Available online at www.sciencedirect.com

SCIENCE DIRECT®

Computers & Operations Research 33 (2006) 1939–1950

computers & operations research

www.elsevier.com/locate/cor

Optimal batting orders in one-day cricket☆

Tim B. Swartz[a,*], Paramjit S. Gill[b], David Beaudoin[c], Basil M. deSilva[d]

[a]*Department of Statistics and Actuarial Science, Simon Fraser University, Burnaby, BC, Canada, V5A 1S6*
[b]*Department of Mathematics and Statistics, Okanagan University College, Kelowna, BC, Canada, V1V 1V7*
[c]*Départment de mathématiques et de statistique, Cité universitaire, Québec, QC, Canada, G1K 7P4*
[d]*Department of Mathematics and Statistics, RMIT University, GPO Box 2476V, Melbourne, Victoria, 3001, Australia*

Available online 25 November 2004

Abstract

This paper concerns the search for optimal or nearly optimal batting orders in one-day cricket. A search is conducted over the space of permutations of batting orders where simulated annealing is used to explore the space. A non-standard aspect of the optimization is that the objective function (which is the mean number of runs per innings) is unavailable and is approximated via simulation. The simulation component generates runs ball by ball during an innings taking into account the state of the match and estimated characteristics of individual batsmen. The methods developed in the paper are applied to the national team of India based on their performance in one-day international cricket matches.
© 2004 Elsevier Ltd. All rights reserved.

Keywords: Log-linear models; Markov chain methods; Monte Carlo simulation; Simulated annealing; WinBUGS software

1. Introduction

In cricket, there is a long-standing tradition of general strategy that places the better batsmen near the beginning of the batting order and the weaker batsmen near the end of the batting order ([1], p. 92–94). Within these general guidelines, subtle permutations of the batting order seem to be as much

☆ This work was initiated while Beaudoin was an M.Sc. candidate in the Department of Statistics and Actuarial Science at Simon Fraser University. Swartz and Gill were partially supported by grants from the Natural Sciences and Engineering Research Council of Canada. The authors thank two anonymous referees for valuable comments.

* Corresponding author. Fax: 604 291 4368.

E-mail addresses: tim@stat.sfu.ca (T.B. Swartz), pgill@ouc.bc.ca (P.S. Gill), packersbills@hotmail.com (D. Beaudoin), desilva@rmit.edu.au (B.M. deSilva).

0305-0548/$ - see front matter © 2004 Elsevier Ltd. All rights reserved.
doi:10.1016/j.cor.2004.09.031

art as science, and depend on factors such as cohesive partnerships and team psychology [2]. Since not all batsmen necessarily bat in a match, the general strategy is intuitively sensible as it provides greater opportunities for the best batsmen to bat.

What is not clear is the optimal batting order or even what is meant by the 'best batsmen'. There are various criteria for ranking batsmen (for example, batting average and strike rate), and these criteria can provide conflicting rankings [3]. In this paper, we explore optimal batting orders in the context of one-day cricket. In particular, we use simulation techniques to identify promising batting orders for the national team of India.

Although our literature searches have found no previous work on this topic, there exist papers on the related topic of optimal scoring rates. See, for example, the papers by Clarke [4] and Preston and Thomas [5] which utilize dynamic programming methods. A common theme in these papers is that teams would be better served by scoring at higher rates early in the innings. Our focus is different, and does not concern the way that cricketers should bat. Rather, given the way that they actually bat, we are interested in finding optimal or nearly optimal batting orders. It is our hope that the techniques developed in this paper may be of practical interest to teams that wish to explore alternative batting orders.

An initial thought may be that this is a straightforward problem. Simply look at a team's scoring results under different batting orders and select the batting order that produces the highest number of runs in an innings. Clearly, this is infeasible since there are many possible batting orders and relatively few matches. Note also that there is randomness in the number of runs scored in an innings which compounds the problem of finding the optimal batting order. Our approach to finding optimal batting orders is based on simulation where we are able to generate the number of runs scored in an innings under any proposed batting order. The generation of runs in one-day cricket is facilitated by the fact that there are a finite number of batting outcomes on any ball bowled and there are a finite number of balls. Surprisingly, simulation techniques in one-day cricket have not been fully explored. In the pre-computer days, Elderton [6] and Wood [7] fit the geometric distribution to individual runs scored based on results from test cricket. Kimber and Hansford [8] argue against the geometric distribution and obtain probabilities for selected ranges of individual scores in test cricket using product-limit estimators. More recently, Dyte [9] simulates batting outcomes between a specified test batsman and bowler using career batting and bowling averages as the key inputs without regard to the state of the match. We simulate runs against an average opponent by first estimating batting characteristics of team members based on their detailed performance in previous matches. The estimation procedure uses WinBUGS software [10] implemented for a Bayesian log-linear model. Given the estimated characteristics, it is straightforward to generate batting outcomes at any state of the match.

