STAT 8700 Final Question 4

Brian Detweiler

Thursday, December 15th

4. The English Premier League (EPL) is the premier soccer (football to the rest of the word) competition in England. 20 teams play in the competition. Each team plays every other team both at home and on the road (or, to use English terminology, away), for a total of 38 games per team, and 380 total games. Each season runs from August to May. So far in the 2016/17 season, there have been 140 games played, with each team playing 14 games. The epl.csv contains the score data from the 140 games played. The data set contains 6 columns: the first column is the name of the home team, the second column contains an identifier for the home team (a number between 1 and 20, see table below), the third column is the number of goals scored by the home team in that game. Columns 4, 5, and 6, mirror the information in columns 1, 2, and 3, but for the away team.

Team ID	Team
1	Arsenal
2	Burnley
3	Bournemouth
4	Chelsea
5	Crystal Palace
6	Everton
7	Hull City
8	Leicester City
9	Liverpool
10	Manchester City
11	Manchester United
12	Middlesbrough
13	Southampton
14	Stoke City
15	Sunderland
16	Swansea City
17	Tottenham Hotspur
18	Watford
19	West Bromwich Albion
20	West Ham United

We would like to fit a model to the data in order to analyze the quality of each team. Since goals scored can be thought of as counts, a Poisson regression model may be suitable.

Let H_{ij} be the number of goals scored by the home team (team i) in a game between team i and team j played at team i's home. Likewise, let A_{ij} be the number of goals scored by the away team (team j) in the same game.

Now suppose that

$$H_{ij} \sim Poisson(\lambda_{ij})$$
$$A_{ij} \sim Poisson(\theta_{ij})$$

where

$$log(\lambda_{ij}) = \mu + \alpha_i + \delta_j + \gamma$$
$$log(\theta_{ij}) = \mu + alpha_j + \delta_i$$

Interpreting the paramters, μ is a measure of the average goal scoring rate, α_i and α_j are measures of the attacking ability of teams i and j respectively, δ_i and δ_j are measures of the defensive ability of teams i and j respectively, and γ is a measure of "home-feld advantage".

Since the α's and δ's measure each team's attacking and defensive ability relative to the average, we also need the following restrictions:

$$\sum_{k=1}^{20} \alpha_k = 0, \sum_{k=1}^{20} \delta_k = 0$$

(a) Load the data from the file, using `read.csv("epl.csv", header=T)`. Write a JAGS model file for the above model, and use it to simulate values from the posterior distribution of the parameters.

Sweedish researcher Rasmus Bååth did a similar analysis on La Liga teams. We'll borrow some of his ideas here. [1]

For our priors, we will use fairly vague beliefs by

$$
\begin{aligned}
\mu &\sim Normal(0, 4^2) \\
\alpha_i &\sim Normal(Group_\alpha, Group_\sigma^2) \\
\delta_i &\sim Normal(Group_\delta, Group_\sigma^2) \\
Group_\alpha &\sim Normal(0, 4^2) \\
Group_\delta &\sim Normal(0, 4^2) \\
Group_{sigma}\ dunif(0, 3)
\end{aligned}
$$

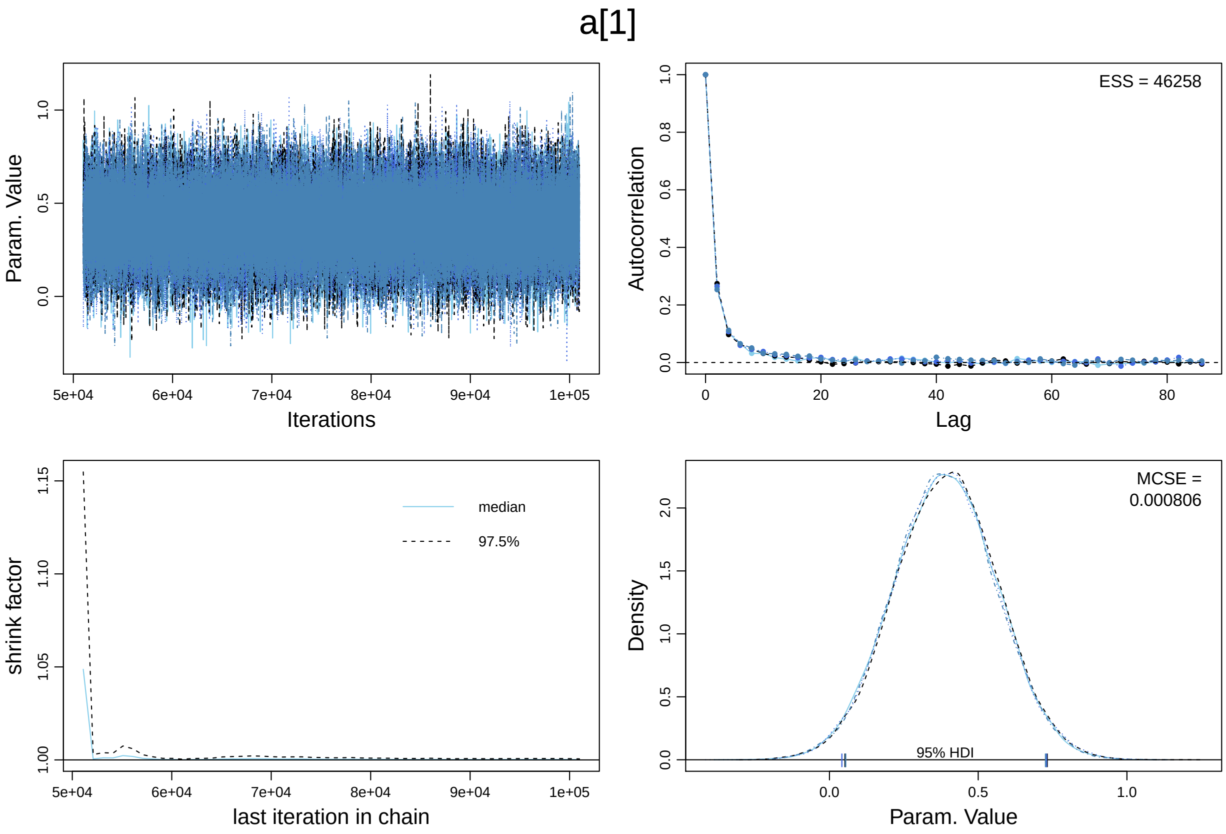
We can assume there is a non-zero homefield advantage, but that it is likely not more significant than the skill of the teams involved. For this reason, we'll use a Gamma prior that favors smaller numbers.

