Asia-Pacific Journal of Operational Research
Vol. 25, No. 4 (2008) 495–511

OPTIMAL STRATEGIES IN ONE-DAY CRICKET

PAM NORTON and RAVI PHATARFOD*

School of Mathematical Sciences
Monash University, Victoria, Australia, 3800
**ravi.phatarfod@sci.monash.edu.au*

Using a dynamic programming formulation, an analysis is presented of both the first and second innings of a one-day cricket match assuming variation in type of ball bowled and subsequent selection of a strategy by the batsman. We assume that the team batting first uses the strategy to maximize the expected score, and the team batting second uses the strategy to maximize the probability of outscoring the first team and thus of maximizing the probability of a win. The dynamic programming formulation allows a calculation, at any stage of the innings, of the optimal scoring strategy depending on the type of ball bowled, along with an estimate of the maximum of the expected number of runs scored in the remainder of the first innings, and the maximum probability of a win in the second innings. Modifications are then introduced to examine the effect of tailender batsmen and a "fifth bowler". Finally a simulation is done to estimate the variance in total score by following the optimal strategy used in the first innings.

Keywords: Dynamic programming; cricket; first innings; second innings; optimal strategies.

1. Introduction

The application of OR techniques in general, and dynamic programming in particular, to problems in sport is growing; see Norman (1995). The reason for this is fairly obvious; most sports involve elements intrinsic to a dynamic programming model — stage, state, decision, reward, uncertainty, etc. For example, in the game of darts, a player decides where to aim on the dart board to optimize his probability of reaching a certain target before his opponent, keeping in mind his present score, and the relative skills of himself and his opponent; see Kohler (1982). In horse racing, a jockey has to decide, at every stage of the race, whether to go forward by going around the pack or to stay in the present position. The latter option means the horse conserves his energy by not covering extra ground; however there is a risk that the horse could be blocked for a run later in the race. The jockey takes that decision which, in his opinion, maximizes the probability of the horse winning the race. Over the past 20

*Corresponding author.

years dynamic programming has been applied to the game of cricket; see Clarke (1988), Preston and Thomas (2000), Clarke and Norman (2004), to name a few. Although all forms of the game of cricket involve the above-mentioned elements of dynamic programming, it is in the one-day (limited overs) form of the game where the need for using optimization is most keenly felt. In the longer version (lasting for five days for a Test match and four days for a first-class match) it is possible to overcome the effects of wrong decisions such as, say, of choosing or not choosing to play a night watchman. In any case such occasions are few and far between. But it is for the one-day game that dynamic programming seems to be a tailor-made tool for analysis. We have here a fixed (usually 300) number of stages (balls to be bowled), a number of states (number of wickets in hand), the batsman making a decision how to respond to a ball, with consequent risk and reward and the uncertainty of the result of such a decision. Increasingly, in recent times, the team batting first, right from the word go, starts — indeed, would need to start — thinking in terms of maximizing the score, for it knows that quite often the difference between optimal and sub-optimal play is the difference between victory and defeat. Indeed, the history of the game is replete with instances where the superior team lost primarily because of not playing in an optimal fashion. Clarke (1988) used wickets in hand as the state to find optimal scoring rates for an over to maximize the expected score for the first innings and to maximize the probability of winning in the second innings. We essentially follow his approach, but go a stage further by making the batsman's response dependent on the quality of the ball bowled at him.

A point of some interest in such situations is whether, for two teams of equal ability and where the wicket and weather conditions remain the same, the team batting first or second has an advantage, i.e., has a probability of a win greater than a half. It would seem that the team batting second has an inherent advantage, in that, once the team batting first has completed its innings, it knows how many runs it needs to score to win, and this information can be explicitly incorporated (as we have done) into the dynamic programming formulation. The team batting first has no such information. Of course, it could guess (as it usually does) at how many runs are sufficient for a win, and consequently use a dynamic programming formulation for maximizing the probability of scoring these many runs. But such a formulation must remain, for obvious reasons, a sub-optimal one in terms of probability of a win. The criterion we have chosen for the first team, namely the maximization of the expected score seems to be the obvious choice of most teams (probably, all teams); however it may not be the optimal one in terms of maximizing the probability of a win.

In Sec. 2 we present the basic model. Sections 3 and 4 deal with the first innings assuming that all the batsmen and the bowlers are of the same ability. In Sec. 5 we consider the realistic situation when the team has batsmen and bowlers of differing ability. In Sec. 6 we give the results of a simulation of a first innings using our model. Finally, Sec. 7 considers the second innings.

2. The Basic Model

One-day cricket is a game between two teams of 11 players. There are two "innings" with each team batting once and bowling once. The team batting has two players at the center of the ground, called the pitch or the wicket, facing balls bowled at them by the bowling team. Balls are bowled in groups of 6 known as "overs". The bowling team's objective is to dismiss the batsmen ("getting out" or "losing a wicket"); there are various ways for doing this. The batting team's objective is to score "runs" by attempting to strike the ball and run between the ends of the wicket. Obviously, all forms of dismissal are made more likely by the batsmen attempting a higher scoring rate, effected by the batsmen making an attacking shot rather than by making a defensive response to the ball bowled. The batting teams therefore need to balance the advantage of a fast scoring rate against the risk of losing wickets. The innings ends after a set number of overs (usually 50) have been bowled or all but one of the batsmen have been dismissed, whichever happens sooner. The total runs accumulated by the team batting first plus one, is the target to be achieved by the team batting second.

