

# Operational Plan for the Amazonoff Company

52086

**A report submitted The Amazonoff Company**

April 24, 2023

# Contents

<b>1</b>	<b>Executive Summary</b>	<b>1</b>
<b>2</b>	<b>Management Report</b>	<b>2</b>
2.1	Warehousing . . . . .	2
2.2	Vehicle Routing . . . . .	3
2.3	Internal Depot Transportation with Uncertainty . . . . .	3
2.3.1	Robust Formulation . . . . .	4
2.3.2	Non-Robust Formulation . . . . .	4
2.4	Product Specific Insurance . . . . .	5
2.4.1	Likelihood of Legal Issues . . . . .	5
2.4.2	Estimation of Capital Stock . . . . .	7
<b>A</b>	<b>Warehousing Methodology</b>	<b>8</b>
A.1	Data Generation . . . . .	8
A.1.1	Grid Generation . . . . .	8
A.1.2	Demand Generation . . . . .	8
A.2	Modelling . . . . .	10
A.2.1	Base-Line Model . . . . .	10
A.2.2	Capacitated Model . . . . .	11
A.2.3	Balanced Loads . . . . .	11
<b>B</b>	<b>Vehicle Routing Methodology</b>	<b>13</b>
B.1	Data Generation . . . . .	13
B.1.1	Composite Client Generation . . . . .	13
B.2	Modelling . . . . .	14
B.2.1	Base-Line Model . . . . .	14
B.2.2	Multiple Vehicles . . . . .	15
<b>C</b>	<b>Internal Depot Transportation with Uncertainty Methodology</b>	<b>17</b>
C.1	Robust Formulation . . . . .	17
C.2	Non-Robust Formulation . . . . .	18

---

<b>D Product Specific Insurance Methodology</b>	<b>21</b>
D.1 Base-Line Model . . . . .	21
D.2 Confidence . . . . .	24
D.3 Variance Reduction . . . . .	25

# Chapter 1

## Executive Summary

**Objective:** This report's objective is to document the modelling and simulation completed on behalf of the Amazonoff Company. This report also outlines changes that we feel could improve operating efficiency, whilst minimising costs.

**Findings:** Section 2.1 describes the cost-minimising assignment of customers to facilities so that we satisfy each customer's demand. We find the optimal operating cost to be 210.64, even after we account for the capacity of each facility. We also suggest methods which allow the company to balance loads between facilities. In Section 2.2 we construct optimal routes for the delivery vehicles to follow. We show that the two processes discussed above can be modelled separately, and therefore are easily actionable. Our routing implementation offers an additional cost of just 88.86, using just one delivery vehicle. We believe it is important to consider uncertainty in our research and in Section 2.3, we offer insights into how we can factor this into the internal running of the depot. Finally, in Section 2.4, we simulate the next 12 months for your insurance subsidiary. It should be noted that we estimate the probability of legal issues for the company to be roughly 40%. To help remedy this, we further estimate the value of your capital holdings to a high degree of certainty and find it to be in the region of 32412.96. We, therefore, suggest a small increase in the monthly rate you charge customers, or sourcing a capital injection before the next year begins.

**Conclusion:** Following our research, if our suggestions are implemented, we find a short-term operations cost of 299.5. In the longer term, we suggest following the advice laid out in the rest of the report so as to avoid legal issues.

## Chapter 2

# Management Report

In this chapter, we lay out a more detailed analysis of your operations, and offer distinct guidelines on how to fulfil your warehousing, routing and e-commerce requirements.

### 2.1 Warehousing

The Amazonoff Company operates in the space of a  $21 \times 21$  grid, in which exist the entire customer base and potential facilities. We provide a map for reference in xxx We suggest only opening a subset of these facilities, as we give a satisfactory assignment of all 60 of your customers with just 14 facilities, allowing the firm to save expenditure on unnecessary rent. We recommend leaving out facility 1. Outlined in Table 2.1 are the assignment and respective facilities we consider to be optimal, and we refer the reader to Appendices A.2.1 and A.2.2 for a technical discussion of our modelling process.

We find that even upon factoring in the demands of each client, the facilities are easily able to hold the required stock, and as such our allocation remains optimal. It may seem odd to the reader we suggest opening so many facilities, but upon consideration of the costs for opening facilities, the motivation becomes clear. As the index of the facility increases, the cost to open it decreases exponentially, and as such it incurs a minimal cost to open an extra facility. With this in hand, it makes sense to open many facilities, so we can truly minimise the distance between them and the clients. This is further backed up by xxx

In Table 2.1, we observe the demand faced by each facility. It is clear that the distribution is extremely unbalanced, with facility 8 facing 145 units, and facility 14 facing just 14. As such, we refer the reader to Appendix A.2.3 for a modelling approach that allows the firm to control this difference.

We calculate the cost of this approach 210.64 if implemented correctly.