$$\gamma \sim Gamma(0.01, 0.01)$$

```
# -1 Away win, 0 tie, 1 Home win
epl$Result <- sign(epl$Home.Goals - epl$Away.Goals)

teams <- unique(c(epl$Home.Team, epl$Away.Team))

mean.goals <- mean(c(epl$Home.Goals, epl$Away.Goals))
mean.home.goals <- mean(c(epl$Home.Goals))
sd.goals <- sd(c(epl$Home.Goals, epl$Away.Goals))

epl.M <- matrix(data = NA, nrow = 20, ncol = 20)

for (i in 1:length(epl[,1])) {
  epl.M[epl$Home.ID[i], epl$Away.ID[i]] <- epl$Home.Goals[i]
  epl.M[epl$Away.ID[i], epl$Home.ID[i]] <- epl$Away.Goals[i]
}

a <- c(rep(NA, 20))
d <- c(rep(NA, 20))
for (i in 1:20) {
  epl.mean <- epl %>% filter(Home.ID == i) %>%
    group_by(Home.ID) %>%
    summarise(a_mean = mean(Home.Goals), d_mean = mean(Away.Goals))

  a[i] <- epl.mean$a_mean - mean.goals
  d[i] <- mean.goals - epl.mean$d_mean
}

gam <- sum(a) - sum(d)

fileName <- "Final.4.a"

modelString ="
model {
  for(i in 1:120) {
    H[i] ~ dpois(lambda[HomeTeam[i], AwayTeam[i]])
    A[i] ~ dpois(theta[HomeTeam[i], AwayTeam[i]])
  }
  for(i in 1:n_teams) {
    for(j in 1:n_teams) {
      lambda[i, j] <- exp(mu + a[i] - d[j] + gamma)
      theta[i, j] <- exp(mu + a[j] - d[i])
    }
  }

  for(j in 1:(n_teams - 1)) {
    a[j] ~ dnorm(group_attack, group_tau)
    d[j] ~ dnorm(group_defense, group_tau)
  }
  a[n_teams] <- -sum(a[1:(n_teams - 1)])
  d[n_teams] <- -sum(d[1:(n_teams - 1)])

  gamma ~ dgamma(0.01, 0.01)

  rank.a <- rank(a)
  rank.d <- rank(d)

  for (k in 1:n_teams) {
    for (m in 1:n_teams) {
      #ranks.a[k, m] ~ dbern(pa[k, m])
      #ranks.d[k, m] ~ dbern(pd[k, m])
      #pa[k, m] <- equals(k, rank.a[m])
      #pd[k, m] <- equals(k, rank.d[m])
      ranks.a[k, m] <- equals(k, rank.a[m])
      ranks.d[k, m] <- equals(k, rank.d[m])
    }
  }

  mu ~ dnorm(0, 0.0625)

  group_attack ~ dnorm(0, 0.0625)
  group_defense ~ dnorm(0, 0.0625)
  group_tau <- 1 / pow(group_sigma, 2)
  group_sigma ~ dunif(0, 3)
}
"


writeLines(modelString, con=fileName)

data.list <- list(H = epl$Home.Goals,
                  A = epl$Away.Goals,
                  HomeTeam = epl$Home.ID,
                  AwayTeam = epl$Away.ID,
                  n_teams = length(teams))


epl.model = jags.model(file=fileName,
                       data=data.list,
                       n.chains=4)
```

```
## Compiling model graph
##    Resolving undeclared variables
##    Allocating nodes
## Graph information:
##    Observed stochastic nodes: 240
##    Unobserved stochastic nodes: 43
##    Total graph size: 6288
##
## Initializing model
```

```
update(epl.model, n.iter=50000)

epl.samples <- coda.samples(epl.model,
                            variable.names = c("a",
                                               "d",
                                               "gamma",
                                               "rank.a",
                                               "rank.d",
                                               "ranks.a",
                                               "ranks.d"),
                            n.iter = 50000,
                            thin = 2)

diagMCMC(epl.samples)

epl.dic <- dic.samples(model = epl.model, n.iter = 20000, thin = 50)
```



```
epl.dic
```

```
## Mean deviance:  692.9
## penalty 27.22
## Penalized deviance: 720.1
```

```
epl.samples.M <- as.matrix(epl.samples)

smry <- summary(epl.samples)
```

