The Trident: Improving the Duckworth-Lewis Method Using Bayesian Inference

Stats C116 Final Project

Anish Deshpande, Bennett Kedrosky

2025-05-29

Abstract:

In rain-affected T20 cricket matches, estimating target scores fairly and accurately is a difficult challenge. The Duckworth-Lewis (DL) method addresses this by estimating a batting team's remaining resources using 2 variables: overs left and wickets lost. However, the DL method suffers from key limitations like a non-monotonic resource table and a lack of adaptability to modern gameplay. In this project, we propose *The Trident*, a Bayesian model that generates an improved, data-driven resource table that better replicates the changing strategies during T20 matches. Leveraging a dataset of roughly 10,000 professional T20 innings from Cricsheet, we constructed a smoothed, monotonic estimate of scoring potential using Stan-based Bayesian inference. Our model captures more realistic game dynamics by explicitly modeling 3 separate innings phases (powerplay, middle overs, and death overs) and accounts for a monotone structure in its resource table. Using RMSE ratios between *The Trident* and the DL method, empirical results show that *The Trident* outperforms the DL, especially during the powerplay and middle overs. These improvements highlight the promise of Bayesian modeling for addressing shortcomings in the DL method.

Introduction / Problem Statement:

T20 Cricket is the shortest international format of the game and has seen an explosion in popularity over the last 2 decades because of its fast pace and strategic complexity. Compared to other formats, which can last between 8 hours and sometimes even 5 days, the T20 game is well suited for the modern day viewer. Unlike Test or One Day matches, T20 games are limited to a single inning per team, with each inning restricted to 20 overs (i.e. 120 legal deliveries, or "pitches" in baseball terms). The match begins with a coin toss, where the winning team elects to either bat or bowl first. Team A bats first, aiming to score as many runs as possible before either losing all 10 wickets or completing 20 overs. Team B then attempts to chase down this target, and the match concludes when the target is surpassed, the full quota of 20 overs is bowled, or all 10 wickets are lost.

What makes T20 cricket particularly dynamic is the high scoring rate and rapid momentum shifts. Due to the limited duration, especially in the early overs, batters tend to play aggressively, aiming for boundaries rather than survival. However, this risk-heavy approach comes at the expense of potential wicket loss: teams that lose early wickets often shift to a more conservative style of play to preserve their remaining batters. The balance between aggression and caution is a function of the number of overs left, runs on the board, and wickets in hand.

Unlike other outdoor sports like baseball, soccer, and American football, cricket requires dry weather. Moderate to heavy rains can cause trouble to fast bowlers who run in hard to deliver the ball, needing lots of grip and traction. Additionally, batters use wooden bats which can easily depreciate in quality when exposed to water. For this reason, games are often temporarily stopped or called off when there is rain. In most cricket games, there is a set period of time allocated for the game, and thus if there is a rain delay of 1 hour, we can expect that 1 hour of game time will be cut off. In other words, the batting team will have less than 20 overs to bat. Take, for example, a team who had initially planned to bat as if they had a full 20 overs to chase down a target of 150 runs, but a mid innings rain delay shortens their innings to just 10 overs. To generate a fairly revised target score, the Duckworth-Lewis method is used.

The DL method, developed in 1997 and adopted by the international cricket board in 1999 (later improved into the DLS method) is a system to fairly revise target scores when interruptions in play happen, and it stems from the basic idea that a batting team has 2 main resources at their disposal when they want to score runs:

1. The number of overs remaining
2. The number of wickets remaining

The Duckworth-Lewis method for T20 games has an official resource table, which states the percentage of total resources a team has available for every possible combination of overs remaining and wickets lost. Below is the standard resource table used for T20 games as per the DL method.

Duckworth-Lewis Standard Resource Table for T20s

u	w = 0	w = 1	w = 2	w = 3	w = 4	w = 5	w = 6	w = 7	w = 8	w = 9
20	100.0	96.8	92.6	86.7	78.8	68.2	54.4	37.5	21.3	8.3
19	96.1	93.3	89.2	83.9	76.7	66.6	53.5	37.3	21.0	8.3
18	92.2	89.6	85.9	81.1	74.2	65.0	52.7	36.9	21.0	8.3
17	88.2	85.7	82.5	77.9	71.7	63.3	51.6	36.6	21.0	8.3
16	84.1	81.8	79.0	74.7	69.1	61.3	50.4	36.2	20.8	8.3
15	79.9	77.9	75.3	71.6	66.4	59.2	49.1	35.7	20.8	8.3
14	75.4	73.7	71.4	68.0	63.4	56.9	47.7	35.2	20.8	8.3
13	71.0	69.4	67.3	64.5	60.4	54.4	46.1	34.5	20.7	8.3
12	66.4	65.0	63.3	60.6	57.1	51.9	44.3	33.6	20.5	8.3
11	61.7	60.4	59.0	56.7	53.7	49.1	42.4	32.7	20.3	8.3
10	56.7	55.8	54.4	52.7	50.0	46.1	40.3	31.6	20.1	8.3
9	51.8	51.1	49.8	48.4	46.1	42.8	37.8	30.2	19.8	8.3
8	46.6	45.9	45.1	43.8	42.0	39.4	35.2	28.6	19.3	8.3
7	41.3	40.8	40.1	39.2	37.8	35.5	32.2	26.9	18.6	8.3
6	35.9	35.5	35.0	34.3	33.2	31.4	29.0	24.6	17.8	8.1
5	30.4	30.0	29.7	29.2	28.4	27.2	25.3	22.1	16.6	8.1
4	24.6	24.4	24.2	23.9	23.3	22.4	21.2	18.9	14.8	8.0
3	18.7	18.6	18.4	18.2	18.0	17.5	16.8	15.4	12.7	7.4
2	12.7	12.5	12.5	12.4	12.4	12.0	11.7	11.0	9.7	6.5
1	6.4	6.4	6.4	6.4	6.4	6.2	6.2	6.0	5.7	4.4

By putting on our Bayesian hats, we aim to create a model that performs better than the official DL resource table.

DL Method Explained:

The DL method assumes an exponential decay relationship with 2 sets of parameters. The model is as follows:

$$R(u,w) = a_w \left(1 - e^{-b_w u}\right), \quad w \in \{0,1,2,\ldots,9\}, \quad u \in \{0,1,2,\ldots,20\}$$

$R(u,w)$ represents the average runs scored by a team from the point in the inning where u overs remain and w wickets have been lost, until the end of the inning.

The parameters are a_w and b_w, and they depend on the value of w. Unfortunately, the exact values of these parameters for the DL method are not publicly available. In the context of the problem, the a_w parameter states the average runs scored by a team in the remaining overs with w wickets lost, while the b_w parameter is the decay rate, explaining how fast resources decay at the fall of every additional wicket lost and over played.