Clarke (1988) presented a dynamic programming formulation of both the first and the second innings. This allows a calculation, at any stage of the innings, of the optimal scoring rate along with an estimate of the maximum expected score (in the case of the first innings) or the maximum chance of winning (in the second innings). His model does not take into account the difference between the quality of balls bowled. In this paper we make the realistic assumption that balls bowled vary in quality, taken here for simplicity as Good, Medium and Bad, and that the bowler does not have complete control over the quality of the ball bowled. In effect, balls bowled are independent trials, with the probabilities of the ball bowled being Good, Medium and Bad to be $P(G)$, $P(M)$ and $P(B)$ respectively. These probabilities would be different for different teams (depending on the bowlers chosen) as well as for different wicket and weather conditions. We assume that the team batting has some knowledge of these probabilities of the other team.

Let us briefly consider what we mean by balls being Good, Medium or Bad. A Good ball is one where the length and line are correct, i.e., one where, if the batsman decides to try to score runs, there is a high probability of him losing his wicket. So, usually the batsman either has a defensive stroke or lets the ball go to the wicket-keeper; in either case no run is scored. Of course, towards the end of the innings, a batsman would try to score and take the risk of losing his wicket. A Medium ball is one where the risk of losing one's wicket is a lot less, and the batsman usually tries to take a single run (to keep the score ticking over — particularly in the middle stages of the match), although he is not always successful in doing so. A Bad ball is one where the risk of losing one's wicket is minimum, and the batsman tries to score runs, typically a four or a six, although the shot may also result in his getting zero to three runs, or quite possibly losing his wicket. Of course, these terms are subjective and a Good ball for an ordinary batsman may well be a Medium one for a top-notch batsman.

Table 1. Parameters for Scenario 1.

Ball type	Action	Probability of dismissal	Prob(0)	Expected no. of runs
Good	1-Defend	0.02	0.98	0
	2-Single	0.1	0.4	0.5
	3-Attack	0.2		1
Medium	1-Defend	0.01	0.99	0
	2-Single	0.05	0.3	0.65
	3-Attack	0.15		1.5
Bad	1-Defend	0.001	0.999	0
	2-Single	0.01	0.2	0.79
	3-Attack	0.02		2

When the ball is bowled, the batsman has three choices of action — defend, play for a possible single, or attack with a view to scoring some runs, preferably four or six. For simplicity we shall call the actions "defend", "single" and "attack", and for brevity we will denote "defend" $\equiv 1$, "single" $\equiv 2$, and "attack" $\equiv 3$. Each type of ball and each type of action has a probability of dismissal and expected score. Table 1 gives what we consider are typical values of the above parameters — this will be called Scenario 1. For these values, the minimum expected score in an over is 0 and the maximum 12. As we shall see later on, the solution obtained by dynamic programming shows that, for the first innings, to attain the maximum expected score the predominant strategy employed by the batsmen should be "defend for Good, single for Medium, and attack for Bad", assuming $P(G) = P(M) = P(B) = 1/3$. For this strategy Table 1 gives the average probability of dismissal as 0.03, and the average runs per ball as 0.8833. With these values the average number of wickets fallen in an innings is 9, and the average number of runs scored is 265, values fairly close to what happens in practice.

Note the values in Table 1 are mainly for illustrative purposes, and there is no suggestion that they are the correct values. In a sense, there are no "correct" values. All the values would vary widely over teams and batting conditions. We suggest that the values given are average for average weather and wicket conditions and for the major eight cricket playing international teams.

In Secs. 3, 4, 6 and 7 we shall assume for simplicity that $P(G) = P(M) = P(B) = 1/3$ for the whole innings, and that the batsmen and the bowlers are of equal quality. For section 5, when we consider a realistic situation by the introduction of the "fifth bowler" and tailender batsmen we shall make suitable adjustments to the probabilities.

Let C denote the set of actions the batsman has, i.e., $C = \{\text{defend}, \text{single}, \text{attack}\} \equiv \{1, 2, 3\}$. Let $P_k^G(D)$ be the probability of dismissal of the batsman if the ball is Good and the batsman takes action k, $k \varepsilon C$, let $P_k^G(j)$ be the probability of scoring j runs ($j = 0, \ldots, 6$) if the ball is Good and the batsman takes action k, $k \varepsilon C$, and let $E_k^G(R)$ be the corresponding expected value of the runs scored on that ball. Similar definitions apply to $P_k^M(D)$, $P_k^M(j)$, $E_k^M(R)$, $P_k^B(D)$, $P_k^B(j)$ and $E_k^B(R)$. For each ball, ignoring such things as noballs or wides (i.e., keeping the total number of balls to 300), the batsman can either lose his wicket (including a run-out for attempting to take a single, but not for more than one run) or score j runs, $j = 0, 1, 2, 3, 4$ or 6.

Let $P(D)$ be the probability of dismissal and $P(j)$ be the probability of scoring j runs, $(j = 0-6)$, where we have omitted the subscripts and superscripts denoting the type of the ball bowled and the action taken. We have $P(D) + \sum_{j=0}^{6} P(j) = 1$.

