Project GI Joe

Final Report

Team: Luis de la Garza, Grace Dennis, Ellen Erpenbeck,
Jonny Glazier, Adam Jalali, & Amy Yi
Advisor: Professor Nohadani
Client: NorthShore University HealthSystem Gastroenterology (GI) Units
IEMS 393-2
3/9/17

Table of Contents

Executive Summary **3**

Problem Statement **4**

Objective and Project Strategy **4**

Research **4**

Data Processing and Analysis **5**

Model Formulation **5**

Results **8**

Limitations **9**

Recommendations **10**

Implementation Plan 10

Conclusion **10**

References **11**

Appendix **13**

A. Optimization Program (Julia) 13

B. Inverse Matrix Calculation (R) 15

C. Data transformation (MATLAB) 16

D. Finding Uncertainty Factors (AMPL) 24

Executive Summary

Our project was to create a financially efficient staffing model for NorthShore University Health System's Gastroenterology (GI) Units. This model addresses fluctuations in patient volume. After analyzing the data provided to us by NorthShore, we were able to create a model that accounts for predictable variation. This model minimizes cost by finding the optimal staffing complement of part-time and full-time hires. This model was designed around the Evanston location, however lessons learned from the model can be extended to all six NorthShore locations. This will allow the labs to allocate funds elsewhere in order to improve upon other areas of patient care. If our model is effective, it will have the potential to extend to other areas beyond the gastroenterology labs.

In this project, we worked heavily with Karen Ilag, the director of all six gastroenterology labs within NorthShore and Chris Cheney, an administrative fellow for NorthShore. The president of NorthShore Skokie Hospital, Kristen Murtos, was also available as a resource. They provided us with data that included employee salary information, procedure descriptions and lengths, and appointment data including check-in and check-out times. After transforming this data, we were able to use it in our robust linear optimization program.

There were a variety of lessons learned from the output of the models. First and foremost, NorthShore should use more part-time staff. This type of staff is more flexible than full-time staff in terms of overall hours worked in a week, and the increased usage of the part-time staff will cut NorthShore's staffing costs. Next, NorthShore should examine how shifts are scheduled, particularly, when they start. The model assumes that all shifts are perfectly scheduled around demand. NorthShore should consider scheduling appointments more tightly around their shifts and having certain part-time employees start shifts in hours of the day where demand starts to rise.

One of the potential complications of this recommendation is that NorthShore will have to acquire and maintain additional part-time staff. This will require a recruiting effort to find, hire and train these new staff members. Similarly, it could be difficult to indefinitely keep these employees in a part-time status. These types of employees may be less committed to their future at NorthShore and this may result in a revolving door of part-time staff members, which could cause inefficiencies for NorthShore.

Problem Statement

NorthShore's GI Unit operates with a variety of employees, who work on various schedules. Northshore lacks a developed methodology for staffing each type of employee on any given day, specifically with regards to the nurses and GI lab technicians. NorthShore has cited experiencing both cost and time inefficiencies as a result of the current way they staff.

Objective and Project Strategy

Our objective is to create a streamlined staffing model that optimizes the use of the staff and minimizes costs. We intend to be able to staff the full-time and part-time resource nurses, the full-time and part-time lab technicians, and the additional resource staff. The staffing model will staff by the hour, taking into consideration the month of the year, the day of the week, and the location of the facility.

Research

When doing research and literature reviews on current effective hospital staffing models, we came across several applicable ideas. After an initial search, we found that staffing accounts for roughly 50% of a hospital's costs. Thus, minimizing the costs related to staffing the GI unit will substantially reduce overall costs. When hospitals staff through predictive analytics, it has been shown that it can reduce labor costs by 4-7% with 97% accurate predictions from 30 days out.[4]

The use of analytics can greatly reduce the a hospital's costs. Thus, optimizing for the most efficient staffing model is incredibly important when it comes to cutting costs. However, one study from the *International Journal for Quality in Healthcare* found that the quality of care is highly correlated to hospital staffing and organization, meaning that the number of nurses and physicians caring for a patient is highly correlated with the quality of the care.[1] This study further proved that nurses provided lower quality of care when understaffed. Based on the findings of this research and Northshore's commitment to a high quality of care, we decided to ensure that our model will have the propensity to over-staff nurses as opposed to under-staff them in the GI unit, by including a buffer.

