summarize(AB = sum(AB), H = sum(H), BB = sum(BB), HBP = sum(HBP), X2B = sum(X2B),
          X3B = sum(X3B),HR = sum(HR), SF = sum(SF), HA = sum(HA), BBA = sum(BBA),
          IPouts = sum(IPouts),avgFP = mean(FP), X1B = sum(X1B)) %>%
  mutate(OBP = (H + BB + HBP)/(AB + BB + HBP + SF)) %>%
    mutate(SLG = (X1B + 2*X2B + 3*X3B + 4*HR)/AB) %>%
    mutate(avgOPS = OBP + SLG) %>%
    mutate(avgWHIP = 3*(HA + BBA)/IPouts) %>% ungroup() %>%
  select(yearID, avgWHIP, avgOPS, avgFP)

scale_data <- merge(dat, avg_data, by="yearID")
scale_data <- scale_data %>%
  mutate(WHIPscale = avgWHIP/WHIP) %>%
  mutate(OPSscale = OPS/avgOPS) %>%
  mutate(FPscale = avgFP/FP)

mod_1b <- lm(RD ~ OPSscale + WHIPscale + FPscale, data = scale_data)
summary(mod_1b)
```

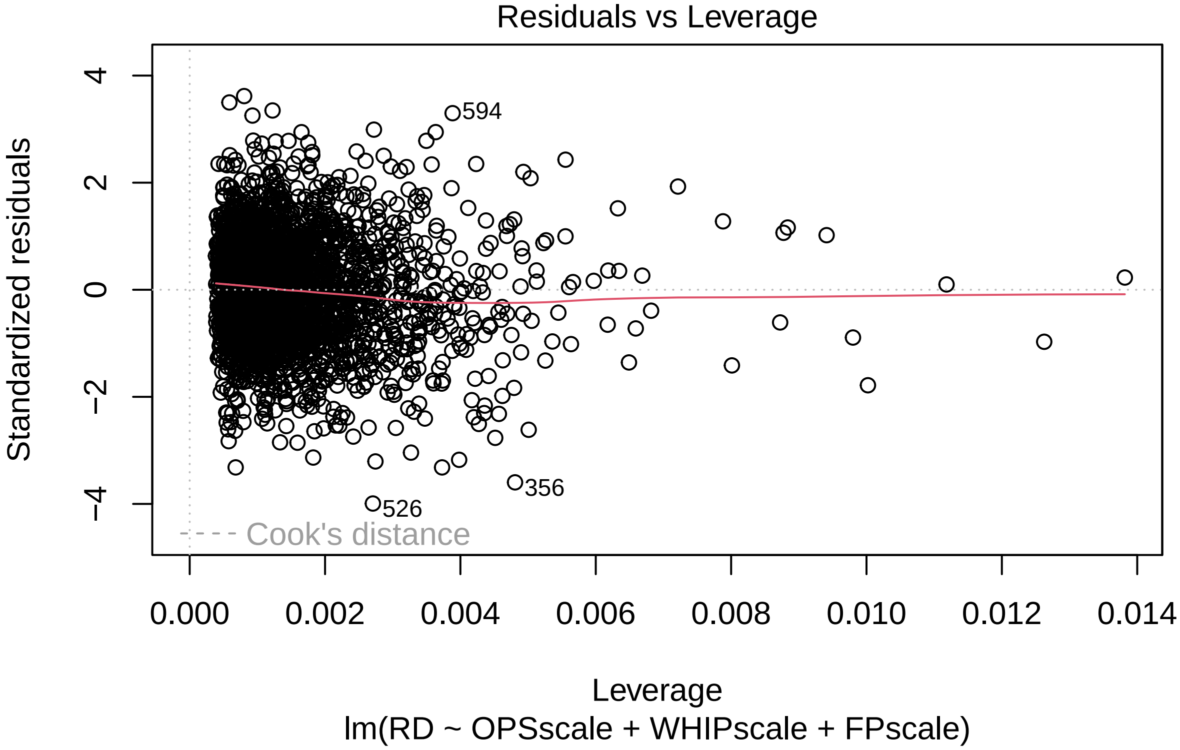
```
##
## Call:
## lm(formula = RD ~ OPSscale + WHIPscale + FPscale, data = scale_data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.07171 -0.17622  0.00406  0.17643  0.97196
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   6.8981     1.6882   4.086 4.52e-05 ***
## OPSscale      9.0333     0.1075  84.028  < 2e-16 ***
## WHIPscale     7.0594     0.0887  79.590  < 2e-16 ***
## FPscale     -23.0130     1.6281 -14.135  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.2688 on 2606 degrees of freedom
## Multiple R-squared:  0.8753, Adjusted R-squared:  0.8752
## F-statistic:  6098 on 3 and 2606 DF,  p-value: < 2.2e-16
```

