Bradley-Terry Models in R: The BradleyTerry2 Package

For BradleyTerry2 version 1.1-0, 2019-05-11
https://github.com/hturner/BradleyTerry2

Heather Turner
University of Warwick

David Firth
University of Warwick

Abstract

This is a short overview of the R add-on package **BradleyTerry2**, which facilitates the specification and fitting of Bradley-Terry logit, probit or cauchit models to pair-comparison data. Included are the standard 'unstructured' Bradley-Terry model, structured versions in which the parameters are related through a linear predictor to explanatory variables, and the possibility of an order or 'home advantage' effect or other 'contest-specific' effects. Model fitting is either by maximum likelihood, by penalized quasi-likelihood (for models which involve a random effect), or by bias-reduced maximum likelihood in which the first-order asymptotic bias of parameter estimates is eliminated. Also provided are a simple and efficient approach to handling missing covariate data, and suitably-defined residuals for diagnostic checking of the linear predictor.

Keywords: generalized linear model, logistic regression, penalized quasi-likelihood, ranking, tournament analysis, working residuals.

1. Introduction

The Bradley-Terry model (Bradley and Terry 1952) assumes that in a 'contest' between any two 'players', say player i and player j ($i, j \in \{1, \ldots, K\}$), the odds that i beats j are α_i/α_j, where α_i and α_j are positive-valued parameters which might be thought of as representing 'ability'. A general introduction can be found in Bradley (1984) or Agresti (2002). Applications are many, ranging from experimental psychology to the analysis of sports tournaments to genetics (for example, the allelic transmission/disequilibrium test of Sham and Curtis 1995 is based on a Bradley-Terry model in which the 'players' are alleles). In typical psychometric applications the 'contests' are comparisons, made by different human subjects, between pairs of items.

The model can alternatively be expressed in the logit-linear form

$$\text{logit}[\text{pr}(i \text{ beats } j)] = \lambda_i - \lambda_j, \tag{1}$$

where $\lambda_i = \log \alpha_i$ for all i. Thus, assuming independence of all contests, the parameters $\{\lambda_i\}$ can be estimated by maximum likelihood using standard software for generalized linear models, with a suitably specified model matrix. The primary purpose of the **BradleyTerry2** package (Turner and Firth 2012), implemented in the R statistical computing environment (Ihaka and Gentleman 1996; R Development Core Team 2012), is to facilitate the specification and fitting of such models and some extensions.

The **BradleyTerry2** package supersedes the earlier **BradleyTerry** package (Firth 2005), providing a more flexible user interface to allow a wider range of models to be fitted. In particular, **BradleyTerry2** allows the inclusion of simple random effects so that the ability parameters can be related to available explanatory variables through a linear predictor of the form

$$\lambda_i = \sum_{r=1}^{p} \beta_r x_{ir} + U_i. \tag{2}$$

The inclusion of the prediction error U_i allows for variability between players with equal covariate values and induces correlation between comparisons with a common player. **BradleyTerry2** also allows for general contest-specific effects to be included in the model and allows the logit link to be replaced, if required, by a different symmetric link function (probit or cauchit).

The remainder of the paper is organised as follows. Section 2 demonstrates how to use the **BradleyTerry2** package to fit a standard (i.e., unstructured) Bradley-Terry model, with a separate ability parameter estimated for each player, including the use of bias-reduced estimation for such models. Section 3 considers variations of the standard model, including the use of player-specific variables to model ability and allowing for contest-specific effects such as an order effect or judge effects. Sections 4 and 5 explain how to obtain important information about a fitted model, in particular the estimates of ability and their standard errors, and player-level residuals, whilst Section 6 notes the functions available to aid model search. Section 7 explains in more detail how set up data for use with the **BradleyTerry2** package, Section 8 lists the functions provided by the package and finally Section 9 comments on two directions for further development of the software.

2. Standard Bradley-Terry model

2.1. Example: Analysis of journal citations

The following data come from page 448 of Agresti (2002), extracted from the larger table of Stigler (1994). The data are counts of citations among four prominent journals of statistics and are included the **BradleyTerry2** package as the data set `citations`:

```
R> library("BradleyTerry2")
```

```
R> data("citations", package = "BradleyTerry2")
```

```
R> citations
```

```
              citing
cited          Biometrika Comm Statist JASA JRSS-B
  Biometrika          714          730  498    221
  Comm Statist         33          425   68     17
  JASA                320          813 1072    142
  JRSS-B              284          276  325    188
```

Thus, for example, *Biometrika* was cited 498 times by papers in *Journal of the American Statistical Association* (JASA) during the period under study. In order to fit a Bradley-Terry model to these data using `BTm` from the **BradleyTerry2** package, the data must first be converted to binomial frequencies. That is, the data need to be organised into pairs (`player1`, `player2`) and corresponding frequencies of wins and losses for `player1` against `player2`. The **BradleyTerry2** package provides the utility function `countsToBinomial` to convert a contingency table of wins to the format just described:

```
R> citations.sf <- countsToBinomial(citations)
R> names(citations.sf)[1:2] <- c("journal1", "journal2")
R> citations.sf
```

```
      journal1     journal2 win1 win2
1   Biometrika Comm Statist  730   33
2   Biometrika         JASA  498  320
3   Biometrika       JRSS-B  221  284
4 Comm Statist         JASA   68  813
5 Comm Statist       JRSS-B   17  276
6         JASA       JRSS-B  142  325
```

Note that the self-citation counts are ignored – these provide no information on the ability parameters, since the abilities are relative rather than absolute quantities. The binomial response can then be modelled by the difference in player abilities as follows:

```
R> citeModel <- BTm(cbind(win1, win2), journal1, journal2, ~ journal,
+      id = "journal", data = citations.sf)
R> citeModel
```

```
Bradley Terry model fit by glm.fit

Call:  BTm(outcome = cbind(win1, win2), player1 = journal1, player2 = journal2,
    formula = ~journal, id = "journal", data = citations.sf)

Coefficients:
journalComm Statist          journalJASA        journalJRSS-B
            -2.9491              -0.4796               0.2690

Degrees of Freedom: 6 Total (i.e. Null);  3 Residual
Null Deviance:            1925
Residual Deviance: 4.293         AIC: 46.39
```

The coefficients here are maximum likelihood estimates of $\lambda_2, \lambda_3, \lambda_4$, with λ_1 (the log-ability for *Biometrika*) set to zero as an identifying convention.