3. First Innings

Let $f^G(n,i)$ be the maximum of the expected further score when n balls and i wickets are remaining, (n = 0–300, i = 0–10), and the ball currently being bowled is of type Good. Similar definitions apply to $f^M(n,i)$ and $f^B(n,i)$ for balls of type Medium and Bad respectively. Also, let $f\ (n,i)$ be the corresponding value when it is not known what type the ball being bowled is. Assuming a probability distribution $P(G)$, $P(M)$, $P(B)$ for Good, Medium and Bad balls respectively, $(P(G)+P(M)+P(B) = 1)$, we have

$$f(n,i) = P(G)f^G(n,i) + P(M)f^M(n,i) + P(B)f^B(n,i) \tag{3.1}$$

The dynamic programming equation for the case when the ball bowled is Good is

$$\begin{aligned} f^G(n,i) &= \operatorname*{Max}_{k\in C}\left\{P_k^G(D)f(n-1,i-1) + \sum_{j=0}^{6} P_k^G(j)(j + f(n-1,i))\right\} \\ &= \operatorname*{Max}_{k\in C}\{P_k^G(D)f(n-1,i-1) + (1 - P_k^G(D))f(n-1,i) + E_k^G(R)\} \end{aligned} \tag{3.2}$$

Note, (3.2) holds only because, on the right-hand side we are considering the situation when $n-1$ balls are remaining and we do not know what type the $(n-1)$th ball is; hence $f\ (n-1,i)$ and $f\ (n-1,i-1)$ have no superscripts. Equations similar to (3.2) can be written for the cases when the ball bowled is Medium or Bad.

Since the innings finishes when there are either no balls to be bowled nor wickets remaining, we have the boundary conditions:

$$f(0,i) = f^G(0,i) = f^M(0,i) = f^B(0,i) = 0, \quad i = 0 - 10 \tag{3.3}$$

$$f(n,0) = f^G(n,0) = f^M(n,0) = f^B(n,0) = 0, \quad n = 0 - 300 \tag{3.4}$$

Using the parameters given in Table 1, we have from (3.2)

$$\begin{aligned} f^G(n,i) = \text{Max}\{&0.02f(n-1,i-1) + 0.98f(n-1,i), \\ &0.1f(n-1,i-1) + 0.9f(n-1,i) + 0.5, \\ &0.2f(n-1,i-1) + 0.8f(n-1,i) + 1\} \\ f^M(n,i) = \text{Max}\{&0.01f(n-1,i-1) + 0.99f(n-1,i), \\ &0.05f(n-1,i-1) + 0.95f(n-1,i) + 0.65, \\ &0.15f(n-1,i-1) + 0.85f(n-1,i) + 1.5\} \end{aligned}$$

$$\begin{aligned} f^B(n,i) = \text{Max}\{&0.001f(n-1,i-1) + 0.999f(n-1,i), \\ &0.01f(n-1,i-1) + 0.99f(n-1,i) + 0.79, \\ &0.02f(n-1,i-1) + 0.98f(n-1,i) + 2\} \end{aligned} \quad (3.5)$$

Table 2. Maximum expected scores for Scenario 1.

Overs to go	Wickets in hand									
	1	2	3	4	5	6	7	8	9	10
1	7	9	9	9	9	9	9	9	9	9
5	19	28	35	39	42	44	45	45	45	45
10	27	43	54	62	69	75	79	83	86	88
15	32	54	69	80	89	97	104	110	115	120
20	35	62	81	95	106	116	124	132	139	145
25	37	68	90	107	122	133	143	151	159	167
30	38	72	98	118	134	148	160	170	178	187
35	39	74	104	127	146	161	175	186	196	206
40	39	76	109	135	156	173	188	202	213	223
45	40	78	112	141	165	184	201	216	229	240
50	40	78	114	146	172	194	212	228	243	256

Using (3.3) in (3.5) for $n = 1$, we obtain $f^G(1,i)$, $f^M(1,i)$, $f^B(1,i)$, and hence, given values of $P(G)$, $P(M)$ and $P(B)$, $f(1,i)$ using (3.1). These in turn are used in (3.5) for $n = 2$, etc., all the way to $n = 300$.

Table 2 gives the values of $f(n,i)$ by taking $P(G) = P(M) = P(B) = 1/3$, for $i = 1 - 10$ and a selection of values of n (converted to overs), and Table 3 gives the strategy employed for $i = 1 - 10$ and selected values of n (converted to overs), i.e., the optimal values of $k \varepsilon C$ which gives the maximum value of the expected score. The strategy denoted by *jkl* is short for using the strategies j, k, and l for ball types Good, Medium, and Bad respectively (j, k, $l = 1, 2, 3$), where as mentioned before, $1 \equiv$ defend, $2 \equiv$ single, and $3 \equiv$ attack. Between the two tables, Table 3 is the more important one as it gives the optimal strategy at each stage for each state. Table 2 merely gives the values of the maximum expected scores. The distribution of the strategies in Table 3 is as may be expected. The batsmen start with one strategy, and become more attacking if wickets do not fall but more defensive if wickets do fall.

Clarke's (1988) formulation of the problem is as follows: With the same notation as above, the dynamic programming equation is

$$\begin{aligned} f(n,i) &= \underset{R}{\text{Max}} \left\{ P(D)f(n-1,i-1) + \sum_{j=0}^{6} P(j)(j + f(n-1,i)) \right\} \\ &= \underset{R}{\text{Max}} \left\{ P(D)f(n-1,i-1) + \frac{R}{6} + (1 - P(D))f(n-1,i) \right\} \end{aligned}$$

where $R = \{0, 1, 2, \ldots, 12\}$ is the run rate per over and $P(D)$ depends only on R. He obtains a table similar to Table 2 and another with values from 0 to 12, giving the optimal run rate; this, in place of our Table 3. The difference between his solution

Table 3. Optimal strategies for Scenario 1.