Facility	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Customers Assigned	10	14	40	4	5	42	3	1	17	8	26	12	45	7
	27	21	46	58	19		6	2	25	23	34	18		13
	30		47	60	22		9	28	39	24	43	29		15
	33		50		31		11	57		32	59			38
	37				49		15			36				
	41						20			53				
	44						35			54				
	48						52			56				
	51						55							

Table 2.1: Assignment of customers to facilities

Facility	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Demand Faced	104	34	50	47	54	17	145	35	44	91	53	52	14	65
Capacity	200	300	400	500	600	700	800	900	1000	1100	1200	1300	1400	1500

Table 2.2: Demands faced by each facility

## 2.2 Vehicle Routing

We next tackle the routing operation. The company has 25 *composite* clients whose deliveries are to be prioritised. There is a single depot and the delivery vehicles have no capacity limit. In our modelling, we considered using one or three delivery vehicles and found that the total cost of using either is the same, that being 88.86. This is to be expected since all routes must return to the depot, and if we used more vehicles, we would pay unnecessary costs on the return leg. In Figure 2.1 we outline the optimal route, where the depot is highlighted in red and labelled d1. Each composite client is then labelled by their index and prefixed with a c. For a technical discussion of our models, please refer to Appendices B.2.1 and B.2.2

## 2.3 Internal Depot Transportation with Uncertainty

Given the nature of the company, it is important to factor in uncertainty when modelling, specifically in the internal depot operations. In this section, we propose two models that allow the firm varying degrees of control. The first is a *robust* model, meaning regardless of the dimensions of each item, there will always be some way to pack the electric vehicle. The second allows the firm some control over the likelihood that the vehicle cannot be packed. We, therefore, label this model as the *non-robust*



a small chance that the allocation of goods will not fit on the vehicle, but this can be remedied with the appropriate setting of the risk level. Running simulations on this situation will inform this decision, and with this, we hypothesise that the overall profit would be higher in this model.

The choice between these two approaches will depend largely on the needs of the company. If the consequences of allocating too many items to one vehicle are large, we recommend the robust approach. Otherwise, we suggest the second model since the profit per vehicle would certainly be higher. It is our opinion that the firm would be better off implementing the second.

## 2.4 Product Specific Insurance

We now bring our attention to the insurance subsidiary of the company. In particular, we wish to simulate the next 12 months of your operations and deduce meaningful estimates of key values, so as to inform your decision-making in the next year. The key values we considered are the likelihood of legal issues for your firm, and the expected capital stock at the end of the term.

Based on your data, we run the simulation with a starting capital stock of 50,000 and a fixed monthly incoming fee of 300. By then considering the entry and leaving rates of your service, and the rate at which customers lodge claims, we can generate estimates of these values in the long run.

### 2.4.1 Likelihood of Legal Issues

We first wish to estimate the probability that your firm will encounter legal issues. After running the simulation 1000 times, we return with the distribution laid out in 2.4. We immediately see how even the distribution is, and precisely we found the probability of legal issues to be 0.401. We deem this figure to be far too high, and to counteract this we suggest raising the monthly fee. In Figure 2.3 we depict the results of two further simulations in which we raise the monthly fee for the service to  $M = 350$  and  $M = 400$ . We see that with an increase of 350, the failure decreases by almost half, and in the case of 400 it halves again. Specifically, these probabilities are 0.211, and 0.086 respectively.

There is also the option of sourcing a capital injection, and we ran further experiments with this idea. However, we found that an increase of around 20,000 in initial capital gave us comparable results to the monthly fee increases, and as such, we suggest an increase in only the monthly fee to decrease the chances of legal issues.



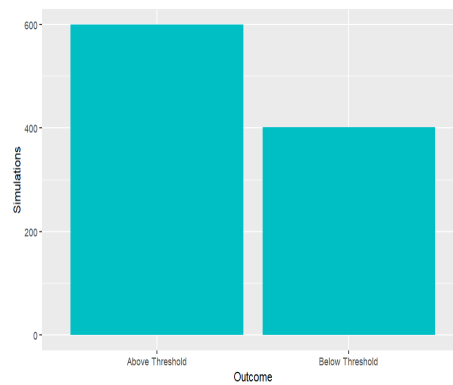
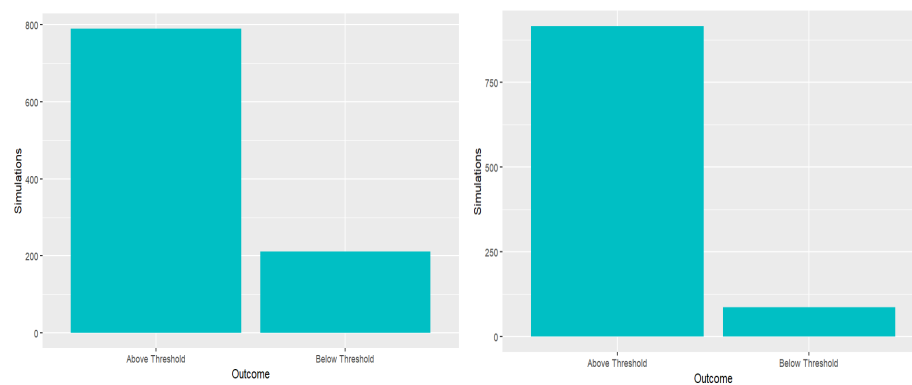