The final relevant research finding related to the use of the Newsvendor Model for staffing in the healthcare space. The Newsvendor Model is a mathematical model in operations management, typically used to determine optimal inventory levels.[3] It can, however, also be applied and implemented for effective and efficiency hospital staffing. This has come to be known as the "Nursevendor Model." Past models have utilized this optimization model to build a predictive model that takes into account both the cost parameters as well as the uncertainty in patient demand.[5]

Data Processing and Analysis

We received data from NorthShore regarding the hourly cost (wage plus fringe cost for benefits) to the hospital per employee by employee type, patient check-in and appointment data from January 2012 to December 2016, and a sample of current employee schedules.

Using this data, we decided to transform the data in such a way that it would represent demand data by hospital. Once we determined that we were going to staff based on the number of occupied rooms per hour, we were able to use the patient appointment data to approximate the number of rooms used in any given hour. NorthShore currently schedules in 40 minute appointment timeslots, regardless of the procedure. Using the appointment time as the metric, we counted the aggregated appointments in 5 minute intervals. Using this data, we created a program to calculate the number of rooms utilized in MATLAB (see Appendix C).

The next phase of the data processing involved taking seasonality into consideration. NorthShore experiences high demand volume towards the end of every year due to expiration of insurance deductibles. As a result, many patients will chose to carry out with procedures which they would otherwise put off until a later date. We wanted to be able to measure how variable each month's demand was from the average. Therefore, we created monthly factors by which to multiply the average hourly rooms occupied. The monthly factors are calculated by dividing the aggregate number of rooms occupied in that month by the average number of rooms occupied across all of the months. Only times when the GI unit was open at that specific location were taken into account. This is particularly important at the end of the year - November and December - when NorthShore opens for additional hours on the weekends to meet increased demand. We had found that there are statistically significant differences between the demand of days of the week, thus we staffed by days of the week. Northshore has found that its patients prefer to have appointments sandwiched around the weekends, making mondays and fridays the peak days of the weak.

Model Formulation

Based on our research findings from the Newsvendor Model applications to healthcare staffing, we thought it would be useful to begin by creating a predictive model to forecast patient demand in the GI Unit using the patient appointment information from 2012-2016. We initially began by breaking this data down by hour of the day, day of the week, month, and year. We then looked at the data in a time series using Excel, Tableau, and R. We expected to see significant seasonality in the data, with high demand periods at the end of the year. After an initial analysis of the data, we found no trend in the data. This made forecasting the data based on a stationary time series difficult because it was not representative of the expected demand in the GI Unit throughout the year. Ultimately, due to the nature of the staffing in the GI Unit at NorthShore, we decided to develop a staffing model utilizing robust linear optimization that accounts for uncertainty in demand, as opposed to using time-series forecasting to create a predictive model. Seeing as NorthShore schedules their own appointments, it would be unnecessary to do a computationally intensive predictive model.

Our first pass attempt of the linear optimization model is as follows. We formulated a data independent model minimizing the costs of the staff, both full-time and part-time.

Decision Variables:

P_i: # of part-time workers started in hour i

F_i: # of full-time workers started in hour i

Parameters:

d_i: # of workers needed in hour i

c_j: Cost of worker j ($j = P, F$)

λ_P: Effect of one part-time employee

λ_F: Effect of one full-time employee

min $\sum_{i=1}^{n} c_P P_i + c_f F_i$

s.t. $\sum_{i=1}^{n} \lambda_P P_i + \lambda_F F_i \geq d_i$

This model assumes that both full-time and part-time staff work a set number of hours in any given shift. Further, the costs for each employee are represented by the their respective salaries. The demand parameter was assumed to be a fixed value of workers needed in a given hour that was to be determined from our analysis of the data provided by NorthShore. This model was designed to provide a foundation for our data dependent model that would incorporate demand uncertainty along with limitations from available rooms at the GI Unit.

We then formulated the following data dependent model to address the uncertainty within the data, as well as create a model that can be used indefinitely and updated as additional data points are recorded. This model will use all previous observations of room demand and use them to forecast all future observations of a given room and month with a maximum uncertainty.