Once the simulation technique has been developed, we turn to the problem of finding an optimal or nearly optimal batting order at the start of a team's first innings. A non-standard aspect of the optimization is that the objective function (which is the mean number of runs per innings) is unavailable and is approximated via simulation. The optimization is computationally intensive for two reasons: (1) there are many batting orders to consider and (2) the estimation of the objective function requires many simulations. Simulated annealing [11] is used to explore the space of permutations of batting orders. Simulated annealing is a general approach used in combinatorial optimization, and fine tuning of the algorithm is required to address our particular problem.

In Section 2, we explain the essential features of one-day cricket with a particular emphasis on how runs are scored. We also motivate the paper by providing a hypothetical example that contradicts prevailing wisdom concerning optimal batting orders. In Section 3, we describe a simple simulation technique that generates first innings runs on a ball by ball basis. An important feature of the approach is that the state of the match is taken into consideration so that batting outcomes are not independent. The simulation is based upon the estimated characteristics of batsmen which are obtained from a Bayesian log-linear model. Inference from the log-linear model is readily facilitated using WinBUGS software and is demonstrated on data obtained from one-day international (ODI) matches involving India. In Section 4, simulated annealing is used to obtain optimal or nearly optimal batting orders. The optimization procedure is applied to the national team of India and provides some surprising results. We conclude with a short discussion in Section 5.

2. The basics of one-day cricket

At a very simple level, one-day cricket (often called limited overs cricket) differs from the more traditional variety of cricket in that matches last for only a single day. One-day cricket was introduced in the 1960s to reduce the frequency of drawn matches and to increase excitement through more aggressive batting.

One-day cricket involves one team batting (their innings) followed by the opposing team batting (their innings). Whoever scores the most runs wins. A team's innings is terminated whenever the team has completed 50 *overs* or has lost 10 *wickets*. An exception to this is that the second team stops batting whenever their score exceeds the score of the first team. The two criteria of 50 overs and 10 wickets are the complicating factors. Ignoring some details, an over consists of 6 balls bowled by a bowler. Therefore, there is a maximum of $50 \times 6 = 300$ balls and the batsman can score runs on each ball bowled. On each ball, the batsman can be *dismissed* (this is called a wicket), or he can score 0, 1, 2, 3, 4, 5 or 6 runs. When a batsman is dismissed, a new batsman enters the match. There are 11 players on a team and two batsmen alternate batting at any one time. The pair of batsmen is known as a *partnership* and the batting alternates from one batsman to his partner after a ball is bowled if either 1, 3 or 5 runs is scored, or if the ball is the last (i.e. sixth) ball of the over (but not both).

Essentially, there are two extreme batting strategies from which intermediate strategies can be obtained: (1) aggressive where a batsman tries to score runs at a high rate while at a greater risk of losing a wicket and (2) conservative where the batsman tries to preserve wickets while scoring runs at a lower rate. The state of the match influences the style of batting. For example, if there are ample wickets in hand but few overs left in an innings, batsmen tend to be aggressive.

With this basic understanding of one-day cricket, the question arises as to the determination of optimal batting orders. Consider then the unrealistic scenario where there is a team of 11 cricketers, 10 of whom are fantastic batsmen who score one run with probability 1.0 on every ball bowled. The 11th batsman is poor (by almost anyone's standards) as he is dismissed with probability 0.5 and scores 6 runs with probability 0.5 on every ball bowled to him. Now, conventional wisdom places the 'poor' batsman at the end of the batting order, and the team always scores 300 runs in the first innings since the first partnership remains intact, scoring 1 run on each ball bowled. If, however, we place the poor batsman at the beginning of the batting order, then the expected number of runs scored by the team in the first innings is

$$\frac{1}{2}(0+299)+\frac{1}{4}(6+298)+\frac{1}{8}(12+297)+\cdots+\frac{1}{2^{300}}(1794+0)+\frac{1}{2^{300}}(1800)\approx 304.$$

Therefore the accepted practice is not optimal in this hypothetical example.

