Athens Journal of Sports - Volume 5, Issue 2 – Pages 95-114

Simulation of a Probabilistic Model for Multi-Contestant Races

By Konstantinos Gakis[*]
Panos Pardalos[†]
Chang-Hwan Choi[‡]
Jae-Hyeon Park[+]
Jiwun Yoon[•]

Predictions of sports games have been recognized as an important area of study for their economic significance. Most models for such games cover two-player games and the resulting championships or study individual players or teams and their resulting comparative position. In this paper, we elaborate on a model for such multi-contestant races based on order statistics of the negative exponential distribution for race times. The games involve only a fixed number of athletes out of a broader pool of several athletes. Several large samples of games of ten athletes that play four at a time are generated, and Maximum Likelihood Estimators are calculated for the relative dominance parameters of the athletes. The estimated parameters are then used to test the predictive validity of the model. The results are discussed. Subsequently the model is modified for Erland-2 distributed race times. New samples are generated and the predictive validity of the new model is discussed. The paper concludes with a discussion of future research directions for improving the predictive validity and establishing confidence levels.

Keywords: *Ranking, Joint Exponential Distribution, Order Statistics, Multi-contestant Races*

Introduction

Predictions of sports games have been recognized as an important area of study for their economic significance. The majority of models for such games cover two-player games and the resulting championships or study individual players or teams and their resulting comparative position. Most popular sports involve predictions of the final score, by fans and bookmakers, making necessary the creation of a performance index for athletes or teams.

[*]Adjunct Assistant Professor, Department of Industrial & Systems Engineering, University of Florida, USA.
[†]Distinguished Professor, Department of Industrial & Systems Engineering, University of Florida, USA.
[‡]Post-Doctoral Researcher, Kinesmetrics Laboratory & Center for Performance and Sports Analysis, Korea National Sport University, Korea.
[+]Associate Professor, Kinesmetrics Laboratory & Center for Performance and Sports Analysis, Korea National Sport University, Korea.
[•]Research Professor, Center for Sports and Performance Analytics, Korea National Sport University, Korea.

https://doi.org/10.30958/ajspo.5-2-1 doi=10.30958/ajspo.5.2.1

A "multi-contestant" race is one where the athletes run to finish first by achieving the minimum time. Gakis et al. (2016) have presented a discussion of the general problem and have provided insight to the problem. Among the main points has been that most of the research and the literature have been concerned with two contestant games, in many cases with a score, where one wins and one loses or there can be a tie. Other kinds of sports are not based on one-to-one matches and the results are a classification. These are characterized as "multi-contestant races" or "games." For such games, prediction models become more complex, and usually they are based on physiological information about athletes. However, often it is necessary to predict results based only on easily available data, such as previous results, keeping a reliable model.

From a theoretical standpoint, multi-contestant races are difficult to study because they involve several random variables (the times or scores of the athletes), which in general cannot be assumed to be identically or independently distributed. The winner is usually the best (maximum or minimum) scorer. Thus the problem, in theoretical terms, is one of order statistics. Gakis et al. (2016) presented two theoretical models, which present a series of challenges and bring the promise of good predictions if further elaborated and appropriately applied.

The current work builds upon the theoretical model for independently (but not identically) distributed race times that follow an exponential distribution. Using well-established results about the distribution of the order statistics of the exponential distribution, it is shown that the proposed estimation approach can deliver very good results in predicting then ranking or relative strength of the athletes and thus predicting the winners of races.

In the next section, we present the literature review, where we connect our work to previous work on the same topic.

In section 3 we present the general methodological foundation of our work and in section 4 we present the specific model that has been employed and simulated.

In section 5 we describe our simulation experiments and in section 6 we present our conclusions.

Finally, in section 7 we discuss ongoing research and directions for potential future research that will expand on the presented results.

Literature Review

The literature on sports and games is very rich. We can categorize the literature in several ways, for instance depending on the types of models they are using, the objects they study, the purpose they serve, etc. Our current work is focused on predictions or forecasting of outcomes of sports. Rich literature exists to cover other topics, such as economics, valuation of athletes and teams, etc.

Sports forecasting is a diverse and rich field. What we are looking here is the *forecasting of game or contest outcomes*, not the return on investment or

the income that can be generated by teams, athletes or tournaments. Stekler, Sendor and Verlander (2010) present a concise and comprehensive account of the issues in sports forecasting. They have recognized that there are three approaches in making forecasts for sports:

(1) Betting markets
(2) Models
(3) Experts

They review all three approaches for several types of sports and they present a comparison of their results, to the extent that this can be established through existing bibliography. Here, we limit ourselves to item (2), i.e. models. It should be noted that (1) and (3) are qualitative approaches with significant weight and predicting ability. Of the sports studied in this work only one falls into the category of multi-contestant games, namely that of horseracing. Other sports reviewed are: baseball, American football, basketball and soccer.

The forecasting may focus on a single game, or it may contain a series of games that may constitute a tournament or a championship. In the present work, we limit ourselves to forecasts of individual events (games, races or contests).

The forecasting analytical models are mostly based on past performance data. It has already been said that multi-contestant games have been much less studied that two-player games. Interestingly, horseracing is the type of multi-contestant sport that has attracted more attention, presumably due to its financial importance in betting markets. Lessman, Sung and Johnson (2010) have recently presented a model for predicting outcomes in horseracing, which can be used for predictions in all multi-contestant sports. The model they present is based on a much earlier work by McFadden (1974), which in turn contains propositions presented in Luce and Suppes (1965). From these references, we can see that sports forecasting models can be drawn for a very broad spectrum of disciplines ranging from econometrics to quantitative or mathematical psychology. In the classical work of McFadden (1974), the problem tackled is that of qualitative choice and it is coined as "conditional logit estimation." So, Lessman, Sung and Johnson (2010) see their problem as a problem of choice behavior: the eventual outcome of a multi-contestant race is the result of successive "choices." Starting from an initial set of n contestants, the winner is chosen, thus leaving n-1 contestants competing and successively reducing the number of contestants, until all contestants are ranked. The same problem seen from a probabilistic point of view can be seen as a problem of ranking order statistics. In an earlier work, Harville (1973) had presented a similar simpler approach for horse races, based on the same assumption of successive choice, without providing any further analysis of the factors that lead to winning. Chapman and Staelin (1982) are using the same basis in a different context to discuss a stochastic utility model and address the issue of the data supporting data set. The methodology of McFadden (1974) allows for a further analysis or explanation of the parameters of the race, such as the rider, the track etc.

