

# **Supplementary material for: Combining historical data and bookmakers' odds in modelling football scores**

**Leonardo Egidi <sup>1</sup>, Francesco Pauli <sup>1</sup>, and Nicola Torelli <sup>1</sup>**

<sup>1</sup> Dipartimento di Scienze Economiche, Aziendali, Matematiche e Statistiche 'Bruno de Finetti', Università degli Studi di Trieste, Via Tigor 22 - 34124 Trieste, Italy

---

**Address for correspondence:** Leonardo Egidi, Dipartimento di Scienze Economiche, Aziendali, Matematiche e Statistiche 'Bruno de Finetti', Università degli Studi di Trieste, Via Tigor 22 - 34124 Trieste, Italy.

**E-mail:** `legidi@units.it`.

**Phone:** (+39) 040 558 3041.

**Fax:** (+39) 040 558 7005.

---

**Abstract:**

---

**Key words:**

## Contents

|          |   |           |
|----------|---|-----------|
| <b>1</b> | <b>PP checks</b>  | <b>2</b>  |
| <b>2</b> | <b>Checking model assumptions</b>                             | <b>6</b>  |
| 2.1      | Conditional independence . . . . .                            | 6         |
| 2.2      | Overdispersion . . . . .                                      | 6         |
| <b>3</b> | <b>Code</b>   | <b>12</b> |
| 3.1      | Reading data . . . . .  | 12        |
| 3.2      | Transforming betting odds in implicit scoring rates . . . . . | 14        |
| 3.3      | JAGS code . . . . .   | 17        |

## 1 PP checks

Analogously to Figure 4 of the paper with the English Premier League, in this Section we include the PP checks of Section 4.3 for the following leagues: German Bundesliga, Spanish La Liga and Italian Serie A (Figures [1](#), [2](#), [3](#)).

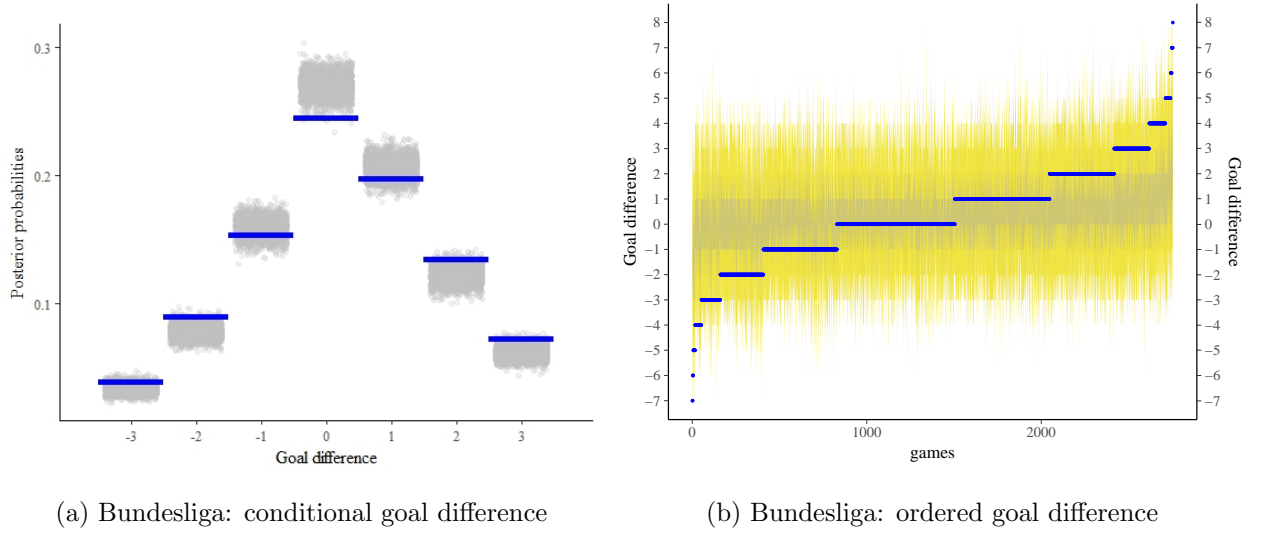


Figure 1: PP checks for the goal difference  $y_1 - y_2$  against the replicated goal difference  $y_1^{rep} - y_2^{rep}$  for German Bundesliga. Panel (a): goal differences distribution (gray areas) conditioned on the observed probability (blue segments). Panel (b): 95% posterior intervals (lightyellow) and 50% posterior intervals (dark yellow) for estimated goal difference  $y_1^{rep} - y_2^{rep}$  in Bundesliga. Blue points are the ordered observed goal differences.