Overs to go	Wickets in hand									
	1	2	3	4	5	6	7	8	9	10
1	233	333	333	333	333	333	333	333	333	333
5	113	123	133	333	333	333	333	333	333	333
10	113	123	123	133	133	233	333	333	333	333
15	113	113	123	123	123	133	133	233	233	333
20	113	113	113	123	123	123	133	133	133	133
25	113	113	113	113	123	123	123	123	133	133
30	113	113	113	113	123	123	123	123	123	133
35	113	113	113	113	113	123	123	123	123	123
40	113	113	113	113	113	113	123	123	123	123
45	113	113	113	113	113	113	113	123	123	123
50	113	113	113	113	113	113	113	123	123	123

and ours is as follows. For example, at stage $n = 150$ (25 overs to go) and $i = 5$, the optimal course of action for the batsman is 123, i.e., to defend if the ball is Good, play for a single if Medium and attack if Bad. Suppose the optimal value of R in Clarke's formulation for an equivalent situation is 5; the batsman playing an optimal strategy would attempt to score 5 runs in that over, irrespective of whether the ball faced is Good, Medium or Bad. If all the balls of that over are Good, then such a course of action would be disastrous; if on the other hand they happen to be Bad, then it would be a wasted opportunity.

4. First Innings: Change of Parameters

We shall now vary the parameters of the probability of dismissal and expected runs per ball. We shall show that for different values of these parameters we get different expected scores and strategies. Table 4 gives another scenario — called Scenario 2, and Table 5 gives the results of the computation of the maximum expected scores, where we have, once again, taken $P(G) = P(M) = P(B) = 1/3$. A table of strategies similar to Table 3 could be produced, but is not given here.

Table 5 exhibits some odd patterns. It shows that for each case of number of wickets in hand, the maximum expected score barely changes as the number of overs to go increases, and, in fact, converges to a limit. Secondly, the limit for i wickets in hand is about i times the limit for one wicket in hand.

To see why this is so, first consider when only one wicket is in hand. The strategy table similar to Table 3 for Scenario 1 (which is not given here) shows that for this case, apart from for five or less overs to go, and for 6 to 10 wickets in hand with 15 or less overs to go, the optimal strategy is mainly 133 with average $P(D) = 0.06$. From Table 4 we therefore have, using (3.2),

$$\begin{aligned} f^G(n,i) &= 0.03f(n-1,i-1) + 0.97f(n-1,i) \\ f^M(n,i) &= 0.10f(n-1,i-1) + 0.90f(n-1,i) + 1.8 \\ f^B(n,i) &= 0.05f(n-1,i-1) + 0.95f(n-1,i) + 2.2 \end{aligned} \tag{4.1}$$

Table 4. Parameters for Scenario 2.

Ball type	Action	Probability of dismissal	Prob(0)	Expected no. of runs
Good	1-Defend	0.03	0.97	0
	2-Single	0.12	0.38	0.5
	3-Attack	0.15		1.1
Medium	1-Defend	0.02	0.98	0
	2-Single	0.08	0.27	0.65
	3-Attack	0.1		1.8
Bad	1-Defend	0.01	0.99	0
	2-Single	0.03	0.17	0.8
	3-Attack	0.05		2.2

Table 5. Maximum expected scores for Scenario 2.

Overs to go	Wickets in hand									
	1	2	3	4	5	6	7	8	9	10
1	8	10	10	10	10	10	10	10	10	10
5	19	32	41	46	49	50	51	51	51	51
10	22	42	58	71	81	89	94	98	100	101
15	22	44	64	83	98	111	121	130	137	143
20	22	44	66	87	107	124	139	151	162	171
25	22	44	67	88	110	130	148	165	179	192
30	22	44	67	89	111	132	153	172	190	206
35	22	44	67	89	111	133	155	176	196	215
40	22	44	67	89	111	133	155	177	198	219
45	22	44	67	89	111	133	156	178	200	221
50	22	44	67	89	111	133	156	178	200	222

Since $f(n,0) = 0$, $\forall n$ we have from (4.1), and taking $P(G) = P(M) = P(B) = 1/3$

$$f(n,1) = 0.94 f(n-1,1) + 1.33333$$

giving the solution,

$$f(n,1) = A(0.94)^n + 1.33333/0.06 \tag{4.2}$$

where A is an arbitrary constant.

A ball by ball version of the maximum expected score table (Table 5) for this scenario shows that $f(9,1) = 10.4$. Putting this value in (4.2) we obtain $A = -20.6323$, so that

$$f(n,1) = -20.6323(0.94)^n + 22.22222 \tag{4.3}$$

This procedure can be extended to $i = 2 - 10$, where for the difference equation for $f(n,i)$ we use the known solution for $f(n,i-1)$. The part of the solution with powers of 0.94 converges to zero, giving $f(n,i) \to 1.3333\, i/0.06$.

Let us consider the difference between the two scenarios. For Scenario 1, there are two main strategies 113 and 123 with average probabilities of dismissal 0.03 and 0.0167 as against one dominant strategy 133 for Scenario 2 with average probability of dismissal 0.06. The fact that we have two strategies and that 0.97^n and 0.9833^n converge to 0 more slowly than 0.94^n means that (for the final score) there is no pronounced convergence to a value as in the case of Scenario 2. The above discussion shows that for a given set of parameters, we do not know, before carrying out the dynamic programming exercise, whether there is one or more than one predominant strategy, and if so what they are — as the parameters act in an unobvious manner to produce the strategies.

