

BoxCar Ride-Sharing Simulation Study

Jasper Zhou (s2477464), Chris Liu (s2156635)

February 11, 2026

Brief Summary

This report evaluates BoxCar’s ride-sharing operations in Squareshire using a discrete-event simulation based on empirical rider and driver data. The model describes demand and driver supply through stochastic processes, accounting for distance based matching, trip execution, and related revenues as well as costs. We found that the parameter values given in the project brief do not match the empirical data. Therefore, we modified the parameters to build a baseline that is consistent with data and re-evaluated system performance, service reliability, driver outcomes, and financial results.

The results demonstrate that the configuration based on given parameters places the system near capacity, with longer waiting times, higher cancellations, and fewer completed trips. In contrast, the baseline based on the observed data operates more steadily with improved reliability. Further on, we examined six strategic scenarios (service quality, passenger experience, profitability, affordability, driver outcomes, and system balance). A clear trade-off was observed: configurations that optimised rider outcomes usually require excess driver capacity, which reduces utilisation and hourly earnings. Among the scenarios examined, *System Balance* has the most robust overall compromise, while the *Cost* scenario demonstrates the potential of reducing empty driving to improve profitability without depending primarily on oversupply.

1 Introduction

1.1 Background

BoxCar operates a ride-sharing service in Squareshire, a square region measuring 20×20 miles. The spatial structure of Squareshire is illustrated in Figure 1.

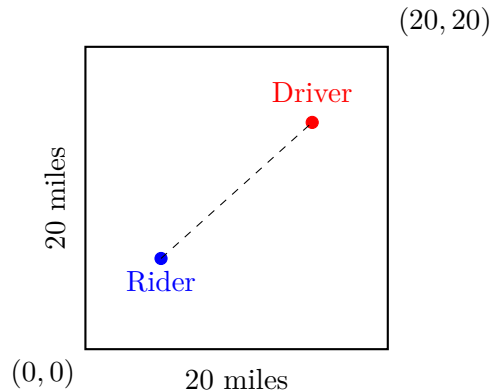


Figure 1: Spatial representation of Squareshire and Euclidean matching distance.

Riders request trips via a mobile application system, while drivers become available randomly in both time and location. Matching is primarily determined by the spatial distance between riders and available drivers at the moment a driver enters the system. Trip duration depends on the distance travelled and the speed of travel.

This system is inherently stochastic, meaning that rider arrivals, driver availability, trip durations, and waiting times are all subject to uncertainty. To capture this complexity, BoxCar has developed an analytical model that characterises each of these components through probability distributions. This model also includes a framework for driver–rider matching, and enables the computation of key performance measures such as service level and driver earnings.

BoxCar has requested us to conduct a quantitative analysis of the system’s current performance, with a focus on service quality experienced by riders and the economic performance for drivers. The main objective is to identify opportunities for improvement, such as enhancing rider satisfaction, increasing driver earnings, and promoting earnings equity while also improving the general efficiency of the system.

Therefore, the goal of this study is to evaluate the current system performance and explore potential directions for improvements. Due to the stochastic nature of the system, we developed a discrete-event simulation to model its operations and assess the impact of changes on key performance indicators.

1.2 Objectives

We aim to develop a discrete-event simulation model using empirical data to accurately represent BoxCar’s ride-sharing operations and provide an evidence-based foundation for decision-making.

First, we design a simulation framework that captures the key stochastic processes of the system, including rider arrivals, driver availability, the matching mechanism, spatial movement, and trip execution. Second, we conduct statistical analysis of available rider and driver records to validate distributional assumptions and modify model parameters accordingly.

Third, we evaluate system performance through key performance indicators for both riders and drivers, including rider waiting time, ride cancellation rate, driver earnings, driver utilisation, and earnings fairness. Finally, we use the simulation results to propose targeted improvements to the system.

1.3 Report Structure

Using observed rider and driver data, we modified the key stochastic inputs and run simulation experiments to give practical recommendations for BoxCar.

First, in section 2 we presented the baseline Discrete-Event Simulation (DES) specifications (state representation, event structure, spatial setting, matching policy, pricing/cost rules, and KPIs). Then, in section 3 we evaluated the independence and distributional assumptions through MLE and goodness-of-fit testing, the results motivated us to modify data-driven parameters to improve the accuracy of the model. In section 4 we first compared the system performance under the given parameters and under the modified parameters, we then examined the differences across rider service quality, driver outcomes, spatial disparities, and platform financial metrics. In section 5 we proposed and assessed improvement strategies through scenario-based optimisation, and translated the findings into practical recommendations. Finally, the limitations of our model are discussed in section 6.

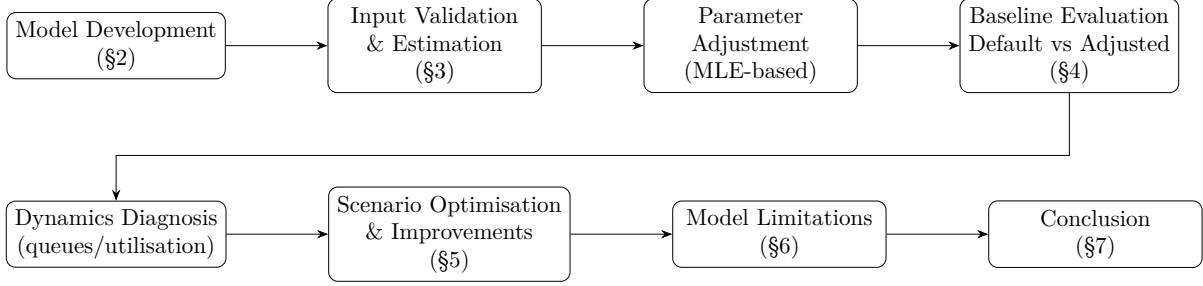


Figure 2: Analytical workflow of the study.

2 Model Development

This section describes the DES model developed to capture the stochastic dynamics in BoxCar’s ride-sharing system. The simulation clock advances from one discrete event to the next and updates the system state when each event occurs. Future events are scheduled during event execution, which allows our system to evolve in an event-driven manner.

The model is implemented using an object-oriented structure, as a DES model is well suited to this approach. The complete code is included in the GitHub repository.

This section outlines the principal components of the model architecture. These include:

- (i) the system environment;
- (ii) the event mechanism governing state changes;
- (iii) the stochastic assumptions for key variables;
- (iv) the spatial representation of the service area.

The model also generates rider–driver matching rules, individual-level revenue and cost calculations, and several key performance indicators (KPIs) used to evaluate system performance.

2.1 System State Representation

At a specific time t , the state of the system is defined by a set of variables characterising the operational configuration of the platform. The state variables remain constant between events in a DES environment and can only be updated when an event is processed. This state representation ensures the consistent scheduling of future events and enables the direct computation of performance measures from simulation outputs.

In addition to the state variables defined for each entity, the model also keeps time-integrated accumulators to compute time-weighted averages of several key measures. In particular, the simulation follows:

$$\int_0^T Q(t)dt, \quad \int_0^T A(t)dt, \quad \int_0^T B(t)dt,$$

where $Q(t)$, $A(t)$, and $B(t)$ denote the numbers of waiting riders, available drivers, and busy drivers, respectively. These integrals are updated at each event transition.

Table 1 summarises the main state variables for the system.

Category	State Variable	Description
Global	Simulation clock (t)	Current simulation time, updated at each event.
	Event calendar	Time-ordered list of future scheduled events.
	Available drivers	Set of drivers currently online and idle.
	Busy drivers	Set of drivers currently serving riders.
	Waiting riders	Set of riders waiting assignment.
	Matched riders	Riders assigned a driver but not yet picked up.
	Cumulative statistics	Aggregated statistics used for KPI computation (e.g., total waiting time, completed rides, cancellations).
Driver	Location (x_d, y_d)	Current spatial coordinates of the driver.
	Online window	Scheduled availability interval.
	Status	Current service status (idle or busy).
	Busy start time	Time at which the current service period begins.
	Earnings	Accumulated fare revenue.
	Total busy time	Cumulative time spent serving riders.
Rider	Origin / Destination	Trip origin and destination coordinates within Squareshire.
	Request time	Time at which the ride request arrives.
	Patience deadline	Cancellation if unmatched long time.
	Assigned driver ID	Identifier of the matched driver (if assigned).
	Status	Waiting, Matched, Completed, or Cancelled.

Table 1: System State Components

Drivers and riders follow well-defined operational state transitions.
The drivers state transition is:

$$\text{Available} \rightarrow \text{Busy} \rightarrow \text{Available} / \text{Offline}.$$

The riders state transition follows:

$$\text{Waiting} \rightarrow \text{Matched} \rightarrow \text{Completed}, \quad \text{or} \quad \text{Waiting} \rightarrow \text{Cancelled}.$$

These transitions determine how the system state changes at event times and how this affects the service performance and driver outcomes during the simulation horizon.