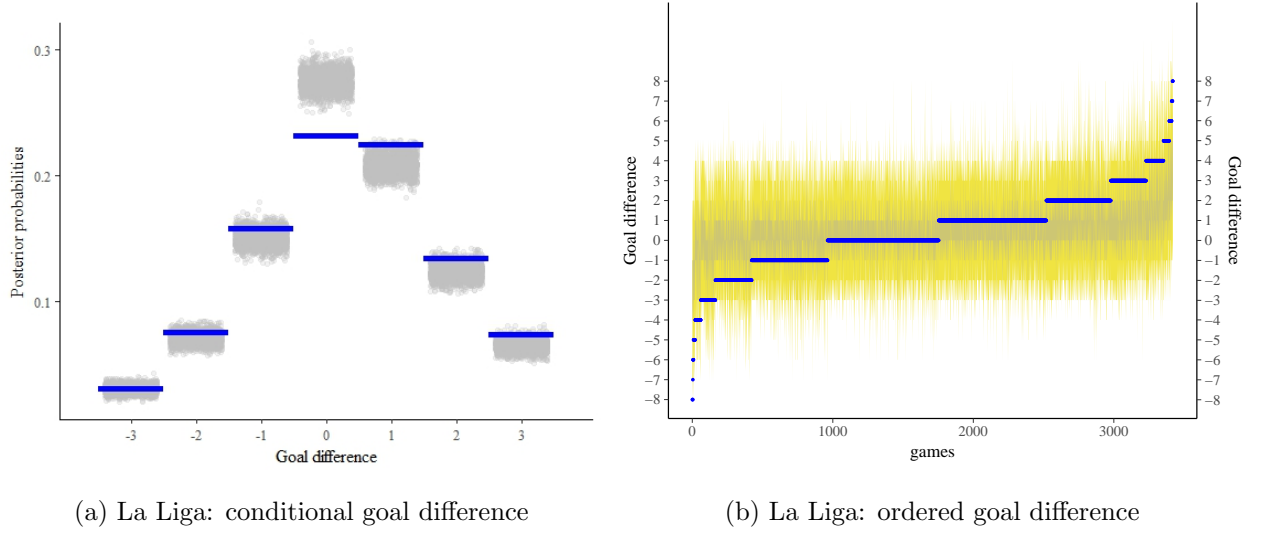


Figure 2: PP checks for the goal difference  $y_1 - y_2$  against the replicated goal difference  $y_1^{rep} - y_2^{rep}$  for Spanish La Liga. Panel (a): goal differences distribution (gray areas) conditioned on the observed probability (blue segments). Panel (b): 95% posterior intervals (lightyellow) and 50% posterior intervals (dark yellow) for estimated goal difference  $y_1^{rep} - y_2^{rep}$  in La Liga. Blue points are the ordered observed goal differences.

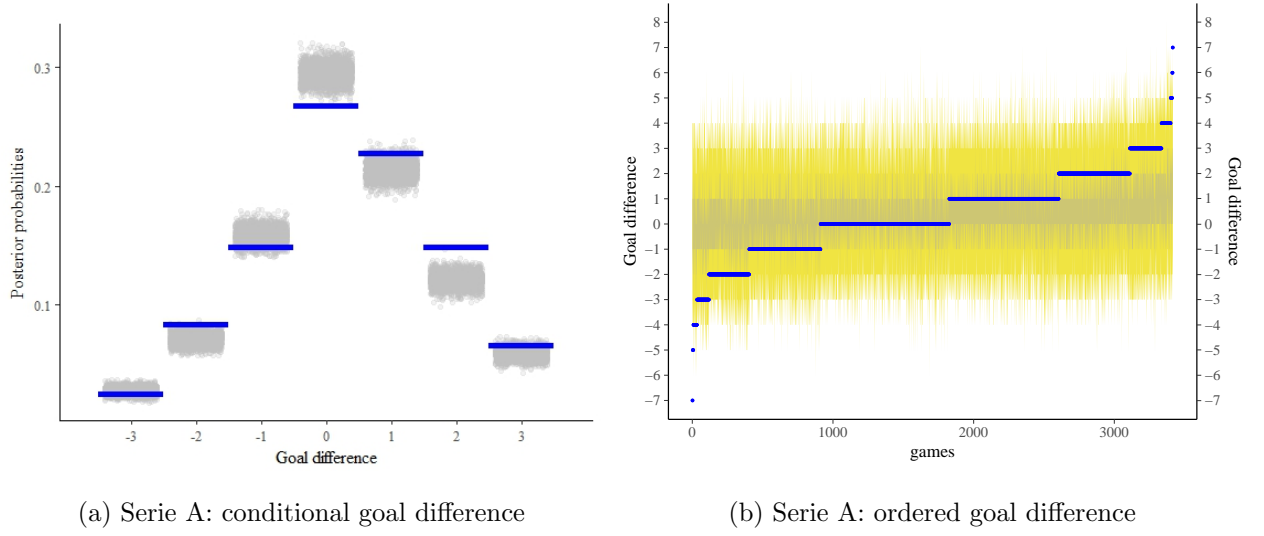


Figure 3: PP checks for the goal difference  $y_1 - y_2$  against the replicated goal difference  $y_1^{rep} - y_2^{rep}$  for Italian Serie A. Panel (a): goal differences distribution (gray areas) conditioned on the observed probability (blue segments). Panel (b): 95% posterior intervals (lightyellow) and 50% posterior intervals (dark yellow) for estimated goal difference  $y_1^{rep} - y_2^{rep}$  in Serie A. Blue points are the ordered observed goal differences.