Sets:

$j =$ {1, 2, 3}, employee type where the numbers correspond to full-time nurse, part-time nurse, and resource nurse, respectively

$k =$ {1, 2, 3, 4, 5, 6}, room number

Decision Variables:

$x_{j,k}$ = Staffing vector in which each row corresponds to the number of employee type j recommended for each room k

Parameters:

u_k = Uncertainty of room k

Σ = Covariance matrix of b_k

ρ^2 = Tuning parameter for uncertainty based on Chi-squared distribution (recommendation based on $\chi^2_{.900,\,6} = 2.204$)

c_j = Cost of employee type j

$A_{j,k}$ = Staffing efficacy matrix for each employee type, j, corresponding to number of rooms used, k

r_k = Room to nurse ratio for room k

b_k = Utilization (hours) of room k for each day of the week in each month for each given year

Step 1 - Finding uncertainty vector:

$$\max \quad u_k \rightarrow u_k^* \qquad \forall\ k \tag{1}$$

$$\text{s.t} \quad u_k^T \Sigma^{-1} u_k \leq \rho^2 \qquad \forall\ k \tag{2}$$

Step 2 - Solving robust linear optimization model:

$$\min \quad \sum_{j=1}^{3} \sum_{k=1}^{6} c_j^T x_{j,k} \tag{3}$$

$$\text{s.t} \quad \sum_{j=1}^{3} A_{j,k} x_{j,k} \geq r_k (b_k + u_k^*) \quad \forall\ k \tag{4}$$

Our final model as shown above utilizes the theory of robust optimization with ellipsoidal uncertainty.[2] The first step in solving our optimal staffing model is to obtain the uncertainty vector. The first 12 constraints are used to solve for this uncertainty set for each room k. The idea behind this model is to maximize the uncertainty of the hours utilized for each room, while ensuring the uncertainty remains less than or equal to the constraint of ρ^2, our tuning parameter as determined by the Chi-Squared distribution. The ρ^2 value of 2.204 ensures that we are 90% confident with 6 degrees of freedom (for the 6 rooms being staffed). Maximizing the uncertainty vector for each room ensures that our linear program staffing model is as robust as possible, accounting for worst case scenarios.

In the robust linear optimization model, our variable $x_{j,k}$, which represents the number of each type of nurse to be staffed, will be our ultimate staffing recommendations. $x_{j,k}$ is calculated for each day of the week in any given month. Given that our primary objective is to minimize the staffing costs for the GI unit, our objective function is to minimize the sum of the cost of each worker multiplied by the number of workers of type j. The constraint to this objective (14) ensures that the for each type of worker, enough workers are staffed to accommodate the room utilization (demand), while also taking into account the efficacy of each worker and the required room to nurse ratio. The uncertainty vector for each room, obtained in Step 1, is added to the room utilization to account for fluctuations in demand within the constraints of the tuning parameter, as explained above.

In order to implement this model, we used a variety of programming languages, each with unique capabilities that enabled us to complete certain tasks in the processing of our model. After transforming the data, as described above, we calculated the inverse matrix by manipulating the data in R (see Appendix B). We also used AMPL to calculate the uncertainty factors for each day and month (see Appendix D). Both of these were subsequently used in our final model, which was written and solved in the optimization programming language, Julia (see Appendix A).

Results

The results of our staffing model give the optimal complement of full-time and part-time nurses for each day, Monday to Friday, in any given month. A matrix of our results is shown below:

	Mon	**Tues**	**Wed**	**Thurs**	**Fri**
January	(5, 9)	(6, 3)	(4, 4)	(7, 3)	(5, 3)
February	(7, 3)	(6, 2)	(5, 3)	(7, 3)	(5, 4)
March	(10, 0)	(7, 1)	(5, 3)	(7, 3)	(7, 2)
April	(6, 4)	(9, 2)	(5, 3)	(6, 3)	(5, 3)
May	(4, 4)	(9, 1)	(6, 2)	(5, 4)	(7, 1)
June	(9, 1)	(8, 2)	(6, 2)	(6, 2)	(6, 3)
July	(9, 2)	(9, 2)	(6, 4)	(8, 2)	(6, 2)
August	(6, 3)	(6, 5)	(5, 3)	(6, 3)	(6, 2)
September	(7, 3)	(8, 2)	(6, 2)	(8, 3)	(7, 2)
October	(10, 1)	(6, 4)	(5, 3)	(7, 3)	(6, 2)
November	(10, 0)	(9, 1)	(6, 2)	(7, 2)	(5, 3)
December	(7, 3)	(7, 3)	(5, 3)	(6, 3)	(5, 3)

Our model allows NorthShore to staff indefinitely in the future, erasing much of the complexity of forecasting future staffing. Additional data points can be added to the model as time progresses, increasing the accuracy of the predictions. NorthShore can also adjust the tuning parameter according to their risk tolerance and what works best for each location, based on experience.