2.2 Event Structure

The system evolves through a sequence of discrete events, shown in Table 2. Each event updates the system state and may generate additional future events which allows the system to continue.

Event Type	Description
Driver Online	A new driver becomes available
Driver Offline	A driver leaves the platform
Rider Arrival	A new ride request enters the system
Rider Cancellation	A rider leaves due to expired patience
Pickup Completion	Driver reaches rider's origin
Drop-off Completion	Rider is delivered to the destination
Termination	Simulation horizon is reached

Table 2: Event Types in the Simulation

All events are maintained in a time-ordered event calendar (implemented as a priority queue). At each iteration, the earliest scheduled event is processed, the simulation clock advances to the event time, and the system state is updated accordingly.

2.3 Stochastic Input Assumptions

Baseline stochastic inputs are specified by BoxCar and implemented directly within the simulation using standard random variate generators (e.g., exponential distributions with a given rate parameter and uniform sampling). The baseline assumptions are summarised in Table 3.

Component	Distribution	Parameter (Rate/Interval)
Driver interarrival time	Exponential	$\lambda_D = 3 \text{ hour}^{-1}$
Rider interarrival time	Exponential	$\lambda_R = 30 \text{ hour}^{-1}$
Rider patience time	Exponential	$\lambda_P = 5 \text{ hour}^{-1}$
Driver online duration	Uniform	$[5, 8] \text{ hours}$
Travel time factor	Uniform	$[0.8, 1.2]$

Table 3: Stochastic Input Distributions (Baseline)

Let d denote Euclidean distance (in miles) and let $v = 20 \text{ mph}$ denote the nominal travel speed. For a travel distance d , the expected travel time (in hours) is given by $\mu(d) = \frac{d}{v}$. To incorporate variability in travel conditions, the actual travel time is modelled as

$$T(d) \sim \text{Uniform}(0.8\mu(d), 1.2\mu(d)).$$

This travel-time model is applied to both (i) the pick-up distance from the driver’s current location to the rider’s origin and (ii) the trip distance from the rider’s origin to the rider’s destination.

2.4 Spatial Representation

Squareshire is represented as the square region $\mathcal{S} = [0, 20] \times [0, 20]$. When a driver comes online, their initial location is sampled uniformly over $\mathcal{S}$. Likewise, each rider’s origin and destination are sampled independently and uniformly over $\mathcal{S}$.

Distances are computed using the Euclidean metric. For two locations (x_1, y_1) and (x_2, y_2) ,

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}.$$

Drivers remain at their current location until completing a ride. Upon drop-off, the driver’s location is updated to the rider’s destination.

2.5 Matching Policy

BoxCar uses a nearest-neighbour matching rule based on Euclidean distance. Matching decisions are executed immediately upon rider arrival, driver availability, or trip completion.

Trigger	Action	Rationale
Rider arrival	Assign closest idle driver	Minimise pick-up distance
Driver online	Assign driver to closest waiting rider	Reduce waiting queue
Drop-off completion	Rematch idle driver to closest waiting rider	Reduce waiting queue

Table 4: Matching Logic

If no suitable match is available at the trigger time, the driver or rider remains in its current state. Riders have a patience limit: a cancellation event is scheduled when they arrive and removes the rider only if they are still unmatched when the limit is reached. Once a rider is assigned to a driver (i.e., moved to the matched state), the patience limit no longer applies, since riders are considered served once a driver is on the way.

2.6 Revenue and Cost Structure

Financial updates are applied when a trip is completed. We use d_{OD} to denote the origin-to-destination distance and d_{pickup} the driver-to-origin distance. The pricing and cost structure is displayed in Table 5 below.

Component	Formula
Base fare	£3
Fare per mile	£2 $\times$ d_{OD}
Operating cost	£0.20 $\times$ ($d_{pickup} + d_{OD}$)

Table 5: Revenue and Cost Model

2.7 Performance Measurement and Key Metrics

System performance is evaluated using operational, financial, and service-quality indicators derived from simulation outputs. Time-averaged measures are computed to reflect sustained system behaviour over the simulation horizon.

These metrics assess rider service quality, driver economic performance, and overall system efficiency, as summarised in Table 6.

Category	Metric	Interpretation
Rider	Completion rate	Proportion of ride requests successfully served
	Cancellation rate	Proportion of riders leaving due to delay
	Mean waiting time (min)	Average rider waiting time
	Waiting time percentiles	75th, 90th, 95th percentile wait times
	Waiting time std. deviation	Variability in rider waiting time
System	Avg waiting riders	Mean number of riders in queue
	Avg busy drivers	Mean number of drivers serving rides
	System utilization	Fraction of total driver-hours spent busy
Driver	Earnings per ride	Revenue efficiency
	Earnings per hour	Income sustainability
	Busy-time utilization	Work intensity
	Earnings CV ¹	Earnings fairness across drivers
	Avg earnings per driver	Mean total earnings per driver
	Avg earnings per hour	Earnings normalised by online duration
	Min/Max earnings	Earnings dispersion indicators
Financial	Total revenue	Aggregate driver fare income
	Total operating cost	Aggregate driver operating cost
	Net revenue ²	Revenue minus operating cost

Table 6: Key Performance Indicators

In addition to global metrics, the riders across Squareshire are divided into three equally sized concentric zones, with the mid-point of Squareshire as the center of these zones. The region-level key performance indicators (KPI) are used to assign numerical values to the number of requests for service, percentage of completion, percentage of cancellations, and the average wait time. The examination of these items helps determine any spatial differences in the performance of the service.

The preceding sections introduced the baseline structural and stochastic assumptions of the model. In the following sections, these assumptions are evaluated against empirical rider and driver data, and parameters are modified where necessary.

3 Statistical Analysis and Validation of Input Data

The statistical properties of the model inputs are evaluated in this section. The three observable operational variables are driver inter-arrival times, driver online durations, and rider inter-arrival times. These variables are directly observable and therefore can be tested for independence and distribution.

However, rider patience time cannot be empirically validated since it is a latent variable and cannot be directly observed. As such, only the realised waiting time of matched riders is available, which provides right-censored observations of total patience. For cancelled riders, the exact cancellation timestamp is not recorded, so the actual patience duration cannot be recovered. Consequently, the assumed exponential distribution of rider patience cannot be statistically tested due to the lack of observable data.

3.1 Independence Analysis of Operational Variables

Independence is a key assumption of both exponential and uniform models. Therefore, we assessed the temporal independence of three major stochastic variables: driver inter-arrival times, driver online durations, and rider inter-arrival times. We used the asymptotic normal approximation to compute and evaluate the lag-1 sample autocorrelation for each variable.

Variable	Lag-1 ACF ³	Test Statistic	p-value ⁴	Conclusion
Driver Inter-arrival Time	0.0132	$z = 0.9106$	0.3625	Independent
Driver Online Duration	0.0099	$z = 0.6791$	0.4971	Independent
Rider Inter-arrival Time	-0.0067	$z = -1.2444$	0.2134	Independent

Table 7: Independence Tests for Operational Variables

Across all variables, the null hypothesis of independence cannot be rejected at the 5% significance level. Therefore, the empirical data is consistent with the assumption of temporal independence required for subsequent distributional modelling.

3.2 Distributional Validation Under Unknown Parameters

Since the independence assumption is not rejected, we next examine whether the empirical data are consistent with the proposed distributional families when parameters are treated as unknown.

¹CV (Coefficient of Variation) equals the standard deviation divided by the mean. A higher CV indicates greater inequality in driver earnings distribution.

²In this report, the “net revenue” refers to total passenger fare payments minus driver operating costs (fuel/vehicle cost proxy), and does not model any platform commission.

³Lag-1 ACF represents the sample autocorrelation at lag 1.

⁴p-values are tested at the 5% significance level. A p-value above 0.05 means there is no strong evidence against independence.

We apply maximum likelihood estimation (MLE) to estimate the model parameters. For exponential distributions, the rate parameter is estimated as

$$\hat{\lambda} = \frac{1}{\bar{x}},$$

where $\bar{x}$ denotes the sample mean of inter-arrival times. For the uniform distribution,

$$\hat{a} = \min(x), \quad \hat{b} = \max(x).$$

The resulting parameter estimates are summarised below.