Another thing to note is that the values obtained in Table 5 seem to be at odds with the parameters for the average runs per ball. The strategy for the most part is 133, so that the expected number of runs per ball is 4/3. This would imply the expected score should be 400, and not 221.8 as given in the table. The discrepancy is explained when one considers the probability of dismissal using the strategy 133. The average probability of dismissal for the strategy is 0.06. This means on average a batsman lasts for $1/0.06 = 16.6667$ balls. The score per wicket is $16.6667 \times 1.3333 = 22.2217$ so that the expected total score for the innings is 222. However, with 16.6667 balls per wicket the number of balls on average for the innings is 166.667, so that about 133 balls are left at the end of the innings. If such a situation were to occur at a real match, most match analysts would argue that the batsmen were reckless — wrongly using the strategy 133, and had they used the strategy 123, say, it would have been better. A look at Table 4 suggests otherwise. The change in the dismissal probability for a Medium ball (M) from action 2 to 3 is very small, 0.02. On the other hand the increase in average runs scored is 1.15. Thus if the criterion is maximum expected score then the correct strategy for a Medium ball is 3. So in some situations aiming to last for 300 balls may not be the optimal strategy for a team. One of the cases where such a phenomenon occurred for the major part of the series was the series between India and New Zealand in the 2002/2003 season with 7 ODI matches. For India the average score was 149.7 and the average number of overs was 37.5. For New Zealand, the corresponding numbers were 155.28 and 34.7 respectively.

Also, note that it is possible to draw wrong conclusions from Table 5. Column 1 gives the maximum expected score with one wicket in hand as 22 from 20 overs to go onwards. One is tempted to draw the inference that the last pair of batsmen are not making any progress at all in terms of scoring runs. Of course, such a conclusion is erroneous. What Table 5 says is that at any stage with between 20 to 50 overs to go, the maximum expected additional score is 22 and this is so because with a probability of 0.06 of losing a wicket, the expected number of balls the partnership lasts is 16.67, and the expected number of runs scored is $1.33/0.0 = 22.2$, never mind how many balls are left in the innings.

Finally we note that the strategy recommended most times gives greater scoring rate than the average required. For example, for Scenario 1 the strategy at 50 overs to go and 10 wickets in hand is 123 giving the expected runs per ball as 0.8833. However the average run rate is $255.7/300 = 0.85$. This situation holds for most cases. It should be noted that Clarke (1988) reached similar conclusions.

The question arises, "What can we learn from the solution provided by dynamic programming that we do not from intuition or experience?" As is the case with most operations research techniques, analyzing a situation with dynamic programming gives a quantitative insight and could sometimes supplement qualitative reasoning. As an example, consider the problem faced by a team management, i.e., the coach, the captain, the selection committee etc., in selecting a team. The management usually has a squad of about fourteen members from which to choose a team. It is known that a lot of consideration is given to the choice of bowlers — spin, swing, seam etc. Consideration is also given to the team composition, i.e., the batsmen/bowlers/all rounders composition. Sometimes deleting batsmen to boost bowling is an option. Sometimes we have the competing options of selecting reliable but slow-scoring batsmen versus somewhat unreliable but fast scoring ones. Up till now we have assumed all batsmen to be equal. Obviously, in practice, this is not a reasonable assumption to make. The choice may be to have seven good batsmen (including one or more who would be able to bowl), with the remaining four spots in the team to be filled by bowlers of varying batting ability. In the next section we consider the two competing options of having four bowlers who can contribute substantially to the score (often called all-rounders) and having four bowlers who are unlikely to contribute significantly to the score.

5. Tailender Batsmen and the Fifth Bowler

We shall now consider the real-life situation when not all the batsmen nor the bowlers are of equal calibre. Typically, the batsmen from eight onwards — called tailender batsmen — are bowlers and their batting prowess is not of the same calibre as that of the top-order batsmen, who normally bat in positions 1 to 7. Similarly, since ten is the maximum number of overs to be bowled by any bowler, the fielding team has to use one batsman or more to act as "the fifth bowler". We shall here take these considerations into account by modifying the parameters of Scenario 1. We shall call this Scenario 3. For simplicity, we shall take the tailender effect into account by giving different parameters for dismissal probability and expected number of runs per ball, when $i = 0 - 4$, i.e., when six wickets have fallen, as then there will be at least one tailender batsman batting. Table 6 gives the parameters for this situation. This assumes that the last four batsmen are "all-rounders" and can contribute significantly to the score, but are not of the same ability as recognized batsmen. These parameters replace the parameters in Eq. (3.5) when $i \leq 4$. Table 6 also gives the parameters for the situation (Scenario 4) when the last four batsmen are primarily bowlers and are unlikely to contribute much to the score.

To take into account the lower bowling ability of the fifth bowler, we change the probabilities $P(G)$, $P(M)$, and $P(B)$ from 1/3 each to 0.2, 0.4, 0.4 for the duration when he is bowling in tandem with one of the main bowlers. Table 7 gives the results for the maximum expected score for Scenario 3. A table of optimal strategies could also be produced. A table for Scenario 4, similar to Table 7 for Scenario 3, has

Table 6. Batting parameters for tailenders for Scenarios 3 and 4.

Ball type	Action	Probability of dismissal		Prob(0)		Expected no. of runs	
		Scenario 3	Scenario 4	Scenario 3	Scenario 4	Scenario 3	Scenario 4
Good	1-Defend	0.02	0.05	0.98	0.95	0	0
	2-Single	0.15	0.2	0.35	0.6	0.5	0.2
	3-Attack	0.25	0.5			1	0.5
Medium	1-Defend	0.02	0.04	0.98	0.96	0	0
	2-Single	0.08	0.2	0.27	0.5	0.65	0.3
	3-Attack	0.18	0.3			1.5	0.8
Bad	1-Defend	0.001	0.03	0.999	0.97	0	0
	2-Single	0.02	0.2	0.19	0.4	0.79	0.4
	3-Attack	0.03	0.2			2	1

Table 7. Maximum expected scores for Scenario 3 (Scenario 1 with all-rounder tailenders and fifth bowler).