Figure 2.2: Distribution of outcomes (1000 simulations)

Figure 2.3: Distribution of outcomes with  $M = 350, 400$  (1000 Simulations)

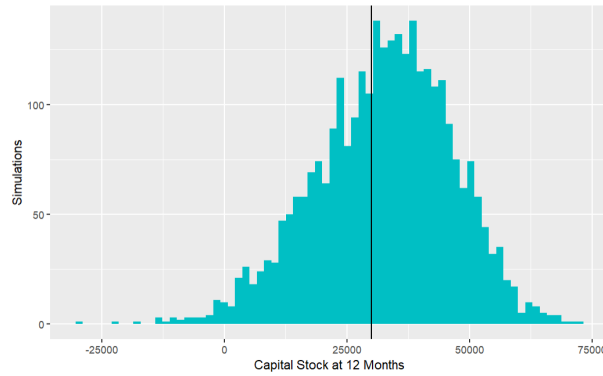


Figure 2.4: Distribution of capital stock after 12 months (3000 Simulations)

### 2.4.2 Estimation of Capital Stock

The next section is dedicated to the estimation of your capital stock after a 12-month period, which is only more justified given the analysis in the previous section. If we can gain a good estimate, we can provide a more accurate value for a suggested monthly fee.

We aim to find an estimate for the capital stock after 12, which we denote  $C_{12}$ , that we can be 90% sure is within 500 of the true value. To build a better picture, we ran the simulation many more times than in the previous section, and at iteration 2509, we find 32412.96 as our estimator of capital stock. We also have the standard deviation of this estimate to be 303.88, and with this estimate, we can give an interval to which we are 95% the true value lies within, and we calculate this to be  $C_{12} \pm 595.59$ . A detailed description of the simulation process, and how we arrived at our estimates, can be found in Appendices D.1 and D.2.

Upon reviewing Figure 2.4, we observe there is a massive variance in our estimate. In some cases, we finish with negative capital stock. This variance is due to the insurance payout formula, which can sometimes provide extortionately high payouts. In Appendix D.3, we describe measures to reduce this variance in our estimate.

We have now presented a clear image of the 12 months for your insurance subsidiary, and with this, we would like to recommend a specific monthly service charge, with the intent of reducing the likelihood you encounter legal issues. Using our analysis laid out in this section, we suggest  $M = 420$ , which will allow the firm to avoid legal issues 95% of the time.

# Appendix A

## Warehousing Methodology

Contained in this appendix is detailed, technical report of the methodology applied in the warehousing scenario.

### A.1 Data Generation

#### A.1.1 Grid Generation

Associated Files: `grid_generation.Rmd`

Before we carried out our modelling, it was necessary to generate certain data that we could feed into our models. First, we generated the coordinates of each client. Our method is outlined in Algorithm 1, and implemented in `grid_generation.Rmd`

We first let  $n_f$  and  $n_c$  be the number of facilities and customers respectively. Then for each axis, we randomly generate the required amount of coordinates, and transform them to be in a  $[-10, 10]$  interval. Using the loop on lines 2 and 4 as an example, we multiply each random coordinate on the  $y$  axis of the facilities by 20, to get  $n_f$  random variables between 0 and 20. Then we -10 from these values, to arrive at  $n_f$  values between  $[-10, 10]$ . The  $x$  axis is repeated like this, and so are the axes for the customers.

We are left with the coordinates as required, and after some wrangling to get into the right format, we are left with xxx.

#### A.1.2 Demand Generation

Associated Files: `demand_generation.Rmd`

Upon expansion of our model, it was also necessary to generate the demand of each of the 60 clients in  $C$ . We were given that each demand is between 1 and 25, and was uniformly distributed. The method for calculating these random variables is outlined in Algorithm 2 and implemented in `demand_generation.Rmd`

In the first line, we initialise a  $60 \times 1$  vector of random `uniform(0,1)` variables.

---

**Algorithm 1:** Coordinate Generation

---

**Data:**  $n_f = 15, n_c = 60$ **Result:** Coordinates on a  $[-10, 10] \times [-10, 10]$  grid

```

1  $f_y \leftarrow n_f \times 1$  vector of random uniform(0,1) variables;
2 for  $f \in f_y$  do
3    $f \leftarrow 20 \cdot f$  ;
4    $f \leftarrow f - 10$  ;
5 end
6  $f_x \leftarrow n_f \times 1$  vector of random uniform(0,1) variables;
7 for  $f \in f_x$  do
8    $f \leftarrow 20 \cdot f$  ;
9    $f \leftarrow f - 10$  ;
10 end
11  $c_y \leftarrow c_f \times 1$  vector of random uniform(0,1) variables;
12 for  $c \in c_y$  do
13    $c \leftarrow 20 \cdot c$  ;
14    $c \leftarrow c - 10$  ;
15 end
16  $c_x \leftarrow c_f \times 1$  vector of random uniform(0,1) variables;
17 for  $c \in c_y$  do
18    $c \leftarrow 20 \cdot c$  ;
19    $c \leftarrow c - 10$  ;
20 end

```

---



---

**Algorithm 2:** Demand Generation

---

**Data:** Set of customers  $C$ **Result:** Set of demands for each customer  $c$ 

```

1  $U \leftarrow 60 \times 1$  vector of random uniform(0,1) variables ;
2 for  $u \in U$  do
3    $u \leftarrow 25 \cdot u$  ;
4    $u \leftarrow \text{upper}(u)$  ;
5 end
6 Let each index of  $U$  represent the corresponding customer in  $C$  ;

```

---

Then the loop on lines 2-4 firstly multiplies those values by 25, to give us 25 random variables between 0 and 25, and then returns the upper integer of that value, which gives us 25 random integers between 1 and 25, as required.

