RESOURCE OPTIMISATION IN CONTACT CENTRE OPERATIONS

BDC5101 – Deterministic Operations Research Models

Jonathan Simon Wagner (A0152784X)
Rex Chua (A0206487X)
Muhamad Imran Bin Mohd Yusof (A0206485B)
Nicholas David Gabriele (A0206490J)
Jerome Kuah (A0052238L)
Seah Fang Ying (A0206532N)

List of Tables

Executive Summary

This report provides a practical approach to the problem of resource optimization and, more specifically, manpower optimisation in a contact centre setting. "Practical approach" refers to the fact that objectives and constraints are chosen based on real life business problems consistently arising in modern day contact centres. The methods of analysis used include basic statistical modelling to cater for demand uncertainty as well as linear programming to solve the optimisation problem. The assumption that forms the basis of further analysis is that there are different types of contact centre agents; namely, experienced full-time hires, complemented by on-demand part-time hires. The results of the analysis show that under every assumption made, the model leverages the more expensive full-time hires for those call types with smaller demand while utilizing part-time hires to satisfy the high demand types. While this makes sense from an economical point of view, a practical concern is that deploying experienced full-time hires in this way may reduce the quality of call types not assigned to them. In addition to the recommendation of manpower hire, the analysis also presents the required training of part-time hires, since one hire cannot be trained for all available call types. Given the demand distribution between different types, a contact centre manager can also use these findings for training purposes. One of the critical assumptions made is that calls cannot be abandoned or queued. While this ensures an exceptional level of service quality, customers may be accustomed to a queueing system; hence, further research might focus on combining the practical approach introduced in this report with a queuing simulation that would slightly relax the manpower constraints, especially during high demand periods, but also reduce service quality.

1 Introduction

"It takes months to find a customer … seconds to lose one"

– Vince Lombardi

A contact centre is often the link between an organisation and its customers. Communication can be made through telephone, emails and digital channels such as live chat.

With the proliferation of digital communications, traditional contact channels such as telephony are trending down but still represents the bulk of a contact centre's volume today. While traditional contact channels are trending down, new means of communication are similar in concept and any findings for telephone communication can be readily applied to these emerging channels as well.

Especially in the digital age, customer experience is the key differentiator for businesses trying to satisfy their customers (Wladawsky-Berger, 2018). Having said this, one of the most important aspects of running a functioning and satisfactory contact centre is the staffing aspect. Only with enough call agents, the contact centre can appropriately respond to call demands and answer the questions in a timely and qualified manner.

Managing a contact centre's workforce requires good forecasts of future call volumes for making decisions both in the short run and long run. In the longer term, contact centre managers must decide on strategic decisions such as the capacity of the contact centre and the number of agents needed for each channel or product type while ensuring that the required service quality is delivered to their customers at minimal cost. In the shorter term, daily decisions include scheduling and planning for unexpected surges in contact demand or agent absenteeism. The many complex and interdependent decision-making needed suggest that there are many opportunities for optimisation in managing a contact centre. In this paper, we will concentrate on the staffing and training optimisation for a multi-skilled and multi-product contact centre handling primarily telephone calls.

The next section of this paper describes the overview of the contact centre operations and its key challenges and the specific characteristics as well as assumed data of the contact centre we are optimising for. In section 3, we discuss the objectives, constraints and build the mathematical formulation used in the optimisation. We will discuss the general model and provide sensitivity analysis in section 4 and conclude with an interpretation of our solution.

2 Contact centre operations and challenges

2.1 Performance measures

Typically, a contact centre measures its success based broadly on 2 factors namely cost minimisation and a high overall customer satisfaction. Managing these seemingly conflicting objectives is challenging. Contact centre mangers are often required to find the right balance in agent deployment and avoid over staffing in order to keep service levels healthy. The cost incurred by a contact centre is largely attributed to the cost of its agents (salaries) which varies depending on their skill sets (e.g. the number of product types they are qualified to handle). Other fixed costs, such as rent and equipment, form the remainder of costs. Customer satisfaction is measured on several factors such as call abandonment rate, the percentage of calls that are answered within a particular time frame (for example, 80% within 2 minutes) and the number of repeated calls for a similar nature of enquiry etc.

In this paper, we consider only the salary costs, and confine ourselves to use the call abandonment rate as the measure of service quality provided to the customer in our optimisation solution. This limitation is imposed because information on repeated calls were not available from our assumed contact centre. Also, our assumption of a more stringent case where customers have zero willingness to hold on the line (discussed in section 2.2) meant that percentage of calls answered within a time frame is no longer relevant.

2.2 Contact Demand Management

In most contact centres, each incoming call can be routed into one of many different call types that is determined by the customer's language preference, call topic (or product type) and complexity of call (e.g. call belonging to customers with multiple product relationships) etc. Call agents are assigned to handle each call type based on their product knowledge acquired through training and experience in handling complex cases. When a call arrives, it is routed to an available agent who has been assigned to the call type. If an agent is not available at the point of call, the customer would be placed in a queue of the assigned call type waiting for the next available agent. Here, we assume a contact centre with multiple call types handling different product types and agents have varying skill levels that determine the number of call types they can handle, the number of calls they are able to handle within an hour and their cost (salary). Table 1 and 2 below presents the case in discussion.

Call Type	% of total demand
Product 1	50%
Product 2	20%
Product 3	20%
Product 4	10%

Table 1: Demand distribution based on product type

Agent Type	Skill set	Call handling rate (per hr)	Salary (per hr)	Max number of agents
Experienced Agents	All 4 product types	6.0	$40	30
Temporary Agents	Up to 2 product types	4.5	$10	Unrestricted

Table 2: Introduction to agent types

Calls waiting in queue may be abandoned and it is dependent on the amount of time the customer waited on the line and the willingness of the customer to hold on the line. There is a lack of data (if any) on the willingness of the customer and it is therefore difficult to estimate. Customers who abandon a call may also retry at a later time in the day, inflating the expected demand especially at times when demand is high. In this paper, we assume a more stringent scenario where customers have zero willingness to wait and will abandon the call if it is not answered immediately. Abandonment therefore occurs when the supply of agent is not able to match the demand of calls in a particular period. The maximum tolerable abandonment ratio by the contact centre discussed in this paper is shown in table 3 below.

Maximum allowable abandonment ratio	5%

Table 3: Maximum allowable call abandonment ratio

2.3 Call forecasting

Call forecasting is critical for the planning of resources in a contact centre. Call demand estimates are needed months in advance to allow sufficient time to acquire and train agents. At a micro level, intraday fluctuation in demand also affects the overall service level and is required for optimal deployment and scheduling of agents. Good forecasts of call demand are therefore required before we can find an optimal deployment solution. Here, we will assume that the demand forecasting had been provided by the contact centre's analytics department based on historical trend analysis. We will further discuss uncertainties in demand and the sensitivity analysis in section 4. The demand forecast provided for our problem is in table 4 below.