For some, the general strategy of placing the better batsmen near the beginning of the batting order may be obvious (or even sacred), and the previous example may be dismissed as pathological. However, with a team of 11 cricketers, there are nearly 40 million (i.e. 11!) potential batting orders, and even skeptics ought to concede that there may exist undiscovered yet promising batting orders. This is the essential motivation of our work.

3. Simulation of runs

In this section, we consider the simulation of runs in the first innings for a specified batting order. For simplicity, we investigate the first innings runs since the batting strategy of the team batting second depends on the number of runs scored by the team batting first.

In one-day cricket, there are a finite number of outcomes arising from each ball bowled. Ignoring certain rare events (such as scoring 5 runs), and temporarily ignoring wide-balls, no-balls, byes and leg-byes, let X_i denote the outcome of the ith ball bowled, $i = 1, \ldots, 300$ where

$$X_i = \begin{cases} 0 & \text{if the batsman scores 0 runs,} \\ 1 & \text{if the batsman scores 1 run,} \\ 2 & \text{if the batsman scores 2 runs,} \\ 3 & \text{if the batsman scores 3 runs,} \\ 4 & \text{if the batsman scores 4 runs,} \\ 5 & \text{if the batsman scores 6 runs,} \\ 6 & \text{if the batsman is dismissed} \end{cases}$$

and $X_{m+1} = \cdots = X_{300} = 0$ if the innings terminate on the mth ball where $m < 300$.

Using square brackets to generically denote probability mass functions, the joint distribution of $X_1, \ldots, X_{300}$ can be written as

$$[X_1, \ldots, X_{300}] = [X_{300} \mid X_1, \ldots, X_{299}][X_{299} \mid X_1, \ldots, X_{298}] \cdots [X_2 \mid X_1][X_1 \mid X_0], \quad (1)$$

where we express $[X_1] = [X_1 \mid X_0]$ for convenience. Letting q_1 denote the probability of a wide-ball or a no-ball and letting q_2 denote the probability of a bye or a leg-bye, we then use (1) to obtain the following simulation algorithm for generating the number of runs R scored in the first innings:

```
•    $R = 0$,
•    for $i = 1, \ldots, 300$.
     1. Generate $u_1 \sim$ uniform(0,1).
     2. If $u_1 < q_1$,
          then
               $R \leftarrow R + 1$
               return to step 1
          else
               if $u_1 < q_1 + q_2$,
               then
                    $R \leftarrow R + 1$
```

```
    else
        generate $X_i \sim [X_i \mid X_1, \ldots, X_{i-1}]$
        $R \leftarrow R + X_i I(X_i \leqslant 4) + 6I(X_i = 5)$.
```

Note that in the case of wide-balls and no-balls, a run is assigned to the batting team but the ball is not counted. We have considered the cases of extras (i.e. wide-balls, no-balls, byes and leg-byes) separately since data on these outcomes are not typically collected for individual batsmen but are recorded as aggregate team values. We also recognize that the treatment of extras could be improved slightly by allowing the possibility of more than one run. The proposed algorithm is extremely simple and requires only that we be able to generate from the discrete distributions $[X_i \mid X_1, \ldots, X_{i-1}]$. In the following subsection, we estimate the distributions $[X_i \mid X_1, \ldots, X_{i-1}]$ using historical batting records.

3.1. Estimation of batting characteristics

The conditional distributions $[X_i \mid X_1, \ldots, X_{i-1}]$ depend on many factors including:

- the batsmen,
- the number of wickets lost,
- the number of balls bowled,
- the bowler,
- the opposing team,
- the coach's advice,
- the condition of the pitch, etc.

We consider the first three factors and define p_{jwbk} as the probability corresponding to outcome $k = 0, \ldots, 6$ for the *j*th batsman when $w = 0, \ldots, 9$ wickets have been lost and $b = 0, \ldots, 299$ balls have been bowled. Since extras have been excluded as possible outcomes of X_i, we have $\sum_k p_{jwbk} = 1$ for all j, w, b. With estimates $\hat{p}_{jwbk}$, the simulation algorithm generates outcomes according to

$$\text{Prob}(X_i = k \mid X_1, \ldots, X_{i-1}) = \hat{p}_{jwbk}$$

for $i = 1, \ldots, 300$ and $k = 0, \ldots, 6$ where extras are excluded and less than 10 wickets have been lost.