## A.2 Modelling

### A.2.1 Base-Line Model

Associated Files: `model(1.1).mod`, `f_loc(1.1).dat`, `c_loc(1.1).dat`,  
`data(1.1).dat`, `run(1.1).run`, `out(1.1).txt`

With our data in hand, we can now move to modelling our problems, the first of which can be modelled in the form of an integer program. The formal definition of our model is outlined below:

#### Sets

$C$  set of Customers

$F$  set of Facilities

#### Parameters

$c_{ij}$   $(i, j) \in C \times F$  cost of assigning customer  $i$  to facility  $j$

$k_f$   $f \in F$  cost of opening facility  $f$

#### Variables

$x_{ij} \in \{0, 1\}$   $(i, j) \in C \times F$  Assigning customer  $i$  to facility  $j$

$b_f \in \{0, 1\}$   $f \in F$  Opening facility  $f$

#### Objective

$$\min \sum_{(i,j) \in C \times F} c_{ij} x_{ij} + \sum_{f \in F} k_f b_f \quad (\text{A.1})$$

#### Constraints

$$\sum_{j \in F} x_{ij} = 1 \quad \forall i \in C \quad (\text{A.2})$$

$$x_{ij} \leq b_j \quad \forall (i, j) \in C \times F \quad (\text{A.3})$$

We first define sets  $C$  and  $F$  to denote customers and facilities respectively. For our model parameters, we define  $c_{ij}$  to denote the cost of assigning customer  $i$  to facility  $j$ , which is given by the  $l_1$  norm between the two corresponding coordinates. Here we also define  $k_f$  as the cost of opening facility  $f$ , which is given explicitly by  $100 \cdot 3^{-f}$ . For decision variables we let  $x_{ij}$  be a binary variable that is equal to 1 if we assign customer  $i$  to  $j$ . Similarly,  $b_f$  is another binary variable that is equal to 1 if we open facility  $f$ .

We then wish to minimise the cost of the operation. We model the objective function (A.1) as the cost of assigning customer  $i$  to  $j$ , multiplied by the decision

variable. We then add this to the cost of opening the facilities, constructed similarly. The problem is constrained by (A.2), which ensures that every customer is assigned to at least one facility, and (A.3), which ensures that only facilities that are open can have customer assigned to them.

### A.2.2 Capacitated Model

Associated Files: `model(1.2).mod`, `f_loc(1.2).dat`, `c_loc(1.2).dat`,  
`data(1.2).dat`, `run(1.2).run`, `out(1.2).txt`

We then wish to expand the model, by factoring in the demands we generated earlier. We do this first by simply adding two variables.  $d_c$  defines the demand of customer  $c$ , and  $l_f$  defines the capacity limit of facility  $f$ . The parameter  $l_f$  is more specifically given as  $100 \cdot 2^f$ . We must then add one more constraint, namely:

$$\sum_{i \in C} d_i x_{ij} \leq l_j b_j \quad \forall j \in F \quad (\text{A.4})$$

This constraint (A.4) ensures that, summed over all customers assigned to facility  $f$ , the demand must not exceed the capacity of the facility. The binary variables here ensure this constraint holds only for open facilities. Other than this, our model remains unchanged from the previous section

### A.2.3 Balanced Loads

We are now presented with the situation in which the firm wishes to retain some control over the demands faced by each facility. In Section 2.1, we learn that the optimal solution is indeed unbalanced, and there are numerous reasons why we may want to control this, one being to protect against shocks in demand.

For this, we let there be some  $L$  that defines the greatest allowable difference between the smallest demand faced, and the greatest. We define two new positive decision variables,  $u_{min}$  and  $u_{max}$ , to describe these values specifically. Finally, we add 3 more constraints to our current model, which are outlined below:

$$\sum_{i \in C} d_i x_{ij} \geq u_{min} \quad \forall j \in F \quad (\text{A.5})$$

$$\sum_{i \in C} d_i x_{ij} \leq u_{max} \quad \forall j \in F \quad (\text{A.6})$$

$$u_{max} - u_{min} \leq L \quad (\text{A.7})$$

Constraints (A.5) and (A.6) ensure the demands faced do not break the bounds of the limits described by  $u_{min}$  and  $u_{max}$ , and then constraint (A.7) ensures these

---

two limits do not exceed our predefined limit  $L$ . This proposed model forces the largest and smallest demands to adhere the limit  $L$  in any solution, as required.

## Appendix B

# Vehicle Routing Methodology

Contained in this appendix is detailed, technical report of the methodology applied in the vehicle routing scenario.