Variable	Estimated Parameter(s)	KS Statistic	p-value	Distribution
Driver Inter-arrival	$\hat{\lambda}_D = 4.742$	$D = 0.0104$	0.4684	Exponential
Driver Online Duration	$[\hat{a}, \hat{b}] = [6.0007, 7.9998]$	$D = 0.00985$	0.7402	Uniform
Rider Inter-arrival	$\hat{\lambda}_R = 34.596$	$D = 0.00234$	0.9814	Exponential

Table 8: Maximum Likelihood Parameter Estimates

We performed Kolmogorov–Smirnov tests with Monte Carlo adjustment⁵, and the results provide no statistical evidence against the assumed distributional families.

Therefore, the proposed distributional assumptions remain consistent with the empirical data when the parameters are estimated from the empirical data.

3.3 Consistency of Parameters Given in the Project Brief

Next we examine whether the parameter values given in the project brief are consistent with the empirical data. The assumed parameters are:

$$\lambda_D = 3, \quad \lambda_R = 30, \quad \text{Online Duration} \sim U(5, 8).$$

Goodness-of-fit tests were conducted under these fixed parameter values. The Kolmogorov–Smirnov (KS) test and the chi-square goodness-of-fit test were applied where appropriate. The results are summarised below.

Variable	Test ⁶	p-value	Conclusion
Driver Inter-arrival	KS / χ^2	< 0.05	Reject specified λ_D
Driver Online Duration	KS / χ^2	< 0.05	Reject $U(5, 8)$
Rider Inter-arrival	KS / χ^2	< 0.05	Reject specified λ_R

Table 9: Goodness-of-Fit Tests Under Project Parameters

Across all three variables, the null hypothesis of consistency with the project specified parameter values is rejected at the 5% significance level. This suggests that while the assumed stochastic structures are supported by the empirical data, the parameter values given in the project brief do not match the empirical data. The model therefore needs to be modified so that the simulation better reflects the observed system behaviour.

⁵The Monte Carlo adjustment takes parameter estimation into account when computing the KS statistic.

⁶KS denotes the Kolmogorov–Smirnov test; χ^2 denotes the chi-square goodness-of-fit test.

3.4 Parameter Re-estimation and Model Modification

Since the parameter values given in the project brief are inconsistent with the empirical data, we re-estimated the parameters using maximum likelihood estimation (MLE). The comparison between the given parameters and the estimated parameters is shown below.

Variable	Estimated Parameter(s)	Given Parameter(s)	Deviation
Driver Inter-arrival	$\hat{\lambda}_D = 4.742$	$\lambda_D = 3$	+58%
Driver Online Duration	[6.0007, 7.9998]	[5, 8]	Right-shifted (narrower)
Rider Inter-arrival	$\hat{\lambda}_R = 34.596$	$\lambda_R = 30$	+15%

Table 10: Modified Parameter Estimates

From this comparison, we observe that the system operates with higher arrival intensities than originally specified. In particular, the driver arrival rate is substantially higher than assumed, while rider arrival rate is moderately higher. Driver online durations are largely concentrated toward the upper end of the originally assumed interval.

Overall, the stochastic structure of the model is supported by the empirical data, but the parameter values require adjustment so that the simulation better reflects the observed system behaviour.

In next section, we compared the simulation outcomes using two parameter settings: the given parameters in project brief and the re-estimated parameters.

4 Baseline Performance Evaluation and Output Analysis

This section compares simulation outcomes under two parameter settings: (i) the given parameters specified in the project brief, and (ii) the re-estimated parameters based on maximum likelihood estimates derived in Section 3.

We have seen from Section 3 that the given parameter values are statistically inconsistent with the data. Therefore, it is necessary for us to check how this difference would affect system performance in terms of operations, spatial patterns, and financial outcomes.

4.1 Overall System Performance

The modified (MLE-based) baseline produces consistent performance across the ten independent simulation replications (i.e., low levels of inter-trial operational variability). The mean numbers of busy and idle drivers are 25.56 and 9.01, respectively, yielding a system utilization of 73.95%. The 94.77% completion rate and a 5.19% cancellation rate indicate that there is no observable pressure of supply/demand congestion on the operation of the system.

In contrast, the default configuration as described in the Project Roadmap operates very differently from the system configured using the MLE estimates. The utilization level of the system has grown to 94.24%. During each of the trials to date, the average number of idle drivers has decreased to 1.20. The completion rate has decreased to 72.91%, while the cancellation rate has risen to 27.05%.

The difference in performance between the two configurations illustrates how the default configuration has forced the system into a near-capacity saturated state, resulting in lower service level of performance. In contrast, the MLE modified baseline has provided sufficient spare capacity, thereby enhancing the reliability of the provided service. Thus, the importance of parameter estimation based on actual usage data cannot be overstated.

Metric	Mean $\pm$ Std (Re-estimated)	Mean $\pm$ Std (Default)	Unit
Busy Drivers	25.56 $\pm$ 0.30	19.55 $\pm$ 0.45	drivers
Idle Drivers	9.01 $\pm$ 0.77	1.20 $\pm$ 0.12	drivers
System Utilization	73.95 $\pm$ 1.85	94.24 $\pm$ 0.50	%
Completion Rate	94.77 $\pm$ 0.62	72.91 $\pm$ 1.61	%
Cancellation Rate	5.19 $\pm$ 0.62	27.05 $\pm$ 1.61	%

Table 11: Comparison of Overall System Performance under Re-estimated and Default Parameter Sets (10 Independent Replications)

4.2 Rider Service Quality

With the re-estimated parameters, the average wait is 15.87 minutes with a median wait of 12.26 minutes. This indicates a reasonable level of responsiveness. Looking further into the upper tail of the distribution gives a similar picture: the 95th percentile wait is around 42.17 minutes, and the maximum waiting time remains below 90 minutes on average.

In contrast, the default set of operational parameters resulted in an increase in average wait time to 24.97 minutes and median wait time to 22.23 minutes. This indicates that the entire range of distribution was shifted upward. This increase also holds true for the tail of the distribution with 95th percentiles being greater than 53 minutes and average maximum waits over 100 minutes.

It appears that when the system operates at close to capacity, with the default operational parameters set, customers experience longer queue times and are subjected to much more frequent and extreme delays than when using the newly re-estimated set of parameters (for example, the new average wait time is 15.87 compared to the original average wait time of 24.97). In summary, using the re-estimated set of parameters will likely result in shorter total wait times for customers and lower probabilities of having very long wait times thus enhancing overall service reliability.

Metric	Mean $\pm$ Std (Re-estimated)	Mean $\pm$ Std (Default)	Unit
Mean Waiting Time	15.87 $\pm$ 0.63	24.97 $\pm$ 0.18	minutes
Median Waiting Time	12.26 $\pm$ 0.63	22.23 $\pm$ 0.19	minutes
90th Percentile	33.61 $\pm$ 1.17	45.96 $\pm$ 0.38	minutes
95th Percentile	42.17 $\pm$ 1.13	53.13 $\pm$ 0.50	minutes
Maximum Waiting Time	86.50 $\pm$ 3.29	104.43 $\pm$ 11.89	minutes

Table 12: Comparison of Rider Waiting Time Distribution under Re-estimated and Default Parameter Sets (10 Independent Replications)

4.3 Driver Economic Outcomes and Capacity Conditions

Driver earnings and system capacity conditions are also examined under the two parameter configurations.

In the modified model, drivers earn about £165.31 on average over the simulation horizon, or roughly £22.57 per active hour. Busy-time utilisation is around 73.95%, indicating that the system remains productive but still has some spare capacity. Earnings dispersion is non-trivial ($CV = 0.218$), and minimum earnings remain zero, suggesting uneven matching outcomes for a subset of drivers. Average waiting riders remain low (0.36), with no indication of structural congestion.

Under the default parameter configuration, driver earnings increase to £174.00 on average, and hourly earnings also rise to £24.98. Utilisation increases to 94.24%, while earnings variability drops ($CV = 0.190$), meaning that the workload is more evenly distributed among drivers. However, the average number of waiting riders rises to 1.62, suggesting heavier queue pressure.

From these results we see a clear trade-off: the default configuration improves driver income and utilisation, but it also reduces service performance and queue stability. In contrast, the modified model keeps the system less tight, creating a better balance between driver earnings and demand conditions.

Metric	Mean $\pm$ Std (Re-estimated)	Mean $\pm$ Std (Default)	Unit
Average Earnings per Driver	165.31 $\pm$ 2.16	174.00 $\pm$ 1.27	£
Average Earnings per Active Hour	22.57 $\pm$ 0.28	24.98 $\pm$ 0.15	£/hour
Busy-Time Utilization	73.95 $\pm$ 1.85	94.24 $\pm$ 0.50	%
Earnings Coefficient of Variation	0.218 $\pm$ 0.007	0.190 $\pm$ 0.003	–
Minimum Earnings	0.00 $\pm$ 0.00	0.00 $\pm$ 0.00	£
Average Waiting Riders	0.36 $\pm$ 0.04	1.62 $\pm$ 0.11	riders

Table 13: Comparison of Driver Economic and Capacity Metrics under re-estimated and Default Parameter Sets (10 Independent Replications)

4.4 Spatial Performance Disparities

We next examine how system performance varies across regions.