To obtain the final resource table, Duckworth-Lewis used the following equation:

$$P(u, w) = R(u, w)/R(20, 0)$$

This would make sure that $P(20, 0) = 100\%$, while $P(0, 9) = 0\%$

Here are some pros and cons of the DL method which drive our motivation to introduce a better model using Bayesian methods:

Pros:

- **Simplicity**: The DL method is straightforward to apply in match settings because only the resource table and a calculator is required. Fans are able to easily compute the targets using the 2 resources: overs and wickets.
- **Accuracy**: The DL method gives a reasonable and sensible target in most situations

Cons:

- **Non-Monotonic Resource Table**: For certain overs remaining and wickets lost, the resource percentage is constant (ex. 9 wickets lost). This contradicts our intuitive understanding of cricket that each loss of an over or wicket should reduce scoring capabilities.
- **Lack of Adaptivity**: The DL table is static and does not evolve with changes in T20 strategies over time. It weights matches played 20 years ago the same as games today, even though mean scores have increased.
- **Biased Assumptions**: It assumes that teams always aim to maximize runs, which is true for the first inning (the inning the resource is modeled on), but less so for the second inning where chasing teams prioritize winning over maximizing total score. This introduces potential bias in seconding-innings adjustments.

How does the resource table work?

The central purpose of the resource table in the DL method (and in our Bayesian model) is to quantify how much scoring potential remains for a batting team at any given point in the inning, based on the number of overs remaining and wickets lost. This value is expressed as a percentage of total resources available at the start of the inning.

If a team has R% resources remaining with u overs remaining and w wickets lost, and we assume that the full 100% of resources (20 overs remaining and 0 wickets lost) would allow them to score Z runs, then the expected final score is:

$$\text{Predicted Final Score} = \frac{\text{Current Score}}{(1 - \frac{R}{100})}$$

Data Collection:

The data for this project was collected from cricsheet.org, an open-access platform that provides detailed ball-by-ball records of professional cricket matches. We downloaded files for over 18,500 matches played over the past 20 years, including both international fixtures and major domestic T20 leagues such as the Indian Premier League (IPL), Big Bash League (BBL), and Caribbean Premier League (CPL).

Each JSON file contains lots of structured information, including team rosters, innings summaries, and individual ball events (e.g., runs scored, wickets taken, extras, and player dismissals). We extracted and cleaned the relevant game-level and delivery-level features. This involved resolving player identifiers, inferring innings structure, handling missing or corrupt entries, and standardizing cumulative run and wicket tallies across deliveries.

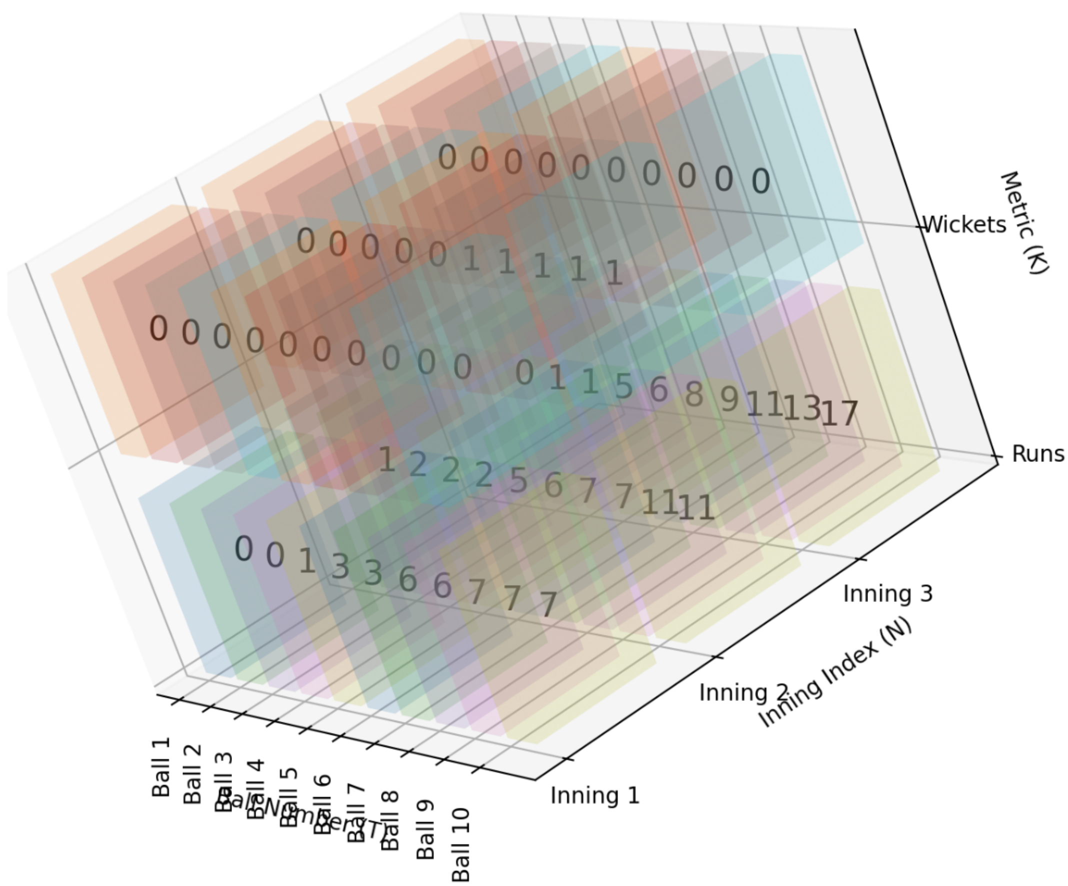
To manage the data at scale, we stored it in a MySQL relational database hosted on AWS. Tables were designed to hold match metadata, player mappings, team configurations, and full ball-by-ball logs, enabling flexible querying for further analysis. We used `SQLAlchemy` for structured interaction with the database.

For the purposes of modeling, we focused specifically on T20 matches and extracted all innings into a 3-dimensional tensor of shape $(22312, 120, 3)$, where:

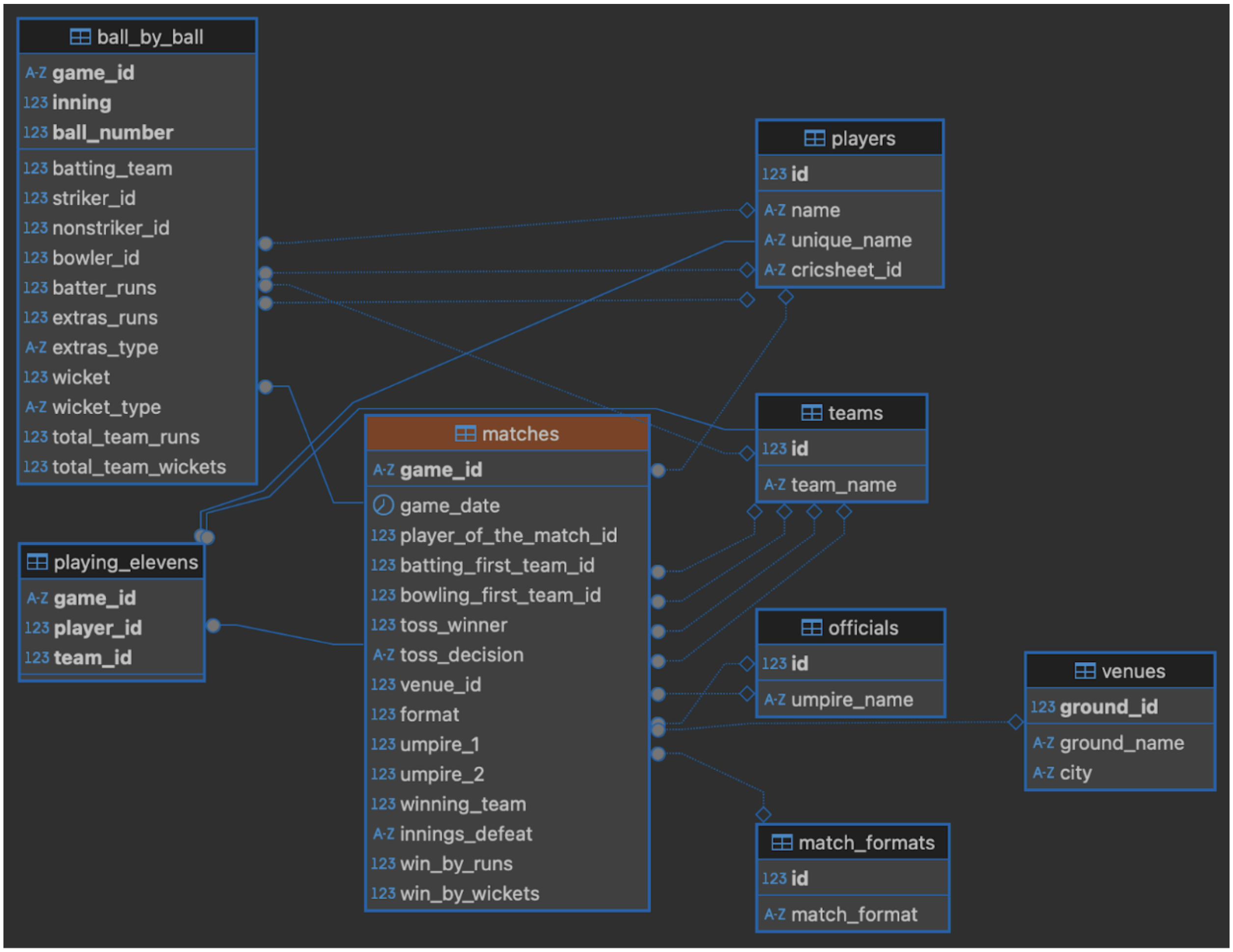
- The first dimension corresponds to each individual inning.
- The second dimension spans the 120 legal deliveries (balls) of a full 20-over inning.
- The third dimension holds 3 cumulative metrics: total runs scored, total wickets lost, and whether it is either the first or second inning of the game.

The tensor can be roughly visualized using this graphic:

Tensor Visualization (N=3, T=10, K=2)



And the relational database structure is as follows:



Note that the data collected in the database includes all types of game formats (T20, ODI, Test, etc.), and attempts to retain as much information as possible from the raw JSON files. Even though the rest of our analysis focuses solely on the T20 games, we believe it will be useful to store as much information as possible, so that if we would like to do further analysis in the future or perhaps a comparative analysis between formats, we will have that data readily available. Lastly, we split the data using a 80/20 split in which the training data has 8,359 innings and the testing data has 2,090 innings.

Methods:

Nonparametric Approach:

Because the DL resource table is not monotone, we decided to first implement a nonparametric approach, similar to what was discussed in the paper by Bhattacharya et al. 2011. First, we filter units in the tensor which only correspond to first innings. Duckworth and Lewis argued that only first innings data is relevant to producing resource percentages, since teams batting first aim to maximize runs, while teams batting second employ a different strategy that aims to simply beat the other teams score, and this doesn't necessarily optimize run scoring. Next, out of those first innings, we further filter for only complete innings. We define a complete inning as an inning which lasted all 120 balls. In other words, we don't want to include shortened or abandoned innings that end up being affected by delays like rain. Our resulting training tensor post-filtering and post train/test split now contains $N = 8,359$ innings, which was less than half of our initial amount of data of $N = 22,312$ innings.

Once we have our cleaned tensor data, we use the following strategy for our nonparametric approach (note that Bhattacharya et al. 2018 use the same nonparametric approach, however they create the resource table for 50 over games, while we do the same for T20 games):

- Define $R(u, w(u))$ as the runs scored from the stage in the innings at which u overs are available and $w(u)$ wickets are lost, until the end of the innings
- Calculate $R(u, w(u))$ for all values of u that occurred in the first innings.
- For each combination of u and w, average $R(u, w(u))$, and then average that value by $R(50, 0)$

The resulting nonparametric resource table is not monotonic and contains many missing values, since there are certain extreme (u, w) combinations which do not occur in our samples of innings.

Non-Parametric approach resource table

Overs	0 Wkts	1 Wkt	2 Wkts	3 Wkts	4 Wkts	5 Wkts	6 Wkts	7 Wkts	8 Wkts	9 Wkts
20	0.995	0.933	—	—	—	—	—	—	—	—
19	0.967	0.921	0.865	0.715	—	—	—	—	—	—
18	0.931	0.891	0.839	0.748	0.939	—	—	—	—	—
17	0.890	0.858	0.804	0.761	0.634	0.482	—	—	—	—
16	0.849	0.818	0.775	0.734	0.646	0.666	0.469	—	—	—
15	0.807	0.775	0.738	0.702	0.638	0.538	—	0.444	—	—
14	0.764	0.734	0.702	0.665	0.634	0.538	0.459	0.444	—	—
13	0.728	0.695	0.670	0.629	0.596	0.551	0.456	0.444	—	—
12	0.686	0.656	0.632	0.599	0.563	0.517	0.420	0.533	—	—
11	0.641	0.618	0.592	0.564	0.527	0.483	0.443	0.385	—	—
10	0.594	0.578	0.552	0.528	0.493	0.445	0.420	0.340	0.552	—
9	0.547	0.532	0.510	0.490	0.458	0.409	0.385	0.340	0.336	—
8	0.493	0.490	0.464	0.447	0.421	0.383	0.352	0.310	0.303	0.159
7	0.446	0.437	0.419	0.403	0.382	0.352	0.325	0.267	0.298	0.152
6	0.396	0.382	0.371	0.356	0.340	0.317	0.288	0.253	0.232	0.149
5	0.339	0.324	0.315	0.308	0.293	0.274	0.253	0.222	0.206	0.155
4	0.279	0.263	0.261	0.253	0.244	0.228	0.212	0.192	0.171	0.150
3	0.210	0.202	0.199	0.195	0.190	0.178	0.161	0.153	0.139	0.122
2	0.162	0.134	0.134	0.134	0.128	0.123	0.113	0.104	0.095	0.084
1	0.081	0.063	0.063	0.065	0.062	0.059	0.057	0.050	0.048	0.042
0	—	—	—	—	—	—	—	—	—	—