```
plot(mod_1b)
```


Residuals vs Fitted
Residuals
Fitted values
lm(RD ~ OPSscale + WHIPscale + FPscale)
802
356
526
1.0
0.5
0.0
−0.5
−1.0
−2
−1
0
1
2


Normal Q−Q
Standardized residuals
Theoretical Quantiles
lm(RD ~ OPSscale + WHIPscale + FPscale)
802
356
526
4
2
0
−2
−4
−3
−2
−1
0
1
2
3





This model performs better than the model in the notes because it provides context for the OPS, WHIP, and Fielding Percentage numbers based on the year and type of batted ball environment. By scaling each of these values by the league average, we can get a better understanding of how a team performed compared to the other teams that season.

Question 2 Choose 3 batters and 3 pitchers that have played in at least 10 seasons and do the following:

- Display the seasonal statistics for these players. The following statistics should be included for batters (derivations of unconventional statistics are in parentheses): year, G, AB, R, H, X2B, X3B, HR, RBI, SB, CS, SBpct (SB / (SB + CS)), BB, SO, OBP, SLG, OPS. The following statistics should be included for pitchers: year, W, L, IPouts, H, ER, HR, BB, HBP, SO, ERA, WHIP, SOper9 (SO / IP * 9), SOperBB (SO / BB). These statistics can be found in or computed from statistics that are found in the `Batting` and `Pitching` dataframes in the `Lahman` package.

Solution

```r
batters <- Batting %>%
  filter(playerID == "bondsba01" | playerID == "thomafr04" | playerID == "schmimi01") %>%
  mutate(X1B = H - (X2B + X3B + HR)) %>%
  mutate(SBpct = SB / (SB + CS)) %>%
  mutate(OBP = (H + BB + HBP)/(AB + BB + HBP + SF)) %>%
  mutate(SLG = (X1B + 2*X2B + 3*X3B + 4*HR)/AB) %>%
  mutate(OPS = OBP + SLG) %>%
  select(yearID, playerID, G, AB, R, H, X2B, X3B, HR, RBI, SB, CS, BB, SO, SBpct, OBP, SLG, OPS)

batters %>% filter(playerID == "bondsba01")
```

```
##    yearID  playerID   G  AB   R   H X2B X3B HR RBI SB CS  BB  SO     SBpct
## 1    1986 bondsba01 113 413  72  92  26   3 16  48 36  7  65 102 0.8372093
## 2    1987 bondsba01 150 551  99 144  34   9 25  59 32 10  54  88 0.7619048
## 3    1988 bondsba01 144 538  97 152  30   5 24  58 17 11  72  82 0.6071429
## 4    1989 bondsba01 159 580  96 144  34   6 19  58 32 10  93  93 0.7619048
## 5    1990 bondsba01 151 519 104 156  32   3 33 114 52 13  93  83 0.8000000
## 6    1991 bondsba01 153 510  95 149  28   5 25 116 43 13 107  73 0.7678571
## 7    1992 bondsba01 140 473 109 147  36   5 34 103 39  8 127  69 0.8297872
## 8    1993 bondsba01 159 539 129 181  38   4 46 123 29 12 126  79 0.7073171
## 9    1994 bondsba01 112 391  89 122  18   1 37  81 29  9  74  43 0.7631579
## 10   1995 bondsba01 144 506 109 149  30   7 33 104 31 10 120  83 0.7560976
## 11   1996 bondsba01 158 517 122 159  27   3 42 129 40  7 151  76 0.8510638
## 12   1997 bondsba01 159 532 123 155  26   5 40 101 37  8 145  87 0.8222222
## 13   1998 bondsba01 156 552 120 167  44   7 37 122 28 12 130  92 0.7000000
## 14   1999 bondsba01 102 355  91  93  20   2 34  83 15  2  73  62 0.8823529
## 15   2000 bondsba01 143 480 129 147  28   4 49 106 11  3 117  77 0.7857143
## 16   2001 bondsba01 153 476 129 156  32   2 73 137 13  3 177  93 0.8125000
## 17   2002 bondsba01 143 403 117 149  31   2 46 110  9  2 198  47 0.8181818
## 18   2003 bondsba01 130 390 111 133  22   1 45  90  7  0 148  58 1.0000000
## 19   2004 bondsba01 147 373 129 135  27   3 45 101  6  1 232  41 0.8571429
## 20   2005 bondsba01  14  42   8  12   1   0  5  10  0  0   9   6       NaN
## 21   2006 bondsba01 130 367  74  99  23   0 26  77  3  0 115  51 1.0000000
## 22   2007 bondsba01 126 340  75  94  14   0 28  66  5  0 132  54 1.0000000
##          OBP       SLG       OPS
## 1  0.3298755 0.4164649 0.7463404
## 2  0.3289689 0.4918330 0.8208019
## 3  0.3680782 0.4907063 0.8587845
## 4  0.3510324 0.4258621 0.7768945
## 5  0.4057971 0.5645472 0.9703443
## 6  0.4100946 0.5137255 0.9238201
## 7  0.4558824 0.6236786 1.0795610
## 8  0.4584570 0.6771800 1.1356369
## 9  0.4261603 0.6470588 1.0732192
## 10 0.4314961 0.5770751 1.0085712
## 11 0.4607407 0.6150870 1.0758278
```