To obtain an estimate of q_1, we look at first innings results from 239 ODI matches dating back from May 2003. We include all matches between teams belonging to the International Cricket Council (ICC). Over these 239 matches, we calculate $\hat{q}_1$ as the total number of runs awarded as wide-balls or no-balls in the first innings divided by the total number of balls bowled over these first innings. We obtain $\hat{q}_1 = 0.0356$. The simulation algorithm also requires an estimate $\hat{q}_2$ of the probability of byes or leg-byes. Using the same set of matches, we obtain $\hat{q}_2 = 0.0208$.

To obtain the estimates $\hat{p}_{jwbk}$, we introduce a Bayesian log-linear model. We first summarize historical match data so that T_{jwbk} is the number of times that outcome $k = 0, \ldots, 6$ occurs over the N_{jwb} trials involving the *j*th batsman where *w* wickets have been lost and *b* balls have been bowled. The model also requires the resource function $R(w, b)$ developed by Duckworth and Lewis [12] in the context of resetting targets in interrupted one-day cricket matches. The function $R(w, b)$ expresses the percentage of resources used in a match when *w* wickets have been lost and *b* balls have been bowled. For example,

$R(0,0)=0$ and $R(10,b)=100.0$. The data are distributed according to

$$[T_{jwb0},\ldots,T_{jwb6}] \sim \text{multinomial}(N_{jwb}, p_{jwb0},\ldots,p_{jwb6}), \tag{2}$$

where the parameters p_{jwbk} are forced to be probabilities via $p_{jwbk}=q_{jwbk}/\sum_k q_{jwbk}$ and

$$\log(q_{jwbk}) = \mu_{jk} + \alpha_k\left(\frac{w}{9}\right) + \beta_k\left(\frac{b}{299}\right) + \theta_k\left(\frac{R(w,b)}{100}\right). \tag{3}$$

To overcome nonidentifiability in the model, we assign $\alpha_6=\beta_6=\theta_6=0$ and $\mu_{j6}=0$ for each batsman j.

Note that the covariates in the log-linear relationship (3) have been standardized to the interval [0,1]. This allows us to more readily assess the contribution of the individual parameters. Observe that the parameter μ_{jk} may be interpreted as a baseline value at the beginning of the match when $w=b=R(0,0)=0$. The rationale behind (3) is based on the recognition that batsmen have different abilities, yet as the match progresses we assume that their characteristics change in a common manner. The resource function may be thought of as an interaction term involving the wickets and the balls.

To complete the model specification, we assign the following independent prior distributions:

$$\mu_{jk} \sim \text{normal}(0,\sigma_k^2), \quad \text{where } \sigma_k^{-2} \sim \text{gamma}(0.001, 0.001),$$

$$\alpha_k \sim \text{normal}(0,\sigma_\alpha^2), \quad \text{where } \sigma_\alpha^{-2} \sim \text{gamma}(0.001, 0.001),$$

$$\beta_k \sim \text{normal}(0,\sigma_\beta^2), \quad \text{where } \sigma_\beta^{-2} \sim \text{gamma}(0.001, 0.001),$$

$$\theta_k \sim \text{normal}(0,\sigma_\theta^2), \quad \text{where } \sigma_\theta^{-2} \sim \text{gamma}(0.001, 0.001),$$

for each batsman j and $k=0,\ldots,5$. The hyper-parameters of the gamma distributions have been chosen so that the prior distributions of the variance parameters are diffuse. One of the advantages of the model is that it allows us to infer characteristics of batsmen at stages of a match where the batsmen have little or no actual batting experience. For example, bowlers typically bat near the end of the batting order, yet the model allows us to estimate how they would bat near the beginning of the batting order.