Time interval	Call volume	Standard Deviation
8.00am - 9.00am	170	31.6

9.00am - 10.00am	250	23.8
10.00am - 11.00am	310	39.6
11.00am - 12.00pm	290	15.8
12.00pm - 1.00pm	220	41.6
1.00pm - 2.00pm	270	50.3
2.00pm - 3.00pm	310	35.1
3.00pm - 4.00pm	380	70
4.00pm - 5.00pm	300	27.1
Total	2500	

Table 4: Demand distribution based on time throughout day

2.4 Human resource issues in contact centres

High turn-around of employees is frequent in contact centres. Contact centre managers often also have to plan for high absenteeism in the daily agent supply and prevent agent burnouts when call volumes remain high for a sustained period of time. While managers seek to maximise the productive hours of their agents, it is equally essential to allocate sufficient down time to improve agent retention and reduce absenteeism. Planning for uncertainties in agent availability is therefore equally crucial as uncertainties in demand forecasts. These will be discussed in section 4 of this paper. Details of the expected absenteeism and allocated down time for our contact centre is shown in table 5 below.

Allocated down time per day	5%
Expected absenteeism rate per day	20%

Table 5: Absenteeism overview

2.5 Literature Review

The practice of optimizing manpower in a contact centre environment is not a new concept. Many researchers and practitioners have been looking at this topic in the past since it is touching a very tangible and prominent practical problem. The overall topics in contact centre optimization range from performance analysis (Koole, 2004) over the optimization of call routing (Otsuka, 2005) to manpower optimization which is being looked at in this paper. Within manpower optimization, Aksin, Armony and Mehrotra were already summarising in 2007 what needed to be done in the field of operations research to achieve operational excellence within a contact centre. They were already stressing the point at that time that within a centre there will be different skill levels that needed to be accounted for. The following research however mainly focused on the queuing aspect (Chromy, Kavacky, Baronak, 2017) while ignoring the nature of skill-based staffing and hiring as well as training.

This paper is trying to contribute to the current research of contact centre optimization by coming from a more practical approach, having authors who have worked in and managed contact centres before. With a focus on skill-based routing and skill-based demand satisfaction while accounting for the fact of specific manpower downtimes as well as average absenteeism, the optimization problem described in this paper is closer to reality than previous research.

3 Problem Formulation

In this section, we will start off with a basic formulation of the problem to derive the minimum cost of hiring temporary call agents to meet its daily operations. This objective function is derived by considering the following constraints:

i. **Manpower Constraints**. As highlighted in section 2, the contact centre has two types of call agents who differ in their capacities and types of calls they can handle. While the number of temporary agents can vary, the contact centre's experienced agents is assumed to be fixed at 30, limited by the centre's headcount requirements. Experienced agents are required to fulfil a minimum amount of time handling calls and are not allowed to be not assigned any calls totally.

ii. **Call Demand Constraints**. Although we do have information on the proportions of the call volumes for the 4 product types, the aggregated call volume is normally distributed. This creates uncertainties in operational requirements and outcomes which we will have to take into consideration.

3.1 Objective Function

Based on the objectives and constraints highlighted above, we will be using linear programming to model the setting of our contact centre operations. We formulated the following objective function given that the objective of the contact centre is to minimise the human resource cost of hiring temporary call agents:

$$\min \sum_{i=1}^{7} \sum_{j=1}^{4} c_{ij} x_{ij}$$

$c_{i,j}$ denotes the hourly cost to hire a call agent, and $x_{i,j}$ denotes the manhours required for agent type i to handle calls for product type j. The figures indicate that the contact centre has 7 types of agents where agent denoted with $i = 1$ are the experienced agents while the agents denoted with $i = 2\ to\ 7$ refer to the temporary agents who can handle a permutation of two out of the 4 product types.

3.2 Constraints

As mentioned above, the objective function is subjected to several constraints such as manpower and the varying call demands. The formulation of the constraints is provided in the sections below.

3.2.1 Manpower constraint

In Section 2, it was highlighted that the contact centre has 30 experienced agents. However, these agents only spend 70% of their time handling calls. The rest of their time is spent on ad-hoc projects, training, administrative tasks, meetings etc. This constraint is denoted by the following equation:

$$\sum_{j=1}^{4} x_{1,j} \quad = \quad 30 \times 0.7$$

where agent type $i = 1$ refers to the experienced agents, and $x_{1,j}$ denotes the manhours required for the experienced agents to handle calls for product type j. This works out for our contact centre to have 21 full-time equivalent manpower for experienced agents.

3.2.2 Call demand constraint

The forecasted call demand provided in Table 4 of Section 2.3 is assumed to take on a normal distribution and will be met by the experienced agents who can handle 6 calls per hour and temporary agents who can handle 4.5 calls per hour. We also know the distribution of the calls by the 4 product types (Table 1, Section 2.2). Given that our decisions variable is the number of temporary call agents manhours to acquire for the contact centre to meet the call demands highlighted, we derive the call demand constraints as follows:

$$\begin{array}{rcll} \sum_{i=1}^{7} v_{i,1}x_{i,1} & \geq & 0.5D_k & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} v_{i,2}x_{i,2} & \geq & 0.2D_k & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} v_{i,3}x_{i,3} & \geq & 0.2D_k & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} v_{i,4}x_{i,4} & \geq & 0.1D_k & \forall\, k = 1,2,\ldots,9 \\ x_{ij} & \geq & 0 & \forall\, i,j \end{array}$$

where $v_{i,1}$ denotes the number of calls per hour that agent type i can handle for product type 1, $x_{i,1}$ denotes the manhours required for agent type i and call for product type 1 and D_k denotes the call demand at the hour interval k.

3.2.3 Total infrastructure constraint

In addition to the already existing headcount constraint for experienced agents, an additional constraint was introduced to limit the total number of agents consisting of experienced and temporary agents. Due to limited infrastructure on site, the limit for total agents was chosen to be 200. During the results discussion, it can be observed that this constraint is never binding under any case but was introduced for a more meaningful sensitivity analysis and also in order to provide a more accurate representation of the real world setting.

$$\sum_{i=1}^{7} \sum_{j=1}^{4} x_{ij} \quad \leq \quad 200$$

3.2.4 Others

Our model above only illustrates the basic requirements of minimising cost of hiring for the contact centre while meeting the daily operation requirements. However, there could be uncertainties such as absenteeism or call burnout that may impact the objective that we have yet to include to in our model. These uncertainties will be highlighted in Section 4 where we cover the sensitivity analysis of our model.