```
## 12 0.4463768 0.5845865 1.0309633
## 13 0.4382184 0.6086957 1.0469140
## 14 0.3894009 0.6169014 1.0063023
## 15 0.4398682 0.6875000 1.1273682
## 16 0.5150602 0.8634454 1.3785056
## 17 0.5816993 0.7990074 1.3807068
## 18 0.5290909 0.7487179 1.2778089
## 19 0.6094003 0.8123324 1.4217328
## 20 0.4038462 0.6666667 1.0705128
## 21 0.4543611 0.5449591 0.9993202
## 22 0.4800839 0.5647059 1.0447897
```

```r
batters %>% filter(playerID == "thomafr04")
```

```
##    yearID  playerID   G  AB   R   H X2B X3B HR RBI SB CS  BB  SO     SBpct
## 1    1990 thomafr04  60 191  39  63  11   3  7  31  0  1  44  54 0.0000000
## 2    1991 thomafr04 158 559 104 178  31   2 32 109  1  2 138 112 0.3333333
## 3    1992 thomafr04 160 573 108 185  46   2 24 115  6  3 122  88 0.6666667
## 4    1993 thomafr04 153 549 106 174  36   0 41 128  4  2 112  54 0.6666667
## 5    1994 thomafr04 113 399 106 141  34   1 38 101  2  3 109  61 0.4000000
## 6    1995 thomafr04 145 493 102 152  27   0 40 111  3  2 136  74 0.6000000
## 7    1996 thomafr04 141 527 110 184  26   0 40 134  1  1 109  70 0.5000000
## 8    1997 thomafr04 146 530 110 184  35   0 35 125  1  1 109  69 0.5000000
## 9    1998 thomafr04 160 585 109 155  35   2 29 109  7  0 110  93 1.0000000
## 10   1999 thomafr04 135 486  74 148  36   0 15  77  3  3  87  66 0.5000000
## 11   2000 thomafr04 159 582 115 191  44   0 43 143  1  3 112  94 0.2500000
## 12   2001 thomafr04  20  68   8  15   3   0  4  10  0  0  10  12       NaN
## 13   2002 thomafr04 148 523  77 132  29   1 28  92  3  0  88 115 1.0000000
## 14   2003 thomafr04 153 546  87 146  35   0 42 105  0  0 100 115       NaN
## 15   2004 thomafr04  74 240  53  65  16   0 18  49  0  2  64  57 0.0000000
## 16   2005 thomafr04  34 105  19  23   3   0 12  26  0  0  16  31       NaN
## 17   2006 thomafr04 137 466  77 126  11   0 39 114  0  0  81  81       NaN
## 18   2007 thomafr04 155 531  63 147  30   0 26  95  0  0  81  94       NaN
## 19   2008 thomafr04  16  60   7  10   1   0  3  11  0  0  11  13       NaN
## 20   2008 thomafr04  55 186  20  49   6   1  5  19  0  0  28  44       NaN
##          OBP       SLG       OPS
## 1  0.4541667 0.5287958 0.9829625
## 2  0.4528571 0.5527728 1.0056300
## 3  0.4388186 0.5357766 0.9745952
## 4  0.4260355 0.6065574 1.0325929
## 5  0.4874275 0.7293233 1.2167508
## 6  0.4544049 0.6064909 1.0608958
## 7  0.4591680 0.6261860 1.0853539
## 8  0.4560863 0.6113208 1.0674070
## 9  0.3806180 0.4803419 0.8609599
## 10 0.4135593 0.4711934 0.8847527
## 11 0.4356436 0.6254296 1.0610731
## 12 0.3164557 0.4411765 0.7576322
## 13 0.3614650 0.4722753 0.8337403
## 14 0.3897281 0.5622711 0.9519992
## 15 0.4340836 0.5625000 0.9965836
## 16 0.3145161 0.5904762 0.9049923
## 17 0.3810376 0.5450644 0.9261019
## 18 0.3766026 0.4802260 0.8568286
## 19 0.3055556 0.3333333 0.6388889
```