To investigate the Bayesian log-linear model, we collected data on ODI matches involving the national team of India. We chose India for two reasons: (1) there is considerable current interest in India as they are a formidable side having finished second to Australia in the 2003 World Cup and they are supported by a populous country with a strong tradition in cricket, and (2) the Indian team has kept a fairly stable lineup over the past few years which has resulted in a good set of data for individual team members. The 13 members of the Indian team for whom we collected data are recorded in Table 1. Bearing the constraints in mind, there are $13(6)+6+6+6=96$ primary parameters of interest corresponding to (3). To reduce the computational burden, the data were grouped forming larger N_{jwb} values in (2). Specifically, the data were collapsed according to whether the number of balls b resided in the first 5 overs, the next 5 overs, etc. It is our opinion that batting characteristics do not change substantively within these narrow categories. The data summarization resulted in 404 multinomial distributions. The model was fit using WinBUGS software [10] where posterior estimates of the parameters are obtained by averaging output from a Markov chain. The estimates are recorded in Table 1. We observe that the better batsmen (e.g. Tendulkar, Ganguly and Dravid) have higher baseline values (i.e. μ_{jk}) for producing 1 run, 2 runs and 4 runs. We also observe that the signs of the common parameters (i.e. the α's, β's and θ's) generally agree with our intuition. For example, as wickets are lost, a batsman typically becomes more conservative, and hence the probabilities of 2's, 3's, 4's and 6's decrease (i.e. $\hat{\alpha}_2$, $\hat{\alpha}_3$, $\hat{\alpha}_4$ and $\hat{\alpha}_5$ are negative).

Table 1
Estimated characteristics of India's batting behaviour based on 71 first innings ODI matches dating from December 1998 through the completion of the World Cup in March 2003. The indices 1 through 11 correspond to India's batting order in the 2003 World Cup final

			$k=6$	$k=0$	$k=1$	$k=2$	$k=3$	$k=4$	$k=5$
Common parameters		$\hat{\alpha}_k$	0.00	1.49	1.41	−0.34	−0.28	−1.19	−2.04
		$\hat{\beta}_k$	0.00	−0.35	4.22	0.25	−1.33	−1.04	1.91
		$\hat{\theta}_k$	0.00	−2.44	−4.54	0.29	−1.02	1.13	0.50
1	S.R. Tendulkar	$\hat{\mu}_{1k}$	0.00	4.15	2.56	1.40	−0.26	2.12	−1.41
2	V. Sehwag	$\hat{\mu}_{2k}$	0.00	3.30	1.29	0.50	−0.63	1.67	−1.61
3	S.C. Ganguly	$\hat{\mu}_{3k}$	0.00	4.06	2.37	0.86	−0.26	1.70	−0.79
4	M. Kaif	$\hat{\mu}_{4k}$	0.00	3.59	1.86	0.33	−0.23	1.24	−2.43
5	R. Dravid	$\hat{\mu}_{5k}$	0.00	4.30	2.56	1.10	−0.04	1.77	−2.29
6	Y. Singh	$\hat{\mu}_{6k}$	0.00	3.84	1.67	0.50	−0.17	1.42	−1.79
7	D. Mongia	$\hat{\mu}_{7k}$	0.00	3.68	1.91	0.49	−0.19	1.54	−2.05
8	H. Singh	$\hat{\mu}_{8k}$	0.00	3.03	0.81	−0.05	−0.29	1.28	−2.08
9	Z. Khan	$\hat{\mu}_{9k}$	0.00	3.36	0.44	0.13	−0.25	1.06	−1.13
10	J. Srinath	$\hat{\mu}_{10k}$	0.00	3.55	1.01	−0.77	0.04	0.89	−2.33
11	A. Nehra	$\hat{\mu}_{11k}$	0.00	3.60	0.93	−0.68	−0.07	−0.89	−2.22
12	A. Kumble	$\hat{\mu}_{12k}$	0.00	4.01	1.38	−0.24	0.21	0.45	−3.54
13	A. Agarkar	$\hat{\mu}_{13k}$	0.00	3.31	1.21	0.09	−0.32	1.06	−2.13

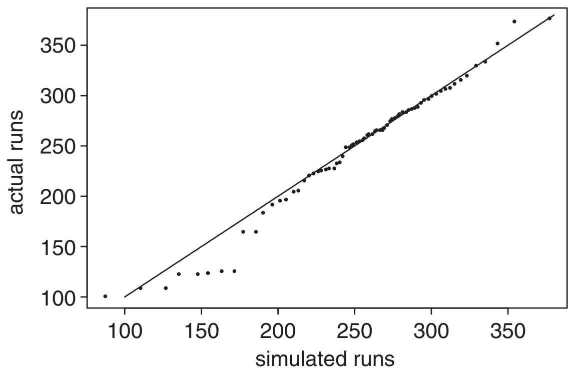

Fig. 1. Q–Q plot corresponding to the actual runs scored in 71 first innings ODI matches for India dating from December 1998 through the completion of the World Cup in March 2003 versus simulated runs based on India's batting lineup in the 2003 World Cup final.