The adjusted parameter set shows only moderate spatial differences in performance across regions. Average waiting time rises gradually from 14.34 minutes in the centre to 17.21 minutes in the outer region, while cancellation rates increase from 4.41% to 5.77%. Although peripheral areas perform less well, the difference remains relatively small.

By contrast, the default parameter values lead to clear declines in both service quality and spatial balance. Waiting times increase across all regions, reaching 27.56 minutes in the outer area, while cancellation rates exceed 23% in the centre and approach 30% in the outer region. The difference between the centre, middle, and outer regions becomes larger, indicating that congestion increases geographic inequality.

Region	Waiting (min)	Cancellation (%)	Source
Center	14.34 $\pm$ 0.57	4.41 $\pm$ 0.50	Re-estimated
	21.97 $\pm$ 0.20	23.35 $\pm$ 1.53	Default
Middle	14.97 $\pm$ 0.56	4.89 $\pm$ 0.66	Re-estimated
	23.59 $\pm$ 0.19	25.79 $\pm$ 1.84	Default
Outer	17.21 $\pm$ 0.72	5.77 $\pm$ 0.72	Re-estimated
	27.56 $\pm$ 0.30	29.73 $\pm$ 1.60	Default

Table 14: Comparison of Regional Service Performance under Re-estimated and Default Parameter Sets (10 Independent Replications)

4.5 Platform Financial Performance

The financial outcomes of the two parameter configurations are compared below.

Total fare revenue reaches £563,263 under the adjusted parameter set, with operating costs of £73,566 and net revenue of £489,698. The resulting cost-to-revenue ratio of 13.06% indicates strong financial efficiency under balanced system conditions.

In contrast, the default parameter values produce lower fare revenue (£373,846) and net revenue (£317,570). Although operating costs decrease in absolute terms, the cost-to-revenue ratio rises to 15.05%, indicating weaker financial performance.

The weaker financial performance under the default parameter values is consistent with higher cancellation rates and fewer completed trips. While higher driver utilisation increases individual earnings, near-capacity congestion reduces total transaction volume, ultimately lowering the total net revenue generated in the simulated system.

Metric	Mean $\pm$ Std (Re-estimated)	Mean $\pm$ Std (Default)	Unit
Total Fare Revenue	563,263 $\pm$ 3,748	373,846 $\pm$ 8,112	£
Total Operating Costs	73,566 $\pm$ 858	56,276 $\pm$ 1,286	£
Net Revenue	489,698 $\pm$ 3,732	317,570 $\pm$ 6,837	£
Cost-to-Revenue Ratio	13.06 $\pm$ 0.17	15.05	%

Table 15: Comparison of Platform Financial Performance under Re-estimated and Default Parameter Sets (10 Independent Replications)

5 Improvement Strategies and Scenario Analysis

Based on the empirical results presented above, we further examine how the system state changes over time. By observing the main operational variables, we identify a recurring pattern between temporary overload and temporary under-utilisation.

As illustrated in Figures 3 and 4, the number of busy drivers (green curve) rises and falls repeatedly over time. These changes are accompanied by corresponding movements in available drivers (orange curve), which increase during slack periods and decrease when demand becomes stronger. Meanwhile, waiting riders (blue curve) increase when driver capacity becomes limited and decrease when more drivers become available.

This pattern reflects a feedback adjustment process: temporary demand surges lead to rider accumulation and higher driver occupancy, followed by supply relaxation once trips are completed. Instead of moving smoothly toward a steady equilibrium, the system repeatedly shifts between periods of demand pressure and supply slack.

Such recurring imbalance leads to higher cancellation risk, lower driver utilisation efficiency, and reduced overall revenue. To address this instability and identify a more stable configuration, we implement a constrained multi-objective optimisation framework.

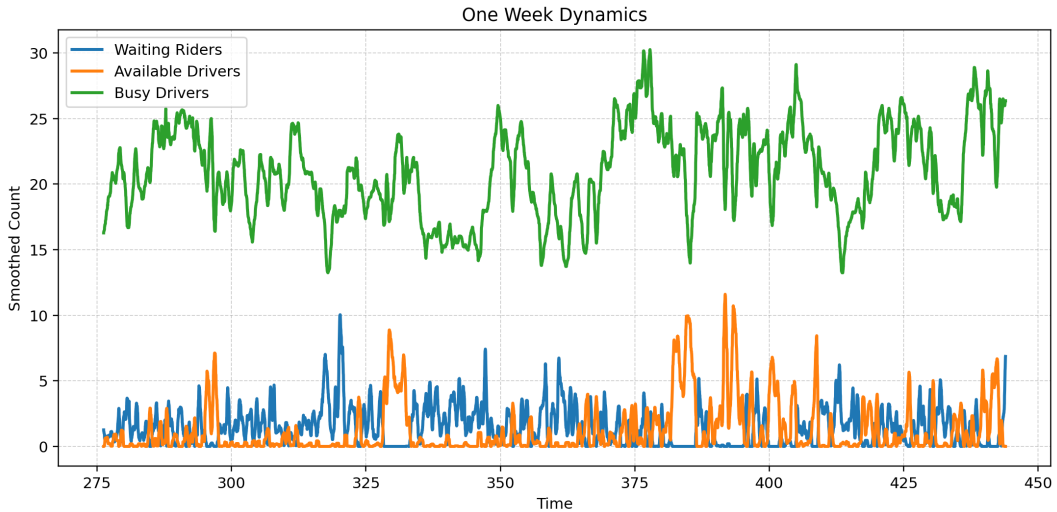


Figure 3: One-week system dynamics (smoothed counts).

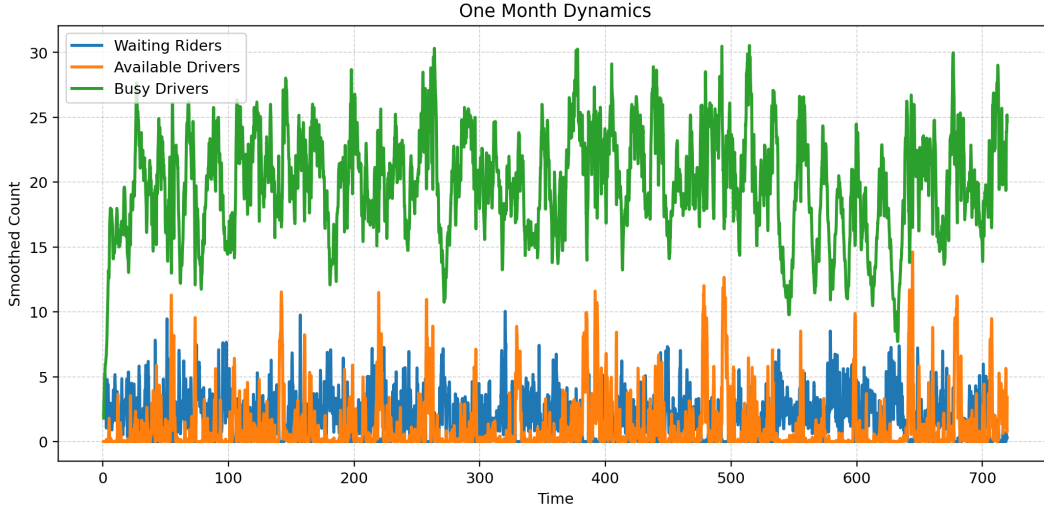


Figure 4: One-month system dynamics (smoothed counts).

5.1 Scenario-Based Optimisation Framework

Optimisation Scenarios

To improve system performance and reduce dynamic imbalance, we adopt a constrained multi-objective optimisation framework. Six optimisation scenarios are defined to represent different operational priorities and to examine trade-offs across key performance dimensions.

Scenario	Optimisation Emphasis (Key KPIs)
Passenger Experience	↓ Avg Wait, ↓ P90 Wait, ↓ Cancellation, ↑ Completion, ↓ Passenger Cost
System Profit	↑ Net Revenue, while maintaining service quality (Wait, Cancellation, Completion)
Passenger Cost	↓ Passenger Cost, with constraints on Wait, Cancellation, Completion
Driver Optimal	↑ Driver Earnings (avg, hourly), ↑ Rides per Driver, bounded Online Hours
System Balance	Weighted trade-off among Revenue, Service Quality, Utilisation, Cost
Quality Priority	↓ Cancellation, ↓ Avg Wait, ↓ P90 Wait, ↓ Queue Length, ↑ Utilisation

Table 16: Scenario design and KPI optimisation emphasis (concise version; see Table 20 for detailed scoring).