The research is most often connected to assessing the utility of a race in conjunction to betting schemes as is done for example in the work of Ali (1977). In a more detailed and extensive work, Bolton and Chapman (2008) discuss the use of a multinomial logit model for the purpose of achieving positive returns at the track.

The major problem that arises from employing this approach is the estimation of the underlying parameters for the calculation of the probabilities for winning in each race. To employ the model of McFadden (1974), we need a rich data set that will allow the use of the *maximum likelihood estimation (MLE)* method. Johnson, Jones and Tang (2006) present an extensive discussion and demonstrate the use of the maximum likelihood estimation method for the purpose of estimating the parameters in the multinomial logit model of McFadden (1974).In their work, they employed *orthogonal polynomial expansion* of order 3 to estimate the parameters of the model. Subsequently, they split the data set of 1200 actual race outcomes into two subsets of 800 and 400 outcomes. The first subset was used for the estimation of the parameters and the second set for the testing and verification of the model.

General Description

As it has already been stated, the outcome of a multi-contestant race or game is basically a ranking problem. Let's assume that we have seven athletes competing in a race. The outcome will depend on the individual performance of each athlete. Let's say that the race outcomes are *running times*, in which case the winner is the athlete that comes first, i.e. the one that achieves the minimum of the times.

So, let $X_i, i = 1, \dots, m$ (assuming m athletes are competing) be the times (which hare random variables) achieved by the athletes in a particular game. Then, the probability of a particular outcome would be expressed as, for e.g. $m = 4$

$$\Pr\{X_1 < X_2 < X_3 < X_4\}$$

or the probability of athlete 1 being the first

$$\Pr\{\text{athlete1 terminates first}\} = \Pr\{X_1 < X_2, X_3, X_4\}$$

The simplest case would be to assume that running times are independently distributed. Then the fact that several athletes compete in a race is really irrelevant. What would matter is just individual performance. From experience we know that this is not true. The fact that athletes are competing against each other makes a difference in the running times and thus the assumption of independence cannot be realistic. We clearly need a more complex assumption. Since for the prediction we only need probability for the relative performance, i.e. the ranking and not the absolute times, we can assume as an approximation that the

simultaneous presence of many athletes in the same competition just reduces the time scale but not the final ranking probabilities. So, when our sample from races is large and we have results for all possible combinations of athletes from the pool, we can build a prediction model that would yield good results in terms of predictive power. If this is the case, and if we have races of *m* athletes out of a pool of *n*>*m*, then to express the probability of a specific outcome it would suffice to know the relative strength of each athlete in the form of a *dominance* vector $D=\{d_1,\ldots$ that not only *ranks* the contesting athletes, but quantifies the dominance as well.

Knowledge of the dominance vector can lead to the calculation of the probabilities of outcomes in races with specific athletes.

The Conditional Logit Model

Here we present the Conditional Logit Model as was presented by McFadden (1974) and was utilized by Lessman, Sung and Johnson (2010).

Let $S=\{\mathbf{x}_i^j\}_{i=1,j=1}^{n_j,R}$ denote a dataset of *R* past races, where $\mathbf{x}_i^j \in \Re^m$ represents the *m* independent characteristic variables (such as the rider) of a single contestant *i* in a race *j*.

Also, let $\boldsymbol{\beta}$ be the vector of coefficients that measure the relative contribution of the independent characteristic variables of contestant-race combination.

Then we define a random variable Y_k^j that expresses the probability of contestant *k* winning race *j* as

$$Y_k^j=\begin{cases}1 \text{ if } \boldsymbol{\beta}\mathbf{x}_k^j+\varepsilon_k^j>\boldsymbol{\beta}\mathbf{x}_i^j+\varepsilon_i^j \;\forall\, k,i=1,\ldots,n_j,k\neq i\\ 0 \text{ otherwise}\end{cases}$$

And ε_i^j is an error factor. If the error factors are independent and identically distributed according to the double exponential distribution, the probability p_k^j of contestant *k* winning race *j* is given by the following conditional logit function:

$$p_k^j=\frac{\exp(\boldsymbol{\beta}\mathbf{x}_k^j)}{\sum_{i=1}^{n_j}\exp(\boldsymbol{\beta}\mathbf{x}_i^j)}$$

The values of vector $\boldsymbol{\beta}$ are the ones estimated by means of a *maximum likelihood estimation* procedure.

The model has the fundamental advantage that it allows for the estimation of the probabilities for the specific composition of the specific race. Assuming that the underlying characteristics remain stable, then the model can describe sufficiently the race outcomes.

Next, assuming that the second position is a choice among the remaining contestants with no dependence on the first winner, then the probability of contestant *k* winning race *j* is given by the following conditional logit function:

$$p_k^j = \frac{\exp(\boldsymbol{\beta}\mathbf{x}_k^j)}{\sum_{i=1}^{n_j-1}\exp(\boldsymbol{\beta}\mathbf{x}_i^j)}$$

Clearly, a simplification of the problem is to consider the factors $\exp(\boldsymbol{\beta}\mathbf{x}_i^j)$ as constant parameters, not depending on the race, but rather only on the contestant. In the next section, we show how some simple assumptions can lead to a simplified model that can provide a way to calculate the probabilities in the absence of more information or a deeper analysis of the underlying success factors.