```
## 20 0.3640553 0.3870968 0.7511521
```

```
batters %>% filter(playerID == "schmimi01")
```

```
##    yearID  playerID   G  AB   R   H X2B X3B HR RBI SB CS  BB  SO     SBpct
## 1    1972 schmimi01  13  34   2   7   0   0  1   3  0  0   5  15       NaN
## 2    1973 schmimi01 132 367  43  72  11   0 18  52  8  2  62 136 0.8000000
## 3    1974 schmimi01 162 568 108 160  28   7 36 116 23 12 106 138 0.6571429
## 4    1975 schmimi01 158 562  93 140  34   3 38  95 29 12 101 180 0.7073171
## 5    1976 schmimi01 160 584 112 153  31   4 38 107 14  9 100 149 0.6086957
## 6    1977 schmimi01 154 544 114 149  27  11 38 101 15  8 104 122 0.6521739
## 7    1978 schmimi01 145 513  93 129  27   2 21  78 19  6  91 103 0.7600000
## 8    1979 schmimi01 160 541 109 137  25   4 45 114  9  5 120 115 0.6428571
## 9    1980 schmimi01 150 548 104 157  25   8 48 121 12  5  89 119 0.7058824
## 10   1981 schmimi01 102 354  78 112  19   2 31  91 12  4  73  71 0.7500000
## 11   1982 schmimi01 148 514 108 144  26   3 35  87 14  7 107 131 0.6666667
## 12   1983 schmimi01 154 534 104 136  16   4 40 109  7  8 128 148 0.4666667
## 13   1984 schmimi01 151 528  93 146  23   3 36 106  5  7  92 116 0.4166667
## 14   1985 schmimi01 158 549  89 152  31   5 33  93  1  3  87 117 0.2500000
## 15   1986 schmimi01 160 552  97 160  29   1 37 119  1  2  89  84 0.3333333
## 16   1987 schmimi01 147 522  88 153  28   0 35 113  2  1  83  80 0.6666667
## 17   1988 schmimi01 108 390  52  97  21   2 12  62  3  0  49  42 1.0000000
## 18   1989 schmimi01  42 148  19  30   7   0  6  28  0  1  21  17 0.0000000
##          OBP       SLG       OPS
## 1  0.3250000 0.2941176 0.6191176
## 2  0.3235294 0.3732970 0.6968264
## 3  0.3953148 0.5457746 0.9410894
## 4  0.3667665 0.5231317 0.8898981
## 5  0.3760684 0.5239726 0.9000410
## 6  0.3933934 0.5735294 0.9669228
## 7  0.3636364 0.4346979 0.7983342
## 8  0.3863299 0.5637708 0.9501007
## 9  0.3803681 0.6240876 1.0044557
## 10 0.4354839 0.6440678 1.0795517
## 11 0.4025357 0.5466926 0.9492283
## 12 0.3991031 0.5243446 0.9234477
## 13 0.3829114 0.5359848 0.9188962
## 14 0.3751938 0.5318761 0.9070699
## 15 0.3896499 0.5471014 0.9367514
## 16 0.3882545 0.5478927 0.9361472
## 17 0.3370288 0.4051282 0.7421570
## 18 0.2965116 0.3716216 0.6681332
```

```
pitchers <- Pitching %>%
  filter(playerID == 'maddugr01' | playerID == 'clemero02' | playerID == 'johnswa01') %>%
  mutate(WHIP = (H + BB) / IPouts * 3) %>%
  mutate(SOper9 = SO / IPouts / 3) %>%
  mutate(SOperBB = SO / BB) %>%
  select(yearID, playerID, W, L, IPouts, H, ER, HR, BB, HBP, SO, ERA, WHIP, SOper9, SOperBB)

pitchers %>% filter(playerID == 'maddugr01')
```

```
##    yearID  playerID  W  L IPouts   H  ER HR BB HBP  SO  ERA      WHIP
## 1    1986 maddugr01  2  4     93  44  19  3 11   1  20 5.52 1.7741935
## 2    1987 maddugr01  6 14    467 181  97 17 74   4 101 5.61 1.6381156
```

```
## 3    1988 maddugr01 18  8    747 230  88 13 81   9 140 3.18 1.2489960
## 4    1989 maddugr01 19 12    715 222  78 13 82   6 135 2.95 1.2755245
## 5    1990 maddugr01 15 15    711 242  91 11 71   4 144 3.46 1.3206751
## 6    1991 maddugr01 15 11    789 232  98 18 66   6 198 3.35 1.1330798
## 7    1992 maddugr01 20 11    804 201  65  7 70  14 199 2.18 1.0111940
## 8    1993 maddugr01 20 10    801 228  70 14 52   6 197 2.36 1.0486891
## 9    1994 maddugr01 16  6    606 150  35  4 31   6 156 1.56 0.8960396
## 10   1995 maddugr01 19  2    629 147  38  8 23   4 181 1.63 0.8108108
## 11   1996 maddugr01 15 11    735 225  74 11 28   3 172 2.72 1.0326531
## 12   1997 maddugr01 19  4    698 200  57  9 20   6 177 2.20 0.9455587
## 13   1998 maddugr01 18  9    753 201  62 13 45   7 204 2.22 0.9800797
## 14   1999 maddugr01 19  9    658 258  87 16 37   4 136 3.57 1.3449848
## 15   2000 maddugr01 19  9    748 225  83 19 42  10 190 3.00 1.0708556
## 16   2001 maddugr01 17 11    699 220  79 20 27   7 173 3.05 1.0600858
## 17   2002 maddugr01 16  6    598 194  58 14 45   4 118 2.62 1.1989967
## 18   2003 maddugr01 16 11    655 225  96 24 33   8 124 3.96 1.1816794
## 19   2004 maddugr01 16 11    638 218  95 35 33   9 151 4.02 1.1802508
## 20   2005 maddugr01 13 15    675 239 106 29 36   7 136 4.24 1.2222222
## 21   2006 maddugr01  9 11    409 153  71 14 23   0  81 4.69 1.2909535
## 22   2006 maddugr01  6  3    221  66  27  6 14   0  36 3.30 1.0859729
## 23   2007 maddugr01 14 11    594 221  91 14 25   6 104 4.14 1.2424242
## 24   2008 maddugr01  6  9    460 161  68 16 26   5  80 3.99 1.2195652
## 25   2008 maddugr01  2  4    122  43  23  5  4   1  18 5.09 1.1557377
##        SOper9   SOperBB
## 1  0.07168459 1.818182
## 2  0.07209136 1.364865
## 3  0.06247211 1.728395
## 4  0.06293706 1.646341
## 5  0.06751055 2.028169
## 6  0.08365019 3.000000
## 7  0.08250415 2.842857
## 8  0.08198086 3.788462
## 9  0.08580858 5.032258
## 10 0.09591945 7.869565
## 11 0.07800454 6.142857
## 12 0.08452722 8.850000
## 13 0.09030544 4.533333
## 14 0.06889564 3.675676
## 15 0.08467023 4.523810
## 16 0.08249881 6.407407
## 17 0.06577480 2.622222
## 18 0.06310433 3.757576
## 19 0.07889237 4.575758
## 20 0.06716049 3.777778
## 21 0.06601467 3.521739
## 22 0.05429864 2.571429
## 23 0.05836139 4.160000
## 24 0.05797101 3.076923
## 25 0.04918033 4.500000
```