■

(b) Using the posterior means as a judge, which is the best offensive team? Which is the worst offensive team? Which is the best defensive team? Which is the worst defensive team?

```
a.means <- smry$quantiles[1:20, 3]
d.means <- smry$quantiles[21:40, 3]
gamma.means <- smry$quantiles[41, 3]
rank.a.means <- smry$quantiles[42:61, 3]
rank.d.means <- smry$quantiles[62:81, 3]

# sort(a.means)
# teams.with.names$Home.Team[teams.with.names$Home.ID == 9]

best.a <- which(a.means == max(a.means))
worst.a <- which(a.means == min(a.means))

best.d <- which(d.means == max(d.means))
worst.d <- which(d.means == min(d.means))

best.a.team.name <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == best.a])
best.d.team.name <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == best.d])

worst.a.team.name <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == worst.a])
worst.d.team.name <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == worst.d])
```

The best offensive team is Liverpool, and the worst offensive team is Hull City.

The best defensive team is Tottenham Hotspur, and the worst defensive team is West Ham United.

■

(c) Rather than using the posterior means to evaluate the offensive and defensive quality of the teams, we could look at where each team's α and δ values rank compared to all the other teams. In JAGS, the rank function takes a vector and returns another vector whose entries are the size ranks of the the elements of the input vector, with 1 indicating that this was the smallest element, 2 indicating the next smallest, and so on.

In your JAGS model, add a line defining a vector of parameters called rank.d so that the kth element of `rank.d`, `rank.d[k]` is the rank of team k's defensive strength compared to all other teams, with 1 indicating they were the best defensive team (δ_k is the smallest of all δ's), and 20 indicating that they were the worst defensive team (δ_k is the biggest of all δ's). Using the posterior mean ranks for each team, which is the best defensive team? Which is the worst defensive team?

```r
best.rank <- which(rank.d.means == max(rank.d.means))
worst.rank <- which(rank.d.means == min(rank.d.means))

best.team.name.rank1 <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == best.rank[[
best.team.name.rank2 <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == best.rank[[
worst.team.name.rank <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == worst.rank[
```

The best defensive team by rank is a tie between Chelsea and Tottenham Hotspur, and the worst defensive team by rank is West Ham United.

■

(d) In your JAGS model, add a line defining a vector of parameters called `rank.a` so that the kth element of `rank.a`, `rank.a[k]` is the rank of team k's offensive strength compared to all other teams, with 1 indicating they were the best offensive team (α_k is the biggest of all α's), and 20 indicating that they were the worst offensive team (α_k is the smallest of all α's). Using the posterior mean ranks for each team, which is the best offensive team? Which is the worst offensive team?