To investigate the simulation procedure using the estimates in Table 1, we collected additional team data on India corresponding to ODI matches from December 1998 through the completion of the World Cup in March 2003. We considered only the 71 matches where India batted in the first innings and we obtained the number of wickets lost and the runs scored in each match. We also simulated 71,000 first innings runs using India's 2003 World Cup final batting order as given in Table 1. In Fig. 1, we present a

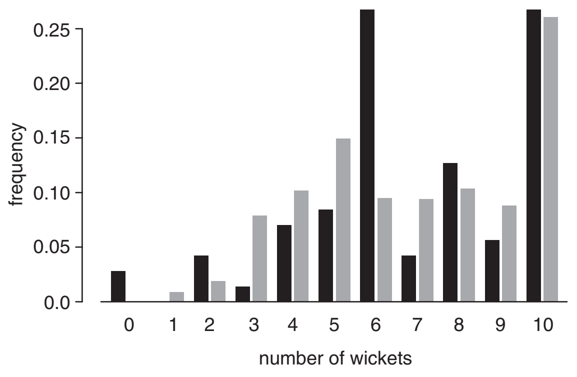

Fig. 2. Frequency plot corresponding to the actual wickets lost in 71 first innings ODI matches for India dating from December 1998 through the completion of the World Cup in March 2003 (dark shading) and simulated wickets lost based on India's batting lineup in the 2003 World Cup final (light shading).

Q–Q plot corresponding to the actual runs scored versus the simulated runs. We do not expect an exact fit as there is sampling variation in the number of runs scored and India's batting order was not the same over all 71 matches. Still, the fit was fairly good except for the unexpected sequence of 122, 122, 123, 125 and 125 actual runs which is clearly a sampling abnormality. In Fig. 2, we present a bar plot of the actual wickets lost and the simulated wickets lost. We observe that the general shapes agree except for the spike of 6 actual wickets lost; this also appears to be a sampling abnormality. Although the multiple modes suggested by Fig. 2 is somewhat unusual, it does correspond to our intuition.

4. Optimal batting orders

Consider the problem of finding the optimal batting order in one-day cricket for the national team of India. Based on a study of recent matches, it appears that two of the four bowlers H Singh, A Nehra, A Kumble and A Agarkar are typically chosen for a match, and the balance of the side consists of the remaining 9 cricketers listed in Table 1. With this constraint, there are $\binom{4}{2}\,11! \approx 240$ million feasible batting orders.

Mathematically, using the coding in Table 1, the problem consists of maximizing the expected number of first innings runs

$$f(\mathbf{i}) = E_{\mathbf{i}}(R), \tag{4}$$

over the set of $\binom{4}{2}\,11!$ permutations $\mathbf{i} = (i_1, \ldots, i_{13})$ where i_j denotes that batsman i_j bats in the jth position in the batting order, and cricketers i_{12} and i_{13} are left off the team.

One of the non-standard aspects of the problem is that the function f in (4) is not readily available and is estimated via simulation. An implication is that repeated simulations of the number of first innings runs under the same batting order $\mathbf{i}$ typically yield different estimates of $f(\mathbf{i})$. Consequently, we simulate the number of first innings runs n times and average the values to produce an estimate of f with less variability. The immediate question arises as to the choice of n. Since we are interested in optimizing the *average* number of runs scored, it seems sensible that n should be chosen, that with confidence, the estimate is within one run of the true average. We choose $n = 10,000$ as this produces a standard error of roughly 0.5 for most permutations. Unfortunately, such a choice rules out the possibility of an exhaustive search for the optimal batting order. On a Sun workstation, a simulation of $n = 10,000$ requires roughly 37 s of computation, and multiplied over the $\binom{4}{2}$ 11! batting orders, this requires more than 280 years of computation!

To address the computational hurdles, we implement the simulated annealing algorithm [11] to obtain optimal or nearly optimal batting orders. Simulated annealing is a probabilistic search algorithm that proceeds (in our context) by generating a candidate permutation $\mathbf{j}$ in a neighbourhood of the current permutation $\mathbf{i}$. If $\hat{f}(\mathbf{j}) > \hat{f}(\mathbf{i})$, then the permutation $\mathbf{j}$ is accepted as the current state. If $\hat{f}(\mathbf{j}) \leqslant \hat{f}(\mathbf{i})$, it is also possible to proceed to state $\mathbf{j}$, and this occurs if a randomly generated uniform(0,1) variate $u < \exp\{(\hat{f}(\mathbf{j}) - \hat{f}(\mathbf{i}))/t\}$ where $t > 0$ is a parameter often referred to as the temperature. The algorithm iterates according to a sequence of non-increasing temperatures $t_k \to 0$. Beginning with an initial state $\mathbf{i}_0$, the states $\mathbf{i}_0, \mathbf{i}_1, \ldots$ form a Markov chain. The algorithm terminates after a fixed number of iterations or when state changes occur infrequently. Under a 'suitable' neighbourhood structure, asymptotic results suggest that the final state will be nearly optimal.