The six scenarios represent distinct optimisation priorities that reflect different perspectives within the platform system.

The *Passenger Experience* and *Quality Priority* scenarios emphasise service performance, with particular focus on reducing average and upper-tail waiting times, lowering cancellation rates, and maintaining high completion levels. In contrast, the *System Profit* scenario prioritises platform-level financial performance while ensuring that minimum service standards are respected. The *Passenger Cost* scenario shifts the emphasis toward affordability, seeking to reduce rider expenditure without materially degrading service quality.

From the supply side, the *Driver Optimal* scenario focuses on improving driver earnings and ride allocation efficiency under bounded working hours. Finally, the *System Balance* scenario aims to achieve a structured compromise across profitability, service quality, utilisation, and cost considerations.

Together, these scenarios provide a systematic framework for examining trade-offs between efficiency, equity, service reliability, and long-term system stability.

Search Space Design

To operationalise the optimisation framework, we define a structured search space over the key control parameters of the model.

Parameter	Base Variable	Range	Step
Driver rate multiplier	Driver interarrival rate	[0.8, 1.5]	0.05
Driver online multiplier	Driver minimum/maximum online hours	[0.8, 1.5]	0.05
Base fare multiplier	Initial trip fare	[0.8, 1.2]	0.05
Cost per mile multiplier	Operating Cost per Mile	[0.8, 1.2]	0.05

Table 17: Grid search parameter configuration (approximately 18,225 combinations).

Each parameter in Table 17 is implemented as a multiplicative scaling factor applied to its corresponding baseline value. The *Driver rate multiplier* changes the baseline driver interarrival rate, thereby modifying the effective supply inflow into the system. Similarly, the *Driver online multiplier* scales the minimum and maximum driver online hours, while the pricing-related multipliers adjust the initial trip fare and the operating cost per mile. This multiplicative design preserves the structure of the re-estimated baseline model while allowing consistent changes in supply and pricing levels.

The parameter ranges are chosen to allow meaningful but reasonable deviations from the real-data baseline. Driver availability parameters are permitted a wider range so that both driver shortages and oversupply situations can be examined. In contrast, pricing parameters are confined to a narrower interval to avoid unrealistic fare levels.

Each parameter combination is simulated over a 168-hour (one-week) horizon and evaluated in parallel using `ProcessPoolExecutor`. The grid step size of 0.05 provides sufficient detail to compare alternative configurations while keeping the total computational cost manageable.

Feasibility Constraints

To ensure practical viability and avoid extreme but unrealistic configurations, a set of hard feasibility constraints is imposed prior to scenario scoring.

Constraint	Threshold	Purpose
Cancellation Rate	≤ 0.06	Control passenger attrition
Average Wait	≤ 30 min	Limit overall service delay
P90 Wait	≤ 50 min	Control tail risk
Utilisation	≥ 0.20	Avoid severe driver idleness
Completion Rate	≥ 0.70	Ensure platform usability
Avg Online Hours	≤ 10	Maintain sustainability/fairness

Table 18: Hard feasibility constraints applied before scoring.

Only parameter combinations satisfying all constraints are retained for subsequent evaluation. These thresholds jointly protect service reliability, operational efficiency, and driver sustainability, thereby preventing optimisation outcomes that improve one objective at the expense of overall system stability.

5.2 Optimisation Mechanism

The optimisation method can be viewed as a constraint-based weighted multi-objective scoring model. It combines feasibility filtering with the structured evaluation of feasible configurations based on specific KPIs.

First, all parameter combinations are tested against the hard feasibility constraints shown in Table 18. Only configurations that satisfy all constraints are retained for further evaluation. This step ensures that impractical solutions are excluded before the optimisation stage.

Second, because the KPIs are measured in different units (minutes, ratios, currency, hours). In order to achieve it, all indicators are normalised to the $[0, 1]$ interval using min-max scaling and for a generic metric X , the normalised value is defined as

$$X_{\text{norm}} = \frac{X - X_{\min}}{X_{\max} - X_{\min}}.$$

This transformation removes scale differences across indicators and ensures that no single KPI dominates the objective function simply because of its magnitude. As a result, by mapping all metrics to the same range, the weighted aggregation reflects strategic priorities rather than differences in measurement units.

To construct a unified objective function, all KPIs are oriented so that larger values correspond to better system performance. Specifically, the benefit-type metrics use X_{norm} directly, while cost-type metrics are transformed as $1 - X_{\text{norm}}$ which converting them into benefit-type indicators and ensuring a consistent optimisation.

Finally, each KPI is assigned a scenario-specific weight, and the weighted scores are summed to obtain an overall performance score. The parameter configuration with the highest score is selected as the optimal solution for that scenario.

This framework allows service quality, profitability, efficiency, and driver incentives to be evaluated together, while feasibility constraints ensure that all selected solutions remain practically viable.

5.3 Key Findings, Trade-offs, and Recommendations

As summarised in Table 19, the optimisation results reveal a clear relationship between passenger experience and driver outcomes. The corresponding optimised parameter configurations for each scenario are reported in Table B, which details the multiplier settings and the associated key performance indicators.

Taken together, these results show that as supply shifts from shortage to excess, system performance changes in predictable ways: lower driver supply benefits drivers but disadvantages passengers, while excess supply improves service quality but reduces efficiency. This trade-off forms the basis of the following analysis.

Analysis of the Passenger-Driver Trade-off Across Supply Regimes

(i) Tight Supply: Good for Drivers, Poor for Passengers Under the default configuration, the system operates close to full capacity, with driver utilisation reaching approximately 0.94. Drivers are almost continuously occupied, leading to higher hourly earnings and relatively even income distribution. However, passenger performance worsens sharply: average waiting time rises to around 25 minutes, and more than 27

This pattern arises from a straightforward mechanism:

- High utilisation $\Rightarrow$ Limited spare capacity
- Limited capacity $\Rightarrow$ Longer waits
- Longer waits $\Rightarrow$ Higher cancellations

In contrast, the real-data baseline reduces utilisation to approximately 0.72074, significantly improving passenger outcomes (waiting time $\approx$ 1516 minutes; cancellation $\approx$ 5%). This baseline represents a more stable operating point.

(ii) Excess Supply: Excellent Passenger Service, Lower Driver Efficiency Several optimised scenarios operate with a large number of available drivers (4346) and low utilisation (around 0.32033). Passenger outcomes become nearly ideal: cancellation below 0.3%, waiting time around 6 minutes, and completion close to 0.997.

However, this better passenger performance is achieved through redundancy:

- Lower waiting time $\Rightarrow$ Higher supply redundancy
- Higher redundancy $\Rightarrow$ Lower utilisation
- Lower utilisation $\Rightarrow$ Lower hourly earnings and higher inequality

Hourly earnings decline to approximately 1213, and earnings dispersion increases substantially ($CV \approx 0.41$).

Across scenarios, a clear trade-off emerges between passenger experience and driver outcomes. Increasing supply improves waiting times and reduces cancellations, but excessive supply lowers utilisation and weakens driver earnings. Conversely, tightening supply strengthens driver income but results in longer waits and higher cancellation rates. The real-data baseline lies between these extremely values and represents a more balanced operating point.

Improvement Directions and Two Operational Methods

The trade-offs identified above indicate that sustainable improvement cannot rely on shifting the system to either extreme. Instead, performance gains should be pursued through operational refinements within the balanced operating state.

Following this principle, we first summarise four strategic improvement directions derived from the optimisation results. After that we translate these directions into two concrete and implementable operational methods designed to stabilise service quality, support driver sustainability, and improve overall efficiency.

(i) Improve Matching Efficiency Since profitability differences are mostly cost-driven, reducing empty driving distance and improving spatial matching should be prioritised over expanding supply.

(ii) Preserve Driver Earnings Stability Improvement strategies should maintain a reasonable earnings floor and limit excessive earnings dispersion to sustain long-term driver participation.

(iii) Avoid Structural Oversupply Persistent oversupply depresses utilisation and is unlikely to remain stable. Dynamic adjustment is preferable to maintaining excess idle capacity.

(iv) Operate Within the Balanced Region Evidence suggests that moderate rule adjustments can improve driver-side outcomes without materially harming passenger indicators, provided the system remains near the balanced operating state.

