引き分けを考慮した BT モデルの性能評価

今井寛* 松田眞一
E-Mail: matsu@nanzan-u.ac.jp

松田 (2002) で提案された引き分けを考慮した BT モデルに対して S（または R）上で動作するプログラムを作成しその性能を評価した。多くの場合は適切と思われる数値結果を示したが, 初期値に関して改良を施しても収束しにくい状況があることも判明した。

1 はじめに

松田 (2002) では Bradley-Terry モデル（以下では BT モデルと略す）に関して改良を施し, 引き分けのある対戦ゲームにおいてもそれを考慮しながら強さを求める方法を示した。(BT モデル自体に関しては竹内・藤野 (1985) 参照)

本論文では, その方法を S（または R）上で動作するプログラムとして作成し, その性能を評価することを目的とする。

2 引き分けを考慮した BT モデル

本論文では, 勝敗を考える対象をチームと呼び, 全部で m チームあるとする。第 i 番目のチームが第 j 番目のチームに対して勝つ確率, 負ける確率, 引き分ける確率をそれぞれ p_{ij}, q_{ij}, r_{ij} とすると

$$p_{ij} + q_{ij} + r_{ij} = 1, p_{ij} = q_{ji}, r_{ij} = r_{ji}$$

が成り立つ。いま, BT モデルと同様に各チームの強さ π_i というパラメータを導入し,

$$p_{ij} = \frac{\pi_i}{\pi_i + \pi_j} \cdot (1 - r_{ij})$$

というモデルが成立しているとする。一方, 引き分けについては次のように 2 つのパラメータ α, β に依存するモデルを考える。

$$r_{ij} = \alpha - \beta \left(\frac{\pi_i - \pi_j}{\pi_i + \pi_j} \right)^2$$

このモデルは 2 つのチームの強さの近さに依存して引き分けの確率が決まると考えるもので, パラメータ β の値が正であればチームの力の差が大きくなると引き分けにくく, 負であれば力が近いほど引き分けにくいことを表すことになる。

3 パラメータの推定

実際に第 i 番目のチームと第 j 番目のチームの対戦が n_{ij} 回行われた場合に第 i 番目のチームが勝つ回数の確率変数を X_{ij}, 負ける回数の確率変数を Y_{ij} , 引き分けとなる回数の

*経営学部情報管理学科

確率変数を Z_{ij} とおくとそれらは多項分布に従っていると考えられ, 次のように確率分布が定まる。

$$\Pr\{X_{ij}=x_{ij}, Y_{ij}=y_{ij}, Z_{ij}=z_{ij}; 1\leq i<j\leq m\}=\prod_{i<j}\prod\frac{n_{ij}!}{x_{ij}!y_{ij}!z_{ij}!}p_{ij}^{x_{ij}}q_{ij}^{y_{ij}}r_{ij}^{z_{ij}}$$

これに対して, 前節のモデルを導入し, 対数尤度の最大化で方程式を立てると最終的に

$$\begin{cases}\pi_i=\dfrac{T_i}{\displaystyle\sum_{j\neq i}\frac{n_{ij}-z_{ij}}{\pi_i+\pi_j}-\sum_{j\neq i}\frac{4\beta\pi_j(\pi_i-\pi_j)}{\pi_i+\pi_j}\frac{(\alpha n_{ij}-z_{ij})P_{ij}-\beta n_{ij}Q_{ij}}{\{(1-\alpha)P_{ij}+\beta Q_{ij}\}\{\alpha P_{ij}-\beta Q_{ij}\}}}\\ \alpha=\dfrac{\displaystyle\sum_{i<j}\sum z_{ij}+\beta\sum_{i<j}\sum\frac{(n_{ij}-z_{ij})Q_{ij}}{(1-\alpha)P_{ij}+\beta Q_{ij}}}{\displaystyle\sum_{i<j}\sum n_{ij}}\\ \beta=\dfrac{\displaystyle\sum_{i<j}\sum(\alpha n_{ij}-z_{ij})}{\displaystyle\sum_{i<j}\sum\frac{z_{ij}Q_{ij}}{\alpha P_{ij}-\beta Q_{ij}}}\end{cases}$$