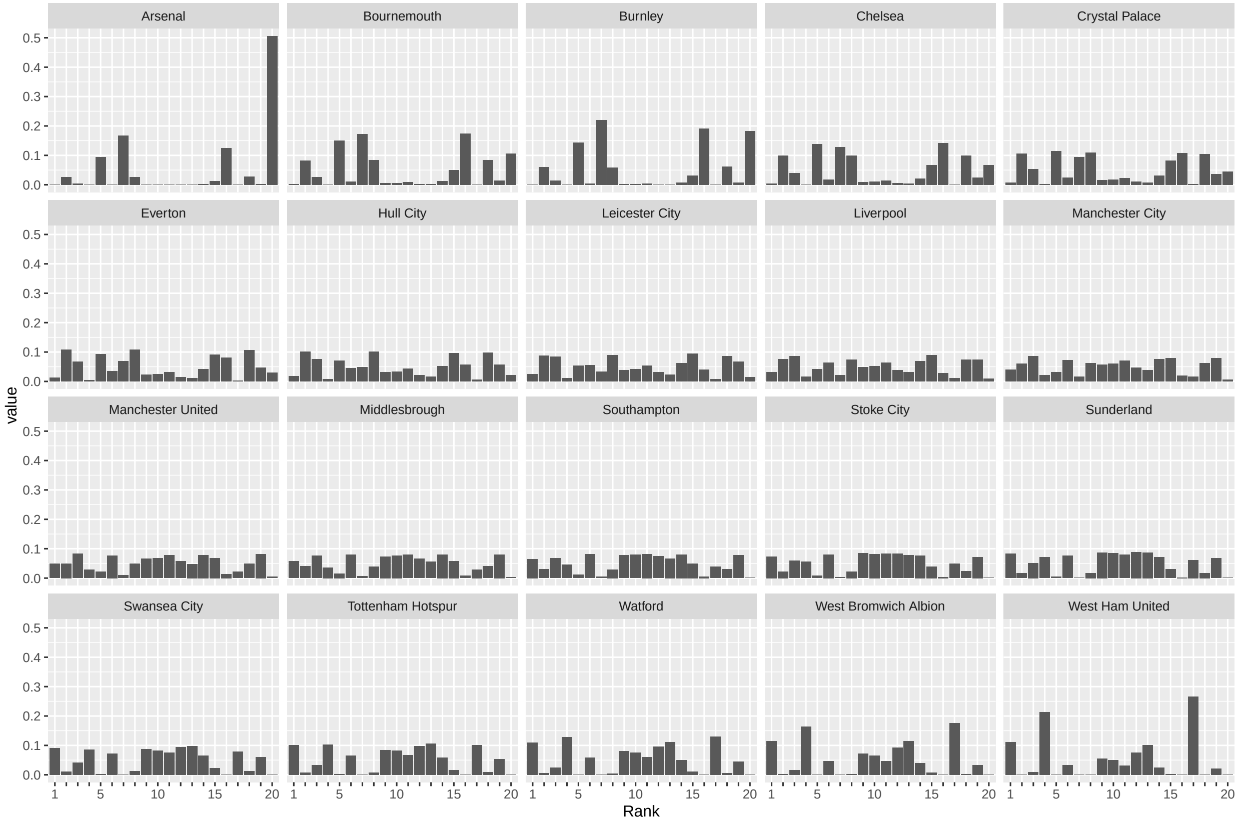
```r
best.rank <- which(rank.a.means == max(rank.a.means))
worst.rank <- which(rank.a.means == min(rank.a.means))

best.team.name.rank1 <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == best.rank[[
worst.team.name.rank <- as.character(teams.with.names$Home.Team[teams.with.names$Home.ID == worst.rank[
```

The best offensive team by rank is a tie between Liverpool, and the worst offensive team by rank is Hull City.