## 2 Checking model assumptions

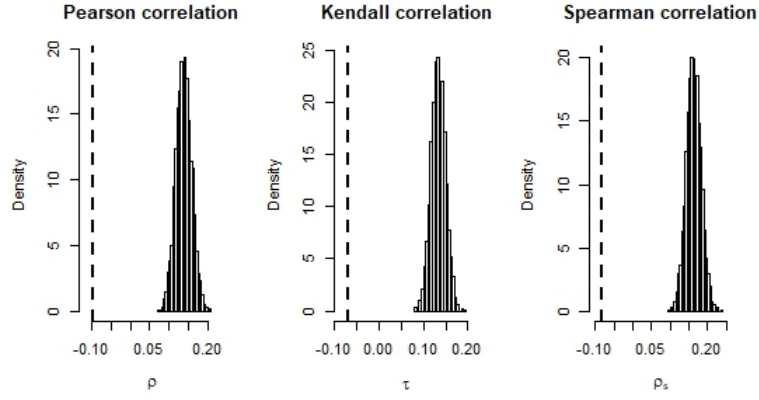
We report here the plots for checking the model assumptions (Section 4.4) for the following leagues: Spanish La Liga, German Bundesliga, Italian Serie A.

### 2.1 Conditional independence

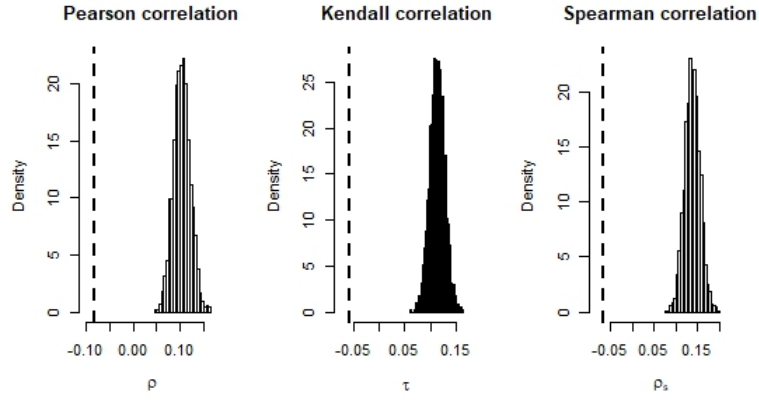
Analogously to Figure 5 in the paper, Figure 4 displays the distribution of three MCMC correlation coefficients (Pearson  $\rho$ , Kendall  $\tau$  and Spearman  $\rho_s$ ) along with the observed correlation for the marginal distributions of  $y_{m1}$  and  $y_{m2}$  (black dashed line).

### 2.2 Overdispersion

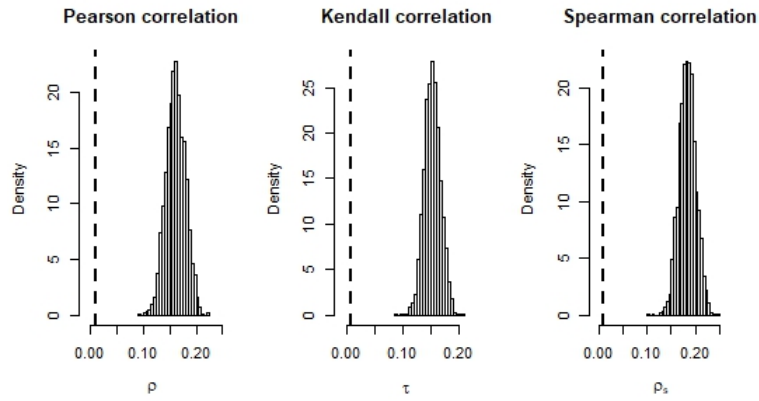
Figure 5, 6, 7, 8 show the MCMC distributions for the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$  against the observed value for the marginal data distributions (top row) and the joint MCMC distribution for the mean and the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$  (bottom row). As clearly emerges from the bottom row plots, replicated variances are greater than the replicated means; analogously as the interpretation for the conditional dependence, MCMC replications suggest there is no need of explicitly modelling the marginal overdispersion.



(a) Bundesliga



(b) La Liga



(c) Serie A

Figure 4: MCMC distribution for the Pearson correlation coefficient  $\rho$ , the Kendall correlation coefficient  $\tau$  and the Spearman correlation coefficient  $\rho_s$  between the replicated scores  $y_{m1}^{rep}$  and  $y_{m2}^{rep}$  for  $m = 1, \dots, M$ . The dashed black line denotes the observed correlations between the marginal distribution of the observed scores  $y_{m1}, y_{m2}$ .