Success of the simulated annealing algorithm depends greatly on fine tuning of the algorithm. In particular, the user must specify the cooling schedule (i.e. t_k) and also the neighbourhood structure for generating successive states from a given state. Aarts and Korst [13] provide discussion on the issues of fine tuning the algorithm.

Our implementation of simulated annealing begins with the recognition that our problem shares some similarities with the well-studied travelling salesman problem. For example, like our problem, the state space in the travelling salesman problem consists of permutations, permutations of cities that are visited by the salesman. Also, in the same way that an interchange in the order of two adjacent cities in a permutation should not greatly affect the total travelling distance for the salesman, an interchange in the order of two adjacent batsmen in a permutation should not greatly affect the expected number of first innings runs. Accordingly, our implementation of simulated annealing uses an exponential cooling schedule defined by a sequence of temperature plateaux; this approach has been successively used in the travelling salesman problem [13]. Specifically, we iterate the Markov chain 100 times at temperatures $t = 0.5(0.9)^0, 0.5(0.9)^1, \ldots, 0.5(0.9)^{14}$, escaping any temperature prematurely if there are more than 10 state changes at the temperature. Tied to the cooling schedule, we also increment the number of simulations n at each temperature level according to 200, 900, 1600, ..., 10, 000. To address the infeasible batting orders, we introduce a penalty and reduce $\hat{f}$ by 5 runs whenever more than two of H Singh, A Nehra, A Kumble and A Agarkar are included in the line-up. It is important that the penalty not be too extreme; otherwise the simulated annealing algorithm may experience difficulties in moving across neighbourhoods. The specification of the algorithm is complete by specifying the generating mechanism which determines the neighbourhood structure. For this, consider the current state $\mathbf{i} = (i_1, \ldots, i_{13})$ and generate a discrete uniform variable k on $(1, \ldots, 13 - m + 1)$ where the parameter m is user-specified. We then randomly permute $(i_k, i_{k+1}, \ldots, i_{k+m-1})$ yielding $(j_k, j_{k+1}, \ldots, j_{k+m-1})$. The candidate state in the algorithm is then given by $(i_1, \ldots, i_{k-1}, j_k, \ldots, j_{k+m-1}, i_{k+m}, \ldots, i_{13})$. We found $m = 4$ to be a good choice; sufficiently small that local minima can be found, and sufficiently large that it is not too difficult to move across neighbourhoods. A typical run of simulated annealing using this implementation requires approximately 7.5 h of computation on a Sun workstation.

Table 2
Two potentially productive batting orders for India identified via simulated annealing. An estimate of the expected number of first innings runs $\hat{f}$ for each batting order is provided

Order	(1)	(2)
1	Dravid	Tendulkar
2	Tendulkar	Ganguly
3	Ganguly	Dravid
4	Sehwag	Sehwag
5	Mongia	Mongia
6	Singh (Y)	Singh (Y)
7	Khan	Kaif
8	Kaif	Singh (H)
9	Singh (H)	Khan
10	Agarkar	Agarkar
11	Srinath	Srinath
$\hat{f}$	256.4	256.0

In Table 2, we present two promising batting orders that were identified using our implementation of simulated annealing. To put the results into perspective, we note that the batting order used by India in the 2003 World Cup final (see Table 1) produces an average of 250.1 first innings runs. Therefore the proposed batting orders offer the potential of improved ODI performance for India by approximately 6 runs. To appreciate the impact of a poor batting order, we reverse the batting order used by India in the 2003 World Cup final, and obtain an average of 201.2 first innings runs. Several observations are worthy of note:

- In 10 out of the 11 2003 World Cup matches in which India participated, the opening batsmen, without regard to order, were (Tendulkar & Sehwag). Our approach identifies (Dravid & Tendulkar) and (Tendulkar & Ganguly) as the most promising opening batsmen. Generally speaking, India places Dravid too far down in the batting order. As might be expected, interchanging the number 1 and number 2 batsmen has a negligible effect on the expected number of first innings runs.
- In India’s 11 World Cup 2003 matches, Kaif batted in the fourth position 6 times. This is strikingly different from the two promising batting orders from Table 2 where Kaif bats in the eighth and seventh positions.
- Our methods suggest that Sehwag, Mongia and Y Singh ought to bat in the fourth, fifth and sixth positions respectively; Mongia’s recommended position is typically a little earlier in the batting order than is currently the practice.
- As might be expected, the simulations suggest that permutations in the latter part of nearly optimal batting orders (positions 7–11) have a lesser effect on the expected number of first innings runs. There is little impact when the batsmen in these positions are interchanged. These batsmen tend to be bowlers and their inclusion is far more related to bowling than to batting.

5. Discussion

In this paper, we have presented a methodology for identifying promising batting orders in one-day cricket. In particular, we have suggested some batting orders that have never been tried by the Indian team and contradict prevailing wisdom.

As mentioned previously, we chose to study first innings data so that the results of all batsmen could be compared on the same footing. However, having identified promising first innings batting orders, it seems reasonable that these batting orders would also be promising batting orders for the second innings.

As a by-product of our investigation, we have developed a simulation procedure for generating first innings runs against an average opponent. Since one-day cricket data are sparse (i.e. an ICC side may schedule fewer than 20 matches per year), realistic simulation techniques may have value in investigating a wide range of questions. Accordingly, it would be useful to have a general simulation procedure that also took into account the strength of the opposition and second innings batting. This is a topic of future research.

We also note that the simulation procedure was based on estimates from a Bayesian log-linear model. Although our natural inclination is Bayesian, there is no reason why maximum likelihood estimates could not have been used as inputs to the simulation procedure. For the national team of India, we had a substantial data set and chose diffuse prior distributions. With less substantial data sets, an advantage of the Bayesian approach is that subjective prior information can be incorporated to overcome deficiencies in the data.

Finally, our methods were developed with the intention of finding optimal or nearly optimal batting orders at the start of a team's innings. In practice, a team has the prerogative to modify the initial batting order as the match is in progress, and this is frequently done in rain-delayed matches. Given the status of the match (i.e. the number of overs, the number of wickets and the batsmen dismissed), computations could be run in real time to obtain an optimal or nearly optimal batting order for the remaining batsmen. The computations would be less demanding since the number of permutations decrease as batters are dismissed. The simulations would also be less demanding for less than full matches. Clearly, our software could be a valuable asset to teams that are serious about performing to the best of their ability.

References

[1] Hankinson JT. Cricket for schools. London: George Allen and Unwin Limited; 1946.

[2] India Express Bureau. Ganguly plans new strategy, reshuffle of batting order. Website article at www.indiaexpress.com/news/sports/cricket/20020215-0.html, February 15.

[3] Beaudoin D, Swartz TB. The best batsmen and bowlers in one-day cricket. South African Statistical Journal 2003;37:203–22.

[4] Clarke SR. Dynamic programming in one-day cricket—optimal scoring rates. Journal of the Operational Research Society 1988;39(4):331–7.

[5] Preston I, Thomas J. Batting strategy in limited overs cricket. The Statistician 2000;49(1):95–106.

[6] Elderton WE. Cricket scores and some skew correlation distributions. Journal of the Royal Statistical Society, Series A 1945;108:1–11.

[7] Wood GH. Cricket scores and geometrical progression. Journal of the Royal Statistical Society, Series A 1945;108:12–22.

[8] Kimber AC, Hansford AR. A statistical analysis of batting in cricket. Journal of the Royal Statistical Society, Series A 1993;156:443–55.

[9] Dyte D. Constructing a plausible test cricket simulation using available real world data. In: de Mestre N, Kumar K, editors. Mathematics and Computers in Sport. Queensland, Australia: Bond University; 1998. p. 153–9.

[10] Spiegelhalter D, Thomas A, Best N. WinBUGS Version 1.3 User Manual. Cambridge: Medical Research Council Biostatistics Unit; 2000.
[11] Kirkpatrick S, Gelatt Jr CD, Vecchi MP. Optimization by simulated annealing. Science 1983;220:671–80.
[12] Duckworth FC, Lewis AJ. A fair method for resetting targets in one-day cricket matches. Journal of the Operational Research Society 1998;49:220–7.
[13] Aarts E, Korst J. Simulated annealing and Boltzmann machines. New York: Wiley; 1989.