3.3 Mathematical Formulation

The full linear programming model to meet the objective of the contact centre is as follows:

$$\min \sum_{i=1}^{7}\sum_{j=1}^{4} c_{ij}x_{ij} \quad s.t. \begin{cases} \sum_{j=1}^{4} x_{1,j} & = & 30 \times 0.7 & \\ \sum_{i=1}^{7}\sum_{j=1}^{4} x_{ij} & \leq & 200 & \\ \sum_{i=1}^{7} v_{i,1}x_{i,1} & \geq & 0.5D_k & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} v_{i,2}x_{i,2} & \geq & 0.2D_k & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} v_{i,3}x_{i,3} & \geq & 0.2D_k & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} v_{i,4}x_{i,4} & \geq & 0.1D_k & \forall\, k = 1,2,\ldots,9 \\ x_{ij} & \geq & 0 & \forall\, i,j \end{cases}$$

Based on this baseline model, we derived the manpower requirements to be:

Agent – call type combination	**Man-hours**
Experienced Agent handling call type 2	2.7
Experienced Agent handling call type 3	12.2
Experienced Agent handling call type 4	6.1
Temporary Agent handling call type 1	40.6
Temporary Agent handling call type 2	12.6

Table 6: Baseline Model Man-hours

This translates to:

Agent – call training combination	**Number of Agents required**
Experienced agents answering call types 2, 3 and 4	21
Temporary agents answering call types 1 and 2	54 (rounded up from 53.2)

Table 7: Baseline Model Manpower Demand

The average cost of operating the contact centre over a 9-hour work day would be $16,129.24 when using the above baseline model.

4 Managing Risks in meeting service levels

In this section, we expand upon our baseline model to cater for two important issues mentioned above in Section 2.2 and 2.4, namely absenteeism and the abandonment ratio.

4.1 Planning for Absenteeism

To cater for a 20% absenteeism and 5% downtime amongst both our experienced and temporary agents, we introduced a scalar multiplier of 0.76 into our call demand constraints. This is because at any particular hour, our total manpower would be reduced by 20% due to agents being absent and of the remaining manpower, 5% would be on breaks or other activities. Therefore, the mathematical formulation of our model would be as follows:

$$\min \sum_{i=1}^{7}\sum_{j=1}^{4} c_{ij}x_{ij} \quad s.t. \begin{cases} \sum_{j=1}^{4} x_{1,j} & = & 30 \times 0.7 & \\ \sum_{i=1}^{7} 0.76 v_{i,1}x_{i,1} & \geq & 0.5D_k & \forall\, k = 1,2,\dots,9 \\ \sum_{i=1}^{7} 0.76 v_{i,2}x_{i,2} & \geq & 0.2D_k & \forall\, k = 1,2,\dots,9 \\ \sum_{i=1}^{7} 0.76 v_{i,3}x_{i,3} & \geq & 0.2D_k & \forall\, k = 1,2,\dots,9 \\ \sum_{i=1}^{7} 0.76 v_{i,4}x_{i,4} & \geq & 0.1D_k & \forall\, k = 1,2,\dots,9 \\ x_{ij} & \geq & 0 & \forall\, i,j \end{cases}$$

Naturally, this results in a reallocation of resources to meet call demand and the hiring of additional temporary agents. The manpower requirements in this case would be as follows:

Agent – call type combination	Man-hours
Experienced Agent handling call type 3	12.98
Experienced Agent handling call type 4	8.02
Temporary Agent handling call type 1	53.47
Temporary Agent handling call type 2	21.39
Temporary Agent handling call type 3	4.08

Table 8: Man-hours accounting for Absenteeism

This translates to:

Agent – call training combination	Number of Agents required
Experienced agents answering call types 3 and 4	21
Temporary agents answering call types 1 and 2	49 (rounded up from 48.1)
Temporary agents answering call types 1 and 3	31 (rounded up from 30.8)

Table 9: Manpower demand accounting for Absenteeism

Temporary Agent man-hours for call type 1 is split into two and paired with call types 2 and 3 evenly to form the training pairs (1,2 and 1,3). Due to the additional manpower requirements to cater to the potential absenteeism, the average manpower cost of operating the contact centre per workday would now be $17,905.35, which is 11.01% higher than our baseline model.

4.2 Planning for A Worst-Case Scenario

Contact centre managers are also tasked with planning for worst-case scenarios, where incoming call demand is consistently high while facing the similar issues of absenteeism amongst staff. To model the case of high demand, we take the 95th percentile of our normally distributed demand for each hour. This is in line with our maximum allowable abandonment ratio of 5%. Therefore, our call demand for the hourly intervals is as follows:

Time interval	**High Call volume** (rounded to 3 decimal places)
8.00am - 9.00am	221.977
9.00am - 10.00am	289.148
10.00am - 11.00am	375.136
11.00am - 12.00pm	315.989
12.00pm - 1.00pm	288.426
1.00pm - 2.00pm	352.736
2.00pm - 3.00pm	367.734
3.00pm - 4.00pm	495.140
4.00pm - 5.00pm	344.576

Table 10: Peak demand per hour

Therefore, we modify the formulation of our model to the following:

$$\min \sum_{i=1}^{7}\sum_{j=1}^{4} c_{ij}x_{ij} \quad s.t. \begin{cases} \sum_{j=1}^{4} x_{1,j} & = & 30 \times 0.7 & \\ \sum_{i=1}^{7} 0.76 v_{i,1} x_{i,1} & \geq & 0.5 D_k^H & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} 0.76 v_{i,2} x_{i,2} & \geq & 0.2 D_k^H & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} 0.76 v_{i,3} x_{i,3} & \geq & 0.2 D_k^H & \forall\, k = 1,2,\ldots,9 \\ \sum_{i=1}^{7} 0.76 v_{i,4} x_{i,4} & \geq & 0.1 D_k^H & \forall\, k = 1,2,\ldots,9 \\ x_{ij} & \geq & 0 & \forall\, i,j \end{cases}$$

Therefore, we derived the manpower requirements to be:

Agent – call type combination	**Man-hours**
Experienced Agent handling call type 3	10.14
Experienced Agent handling call type 4	10.86
Temporary Agent handling call type 1	72.39
Temporary Agent handling call type 2	28.96
Temporary Agent handling call type 3	15.43

Table 11: Peak demand Man-hours

This translates to:

Agent – call training combination	**Number of Agents required**
Experienced agents answering call types 3 and 4	21
Temporary agents answering call types 1 and 2	66 (rounded up from 65.1)
Temporary agents answering call types 1 and 3	52 (rounded up from 51.6)

Table 12: Peak demand Manpower demand

Temporary Agent man-hours for call type 1 is split into 2 and paired with call types 2 and 3 evenly to form the training pairs (1,2 and 1,3). Due to the additional manpower requirements to cater to high call volume and the average absenteeism, the average manpower cost of operating the contact centre per workday would now be $21,309.99, which is 33.12% higher than our baseline model.

4.3 Simulating over a 30-day period

Lastly, to ensure that this level of manpower is meeting our requirement of having less than 5% of calls abandoned, we simulate a month of contact centre operations. We find that on average over a 30-day period, only 0.055% of calls would be dropped with this level of manpower. This falls well within the pre-defined 5% benchmark of calls that are allowed to be abandoned. On the daily level, there are some observations where the calls abandoned rate reaches close to 1% but at no time does it go beyond 5%. As the percentage of missed calls is significantly below the 5% service level, no further robustness tests have been performed. This clearly shows the level of risk minimization taken in the worst-case approach.

For comparison purposes, the same 30-day simulation was performed using the extended baseline model accounting for absenteeism and breaks. With this model, 3.97% of calls were abandoned over a 30 day period. Due to this number being closer to the maximum abandonment ratio, a simulation over 100 months was performed and the results show that the real missed call rate lies between 5-6%. This would therefore not meet the service levels imposed by management.