It should be noted that the model conditional logit model was developed for a general class of qualitative choice behavior model and has been applied for horse races in the existing bibliography. Horse races have some easily discernible characteristics e.g. horse-riders, tracks etc. which may not be as easy to recognize in the case of other sports, e.g. bicycle races, where there is no underlying reason to connect a bicyclist with the bicycle, the track, the season or other influencing factors.

Independently Distributed Exponential Race Times

The Basic Problem

We have a pool of $N, N = 1,2,...$ athletes competing in groups of $M, M = 1,...,N$ athletes each time. We have data results from $K, K >> N$, races.

In this first approach we assume independently and identically distributed negative exponential random variables ("expo r.v.'s") for the individual race time of each athlete. Say the individual parameters of the distributions are $i, i = 1,...,N$.

With knowledge of the actual ranks, without knowing the actual race times, we can obtain an estimate of the relative rank. We know that when comparing *M* iid expo r.v.'s, then the probability of 1 coming ahead of the other is

$$p_{[1]} = \frac{\lambda_{[1]}}{\lambda_{[1]} + \lambda_{[2]} + \cdots + \lambda_{[M]}} \qquad (1)$$

where $\lambda_{[n]}$ denotes the distribution parameter of the athlete that ranked *n*-th.

Thus, the *likelihood* of a particular outcome is

$$p_{[1,\ldots,M]} = \frac{\lambda_{[1]}}{\lambda_{[1]} + \lambda_{[2]} + \cdots + \lambda_{[M]}} \frac{\lambda_{[2]}}{\lambda_{[2]} + \cdots + \lambda_{[M]}} \cdots \frac{\lambda_{[M-1]}}{\lambda_{[M-1]} + \lambda_{[M]}} \qquad (2)$$

For a discussion of the order statistics of the exponential distribution and a proof of the above, the reader is referred to Ahsanullah, Nevzorov andShakil (2013).

Knowing the values of the parameters $\lambda_{[.]}$, we can obtain the probability of each outcome or just the first athlete in each race. Inversely, knowing several outcomes of races (without knowing the underlying scores or race times), we can estimate relative values for the parameters $\lambda_{[.]}$, to maximize the likelihood of all the outcomes.

Thus, in our approach we generate random outcomes of games (with given distributions and parameter values) and we proceed to estimating the parameters. The comparison between the original values and the estimated ones give us an indication of the ability of our approach to generate good predictions of races.

If we generate *n* races and we denote with $p^j_{[1,\ldots,M]}, j = 1,\ldots,N$ the likelihood of each race, then the likelihood function for the simulation run is

$$f^n(\boldsymbol{\lambda}) = \prod_{j=1}^{n} \frac{\lambda^j_{[1]}}{\lambda^j_{[1]} + \lambda^j_{[2]} + \cdots + \lambda^j_{[M]}} \frac{\lambda^j_{[2]}}{\lambda^j_{[2]} + \cdots + \lambda^j_{[M]}} \cdots \frac{\lambda^j_{[M-1]}}{\lambda^j_{[M-1]} + \lambda^j_{[M]}}$$

and the log-likelihood function is

$$F(\boldsymbol{\lambda}) = \log f^n(\boldsymbol{\lambda}) = \sum_{j=1}^{n} \sum_{k=1}^{M-1} \left(\log \lambda^j_{[k]} - \sum\nolimits_{m=k}^{M} \log \lambda^j_{[m]} \right)$$

It should be noted that the vector of the values $\lambda_{[.]}$ is the dominance vector *D*, discussed in Section 3.

The values of the estimator vector $\boldsymbol{\lambda}^*$ are the ones that maximize $F(\boldsymbol{\lambda})$. A log-likelihood function can be calculated for the first athlete only in each race as follows:

$$F(\boldsymbol{\lambda}) = \log f^n(\boldsymbol{\lambda}) = \sum_{j=1}^{n} \left(\log \lambda^j_{[1]} - \sum\nolimits_{m=1}^{M} \log \lambda^j_{[m]} \right)$$

This log-likelihood function can be used for computational simplicity although it should not be expected to yield more accurate results.

The Simulated Problem

In our simulation, we considered the case of a pool of ten (10) athletes, running in groups of four (4) athletes each time. We generate 500 races with random participation of athletes with each probability. Race times (exponentially and independently distributed) are generated based on assumed values of the parameters$\lambda_{[.]}$according to the following table:

λ(1)	**λ(2)**	**λ(3)**	**λ(4)**	**λ(5)**	**λ(6)**	**λ(7)**	**λ(8)**	**λ(9)**	**λ(10)**
1.0000	0.9091	0.8333	0.7692	0.7143	0.6667	0.6250	0.5882	0.5556	0.5263

The first 400 races are used to generate Maximum Likelihood Estimators (MLEs). The estimators are always estimated relative to that of the first athlete, whose value is set to 1. The last set of 100 values is used to compare the value of the Likelihood Function for the *first-ranking athlete* in each race with the assumed and the estimated parameter values to appraise the predictive power of the estimators.

We run a total of 100 simulation runs and we are additionally calculating the percentage of times that athlete 1 is the dominant athlete, i.e. there is no other athlete with a parameter value higher than 1. Finally, we calculate the average estimated parameter values, we order them and we compare the rankings with the original rankings. The number of permutations needed to recover the original order is a measure of performance of the estimation since it reflects the degree to which we have recovered the original order.

Furthermore, we repeat the above process with times that follow the Erlang-2 and Erlang-4 distributions, with the same -parameter values.