が得られる。ただし,

$$T_i=\sum_{j\neq i}x_{ij}$$

$$P_{ij}=(\pi_i+\pi_j)^2, Q_{ij}=(\pi_i-\pi_j)^2$$

とおく。(導出過程は松田 (2002) を参照。ただし, 式中に誤植とミスがあるため最終結果が上式と異なっているが, 本論文の方が正しい。)

4 計算手順

本論文では初期値の与え方について松田 (2002) を改良し, 以下の手順でプログラムを作成した。

手順 1 初期値を決定する。

$$\pi_i=50;\quad i=1,\ldots,m,\qquad k=50m$$

$$\alpha=\frac{\displaystyle\sum_{i\neq j}\sum z_{ij}}{\displaystyle\sum_{i\neq j}\sum n_{ij}},\quad \beta=-0.05$$

手順 2 以下の関係式を用いて強さの推定値 π_i' を求める。

$$\pi_i'=\frac{T_i}{\displaystyle\sum_{j\neq i}\frac{n_{ij}-z_{ij}}{\pi_i+\pi_j}-\sum_{j\neq i}\frac{4\beta\pi_j(\pi_i-\pi_j)}{\pi_i+\pi_j}\frac{(\alpha n_{ij}-z_{ij})P_{ij}-\beta n_{ij}Q_{ij}}{\{(1-\alpha)P_{ij}+\beta Q_{ij}\}\{\alpha P_{ij}-\beta Q_{ij}\}}}$$

ただし,

$$T_i = \sum_{j \neq i} x_{ij}$$

$$P_{ij} = (\pi_i + \pi_j)^2, Q_{ij} = (\pi_i - \pi_j)^2$$

である。

手順 3 π_i' が $\sum \pi_i' = k$ を満たすように基準化する。なお、π_i' が負となった場合は, 0.1 とする。

手順 4 π_i' が次の式を満たす場合, それらを新たな π_i として手順 2 に戻って手順 2〜4 を繰り返す。

$$\sqrt{\sum \{(\pi_i - \pi_i')/50\}^2} > 10^{-8}$$

手順 5 以下の関係式より α', β' を求める。

$$\alpha' = \frac{\sum\sum_{i<j} z_{ij} + \beta \sum\sum_{i<j} \frac{(n_{ij} - z_{ij})Q_{ij}}{(1-\alpha)P_{ij} + \beta Q_{ij}}}{\sum\sum_{i<j} n_{ij}}$$

$$\beta' = \frac{\sum\sum_{i<j} (\alpha n_{ij} - z_{ij})}{\sum\sum_{i<j} \frac{z_{ij} Q_{ij}}{\alpha P_{ij} - \beta Q_{ij}}}$$

ただし,

$$P_{ij} = (\pi_i + \pi_j)^2, Q_{ij} = (\pi_i - \pi_j)^2$$

である。

手順 6 α' が自然な状況になるように以下の修正を施す。

もし $\alpha' < 0$ ならば $\alpha' = 0.00001$ とする。

もし $\alpha' > 1$ ならば $\alpha' = 0.99999$ とする。

手順 7 α', β' が次の式を満たす場合, 新たな $\alpha = (\alpha' + 9\alpha)/10$, $\beta = (\beta' + 9\beta)/10$ として手順 2 に戻って手順 2〜7 を繰り返す。

$$\sqrt{(\alpha - \alpha')^2 + (\beta - \beta')^2 + \sum \{(\pi_i - \pi_i'')/50\}^2} > 10^{-6}$$