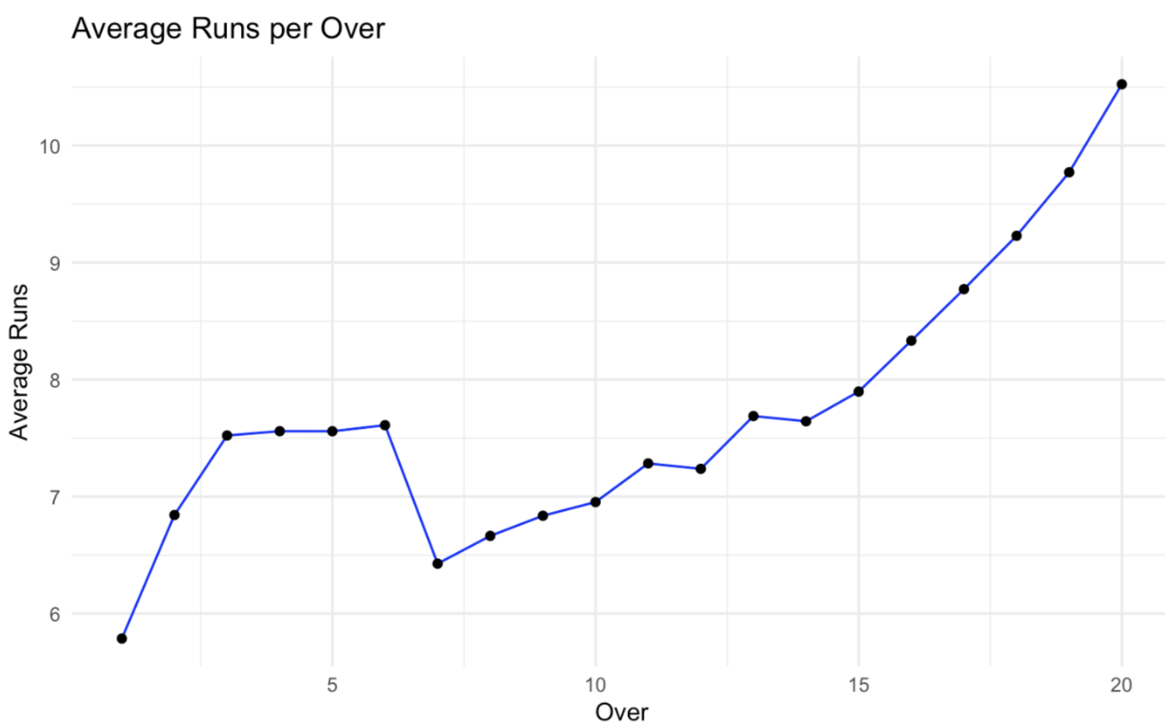
Why Bayesian?

We decide to use a Bayesian approach to model the resource percentage table due to the missing values in the nonparametric table as well as the lack of monotonicity. The Bayesian approach will allow us to impute the missing values in the resource table given monotonicity constraints. For the rest of our analysis, we build off of Bhattacharya et al. 2018 by using a similar baseline framework for our model constraints, however we make a few key changes to improve the accuracy of our model. The approach by Bhattacharya et. al. 2018 uses the RJAGS library in R to create their resource table, while **we use the newer RSTAN library** in R to generate our resource table. Secondly, **we account for the 3 different phases** in T20 games, which the other approach does not consider.

The Powerplay in T20 Cricket:

In limited overs cricket, the powerplay is a phase at the start of each innings. In 50 overs cricket, the powerplay goes from overs 1 to 10, while in T20 cricket, the powerplay lasts from overs 1 through 6. During the powerplay, the fielding side is only allowed to have a maximum of 2 outfielders. After the powerplay is over, the fielding side is allowed to keep up to 5 outfielders.

The reduced number of outfielders during the powerplay often makes it easier for the batters to score runs, as long as they either find a gap between fielders or hit it over the infielders. We can see the effect this has on average runs scored per over during the course of the game:



When analyzing thousands of T20 innings, there is a clear shift in scoring rates during different phases of each inning. This is how we have divided *The Trident*:

1. Overs 1-6 (Powerplay phase):

- The average run rate increases rapidly from 5.8 in the 1st over to 7.6 in the 6th over. This is a result of the fielding restrictions in place during the powerplay that allow batters to take greater risks and accelerate early scoring.

2. Overs 7-15 (Middle Phase):

- After the power play ends, average run rate dips noticeably–especially in over 7–before gradually increasing again. This drop is consistent with more defensive field settings and in many cases, the introduction of spin bowlers. Teams tend to try to consolidate in this phase by minimizing the number of wickets lost while still maintaining a steady run rate. There are often fewer boundaries scored during the middle phase, as batters will look to rotate strike through singles and doubles.

3. Overs 16-20 (Death Overs):

- From around the 16th over onward, there is a sharp increase in run rate, with a peak of 10.5 run rate in the final over. This rise in run rate is due to the aggressive approach adapted by batters in late-innings. With minimal overs left, a riskier batting style is common because the marginal value of wickets decreases, which allows teams to more aggressively target a higher run rate.

Implications for Modeling:

Resource models should account for these 3 different phases of the game in order to best predict target scores. The standard DL method nor Bhattacharya et. al. 2018 explicitly models the power play or other tactical phases. As a result, its resource percentages assume the same underlying decay model across all overs, which doesn't reflect real-world scoring and strategy that is dynamic across overs.

Therefore, our Bayesian model, using RSTAN, incorporates a more dynamic and realistic structure by allowing different parameters for each phase. This allows the resource percentages to better model the nonlinear and phase-sensitive scoring patterns seen in actual T20 matches.

To best model the changing strategies during the inning, we utilized phase-specific scaling parameters. These scaling parameters adjust the decay rate, enabling the model to more realistically reflect the aggressive early innings behavior, consolidation in the middle overs, and acceleration in the death overs. By explicitly modeling each phase with its own functional shape, we improve the model's flexibility and predictive accuracy while still preserving monotonicity. By treating these 3 phases of the game as individual elements, we were able to tweak the model to better represent the realities of T20 cricket strategies.

How RSTAN works:

We used RSTAN, a Bayesian statistical modeling and computation platform, to obtain the full posterior distributions for model parameters using HMC. Stan is faster in posterior sampling than Gibbs sampling which makes it an ideal choice for this

We modeled the expected remaining resources (i.e., the proportion of total scoring potential left) as a function of 2 key variables: overs remaining u and wickets lost w. We aim to estimate a smooth, monotonic surface and impute missing or sparsely observed values in our nonparametric estimates.

In RStan, the model is specified with 3 key blocks:

1. Data Block

- Specifies inputs like observed outcomes y, overs u, wickets w, and the precision modifier nuw.

2. Parameters Block

- Declares parameters a_w, b_w in transformed monotonic form:
  - $a_0 \sim Uniform(0, 3000)$, and differences $\triangle a_j \geq 0$ ensure that a_w is non-increasing in w
  - $b_0 \sim Uniform(0, 100)$ and differences $\triangle b_j \geq 0$ enforces b_w is non-decreasing, ensuring concavity in overs.

3. Model Block

- For each data point i, the likelihood is:

$$y_i \sim \mathcal{N}\left(a_{w_i}\left(1 - \exp(-b_{w_i} u_i)\right), \sqrt{\frac{\sigma^2}{\text{nuw}_i}}\right)$$

This approach ensures that:

- $\mu(u, w)$ increases with u for fixed w
- $\mu(u, w)$ decreases with w for fixed u

This accounts for varying precision depending on data availability across (u, w) states. As you can see from the following table, there are large discrepancies between nuw values, as some states are much rarer than others. For example, there will be very few occurrences where a team has lost 7 wickets with 15 overs remaining ($nuw = 4$), yet still manage to bat out the full 20 overs.