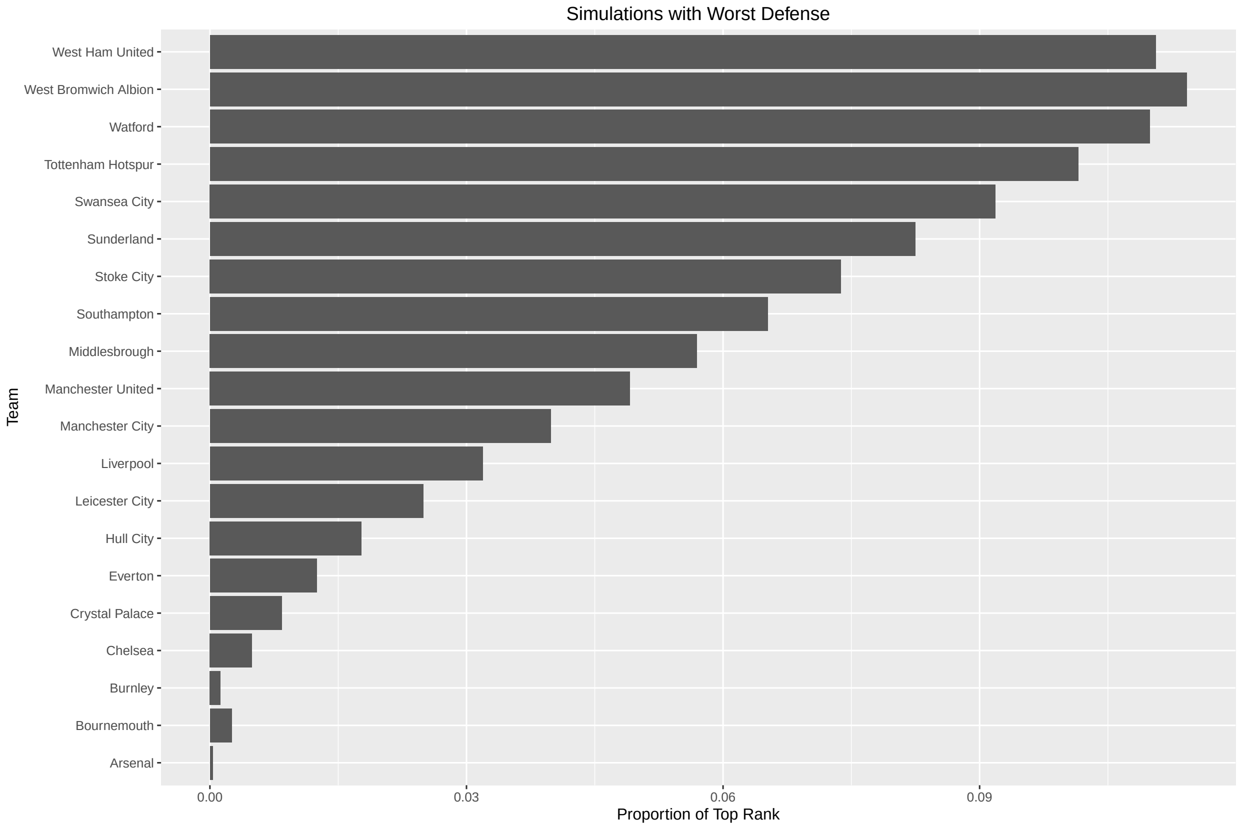
■

(e) Yet another way to evaluate the teams, would be to see in what proportion of simulations were the team's parameters ranked in a particular spot. In your JAGS model, add code to define `ranks.d[k,m]` which takes the value 1 is the defensive rank of team k is equal to m, and 0 otherwise (hint, check out the equals function in JAGS). This should be done for all teams and all possible ranks. With `ranks.d[k,m]` defined this way, the posterior mean of `ranks.d[k,m]` would be equal to the proportion of simulations in which team k's defensive rank was equal to m. Which team had the highest proportion of rank 1 for the defensive parameter? Which team had the highest proportion of rank 20 for the defensive parameter?

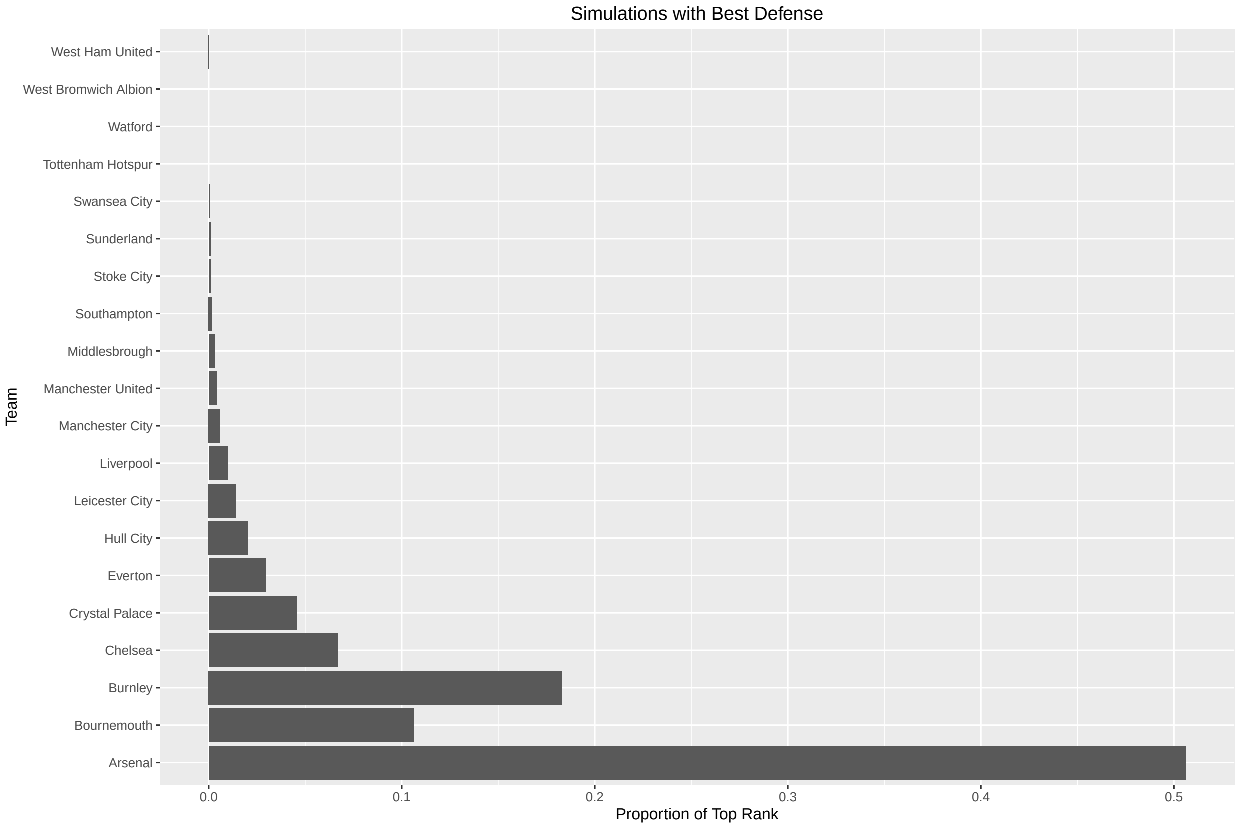
```r
ranks.df <- data.frame(teams.with.names)
ranks.df$rank1 <- 0
ranks.df$rank2 <- 0
ranks.df$rank3 <- 0
ranks.df$rank4 <- 0
ranks.df$rank5 <- 0
ranks.df$rank6 <- 0
ranks.df$rank7 <- 0
ranks.df$rank8 <- 0
ranks.df$rank9 <- 0
ranks.df$rank10 <- 0
ranks.df$rank11 <- 0
ranks.df$rank12 <- 0
ranks.df$rank13 <- 0
ranks.df$rank14 <- 0
ranks.df$rank15 <- 0
ranks.df$rank16 <- 0
ranks.df$rank17 <- 0
ranks.df$rank18 <- 0
ranks.df$rank19 <- 0
ranks.df$rank20 <- 0

for (i in 1:20) {
  for (j in 1:20) {
    ranks.df[i, paste0("rank", j)] <- mean(epl.samples.M[,paste0("ranks.d[", i, ",", j, "]")])
  }
}

ranks.sub.df <- ranks.df[,-1]

ranks.melt <- melt(ranks.sub.df, id.vars = "Home.Team")
ggplot(ranks.melt, aes(x=variable, y=value)) +
  geom_bar(stat="identity") +
  facet_wrap(~Home.Team) +
  scale_x_discrete("Rank", labels = c("1", "", "", "",
                                      "5", "", "", "", "",
                                      "10", "", "", "", "",
                                      "15", "", "", "", "",
                                      "20"))
```



```
ranks.sub.df.rank1 <- ranks.sub.df %>% select(Home.Team, rank1) %>% arrange(-rank1)
ranks.sub.df.rank20 <- ranks.sub.df %>% select(Home.Team, rank20) %>% arrange(-rank20)

worst.d.team.name <- as.character(ranks.sub.df.rank1[1, 1])
best.d.team.name <- as.character(ranks.sub.df.rank20[1, 1])

ggplot(ranks.sub.df.rank1, aes(x=Home.Team, y=rank1)) +
  geom_bar(stat="identity") +
  coord_flip() +
  labs(title="Simulations with Worst Defense", x="Team", y="Proportion of Top Rank")
```



```
ggplot(ranks.sub.df.rank20, aes(x=Home.Team, y=rank20)) +
  geom_bar(stat="identity") +
  coord_flip() +
  labs(title="Simulations with Best Defense", x="Team", y="Proportion of Top Rank")
```