The one-sided model formula

```
~ journal
```

specifies the model for player ability, in this case the 'citeability' of the journal. The `id` argument specifies that `"journal"` is the name to be used for the factor that identifies the player – the values of which are given here by `journal1` and `journal2` for the first and second players respectively. Therefore in this case a separate citeability parameter is estimated for each journal.

If a different 'reference' journal is required, this can be achieved using the optional `refcat` argument: for example, making use of `update` to avoid re-specifying the whole model,

```
R> update(citeModel, refcat = "JASA")

Bradley Terry model fit by glm.fit

Call:  BTm(outcome = cbind(win1, win2), player1 = journal1, player2 = journal2,
    formula = ~journal, id = "journal", refcat = "JASA", data = citations.sf)

Coefficients:
  journalBiometrika  journalComm Statist        journalJRSS-B
             0.4796              -2.4695               0.7485

Degrees of Freedom: 6 Total (i.e. Null);  3 Residual
Null Deviance:            1925
Residual Deviance: 4.293         AIC: 46.39
```

– the same model in a different parameterization.

The use of the standard Bradley-Terry model for this application might perhaps seem rather questionable – for example, citations within a published paper can hardly be considered independent, and the model discards potentially important information on self-citation. Stigler (1994) provides arguments to defend the model's use despite such concerns.

2.2. Bias-reduced estimates

Estimation of the standard Bradley-Terry model in `BTm` is by default computed by maximum likelihood, using an internal call to the `glm` function. An alternative is to fit by bias-reduced maximum likelihood (Firth 1993): this requires additionally the **brglm** package (Kosmidis 2007), and is specified by the optional argument `br = TRUE`. The resultant effect, namely removal of first-order asymptotic bias in the estimated coefficients, is often quite small. One notable feature of bias-reduced fits is that all estimated coefficients and standard errors are necessarily finite, even in situations of 'complete separation' where maximum likelihood estimates take infinite values (Heinze and Schemper 2002).

For the citation data, the parameter estimates are only very slightly changed in the bias-reduced fit:

```
R> update(citeModel, br = TRUE)

Bradley Terry model fit by brglm.fit
```

```
Call:  BTm(outcome = cbind(win1, win2), player1 = journal1, player2 = journal2,      formu

Coefficients:
journalComm Statist          journalJASA        journalJRSS-B
           -2.9444              -0.4791               0.2685

Degrees of Freedom: 6 Total (i.e. Null);  3 Residual
Deviance:            4.2957
Penalized Deviance: -11.4816         AIC: 46.3962
```

Here the bias of maximum likelihood is small because the binomial counts are fairly large. In more sparse arrangements of contests – that is, where there is less or no replication of the contests – the effect of bias reduction would typically be more substantial than the insignificant one seen here.

3. Abilities predicted by explanatory variables

3.1. 'Player-specific' predictor variables

In some application contexts there may be 'player-specific' explanatory variables available, and it is then natural to consider model simplification of the form

$$\lambda_i = \sum_{r=1}^{p} \beta_r x_{ir} + U_i, \tag{3}$$

in which ability of each player i is related to explanatory variables $x_{i1}, \ldots, x_{ip}$ through a linear predictor with coefficients $\beta_1, \ldots, \beta_p$; the $\{U_i\}$ are independent errors. Dependence of the player abilities on explanatory variables can be specified via the `formula` argument, using the standard S-language model formulae. The difference in the abilities of player i and player j is modelled by

$$\sum_{r=1}^{p} \beta_r x_{ir} - \sum_{r=1}^{p} \beta_r x_{jr} + U_i - U_j, \tag{4}$$

where $U_i \sim N(0, \sigma^2)$ for all i. The Bradley-Terry model is then a generalized linear mixed model, which the `BTm` function currently fits by using the penalized quasi-likelihood algorithm of Breslow and Clayton (1993).

As an illustration, consider the following simple model for the `flatlizards` data, which predicts the fighting ability of Augrabies flat lizards by body size (snout to vent length):

```
R> options(show.signif.stars = FALSE)
R> data("flatlizards", package = "BradleyTerry2")
R> lizModel <- BTm(1, winner, loser, ~ SVL[..] + (1|..),
+                  data = flatlizards)
```

Here the winner of each fight is compared to the loser, so the outcome is always 1. The special name '`..`' appears in the formula as the default identifier for players, in the absence of a user-specified `id` argument. The values of this factor are given by `winner` for the winning lizard

and `loser` for the losing lizard in each contest. These factors are provided in the data frame `contests` that is the first element of the list object `flatlizards`. The second element of `flatlizards` is another data frame, `predictors`, containing measurements on the observed lizards, including `SVL`, which is the snout to vent length. Thus `SVL[..]` represents the snout to vent length indexed by lizard (`winner` or `loser` as appropriate). Finally a random intercept for each lizard is included using the bar notation familiar to users of the **lme4** package (Bates, Mächler, and Bolker 2011). (Note that a random intercept is the only random effect structure currently implemented in **BradleyTerry2**.)

The fitted model is summarized below:

```
R> summary(lizModel)

Call:

BTm(outcome = 1, player1 = winner, player2 = loser, formula = ~SVL[..] +
    (1 | ..), data = flatlizards)

Fixed Effects:
        Estimate Std. Error z value Pr(>|z|)
SVL[..]   0.2051     0.1158   1.772   0.0765

(Dispersion parameter for binomial family taken to be 1)

Random Effects:
          Estimate Std. Error z value Pr(>|z|)
Std. Dev.    1.126      0.261   4.314  1.6e-05

Number of iterations: 8
```

The coefficient of snout to vent length is weakly significant; however, the standard deviation of the random effect is quite large, suggesting that this simple model has fairly poor explanatory power. A more appropriate model is considered in the next section.