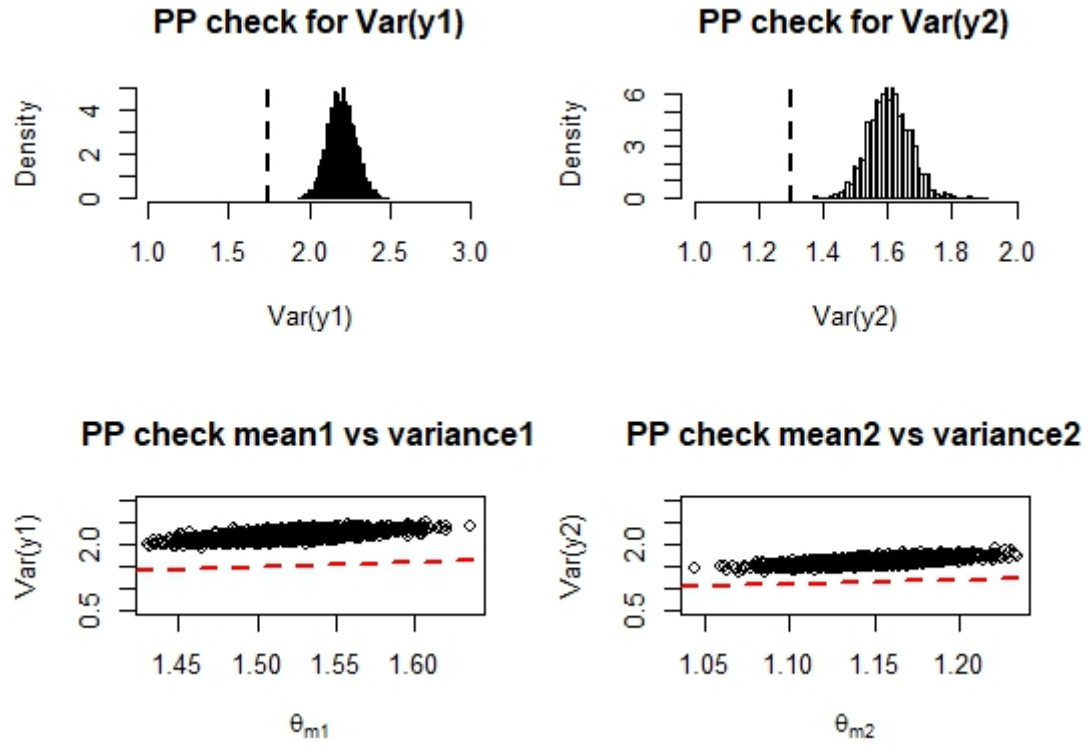


Figure 5: Top row plots: MCMC distribution for the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$  in EPL against the observed value for the marginal data distributions (black dashed line). Bottom row plots: joint MCMC distribution for the mean and the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$ .



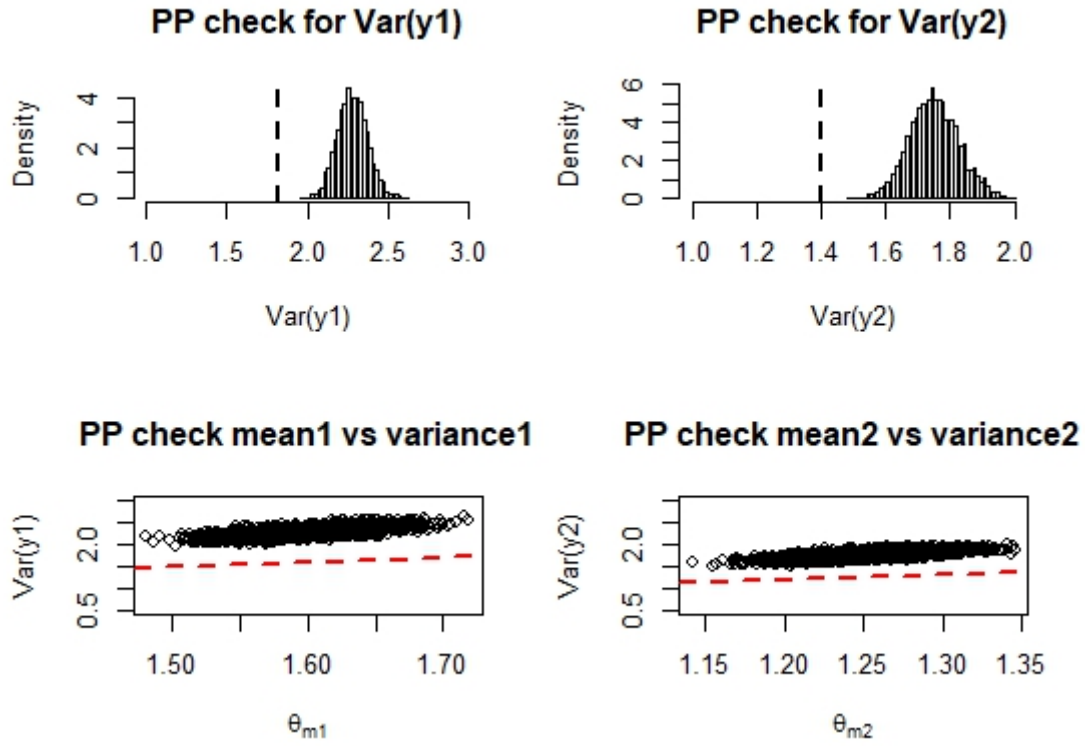


Figure 6: Top row plots: MCMC distribution for the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$  in German Bundesliga against the observed value for the marginal data distributions (black dashed line). Bottom row plots: joint MCMC distribution for the mean and the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$ .