Overs to go	Wickets in hand									
	1	2	3	4	5	6	7	8	9	10
1	6	8	9	9	9	9	9	9	9	9
5	16	26	32	37	41	43	44	45	45	45
10	23	37	48	56	65	71	77	81	85	87
15	28	49	63	74	87	96	104	111	117	122
20	31	56	75	89	105	117	127	136	144	151
25	32	61	84	102	121	136	148	159	168	176
30	33	63	90	112	135	152	167	180	190	200
35	31	62	91	115	142	163	179	194	206	217
40	30	60	90	117	148	172	191	207	221	233
45	29	59	89	118	152	179	200	218	234	248
50	29	58	88	117	154	184	209	229	246	261

been determined but is not given here. The important quantity is the maximum expected score for the start of the innings. This comes to be 219, compared to 261 for Scenario 3.

6. Simulation

So far in this paper we have considered the problem entirely in terms of *expected* values — maximum *expected* scores, *expected* times of falling of wickets, etc. The discussion so far does not tell us anything about the variation in innings score that can result due to the intrinsic random nature of the game — the number of Good, Medium and Bad balls actually bowled, the occurrence of the random event of a wicket falling, etc. To get an idea about this variation, we simulated 1000 innings for Scenario 1. To simulate an innings we need the probability distribution of the number of runs scored per ball for each combination of the type of ball and the action taken by the batsman. For all three types of balls, and for the actions "defend" and "single", this is already given in Tables 1 and 3, and so it's only for the action "attack" we need to formulate a probability distribution. The authors

Table 8. Probabilities of dismissal and of scoring various runs from data (Ovens, 2005) and simulation.

	$P(D)$	$P(0)$	$P(1)$	$P(1)$	$P(3)$	$P(4)$	$P(6)$
Data	0.0245	0.5028	0.3258	0.0631	0.0103	0.0640	0.0095
Simulation	0.0336	0.4791	0.3069	0.0708	0.0117	0.0868	0.0111

Table 9. Probability distribution of runs by ball type and action "attack".

Batsman	Ball type	P(0)	$P(1)$	$P(1)$	$P(3)$	$P(4)$	$P(6)$	$E(R)$
Regular	Good	0.1195	0.537	0.06	0.01	0.064	0.0095	1.0
	Medium	0.34	0.07	0.18	0.03	0.2	0.03	1.5
	Bad	0.0695	0.43	0.189	0.031	0.232	0.0285	2.0

are not cricketers, even in the amateur class, and so rather than just guess values we decided to use empirical data. The data used are the relative frequencies per ball of the results — dismissal, 0, 1, 2, 3, 4 and 6 of the first innings of 173 matches played in Australia in which Australia was involved, during the past decade or so. The data was collected by Champion Data and is reproduced from Ovens (2005). This is given in the first row of Table 8. Note the values are not necessarily for the Australian teams. They are for the teams batting first; on average this would be for the Australian team for about 85 matches.

It is clear that we cannot take the empirical distribution as the distribution of number of runs for the action "attack", for the simple reason that the empirical data is over all balls bowled, and over all actions, whereas we need a distribution only when the action is "attack", which is used for about a third of the balls. In fact, since the strategy predominantly used is 123, and in some cases 133, it's only for the combination Bad/attack, and to some extent Medium/attack, that the probability distribution is of any significance. Accordingly, we take the distribution as in Table 9, where we have also taken into account the previously stipulated expected run rates of 1, 1.5 and 2. Essentially, we trebled the empirical probabilities for scores 2, 3, 4, and 6 and adjusted the probabilities for 0 and 1 so that we have total probability 1 and the requisite expected run rate.

Table 10 summarizes the results of 1000 simulations for Scenario 1, giving descriptive statistics for the number of wickets lost, total score achieved, number of balls remaining, and numbers of zeros, ones, twos, threes, fours and sixes scored. These results are truly amazing — particularly the wide variation between the minimum score (106) and the maximum score (401), and the value of the standard deviation (44.56). It is important to remember that this variability is not due to the strength or weakness of the teams involved, nor due to the weather or batting conditions, nor due to the poor judgement — recklessness — of the batsmen. The parameters regarding probability of dismissal and average number of runs per ball scored were kept constant. Moreover in the way dynamic programming is formulated, the batsmen concerned have no latitude about the kind of stroke to

Table 10. Summary statistics for a simulation of 1000 games (Scenario 1).

Variable	Mean	SE Mean	St. Dev	Minimum	Median	Maximum
Wickets	9.6160	0.0205	0.6473	6	10	10
Score	256.95	1.41	44.56	106	260.0	401
Balls remaining	13.635	0.693	21.926	0	4.5	138
Zeros	137.21	0.499	15.78	82	135	191
Ones	87.885	0.529	16.739	26	91	132
Twos	20.279	0.176	5.576	6	20	39
Threes	3.357	0.0605	1.9140	0	3	11
Fours	24.844	0.202	6.378	9	24	46
Sixes	3.1770	0.0576	1.8224	0	3	11

play — it is completely determined by the strategy stipulated, which itself depends on the number of wickets and the number of balls remaining. The variability in the score is due to the inherent variability involved in the exercise. For example, although the ratio of Good, Medium, Bad balls is 1:1:1, in any particular simulation the number of G, M, B balls could differ significantly from the expected 100:100:100. Moreover, whether the batsman is dismissed on a particular ball is a random event. The probability distribution of the scores per ball obtained from the simulation have been calculated (assuming $300 - 13.635 = 286.365$ balls per innings) and are given in the second row of Table 8. It is interesting that they are fairly close to the empirical ones. Finally, the results are in general agreement with the empirical finding — Ovens (2005) — that the total score for completed first innings in Australia in recent years has a normal distribution with mean 263.8 and standard deviation 42.3.

7. Second Innings

We shall now consider the second innings. At the start of the second innings, the team batting knows exactly how many runs it needs to score to win the match.