3.2. Missing values

The contest data may include all possible pairs of players and hence rows of missing data corresponding to players paired with themselves. Such rows contribute no information to the Bradley-Terry model and are simply discarded by `BTm`.

Where there are missing values in player-specific *predictor* (or *explanatory*) variables which appear in the formula, it will typically be very wasteful to discard all contests involving players for which some values are missing. Instead, such cases are accommodated by the inclusion of one or more parameters in the model. If, for example, player 1 has one or more of its predictor values $x_{11}, \ldots, x_{1p}$ missing, then the combination of Equations 1 and 4 above yields

$$\text{logit}[\text{pr}(1 \text{ beats } j)] = \lambda_1 - \left(\sum_{r=1}^{p} \beta_r x_{jr} + U_j \right), \tag{5}$$

for all other players j. This results in the inclusion of a 'direct' ability parameter for each player having missing predictor values, in addition to the common coefficients $\beta_1, \ldots, \beta_p$ – an approach which will be appropriate when the missingness mechanism is unrelated to contest success. The same device can be used also to accommodate any user-specified departures from a structured Bradley-Terry model, whereby some players have their abilities determined by the linear predictor but others do not.

In the original analysis of the `flatlizards` data (Whiting, Stuart-Fox, O'Connor, Firth, Bennett, and Blomberg 2006), the final model included the first and third principal components of the spectral reflectance from the throat (representing brightness and UV intensity respectively) as well as head length and the snout to vent length seen in our earlier model. The spectroscopy data was missing for two lizards, therefore the ability of these lizards was estimated directly. The following fits this model, with the addition of a random intercept as before:

```
R> lizModel2 <- BTm(1, winner, loser,
+               ~ throat.PC1[..] + throat.PC3[..] +
+               head.length[..] + SVL[..] + (1|..),
+               data = flatlizards)
R> summary(lizModel2)

Call:

BTm(outcome = 1, player1 = winner, player2 = loser, formula = ~throat.PC1[..] +
    throat.PC3[..] + head.length[..] + SVL[..] + (1 | ..), data = flatlizards)

Fixed Effects:
                  Estimate Std. Error z value Pr(>|z|)
..lizard096      3.668e+01  3.875e+07   0.000   1.0000
..lizard099      9.531e-01  1.283e+00   0.743   0.4576
throat.PC1[..]  -8.689e-02  4.120e-02  -2.109   0.0349
throat.PC3[..]   3.735e-01  1.527e-01   2.445   0.0145
head.length[..] -1.382e+00  7.390e-01  -1.870   0.0614
SVL[..]          1.722e-01  1.373e-01   1.254   0.2098

(Dispersion parameter for binomial family taken to be 1)

Random Effects:
          Estimate Std. Error z value Pr(>|z|)
Std. Dev.   1.1099     0.3223   3.443 0.000575

Number of iterations: 8
```

Note that `BTm` detects that lizards 96 and 99 have missing values in the specified predictors and automatically includes separate ability parameters for these lizards. This model was found to be the single best model based on the principal components of reflectance and the other predictors available and indeed the standard deviation of the random intercept is much reduced, but still highly significant. Allowing for this significant variation between lizards

with the same predictor values produces more realistic (i.e., larger) standard errors for the parameters when compared to the original analysis of Whiting *et al.* (2006). Although this affects the significance of the morphological variables, it does not affect the significance of the principal components, so in this case does not affect the main conclusions of the study.

3.3. Order effect

In certain types of application some or all contests have an associated 'bias', related to the order in which items are presented to a judge or with the location in which a contest takes place, for example. A natural extension of the Bradley-Terry model (Equation 1) is then

$$\text{logit}[\text{pr}(i \text{ beats } j)] = \lambda_i - \lambda_j + \delta z, \tag{6}$$

where $z = 1$ if i has the supposed advantage and $z = -1$ if j has it. (If the 'advantage' is in fact a disadvantage, δ will be negative.) The scores λ_i then relate to ability in the absence of any such advantage.

As an example, consider the baseball data given in Agresti (2002), page 438:

```
R> data("baseball", package = "BradleyTerry2")
R> head(baseball)
```

```
  home.team away.team home.wins away.wins
1 Milwaukee   Detroit         4         3
2 Milwaukee   Toronto         4         2
3 Milwaukee  New York         4         3
4 Milwaukee    Boston         6         1
5 Milwaukee Cleveland         4         2
6 Milwaukee Baltimore         6         0
```

The data set records the home wins and losses for each baseball team against each of the 6 other teams in the data set. The `head` function is used to show the first 6 records, which are the Milwaukee home games. We see for example that Milwaukee played 7 home games against Detroit and won 4 of them. The 'standard' Bradley-Terry model without a home-advantage parameter will be fitted if no formula is specified in the call to `BTm`:

```
R> baseballModel1 <- BTm(cbind(home.wins, away.wins), home.team, away.team,
+                        data = baseball, id = "team")
R> summary(baseballModel1)
```

```
Call:
BTm(outcome = cbind(home.wins, away.wins), player1 = home.team,
    player2 = away.team, id = "team", data = baseball)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.6539  -0.0508   0.4133   0.9736   2.5509
```

```
Coefficients:
             Estimate Std. Error z value Pr(>|z|)
teamBoston      1.1077     0.3339   3.318 0.000908
teamCleveland   0.6839     0.3319   2.061 0.039345
teamDetroit     1.4364     0.3396   4.230 2.34e-05
teamMilwaukee   1.5814     0.3433   4.607 4.09e-06
teamNew York    1.2476     0.3359   3.715 0.000203
teamToronto     1.2945     0.3367   3.845 0.000121

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 78.015  on 42  degrees of freedom
Residual deviance: 44.053  on 36  degrees of freedom
AIC: 140.52

Number of Fisher Scoring iterations: 4
```

The reference team is Baltimore, estimated to be the weakest of these seven, with Milwaukee and Detroit the strongest.