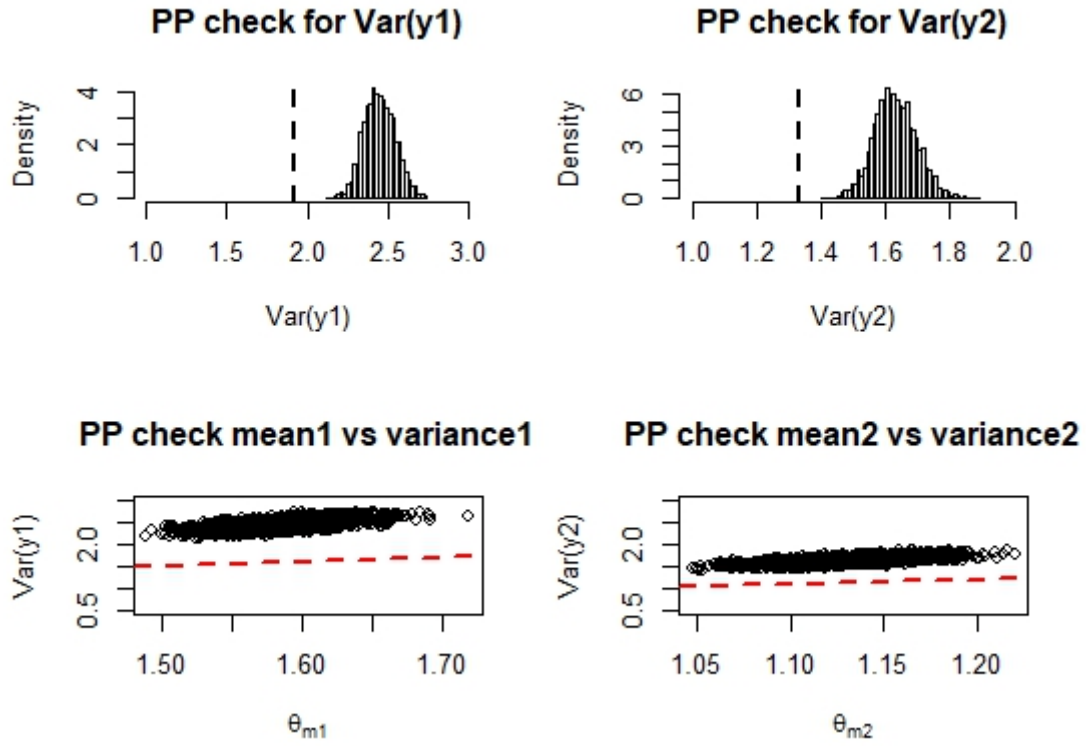


Figure 7: Top row plots: MCMC distribution for the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$  in Spanish La Liga against the observed value for the marginal data distributions (black dashed line). Bottom row plots: joint MCMC distribution for the mean and the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$ .

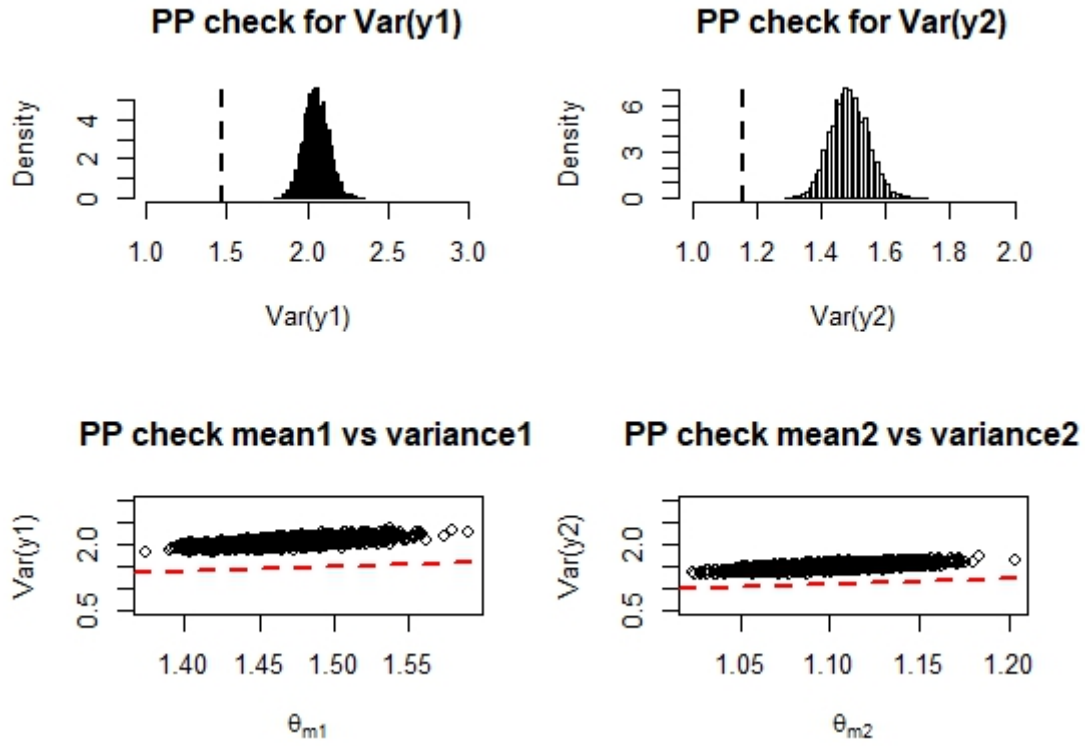


Figure 8: Top row plots: MCMC distribution for the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$  in Italian Serie A against the observed value for the marginal data distributions (black dashed line). Bottom row plots: joint MCMC distribution for the mean and the variance of the replicated scores  $y_1^{rep}, y_2^{rep}$ .