ただし, π_i'' は前回の π_i の値である。

手順 8 得られた π_i, α, β を推定値とする。

5 S 上でのプログラム

前節の手順で S（または R）上で動作するプログラムを作成した。プログラミングする上で時間的制約から手順内の繰り返しについて手順 2〜4 は 10 回, 手順 2〜7 は 2000 回を限度とした。

```
DBT <- function(x,y,z){
      m <- sqrt(length(x))
      n <- x + y + z
      pi <- oldpi <- rep(50,m)
      t <- pi1 <- newpi1 <- rep(0,m)
      count2 <- zsum <- nsum <- 0
      P <- Q <- matrix(0,m,m)
      k <- sum(pi)
      for(i in 1:m){
         t[i] <- sum(x[i,])
         zsum <- zsum + sum(z[i,])
         nsum <- nsum + sum(n[i,])
      }
      A <- zsum/nsum
      B <- -0.05
      repeat{
         count1 <- 0
         repeat{
            for(i in 1:m){
               for(j in 1:m){
                  P[i,j] <- (pi[i] + pi[j])^2
                  Q[i,j] <- (pi[i] - pi[j])^2
               }
            }
            s1 <- s2 <- rep(0,m)
            for(i in 1:m){
               for(j in 1:m){
                  if(j != i){
                    s1[i] <- s1[i] + ((n[i,j] - z[i,j])/(pi[i] + pi[j]))
                    s2[i] <- s2[i] + ((4 * B * pi[j] * (pi[i] - pi[j]))
                                      /(pi[i]+pi[j]))
                                * (((A * n[i,j] - z[i,j]) * P[i,j])
                                  - (B * n[i,j] * Q[i,j]))
                                /(((1 - A) * P[i,j]) + (B * Q[i,j]))
                                  * ((A * P[i,j]) - (B * Q[i,j])))
                  }
               }
            }
            for(i in 1:m)
               pi1[i] <- t[i]/(s1[i] - s2[i])
            pi1sum <- sum(pi1)
```

```
   for(i in 1:m){
      newpi1[i] <- (k * pi1[i])/pi1sum
      if(newpi1[i] < 0)
        newpi1[i] <- 0.1
   }
   d1 <- 0
   for(i in 1:m)
      d1 <- d1 + (((pi[i] - newpi1[i])/50)^2)
   if(sqrt(d1) < 1e-08)
     break
   count1 <- count1 + 1
   for(i in 1:m)
      pi[i] <- newpi1[i]
   if(count1 >= 10)
     break
}
if(count2 >= 2000)
  break
s3 <- s4 <- s5 <- s6 <- s7 <- s8 <- 0
for(i in 1:m){
   for(j in 1:m){
      if(i < j){
        s3 <- s3 + z[i,j]
        s4 <- s4 + (((n[i,j] - z[i,j]) * Q[i,j])/
               (((1 - A) * P[i,j]) + (B * Q[i,j])))
        s5 <- s5 + n[i,j]
      }
   }
}
A1 <- (s3 + B * s4)/s5
for(i in 1:m){
   for(j in 1:m){
      if(i < j){
        s6 <- s6 + (A * n[i,j] - z[i,j])
        s7 <- s7 + ((z[i,j] * Q[i,j])/((A * P[i,j])
               - (B * Q[i,j])))
      }
   }
}
B1 <- s6/s7
if(A1 < 0)
```

```
        A1 <- 0.00001
      else if(A1 > 1)
        A1 <- 0.99999
      for(i in 1:m)
         s8 <- s8 + ((pi[i] - oldpi[i])/50)^2
      d2 <- (A - A1)^2 + (B - B1)^2 + s8
      if(sqrt(d2) < 1e-06)
        break
      count2 <- count2 + 1
      A <- (A1 + (9 * A))/10
      B <- (B1 + (9 * B))/10
      for(i in 1:m)
         oldpi[i] <- pi[i]
   }
   list(Strength = pi,Alpha = A,Beta = B)
}
```

このプログラムは 3 節で述べた勝つ回数, 負ける回数, 引き分ける回数の行列をそれぞれ x, y, z としたとき, それを引数として与えると強さの推定値 π が Strength で引き分けの係数 α, β が Alpha, Beta で出力される。