In the above, the ability of each team is modelled simply as `~ team` where the values of the factor `team` are given by `home.team` for the first team and `away.team` for the second team in each game. To estimate the home-advantage effect, an additional variable is required to indicate whether the team is at home or not. Therefore data frames containing both the team factor and this new indicator variable are required in place of the factors `home.team` and `away.team` in the call to `BTm`. This is achieved here by over-writing the `home.team` and `away.team` factors in the `baseball` data frame:

```
R> baseball$home.team <- data.frame(team = baseball$home.team, at.home = 1)
R> baseball$away.team <- data.frame(team = baseball$away.team, at.home = 0)
```

The `at.home` variable is needed for both the home team and the away team, so that it can be differenced as appropriate in the linear predictor. With the data organised in this way, the ability formula can now be updated to include the `at.home` variable as follows:

```
R> baseballModel2 <- update(baseballModel1, formula = ~ team + at.home)
R> summary(baseballModel2)

Call:
BTm(outcome = cbind(home.wins, away.wins), player1 = home.team,
    player2 = away.team, formula = ~team + at.home, id = "team",
    data = baseball)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-2.03819  -0.40577   0.04326   0.61163   2.26001

Coefficients:
```

```
              Estimate Std. Error z value Pr(>|z|)
teamBoston      1.1438     0.3378   3.386 0.000710
teamCleveland   0.7047     0.3350   2.104 0.035417
teamDetroit     1.4754     0.3446   4.282 1.85e-05
teamMilwaukee   1.6196     0.3474   4.662 3.13e-06
teamNew York    1.2813     0.3404   3.764 0.000167
teamToronto     1.3271     0.3403   3.900 9.64e-05
at.home         0.3023     0.1309   2.308 0.020981

(Dispersion parameter for binomial family taken to be 1)

    Null deviance: 78.015  on 42  degrees of freedom
Residual deviance: 38.643  on 35  degrees of freedom
AIC: 137.11

Number of Fisher Scoring iterations: 4
```

This reproduces the results given on page 438 of Agresti (2002): the home team has an estimated odds-multiplier of $\exp(0.3023) = 1.35$ in its favour.

3.4. More general (contest-specific) predictors

The 'home advantage' effect is a simple example of a contest-specific predictor. Such predictors are necessarily interactions, between aspects of the contest and (aspects of) the two 'players' involved.

For more elaborate examples of such effects, see `?chameleons` and `?CEMS`. The former includes an 'experience' effect, which changes through time, on the fighting ability of male chameleons. The latter illustrates a common situation in psychometric applications of the Bradley-Terry model, where *subjects* express preference for one of two *objects* (the 'players'), and it is the influence on the results of subject attributes that is of primary interest.

As an illustration of the way in which such effects are specified, consider the following model specification taken from the examples in `?CEMS`, where data on students' preferences in relation to six European management schools is analysed.

```
R> data("CEMS", package = "BradleyTerry2")
R> table8.model <-  BTm(outcome = cbind(win1.adj, win2.adj),
+      player1 = school1, player2 = school2, formula = ~ .. +
+      WOR[student] * LAT[..] +  DEG[student] * St.Gallen[..] +
+      STUD[student] * Paris[..] + STUD[student] * St.Gallen[..] +
+      ENG[student] * St.Gallen[..] + FRA[student] * London[..] +
+      FRA[student] * Paris[..] + SPA[student] * Barcelona[..] +
+      ITA[student] * London[..] + ITA[student] * Milano[..] +
+      SEX[student] * Milano[..],
+      refcat = "Stockholm", data = CEMS)
```

This model reproduces results from Table 8 of Dittrich, Hatzinger, and Katzenbeisser (2001) apart from minor differences due to the different treatment of ties. Here the outcome is the binomial frequency of preference for `school1` over `school2`, with ties counted as half a

'win' and half a 'loss'. The formula specifies the model for school 'ability' or worth. In this formula, the default label '..' represents the school (with values given by `school1` or `school2` as appropriate) and `student` is a factor specifying the student that made the comparison. The remaining variables in the formula use R's standard indexing mechanism to include student-specific variables, e.g., `WOR`: whether or not the student was in full-time employment, and school-specific variables, e.g., `LAT`: whether the school was in a 'Latin' city. Thus there are three types of variables: contest-specific (`school1`, `school2`, `student`), subject-specific (`WOR`, `DEG`, ...) and object-specific (`LAT`, `St.Gallen`, ...). These three types of variables are provided in three data frames, contained in the list object `CEMS`.

4. Ability scores

The function `BTabilities` extracts estimates and standard errors for the log-ability scores $\lambda_1, \ldots, \lambda_K$. These will either be 'direct' estimates, in the case of the standard Bradley-Terry model or for players with one or more missing predictor values, or 'model-based' estimates of the form $\hat{\lambda}_i = \sum_{r=1}^{p} \hat{\beta}_r x_{ir}$ for players whose ability is predicted by explanatory variables.

As a simple illustration, team ability estimates in the home-advantage model for the `baseball` data are obtained by:

```
R> BTabilities(baseballModel2)
```

```
            ability      s.e.
Baltimore 0.0000000 0.0000000
Boston    1.1438027 0.3378422
Cleveland 0.7046945 0.3350014
Detroit   1.4753572 0.3445518
Milwaukee 1.6195550 0.3473653
New York  1.2813404 0.3404034
Toronto   1.3271104 0.3403222
```

This gives, for each team, the estimated ability when the team enjoys no home advantage.