### 3 Code

We include here the main part of the R code used for reading data of the EPL, along with the JAGS model code for fitting the hierarchical Bayesian model proposed in Section 3 of the paper.

#### 3.1 Reading data

Data are downloaded from the web page <http://www.football-data.co.uk/> and saved in a local working directory as .csv files, for instance PL07-08.csv.

```
library(runjags)
library(R2jags)
library(MCMCpack)
library(rjags)
library(R2WinBUGS)
library(readr)

mydir <- getwd()
#global function for importing several seasons
import_data <- function(season){
  season <- read_csv(paste(mydir, "/", season, ".csv", sep=""))
  pl <- cbind( season$HomeTeam, season$AwayTeam,
              season$FTHG, season$FTAG, season$FTR,
              season$HS, season$AS, season$HST, season$AST,
              season$HY, season$AY, season$HR, season$AR,
              season$B365H, season$B365D, season$B365A,
              season$BWH, season$BWD, season$BWA,
              season$IWH, season$IWD, season$IWA,
              season$LBH, season$LBD, season$LBA,
              season$SBH, season$SBD, season$SBA,
              season$WHH, season$WHD, season$WHA,
              season$VCH, season$VCD, season$VCA)
  return(list(pl=pl))
}
#data concatenation
seasons_string <- c("PL07-08", "PL08-09", "PL09-10", "PL10-11",
                   "PL11-12", "PL12-13", "PL13-14",
```

```

        "PL14-15", "PL15-16", "PL16-17" )

#full dataset
PL <- rbind(import_data("PL07-08")$pl,
            import_data("PL08-09")$pl,
            import_data("PL09-10")$pl,
            import_data("PL10-11")$pl,
            import_data("PL11-12")$pl,
            import_data("PL12-13")$pl,
            import_data("PL13-14")$pl,
            import_data("PL14-15")$pl,
            import_data("PL15-16")$pl,
            import_data("PL16-17")$pl)

#season index
T <- 10
agenzie <- 7
season_or <- rep(1:T, each=380)
#number of season
nseason <- length(unique(season_or))
season_time <- c(1:nseason)
#teams for all the nine seasons
teams_matrix <- matrix(NA, nseason, 20)
  for (i in 1:length(seasons_string)){
    launch <- import_data(seasons_string[i])
    teams_matrix[i,] <- unique(launch$pl[,1])
  }
teams <- unique( as.vector(t(teams_matrix)))
PL16_17 <- import_data("PL16-17")
teams16_17 <- unique(PL16_17$season$HomeTeam)
teams16_17_index <- match(teams16_17, teams)
nteam <- length(teams)
#total games
ngames_tot <- length(PL[,1])
#test set games
ngames_test <- 380
#train set games
ngames_train <- ngames_tot-ngames_test
season_train <- season_or[1:ngames_train]
#team index, scores and kicks
team1 <- match (as.vector(PL[,1])[1:ngames_train], teams)
score1 <- as.numeric(as.vector(PL[,3])[1:ngames_train])
team2 <- match (as.vector(PL[,2])[1:ngames_train], teams)
score2 <- as.numeric(as.vector(PL[,4])[1:ngames_train])

```

```

kicks1      <- as.vector(as.numeric(PL[1:ngames_train,6]))
kicks2      <- as.vector(as.numeric(PL[1:ngames_train,7]))
kicks1_target <- as.vector(as.numeric(PL[1:ngames_train,8]))
kicks2_target <- as.vector(as.numeric(PL[1:ngames_train,9]))
score1_prev  <- as.numeric(as.vector(PL[,3])[(ngames_train+1):
                                           ngames_tot])
score2_prev  <- as.numeric(as.vector(PL[,4])[(ngames_train+1):
                                           ngames_tot])
team1_prev   <- match (as.vector(PL[,1])[(ngames_train+1):
                                           ngames_tot], teams)
team2_prev   <- match (as.vector(PL[,2])[(ngames_train+1):
                                           ngames_tot], teams)
season_prev  <- season_or[(ngames_train+1):ngames_tot]

```

### 3.2 Transforming betting odds in implicit scoring rates

We create an array for the odds arising from seven selected bookmakers:

```

bet365_PL<-matrix(NA,ngames_tot_PL,3)
bet365_PL[,1]<-as.numeric(as.vector(PL[1:ngames_tot_PL,14]))
bet365_PL[,2]<-as.numeric(as.vector(PL[1:ngames_tot_PL,15]))
bet365_PL[,3]<-as.numeric(as.vector(PL[1:ngames_tot_PL,16]))

bwin_PL<-matrix(NA,ngames_tot_PL,3)
bwin_PL[,1]<-as.numeric(as.vector(PL[1:ngames_tot_PL,17]))
bwin_PL[,2]<-as.numeric(as.vector(PL[1:ngames_tot_PL,18]))
bwin_PL[,3]<-as.numeric(as.vector(PL[1:ngames_tot_PL,19]))

IW_PL<-matrix(NA,ngames_tot_PL,3)
IW_PL[,1]<-as.numeric(as.vector(PL[1:ngames_tot_PL,20]))
IW_PL[,2]<-as.numeric(as.vector(PL[1:ngames_tot_PL,21]))
IW_PL[,3]<-as.numeric(as.vector(PL[1:ngames_tot_PL,22]))

LB_PL<-matrix(NA,ngames_tot_PL,3)
LB_PL[,1]<-as.numeric(as.vector(PL[1:ngames_tot_PL,23]))
LB_PL[,2]<-as.numeric(as.vector(PL[1:ngames_tot_PL,24]))
LB_PL[,3]<-as.numeric(as.vector(PL[1:ngames_tot_PL,25]))

PS_PL<-matrix(NA,ngames_tot_PL,3)
PS_PL[,1]<-as.numeric(as.vector(PL[1:ngames_tot_PL,26]))