一方, 提案するモデルで設定した α の初期値では推定値が収束しない可能性があるため, 引き分けを考慮した BT モデルで α の初期値を入力するプログラムも用意した。なお, 上のプログラムと同様の部分については省略する。

```
DBT <- function(x,y,z,a){
   m <- sqrt(length(x))
   n <- x + y + z
   pi <- oldpi <- rep(50,m)
   t <- pi1 <- newpi1 <- rep(0,m)
   count2 <- 0
   P <- Q <- matrix(0,m,m)
   k <- sum(pi)
   for(i in 1:m)
      t[i] <- sum(x[i,])
   A <- a
   B <- -0.05
   repeat{

      <<省略>>

   }
   list(Strength = pi,Alpha = A,Beta = B)
}
```

6　データ解析

作成したプログラムで J リーグの 2001 年, 2002 年度のリーグ戦のデータに関して分析する。データは J1 のファーストステージ, セカンドステージ, 総合成績と J2 の第 1 節〜第 22 節, 第 23 節〜第 44 節, 総合成績, さらに 2002 年度の J2 のデータ以外を延長戦を引き分けとした場合の全 21 通りについて調べた。

実行結果は 4 つの場合に分けることができる。(S バージョンのプログラムは実行時間がかかるため, 繰り返し回数があまり上げられなかったので Java バージョンも併用して用いた。Java バージョンでは繰り返し回数を最大内側 1500 回, 外側 5000 回とした。)

1. S バージョンでデータに依存した α の初期値で収束した場合
    - 2001 年度 J1 ファーストステージ $[\alpha = 0.058, \beta = 0.000]$
    - 2001 年度 J1 セカンドステージ $[\alpha = 0.087, \beta = -0.189]$
    - 2001 年度 J1 総合成績 $[\alpha = 0.113, \beta = 0.118]$
    - 2001 年度 J1 ファーストステージ (延長引き分け) $[\alpha = 0.235, \beta = -0.264]$
    - 2001 年度 J1 総合成績 (延長引き分け) $[\alpha = 0.296, \beta = 0.000]$
    - 2001 年度 J2 総合成績 (延長引き分け) $[\alpha = 0.307, \beta = 0.171]$
    - 2002 年度 J1 ファーストステージ $[\alpha = 0.075, \beta = 0.000]$
    - 2002 年度 J1 セカンドステージ $[\alpha = 0.051, \beta = -0.042]$
    - 2002 年度 J1 ファーストステージ (延長引き分け) $[\alpha = 0.227, \beta = -0.145]$
    - 2002 年度 J1 セカンドステージ (延長引き分け) $[\alpha = 0.269, \beta = 0.000]$
    - 2002 年度 J2 第 1 節〜第 22 節 $[\alpha = 0.401, \beta = 0.365]$
    - 2002 年度 J2 総合成績 $[\alpha = 0.407, \beta = 0.486]$
2. Java バージョンでデータに依存した α の初期値で収束した場合
    - 2001 年度 J2 第 1 節〜第 22 節 $[\alpha = 0.057, \beta = -0.100]$
    - 2001 年度 J2 第 23 節〜第 44 節 (延長引き分け) $[\alpha = 0.344, \beta = 0.237]$
    - 2002 年度 J1 総合成績 (延長引き分け) $[\alpha = 0.276, \beta = 0.000]$
3. どちらかのバージョンで α の初期値を再設定することで収束した場合
    - 2001 年度 J1 セカンドステージ (延長引き分け) $[\alpha = 0.246, \beta = -0.129]$
    - 2001 年度 J2 総合成績 $[\alpha = 0.115, \beta = 0.087]$
    - 2001 年度 J2 第 1 節〜第 22 節 (延長引き分け) $[\alpha = 0.297, \beta = 0.059]$
    - 2002 年度 J1 総合成績 $[\alpha = 0.142, \beta = 0.223]$
    - 2002 年度 J2 第 23 節〜第 44 節 $[\alpha = 0.000, \beta = -0.591]$
4. 収束しなかった場合
    - 2001 年度 J2 第 23 節〜第 44 節