5 Sensitivity Analysis

Constraint Name	Shadow Price	Constraint Value	Slack	Upper Range	Lower Range
experienced agent	26.67	21.00	0.00	32.57	10.86
Max headcount capacity	0.00	200.00	62.22	1.00e+100	137.78
type 1 calls	2.92	247.57	0.00	460.37	0.00
type 2 calls	2.92	99.03	0.00	311.82	0.00

type 3 calls	2.92	99.03	0.00	311.82	46.24
type 4 calls	2.92	49.51	0.00	95.76	0.00

Table 13: Sensitivity Analysis

The sensitivity analysis table above allows us to understand how a change in demand would impact the overall cost minimisation objective. A unit increase in either of the call type demand would increase the overall cost by $2.92. The cost of $2.92 has to be taken with caution though. Due to the rounding used to arrive at the optimal amount of agents to hire, a slight increase in the number of incoming calls does not necessarily increase the cost immediately. Since the linear programming model was chosen to be not an integer based model in order to account for man-hours, the shadow price might not be reflecting the true nature of the problem. The shadow price and optimal solution remains constant where call demand does not exceed 461 for call type 1, 312 for call types 2 and 3, and 96 for call type 4. This can be explained by the rational that part-time agents are allowed to be hired with only minimal constraints. Also, the cost of part-time hires stays constant regardless of the number of hires. The only constraint limiting the amount of part-time hires is represented by the headcount capacity constraint. Under the given optimal level of agents, the centre can hire an additional 62 agents should there be a change in forecasted demand and more agents are required. Because of this constraint, the upper range of all call types is limited. As can be seen from Table 13, the only call type with a positive lower range is call type 3. This can be explained by the fact that only call type 3 has a combination of full-time and part-time agents answering the calls. With an increase in demand for this particular call type, the composition of agent types answering this specific call type will shift and hence change the shadow price. A similar phenomenon can be observed for the upper range of call type 4 where currently only full-time agents are answering these incoming calls. With an increase of more than 96, the agent composition would change from only full-time agents to either a mix of full-time and part-time or only part-time and therefore the shadow price as well as optimal solution would be different.

6 Conclusion

The aim of this paper was to provide a practical approach to solving the optimisation problem in a contact centre setting. Throughout the paper, different concepts were discussed that are arising in today's contact centres. These include accounting for average absenteeism, break times in between calls as well as following a risk management approach to see how many times over a given month, the optimal solution does not meet the previously stipulated service level. In addition to providing the contact centre management with the optimal level of hiring, the created model also serves as a training planning tool. Temporary agents can only be trained in limited call types and it is therefore crucial to have the right training plan to account for the demand in each call type. For these reasons, three models were created, namely the baseline model not accounting for any absenteeism, the extended model including a 20% absenteeism as well as 5% break time as well as a worst-case model taking only the maximum values of the demand normal distribution. The latter worst-case model is by far the most risk-averse model which is shown by the 30-day simulation where only 0.055% of all calls were dropped. With the model only accounting for absenteeism and breaks, the same 30-day simulation arrives at a call abandonment rate of 3.97%, which is still below the previously stipulated service level of not more than 5% abandoned calls. However, when testing for robustness and running the same model for 100 times, the abandoned call rate moves closer to 6% which indicates a miss of the service level. The model choice therefore needs to take into account the trade-off between cost minimization and meeting of the stipulated service level. This is usually a high-level case by case decision for individual contact centres.

The results of all models clearly show that with the number of full-time hires employed in the call distribution process being fixed due to quality constraints, the full-time agents are mostly leveraged within the lower demand call types while more flexible part-time hires are being utilised for high demand call types. There is only one call type where a mix of full-time and part-time agents is present. This distribution might create a quality problem in the long run since full-time agents are not part of every call type. The call types with only part-time agents should be monitored more closely if the quality of answers given and therefore customer satisfaction lies significantly below the other call type levels. If this is the case, an additional constraint should be introduced which distributes full-time agents evenly throughout all call types.

Appendix 1: Code

[1]:
```
from gurobipy import *
import numpy as np
import pandas as pd
```

1 Baseline Model

[2]:
```
def model_setup():

    m = Model("Callcenter")

    # Create variables
    x = m.addVars(M, N, name="x")

    # Set objective
    m.setObjective( quicksum(cost[i,j]*x[i,j] for i in range(M) for j in
range(N)), GRB.MINIMIZE)

    # Experienced Manpower Constraint(only 70% of the 30 headcount for
experienced staff is
    # used for call center operations)
    m.addConstrs(quicksum(x[i,j] for i in range (0,1) for j in range(N)) == 21
for i in range(N))

    # Maximum capacity at the contact centre constraint
    m.addConstr(quicksum(x[i,j] for i in range(M) for j in range(N)) <= 200, "Max
headcount capacity")

    # Hourly constraints
    m.addConstrs(( quicksum(cph[i,0]*x[i,0] for i in range(M)) >= 0.5 * demand[j]
for j in range(9)), "type 1 calls")
    m.addConstrs(( quicksum(cph[i,1]*x[i,1] for i in range(M)) >= 0.2 * demand[j]
for j in range(9)), "type 2 calls")
    m.addConstrs(( quicksum(cph[i,2]*x[i,2] for i in range(M)) >= 0.2 * demand[j]
for j in range(9)), "type 3 calls")
    m.addConstrs(( quicksum(cph[i,3]*x[i,3] for i in range(M)) >= 0.1 * demand[j]
for j in range(9)), "type 4 calls")
```

```
    m.setParam( 'OutputFlag', False)

    return m
```

[3]:
```
### Parameters Set-up ###

# cost per employee - experienced agents cost 40 per hour, temp agents cost 10
per hour
cost = np.array([[40, 40, 40, 40],
                [10, 10, 10000, 10000],
                [10, 10000, 10, 10000],
                [10, 10000, 10000, 10],
                [10000, 10, 10, 10000],
                [10000, 10, 10000, 10],
                [10000, 10000, 10, 10]])

# demand - normally distributed with the following mean and variance
mean_d = np.array([170, 250, 310, 290, 220, 270, 310, 380, 300])

cov_d = np.array([[998.56, 0, 0, 0, 0, 0, 0, 0, 0],
                  [0, 566.44, 0, 0, 0, 0, 0, 0, 0],
                  [0, 0, 1568.16, 0, 0, 0, 0, 0, 0],
                  [0, 0, 0, 249.64, 0, 0, 0, 0, 0],
                  [0, 0, 0, 0, 1730.56, 0, 0, 0, 0],
                  [0, 0, 0, 0, 0, 2530.09, 0, 0, 0],
                  [0, 0, 0, 0, 0, 0, 1232.01, 0, 0],
                  [0, 0, 0, 0, 0, 0, 0, 4900, 0],
                  [0, 0, 0, 0, 0, 0, 0, 0, 734.41]])

# calls per hour that can be taken by a certain officer i,j
# officers that cannot take a particular type of call can take zero calls of that
type
cph = np.array([[6, 6, 6, 6],
                [4.5, 4.5, 0, 0],
                [4.5, 0, 4.5, 0],
                [4.5, 0, 0, 4.5],
                [0, 4.5, 4.5, 0],
                [0, 4.5, 0, 4.5],
                [0, 0, 4.5, 4.5]])

M, N = cost.shape

Sample_Size = 1000

cc_cost = np.zeros(Sample_Size)
```

```
np.random.seed(1988)
for i in range(Sample_Size):
    demand = np.minimum(np.maximum(np.random.multivariate_normal(mean_d, cov_d),
0), 500)
    m = model_setup()
    m.optimize()
    cc_cost[i] = m.objVal

print(demand)

avg_cc_cost = np.average(cc_cost)
```

```
Academic license - for non-commercial use only
[164.58413285 230.94202189 348.81424637 307.0440906  248.26055662
 330.58057554 292.86675365 365.76323557 329.60008708]
```