```r
pitchers %>% filter(playerID == 'clemero02')
```

```
##   yearID  playerID  W  L IPouts   H ER HR  BB HBP  SO  ERA      WHIP
## 1   1984 clemero02  9  4    400 146 64 13  29   2 126 4.32 1.3125000
## 2   1985 clemero02  7  5    295  83 36  5  37   3  74 3.29 1.2203390
```

```
## 3    1986 clemero02 24  4    762 179 70 21  67   4 238 2.48 0.9685039
## 4    1987 clemero02 20  9    845 248 93 19  83   9 256 2.97 1.1751479
## 5    1988 clemero02 18 12    792 217 86 17  62   6 291 2.93 1.0568182
## 6    1989 clemero02 17 11    760 215 88 20  93   8 230 3.13 1.2157895
## 7    1990 clemero02 21  6    685 193 49  7  54   7 209 1.93 1.0817518
## 8    1991 clemero02 18 10    814 219 79 15  65   5 241 2.62 1.0466830
## 9    1992 clemero02 18 11    740 203 66 11  62   9 208 2.41 1.0743243
## 10   1993 clemero02 11 14    575 175 95 17  67  11 160 4.46 1.2626087
## 11   1994 clemero02  9  7    512 124 54 15  71   4 168 2.85 1.1425781
## 12   1995 clemero02 10  5    420 141 65 15  60  14 132 4.18 1.4357143
## 13   1996 clemero02 10 13    728 216 98 19 106   4 257 3.63 1.3269231
## 14   1997 clemero02 21  7    792 204 60  9  68  12 292 2.05 1.0303030
## 15   1998 clemero02 20  6    704 169 69 11  88   7 271 2.65 1.0951705
## 16   1999 clemero02 14 10    563 185 96 20  90   9 163 4.60 1.4653641
## 17   2000 clemero02 13  8    613 184 84 26  84  10 188 3.70 1.3115824
## 18   2001 clemero02 20  3    661 205 86 19  72   5 213 3.51 1.2571861
## 19   2002 clemero02 13  6    540 172 87 18  63   7 192 4.35 1.3055556
## 20   2003 clemero02 17  9    635 199 92 24  58   5 190 3.91 1.2141732
## 21   2004 clemero02 18  4    643 169 71 15  79   6 218 2.98 1.1570762
## 22   2005 clemero02 13  8    634 151 44 11  62   3 185 1.87 1.0078864
## 23   2006 clemero02  7  6    340  89 29  7  29   4 102 2.30 1.0411765
## 24   2007 clemero02  6  6    297  99 46  9  31   5  68 4.18 1.3131313
##        SOper9  SOperBB
## 1  0.10500000 4.344828
## 2  0.08361582 2.000000
## 3  0.10411199 3.552239
## 4  0.10098619 3.084337
## 5  0.12247475 4.693548
## 6  0.10087719 2.473118
## 7  0.10170316 3.870370
## 8  0.09868960 3.707692
## 9  0.09369369 3.354839
## 10 0.09275362 2.388060
## 11 0.10937500 2.366197
## 12 0.10476190 2.200000
## 13 0.11767399 2.424528
## 14 0.12289562 4.294118
## 15 0.12831439 3.079545
## 16 0.09650681 1.811111
## 17 0.10222947 2.238095
## 18 0.10741301 2.958333
## 19 0.11851852 3.047619
## 20 0.09973753 3.275862
## 21 0.11301192 2.759494
## 22 0.09726604 2.983871
## 23 0.10000000 3.517241
## 24 0.07631874 2.193548
```

```
pitchers %>% filter(playerID == 'johnswa01')
```

```
##    yearID  playerID  W  L IPouts   H  ER HR BB HBP  SO  ERA      WHIP
## 1    1907 johnswa01  5  9    331 100  23  1 20   2  71 1.88 1.0876133
## 2    1908 johnswa01 14 14    769 194  47  0 53  11 160 1.65 0.9635891
## 3    1909 johnswa01 13 25    889 247  73  1 84  15 164 2.22 1.1169854
## 4    1910 johnswa01 25 17   1110 262  56  1 76  13 313 1.36 0.9135135
```