To translate these directions into implementable actions, we propose two refined, evidence-based operational methods. Both are based on the optimisation results and are designed to keep the system away from the two extremes identified earlier—default-like shortage and permanent redundancy.

Method 1: Mode-switching policy using optimised “reference settings” (dynamic supply and pricing control). Let (m_D, m_H, m_F, m_C) denote multiplicative factors applied to the driver arrival rate, driver online-duration bounds, base fare, and cost per mile, respectively. The optimisation produces the following reference settings:

Service-oriented setting: $(m_D, m_H, m_F, m_C) = (1.45, 1.35, 1.10, 1.20)$,

which achieves very strong passenger KPIs (cancel $\approx 0.5\%$, wait ≈ 6.17 min, completion ≈ 0.995), but operates with low utilisation (≈ 0.33) and relatively low driver hourly earnings ($\approx \text{£}12.6/\text{h}$).

Driver-sustainability setting: $(m_D, m_H, m_F, m_C) = (0.80, 1.30, 1.20, 1.15)$,

which increases utilisation to a more balanced level (≈ 0.74) and improves driver outcomes (earnings per hour $\approx \text{£}23.1/\text{h}$), but operates close to the passenger cancellation guardrail (cancel $\approx 5.98\%$).

Based on these results, we recommend a *mode-switching* policy:

- (a) *Balanced mode (default)*: operate close to the real-data baseline to avoid both shortages and redundancy.
- (b) *Service-recovery mode (temporary)*: if rider KPIs show sustained decline (e.g. cancellations rising toward the 6% limit, or tail waiting times increasing), temporarily switch toward the service-oriented setting until the system returns to the target band, then revert to balanced mode to avoid long periods of low utilisation
- (c) *Driver-retention mode (targeted)*: if driver sustainability indicators start to worsen (e.g. hourly earnings falling below the balanced-region level or variation in earnings continuing to rise), temporarily move toward the driver-sustainability setting while monitoring cancellations closely due to its proximity to the service constraint.

This method adjusts supply in a controlled way: service is stabilised when needed, but the system is not left with too many drivers on the road.

Method 2: Dispatch-rule refinement within the balanced regime (cost-aware + fairness-aware assignment). To improve performance without structurally increasing supply, dispatch rules can be adjusted so that improvements come from *efficiency* and *allocation design*:

- (a) *Cost-aware matching (reducing empty travel)*: To minimise expenses, the system should prioritise matching drivers with shorter pick-up distances. When longer pick-ups are unavoidable, apply a small penalty if alternative drivers exist so that unnecessary deadheading is reduced. This directly reduces empty driving distance and operating costs, which improve the system net revenue without needing to increase demand or maintain a surplus of drivers.
- (b) *Fairness-aware tie-breaking (earnings stability)*: Fairness should also be considered when selecting among candidate drivers. In particular, when multiple drivers are similarly close to a pick-up, give priority to drivers who have earned less recently or have been idle longer. This helps stabilise income and reduce earnings dispersion while maintaining service quality for passengers.

Together, these dispatch adjustments reflect the principle that the platform should seek improvements through better matching efficiency and more stable earnings allocation, rather than relying on a permanent surplus of drivers.

Table 19: Comprehensive Scenario Comparison (10 runs each): key KPIs shown as mean $\pm$ std. Green marks the best three values for each metric, red marks the worst three (according to metric desirability), and bold indicates the extreme value(s).

Metric	default	real data	passenger	profit	cost	driver	balance	quality	MAX-MIN
avg available drivers	1.195$\pm$ 0.120	9.013 $\pm$ 0.772	43.795 $\pm$ 1.060	43.657 $\pm$ 1.025	45.642$\pm$ 1.021	9.798 $\pm$ 0.689	43.538 $\pm$ 1.180	43.948 $\pm$ 0.996	44.447
avg busy drivers	19.548 $\pm$ 0.448	25.556 $\pm$ 0.301	21.375 $\pm$ 0.154	21.414 $\pm$ 0.117	21.280 $\pm$ 0.214	25.413 $\pm$ 0.308	21.488 $\pm$ 0.189	21.477 $\pm$ 0.151	6.008
avg rides per driver	7.332 $\pm$ 0.043	6.939 $\pm$ 0.094	5.024 $\pm$ 0.087	5.027 $\pm$ 0.084	5.239 $\pm$ 0.099	8.736$\pm$ 0.114	5.048 $\pm$ 0.098	5.022$\pm$ 0.071	3.714
avg earnings per driver	173.997 $\pm$ 1.266	165.306 $\pm$ 2.165	121.265$\pm$ 1.927	121.531 $\pm$ 1.922	121.749 $\pm$ 2.321	213.654$\pm$ 2.947	121.934 $\pm$ 2.427	121.278 $\pm$ 1.647	92.389
avg earnings per hour	24.977$\pm$ 0.151	22.573 $\pm$ 0.285	12.717 $\pm$ 0.210	12.758 $\pm$ 0.192	11.902$\pm$ 0.228	22.789 $\pm$ 0.294	12.787 $\pm$ 0.250	12.721 $\pm$ 0.175	13.075
earnings cv	0.1898$\pm$ 0.0033	0.2175 $\pm$ 0.0071	0.4131 $\pm$ 0.0077	0.4129 $\pm$ 0.0056	0.4138 $\pm$ 0.0094	0.2131 $\pm$ 0.0069	0.4138 $\pm$ 0.0088	0.4151$\pm$ 0.0046	0.2253
max earnings	286.170 $\pm$ 6.015	276.285$\pm$ 8.437	304.624 $\pm$ 20.779	298.891 $\pm$ 9.314	295.003 $\pm$ 10.216	339.921$\pm$ 7.406	305.226 $\pm$ 10.855	303.854 $\pm$ 16.333	63.636
busy time utilization	0.9424$\pm$ 0.0050	0.7395 $\pm$ 0.0185	0.3281 $\pm$ 0.0064	0.3292 $\pm$ 0.0056	0.3181$\pm$ 0.0064	0.7219 $\pm$ 0.0156	0.3306 $\pm$ 0.0072	0.3283 $\pm$ 0.0060	0.6243
completed rides	15752.8$\pm$ 365.1	23643.7 $\pm$ 141.2	24793.8 $\pm$ 163.1	24794.3 $\pm$ 113.3	24757.6 $\pm$ 202.2	23692.9 $\pm$ 238.4	24880.6 $\pm$ 194.8	24914.1$\pm$ 125.0	9161.3
rider cancellations	5844.8$\pm$ 361.6	1294.1 $\pm$ 156.9	66.6 $\pm$ 26.8	65.0 $\pm$ 22.4	60.6 $\pm$ 11.3	1218.9 $\pm$ 140.0	60.9 $\pm$ 16.1	60.4$\pm$ 16.9	5784.4
cancellation rate	0.2705$\pm$ 0.0161	0.0519 $\pm$ 0.0062	0.0027 $\pm$ 0.0011	0.0026 $\pm$ 0.0009	0.0024$\pm$ 0.0004	0.0489 $\pm$ 0.0058	0.0024$\pm$ 0.0007	0.0024$\pm$ 0.0007	0.2681
completion rate	0.7291$\pm$ 0.0161	0.9477 $\pm$ 0.0062	0.9972 $\pm$ 0.0011	0.9973 $\pm$ 0.0009	0.9974 $\pm$ 0.0004	0.9508 $\pm$ 0.0058	0.9974 $\pm$ 0.0007	0.9975$\pm$ 0.0006	0.2684
avg waiting riders	1.6215$\pm$ 0.1062	0.3587 $\pm$ 0.0447	0.0176 $\pm$ 0.0077	0.0183 $\pm$ 0.0066	0.0170 $\pm$ 0.0035	0.3380 $\pm$ 0.0392	0.0168 $\pm$ 0.0048	0.0165$\pm$ 0.0053	1.6050
avg wait	24.970$\pm$ 0.181	15.867 $\pm$ 0.626	5.985 $\pm$ 0.101	5.989 $\pm$ 0.095	5.872$\pm$ 0.062	15.425 $\pm$ 0.506	6.014 $\pm$ 0.133	5.966 $\pm$ 0.144	19.097
median wait	22.225$\pm$ 0.187	12.257 $\pm$ 0.625	4.938 $\pm$ 0.073	4.966 $\pm$ 0.071	4.849$\pm$ 0.060	11.794 $\pm$ 0.459	4.956 $\pm$ 0.088	4.928 $\pm$ 0.103	17.376
max wait	104.431$\pm$ 11.894	86.495 $\pm$ 3.291	70.880 $\pm$ 6.272	74.085 $\pm$ 9.961	69.650 $\pm$ 7.180	91.692 $\pm$ 6.485	73.235 $\pm$ 7.558	65.262$\pm$ 8.872	39.169
std wait	14.794$\pm$ 0.095	12.507 $\pm$ 0.300	4.527 $\pm$ 0.113	4.519 $\pm$ 0.087	4.458$\pm$ 0.052	12.319 $\pm$ 0.287	4.558 $\pm$ 0.167	4.489 $\pm$ 0.164	10.337
avg fare per ride	23.733 $\pm$ 0.068	23.823 $\pm$ 0.073	24.137 $\pm$ 0.052	24.174 $\pm$ 0.056	23.238$\pm$ 0.070	24.456$\pm$ 0.056	24.156 $\pm$ 0.046	24.149 $\pm$ 0.072	1.218
system utilization	0.9424$\pm$ 0.0050	0.7395 $\pm$ 0.0185	0.3281 $\pm$ 0.0064	0.3292 $\pm$ 0.0056	0.3181$\pm$ 0.0064	0.7219 $\pm$ 0.0156	0.3306 $\pm$ 0.0072	0.3283 $\pm$ 0.0060	0.6243
total earnings	373846.2$\pm$ 8112.1	563263.2 $\pm$ 3748.3	598452.9 $\pm$ 3769.5	599379.4 $\pm$ 2672.4	575325.0 $\pm$ 5165.7	579436.7 $\pm$ 6429.4	601012.1 $\pm$ 4484.4	601646.0$\pm$ 3601.8	227799.9
total costs	56275.8 $\pm$ 1285.5	73565.6 $\pm$ 857.8	73834.3 $\pm$ 524.2	73957.3 $\pm$ 377.0	49011.0$\pm$ 479.4	84117.5$\pm$ 1024.5	74211.0 $\pm$ 638.9	74195.6 $\pm$ 517.8	35106.5
net revenue	317570.3$\pm$ 6837.2	489697.6 $\pm$ 3732.4	524618.6 $\pm$ 3293.6	525422.0 $\pm$ 2349.0	526314.0 $\pm$ 4692.8	495319.2 $\pm$ 5840.9	526801.2 $\pm$ 3908.0	527450.4$\pm$ 3185.3	209880.1