[4]:
```
# average call center cost plus the 9 experienced agents that handle other
administrative matters
print('Average manpower cost of operating call center per hour: ', avg_cc_cost +
9 * 40)
print('Average manpower cost of operating call center per day (9 hour workday):
', (avg_cc_cost + 9 * 40)* 9)

for v in m.getVars():
    if v.x > 0:
        #print(v.VarName[2], v.VarName[4], v.x)
        print("\nNumber of officers of type %g required to answer calls of type
%g: %g" % (int(v.VarName[2])+1, int(v.VarName[4])+1, v.x))
```

```
Average manpower cost of operating call center per hour:  1792.137941776398
Average manpower cost of operating call center per day (9 hour workday):
16129.241475987583

Number of officers of type 1 required to answer calls of type 2: 2.71184

Number of officers of type 1 required to answer calls of type 3: 12.1921

Number of officers of type 1 required to answer calls of type 4: 6.09605

Number of officers of type 3 required to answer calls of type 1: 40.6404

Number of officers of type 5 required to answer calls of type 2: 12.6404
```

2 Baseline model with 20% absenteeism & 5% downtime

[5]:
```
def model_setup_absent():

    a = Model("Callcenter_absent")

    # Create variables
    x = a.addVars(M, N, name="x")

    # Set objective
    a.setObjective(quicksum(cost[i,j]*x[i,j] for i in range(M) for j in
range(N)), GRB.MINIMIZE)

    # Experienced Manpower Constraint
    a.addConstrs((quicksum(x[i,j] for i in range (1) for j in range(N)) == 21 for
i in range(N)), "expmanpower")

    # Maximum capacity at the contact centre constraint
    a.addConstr(quicksum(x[i,j] for i in range(M) for j in range(N)) <= 200, "Max
headcount capacity")

    # Hourly constraints
    a.addConstrs(( quicksum(cph[i,0]*x[i,0] for i in range(M)) >= 0.5 * demand[j]
for j in range(9)), "type 1 calls")
    a.addConstrs(( quicksum(cph[i,1]*x[i,1] for i in range(M)) >= 0.2 * demand[j]
for j in range(9)), "type 2 calls")
    a.addConstrs(( quicksum(cph[i,2]*x[i,2] for i in range(M)) >= 0.2 * demand[j]
for j in range(9)), "type 3 calls")
    a.addConstrs(( quicksum(cph[i,3]*x[i,3] for i in range(M)) >= 0.1 * demand[j]
for j in range(9)), "type 4 calls")

    a.setParam( 'OutputFlag', False)

    return a
```

[6]:
```
# for simplicity, scalar for absenteeism and downtime is multiplied into the call
handling rate

cph = np.array([[6, 6, 6, 6],
                [4.5, 4.5, 0, 0],
                [4.5, 0, 4.5, 0],
                [4.5, 0, 0, 4.5],
                [0, 4.5, 4.5, 0],
                [0, 4.5, 0, 4.5],
                [0, 0, 4.5, 4.5]])

#absenteeism rate
```

```
absent = 0.20

#downtime rate
downtime = 0.05
cph = cph * (1-absent) * (1-downtime)

a = model_setup_absent()
a.optimize()
```

[7]:
```
print('Average manpower cost of operating call center per hour: ', a.objVal + 9 *
40)
print('Average manpower cost of operating call center per day (9 hour workday):
', (a.objVal + 9 * 40)* 9)

for v in a.getVars():
    if v.x > 0:
        print("\nNumber of officers of type %g required to answer calls of type
%g: %g" % (int(v.VarName[2])+1, int(v.VarName[4])+1, v.x))
```

```
Average manpower cost of operating call center per hour:  1989.4831449341111
Average manpower cost of operating call center per day (9 hour workday):
17905.348304407

Number of officers of type 1 required to answer calls of type 3: 12.9789

Number of officers of type 1 required to answer calls of type 4: 8.02112

Number of officers of type 3 required to answer calls of type 1: 53.4742

Number of officers of type 5 required to answer calls of type 2: 21.3897

Number of officers of type 5 required to answer calls of type 3: 4.08449
```