### B.1 Data Generation

#### B.1.1 Composite Client Generation

Associated Files: `client_generation.Rmd`

As in the previous problem, it was first necessary to generate data to be used in our modelling. In this case, it was necessary to generate 25 composite clients, who we prioritise the delivery service to. We outline the process used below in Algorithm 3, and the implementation can be found in `client_generation.Rmd`:

---

**Algorithm 3:** Composite Client Generation

---

**Result:** Set of composite clients  $C_{comp}$

```
1 while all  $c \in C_{comp}$  are not unique do
2    $C_{comp} \leftarrow 25 \times 1$  vector of random uniform(0,1) variables ;
3   for  $c \in C_{comp}$  do
4      $c \leftarrow 60 \cdot c$  ;
5      $c \leftarrow \text{upper}(c)$  ;
6   end
7 end
```

---

Our process is very similar to that laid out in Algorithm 2, except we have added a loop enclosing the whole process that ensures the list we produce contains entirely unique numbers, otherwise, we would not have 25 composite customers. The other change is multiplying the random variables by 60, since there are 60 potential customers.



## B.2 Modelling

### B.2.1 Base-Line Model

Associated Files: `model(2.1).mod`, `data(2.1).dat`, `locations(2.1).dat`,  
`run(2.1).run`, `out(2.1).txt`

Now we have the set of our clients, we can move to model the problem. In the first situation, we essentially have an instance of the Travelling Salesman Problem. All demand is satisfied by the depot, and the vehicle has no capacity limit, therefore, we aim simply to minimise the travelling cost, which here is given as the  $l_2$  norm between two nodes. A definition of our model is given below, and the implementation can be found in `model(2.1).mod`:

#### Sets

$C_{comp}$  set of Composite Customers

$D$  set of Depots

$V = C_{comp} \cup D$  set of Nodes

#### Parameters

$c_{ij}$   $(i, j) \in V$  cost of travelling from node  $i$  to node  $j$

$N$  The cardinality of set  $V$

#### Variables

$x_{ij} \in \{0, 1\}$   $(i, j) \in V$  Travelling from  $i$  to facility  $j$  on a given route

$u_i \in \mathbb{Z}_+$   $i \in V$  Indication of where node  $i$  is on the tour (MTZ Variable)

#### Objective

$$\min \sum_{i \in V} \sum_{j \neq i \in V} c_{ij} x_{ij} \quad (\text{B.1})$$

#### Constraints

$$\sum_{j \neq i \in V} x_{ij} = 1 \quad \forall i \in V \quad (\text{B.2})$$

$$\sum_{i \neq j \in V} x_{ij} = 1 \quad \forall j \in V \quad (\text{B.3})$$

$$u_i - u_j + Nx_{ij} \leq N - 1 \quad \forall i, j \in V, \text{ where } j \neq i, 1. \quad (\text{B.4})$$

We first define sets  $C_{comp}$  and  $D$  to represent the composite customers generated, and the depot set, i.e  $D = \{d_1\}$ . We then define the union of these two sets to be  $V$ , the set of all nodes. For parameters, we then have  $c_{ij}$  which represents the cost as defined above from travelling from  $i$  to  $j$ , and  $N$  is the number of nodes. We now define two decision variables:  $x_{ij}$  is a binary variable indicating whether a given  $i, j$  connection is on the route; and  $u_i$  is the sub-tour elimination constraint, which forces the nodes to appear in some order on the route. In (B.1) we define the objective function, that is, the cost of travelling from node  $i$  to  $j$  summed over the route. The

constraints (B.2) and (B.3) ensure only one route goes in and out of every node, and (B.4) enforces the ordering of  $u_i$ . It is also important to note that all demand is inherently satisfied here, due to the vehicle's unlimited capacity.

The result of our model will be a route connecting all of our composite clients, minimising the distance cost of the route. We laid out a visual solution in Figure 2.1, but in Table B.2.1, we give the precise routes:

### B.2.2 Multiple Vehicles

**Associated Files:** `model(2.2).mod`, `data(2.2).dat`, `locations(2.2).dat`, `run(2.2).run`, `out(2.2).txt`

We are now presented with the opportunity of multiple vehicles. As before, all vehicles have unlimited capacity, and they all must leave and return to the original depot  $d_1$ . We can expand our previous model to facilitate this, by first adding a new set  $K$ , with cardinality  $k = 3$  to represent the vehicles. Then, we add two more constraints:

$$\sum_{i \in V} x_{d_1, i} = k \tag{B.5}$$

$$\sum_{i \in V} x_{i, d_1} = k \tag{B.6}$$

These constraints (B.5) and (B.6) ensure that the 3 vehicles all start and finish with the depot  $d_1$ . Finally, we relax the constraints (B.2) and (B.3) from the previous formulation to only hold for the composite customer nodes.

Upon solving our problem, we find the same route as in Table B.2.1 and Figure 2.1. The other vehicles are not to be used. This is to be expected, since all vehicles return to the same node, and there are no restrictions on the capacity of the vehicle. We can imagine that if there were capacity restrictions, we would need to limit the number of nodes each vehicle could visit, which can be done by changing  $N$  in (B.4), to reflect this. We find the objective function is the same as in Section B.2.1, since it is the identical route.