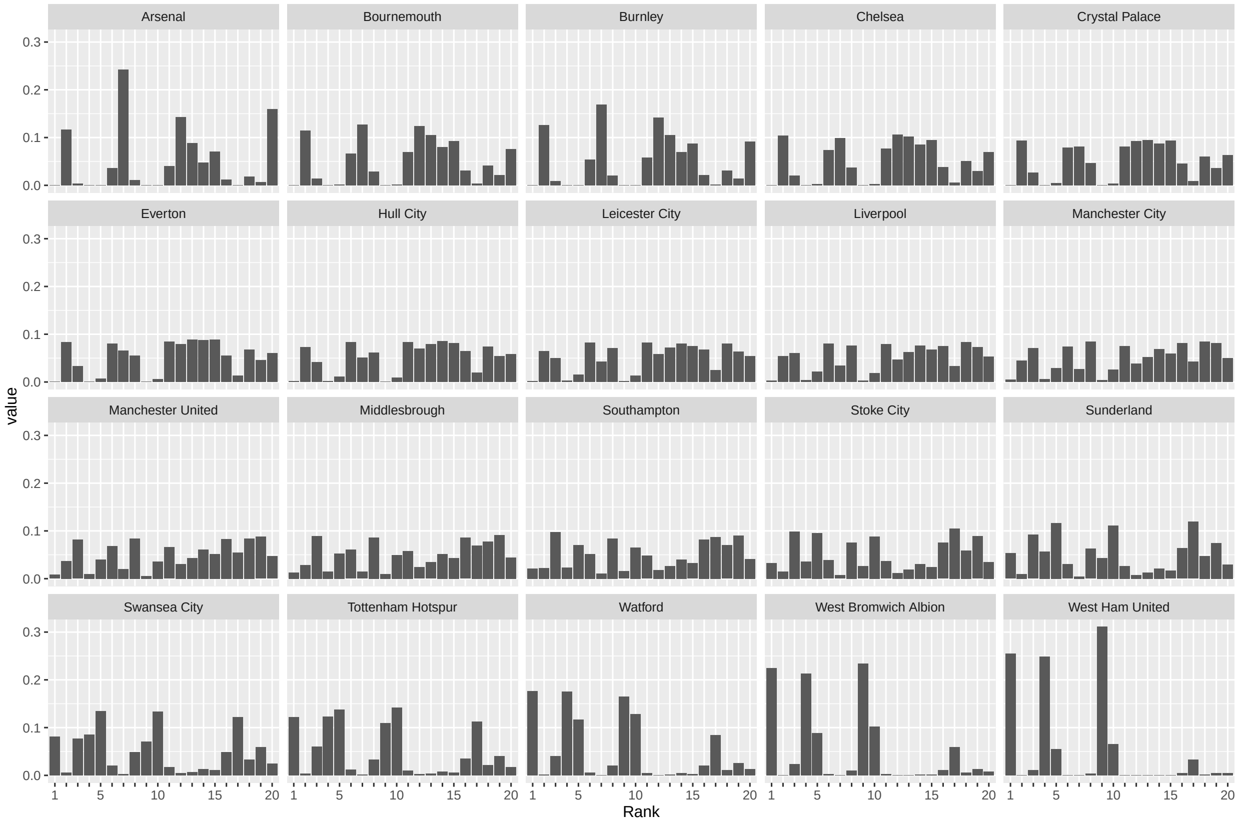
The best and worst teams on defensive respectively are Arsenal and West Bromwich Albion.