[8]:
```
### Percentage of Missed calls over a 30 day period ###

#number of days
D = 30

mean_d = np.array([170, 250, 310, 290, 220, 270, 310, 380, 300])

cov_d = np.array([[998.56, 0, 0, 0, 0, 0, 0, 0, 0],
                  [0, 566.44, 0, 0, 0, 0, 0, 0, 0],
                  [0, 0, 1568.16, 0, 0, 0, 0, 0, 0],
                  [0, 0, 0, 249.64, 0, 0, 0, 0, 0],
                  [0, 0, 0, 0, 1730.56, 0, 0, 0, 0],
                  [0, 0, 0, 0, 0, 2530.09, 0, 0, 0],
                  [0, 0, 0, 0, 0, 0, 1232.01, 0, 0],
```

```
                   [0, 0, 0, 0, 0, 0, 0, 4900, 0],
                   [0, 0, 0, 0, 0, 0, 0, 0, 734.41]])

missedcalls1 = np.zeros(D)
totaldemand1 = np.zeros(D)
missedcalls2 = np.zeros(D)
totaldemand2 = np.zeros(D)
missedcalls3 = np.zeros(D)
totaldemand3 = np.zeros(D)
missedcalls4 = np.zeros(D)
totaldemand4 = np.zeros(D)
missedcallstotal = np.zeros(D)

#type 1 total capacity
t1 = (1-absent)*(1-downtime)*round(53.4742 * 4.5, 0)
#type 2 total capacity
t2 = (1-absent)*(1-downtime)*round(21.3897 * 4.5, 0)
#type 3 total capacity
t3 = (1-absent)*(1-downtime)*round(12.9789 * 6 + 4.08449 * 4.5, 0)
#type 4 total capacity
t4 = (1-absent)*(1-downtime)*round(8.02112 * 6, 0)

np.random.seed(1234)

for d in range(D):
    demand = np.minimum(np.maximum(np.random.multivariate_normal(mean_d, cov_d),
0), 500)
    demand1 = demand * 0.5
    demand2 = demand * 0.2
    demand3 = demand * 0.2
    demand4 = demand * 0.1
    for i in range(9):
        totaldemand1[d] += demand1[i]
        totaldemand2[d] += demand2[i]
        totaldemand3[d] += demand3[i]
        totaldemand4[d] += demand4[i]
        #number of missed calls of type 1
        if demand1[i] - t1 > 0:
            missedcalls1[d] += demand1[i] - t1
        #number of missed calls of type 2
        if demand2[i] - t2 > 0:
            missedcalls2[d] += demand2[i] - t2
        #number of missed calls of type 3
        if demand3[i] - t3 > 0:
            missedcalls3[d] += demand3[i] - t3
        #number of missed calls of type 4
        if demand4[i] - t4 > 0:
```

```
            missedcalls4[d] += demand4[i] - t4

percentagemissed1 = (missedcalls1 / totaldemand1) * 100
percentagemissed2 = (missedcalls2 / totaldemand2) * 100
percentagemissed3 = (missedcalls3 / totaldemand3) * 100
percentagemissed4 = (missedcalls4 / totaldemand4) * 100

print('Percentage of Type 1 calls missed over 30 days:')
print()
print(percentagemissed1)
print('\nPercentage of Type 2 calls missed over 30 days:')
print()
print(percentagemissed2)
print('\nPercentage of Type 3 calls missed over 30 days:')
print()
print(percentagemissed3)
print('\nPercentage of Type 4 calls missed over 30 days:')
print()
print(percentagemissed4)
print('\nTotal')
percentagetotal = percentagemissed1 + percentagemissed2 + percentagemissed3 +
percentagemissed4
print(percentagetotal)
print('\nAverage percentage of calls dropped over a 30 day period:', round(np.
 ↪average(percentagetotal), 4))
```

```
Percentage of Type 1 calls missed over 30 days:

[1.84402122 0.         4.17994963 0.         2.37161205 0.
 0.         0.96667183 3.29691175 0.         0.79231896 0.
 3.97807736 0.         0.         0.         3.5321066  0.40177394
 0.         0.         0.         0.         1.40034191 0.67405373
 2.49418067 0.         2.03396838 0.         0.         1.04976168]

Percentage of Type 2 calls missed over 30 days:

[1.90406588 0.         4.23979758 0.         2.4902474  0.
 0.         1.05193953 3.41322847 0.         0.85481972 0.
 4.038909   0.         0.         0.         3.58799851 0.46465907
 0.         0.         0.         0.         1.46319244 0.77604363
 2.55151562 0.         2.09216027 0.         0.         1.10662316]

Percentage of Type 3 calls missed over 30 days:

[1.90406588 0.         4.23979758 0.         2.4902474  0.
 0.         1.05193953 3.41322847 0.         0.85481972 0.
 4.038909   0.         0.         0.         3.58799851 0.46465907
```

```
 0.         0.         0.         0.         1.46319244 0.77604363
 2.55151562 0.         2.09216027 0.         0.         1.10662316]

Percentage of Type 4 calls missed over 30 days:

[1.90406588 0.         4.23979758 0.         2.4902474  0.
 0.         1.05193953 3.41322847 0.         0.85481972 0.
 4.038909   0.         0.         0.         3.58799851 0.46465907
 0.         0.         0.         0.         1.46319244 0.77604363
 2.55151562 0.         2.09216027 0.         0.         1.10662316]

Total
[ 7.55621886  0.          16.89934238  0.           9.84235424  0.
  0.           4.12249042 13.53659717  0.           3.35677813  0.
 16.09480437  0.           0.           0.          14.29610212  1.79575114
  0.           0.           0.           0.           5.78991922  3.00218462
 10.14872753  0.           8.31044918  0.           0.           4.36963116]

Average percentage of calls dropped over a 30 day period: 3.9707
```

[9]:

```python
### Percentage of Missed calls over a 30 day x 12 month period ###

#number of days and number of months
D = 30
MN = 100

mean_d = np.array([170, 250, 310, 290, 220, 270, 310, 380, 300])

cov_d = np.array([[998.56, 0, 0, 0, 0, 0, 0, 0, 0],
                  [0, 566.44, 0, 0, 0, 0, 0, 0, 0],
                  [0, 0, 1568.16, 0, 0, 0, 0, 0, 0],
                  [0, 0, 0, 249.64, 0, 0, 0, 0, 0],
                  [0, 0, 0, 0, 1730.56, 0, 0, 0, 0],
                  [0, 0, 0, 0, 0, 2530.09, 0, 0, 0],
                  [0, 0, 0, 0, 0, 0, 1232.01, 0, 0],
                  [0, 0, 0, 0, 0, 0, 0, 4900, 0],
                  [0, 0, 0, 0, 0, 0, 0, 0, 734.41]])

missedcalls1 = np.zeros(D)
totaldemand1 = np.zeros(D)
missedcalls2 = np.zeros(D)
totaldemand2 = np.zeros(D)
missedcalls3 = np.zeros(D)
totaldemand3 = np.zeros(D)
missedcalls4 = np.zeros(D)
totaldemand4 = np.zeros(D)
missedcallstotal = np.zeros(D)
```

```
#type 1 total capacity
t1 = (1-absent)*(1-downtime)*round(53.4742 * 4.5, 0)
#type 2 total capacity
t2 = (1-absent)*(1-downtime)*round(21.3897 * 4.5, 0)
#type 3 total capacity
t3 = (1-absent)*(1-downtime)*round(12.9789 * 6 + 4.08449 * 4.5, 0)
#type 4 total capacity
t4 = (1-absent)*(1-downtime)*round(8.02112 * 6, 0)

np.random.seed(2013)

for m in range(MN):
    for d in range(D):
        demand = np.minimum(np.maximum(np.random.multivariate_normal(mean_d,
cov_d), 0), 500)
        demand1 = demand * 0.5
        demand2 = demand * 0.2
        demand3 = demand * 0.2
        demand4 = demand * 0.1
        for i in range(9):
            totaldemand1[d] += demand1[i]
            totaldemand2[d] += demand2[i]
            totaldemand3[d] += demand3[i]
            totaldemand4[d] += demand4[i]
            #number of missed calls of type 1
            if demand1[i] - t1 > 0:
                missedcalls1[d] += demand1[i] - t1
            #number of missed calls of type 2
            if demand2[i] - t2 > 0:
                missedcalls2[d] += demand2[i] - t2
            #number of missed calls of type 3
            if demand3[i] - t3 > 0:
                missedcalls3[d] += demand3[i] - t3
            #number of missed calls of type 4
            if demand4[i] - t4 > 0:
                missedcalls4[d] += demand4[i] - t4

percentagemissed1 = (missedcalls1 / totaldemand1) * 100
percentagemissed2 = (missedcalls2 / totaldemand2) * 100
percentagemissed3 = (missedcalls3 / totaldemand3) * 100
percentagemissed4 = (missedcalls4 / totaldemand4) * 100

print('Percentage of Type 1 calls missed over 30 days:')
print()
print(percentagemissed1)
print('\nPercentage of Type 2 calls missed over 30 days:')
```

```
print()
print(percentagemissed2)
print('\nPercentage of Type 3 calls missed over 30 days:')
print()
print(percentagemissed3)
print('\nPercentage of Type 4 calls missed over 30 days:')
print()
print(percentagemissed4)
print('\nTotal')
percentagetotal = percentagemissed1 + percentagemissed2 + percentagemissed3 +
percentagemissed4
print(percentagetotal)
print('\nAverage percentage of calls dropped over a 30 day period:', round(np.
 ↪average(percentagetotal), 4))
```

```
Percentage of Type 1 calls missed over 30 days:

[1.13489728 1.36319859 1.26888818 1.48031376 1.02827724 1.39150924
 1.47539672 1.50647904 1.14362106 1.24596106 1.55557842 1.51562446
 1.3319472  1.22926431 1.31907711 1.37668006 1.56949843 1.37538412
 1.54104408 1.32764669 1.42604725 1.50630602 1.25627308 1.57707345
 1.47643866 1.2636965  1.69072358 1.60655466 1.29551704 1.58664851]

Percentage of Type 2 calls missed over 30 days:

[1.17563797 1.404935   1.31765321 1.5245462  1.06775463 1.43946492
 1.52198351 1.55169203 1.18402697 1.28507641 1.59760132 1.55994179
 1.37878508 1.26911498 1.36051081 1.41632317 1.61625154 1.42322987
 1.58949519 1.37372844 1.47192302 1.55509785 1.29783863 1.62254023
 1.52129197 1.31232538 1.74375088 1.65825745 1.34025759 1.63191965]

Percentage of Type 3 calls missed over 30 days:

[1.17563797 1.404935   1.31765321 1.5245462  1.06775463 1.43946492
 1.52198351 1.55169203 1.18402697 1.28507641 1.59760132 1.55994179
 1.37878508 1.26911498 1.36051081 1.41632317 1.61625154 1.42322987
 1.58949519 1.37372844 1.47192302 1.55509785 1.29783863 1.62254023
 1.52129197 1.31232538 1.74375088 1.65825745 1.34025759 1.63191965]

Percentage of Type 4 calls missed over 30 days:

[1.17563797 1.404935   1.31765321 1.5245462  1.06775463 1.43946492
 1.52198351 1.55169203 1.18402697 1.28507641 1.59760132 1.55994179
 1.37878508 1.26911498 1.36051081 1.41632317 1.61625154 1.42322987
 1.58949519 1.37372844 1.47192302 1.55509785 1.29783863 1.62254023
 1.52129197 1.31232538 1.74375088 1.65825745 1.34025759 1.63191965]
```

```
Total
[4.66181118 5.57800361 5.2218478  6.05395235 4.23154113 5.70990399
 6.04134726 6.16155515 4.69570196 5.10119027 6.34838237 6.19544981
 5.46830245 5.03660923 5.40060953 5.62564958 6.41825306 5.64507373
 6.30952966 5.44883199 5.84181631 6.17159959 5.14978899 6.44469414
 6.04031455 5.20067265 6.92197622 6.58132701 5.3162898  6.48240745]

Average percentage of calls dropped over a 30 day period: 5.7168
```