Notes: MAX-MIN is computed from scenario means only (not from mean $\pm$ std). Colors indicate the best/worst three values by metric desirability.

6 Model Limitations

The simulation provides a structured quantitative approximation of BoxCar’s operations; however, it is subject to several limitations.

1. Key stochastic components — such as rider arrivals, driver availability, and rider patience — are modelled using simple probability distributions with time-invariant parameters. Because observational data on rider patience are unavailable, the assumed distribution cannot be empirically validated. In addition, differences between peak and off-peak periods are not explicitly modelled.
2. As is common with simulations which track events over time, our system needs a little while at the beginning of the start to settle into a steady state. The very first observations might slightly effect the outcomes. However, the system stabilises quite rapidly, and given the long duration of our simulations, this initial effect is negligible.
3. Further limitations arise from the structural simplifications we’ve made. For instance, we’re calculating travel distances using simple straight-line (Euclidean) geometry, ignoring factors like real road networks, traffic jams, or weather conditions. Driver behaviour is also simplified: the model assumes that drivers will take every ride offered and simply stay put after dropping someone off, rather than strategically moving to areas with more demand, taking breaks, or declining rides. The pricing is also static, not adjusting based on how busy things are. (e.g., surge pricing).
4. While we’ve run enough simulations to effectively compare different strategies, getting highly precise real-world predictions would require more precise data. Therefore, the key performance indicators we’re reporting should be seen as helpful for evaluating policy choices rather than definitive forecasts of what will happen in real life. Nevertheless, the model is still valuable

7 Conclusion

This study evaluates the operational performance of BoxCar’s ride-sharing system using a data-driven simulation framework. A key finding is that the project parameters do not align well with the observed data. Under the default configuration, the system operates near capacity, leading to longer waits, higher cancellations, and weaker net revenue. After adjusting key inputs using observed data, the baseline operates more stably and service reliability improves significantly. We therefore recommend using data-based calibration to guide our operational decisions.

Analysis also shows that there are clear trade-offs in providing low wait times. To achieve shorter wait times, companies must keep more excess drivers on the road than necessary, which reduces hourly earnings and reduces long-term driver retention. Among the scenarios that were evaluated, the System Balance scenario represents the highest level of trade-off to BoxCar; this scenario offers companies a very high level of service reliability and financial performance without considering driver productivity (this would be considered an unsustainable level of impact). Additionally, the Cost scenario illustrates that improving the efficiency of matching and the reduction of empty miles is a more sustainable and profitable approach than relying on oversupply as a means of achieving these results.

There are further opportunities to improve the accuracy of the model by incorporating additional behavioural variables, such as cancellation timing and demand variability during peak and off-peak periods. These variables would improve the model’s ability to predict simulation results. Although the limitations of the simulations will continue to exist, the simulation framework continues to be a very valuable tool for evaluating and defining the trade-offs between rider service levels, driver outcomes, and overall business profitability.

Appendix

A Pseudocode for BoxCar DES

Pseudo-Code for Simulating the BoxCar System (Main Loop)

```

1: Input: Termination  $T$ ; random generators for driver/rider processes; cost/fare parameters.
2: Output: KPI estimates (wait times, cancellation/completion, utilization, profit, etc.).
3: State:  $A$  (available drivers),  $B$  (busy drivers),  $W$  (waiting riders),  $M$  (matched riders).
4: Accumulators:  $\text{areaW}, \text{areaA}, \text{areaB} \leftarrow 0$   $\triangleright \int |W(t)|dt, \int |A(t)|dt, \int |B(t)|dt$ 
5: Initialise:  $t \leftarrow 0$ ;  $A, B, W, M \leftarrow \emptyset$ ; counters  $\leftarrow 0$ .
6: Initialise: Event calendar  $EC$  with first DRIVERONLINE, first RIDERARRIVAL, and TERMINATION at time  $T$ .
7: while  $EC \neq \emptyset$  do
8:   Extract next event  $(t_n, \text{type}, \text{data})$  from  $EC$  (minimum time).
9:    $\Delta t \leftarrow t_n - t$ ; update accumulators:
10:   $\text{areaW} \leftarrow \text{areaW} + \Delta t \cdot |W|$ ,  $\text{areaA} \leftarrow \text{areaA} + \Delta t \cdot |A|$ ,  $\text{areaB} \leftarrow \text{areaB} + \Delta t \cdot |B|$ .
11:  Set  $t \leftarrow t_n$ .
12:  if  $\text{type} = \text{DRIVERONLINE}$  then DRIVERONLINE( $t$ )
13:  else if  $\text{type} = \text{DRIVEROFFLINE}$  then DRIVEROFFLINE( $t, \text{data}$ )
14:  else if  $\text{type} = \text{RIDERARRIVAL}$  then RIDERARRIVAL( $t$ )
15:  else if  $\text{type} = \text{RIDERCANCEL}$  then RIDERCANCEL( $t, \text{data}$ )
16:  else if  $\text{type} = \text{PICKUPCOMPLETE}$  then PICKUPCOMPLETE( $t, \text{data}$ )
17:  else if  $\text{type} = \text{DROPOFFCOMPLETE}$  then DROPOFFCOMPLETE( $t, \text{data}$ )
18:  else if  $\text{type} = \text{TERMINATION}$  then TERMINATION( $t$ ); break end if
19: end while
20: Compute KPI summary from counters and  $\text{areaW}, \text{areaA}, \text{areaB}$ 

```

Procedure MatchDriverRider($t, (d, r)$)

```

1: MATCHDRIVERRIDER( $t, (d, r)$ ):
2:   Remove  $d$  from  $A$ ; remove  $r$  from  $W$ .
3:   Assign rider  $r$  to driver  $d$ ; move rider  $r$  to  $M$ ; move driver  $d$  to  $B$ .
4:   Set driver busy-start  $\leftarrow t$  and mark driver as busy.
5:   Compute pickup distance  $d_p \leftarrow \text{Distance}(\text{loc}(d), \text{org}(r))$ .
6:   Expected pickup time  $\tau_p \leftarrow \text{TravelTimeHours}(d_p)$ .
7:   Sample actual pickup time  $\hat{\tau}_p \leftarrow \text{TripDuration}(\tau_p)$ .
8:   Schedule event PICKUPCOMPLETE at time  $t + \hat{\tau}_p$  with data  $(d, r, d_p)$ .