■

(f) Repeat part (e) for the offensive parameter, defining `ranks.a[k.m]`.

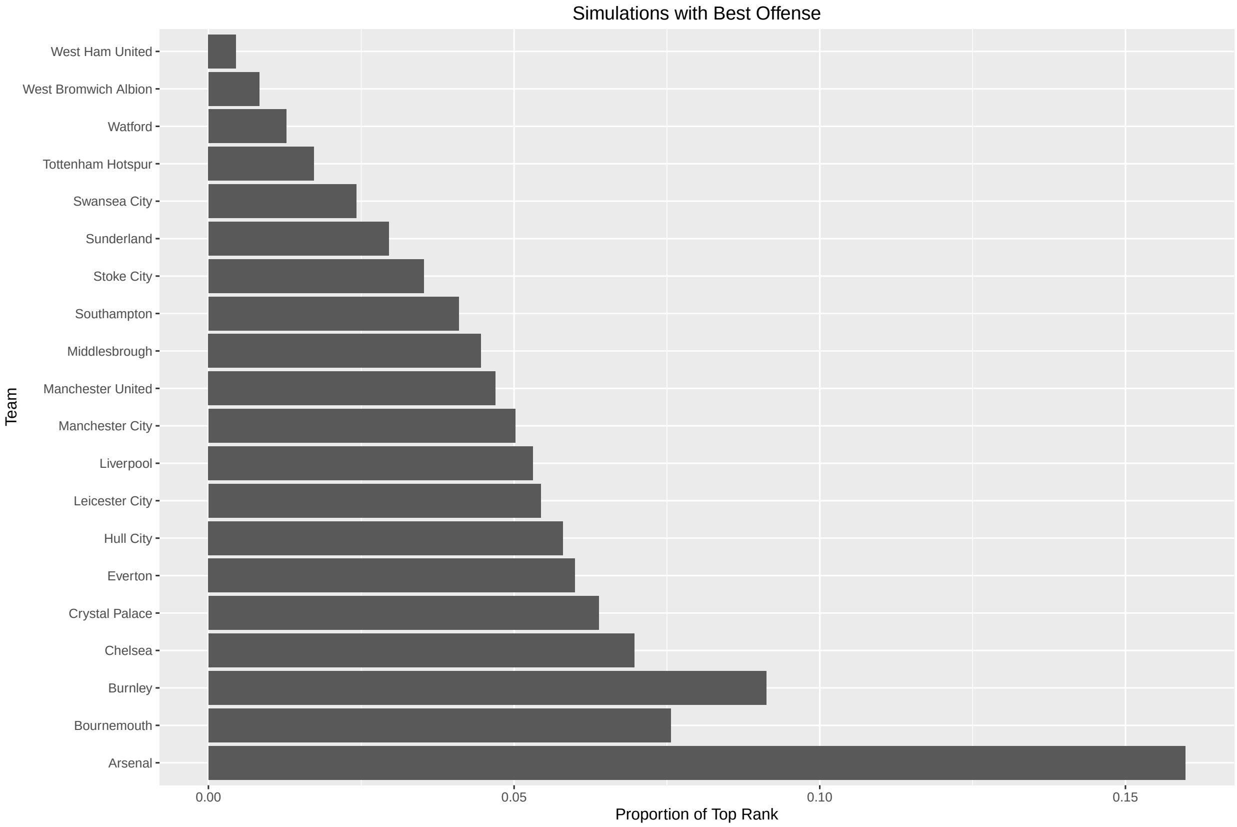
Show all working.

```r
ranks.df <- data.frame(teams.with.names)
ranks.df$rank1 <- 0
ranks.df$rank2 <- 0
ranks.df$rank3 <- 0
ranks.df$rank4 <- 0
ranks.df$rank5 <- 0
ranks.df$rank6 <- 0
ranks.df$rank7 <- 0
ranks.df$rank8 <- 0
ranks.df$rank9 <- 0
ranks.df$rank10 <- 0
ranks.df$rank11 <- 0
ranks.df$rank12 <- 0
ranks.df$rank13 <- 0
ranks.df$rank14 <- 0
ranks.df$rank15 <- 0
ranks.df$rank16 <- 0
ranks.df$rank17 <- 0
ranks.df$rank18 <- 0
ranks.df$rank19 <- 0
ranks.df$rank20 <- 0

for (i in 1:20) {
  for (j in 1:20) {
    ranks.df[i, paste0("rank", j)] <- mean(epl.samples.M[,paste0("ranks.a[", i, ",", j, "]")])
  }
}

ranks.sub.df <- ranks.df[,-1]

ranks.melt <- melt(ranks.sub.df, id.vars = "Home.Team")
ggplot(ranks.melt, aes(x=variable, y=value)) +
  geom_bar(stat="identity") +
  facet_wrap(~Home.Team) +
  scale_x_discrete("Rank", labels = c("1", "", "", "",
                                      "5", "", "", "", "",
                                      "10", "", "", "", "",
                                      "15", "", "", "", "",
                                      "20"))
```