Similarly, estimates of the fighting ability of each lizard in the `flatlizards` data under the model based on the principal components of the spectral reflectance from the throat are obtained as follows:

```
R> head(BTabilities(lizModel2), 4)
```

```
            ability      s.e.
lizard003  1.562453 0.5227564
lizard005  0.869896 0.5643448
lizard006 -0.243853 0.5939836
lizard009  1.211622 0.6476100
```

The ability estimates in an unstructured Bradley-Terry model are particularly well suited to presentation using the device of *quasi-variances* (Firth and de Menezes 2004). The **qvcalc** package (Firth 2010, version 0.8-5 or later) contains a function of the same name which does the necessary work:

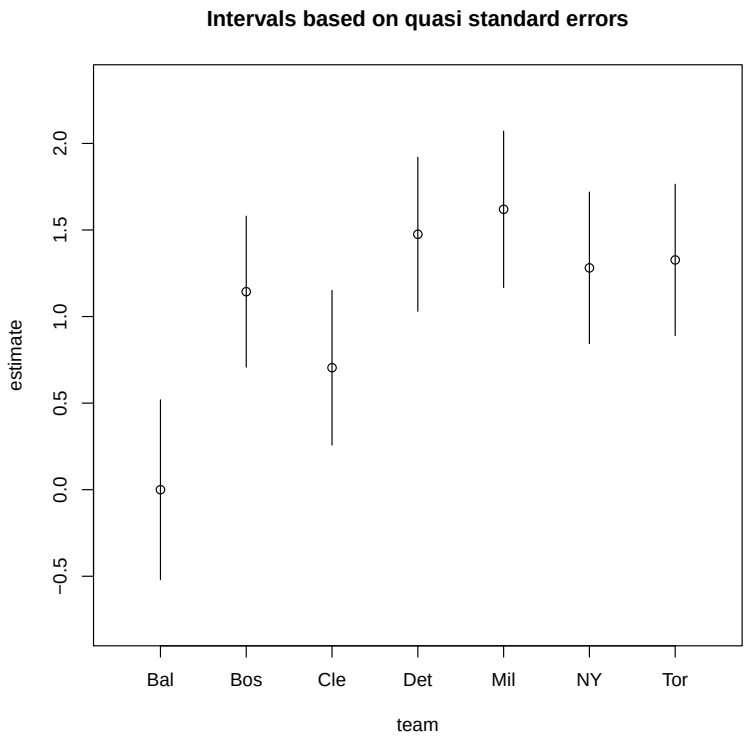

Figure 1: Estimated relative abilities of baseball teams.

```
> library("qvcalc")
> baseball.qv <- qvcalc(BTabilities(baseballModel2))
> plot(baseball.qv,
+      levelNames = c("Bal", "Bos", "Cle", "Det", "Mil", "NY", "Tor"))
```

The 'comparison intervals' as shown in Figure 1 are based on 'quasi standard errors', and can be interpreted as if they refer to *independent* estimates of ability for the journals. This has the advantage that comparison between any pair of journals is readily made (i.e., not only comparisons with the 'reference' journal). For details of the theory and method of calculation see Firth and de Menezes (2004).

5. Residuals

There are two main types of residuals available for a Bradley-Terry model object.

First, there are residuals obtained by the standard methods for models of class `"glm"`. These all deliver one residual for each contest or type of contest. For example, Pearson residuals for the model `lizModel2` can be obtained simply by

```
R> res.pearson <- round(residuals(lizModel2), 3)
R> head(cbind(flatlizards$contests, res.pearson), 4)
```

```
     winner     loser res.pearson
1 lizard048 lizard006       0.556
2 lizard060 lizard011       0.664
3 lizard023 lizard012       0.220
4 lizard030 lizard012       0.153
```

More useful for diagnostics on the linear predictor $\sum \beta_r x_{ir}$ are 'player'-level residuals, obtained by using the function `residuals` with argument `type = "grouped"`. These residuals can then be plotted against other player-specific variables.

```
R> res <- residuals(lizModel2, type = "grouped")
R> #  with(flatlizards$predictors, plot(throat.PC2, res))
R> #  with(flatlizards$predictors, plot(head.width, res))
```

These residuals estimate the error in the linear predictor; they are obtained by suitable aggregation of the so-called 'working' residuals from the model fit. The `weights` attribute indicates the relative information in these residuals – weight is roughly inversely proportional to variance – which may be useful for plotting and/or interpretation; for example, a large residual may be of no real concern if based on very little information. Weighted least-squares regression of these residuals on any variable already in the model is null. For example:

```
R> lm(res ~ throat.PC1, weights = attr(res, "weights"),
+     data = flatlizards$predictors)


Call:
lm(formula = res ~ throat.PC1, data = flatlizards$predictors,
    weights = attr(res, "weights"))

Coefficients:
(Intercept)   throat.PC1
 -3.614e-15   -2.454e-15

R> lm(res ~ head.length, weights = attr(res, "weights"),
+     data = flatlizards$predictors)


Call:
lm(formula = res ~ head.length, data = flatlizards$predictors,
    weights = attr(res, "weights"))

Coefficients:
(Intercept)  head.length
 -3.644e-15   -6.726e-14
```

As an illustration of evident *non-null* residual structure, consider the unrealistically simple model `lizModel` that was fitted in Section 3 above. That model lacks the clearly significant predictor variable `throat.PC3`, and the plot shown in Figure 2 demonstrates this fact graphically:

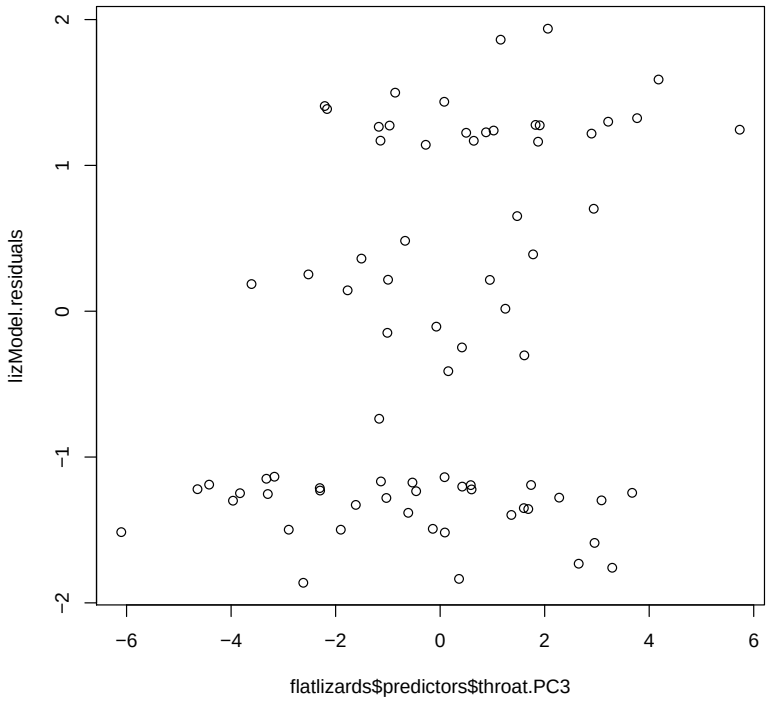

Figure 2: Lizard residuals for the simple model `lizModel`, plotted against `throat.PC3`.

```
> lizModel.residuals <- residuals(lizModel, type = "grouped")
> plot(flatlizards$predictors$throat.PC3, lizModel.residuals)
```

The residuals in the plot exhibit a strong, positive regression slope in relation to the omitted predictor variable `throat.PC3`.