```
## 5    1911 johnswa01 25 13    967 292  68  8 70   8 207 1.90 1.1230610
## 6    1912 johnswa01 33 12   1107 259  57  2 76  16 303 1.39 0.9078591
## 7    1913 johnswa01 36  7   1038 232  44  9 38   9 243 1.14 0.7803468
## 8    1914 johnswa01 28 18   1115 287  71  3 74  11 225 1.72 0.9713004
## 9    1915 johnswa01 27 13   1010 258  58  1 56  19 203 1.55 0.9326733
## 10   1916 johnswa01 25 20   1109 290  78  0 82   9 228 1.90 1.0063120
## 11   1917 johnswa01 23 16    978 248  80  3 68  12 188 2.21 0.9693252
## 12   1918 johnswa01 23 13    978 241  46  2 70   8 162 1.27 0.9539877
## 13   1919 johnswa01 20 14    871 235  48  0 51   7 147 1.49 0.9850746
## 14   1920 johnswa01  8 10    431 135  50  5 27   5  78 3.13 1.1276102
## 15   1921 johnswa01 17 14    792 265 103  7 92   2 143 3.51 1.3522727
## 16   1922 johnswa01 15 16    840 283  93  8 99   7 105 2.99 1.3642857
## 17   1923 johnswa01 17 12    784 263 101  9 73  20 130 3.48 1.2857143
## 18   1924 johnswa01 23  7    833 233  84 10 77  10 158 2.72 1.1164466
## 19   1925 johnswa01 20  7    687 217  78  7 78   7 108 3.07 1.2882096
## 20   1926 johnswa01 15 16    782 259 105 13 73   5 125 3.63 1.2736573
## 21   1927 johnswa01  5  6    323 113  61  7 26   7  48 5.10 1.2910217
##          SOper9  SOperBB
## 1  0.07150050 3.550000
## 2  0.06935414 3.018868
## 3  0.06149231 1.952381
## 4  0.09399399 4.118421
## 5  0.07135471 2.957143
## 6  0.09123758 3.986842
## 7  0.07803468 6.394737
## 8  0.06726457 3.040541
## 9  0.06699670 3.625000
## 10 0.06853021 2.780488
## 11 0.06407635 2.764706
## 12 0.05521472 2.314286
## 13 0.05625718 2.882353
## 14 0.06032483 2.888889
## 15 0.06018519 1.554348
## 16 0.04166667 1.060606
## 17 0.05527211 1.780822
## 18 0.06322529 2.051948
## 19 0.05240175 1.384615
## 20 0.05328218 1.712329
## 21 0.04953560 1.846154
```

- Create career stat lines for each of the players that you selected. Be careful about how these statistics are calculated.