Let $P^G(n, s, i)$ be the maximum probability of scoring at least s runs when n balls and i wickets are remaining, ($n = 0 - 300, i = 0 - 10, s = 0 - s_{\max}$), and the ball currently being bowled is of type Good. Similar definitions apply to $P^M(n, s, i)$ and $P^B(n, s, i)$ for balls of type Medium and Bad respectively. Also, let $P(n, s, i)$ be the corresponding value when it is not known what type the ball being bowled is. Assuming a probability distribution $P(G)$, $P(M)$, $P(B)$ for Good, Medium and Bad balls respectively, ($P(G) + P(M) + P(B) = 1$), we have

$$P(n, s, i) = P(G)P^G(n, s, i) + P(M)P^M(n, s, i) + P(B)P^B(n, s, i) \tag{7.1}$$

The dynamic programming equation for the case when the ball bowled is good is

$$P^G(n, s, i) = \operatorname*{Max}_{k \in C} \left\{ P_k^G(D)P(n-1, s, i-1) + \sum_{j=0}^{6} P_k^G(j)P(n-1, s-j, i) \right\} \tag{7.2}$$

Equations similar to (7.2) can be written for the cases when the ball bowled is Medium or Bad.

Since the innings finishes when there are either no further score to be made, no balls to be bowled nor wickets remaining, we have the boundary conditions:

$$P(n,0,i) = P^G(n,0,i) = P^M(n,0,i) = P^B(n,0,i) = 1, \quad n = 0-300, \quad i = 0-10 \tag{7.3}$$

$$P(0,s,i) = P^G(0,s,i) = P^M(0,s,i) = P^B(0,s,i) = 0, \quad s = 0 \text{ to } s_{\max}, \quad i = 0-10 \tag{7.4}$$

$$P(n,s,0) = P^G(n,s,0) = P^M(n,s,0) = P^B(n,s,0) = 0, \quad n = 0-300, \quad s = 0 \text{ to } s_{\max}, \tag{7.5}$$

We assume that the team batting second has the same parameters as those of the team batting first. Also we shall assume for simplicity that $P(G) = P(M) = P(B) = 1/3$ for the whole innings, and that the batsmen and the bowlers are of equal quality. Note that for the second innings we need probabilities of scoring runs (0 to 6) for each ball type/action combination rather than just the expected values of runs scored. Table 9 is Table 2 with such probabilities.

Using the parameters given in Table 9, we have from (7.2)

$$\begin{aligned}
P^G(n,s,i) = \mathrm{Max}\{&0.02P(n-1,s,i-1) + 0.98P(n-1,s,i),\\
&0.1P(n-1,s,i-1) + 0.4P(n-1,s,i) + 0.5P(n-1,s-1,i),\\
&0.2P(n-1,s,i-1) + 0.1195P(n-1,s,i) + 0.537P(n-1,s-1,i)\\
&+ 0.06P(n-1,s-2,i) + 0.01P(n-1,s-3,i)\\
&+ 0.064P(n-1,s-4,i) + 0.0095P(n-1,s-6,i)\}\\
P^M(n,s,i) = \mathrm{Max}\{&0.01P(n-1,s,i-1) + 0.99P(n-1,s,i),\\
&0.05P(n-1,s,i-1) + 0.3P(n-1,s,i) + 0.65P(n-1,s-1,i),\\
&0.15P(n-1,s,i-1) + 0.34P(n-1,s,i) + 0.07P(n-1,s-1,i)\\
&+ 0.18P(n-1,s-2,i) + 0.03P(n-1,s-3,i)\\
&+ 0.2P(n-1,s-4,i) + 0.03P(n-1,s-6,i)\} \qquad (7.6)\\
P^B(n,s,i) = \mathrm{Max}\{&0.001P(n-1,s,i-1) + 0.999P(n-1,s,i),\\
&0.01P(n-1,s,i-1) + 0.2P(n-1,s,i) + 0.79P(n-1,s-1,i),\\
&0.02P(n-1,s,i-1) + 0.0695P(n-1,s,i) + 0.43P(n-1,s-1,i)\\
&+ 0.189P(n-1,s-2,i) + 0.031P(n-1,s-3,i)\\
&+ 0.232P(n-1,s-4,i) + 0.0285P(n-1,s-6,i)\}
\end{aligned}$$

Of course to implement these we also have to add additional boundary conditions (7.7) to allow for the batsman to score more than the minimum required runs to win a match. (For example, if one run is required for a win, the batsman may actually hit a six.) These are

$$P(n,s,i) = P^G(n,s,i) = P^M(n,s,i) = P^B(n,s,i) = 1,$$
$$n = 0 - 300, \quad i = 0 - 10, \quad s = \{-5,-4,-3,-2,-1\} \tag{7.7}$$

Using (7.3), (7.4), (7.5) and (7.7) in (7.6) for $n = 1$ and $s = 1$ we obtain $P^G(1,1,i)$, $P^M(1,1,i)$, $P^B(1,1,i)$, for $i = 1-10$, and hence, given values of $P(G)$, $P(M)$ and $P(B)$, $P(1,1,i)$ using (1) for $i = 1-10$. These in turn are used in (7.6) for $s = 2$, etc., all the way to $s = 400$. (This last score is about the maximum obtained from the simulation, and has been exceeded in some first innings totals.) This procedure is then repeated for $n = 2$ to $n = 300$.

Table 11 gives the values of $P(n,s,i)$ with 300 balls to go and 10 wickets in hand (that is, at the start of an innings) by taking $P(G) = P(M) = P(B) = 1/3$, for a selection of values of s. These are the maximum probabilities of a win for the team batting second at the start of its innings. Similar tables could be produced for any number of balls to go, wickets in hand and runs to be scored.