Limitations

One of the major limitations of our model is consideration and allocation of the resource staff. Specifically, resource staff are additional nurses that have more flexible schedules and also may be required to move locations throughout the week or even day, as needed. Our linear program only considers a single-location at a time, so the flexibility of the resource staff is not taken into account.

A second limitation regards the staffing of employees on the weekend. Due to the variable hours of operation throughout the year depending on the seasonality of the demand, it is difficult to determine optimal staffing patterns. For example, our client has noted that in months towards the end of the year, particularly October through December, demand for GI procedures is high, so the clinic will open for extended hours on the weekend. However, the additional hours are determined on a rule of thumb basis. Because of the lack of methodology in determining extended hours, our model was unable to accommodate this weekend fluctuations. Thus, at this time, we have staffed for the year on Monday through Friday only.

Our model also only accommodates nurses and not the GI technicians, but a similar takeaway we found was that part-time should be more utilized. The same procedure we used to find the nurse staffing could be used to find the optimal number of full-time and part-time GI technicians. In this scenario, we would not have to consider resource staff since that only applies to nurses.

Finally, the current grid, which will be sent to NorthShore, doesn’t create a complete, smooth staff. Another cost optimization program could be overlaid on the final results, which would eliminate discrepancies between the number of full-time and part-time staff throughout the different days of the week. This would allow NorthShore to implement exact numbers of staff at these particular times.

Recommendations

As stated, we understand that the results from our staffing model are not a perfectly realistic staffing schedule that could be implemented immediately due to the discrepancies between the number of full-time and part-time staff working throughout a particular week. However, there is significant value produced from these results. Our model found ratios of full-time and part-time staff that included more part-time nurses than NorthShore currently has staffed. Therefore, we recommend that NorthShore explores utilizing more part-time nurses.

We also recommend that NorthShore reconsiders when shifts start for part-time employees. The current staffing method for part-time employees has individuals working from morning to lunchtime or lunchtime until the end of the work day. This methodology may create an opportunity for inefficiencies. For example, if the demand peaks in the late morning and extends to the early afternoon, the part-time staff should be scheduled to reflect this rise in patient demand. Therefore, we recommend that NorthShore explores changing start times for part-time shifts to better accommodate for patient demand, as opposed to continuing with the methodology they currently use which does not provide much flexibility.

Implementation Plan

The first step in implementing this staffing recommendation is recruiting and hiring part-time employees. Currently, NorthShore almost exclusively has full-time nursing staff and may be required to fire or relocate certain full-time staff to make room for the new part-time employees. This hiring process may require a certain amount of training, which NorthShore should account for when they are bringing on any new hires. Another important aspect of utilizing this recommendation is changing the tuning parameter. It is yet to be seen how conservative or liberal our estimates are in reality, thus as NorthShore is able to gain valuable experience with our staffing recommendations, they will be able to make adjustments as needed, whether they be heuristics or done through the tuning parameter. A final consideration revolves around the utilization of resource staff. Currently, the only recommendation for resource staff is to use them to smooth the differences in part-time and full-time staff in our recommended model. Realistically, NorthShore will have to do this in hand with past heuristics to find the optimal complement of resource staff at each location. This will not be of the utmost concern, as this type of staff is scheduled the day before and is able to be sent anywhere. As a result, NorthShore will both know the exact amount of appointments per location and be able to staff resource accordingly.

A significant part of this implementation relies on the organizational readiness of NorthShore. This shift to a less permanent type of staff is a deviation from the current state of affairs. This may lead to less commitment to NorthShore and result in a revolving door of nursing staff. The recruiting efforts of NorthShore must be strong and they must be able to maintain a consistent level of part time staff. This makes the possibility of promotion from part-time to full-time slim, as it is disadvantageous to lose part-time staff members.

Conclusion

Overall, our team was able to address NorthShore's lack of a methodology for staffing in the GI unit by proposing a staffing schedule that optimizes for the perfect complement of full-time to part-time staff. We solved for this optimal staffing schedule that minimizes cost, using robust optimization to account for a maximum uncertainty in the room usage demand. The ellipsoidal uncertainty method used ensures that our model accounts for any uncertainty up to a certain tuning parameter, which can be changed as the model increases in accuracy and more data is collected. Prior to creating the model, we were also able to transform and process the data from patient appointment information to a demand metric (# of room hours used per day). While our model has some limitations due to increased complexities of the demand, the results we obtained are certainly applicable and valuable to the NorthShore GI Unit.