3 Worst-case scanerio - meeting 95% of potential call volume with 20% absenteeism & 5% downtime

[10]:
```
import scipy.stats
#set percentile of calls that must be taken
p = 0.95

mean_d = np.array([170, 250, 310, 290, 220, 270, 310, 380, 300])
sd_d = np.array([31.6, 23.8, 39.6, 15.8, 41.6, 50.3, 35.1, 70, 27.1])

# for simplicity, scalar for absenteeism and downtime is multiplied into the call
handling rate
cph = np.array([[6, 6, 6, 6],
                [4.5, 4.5, 0, 0],
                [4.5, 0, 4.5, 0],
                [4.5, 0, 0, 4.5],
                [0, 4.5, 4.5, 0],
                [0, 4.5, 0, 4.5],
                [0, 0, 4.5, 4.5]])

#absenteeism rate
absent = 0.20

#downtime rate
downtime = 0.05
cph = cph * (1-absent) * (1-downtime)

highdemand = np.zeros(9)

for i in range(9):
    hdemand = np.maximum(scipy.stats.norm.ppf(p,mean_d[i],sd_d[i]),0)
    highdemand[i] = hdemand

print(highdemand)
highdemand = np.amax(highdemand)
print(highdemand)
```

```
[221.97737461 289.14751632 375.13620363 315.98868731 288.42591088
```

```
 352.73613744 367.73436231 495.13975389 344.57553329]
495.13975388660305
```

[11]:
```
def model_setup_high_demand():

    n = Model("Callcenter_highdemand")

    # Create variables
    x = n.addVars(M, N, name="x")

    # Set objective
    n.setObjective( quicksum(cost[i,j]*x[i,j] for i in range(M) for j in
range(N)), GRB.MINIMIZE)

    # Experienced Manpower Constraint
    n.addConstr((quicksum(x[i,j] for i in range(0,1) for j in range(N)) == 21),
"experienced manpower" )

    # Maximum capacity at the contact centre constraint
    n.addConstr(quicksum(x[i,j] for i in range(M) for j in range(N)) <= 200, "Max
headcount capacity")

    # Hourly constraints
    # Model must satify requirements for the hour of highest demand to be optimal
for the entire day
    n.addConstr(( quicksum(cph[i,0]*x[i,0] for i in range(M)) >= 0.5 *
highdemand), "type 1 calls")
    n.addConstr(( quicksum(cph[i,1]*x[i,1] for i in range(M)) >= 0.2 *
highdemand), "type 2 calls")
    n.addConstr(( quicksum(cph[i,2]*x[i,2] for i in range(M)) >= 0.2 *
highdemand), "type 3 calls")
    n.addConstr(( quicksum(cph[i,3]*x[i,3] for i in range(M)) >= 0.1 *
highdemand), "type 4 calls")

    n.setParam( 'OutputFlag', False)

    return n
```

[12]:
```
n = model_setup_high_demand()

n.optimize()

print('Average manpower cost of operating call center per hour: ', n.objVal + 9 *
40)
print('Average manpower cost of operating call center per day (9 hour workday):
', (n.objVal + 9 * 40)* 9)
```

```
for v in n.getVars():
    if v.x > 0:
        print("\nNumber of officers of type %g required to answer calls of type
%g: %g" % (int(v.VarName[2])+1, int(v.VarName[4])+1, v.x))
```

```
Average manpower cost of operating call center per hour:  2367.7770581479617
Average manpower cost of operating call center per day (9 hour workday):
21309.993523331657

Number of officers of type 1 required to answer calls of type 3: 10.1417

Number of officers of type 1 required to answer calls of type 4: 10.8583

Number of officers of type 3 required to answer calls of type 1: 72.3889

Number of officers of type 5 required to answer calls of type 2: 28.9555

Number of officers of type 5 required to answer calls of type 3: 15.4333
```