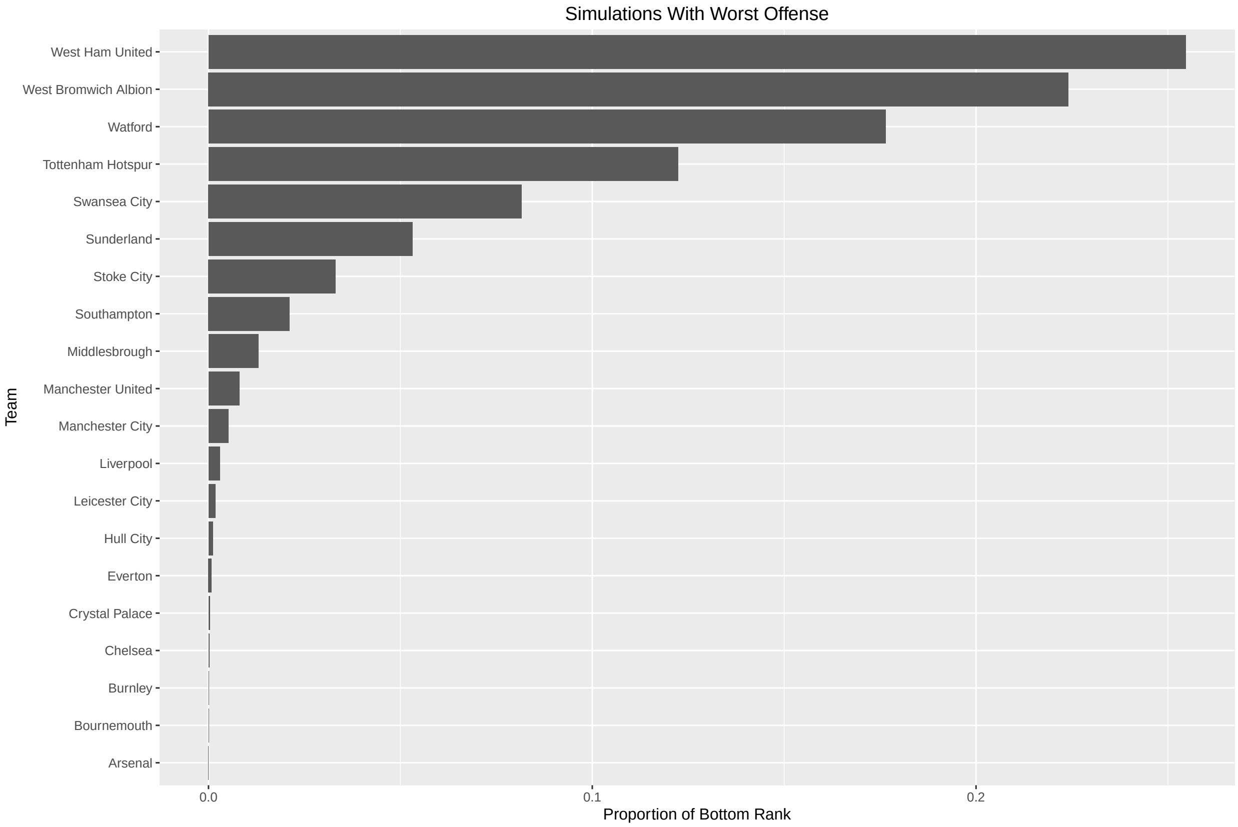


```
ranks.sub.df.rank1 <- ranks.sub.df %>% select(Home.Team, rank1) %>% arrange(desc(rank1))
ranks.sub.df.rank20 <- ranks.sub.df %>% select(Home.Team, rank20) %>% arrange(desc(rank20))
```

```
best.a.team.name <- as.character(ranks.sub.df.rank20[1, 1])
worst.a.team.name <- as.character(ranks.sub.df.rank1[1, 1])

ggplot(ranks.sub.df.rank20, aes(x=Home.Team, y=rank20)) +
  geom_bar(stat="identity") +
  coord_flip() +
  labs(title="Simulations with Best Offense", x="Team", y="Proportion of Top Rank")
```



```
ggplot(ranks.sub.df.rank1, aes(x=Home.Team, y=rank1)) +
  geom_bar(stat="identity") +
  coord_flip() +
  labs(title="Simulations With Worst Offense", x="Team", y="Proportion of Bottom Rank")
```



The best and worst teams on offense respectively are Arsenal and West Ham United.

■

References

[1] Rasmus Bååth *Modeling Match Results in La Liga Using a Hierarchical Bayesian Poisson Model: Part one*
Publishable Stuff, July 21, 2013 http://www.sumsar.net/blog/2013/07/modeling-match-results-in-la-liga-part-one/