6. Model search

In addition to `update()` as illustrated in preceding sections, methods for the generic functions `add1()`, `drop1()` and `anova()` are provided. These can be used to investigate the effect of adding or removing a variable, whether that variable is contest-specific, such as an order effect, or player-specific; and to compare the fit of nested models.

7. Setting up the data

7.1. Contest-specific data

The `outcome` argument of `BTm` represents a binomial response and can be supplied in any of the formats allowed by the `glm` function. That is, either a two-column matrix with the columns giving the number of wins and losses (for `player1` vs. `player2`), a factor where the

first level denotes a loss and all other levels denote a win, or a binary variable where 0 denotes a loss and 1 denotes a win. Each row represents either a single contest or a set of contests between the same two players.

The `player1` and `player2` arguments are either factors specifying the two players in each contest, or data frames containing such factors, along with any contest-specific variables that are also player-specific, such as the `at.home` variable seen in Section 3.3. If given in data frames, the factors identifying the players should be named as specified by the `id` argument and should have identical levels, since they represent a particular sample of the full set of players.

Thus for the model `baseballModel2`, which was specified by the following call:

```
R>  baseballModel2$call

BTm(outcome = cbind(home.wins, away.wins), player1 = home.team,
    player2 = away.team, formula = ~team + at.home, id = "team",
    data = baseball)
```

the data are provided in the `baseball` data frame, which has the following structure:

```
R> str(baseball, vec.len = 2)

'data.frame':        42 obs. of  4 variables:
 $ home.team:'data.frame':        42 obs. of  2 variables:
  ..$ team   : Factor w/ 7 levels "Baltimore","Boston",..: 5 5 5 5 5 ...
  ..$ at.home: num  1 1 1 1 1 ...
 $ away.team:'data.frame':        42 obs. of  2 variables:
  ..$ team   : Factor w/ 7 levels "Baltimore","Boston",..: 4 7 6 2 3 ...
  ..$ at.home: num  0 0 0 0 0 ...
 $ home.wins: int  4 4 4 6 4 ...
 $ away.wins: int  3 2 3 1 2 ...
```

In this case `home.team` and `away.team` are both data frames, with the factor `team` specifying the team and the variable `at.home` specifying whether or not the team was at home. So the first comparison

```
R> baseball$home.team[1,]

       team at.home
1 Milwaukee       1

R> baseball$away.team[1,]

     team at.home
1 Detroit       0
```

is Milwaukee playing at home against Detroit. The outcome is given by

```
R>  baseball[1, c("home.wins", "away.wins")]

  home.wins away.wins
1         4         3
```

Contest-specific variables that are *not* player-specific – for example, whether it rained or not during a contest – should only be used in interactions with variables that *are* player-specific, otherwise the effect on ability would be the same for both players and would cancel out. Such variables can conveniently be provided in a single data frame along with the `outcome`, `player1` and `player2` data.

An offset in the model can be specified by using the `offset` argument to `BTm`. This facility is provided for completeness: the authors have not yet encountered an application where it is needed.

To use only certain rows of the contest data in the analysis, the `subset` argument may be used in the call to `BTm`. This should either be a logical vector of the same length as the binomial response, or a numeric vector containing the indices of rows to be used.

7.2. Non contest-specific data

Some variables do not vary by contest directly, but rather vary by a factor that is contest-specific, such as the player ID or the judge making the paired comparison. For such variables, it is more economical to store the data by the levels of the contest-specific factor and use indexing to obtain the values for each contest.

The `CEMS` example in Section 3.4 provides an illustration of such variables. In this example student-specific variables are indexed by `student` and school-specific variables are indexed by `..`, i.e., the first or second school in the comparison as appropriate. There are then two extra sets of variables in addition to the usual contest-specific data as described in the last section. A good way to provide these data to `BTm` is as a list of data frames, one for each set of variables, e.g.,

```
R> str(CEMS, vec.len = 2)

List of 3
 $ preferences:'data.frame':        4545 obs. of  8 variables:
  ..$ student : num [1:4545] 1 1 1 1 1 ...
  ..$ school1 : Factor w/ 6 levels "Barcelona","London",..: 2 2 4 2 4 ...
  ..$ school2 : Factor w/ 6 levels "Barcelona","London",..: 4 3 3 5 5 ...
  ..$ win1    : num [1:4545] 1 1 NA 0 0 ...
  ..$ win2    : num [1:4545] 0 0 NA 1 1 ...
  ..$ tied    : num [1:4545] 0 0 NA 0 0 ...
  ..$ win1.adj: num [1:4545] 1 1 NA 0 0 ...
  ..$ win2.adj: num [1:4545] 0 0 NA 1 1 ...
 $ students   :'data.frame':        303 obs. of  8 variables:
  ..$ STUD: Factor w/ 2 levels "other","commerce": 1 2 1 2 1 ...
  ..$ ENG : Factor w/ 2 levels "good","poor": 1 1 1 1 2 ...
  ..$ FRA : Factor w/ 2 levels "good","poor": 1 2 1 1 2 ...
```

```
  ..$ SPA : Factor w/ 2 levels "good","poor": 2 2 2 2 2 ...
  ..$ ITA : Factor w/ 2 levels "good","poor": 2 2 2 1 2 ...
  ..$ WOR : Factor w/ 2 levels "no","yes": 1 1 1 1 1 ...
  ..$ DEG : Factor w/ 2 levels "no","yes": 2 1 2 1 1 ...
  ..$ SEX : Factor w/ 2 levels "female","male": 2 1 2 1 2 ...
 $ schools    :'data.frame':        6 obs. of  7 variables:
  ..$ Barcelona: num [1:6] 1 0 0 0 0 ...
  ..$ London   : num [1:6] 0 1 0 0 0 ...
  ..$ Milano   : num [1:6] 0 0 1 0 0 ...
  ..$ Paris    : num [1:6] 0 0 0 1 0 ...
  ..$ St.Gallen: num [1:6] 0 0 0 0 1 ...
  ..$ Stockholm: num [1:6] 0 0 0 0 0 ...
  ..$ LAT      : num [1:6] 1 0 1 1 0 ...
```