4 Sensitivity Analysis

[13]:
```
Sen_constr = pd.DataFrame(columns=['Constraint Name', 'Shadow Price', 'Constraint
Value', 'Slack', 'Upper Range' ,'Lower Range'])

print("\nConstraints Sensitivity Information:")

for d in n.getConstrs():
    #print(d.ConstrName, d.Pi, d.Slack, d.SARHSUP, d.SARHSLow)
    Sen_constr = Sen_constr.append(pd.Series([d.ConstrName, d.Pi, d.RHS, d.Slack,
d.SARHSUP, d.SARHSLow], index=Sen_constr.columns ), ignore_index=True)

Sen_constr.set_index('Constraint Name',inplace=True)
Sen_constr
```

```
Constraints Sensitivity Information:
```

[13]:

Constraint Name	Shadow Price	Constraint Value	Slack	\
experienced manpower	26.666667	21.000000	0.000000	
Max headcount capacity	0.000000	200.000000	62.222294	
type 1 calls	2.923977	247.569877	0.000000	
type 2 calls	2.923977	99.027951	0.000000	
type 3 calls	2.923977	99.027951	0.000000	
type 4 calls	2.923977	49.513975	0.000000	

```
                          Upper Range  Lower Range
Constraint Name
experienced manpower     3.257498e+01    10.858328
Max headcount capacity  1.000000e+100   137.777706
type 1 calls             4.603701e+02    -0.000000
type 2 calls             3.118282e+02    -0.000000
type 3 calls             3.118282e+02    46.246025
type 4 calls             9.576000e+01    -0.000000
```

5 Number of missed calls with manpower at high demand levels

[14]:
```
#numberofdays
D = 30

mean_d = np.array([170, 250, 310, 290, 220, 270, 310, 380, 300])

cov_d = np.array([[998.56, 0, 0, 0, 0, 0, 0, 0, 0],
                  [0, 566.44, 0, 0, 0, 0, 0, 0, 0],
                  [0, 0, 1568.16, 0, 0, 0, 0, 0, 0],
                  [0, 0, 0, 249.64, 0, 0, 0, 0, 0],
                  [0, 0, 0, 0, 1730.56, 0, 0, 0, 0],
                  [0, 0, 0, 0, 0, 2530.09, 0, 0, 0],
                  [0, 0, 0, 0, 0, 0, 1232.01, 0, 0],
                  [0, 0, 0, 0, 0, 0, 0, 4900, 0],
                  [0, 0, 0, 0, 0, 0, 0, 0, 734.41]])

missedcalls1 = np.zeros(D)
totaldemand1 = np.zeros(D)
missedcalls2 = np.zeros(D)
totaldemand2 = np.zeros(D)
missedcalls3 = np.zeros(D)
totaldemand3 = np.zeros(D)
missedcalls4 = np.zeros(D)
totaldemand4 = np.zeros(D)
missedcallstotal = np.zeros(D)

#type 1 total capacity
t1 = (1-absent)*(1-downtime)*round(72.3889 * 4.5, 0)
#type 2 total capacity
t2 = (1-absent)*(1-downtime)*round(28.9555 * 4.5, 0)
#type 3 total capacity
t3 = (1-absent)*(1-downtime)*round(10.1417 * 6 + 15.4333 * 4.5, 0)
#type 4 total capacity
t4 = (1-absent)*(1-downtime)*round(10.8583 * 6, 0)
```

```
np.random.seed(1988)
for d in range(D):
    demand = np.minimum(np.maximum(np.random.multivariate_normal(mean_d, cov_d),
0), 500)
    demand1 = demand * 0.5
    demand2 = demand * 0.2
    demand3 = demand * 0.2
    demand4 = demand * 0.1
    for i in range(9):
        totaldemand1[d] += demand1[i]
        totaldemand2[d] += demand2[i]
        totaldemand3[d] += demand3[i]
        totaldemand4[d] += demand4[i]
        #number of missed calls of type 1
        if demand1[i] - t1 > 0:
            missedcalls1[d] += demand1[i] - t1
        #number of missed calls of type 2
        if demand2[i] - t2 > 0:
            missedcalls2[d] += demand2[i] - t2
        #number of missed calls of type 3
        if demand3[i] - t3 > 0:
            missedcalls3[d] += demand3[i] - t3
        #number of missed calls of type 4
        if demand4[i] - t4 > 0:
            missedcalls4[d] += demand4[i] - t4

percentagemissed1 = (missedcalls1 / totaldemand1) * 100
percentagemissed2 = (missedcalls2 / totaldemand2) * 100
percentagemissed3 = (missedcalls3 / totaldemand3) * 100
percentagemissed4 = (missedcalls4 / totaldemand4) * 100

print('Percentage of Type 1 calls missed over 30 days:')
print()
print(percentagemissed1)
print('\nPercentage of Type 2 calls missed over 30 days:')
print()
print(percentagemissed2)
print('\nPercentage of Type 3 calls missed over 30 days:')
print()
print(percentagemissed3)
print('\nPercentage of Type 4 calls missed over 30 days:')
print()
print(percentagemissed4)
print('\nTotal')
percentagetotal = percentagemissed1 + percentagemissed2 + percentagemissed3 +
percentagemissed4
print(percentagetotal)
```

```
print('\nAverage percentage of calls dropped over a 30 day period:', round(np.
    ↪average(percentagetotal), 4))
```

```
Percentage of Type 1 calls missed over 30 days:

[0.         0.         0.         0.         0.         0.16928633
 0.         0.         0.         0.         0.         0.15934544
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.        ]

Percentage of Type 2 calls missed over 30 days:

[0.         0.         0.         0.         0.         0.22672277
 0.         0.         0.         0.         0.         0.21340906
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.        ]

Percentage of Type 3 calls missed over 30 days:

[0.         0.         0.         0.         0.         0.22672277
 0.         0.         0.         0.         0.         0.21340906
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.        ]

Percentage of Type 4 calls missed over 30 days:

[0.         0.         0.         0.         0.         0.22672277
 0.         0.         0.         0.         0.         0.21340906
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.        ]

Total
[0.         0.         0.         0.         0.         0.84945463
 0.         0.         0.         0.         0.         0.79957263
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.
 0.         0.         0.         0.         0.         0.        ]

Average percentage of calls dropped over a 30 day period: 0.055
```

References

Aksin, Z., Armony, M., Mehrotra, V. (2007). The Modern Call Center: A Multi-Disciplinary Perspective on Operations Management Research. *Production and Operations Management Society. Vol. 16, No. 6, November-December 2007, pp. 665–688*

Chromy, E., Kavacky, M., Baronak, I. (2017). Call center optimization — Step by step. 119-123. 10.1109/TSP.2017.8075949.

Koole, G. (2004). Performance analysis and optimization in customer contact centers. *First International Conference on the Quantitative Evaluation of Systems, 2004. QEST 2004. Proceedings.* IEEE, 2004.

Otsuka, K. (2005). Optimization of routing operation in contact center server. *U.S. Patent Application No. 11/124,242.*

Wladawsky-Berger, I. (2018). Customer Experience Is the Key Competitive Differentiator in the Digital Age. *The Wall Street Journal.* Retrieved from: https://blogs.wsj.com/cio/2018/04/20/customer-experience-is-the-key-competitive-differentiator-in-the-digital-age/