We note that, for Scenario 1, the probability of the second team achieving a score greater than 256 (the maximum expected score calculated for the first innings) is approximately 0.580. Similar calculations for Scenario 2 show that the probability of the team batting second achieving a score greater than 222 (the maximum expected score calculated for the first innings) is approximately 0.457. Scenario 1 is an example where the first innings is expected to last for the full 300 balls, whereas for Scenario 2 the first innings is expected to last only 168 balls.

We shall now derive, for the team batting second, the maximum probability of outscoring the team batting first, which had used the criterion of maximizing its expected score.

Table 11. Maximum probability of scoring at least s runs with 300 balls to go and 10 wickets in hand for Scenario 1.

Score s to get	Probability	Score s to get	Probability
200	0.9239435	251	0.6315833
225	0.8133846	252	0.6231571
250	0.6399076	253	0.6146310
275	0.4107652	254	0.6060079
300	0.2025863	255	0.5972905
325	0.0801837	256	0.5884822
350	0.0268789	257	0.5795868
		258	0.5706083
		259	0.5615510
		260	0.5524199
		261	0.5432200
		262	0.5339568
		263	0.5246361
		264	0.5152641
		265	0.5058471
		266	0.4963918

Let X and Y be the scores of the teams batting first and second respectively using the given optimization criteria. From the second innings calculations, we obtain $P(Y > E(X))$, but what we would like is $P(Y > X)$. Let $f(x)$ be the probability density function of X, and let $g(x) = P[Y > x|X = x]$. Then

$$P[Y > X] = \int g(x)f(x)dx = E(g(X))$$

Since the form of neither $g(x)$ nor $f(x)$ is known, we cannot work out this integral. However an approximate value can be obtained as follows. We have, from simulation of the first innings, 1000 scores denoted $x_1, x_2, \ldots, x_{1000}$. We can thus estimate

$$P[Y > X] \approx \frac{1}{1000}\sum_{i=1}^{1000} P(300, x_i + 1, 10)$$

where $P(300, x_i + 1, 10)$ is the maximum probability of scoring at least $x_i + 1$ runs in the second innings.

Table 12 gives a summary of these probabilities for a small sample of scenarios, some of which (Group I) had first innings expected to last for 300 balls, and some of which (Group II) had first innings expected to finish earlier. Scenarios A and D are the same as Scenarios 1 and 2 discussed earlier.

It is interesting to compare these theoretical values of the percentage of wins for the second team with their actual (historical) values over the past six years. Table 13 gives a summary of the number and percentage of wins by the team batting second

Table 12. Probabilities of a win for the team batting second in a 50 over match.

Group	Scenario	$P(Y > E(X))$	$P(Y > X)$
I	A	0.580	0.566
	B	0.540	0.585
	C	0.576	0.588
II	D	0.457	0.522
	E	0.519	0.527
	F	0.465	0.541

Table 13. Number and percentage of wins for teams batting second in completed One Day International Matches from 2001–2006 (Draws are excluded).

Year	Number of wins for team batting second	Percentage of wins for team batting second	No of matches with win/loss result	No of draws
2001	60	50	120	0
2002	61	45	137	1
2003	69	49	141	1
2004	64	52	122	0
2005	47	47	100	2
2006	85	55	154	0
Total	386	49.9	774	4

in completed One Day International games from 2001 to 2006, with draws excluded. From the table, there does not appear to be any advantage for a team to bat second.

Acknowledgments

We would like to thank the two referees for their valuable suggestions and one of the editors for his editorial comments which have considerably improved this paper.

References

Bellman, R (1957). *Dynamic Programming.* New Jersey: Princeton University Press.

Clarke, SR (1988). Dynamic programming in one-day cricket — Optimal scoring rates. *Journal of the Operational Research Society*, 39(4), 331–337.

Clarke, SR and JM Norman (2004). Dynamic programming in cricket — Choosing a night watchman. *Journal of the Operational Research Society*, 54, 838–845.

Kohler, DA (1982). Optimal strategies for the game of darts. *Journal of the Operational Research Society*, 33, 871–884.

Norman, JM (1995). Dynamic programming in sport: A survey of applications. *IMA Journal of Mathematics Applied in Business and Industry*, 6, 171–176.

Norman, JM and SR Clarke (2004). Dynamic programming in cricket — Batting on a sticky wicket, in *Proceeding of 7th Australasian Conference on Mathematics and Computers in Sport*, RH Morton and S Ganesalingam (eds), pp. 226–232.

Ovens, MJV (2005). Optimal batting strategies in one-day cricket. Unpublished Honours Thesis, Monash University.

Preston I and T Jonathan (2000). Batting strategy in limited overs cricket. *The Statistician*, 49(1), 95–106.

Pam Norton was initially educated at Sydney where she obtained her BSc degree in Pure Mathematics, and then a PhD in Mathematics from the University of Warwick. She has been a lecturer in the School of Mathematical Sciences at Monash University from 1986–2006, and took early retirement in 2007. She has published papers on mathematics education and sporting applications of mathematics.

Ravi Phatarfod was educated at Bombay where he obtained his BSc degree in Mathematics and an MSC degree in Statistics. He later got a PhD in Statistics at the University of Manchester. He was a lecturer in Mathematics at the University of Western Australia during 1959–1961. For the next two years he was the Head of the OR and Productivity Division of SASMIRA, Bombay. He joined the Department of Mathematics, Monash University as a Senior Lecturer, and was promoted to Reader in 1983, and Director of the Statistics section in 1994. He has published about 70 papers, mainly in sequential analysis, stochastic hydrology and computer search algorithms. He took early retirement in December 1996.