The names of the data frames are only used by `BTm` if they match the names specified in the `player1` and `player2` arguments, in which case it is assumed that these are data frames providing the data for the first and second player respectively. The rows of data frames in the list should either correspond to the contests or the levels of the factor used for indexing.

Player-specific offsets should be included in the formula by using the `offset` function.

7.3. Converting data from a 'wide' format

The `BTm` function requires data in a 'long' format, with one row per contest, provided either directly as in Section 7.1 or via indexing as in Section 7.2. In studies where the same set of paired comparisons are made by several judges, as in a questionnaire for example, the data may be stored in a 'wide' format, with one row per judge.

As an example, consider the `cemspc` data from the **prefmod** package (Hatzinger and Dittrich 2012), which provides data from the CEMS study in a wide format. Each row corresponds to one student; the first 15 columns give the outcome of all pairwise comparisons between the 6 schools in the study and the last two columns correspond to two of the student-specific variables: `ENG` (indicating the student's knowledge of English) and `SEX` (indicating the student's gender).

The following steps convert these data into a form suitable for analysis with `BTm`. First a new data frame is created from the student-specific variables and these variables are converted to factors:

```
R> library("prefmod")
R> student <- cemspc[c("ENG", "SEX")]
R> student$ENG <- factor(student$ENG, levels = 1:2,
+                        labels = c("good", "poor"))
R> student$SEX <- factor(student$SEX, levels = 1:2,
+                        labels = c("female", "male"))
```

This data frame is put into a list, which will eventually hold all the necessary data. Then a `student` factor is created for indexing the student data to produce contest-level data. This is put in a new data frame that will hold the contest-specific data.

```
R> cems <- list(student = student)
R> student <- gl(303, 1, 303 * 15) #303 students, 15 comparisons
R> contest <- data.frame(student = student)
```

Next the outcome data is converted to a binomial response, adjusted for ties. The result is added to the `contest` data frame.

```
R> win <- cemspc[, 1:15] == 0
R> lose <- cemspc[, 1:15] == 2
R> draw <- cemspc[, 1:15] == 1
R> contest$win.adj <- c(win + draw/2)
R> contest$lose.adj <- c(lose + draw/2)
```

Then two factors are created identifying the first and second school in each comparison. The comparisons are in the order 1 vs. 2, 1 vs. 3, 2 vs. 3, 1 vs. 4, ..., so the factors can be created as follows:

```
R> lab <- c("London", "Paris", "Milano", "St. Gallen", "Barcelona",
+           "Stockholm")
R> contest$school1 <- factor(sequence(1:5), levels = 1:6, labels = lab)
R> contest$school2 <- factor(rep(2:6, 1:5), levels = 1:6, labels = lab)
```

Note that both factors have exactly the same levels, even though only five of the six players are represented in each case. In other words, the numeric factor levels refer to the same players in each case, so that the player is unambiguously identified. This ensures that player-specific parameters and player-specific covariates are correctly specified.

Finally the `contest` data frame is added to the main list:

```
R> cems$contest <- contest
```

This creates a single data object that can be passed to the `data` argument of `BTm`. Of course, such a list could be created on-the-fly as in `data = list(contest, student)`, which may be more convenient in practice.

7.4. Converting data from the format required by the earlier BradleyTerry package

The **BradleyTerry** package described in Firth (2005) required contest/comparison results to be in a data frame with columns named `winner`, `loser` and `Freq`. The following example shows how `xtabs` and `countsToBinomial` can be used to convert such data for use with the `BTm` function in **BradleyTerry2**:

```
> library("BradleyTerry")  ## the /old/ BradleyTerry package
> ## load data frame with columns "winner", "loser", "Freq"
> data("citations", package = "BradleyTerry")
> ## convert to 2-way table of counts
> citations <- xtabs(Freq ~ winner + loser, citations)
> ## convert to a data frame of binomial observations
> citations.sf <- countsToBinomial(citations)
```

The `citations.sf` data frame can then be used with `BTm` as shown in Section 2.1.

8. A list of the functions provided in BradleyTerry2

The standard R help files provide the definitive reference. Here we simply list the main user-level functions and their arguments, as a convenient overview:

```
BTabilities(model)
glmmPQL(fixed, random = NULL, family = "binomial",
    data = NULL, subset = NULL, weights = NULL, offset = NULL,
    na.action = NULL, start = NULL, etastart = NULL,
    mustart = NULL, control = glmmPQL.control(...),
    sigma = 0.1, sigma.fixed = FALSE, model = TRUE,
    x = FALSE, contrasts = NULL, ...)
countsToBinomial(xtab)
plotProportions(win, tie = NULL, loss, player1, player2,
    abilities = NULL, home.adv = NULL, tie.max = NULL,
    tie.scale = NULL, tie.mode = NULL, at.home1 = NULL,
    at.home2 = NULL, data = NULL, subset = NULL, bin.size = 20,
    xlab = "P(player1 wins | not a tie)", ylab = "Proportion",
    legend = NULL, col = 1:2, ...)
qvcalc(object, ...)
glmmPQL.control(maxiter = 50, IWLSiter = 10, tol = 1e-06,
    trace = FALSE)
BTm(outcome = 1, player1, player2, formula = NULL,
    id = "..", separate.ability = NULL, refcat = NULL,
    family = "binomial", data = NULL, weights = NULL,
    subset = NULL, na.action = NULL, start = NULL,
    etastart = NULL, mustart = NULL, offset = NULL,
    br = FALSE, model = TRUE, x = FALSE, contrasts = NULL,
    ...)
GenDavidson(win, tie, loss, player1, player2, home.adv = NULL,
    tie.max = ~1, tie.mode = NULL, tie.scale = NULL,
    at.home1 = NULL, at.home2 = NULL)
```

9. Some final remarks

9.1. A note on the treatment of ties

The present version of **BradleyTerry2** provides no sophisticated facilities for handling tied contests/comparisons; the well-known models of Rao and Kupper (1967) and Davidson (1970) are not implemented here. At present the `BTm` function requires a binary or binomial response variable, the third ('tied') category of response is not allowed.