The Simulation Program

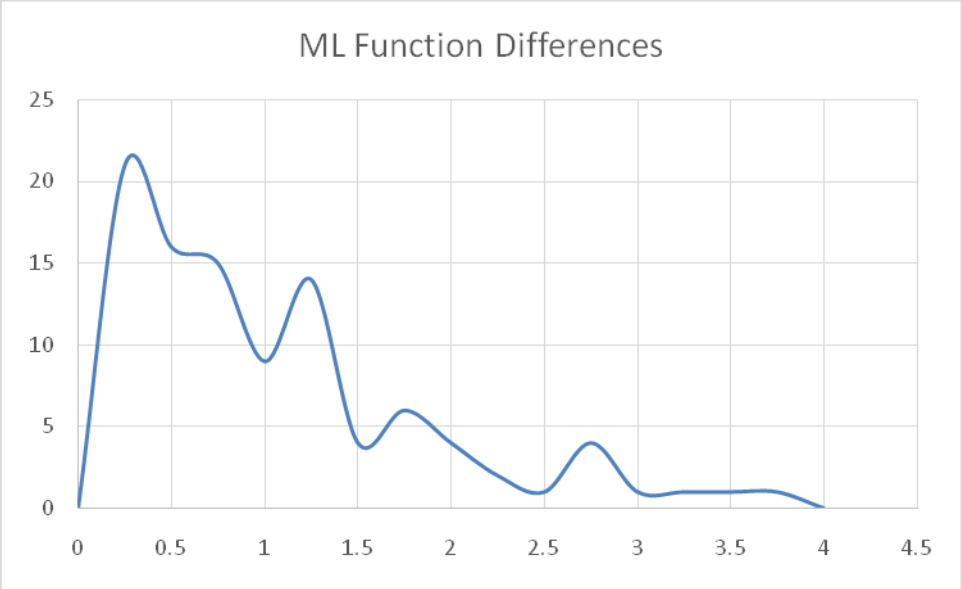
The simulation was performed using MS Excel workbook with the simulation runs programmed with VBA and the MLEs obtained using the Solver add-in.

The simulation program consists of one main procedure and four sub-routines, which are given in the Appendix: a subroutine for initialization, a subroutine for the repetition of the simulations, a subroutine for the generation of the simulated game results and a subroutine for the calculation of the MLEs. The subroutine for the generation of race times is different for the exponential times and for Erlang-*k*times, although *k* is entered as an integer parameter and in general can cover the case of exponential distribution for value $k=1$.

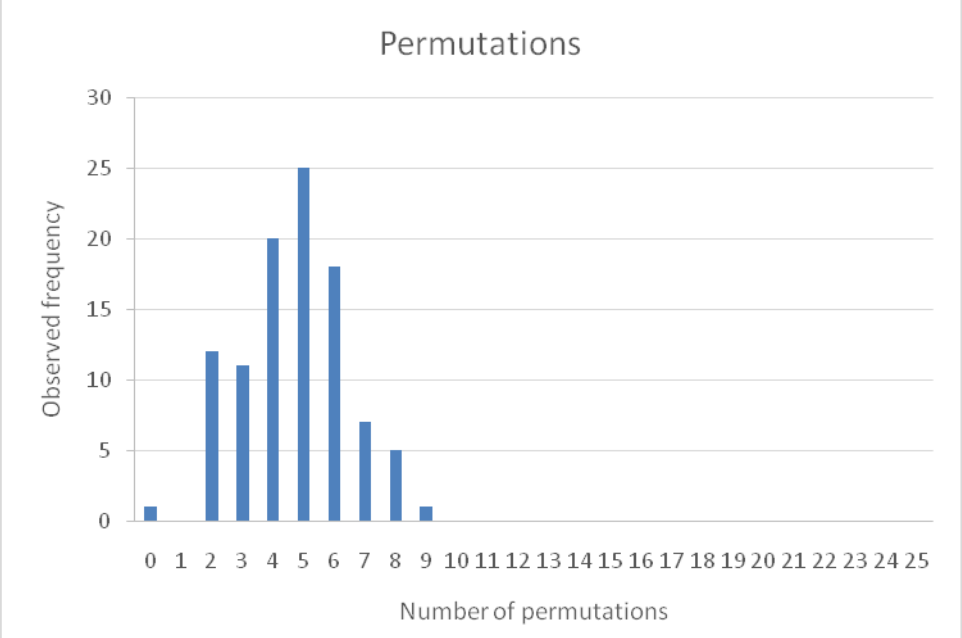
Results of the Simulation Runs
Exponentially Distributed Race Times

For exponentially distributed race times we obtain the following results:

- Athlete 1 was found to be the best athlete in 68% of the cases.
- The ML function with the estimated values was very close to the one with the original values.



- The resulting ordering of athletes was relatively close to the original.



The mean permutations number is 4.68.