```

```

PS_PL[,2]<-as.numeric(as.vector(PL[1:ngames_tot_PL,27]))
PS_PL[,3]<-as.numeric(as.vector(PL[1:ngames_tot_PL,28]))

WH_PL<-matrix(NA,ngames_tot_PL,3)
WH_PL[,1]<-as.numeric(as.vector(PL[1:ngames_tot_PL,29]))
WH_PL[,2]<-as.numeric(as.vector(PL[1:ngames_tot_PL,30]))
WH_PL[,3]<-as.numeric(as.vector(PL[1:ngames_tot_PL,31]))

VC_PL<-matrix(NA,ngames_tot_PL,3)
VC_PL[,1]<-as.numeric(as.vector(PL[1:ngames_tot_PL,32]))
VC_PL[,2]<-as.numeric(as.vector(PL[1:ngames_tot_PL,33]))
VC_PL[,3]<-as.numeric(as.vector(PL[1:ngames_tot_PL,34]))

agenzie<-7
array_odds_PL<-array(NA, dim=c(ngames_tot_PL, agenzie,3 ))

array_odds_PL[,1,]<-bet365_PL
array_odds_PL[,2,]<-bwin_PL
array_odds_PL[,3,]<-IW_PL
array_odds_PL[,4,]<-LB_PL
array_odds_PL[,5,]<-PS_PL
array_odds_PL[,6,]<-WH_PL
array_odds_PL[,7,]<-VC_PL

```

Now the function `bookmakers.quant` transforms the betting in implicit scoring rates

```

library(skellam)

bookmakers.quant=function(ngames_tot, agenzie,
ngames_train, ngames_test, array_odds ){

odds_inv <- 1/array_odds
book_odds_sum <- apply(odds_inv, c(1,2),sum)
book_odds_sum_array <- array(NA, c(ngames_tot, agenzie,3))
for (h in 1:3){
  book_odds_sum_array[, ,h] <- book_odds_sum
}
p_norm <- odds_inv/book_odds_sum_array

```

```

dist=function(p,prob){
  p <- exp(p)
  ((1-pskellam(0,p[1],p[2]))-prob[1])^2+
  ((pskellam(-1,p[1],p[2]))-prob[3])^2+
  ((dskellam(0,p[1],p[2]))-prob[2])^2
}

theta1_bm <- matrix(NA, ngames_train,agenzie)
theta2_bm <- matrix(NA, ngames_train,agenzie)

for (n in 1:ngames_train){
  for (g in 1:agenzie){
    x <- optim(c(1,1),fn=dist,prob=p_norm[n,g, ])
    theta1_bm[n,g] <- exp(x$par[[1]])
    theta2_bm[n,g] <- exp(x$par[[2]])
  }}

theta1_bm_mean <- apply(theta1_bm,1,mean)
theta2_bm_mean <- apply(theta2_bm,1,mean)

theta1_bm_prev <- matrix(NA, ngames_test,agenzie)
theta2_bm_prev <- matrix(NA, ngames_test,agenzie)

for (n in (ngames_train+1):ngames_tot){
  for (g in 1:agenzie){
    x <- optim(c(1,1),fn=dist,prob=p_norm[n,g, ])
    theta1_bm_prev[(n-ngames_train),g]<-exp(x$par[[1]])
    theta2_bm_prev[(n-ngames_train),g]<-exp(x$par[[2]])
  }}

theta1_bm_mean_prev <- apply(theta1_bm_prev,1,mean)
theta2_bm_mean_prev <- apply(theta2_bm_prev,1,mean)

return(list(p_norm=p_norm,theta1_bm=theta1_bm, theta2_bm=theta2_bm,
            theta1_bm_mean=theta1_bm_mean,
            theta2_bm_mean=theta2_bm_mean,
            theta1_bm_prev=theta1_bm_prev,
            theta2_bm_prev=theta2_bm_prev,
            theta1_bm_mean_prev=theta1_bm_mean_prev,
            theta2_bm_mean_prev=theta2_bm_mean_prev))
}

```



```
book <- bookmakers.quant( ngames_tot, agenzie,
ngames_train, ngames_test, array_odds)
```