In several of the data examples (e.g., `?CEMS`, `?springall`, `?sound.fields`), ties are handled by the crude but simple device of adding half of a 'win' to the tally for each player involved; in

each of the examples where this has been done it is found that the result is very similar, after a simple re-scaling, to the more sophisticated analyses that have appeared in the literature. Note that this device when used with `BTm` typically gives rise to warnings produced by the back-end `glm` function, about non-integer 'binomial' counts; such warnings are of no consequence and can be safely ignored.

It is likely that a future version of **BradleyTerry2** will have a more general method for handling ties.

9.2. A note on 'contest-specific' random effects

The current version of **BradleyTerry2** provides facilities for fitting models with random effects in 'player-specific' predictor functions, as illustrated in Section 3. For more general, 'contest-specific' random-effect structures, such as random 'judge' effects in psychological studies (e.g., Böckenholt 2001), **BradleyTerry2** provides (through `BTm`) the necessary user interface but as yet no back-end calculation. It is hoped that this important generalization can be made successfully in a future version of **BradleyTerry2**.

Acknowledgments

This work was supported by the UK Engineering and Physical Sciences Research Council.

References

Agresti A (2002). *Categorical Data Analysis*. 2nd edition. John Wiley & Sons.

Bates D, Mächler M, Bolker B (2011). ***lme4**: Linear Mixed-Effects Models Using S4 Classes*. R package version 0.999375-42, URL http://CRAN.R-project.org/package=lme4.

Böckenholt U (2001). "Hierarchical Modeling of Paired Comparison Data." *Psychological Methods*, **6**(1), 49–66.

Bradley RA (1984). "Paired Comparisons: Some Basic Procedures and Examples." In PR Krishnaiah, PK Sen (eds.), *Nonparametric Methods*, volume 4 of *Handbook of Statistics*, pp. 299 – 326. Elsevier.

Bradley RA, Terry ME (1952). "Rank Analysis of Incomplete Block Designs I: The Method of Paired Comparisons." *Biometrika*, **39**, 324–45.

Breslow NE, Clayton DG (1993). "Approximate Inference in Generalized Linear Mixed Models." *Journal of the American Statistical Association*, **88**(421), 9–25.

Davidson RR (1970). "On Extending the Bradley-Terry Model to Accommodate Ties in Paired Comparison Experiments." *Journal of the American Statistical Association*, **65**, 317–328.

Dittrich R, Hatzinger R, Katzenbeisser W (2001). "Corrigendum: Modelling the Effect of Subject-Specific Covariates in Paired Comparison Studies with an Application to University Rankings." *Applied Statistics*, **50**, 247–249.

Firth D (1993). "Bias Reduction of Maximum Likelihood Estimates." *Biometrika*, **80**, 27–38.

Firth D (2005). "Bradley-Terry Models in R." *Journal of Statistical Software*, **12**(1), 1–12. URL http://www.jstatsoft.org/v12/i01/.

Firth D (2010). ***qvcalc**: Quasi-Variances for Factor Effects in Statistical Models*. R package version 0.8-7, URL http://CRAN.R-project.org/package=qvcalc.

Firth D, de Menezes RX (2004). "Quasi-Variances." *Biometrika*, **91**, 65–80.

Hatzinger R, Dittrich R (2012). "**prefmod**: An R Package for Modeling Preferences Based on Paired Comparisons, Rankings, or Ratings." *Journal of Statistical Software*, **48**(10), 1–31. URL http://www.jstatsoft.org/v48/i10/.

Heinze G, Schemper M (2002). "A Solution to the Problem of Separation in Logistic Regression." *Statistics in Medicine*, **21**, 2409–2419.

Ihaka R, Gentleman R (1996). "R: A Language for Data Analysis and Graphics." *Journal of Computational and Graphical Statistics*, **5**(3), 299–314.

Kosmidis I (2007). ***brglm**: Bias Reduction in Binary-Response GLMs*. R package version 0.5-6, URL http://www.ucl.ac.uk/~ucakiko/software.html.

Rao PV, Kupper LL (1967). "Ties in Paired-Comparison Experiments: A Generalization of the Bradley-Terry Model." *Journal of the American Statistical Association*, **62**, 194–204.

R Development Core Team (2012). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. ISBN 3-900051-07-0, URL http://www.R-project.org/.

Sham PC, Curtis D (1995). "An Extended Transmission/Disequilibrium Test (TDT) for Multi-Allele Marker Loci." *Annals of Human Genetics*, **59**(3), 323–336.

Stigler S (1994). "Citation Patterns in the Journals of Statistics and Probability." *Statistical Science*, **9**, 94–108.

Turner H, Firth D (2012). "Bradley-Terry Models in R: The **BradleyTerry2** Package." *Journal of Statistical Software*, **48**(9), 1–21. URL http://www.jstatsoft.org/v48/i09/.

Whiting MJ, Stuart-Fox DM, O'Connor D, Firth D, Bennett NC, Blomberg SP (2006). "Ultraviolet Signals Ultra-Aggression in a Lizard." *Animal Behaviour*, **72**, 353–363.

Affiliation:

David Firth
Department of Statistics
University of Warwick
Coventry
CV4 7AL, United Kingdom
E-mail: d.firth@warwick.ac.uk
URL: http://go.warwick.ac.uk/dfirth