```r
career_batters <- Batting %>%
  filter(playerID == "bondsba01" | playerID == "thomafr04" | playerID == "schmimi01") %>%
  group_by(playerID) %>%
  summarise(totalG = sum(G), totalAB = sum(AB), totalR = sum(R), totalH = sum(H),
            totalX1B = sum(H - X2B - X3B - HR),
            totalX2B = sum(X2B), totalX3B = sum(X3B), totalHR = sum(HR),
            totalRBI = sum(RBI), totalSB = sum(SB), totalCS = sum(CS),
            totalBB = sum(BB), totalSO = sum(SO), SBpct = totalSB / (totalSB + totalSB),
            totalSF = sum(SF), totalHBP = sum(HBP),
            totalOBP = (totalH + totalBB + totalHBP) / (totalAB + totalBB + totalHBP + totalSF), totalSl
            totalOPS = totalOBP + totalSLG)
career_batters
```

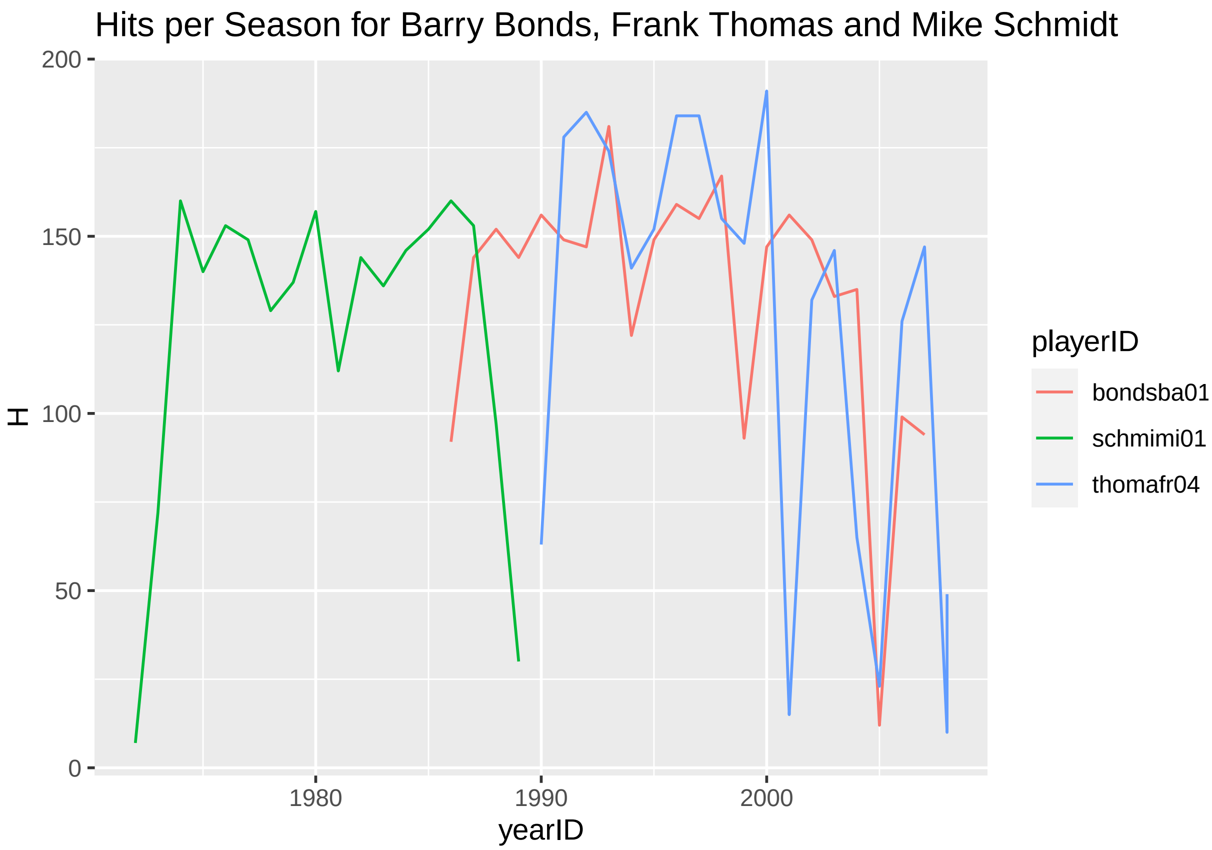
```
## # A tibble: 3 x 20
##   playerID  totalG totalAB totalR totalH total~1 total~2 total~3 totalHR total~4
##   <chr>      <int>   <int>  <int>  <int>   <int>   <int>   <int>   <int>   <int>
## 1 bondsba01   2986    9847   2227   2935    1495     601      77     762    1996
## 2 schmimi01   2404    8352   1506   2234    1219     408      59     548    1595
## 3 thomafr04   2322    8199   1494   2468    1440     495      12     521    1704
## # ... with 10 more variables: totalSB <int>, totalCS <int>, totalBB <int>,
## #   totalSO <int>, SBpct <dbl>, totalSF <int>, totalHBP <int>, totalOBP <dbl>,
## #   totalSLG <dbl>, totalOPS <dbl>, and abbreviated variable names 1: totalX1B,
## #   2: totalX2B, 3: totalX3B, 4: totalRBI
```

```
career_pitchers <- Pitching %>%
  filter(playerID == 'maddugr01' | playerID == 'clemero02' | playerID == 'johnswa01') %>%
  group_by(playerID) %>%
  summarise(totalG = sum(G), totalW = sum(W), totalL = sum(L), totalIPouts = sum(IPouts),
            totalER = sum(ER), totalHR = sum(HR), totalBB = sum(BB), totalHBP = sum(HBP),
            totalSO = sum(SO), totalERA = sum(ER)*27/sum(IPouts),
            totalWHIP = (sum(H) + sum(BB)) *3/sum(IPouts),
            totalSOper9 = 27 * sum(SO)/sum(IPouts), totalSOperBB = sum(SO)/sum(BB))
career_pitchers
```