Number of innings observed for each state (nuw):

Overs Remaining	W=0	W=1	W=2	W=3	W=4	W=5	W=6	W=7	W=8	W=9
20	0	0	0	0	0	0	0	0	0	0
19	6748	1494	112	5	0	0	0	0	0	0
18	5246	2531	521	59	2	0	0	0	0	0
17	3889	3163	1073	204	26	4	0	0	0	0
16	2801	3368	1648	444	90	5	2	1	0	0
15	1996	3262	2079	785	201	27	5	4	0	0
14	1457	2945	2344	1158	357	78	12	3	4	1
13	1089	2650	2471	1461	502	135	39	5	4	2
12	834	2304	2514	1690	706	221	63	17	5	3
11	594	1904	2494	1939	937	321	116	32	11	6
10	442	1577	2342	2082	1196	456	165	60	20	11
9	302	1310	2111	2197	1429	612	240	96	31	13
8	210	1001	1923	2175	1686	794	346	132	37	27
7	143	776	1605	2139	1852	1018	482	195	72	29
6	96	547	1381	1982	1929	1278	649	278	111	49
5	66	393	1119	1764	1990	1478	820	399	170	77
4	43	274	871	1521	1909	1640	1060	533	264	112
3	26	185	624	1247	1745	1733	1295	758	366	179
2	15	131	425	966	1484	1698	1443	1017	613	261
1	9	84	288	704	1155	1531	1483	1245	863	498
0	6	46	158	458	778	1202	1338	1289	1072	860

We adjusted states' resource values from the original nonparametric resource table using a Bayesian approach. Values with a high nuw were less prone to shifting, while values with low or zero nuw had higher variances.

Analysis and Results of *The Trident*:

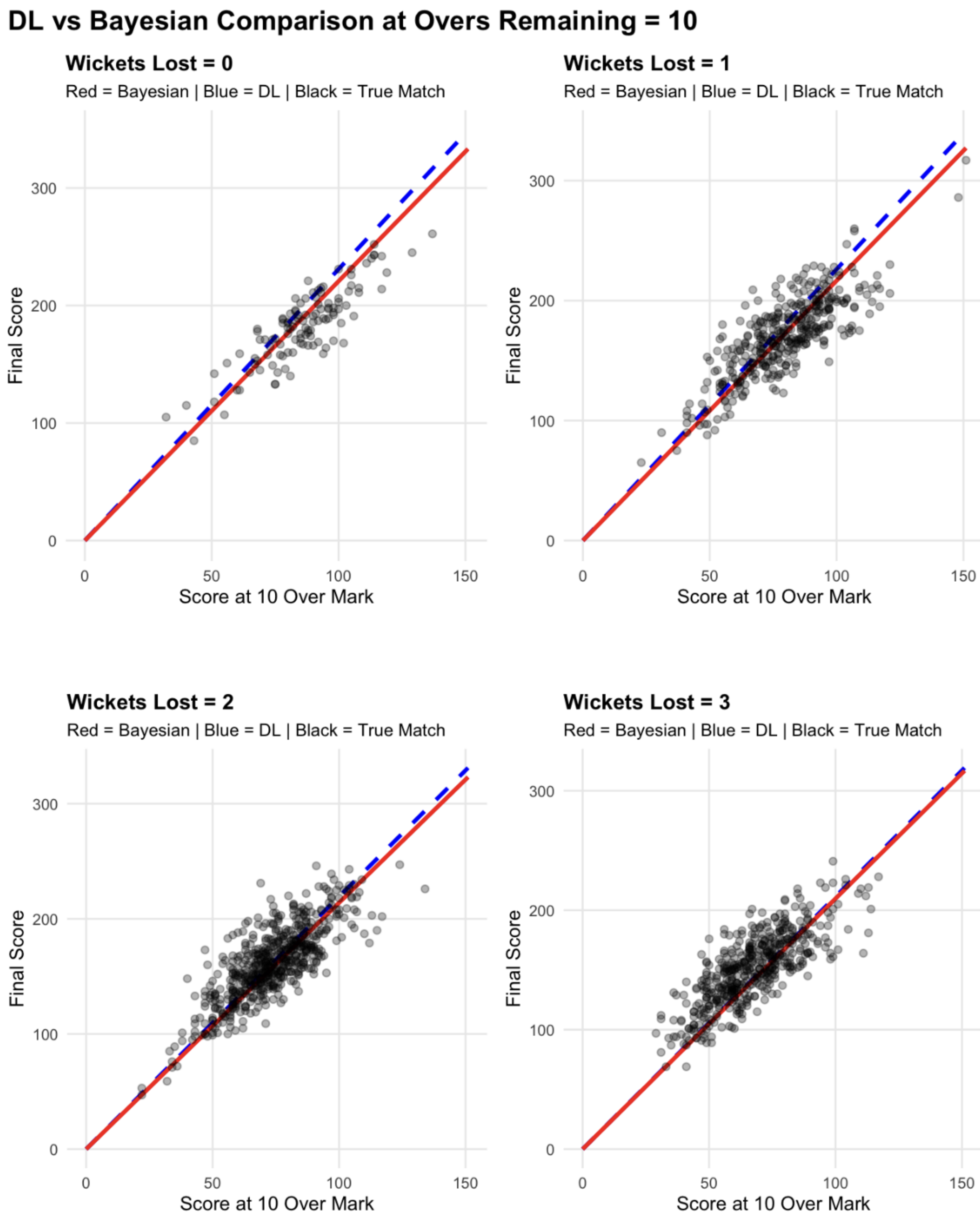
How are we quantifying success?

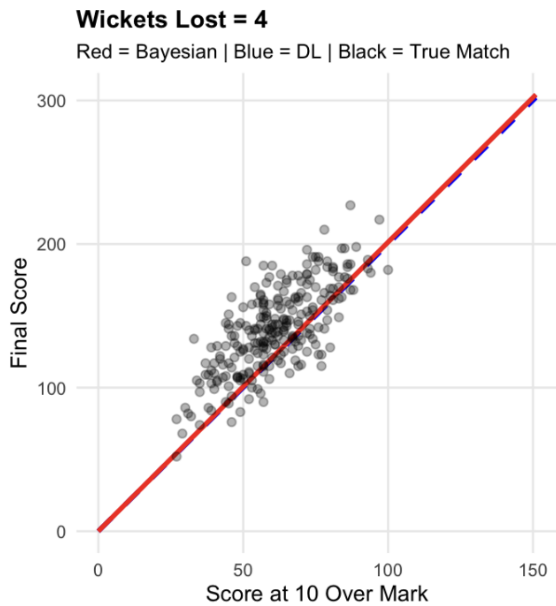
To evaluate the predictive performance of our Bayesian model, we simulate rain-interruption scenarios by truncating the first innings of each match at various overs-remaining marks, denoted u, and given the number of wickets lost w. From this truncation point, we predict the final score using our estimated Bayesian resource table based on the exponential decay model. Specifically, for each match truncated at u overs remaining with w wickets lost, we compare the predicted final score $R^P_{u,w,i}$ to the actual final score $R^A_{u,w,i}$. We quantify predictive accuracy using a residual sum of squares (RSS) metric defined as:

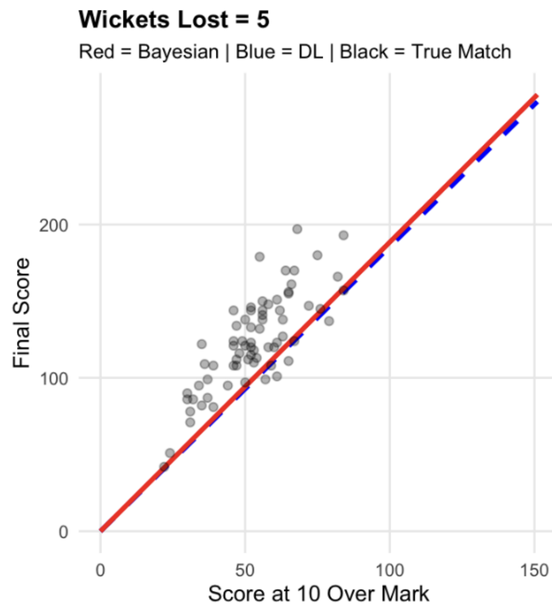
$$\text{RSS}_u = \sum_{w=0}^{9} \sum_{i=1}^{n_{u,w}} \left(R^A_{u,w,i} - R^P_{u,w,i} \right)^2 ,$$

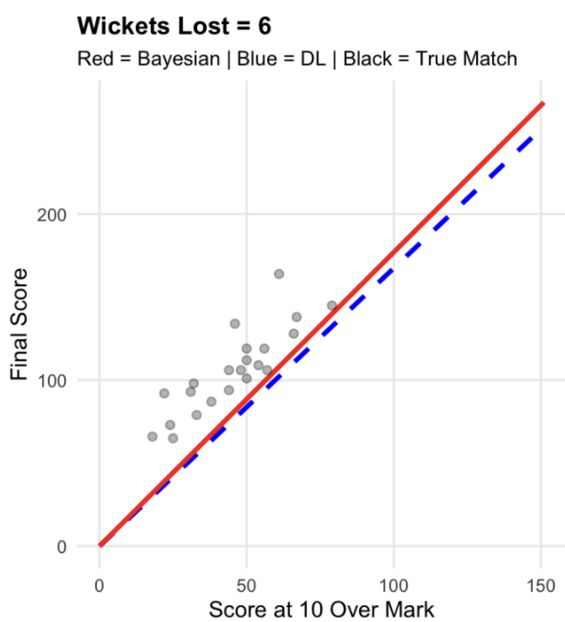
where $n_{u,w}$ is the number of matches observed with w wickets lost and u overs remaining.

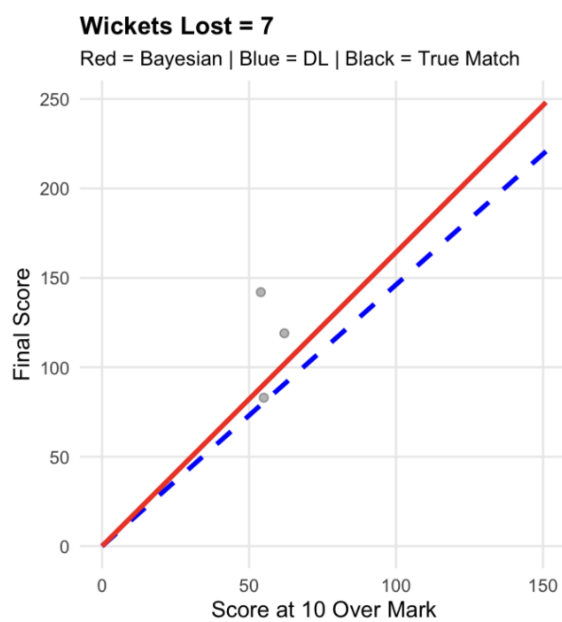
In our analysis, we focus on a variety of cutoffs, such as $u \in \{1, 2, \ldots, 20\}$ overs remaining. For each value of $w \in \{0, 1, \ldots, 9\}$, we generate scatterplots of actual final scores versus scores at truncation. We overlay predictions from both the Bayesian and DL methods. These plots consistently show that the Bayesian model provides more accurate and stable predictions across a wide range of match situations.

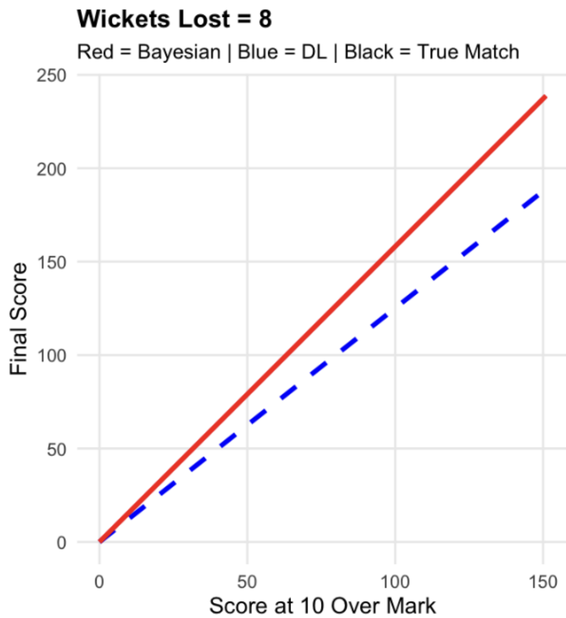


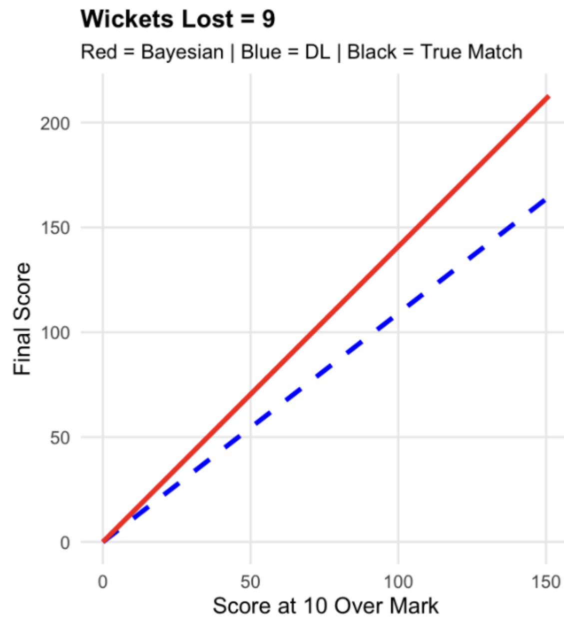

Wickets Lost = 4
Red = Bayesian | Blue = DL | Black = True Match
Final Score
Score at 10 Over Mark
0
100
200
300
0
50
100
150


Wickets Lost = 5
Red = Bayesian | Blue = DL | Black = True Match
Final Score
Score at 10 Over Mark
0
100
200
0
50
100
150


Wickets Lost = 6
Red = Bayesian | Blue = DL | Black = True Match
Final Score
Score at 10 Over Mark
0
100
200
0
50
100
150


Wickets Lost = 7
Red = Bayesian | Blue = DL | Black = True Match
Final Score
Score at 10 Over Mark
0
50
100
150
200
250
0
50
100
150