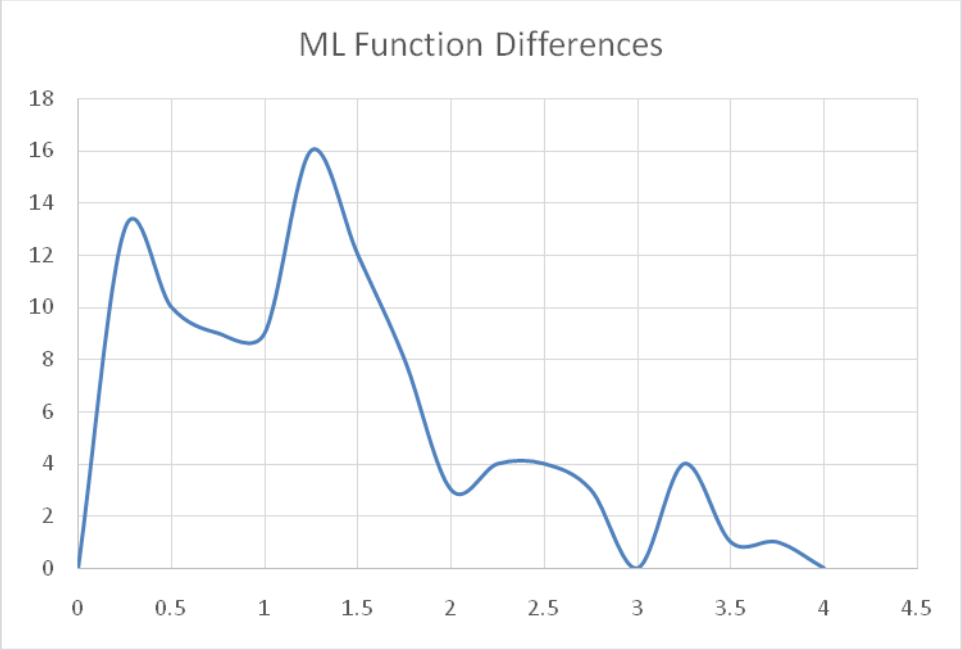
Table of results:

ANALYSIS	λ(1)	λ(2)	λ(3)	λ(4)	λ(5)	λ(6)	λ(7)	λ(8)	λ(9)	λ(10)
Real	1	0.9091	0.8333	0.7692	0.7143	0.6667	0.625	0.5882	0.5556	0.5263
Means	1	0.9115	0.8724	0.7743	0.7224	0.6816	0.629	0.5893	0.5592	0.5266
Median	1	0.9185	0.8682	0.7564	0.7047	0.6768	0.6094	0.576	0.5585	0.5223
St. Dev. (sample)	0	0.1371	0.1278	0.1226	0.1123	0.1034	0.1013	0.0918	0.0797	0.0873
Max	1	1.3327	1.2278	1.0834	1.1673	0.9702	0.9046	0.8272	0.7212	0.843
Min	1	0.6496	0.6032	0.5259	0.5387	0.4257	0.444	0.3839	0.3744	0.3282
Range	0	0.6831	0.6246	0.5574	0.6286	0.5445	0.4606	0.4432	0.3468	0.5148
Range/Mean	0	0.7494	0.7159	0.7199	0.8702	0.7989	0.7322	0.7521	0.6203	0.9776

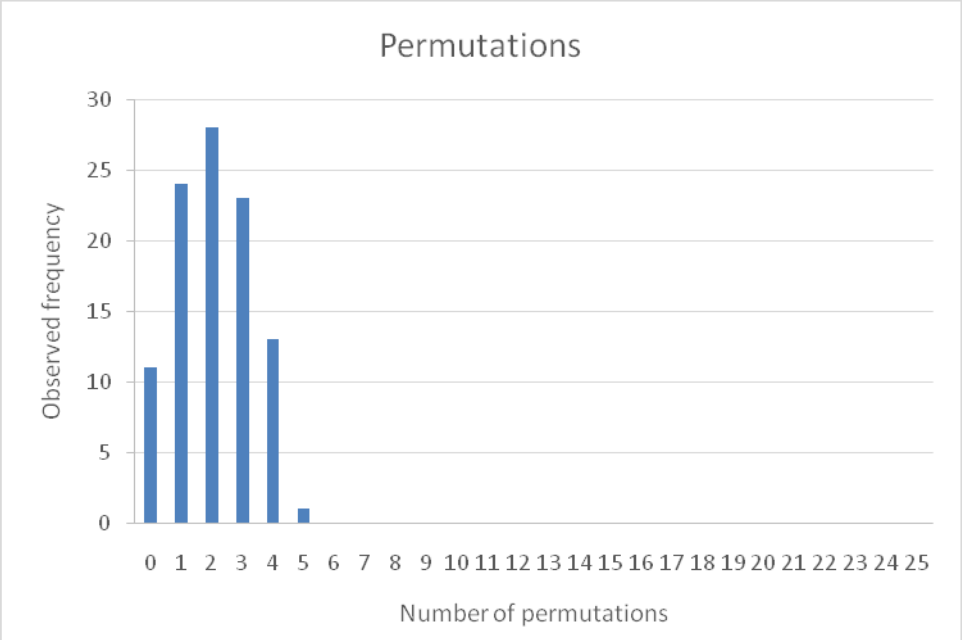
Erlang-2 Distributed Race Times

For Erlang-2 distributed race times we obtain the following results:

- Athlete 1 was found to be the best athlete in 91% of the cases.
- The ML function with the estimated values was very close to the one with the original values.



- The resulting ordering of athletes was relatively close to the original.



The mean permutations number is 2.06.

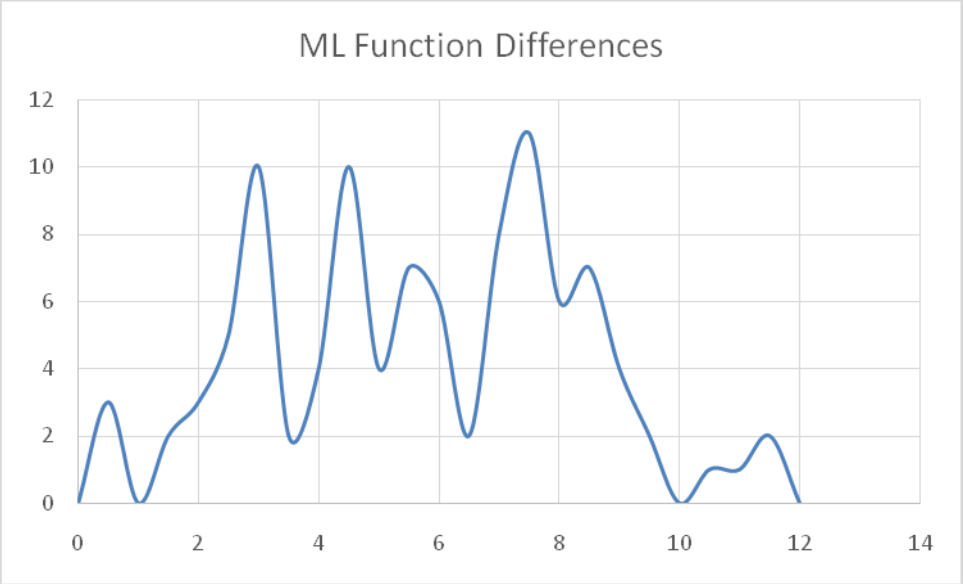
Table of results:

ANALYSIS	λ(1)	λ(2)	λ(3)	λ(4)	λ(5)	λ(6)	λ(7)	λ(8)	λ(9)	λ(10)
Real	1	0.8696	0.7692	0.6897	0.625	0.5714	0.5263	0.4878	0.4545	0.4255
Means	1	0.8228	0.6776	0.5803	0.496	0.4383	0.382	0.3497	0.3162	0.283
Median	1	0.8079	0.6782	0.5663	0.4944	0.4292	0.3742	0.3426	0.3092	0.275
St. Dev. (sample)	0	0.1249	0.1096	0.0954	0.083	0.0657	0.0633	0.0538	0.055	0.0483
Max	1	1.1757	0.9865	0.8243	0.7256	0.6422	0.5836	0.5209	0.5428	0.4127
Min	1	0.5732	0.4649	0.4054	0.339	0.3116	0.257	0.2348	0.2073	0.1903
Range	0	0.6025	0.5215	0.4189	0.3866	0.3307	0.3266	0.2861	0.3355	0.2224
Range/Mean	0	0.7322	0.7697	0.7219	0.7795	0.7544	0.855	0.8181	1.0611	0.7859

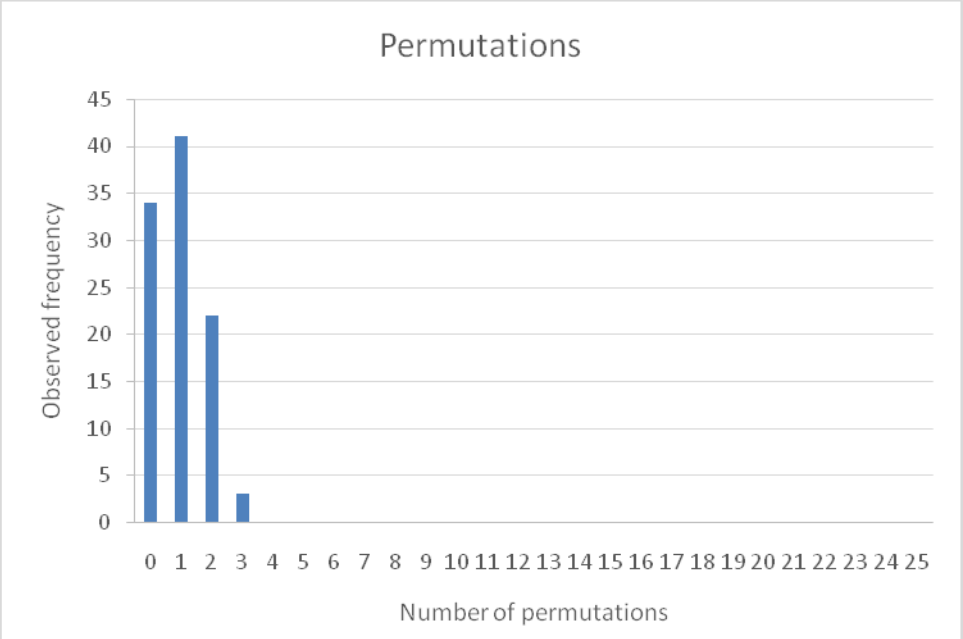
Erlang-4 Distributed Race Times

For Erlang-4 distributed race times we obtain the following results:

- Athlete 1 was found to be the best athlete in 97% of the cases.
- The ML function with the estimated values was very close to the one with the original values.



- The resulting ordering of athletes was relatively close to the original.



The mean permutations number is 0.94.

ANALYSIS	λ(1)	λ(2)	λ(3)	λ(4)	λ(5)	λ(6)	λ(7)	λ(8)	λ(9)	λ(10)
Real	1	0.8696	0.7692	0.6897	0.625	0.5714	0.5263	0.4878	0.4545	0.4255
Means	1	0.7518	0.561	0.439	0.3622	0.2933	0.2503	0.2127	0.1818	0.1562
Median	1	0.7613	0.5688	0.4434	0.3646	0.2909	0.2464	0.2147	0.1752	0.1545
St. Dev. (sample)	0	0.1225	0.0885	0.0729	0.0559	0.0456	0.0438	0.0334	0.0333	0.0271
Max	1	1.0425	0.74	0.6787	0.5375	0.4133	0.3812	0.2993	0.2977	0.2266

Min	1	0.4372	0.3465	0.2967	0.2372	0.1989	0.161	0.1175	0.112	0.0924
Range	0	0.6053	0.3935	0.382	0.3004	0.2144	0.2202	0.1818	0.1857	0.1342
Range/Mean	0	0.8052	0.7015	0.8701	0.8293	0.7309	0.8798	0.855	1.0214	0.8591

Table of results:

Conclusions

From the above results we can see that our approach gives very good results for the given original parameter values. In particular, the estimators we obtain constitute reliable estimators for the prediction of first finishing athlete in races. The rankings are in most case very close to the original ones. Most importantly (and not surprisingly), the basis of exponentially distributed race times for the likelihood function seems to provide results for the dominance vector that work better for Erlang-*k* times.