```
## # A tibble: 3 x 14
##   playerID  totalG totalW totalL total~1 totalER totalHR totalBB total~2 totalSO
##   <chr>      <int>  <int>  <int>   <int>   <int>   <int>   <int>   <int>   <int>
## 1 clemero02    709    354    184   14750    1707     363    1580     159    4672
## 2 johnswa01    802    417    279   17744    1424      97    1363     203    3509
## 3 maddugr01    744    355    227   15025    1756     353     999     137    3371
## # ... with 4 more variables: totalERA <dbl>, totalWHIP <dbl>,
## #   totalSOper9 <dbl>, totalSOperBB <dbl>, and abbreviated variable names
## #   1: totalIPouts, 2: totalHBP
```

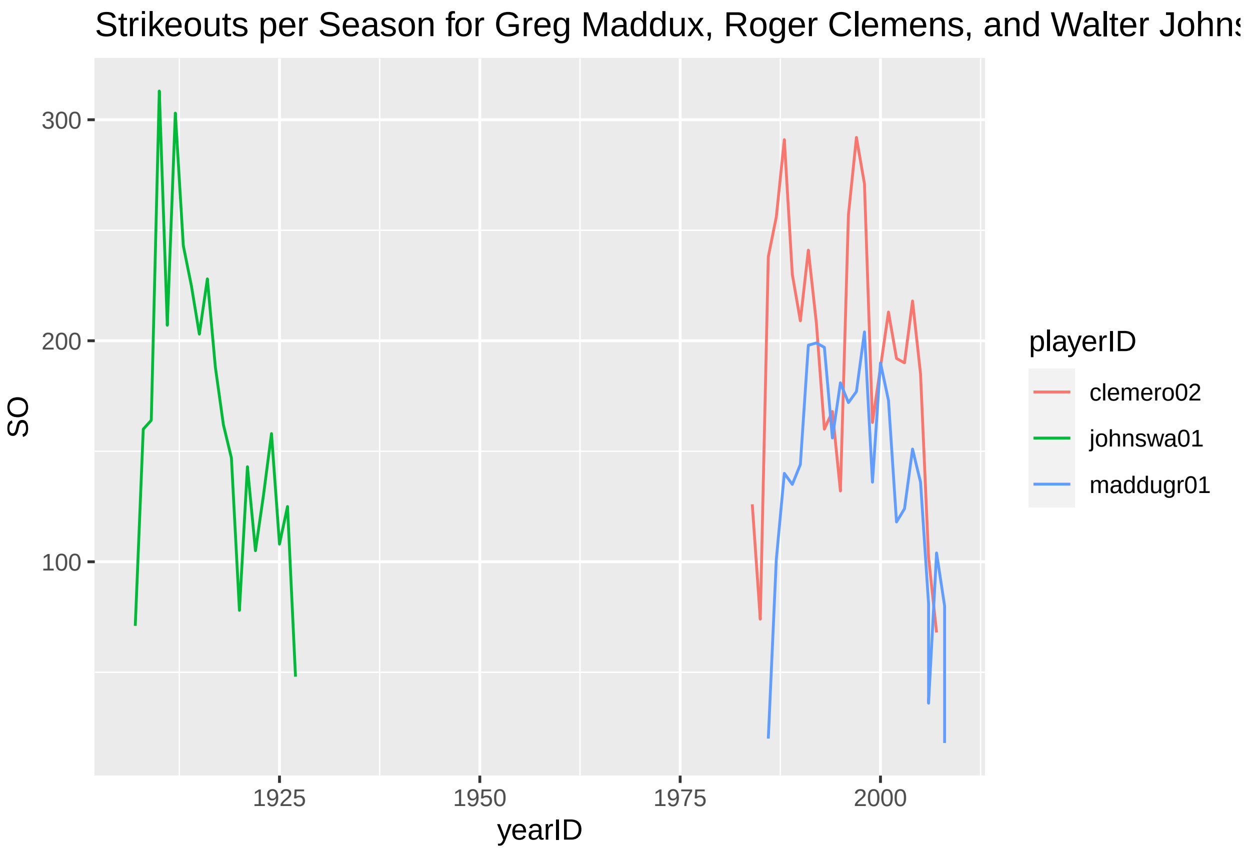
- Provide a plot for career trajectories for one batting and one pitching statistic of your choice. These are two separate graphics, one for the batters and one for the pitchers. The graphics that you produce should display the trajectories of the 3 batters and the 3 pitchers. Provide interesting commentary on your graphic.

```
ggplot(batters, aes(yearID, H, colour = playerID)) + geom_line() + ggtitle("Hits per Season for Barry Bo
```



Mike Schmidt and Barry Bonds show their consistency in hits.

```
ggplot(pitchers, aes(yearID, SO, colour = playerID)) + geom_line() + ggtitle("Strikeouts per Season for
```



Roger Clemens and Greg Maddux both are great pitchers, but clearly Roger Clemens is more powerful in strikeouts than Greg Maddux.

Question 3 Problem 2 on page 28 of Analyzing Baseball Data with R

Solution

```
## a)

Bob_Gibson <- Pitching %>% filter(yearID==1968, playerID=='gibsobo01')

Bob_Gibson$CG / Bob_Gibson$GS
```

```
## [1] 0.8235294
```

```
## b)
Bob_Gibson$SO / Bob_Gibson$BB
```

```
## [1] 4.322581
```

```
## c)
Bob_Gibson$IPouts / 3
```

```
## [1] 304.6667
```

```
## d)

(Bob_Gibson$H + Bob_Gibson$BB) / Bob_Gibson$IPouts *3
```

```
## [1] 0.8533917
```

Question 4 Problem 3 on page 29 of Analyzing Baseball Data with R

```r
library(retrosheet)
data_4 <- getRetrosheet("game", 1964) %>% filter(Date == 19640621, VisTm == 'PHI', DblHdr == 1)

# a)
data_4$Duration
```

```
## [1] 139
```

```r
## Hour
floor(data_4$Duration / 60)
```

```
## [1] 2
```

```r
## minute
data_4$Duration %% 60
```

```
## [1] 19
```

b) The attendance is 0 likely due to the doubleheader being played that day.

```r
## c)
data_4$VisD + data_4$VisT + data_4$VisHR
```

```
## [1] 3
```

```r
## d)
OBP = (data_4$VisH + data_4$VisBB + data_4$VisHBP)/(data_4$VisAB + data_4$VisBB + data_4$VisHBP + data_4
OBP
```

```
## [1] 0.3333333
```