2001 年度 J2 第 23 節〜第 44 節のデータはかなり収束に近づいて終了条件によっては収束する可能性もある。

S バージョンで収束したのは全体の 12/21 であり, Java バージョンを用いても 15/21 であった。S バージョンで収束せずに計算に 2000 回までかかったものは PC だと 1 時間以上も時間を要することとなった。計算手順をさらに工夫して実行時間が短くなるようにすべきであろう。(なお、収束した 12 通りのうちでも 4 通りは 1000 回を超えたので 30 分以上かかっている。)

Java バージョンでは内側の繰り返し回数を最大 1500 回としたが 10 回との違いはそれほど大きくはなかった。収束が早い場合どちらであっても収束回数に大差はなかった。逆に 10 回の方が早く収束する場合もあり, 多ければ良いというものでもないようだった。

α の再設定が必要であったり収束しなかったりした状況について, 任意の 1 チームを除いて解析にかけてみるなどして収束状況や収束した場合の推定値について調べてみると, その原因として

1. カモ・苦手関係が収束を妨げていた。
2. β の値が逆の結果を示すチームが収束を妨げていた。
3. 強さの近さに関係なく引き分ける, 引き分けやすいチームの存在が収束を妨げていた。

といった理由が考えられた。その結果, 特に β に関して正と負の間を行き来する状況が見られたのであろう。収束条件によってはその全く正反対の値に収束したかのように終了し, とても不安定であった。早く収束したものはよいとしてそうでない場合は慎重に扱う必要があるだろう。

7 考察

7.1 引き分け係数に関する考察

引き分け係数のうち α に関して特に気になるのは 2002 年度 J2 第 23 節〜第 44 節である。理論上の制約から α は 0 と 1 の間であるとしているが, この条件に抵触しながら収束したこのデータは安定しているとはいえないであろう。なお, 収束しなかった 2001 年度 J2 第 23 節〜第 44 節もこのタイプのデータに属する。以下の考察ではこのデータを除いて行うこととする。

引き分け係数の β に関しては引き分けがどのような状況で起こっているかを考える上でその符号が重要である。収束した 20 通りのうち 2002 年度 J2 第 23 節〜第 44 節を除いた 19 通りについて β の符号についてまとめると表 1 のようになる。

表 1: β の符号

符号	S（うち総合）	Java（うち総合）	再設定（うち総合）	合計
+	4 (3)	1 (0)	3 (2)	8 (5)
0	4 (1)	1 (1)	0 (0)	5 (2)
−	4 (0)	1 (0)	1 (0)	6 (0)

表 1 を見て分かるように全体としての β の符号は偏りが見られない。しかし, データの種類を見ると差がある。負の値になるのはすべて前半後半に分けたものであり, 力が近いほど引き分けにくいということを表している。その大半は J1 のデータであることから 1 シーズン 2 ステージ制であることと J2 への降格制度が影響していると考えられる。各チームとも常に上位チームであれば優勝争い, 下位チームであれば降格争いと実力が近いチームに勝てるかどうかが順位に大きな影響を与えるためであるといえる。力の離れているチームとの引き分けの確率が高いことは矛盾しているように感じるが, 守りに入られるとゴールできないからであると思われる。

一方, 総合成績のデータは β の値が正または 0 となっている。これは長丁場の間にチームの調子が変わってしまうので全体としての実力が反映するか関係なくなってしまうためと思われる。逆に, 前半後半に分けた場合は負になることから現在の調子や目先の順位を気にして戦っているとも言える。また, J2 で前半後半であっても β の値が正となる場合があるのはどのチームも決定力に欠けるせいかもしれないが, 前半後半といっても 22 節もありその調子が保てないことの方が大きいであろう。

このように引き分け係数は全体として役立っているが, 常に安定した傾向を示すものではないことが伺える。

7.2 手順に関する考察

α の初期値設定は推定値を収束させる上で重要な部分である。松田 (2002) では α の初期値を固定しているがそのせいで推定値が収束しない場合があった。そこで本論文では初期値に全試合での引き分けの割合を設定し, データごとに依存した初期値設定をするようにした。その結果多くの場合で自動的に収束することができた。自動では収束せず α の値を再設定した場合であっても, 収束したときの α の値と元の初期値を比較するとその差は ± 0.04 以内であり, 初期値として適切な値を設定できているといえる。