Directions for Future Research

Further experimentation is needed to reveal the limitations of the method and establish the preconditions for usage, possibly with confidence levels.

Another direction is the establishment of a likelihood function based on the order statistics of the Erlang-*k* distribution to obtain more accurate estimators for a broader range of values.

The authors of the present work are in the process of generalizing the above results for the Weibull distribution in general. It can be shown that the ranking problem and the order statistics of the Weibull distribution can be described by equations (1) and (2) for the same shape parameters. Thus, assuming the same shape parameters.

It should be noted here that Lessman, Sung and Johnson (2010) in their criticism for the conditional logit (CL) model, explain that

> Despite CL's advantage with respect to accounting for within-race competition, it suffers from some important limitations. In particular, the application of CL assumes that errors are distributed according to the double exponential distribution, which is unlikely in practice…

Furthermore, they pinpoint that the conditional logit (CL) model is unable to capture nonlinear interactions among independent variables. Finally, they warn that a maximum likelihood based estimation of the coefficients may be unstable if a large number of independent variables are used. The authors have developed and presented a machine learning method using a *random forest* procedure. The suggested procedure complements the conditional logit model and compensates for the instability of the standard maximum likelihood estimation.

A similar approach is recommended for the presented model. A machine learning approach can be developed to generate estimators and improve them as more results are being added.

Acknowledgment

This work was partially supported by the National Research Foundation of Korea Grant funded by the Korean Government (NRF-2014S1A2A2028551).

References

Ahsanullah M, Nevzorov VB, and Shakil M (2013) *An Introduction to Order Statistics*, Paris, France: Atlantis Press.

Ali MM (1977) Probability and Utility Estimates for Racetrack Bettors. *Journal of Political Economy* 85(4): 803-815.

Bolton RN, Chapman RG(2008) Searching for positive returns at the track: A multinomial logit model for handicapping horse races, in DB Hausch et al. (eds) *Efficiancy of Racetrack Betting Markets*, 151-171.

Chapman RG, Staelin R (1982) Exploiting Rank Ordered Choice Data within the Stochastic Utility Model.*Journal of Marketing Research* 19(3): 288-301.

Gakis K, Pardalos P, Choi CH, and Park JH (2016) A Probabilistic Model for Multi-Contestant Races.*Athens Journal of Sports* 3(2): 111-118.

Harville DA (1973) Assigning Probabilities to the Outcomes of Multi-Entry Competitions. *Journal of the American Statistical Association* 68(342): 312-316.

Johnson JEV, Jones O, and Tang L (2006) Exploring decision makers' use of price information in a speculative market. *Management Science* 52(6): 897–908.

Lessman S, Sung M-C, and Johnson JEV (2010) Alternative methods of predicting competitive events: An application in horserace betting markets. *International Journal of Forecasting* 26(3): 518-536.

Luce RD, Suppes P (1965) Preference, Utility and Subjective Probability, in RD Luce, RR Bush and EH Galanter(eds) *Handbook of Mathematical Psychology*, 3:249-410.

McFadden D (1974) Conditional logit analysis of qualitative choice behavior, in P Zrembka (ed) *Frontiers in Econometrics*, 105-142.

Stekler HO, Sendor D, and Verlander R (2010) Issues in sports forecasting. *International Journal of Forecasting* 26(3): 606-621.

VBA simulation program
The initialization procedure

```
Sub initialize()

Set ws = Sheets("Input_parameters")

athletes_num = Range("athletes_num")
athletes_per_game = Range("athletes_per_game")
games_num = Range("games_num")

ReDimathlete_rates(athletes_num)
Set cur_cell = Range("base_rate")
For i = 1 Toathletes_num
athlete_rates(i) = cur_cell.Offset(0, i).Value
Next i

End Sub
```

The main procedure

```
Sub hundred_simulations()
Dim counter As Integer
Dim my_ws As Worksheet

Set my_ws = Sheets("Simulation_results")

Application.ScreenUpdating = False
Application.Calculation = xlManual
Application.CalculateBeforeSave = True

For counter = 1 To 100
Call simulate
my_ws.Range("a6").Offset(counter, 0).Value = counter
   For i = 1 To 17
my_ws.Range("a6").Offset(counter, i).Value = my_ws.Range("A1").Offset(1, i).Value
   Next
Next counter

Application.ScreenUpdating = True
Application.Calculation = xlAutomatic
Application.CalculateBeforeSave = True

my_ws.Activate

End Sub
```

The simulation subprocedure

```
Sub simulate()

Call generate_games
Call find_MLEs

End Sub
```