References

[1] Aiken, L. H. "Hospital Staffing, Organization, and Quality of Care: Cross-national Findings." International Journal for Quality in Health Care 14.1 (2002): 5-13. ScienceDirect. Web. 4 Jan. 2017.

[2] Bertsimas, Dimitris, David B. Brown, and Constantine Caramanis. “Theory and Applications of Robust Optimization.” *SIAM Review* 53.3 (2011): 464-501. Print.

[3] Green, Linda V., Sergei Savin, and Nicos Savva. “‘Nursevendor’ Problem”: Personnel Staffing in the Presence of Endogenous Absenteeism.” 6 Jan. 2011. Web. 28 Feb. 2017.

[4] Schouten, Pieter, General Manager, Healthcare, Opera Solutions. "Better Patient Forecasts and Schedule Optimization Improve Patient Care and Curb Staffing Costs." *Better Patient Forecasts and Schedule Optimization Improve Patient Care and Curb Staffing Costs*. N.p., n.d. Web. 04 Jan. 2017

[5] Shi, Pengyi, Mabel C. Chou, J. G. Dai, Ding Ding, and Joe Sim. "Models and Insights for Hospital Inpatient Operations: Time-Dependent ED Boarding Time." *Management Science: INFORMS*. N.p., 22 Apr. 2015. Web. 10 Jan. 2017.

Appendix

A. Optimization Program (Julia)

```
using JuMP

using GLPKMathProgInterface

#MODEL CONSTRUCTION
#------------------

sfLpModel = Model(solver=GLPKSolverMIP())

#INPUT DATA
#----------

c = [44.7; 44.7; 33]

A= [ 8;4;2; 0;0;0; 0;0;0; 0;0;0; 0;0;0; 0;0;0;
     0;0;0; 8;4;2; 0;0;0; 0;0;0; 0;0;0; 0;0;0;
     0;0;0; 0;0;0; 8;4;2; 0;0;0; 0;0;0; 0;0;0;
     0;0;0; 0;0;0; 0;0;0; 8;4;2; 0;0;0; 0;0;0;
     0;0;0; 0;0;0; 0;0;0; 0;0;0; 8;4;2; 0;0;0;
     0;0;0; 0;0;0; 0;0;0; 0;0;0; 0;0;0; 8;4;2
    ]

d = [7.99;        6.34;     4.57;     2.5;      .75;      0]
u = [.01; 1.41; 2.05; 2.58; 2.9; 0]
tr=1.66

n=6 # m = number of rows of A, n = number of columns of A

#VARIABLES
#---------

@defVar(sfLpModel, x[1:18] >= 0, Int) # Models x >=0

#CONSTRAINTS
#-----------

for i=1:n # for all rows do the following

  @addConstraint(sfLpModel, 8*x[3*i-2]+4*x[3*i-1]+2*x[3*i] >= tr*(d[i*1]+u[i*1]))

end # end of the for loop

# Number of available staff
```

```
f=12
p=12
r=0

for i=1:n
@addConstraint(sfLpModel, x[3*i-2] <=f)
@addConstraint(sfLpModel, x[3*i-1] <=p)
@addConstraint(sfLpModel, x[3*i] <=r)
end
#OBJECTIVE
#---------

@setObjective(sfLpModel, Min,
(x[1]+x[4]+x[7]+x[10]+x[13]+x[16])*c[1]+
(x[2]+x[5]+x[8]+x[11]+x[14]+x[17])*c[2]+
(x[3]+x[6]+x[9]+x[12]+x[15]+x[18])*c[3]) # minimize c'x



#THE MODEL IN A HUMAN-READABLE FORMAT
#------------------------------------

println("The optimization problem to be solved is:")
print(sfLpModel) # Shows the model constructed in a human-readable form

@time begin
status = solve(sfLpModel) # solves the model
end
#SOLVE IT AND DISPLAY THE RESULTS
#--------------------------------

y=x[1]+x[4]+x[7]+x[10]+x[13]+x[16]
z=x[2]+x[5]+x[8]+x[11]+x[14]+x[17]

println("Objective value: ", getObjectiveValue(sfLpModel)) # getObjectiveValue(model_name) gives the optimum
objective value

println("Optimal solution is full-time = \n", getValue(y))
println("Optimal solution is part-time = \n", getValue(z))
```