Node	Coordinates
c60	(9, 10)
c47	(5, 7)
c55	(0, 9)
c57	(-2, 10)
c54	(-5, 9)
c53	(-6, 9)
c44	(-9, 7)
c42	(0, 5)
c38	(-1, 3)
c30	(-3, 0)
c21	(-3, -2)
c29	(-7, 0)
c16	(-8, -3)
c1	(-7, -10)
c8	(-4, -7)
c13	(-1, -6)
c11	(3, -7)
c4	(10, -10)
c12	(6, -7)
c23	(4, -2)
c24	(6, -2)
c35	(8, 1)
c36	(10, 1)
c40	(8, 4)
d1	(10, 10)

Table B.1: Route Coordinates

## Appendix C

# Internal Depot Transportation with Uncertainty Methodology

Contained in this appendix are the modelling suggestions for the internal operations of the depot. We suggest two formulations, a robust approach, and a non-robust approach. As outlined in Section 2.3, the choice between them depends on the consequences of an infeasible solution, which in this case refers to an electric vehicle that cannot be packed. We still recommend the non-robust formulation, mainly because it offers a more effective maximisation of the expected value of the vehicle. In both of our formulations, we model the problem considering only one vehicle. If implemented these models would be solved individually every time a vehicle is packed.

### C.1 Robust Formulation

The robustness of the formulation comes from the fact that no matter what size of item eventually comes out, the problem will always be feasible. The exact program is defined below:

#### Sets

$I$  set of Items

#### Parameters

$w_i^{max}$   $i \in I$  max weight of item  $i$   
 $v_i^{max}$   $i \in I$  max volume of item  $i$   
 $p_i$   $i \in I$  profit gained from item  $i$   
 $V$  volume capacity of vehicle  
 $W$  weight capacity of vehicle  
 $L$  item capacity of vehicle

#### Variables

$x_i \geq 0$   $i \in I$  quantity of good  $i$  allotted to the vehicle

**Objective**

$$\max \sum_{i \in I} p_i x_i \quad (\text{C.1})$$

**Constraints**

$$\sum_{i \in I} w_i^{max} x_i \leq W \quad (\text{C.2})$$

$$\sum_{i \in I} v_i^{max} x_i \leq V \quad (\text{C.3})$$

$$\sum_{i \in I} x_i \leq L \quad (\text{C.4})$$

We assume we can garner some information about the items  $i$  in set  $I$ . Specifically, the volume and weight of each item  $i$ , and these values are denoted  $v_i^{max}$  and  $w_i^{min}$  respectively. We then define  $p_i$  as the certain profit from each item  $i$ . Then for the vehicle, we have parameters  $W$  for the weight capacity,  $V$  for the volume capacity, and  $L$  for the item capacity. Then for decision variables, we denote positive  $x_i$  for the quantity of good  $i$  allocated for the vehicle. (C.1) We wish to maximise the value of the vehicle, by summing up the profits incurred from each item in the vehicle. In (C.2) and (C.3) we limit the allocation by the maximum weight and volume of each item, this ensures no matter what dimensions of the item come out, there will be a feasible solution. Finally, in (C.4) we limit the number of items we can put in the vehicle.

**C.2 Non-Robust Formulation**

We now outline the second proposed model. Once again we assume we can garner some data about the item, in this case, we want to find the mean and variance in the dimensions of each item  $i$ . This could be done by analysing previous data on each item. With this, we define a model based on these values:

**Sets**

$I$  set of Items

**Parameters**

$w_i$   $i \in I$  expected weight of item  $i$   
 $v_i$   $i \in I$  expected volume of item  $i$   
 $\sigma_{wi}^2$   $i \in I$  variance in weight of item  $i$   
 $\sigma_{vi}^2$   $i \in I$  variance in volume of item  $i$   
 $p_i$   $i \in I$  profit gained from item  $i$   
 $\rho$  Some tolerated variance threshold  
 $V$  volume capacity of vehicle  
 $W$  weight capacity of vehicle  
 $L$  item capacity of vehicle

**Variables**

$x_i \geq 0$   $i \in I$  quantity of good  $i$  allotted to the vehicle

**Objective**

$$\max \sum_{i \in I} p_i x_i \quad (\text{C.1})$$

**Constraints**

$$\sum_{i \in I} w_i x_i \leq W \quad (\text{C.2})$$

$$\sum_{i \in I} v_i x_i \leq V \quad (\text{C.3})$$

$$\sum_{i \in I} x_i \leq L \quad (\text{C.4})$$

$$\sum_{i \in I} x_i (\sigma_{wi}^2 + \sigma_{vi}^2) \leq \rho \quad (\text{C.5})$$

Here we define  $V, W, L$  as before, and we work in the mean and variances of the volumes and weight of each item  $i$ . A key addition to this model is the  $\rho$  value. This can be thought of as a level of risk tolerance.