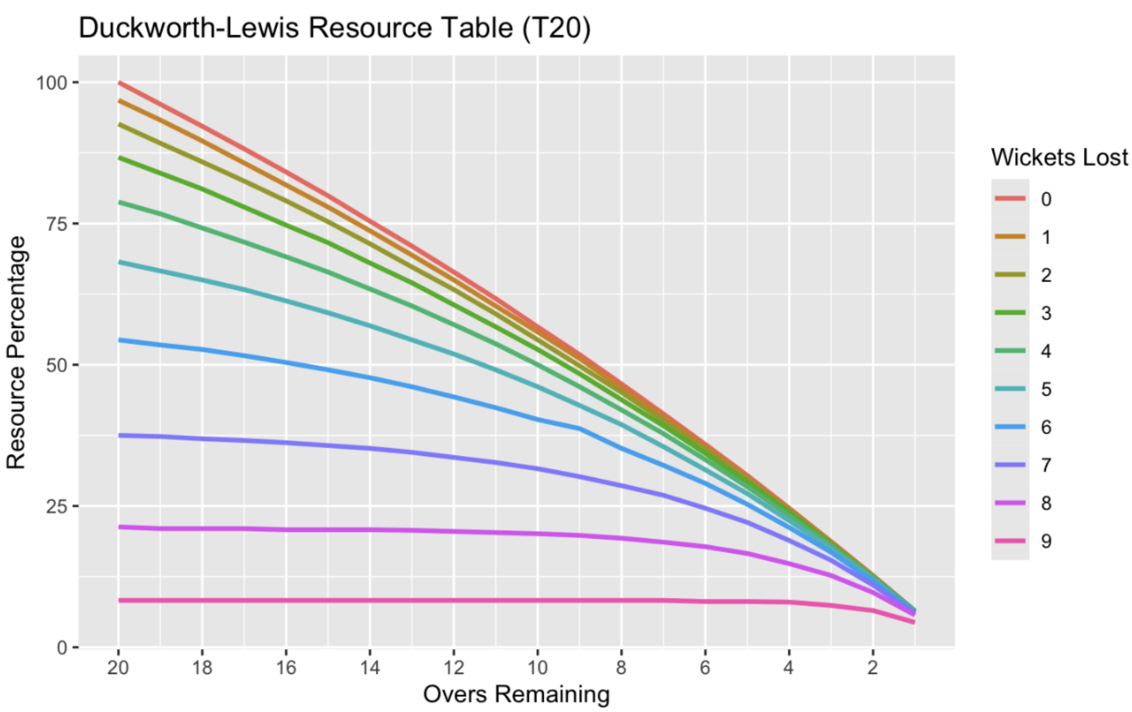
Analysis of graphs above:

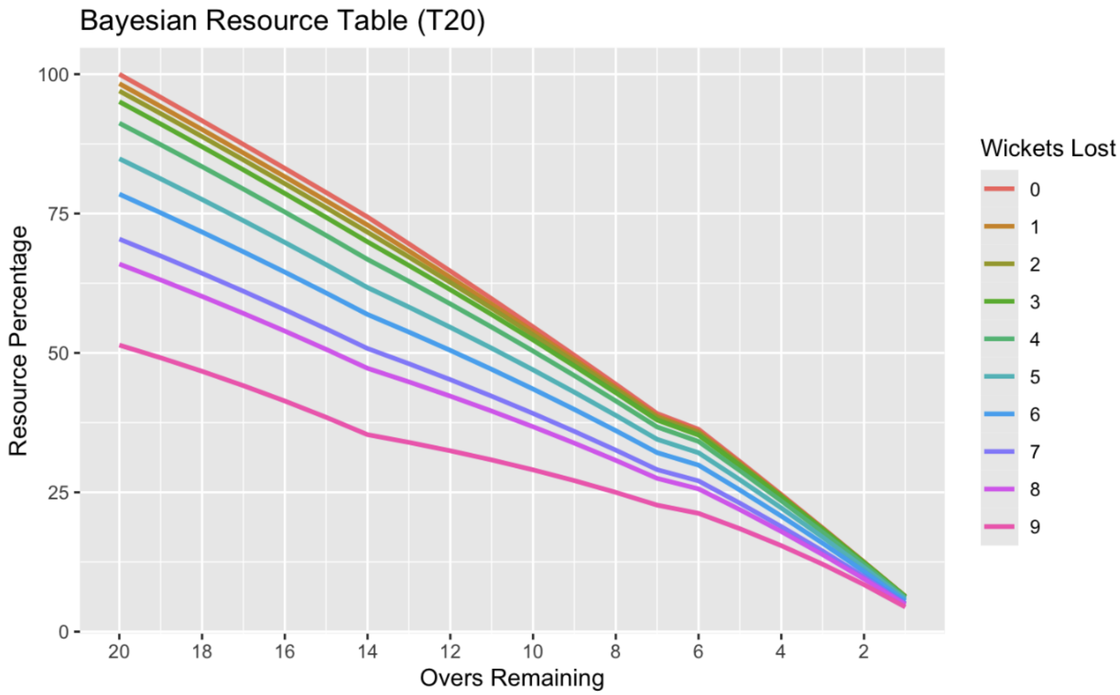
The set of graphs above show us the predicted final score in an innings based on the score 10 overs into the game, for all possibilities of wickets lost, ranging from $w = 0$ to $w = 9$. The black points on the graphs represent true values for innings. We see that for almost all graphs, the Bayes line is a better fit for final scores than the DL line is, and we also see that the Bayes line becomes more distinct from the DL line as the wickets increase, showing that the Bayes model associates a higher percentage of resources remaining despite the loss of wickets.

Below is our final resource table for *The Trident*, and graphics below show how it performs better than the standard DL method.

Bayesian Resource Percentage Table by Overs Remaining (u) and Wickets Lost (w)

u	$w = 0$	$w = 1$	$w = 2$	$w = 3$	$w = 4$	$w = 5$	$w = 6$	$w = 7$	$w = 8$	$w = 9$
20	100.0000	98.3125	96.9942	95.0674	91.2158	84.8389	78.4861	70.4272	65.9454	51.4066
19	95.8486	94.2180	92.9339	91.0673	87.3579	81.2278	75.1334	67.4040	63.0980	49.1306
18	91.6506	90.0706	88.8101	86.9929	83.4164	77.5237	71.6864	64.2853	60.1478	46.7072
17	87.4054	85.8697	84.6427	82.8428	79.3896	73.7244	68.1425	61.0680	57.0910	44.1270
16	83.1127	81.6145	80.3678	78.6157	75.2756	69.8275	64.4989	57.7491	53.9238	41.3797
15	78.7717	77.3044	76.0472	74.3100	71.0725	65.8303	60.7529	54.3254	50.6422	38.4546
14	74.3821	72.9388	71.6590	69.9244	66.7784	61.7303	56.9016	50.7934	47.2421	35.3400
13	69.5675	68.2879	67.2009	65.6869	62.8429	58.2246	53.7407	48.0611	44.8055	33.9545
12	64.6808	63.5568	62.6494	61.3441	58.7934	54.5981	50.4607	45.2130	42.2509	32.4481
11	59.7208	58.7442	58.0027	56.8934	54.6265	50.8468	47.0572	42.2443	39.5725	30.8103
10	54.6865	53.8486	53.2587	52.3322	50.3389	46.9663	43.5255	39.1499	36.7643	29.0296
9	49.5767	48.8685	48.4154	47.6577	45.9271	42.9523	39.8608	35.9245	33.8201	27.0935
8	44.3903	43.8025	43.4708	42.8670	41.3874	38.8000	36.0580	32.5624	30.7333	24.9886
7	39.1261	38.6492	38.4226	37.9574	36.7162	34.5048	32.1120	29.0580	27.4969	22.7000
6	36.3079	35.8749	35.6800	35.2640	34.1270	32.0912	29.8765	27.0489	25.6126	21.2263
5	30.5069	30.1796	30.0743	29.7846	28.8861	27.2386	25.4002	23.0499	21.8910	18.4659
4	24.6081	24.3736	24.3365	24.1520	23.4741	22.1977	20.7337	18.8598	17.9660	15.4332
3	18.6097	18.4549	18.4434	18.3616	17.8854	16.9611	15.8691	14.4696	13.8265	12.1013
2	12.5100	12.4212	12.4118	12.4092	12.1141	11.5213	10.7978	9.8696	9.4607	8.4406
1	6.3073	6.2703	6.2984	6.2903	6.1544	5.8703	5.5112	5.0499	4.8562	4.4187