B. Inverse Matrix Calculation (R)

```
require(MASS)

room1= c(8, 9, 8.25, 6.5, 8.2);
room1<-data.frame(room1);

room2=c(6.5, 6.75, 6.75, 5.75, 6);
room2<-data.frame(room2);

room3=c(6.25, 3.5, 5.75, 4.5, 4.4);
room3<-data.frame(room3);

room4=c(3.25, 1.75, 4, 3.75, 2.2);
room4<-data.frame(room4);

room5=c(.25, .5, .75, .25, 0);
room5<-data.frame(room5);

room6=c(0,0,0,0,0);
room6<-data.frame(room6);

full=cbind(room1,room2,room3,room4,room5,room6);

av1=mean(full$room1);
av2=mean(full$room2);
av3=mean(full$room3);
av4=mean(full$room4);
av5=mean(full$room5);
av6=mean(full$room6);


u=c(u1,u2,u3,u4,u5,u6);

psq=10.645;

cvr<-cov(full);
invcvr<-ginv(cvr);
inv<-as.data.frame(invcvr)

write.csv(inv, file="inversematrixfridec1.csv")
```

C. Data transformation (MATLAB)

```
z=length(Time);


room_matrix=zeros(z, 2);
M=horzcat(Time, Rooms_occupied);

count1=0;

for j=1:8

    room_matrix(j,1)=M(j,1);

end

%First time period
if M(1,2)==1
     count1=count1+1;
    room_matrix(1,2)=count1;

end

if M(1,2)==2
    count1=count1+2;
    room_matrix(1,2)=count1;

end

if M(1,2)==3
     count1=count1+3;
    room_matrix(1,2)=count1;

end

if M(1,2)==4
     count1=count1+4;
    room_matrix(1,2)=count1;

end

if M(1,2)==5
     count1=count1+5;
    room_matrix(1,2)=count1;

end
```

```
if M(1,2)==6
     count1=count1+6;
    room_matrix(1,2)=count1;

end

%Second Time Period
if M(2,2)==1
    count1=count1+1;
    room_matrix(2,2)=count1;

end

if M(2,2)==2
    count1=count1+2;
    room_matrix(2,2)=count1;

end
if M(2,2)==3
     count1=count1+3;
    room_matrix(2,2)=count1;

end

if M(2,2)==4
     count1=count1+4;
    room_matrix(2,2)=count1;

end

if M(2,2)==5
     count1=count1+5;
    room_matrix(2,2)=count1;

end

if M(2,2)==6
     count1=count1+6;
    room_matrix(2,2)=count1;

end
%Third Time Period
if M(3,2)==1
    count1=count1+1;
    room_matrix(3,2)=count1;

end
```

```
if M(3,2)==2
    count1=count1+2;
    room_matrix(3,2)=count1;

end
if M(3,2)==3
     count1=count1+3;
    room_matrix(3,2)=count1;

end

if M(3,2)==4
     count1=count1+4;
    room_matrix(3,2)=count1;

end

if M(3,2)==5
     count1=count1+5;
    room_matrix(3,2)=count1;

end

if M(3,2)==6
     count1=count1+6;
    room_matrix(3,2)=count1;

end

%Fourth Time Period
if M(4,2)==1
    count1=count1+1;
    room_matrix(4,2)=count1;

end

if M(4,2)==2
    count1=count1+2;
    room_matrix(4,2)=count1;

end

if M(4,2)==3
     count1=count1+3;
    room_matrix(4,2)=count1;

end
```

```
if M(4,2)==4
     count1=count1+4;
    room_matrix(4,2)=count1;

end

if M(4,2)==5
     count1=count1+5;
    room_matrix(4,2)=count1;

end

if M(4,2)==6
     count1=count1+6;
    room_matrix(4,2)=count1;

end

%Fifth Time Period
if M(5,2)==1
    count1=count1+1;
    room_matrix(5,2)=count1;

end

if M(5,2)==2
    count1=count1+2;
    room_matrix(5,2)=count1;

end

if M(5,2)==3
     count1=count1+3;
    room_matrix(5,2)=count1;

end

if M(5,2)==4
     count1=count1+4;
    room_matrix(5,2)=count1;

end

if M(5,2)==5
     count1=count1+5;
    room_matrix(5,2)=count1;

end
```

```
if M(5,2)==6
     count1=count1+6;
    room_matrix(5,2)=count1;

end


%Sixth Time Period
if M(6,2)==1
    count1=count1+1;
    room_matrix(6,2)=count1;

end

if M(6,2)==2
    count1=count1+2;
    room_matrix(6,2)=count1;

end
if M(6,2)==3
     count1=count1+3;
    room_matrix(6,2)=count1;

end

if M(6,2)==4
     count1=count1+4;
    room_matrix(6,2)=count1;

end

if M(6,2)==5
     count1=count1+5;
    room_matrix(6,2)=count1;

end

if M(6,2)==6
     count1=count1+6;
    room_matrix(6,2)=count1;

end

%Seventh Time Period
if M(7,2)==1
    count1=count1+1;
    room_matrix(7,2)=count1;
```

```
end

if M(7,2)==2
    count1=count1+2;
    room_matrix(7,2)=count1;

end

if M(7,2)==3
     count1=count1+3;
    room_matrix(7,2)=count1;

end

if M(7,2)==4
     count1=count1+4;
    room_matrix(7,2)=count1;

end

if M(7,2)==5
     count1=count1+5;
    room_matrix(7,2)=count1;

end

if M(7,2)==6
     count1=count1+6;
    room_matrix(7,2)=count1;

end

%{
%Eigth Time Period
if M(8,2)==1
    count1=count1+1;
    room_matrix(8,2)=count1;

end

if M(8,2)==2
    count1=count1+2;
    room_matrix(8,2)=count1;

end
if M(8,2)==3
     count1=count1+3;
```

```
    room_matrix(8,2)=count1;

end

if M(8,2)==4
     count1=count1+4;
    room_matrix(8,2)=count1;

end

if M(8,2)==5
     count1=count1+5;
    room_matrix(8,2)=count1;

end

if M(8,2)==6
     count1=count1+6;
    room_matrix(8,2)=count1;

end

%}

%Iterations
for i = 8:1:z
    count=0;

    if M(i,2)==1
        count=count+1;
    end

    if M(i,2)==2
        count=count+2;
    end

    if M(i,2)==3
        count=count+3;
    end
    if M(i,2)==4
        count=count+4;
    end
    if M(i,2)==5
        count=count+5;
    end

    if M(i,2)==6
        count=count+6;
```

```
    end
    for j = 1:7

        if M(i-j,2)==1
            count=count+1;

        end

        if M(i-j,2)==2
            count=count+2;
        end

        if M(i-j,2)==3
            count=count+3;
        end

         if M(i-j,2)==4
            count=count+4;
        end

         if M(i-j,2)==5
            count=count+5;
         end

          if M(i-j,2)==6
            count=count+6;
          end



    end
    room_matrix(i,1)=Time(i);

    room_matrix(i,2)=count;

end
```