一方, β の設定については固定したままであったが, 表 1 をみて分かるように初期値設定の影響はそれほど受けていない。収束の早さに影響がある可能性は見られるため改良できる余地はあるだろうが, 現段階ではこのままでもよいと考えられる。

初期値と終了条件についてはより適切なものがあるかもしれないが, 本論文の設定でかなりのデータに対応できているといえる。

7.3 BT モデルとの比較

強さの推定値について提案する方法と従来の BT モデルを比較する。

- 引き分けを 0.5 勝 0.5 敗とした場合との比較
  上位チームに比べ下位チームのほうが変化が大きいことが分かる。これは引き分けを全て 0.5 勝 0.5 敗とすることが下位チームに対して得に働いていたためであるといえる。

- 引き分けを除いた場合との比較
  ほとんど変化のないチームと上位チームで弱くなっているチームがある。これは力の離れたチームに対して痛い引き分けを喫したかどうかが影響していると考えられる。

従来の BT モデルでは引き分けを除いた場合は当然であるが, 0.5 勝 0.5 敗とした場合でも強さに対する引き分けの影響をうまく取り込めていないといえる。

表 2: J1(2002 年度セカンドステージ）の強さの推定結果

順位	チーム名	引分を 0.5 勝 0.5 敗	引分を除く	提案する方法
1	磐田	237.1	236.6	216.4
2	G 大阪	75.7	75.2	73.6
3	鹿島	56.4	55.9	55.7
4	東京 V	56.4	60.9	70.2
5	FC 東京	42.6	42.1	42.7
6	横浜 M	49.0	47.6	45.3
7	京都	42.6	42.1	42.7
8	浦和	49.0	49.1	48.5
9	柏	37.1	41.5	49.8
10	神戸	32.4	34.9	42.3
11	市原	24.5	24.1	25.2
12	清水	24.5	24.1	25.2
13	名古屋	21.2	19.8	19.2
14	広島	24.5	21.8	19.2
15	仙台	15.7	14.4	14.0
16	札幌	11.3	9.8	9.8
	α	—	—	0.051
	β	—	—	−0.042

しかし, 引き分けを除いた場合と 0.5 勝 0.5 敗とした場合でどちらの方が強さが提案した方法に近いかというと, 結果からは引き分けを除いた場合のほうが提案した方法に近かったといえる。その理由は β の値があまり大きくないことが多く, 引き分けの強さに与える影響が小さいためである。

例として 2002 年の J1 のセカンドステージの結果を表 2 に示す。これまで述べてきた特徴がよく出ている。このデータ独特なものとしては東京 V が影響をかなり受けている様子が見て取れる。これは全体として延長にもつれ込んでから勝ちを拾ったことによるもので延長を引き分けと考えた分析では順位通りの強さの結果であることから勝ち点制度をうまく使いこなせなかったチームといえる。

8 おわりに

本論文では S 上で引き分けを考慮した BT モデルのプログラムを組み, その性能を評価することを目的とした。作成したプログラムは一部, 収束しないデータも出たがほぼ満足できる結果であったといえる。また複数のデータを分析していく中で提案する方法が引き分けの起きる状況を得ることができるだけでなく, より良い強さの推定ができていることが確

認できた。

今回のモデルでは強さの近さに応じて引き分ける確率が決まるというもので引き分けやすいチームなど, 特定のチームの分析はできていない。今後の課題としてはこの引き分けやすいチームをどのように扱うかという問題があり, 引き分けやすいチームは引き分け係数の α, β には影響はないと考え, 強さの推定にのみデータを加えるなど工夫が必要であると考えられる。

参考文献

竹内啓・藤野和建 (1985): “スポーツの数理科学— もっと楽しむための数字の読み方 —”, 共立出版.

松田眞一 (2002): Bradley-Terry モデルの改良, “アカデミア 数理情報編”, **2**, 1-7.