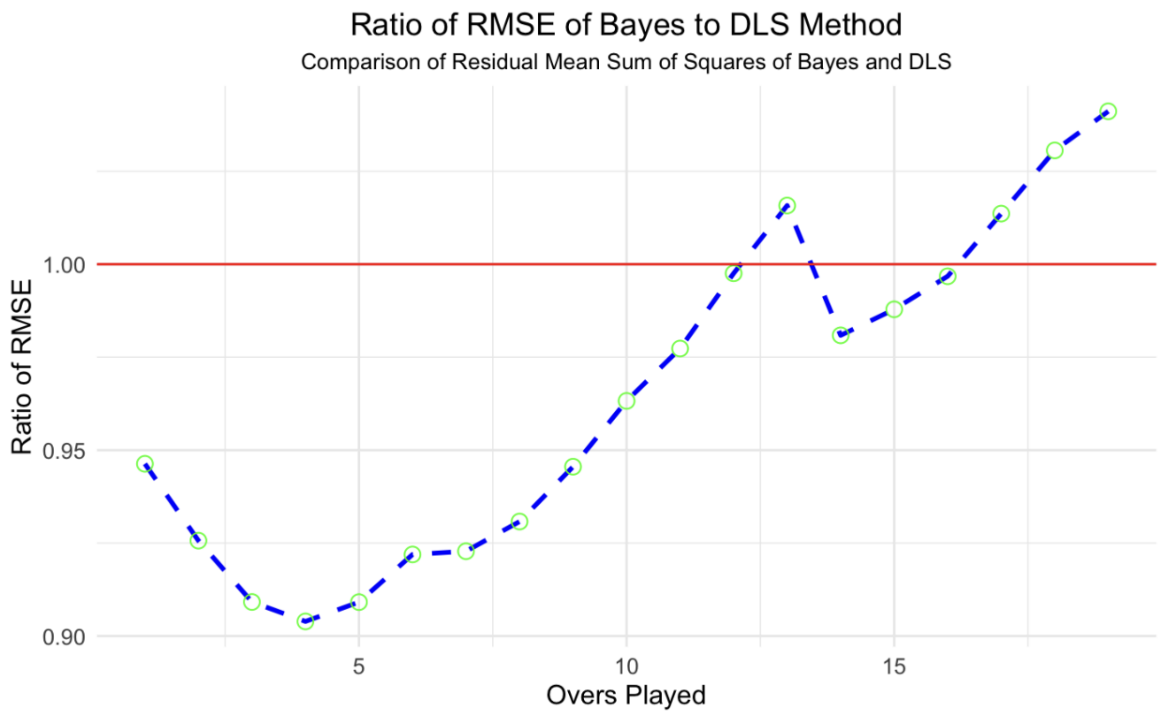




Note that *The Trident* maintains monotonicity and doesn't violate any of our constraints. All lines converge to zero when there are zero overs remaining.

Interestingly, *The Trident* identifies a higher run rate for more wickets lost which supports our intuition that the marginal impact of lost wickets is lower later in the inning.

Furthermore, to summarize model performance across all overs-left scenarios, we compute the ratio of root mean squared error (RMSE) for the Bayesian method to the DL method. As shown in our figures, this ratio is below 1 for 16 out of 20 overs, indicating superior predictive accuracy of the Bayesian approach. Posterior density plots of the RMSE ratios also support the conclusion that the Bayesian model offers a statistically significant improvement in forecast accuracy for match score progression. For states with less overs remaining, *The Trident* performs slightly worse than the DL method. Perhaps the most obvious improvement in accuracy is made in the powerplay phase, showing that splitting the model into 3 phases (powerplay, middle overs, and death) and looking at those phases as essentially 3 individual games has allowed for the model to account for specific trends during the innings and yield higher accuracy. We believe that the absence of the inclusion of the 3 phases was a shortcoming of the Bhattacharya et. al. 2018 model.



Conclusion / Future Work:

While *The Trident* generally outperformed the DL method, there is a lot more that can be done to fairly quantify resource percentages and par scores in rain delayed matches. For example, we do not consider team specific or player specific data. Keeping track of individual and team statistics could help us devise a more complex and robust model that can identify more specific patterns and perhaps step away from the "resource table" approach, as it will include more than just the a_w and b_w parameters.

The Trident performs well for the powerplay and middle overs, but performs slightly worse in the death overs, which leads us to consider tweaking the model in the future to account for the complex late game dynamics.

One interesting use case for this Bayesian model is to predict final scores in the first innings based on how an inning has gone so far. Essentially, the graphs shown above were predicting final inning score based on game status in over number 10. This score prediction ability could be used in team analysis for how they want to structure their innings in terms of aggressiveness. Additionally, building a high accuracy model would open up the possibility to compare the model's prediction to mainstream sportsbook lines, and it would raise the possibility of being more accurate than those lines, which would be a significant achievement.

References:

[1] I. Bhattacharya, R. Ghosal, and S. Ghosh, "A Statistical Exploration of Duckworth-Lewis Method Using Bayesian Inference," *arXiv preprint*, arXiv:1810.00908, Oct. 2018. [Online]. Available: http://arxiv.org/abs/1810.00908v1

[2] T. B. Swartz, "A Statistical Perspective on the Duckworth-Lewis Method for Twenty20 Cricket," Simon Fraser University. [Online]. Available: https://www.sfu.ca/~tswartz/papers/twenty20.pdf

[3] S. Patel, "Understanding the Duckworth-Lewis-Stern Method," *Bruin Sports Analytics*, Jun. 2021. [Online]. Available: https://www.bruinsportsanalytics.com/post/duckworthlewissternmethod

[4] M. Amjad, V. Misra, D. Shah, and D. Shen, "mRSC: Multi-dimensional Robust Synthetic Control," *arXiv preprint*, arXiv:1905.06400, May 2019. [Online]. Available: https://arxiv.org/pdf/1905.06400

[5] "Duckworth–Lewis–Stern method," *Wikipedia*, May 23, 2025. [Online]. Available: https://en.wikipedia.org/wiki/Duckworth%E2%80%93Lewis%E2%80%93Stern_method

[6] "Matches," *Cricsheet*, [Online]. Available: https://cricsheet.org/matches/. [Accessed: May 24, 2025].

[7] R. Bhattacharya, P. S. Gill, and T. B. Swartz, "Duckworth–Lewis and Twenty-Twenty Cricket," *Journal of the Operational Research Society*, vol. 62, pp. 1951–1957, 2011.