D. Finding Uncertainty Factors (AMPL)

MOD File:

```
reset;

set I := 1..21;
set J := 1..6;

param constant{I};

var u{J}>=0;

maximize Uncertainty: u[6];

subject to Constraint:
	constant[1] * u[1]*u[1] + constant[2]*u[1]*u[2] + constant[3]*u[1]*u[3] +
constant[4]*u[1]*u[4]  + constant[5]*u[1]*u[5] + constant[6]*u[1]*u[6] + constant[7]*u[2]*u[2]  +
constant[8]*u[2]*u[3]
	 + constant[9]*u[2]*u[4] + constant[10]*u[2]*u[5] + constant[11]*u[2]*u[6]  +
constant[12]*u[3]*u[3] + constant[13]*u[3]*u[4] + constant[14]*u[3]*u[5]  + constant[15]*u[3]*u[6]
	  + constant[16]*u[4]*u[4]  + constant[17]*u[4]*u[5]  + constant[18]*u[4]*u[6]  +
constant[19]*u[5]*u[5]  + constant[20]*u[5]*u[6] + constant[21]*u[6]*u[6] <=2.2
```

DAT file:

```
param constant:=
1	20.53170783
2	-47.99074346
3	-7.793760552
4	29.17218762
5	-13.71939314
6	0
7	31.0479119
8	8.48559318
9	-33.89643708
10	19.62033805
11	0
12	2.282548582
13	-7.762701846
14	0.91305487
15	0
16	12.1415256
17	-8.029880646
```

18	0
19	3.754253791
20	0
21	0

;

.RUN file:

```
reset;
option solver minos;
model Northshore2.mod;
;
data Northshore2.dat;
solve;
```