```

Procedure DriverOnline(t)

```

1: DRIVERONLINE( $t$ ):
2:   Create new driver  $d$  with location RandomLocation().
3:   Sample online duration  $u \leftarrow \text{DriverOnlineDuration}()$ ; set  $\text{onlineUntil}(d) \leftarrow t + u$ .
4:   Add  $d$  to  $A$  and to the set of all drivers seen.
5:   Schedule DRIVEROFFLINE at time  $\text{onlineUntil}(d)$  with data  $d$ .
6:   Sample next driver interarrival  $\Delta d \leftarrow \text{NextDriverInterarrival}()$ 
7:   Schedule DRIVERONLINE at  $t + \Delta d$ .
8:   if  $W \neq \emptyset$  then
9:     Choose rider  $r^* \leftarrow \text{argmin}_{r \in W} \text{Distance}(\text{loc}(d), \text{org}(r))$  and MATCHDRIVERRIDER( $t, d, r^*$ ).
10:  end if

```

Procedure DriverOffline(t, d)

```

1: DRIVEROFFLINE( $t, d$ ):
2:   if  $d \in A$  then remove  $d$  from  $A$  and set  $\text{offlineTime}(d) \leftarrow t$ .
3:   else if  $d \in B$  then set  $\text{goOfflineAfterDropoff}(d) \leftarrow \text{true}$ .           ▷ offline delayed until ride completes
4:   end if

```

Procedure RiderArrival(t)

```

1: RIDERARRIVAL( $t$ ):
2:   Create new rider  $r$  with origin RandomLocation() and destination RandomLocation().
3:   Set request time  $\text{req}(r) \leftarrow t$ ; region  $\leftarrow \text{ClassifyRegion}(\text{org}(r))$ .
4:   Sample patience  $p \leftarrow \text{RiderPatience}()$ ; set deadline  $\text{dead}(r) \leftarrow t + p$ .
5:   Add  $r$  to  $W$ .
6:   Schedule RIDERCANCEL at time  $\text{dead}(r)$  with data  $r$ .
7:   Sample next rider interarrival  $\Delta t \leftarrow \text{NextRiderInterarrival}()$ ; schedule RIDERARRIVAL at  $t + \Delta t$ .
8:   if  $A \neq \emptyset$  then
9:     Choose driver  $d^* \leftarrow \text{argmin}_{d \in A} \text{Distance}(\text{loc}(d), \text{org}(r))$  and MATCHDRIVERRIDER( $t, d^*, r$ ).

```

Procedure RiderCancel(t, r)

```

1: RIDERCANCEL( $t, r$ ):
2:   if  $r \in W$  then remove  $r$  from  $W$  and increment cancellation counter.           ▷ cancels only if still waiting
3:   end if

```

Procedure PickupComplete($t, (d, r, d_p)$)

```

1: PICKUPCOMPLETE( $t, (d, r, d_p)$ ):
2:   if  $r \notin M$  then return           ▷ e.g. already removed/unexpected
3:   Remove rider  $r$  from  $M$  (pickup occurs).
4:   Record wait time:  $w \leftarrow t - \text{req}(r)$ ; append  $w$  to wait-time list and update totals.
5:   Compute trip distance  $d_t \leftarrow \text{Distance}(\text{org}(r), \text{dest}(r))$ .
6:   Expected trip time  $\tau_t \leftarrow \text{TravelTimeHours}(d_t)$ ; sample actual  $\hat{\tau}_t \leftarrow \text{TripDuration}(\tau_t)$ .
7:   Schedule DROPOFFCOMPLETE at time  $t + \hat{\tau}_t$  with data  $(d, r, d_t, \text{dest}(r), d_p)$ .

```

Procedure DropoffComplete($t, (d, r, d_t, \text{dest}, d_p)$)

```

1: DROPOFFCOMPLETE( $t, (d, r, d_t, \text{dest}, d_p)$ ):
2:   Remove driver  $d$  from  $B$ ; set  $\text{loc}(d) \leftarrow \text{dest}$ .
3:   Update busy time:  $b \leftarrow t - \text{busyStart}(d)$ ; add  $b$  to total busy hours; clear  $\text{busyStart}(d)$ .
4:   Fare:  $f \leftarrow \text{BaseFare} + \text{FarePerMile} \cdot d_t$ ; update driver earnings and system totals.
5:   Cost:  $c \leftarrow \text{CostPerMile} \cdot (d_p + d_t)$ ; update total costs; increment completed rides.
6:   if  $\text{goOfflineAfterDropoff}(d) = \text{true}$  or  $t \geq \text{onlineUntil}(d)$  then
7:     set  $\text{offlineTime}(d) \leftarrow t$  and return.
8:   end if
9:   Add driver  $d$  to  $A$ .
10:  if  $W \neq \emptyset$  then
11:    choose rider  $r^* \leftarrow \text{argmin}_{r \in W} \text{Distance}(\text{loc}(d), \text{org}(r))$  and MATCHDRIVERRIDER( $t, d, r^*$ ).
12:  end if

```

Procedure Termination(t)

```

1: TERMINATION( $t$ ):
2:   for each busy driver  $d \in B$  do
3:     Add remaining busy time ( $t - \text{busyStart}(d)$ ) to total busy hours
4:   end for

```

B Supplementary Tables and Configuration Details

Scenario	Objective	Score Components (higher is better)
scenario 1 Passenger Experience	Passenger experience priority	$0.30(1 - \frac{\text{wait}}{30}) + 0.25(1 - \frac{\text{cancel}}{0.06}) + 0.20(\text{completion}) + 0.15(1 - \frac{\text{cost}}{30}) + 0.10(1 - \frac{\text{p90}}{50})$
scenario 2 System Profit Maximization	Profit priority with service guardrails	$0.65(\frac{\text{profit}}{550k}) + 0.15(1 - \frac{\text{cancel}}{0.06}) + 0.10(1 - \frac{\text{wait}}{30}) + 0.10(\text{completion})$
scenario 3 Passenger Cost Priority	Affordability priority with acceptable service	$0.55(1 - \frac{\text{cost}}{30}) + 0.20(1 - \frac{\text{wait}}{30}) + 0.15(1 - \frac{\text{cancel}}{0.06}) + 0.10(\text{completion})$
scenario 4 Driver Optimal	Driver sustainability and earnings quality	$0.35(\frac{\text{earnings}}{200}) + 0.25(\frac{\text{earn hr}}{30}) + 0.25(\frac{\text{rides dr}}{10}) + 0.15(1 - \frac{\text{online hrs}}{10})$
scenario 5 System Balance	System-wide balance across service/finance/ops	$0.25(\frac{\text{profit}}{550k}) + 0.20(1 - \frac{\text{wait}}{30}) + 0.20(1 - \frac{\text{cancel}}{0.06}) + 0.15(\text{completion}) + 0.10(\text{utilization}) + 0.10(1 - \frac{\text{cost}}{30})$
scenario 6 Quality Priority	Service quality + utilization	$0.25(1 - \frac{\text{cancel}}{0.06}) + 0.20(1 - \frac{\text{wait}}{30}) + 0.15(1 - \frac{\text{p90}}{50}) + 0.15(1 - \frac{\text{waiting riders}}{5}) + 0.15(\text{utilization}) + 0.10(\text{completion})$

Table 20: Scenario Configuration and Scoring Functions

Scenario	m_D	m_H	m_F	m_C	Cancel	Wait	P90	Util.	£/hr	Score
Passenger Experience	1.45	1.35	1.10	1.20	0.0050	6.17	11.65	0.3286	12.58	0.7727
System Profit	1.45	1.35	1.10	1.20	0.0050	6.17	11.65	0.3286	12.58	0.9665
Passenger Cost	1.40	1.45	0.80	0.80	0.0075	6.09	11.49	0.3149	11.62	0.5151
Driver Optimal	0.80	1.30	1.20	1.15	0.0598	15.89	33.57	0.7433	23.06	0.7728
System Balance	1.45	1.35	1.10	1.20	0.0050	6.17	11.65	0.3286	12.58	0.7940
Quality Priority	1.45	1.35	1.10	1.20	0.0050	6.17	11.65	0.3286	12.58	0.8012

Table 21: Optimised parameter multipliers and key KPIs for each scenario (168-hour horizon; 10 replications). Here m_D scales the driver arrival rate, m_H scales the driver online-duration bounds, m_F scales the base fare, and m_C scales the operating cost per mile. Wait and P90 are in minutes; Util. denotes (busy-time) utilisation. *Note:* Several scenarios share the same optimal parameter combination because their objective functions emphasise similar service-quality metrics, leading the optimisation procedure to converge to the same configuration within the feasible region.

C Python Code Repository for the BoxCar DES

The source code for the BoxCar discrete-event simulation (DES) is maintained in the following GitHub repository: https://github.com/zhoujasper/Simulation_Project_1_Group_48.