We define the objective function (C.1) as before and the same with constraints (C.2), (C.3) and (C.4), except we now use the expected value of each dimension instead of the maximum values. We then use  $\rho$  to set some acceptable level of variance of the items in the vehicle. If we were more risk-tolerant, we would use a higher  $\rho$ , and see a more effective maximisation of space. However, we would increase the probability that once the dimensions are realised, the problem will be infeasible. Here we see a trade-off between the accepted level of risk and the quality of the solution.

It is important to consider the level at which we set  $\rho$ . In order to set the right level, we could carry out simulations to see what gives us the best balance

---

between profitability and infeasibility. We could also run Monte-Carlo Markov-Chain simulations to better determine the expected weight and volume, and their variances, of each item.

## Appendix D

# Product Specific Insurance Methodology

Contained in this appendix is a technical review of the simulations of the insurance subsidiary. We focus on the determination of  $P$ , the probability of legal issues, and  $C_{12}$ , an estimate of the capital stock at the end of a 12-month period.

### D.1 Base-Line Model

Associated Files: `baseline.Rmd`

We first wish to simulate operations to determine  $P$ . For this, we run the simulation  $K = 1000$  times and then divide the number of times we encounter issues by 1000.

We now build the simulation model. For this, we must first define two sub-routines for the arrival and departure rates of customers to the service, which are a homogeneous (Algorithm 4) and non-homogeneous (Algorithm 5) Poisson process respectively.

---

**Algorithm 4:** Homogeneous Poisson Process

---

**Data:** Current time  $s$ , process intensity  $\lambda$

**Result:** Next event time  $t$  in Poisson Process

- 1  $U \leftarrow$  Random uniform(0,1) variable;
  - 2  $t \leftarrow s - (\frac{1}{\lambda} \cdot \log(U))$ ;
- 

In Algorithm 5 we use  $\mu(s) = \frac{1}{12+s}$  for our simulation. We also must factor in the rate at which the customers make a claim, which also follows a Homogeneous Poisson Process for each customer. We again use Algorithm 4 but with a different rate, namely  $\frac{\alpha}{12}$ , since our data only gives us a yearly rate.

From here we can define our variables, events, event lists and output variables.



**Algorithm 5:** Non-Homogeneous Poisson Process**Data:** Current time  $s$ , process intensity  $\lambda$ , rate  $\mu(s)$ **Result:** Next event time  $t$  in Non-Homogeneous Poisson Process

---

```

1  $U_1 \leftarrow$  Random uniform(0,1) variable;
2  $t \leftarrow s - (\frac{1}{\lambda} \cdot \log(U))$ ;
3  $U_2 \leftarrow$  Random uniform(0,1) variable;
4 if  $U_2 \geq \frac{\mu(t)}{\lambda}$  then
5   | Go to step 1
6 end

```

---

For variables we let  $t$  be the current time,  $n_A$  be the number of arrivals by time  $t$ . We also let  $cost$  be the current cost accrued through claims, and  $rev$  denotes the revenue generated from monthly subscription. To track this monthly fee, we keep a list  $months$  that tracks what months have elapsed. For the state variable we let  $n_0$  be the number of customers currently in the service.

Now for the event variables, we let  $t_A, t_D$  denote the times after  $t$  where the next arrival and departure occurs. We implement  $t_D$  as a list, since the process at which customers leaved is defined with each customer. It is also required each customer has a unique claim time, and thus we define  $t_C$  as a list, indexed by the customer, that holds the times after  $t$  until the next claim.

Finally, we collect a binary output variable  $B$ , that is 1 if we are below the legal floor  $C$ , and 0 otherwise.

At last, we can define the pseudo-code for the simulation. We first initialise  $t = n_A = 0$ , as well as  $n = 0$ . We then let  $cost = rev = 0$ . We also initialise our  $B$  and  $months$  lists to be empty. Then to begin the simulation, we generate  $t_A$  using Algorithm 4, and set the list  $t_D = t_C = \infty$ , with that as the only item. We enclose the following sub-routines in a while loop, that for runs 12 months, denoted  $T_{total} = 12$ . Each algorithm is executed depending on the values  $t_A, t_D, t_C$ , which are stated in the caption:

**Algorithm 6:**  $t_A = \min(t_A, t_D, t_C, T_{total})$ 


---

```

1  $t \leftarrow t_A$ ;
2  $n_A \leftarrow n_A + 1$ ;
3  $n_0 \leftarrow n_0 + 1$ ;
4 Generate new  $t_A$  with Algorithm 4 ;
5 Generate new  $t_C$  for this customer with Algorithm 4 ;
6 Generate new  $t_D$  for this customer with Algorithm 5 ;

```

---

This first sub-routine (6) is executed if there is first an arrival, lines 1 to 3 set the

current time to the time of the next arrival, and counts the arrival on  $n_A$  and  $n_0$ . We then generate the next arrival time, and the time of the next claim for this arrival on lines 4 and 5. Finally on lines 6 to 8, if this is the first arrival to the service, we generate a new time of departure.