The generation of the games procedure(for exponential distribution)

```
Sub generate_games()

'change the seed
Dim seed As Integer, first As Double
seed = Second((Now))
Call Randomize(seed)
first = Rnd()

Application.Calculate
Call initialize

'declare the array of games and position
Dim game_results() As Integer
ReDimgame_results(games_num, athletes_per_game)
'declare the athletes pool for each game
Dim athlete_pool() As Integer
ReDimathlete_pool(athletes_num)
'declare each athletes participating in each lane
Dim athlete_set() As Integer
ReDimathlete_set(athletes_per_game)

'initialize games: for each game create a set of participating athletes
For i = 1 Togames_num
   'initialize the pool
ReDimathlete_pool(athletes_num)
   For j = 1 Toathletes_num
athlete_pool(j) = j
   Next j

   For j = 1 Toathletes_per_game
      'calculate available athletes in the pool for selection
n = athletes_num - j + 1
      'select an athlete from the pool for the j-th lane
k = Int(Rnd() * n) + 1
      'put selected athlete in the set and remove from pool
athlete_set(j) = athlete_pool(k)
m = UBound(athlete_pool)
athlete_pool(k) = athlete_pool(m)
ReDim Preserve athlete_pool(m - 1)
game_results(i, j) = athlete_set(j)
   Next j
Next i

'go and print participating athletes in lanes
Call ClearSheet(Sheets("Games_initial"), games_num, athletes_per_game)
Set ws = Sheets("Games_initial")
Set top_cell = ws.Range("A1")

For i = 1 Togames_num
```

```
   For j = 1 Toathletes_per_game
top_cell.Offset(i, j).Value = game_results(i, j)
   Next j
Next i

'go and simulate the games
Call ClearSheet(Sheets("Games_simulated"), games_num, athletes_per_game)
Call ClearSheet(Sheets("Times_simulated"), games_num, athletes_per_game)

Dim temp_d As Double
Dim temp_i As Integer
Dim sorted As Boolean

Dim times() As Double
ReDimtimes(athletes_per_game)

Application.Calculate
For i = 1 Togames_num
   'find times
   For j = 1 Toathletes_per_game
times(j) = -1 / athlete_rates(game_results(i, j)) * Log(Rnd())
   Next j

   'rearrange by position

   Do
sorted = True
      For j = 1 Toathletes_per_game - 1
         If times(j) > times(j + 1) Then
temp_i = game_results(i, j)
game_results(i, j) = game_results(i, j + 1)
game_results(i, j + 1) = temp_i
temp_d = times(j)
times(j) = times(j + 1)
times(j + 1) = temp_d
sorted = False
         End If
      Next j
   Loop Until sorted
   'print time
   For j = 1 Toathletes_per_game
Sheets("Times_simulated").Range("A1").Offset(i, j).Value = times(j)
   Next j
Next i

'go and print the results
Set ws = Sheets("Games_simulated")
Set top_cell = ws.Range("A1")
For i = 1 Togames_num
   For j = 1 Toathletes_per_game
top_cell.Offset(i, j).Value = game_results(i, j)
```

```
   Next j
Next i

End Sub

The generation of the games procedure (for Erlang-k disribution)
Sub generate_games()

'change the seed
Dim seed As Integer, first As Double
seed = Second((Now))
Call Randomize(seed)
first = Rnd()

Application.Calculate
Call initialize

'declare the array of games and position
Dim game_results() As Integer
ReDimgame_results(games_num, athletes_per_game)
'declare the athletes pool for each game
Dim athlete_pool() As Integer
ReDimathlete_pool(athletes_num)
'declare each athletes participating in each lane
Dim athlete_set() As Integer
ReDimathlete_set(athletes_per_game)

'initialize games: for each game create a set of participating athletes
For i = 1 Togames_num
   'initialize the pool
ReDimathlete_pool(athletes_num)
   For j = 1 Toathletes_num
athlete_pool(j) = j
   Next j

   For j = 1 Toathletes_per_game
      'calculate available athletes in the pool for selection
n = athletes_num - j + 1
      'select an athlete from the pool for the j-th lane
k = Int(Rnd() * n) + 1
      'put selected athlete in the set and remove from pool
athlete_set(j) = athlete_pool(k)
m = UBound(athlete_pool)
athlete_pool(k) = athlete_pool(m)
ReDim Preserve athlete_pool(m - 1)
game_results(i, j) = athlete_set(j)
   Next j
Next i

'go and print participating athletes in lanes
Call ClearSheet(Sheets("Games_initial"), games_num, athletes_per_game)
```

```
Set ws = Sheets("Games_initial")
Set top_cell = ws.Range("A1")

For i = 1 Togames_num
   For j = 1 Toathletes_per_game
top_cell.Offset(i, j).Value = game_results(i, j)
   Next j
Next i

'go and simulate the games
Call ClearSheet(Sheets("Games_simulated"), games_num, athletes_per_game)
Call ClearSheet(Sheets("Times_simulated"), games_num, athletes_per_game)

Dim temp_d As Double
Dim temp_i As Integer
Dim sorted As Boolean

Dim times() As Double
ReDimtimes(athletes_per_game)

Application.Calculate

k = Range("Erlang_k").Value

For i = 1 Togames_num
   'generate times Erlang-k
   For j = 1 Toathletes_per_game
times(j) = 0
      For m = 1 To k
times(j) = times(j) - 1 / athlete_rates(game_results(i, j)) * Log(Rnd())
      Next m
      'times(j) = times(j)/k
   Next j

   'rearrange by position

   Do
sorted = True
      For j = 1 Toathletes_per_game - 1
         If times(j) > times(j + 1) Then
temp_i = game_results(i, j)
game_results(i, j) = game_results(i, j + 1)
game_results(i, j + 1) = temp_i
temp_d = times(j)
times(j) = times(j + 1)
times(j + 1) = temp_d
sorted = False
         End If
      Next j
   Loop Until sorted
   'print time
```

```
   For j = 1 Toathletes_per_game
Sheets("Times_simulated").Range("A1").Offset(i, j).Value = times(j)
   Next j
Next i

'go and print the results
Set ws = Sheets("Games_simulated")
Set top_cell = ws.Range("A1")
For i = 1 Togames_num
   For j = 1 Toathletes_per_game
top_cell.Offset(i, j).Value = game_results(i, j)
   Next j
Next i

End Sub
```

The estimation of MLEs procedure

```
Sub find_MLEs()
Application.Calculate

Worksheets("MLE_with_four_players").Activate

'reset the problem
SolverReset

'set desired precision
SolverOptions precision:=0.001

'define objective function
SolverOKsetCell:=Range("objfunction"), maxMinVal:=1,
byChange:=Range("variables")

'flow preservation sonstraints
'NO CONSTRAINTS

'non-negativity constraints
SolverAddcellRef:=Range("variables"), relation:=3, formulaText:=0

SolverSolveuserFinish:=True

End Sub
```