### 3.3 JAGS code

We create a list for JAGS data input:

```
dati_PL <- list(nteams=nteams,
                ngames_train=ngames_train,
                ngames_test=ngames_test,
                team1=team1,
                score1=score1,
                team2=team2,
                score2=score2,
                team1_prev=team1_prev,
                team2_prev=team2_prev,
                theta1_bm_mean=book$theta1_bm_mean,
                theta2_bm_mean=book$theta2_bm_mean,
                theta1_bm_mean_prev=book$theta1_bm_mean_prev,
                theta2_bm_mean_prev=book$theta2_bm_mean_prev,
                theta1_bm=book$theta1_bm,
                theta2_bm=book$theta2_bm,
                theta1_bm_prev=book$theta1_bm_prev,
                theta2_bm_prev=book$theta2_bm_prev,
                onesRepNclust_and=rep(1,2),
                onesRepNclust_prev=rep(1,2),
                T=T,
                season=season_train,
                season_prev=season_prev,
                agenzie=agenzie)
```

And here is the JAGS model:

```
model{
# Likelihood:

for (n in 1:ngames_train){
  for (s in 1:agenzie){
```

```

    theta1_bm[n,s] ~ dlnorm(lambda1_book[n], tau1_book)
    theta2_bm[n,s] ~ dlnorm(lambda2_book[n], tau2_book)
  }

  theta1_hat[n] <- pClust1[n,1]*thetaofClust[n,1,1] +
    pClust1[n,2]*thetaofClust[n,1,2]
  theta2_hat[n] <- pClust1[n,1]*thetaofClust[n,2,1] +
    pClust1[n,2]*thetaofClust[n,2,2]

  score1[n] ~ dpois(theta1_hat[n])
  score2[n] ~ dpois(theta2_hat[n])

# Average Scoring intensities (accounting for mixing components)
  pClust1[n,1:2] ~ ddirch(onesRepNclust_and[1:2])
  log(thetaofClust[n,1,1])<-m+home+
    att[team1[n], season[n]]+def[team2[n], season[n]]
  log(thetaofClust[n,1,2])<-log(lambda1_book[n])
  log(thetaofClust[n,2,1])<-m+ att[team2[n],season[n]]+def[team1[n],season[n]]
  log(thetaofClust[n,2,2])<-log(lambda2_book[n])

# Priors for lambda1_book, lambda2_book

  lambda1_book[n]~ dlnorm(theta1_bm_mean[n], 0.01)
  lambda2_book[n]~ dlnorm(theta2_bm_mean[n], 0.01)
}

# Predictive distribution for the number of goals scored

for (n in 1:ngames_test){
  for (s in 1:agenzie){

    theta1_bm_prev[n,s] ~ dlnorm(lambda1_book_prev[n], tau1_book)
    theta2_bm_prev[n,s] ~ dlnorm(lambda2_book_prev[n], tau2_book)
  }

  theta1_hat_prev[n] <- pClust1_prev[n,1]*thetaofClust_prev[n,1,1]+
    pClust1_prev[n,2]*thetaofClust_prev[n,1,2]

  theta2_hat_prev[n] <- pClust1_prev[n,1]*thetaofClust_prev[n,2,1]+
    pClust1_prev[n,2]*thetaofClust_prev[n,2,2]
  score1_prev[n] ~ dpois(theta1_hat_prev[n])
  score2_prev[n] ~ dpois(theta2_hat_prev[n])

  pClust1_prev[n,1:2] ~ ddirch(onesRepNclust_prev[1:2])

```

```

log(thetaofClust_prev[n,1,1]) <- m+home+
      att[team1_prev[n], season_prev[n]]+
      def[team2_prev[n], season_prev[n]]
log(thetaofClust_prev[n,1,2]) <- log(lambda1_book_prev[n])
log(thetaofClust_prev[n,2,1]) <- m+
      att[team2_prev[n], season_prev[n]]+
      def[team1_prev[n], season_prev[n]]
log(thetaofClust_prev[n,2,2]) <- log(lambda2_book_prev[n])

#prior for lambda1_book, lambda2_book

      lambda1_book_prev[n] ~ dlnorm(theta1_bm_mean_prev[n], 0.01)
      lambda2_book_prev[n] ~ dlnorm(theta2_bm_mean_prev[n], 0.01)
}

# Prior: attack/defence abilities
for (t in 1:nteams){
  att.star[t,1] ~ dnorm(mu.att, tau.att)
  def.star[t,1] ~ dnorm(mu.def, tau.def)
  att[t,1] <- att.star[t,1] - mean(att.star[,1])
  def[t,1] <- def.star[t,1] - mean(def.star[,1])
  for (h in 2:T){
    att.star[t,h] ~ dnorm(mu.att+att.star[t,h-1],tau.att)
    def.star[t,h] ~ dnorm(mu.def+def.star[t,h-1],tau.def)
    att[t,h] <- att.star[t,h] - mean(att.star[,h])
    def[t,h] <- def.star[t,h] - mean(def.star[,h])
  }
}

# Priors on the random effects

mu.att ~ dnorm(0,0.0001)
mu.def ~ dnorm(0,0.0001)
tau.att ~ dgamma(.01,.01)
tau.def ~ dgamma(.01,.01)
m ~ dnorm(0,0.0001)
tau1_book <- pow(sigma1.y,-2)
tau2_book <- pow(sigma2.y,-2)
sigma1.y ~ dunif(0,10)
sigma2.y ~ dunif(0,10)
home ~ dnorm(0,0.0001)
}

```