---

**Algorithm 7:** Any  $t_D = \min(t_A, t_D, t_C, T_{total})$

---

```

1  $t \leftarrow t_D$ ;
2  $n_0 \leftarrow n_0 - 1$ ;
3 Set  $t_D$  for this customer to  $\infty$ ;
4 Set  $t_C$  for this customer to  $\infty$ ;

```

---

Algorithm 7 describes the situation in which a departure is the next event. Lines 1 changes the current time to the time of the next departure, and line 2 makes the necessary adjustments to the counter. In line 3 and 4 we set the departure times and claim times for this customer to be  $\infty$ , for they have left the service.

---

**Algorithm 8:** Any  $t_C = \min(t_A, t_D, t_C, T_{total})$

---

```

1  $t \leftarrow t_C$ ;
2 Generate new  $t_C$  for this customer with Algorithm 4;
3  $U_1 \leftarrow \text{Random uniform}(0,1)$  variable ;
4 if  $U_1 \leq 0.6$  then
5    $U_2 \leftarrow \text{Random uniform}(0,1)$  variable ;
6    $X \leftarrow 10 \cdot \text{upper}(U_2)$  ;
7    $\text{payout} \leftarrow 300 \cdot X + 500$  ;
8    $\text{cost} \leftarrow \text{cost} + \text{payout}$  ;
9 end

```

---

Algorithm 8 describes the only other possible situation, where the next event is an insurance claim. This sub-routine executes if and only if there exists a value in  $t_C$  such that it is smallest in  $t_C$ , and smaller than  $t_A$  and  $t_D$ . In line 1 we first set the time to the time of the next claim, and then in line 2 we add a claim to the counter. In line 3 we generate the next claim time for that customer. Lines 4-9 describe the eventuality that there is a claim. We start by generating a random uniform variable, and if it less than 0.6, we calculate the pay out. In line 6 we generate another random uniform variable, and in line 7 we turn this into a random variable between 1 and 10. Finally in lines 8 and 9, we use the given formula to calculate the *payout*, and add it the total running cost.

At each time step, we check if we have entered a new month. We do this by finding the lower integer  $t$ , and if it does not exist in our month list, we must have entered a new month, and we also add it to the list. We then calculate  $M \cdot n_0$  at

that time and add it to  $rev$ . If the month does exist in the list, we do nothing.

Finally, if  $t > T_{total}$ , we stop the simulation, and perform a calculation to determine if our capital holdings are beneath the legal floor, namely  $C_0 + rev - cost$ , where  $C_0$  is our capital stock at the start of the period. If it is, we append a 1 to  $B$ , and 0 otherwise.

As outlined earlier, we run the simulation 1000 times with the parameters  $n_0 = 0, c_0 = 50,000, M = 300, C = 30,000, \lambda = 3, \alpha = 3.5$  and calculate  $P$ , and we came out with a value of 0.401. This is a rather high probability, and as such we carried out further simulations. We experimented with a higher  $M$ , and found that a small increase drastically decreases the  $P$ , as discussed in Section 2.4.

## D.2 Confidence

Associated Files: `confidence.Rmd`

We now move to estimating  $C_{12}$ . We can largely use the same model as in the previous section, however we now use an output variable of  $C_{12}$  at the end of each simulation. Instead of looping 1000 times, we run the loop until we can be confident we have an estimate we are satisfied with, say 3000. We also append a new subroutine at the end of each simulation, this is designed to update the sample mean  $\bar{C}_{12}$  and variance  $S^2$  of  $C_{12}$ . It's process is outlined in Algorithm 9.

---

### Algorithm 9: Update $\bar{C}_{12}$ and $S^2$

---

**Data:** New  $C_{12}$ , Current  $\bar{C}_{12}$ , Current  $S^2$ , Iteration  $k$

- 1 New  $\bar{C}_{12} \leftarrow \bar{C}_{12} + \frac{C_{12} - \bar{C}_{12}}{k+1}$ ;
  - 2 New  $S^2 \leftarrow \left(1 - \frac{1}{k}\right) S_k^2 + (k+1) (\text{New } \bar{C}_{12} - \bar{C}_{12})^2$
- 

Then, when these values satisfy

$$\frac{S}{\sqrt{k}} < \frac{500}{1.645}$$

We record the  $\bar{C}_{12}$  and use this as our estimate for  $C_{12}$ , and record  $S^2$  and  $k$  the first iteration this inequality was satisfied. We use the values 500 since this is the interval to which we would like to be 90% sure we are within, and 1.645 since this is a 5% deviation from the standard normal distribution mean. It is worth noting that we assume the Central Limit Theorem applies here. In our simulation, we attained estimates of  $C_{12} = 32412.96, k = 2059, S = 303.88$ , where we calculated the last value by taking the square root of  $S^2$

Finally, we wish to estimate an interval to which we are 95% sure the true value is within, and for this we can use  $S$  from earlier. We take 1.96, which is a 2.5% deviation

from the standard normal distribution mean, and calculate  $1.96 \cdot S = 1.96 \cdot 303.88 = 595.59$ . So our interval is  $C_{12} \pm 595.59$

### D.3 Variance Reduction

Lastly, we wish to outline specifically a measure to reduce variance in the estimates attained 2.4. We will employ a variance reduction method using a Control Variate. We outlined in the management report that a key source of variability is within the insurance payout formula.