usc-dso-570 (/github/pengshi-usc/usc-dso-570/tree/master)
/ Handouts and Notes (/github/pengshi-usc/usc-dso-570/tree/master/Handouts and Notes)

DSO 570 Final Project Assignment Description

Learning Objectives:

- Model a complex business case as a precise mathematical problem, and clearly describe the assumptions.
- Use data analysis or simulation to rigorously identify opportunities for improvement to the status quo.
- Formulate an appropriate optimization problem that can be numerically solved using Python and Gurobi.
- Implement the optimization formulation in Python, and supplying appropriate test inputs.
- Communicate the findings using both technical and business language.

Contents

1. Problem Description ¶

Shannon Faris is the Assistant Dean of Institutional Research and Academic Administration at the USC Marshall School of Business. She and her colleague Hal Warning from the Marshall Office of Finance and Administration are in charge of coordinating classroom scheduling for the entire business school, which has 7 departments, 22 academic programs, and enrollment of around 5000 undergraduate students and 1000 graduate students. Since taking charge of the scheduling process in 2016, they have been implementing a series of reforms on course scheduling to improve efficiency and transparency. While they have already conducted extensive analysis on the current schedules and scheduling process, they are innovative managers who are open to explore the potential benefits of advanced business analytics techniques.

Your team has been hired as elite consultants to investigate the possibility of applying optimization to improve the current system and to rigorously quantify the potential gains. How you go about this task is open-ended: you can be creative in what part of the current system to introduce optimization, how to measure success, and how to implement the optimization. But you must be able to justify these choices and defend your assumptions. You will present your findings to Shannon and her team, as well as document your methodology and results in a report that will be scrutinized over by both senior administrators and data analysts.

1.1 Status Quo

The scheduling of courses and classrooms at USC Marshall begins almost one year before the semester begins. For courses in the fall 2018 semester, Shannon and her team would allocate to each department coordinator a set of classroom-time slots at the beginning of fall 2017. For spring 2019, the initial allocation is made at the beginning of spring 2018. These slots are based on the historical allocation for courses from that department, as well as on any special requests from the department coordinators. Each slot indicates a classroom as well as a time period in the week. (i.e. JKP 102 on Mondays at 2-4pm.)

Marshall has seven departments, each with its own scheduling coordinator, who is typically the head administrator for that department. The seven departments along with their commonly used abbreviations are as follows: Business Communication (BUCO), Data Sciences and Operations (DSO), Finance and Business Economics (FBE), Leventhal School of Accounting (ACCT), Management and Organization (MOR), Marketing (MKT), and Lloyd Greif Center for Entrepreneurial Studies (BAEP).

Upon receiving the initial allocation of time slots from Shannon's team, each department coordinator has a few months to populate the slots with courses, working in cooperation with the chair of the department, who assigns teaching duties to individual professors. For concreteness, we refer to this as Phase I of the classroom scheduling process. The current assignment of courses is largely based on historical schedules. However, changes must be made when there are new courses or new sections of existing courses that need to be allocated. Classrooms also have different sizes and capabilities, and the allocation should satisfy the particular needs of each course. During the scheduling process, individual professors may also reach out to the department coordinator to indicate preferences for teaching times. However, this is done at an ad-hoc basis and there is currently no systematic way of eliciting and accommodating faculty preferences.

After Phase I ends, any unused slots become open for any department to use. Over the next few months before the semester begins, the department coordinators work with Shannon and her team to schedule the remaining courses. We refer to this as Phase II of the scheduling process. Preferences for classrooms and times become increasingly difficult to accommodate during Phase II as previously available slots become taken. At this time, Shannon's team may need to find potential swaps of course times in order to find a feasible allocation that satisfies all courses. The schedule must be finalized before the student course selection process begins.

Currently, a significant fraction of courses and sections are not scheduled by the end of Phase I. This implies a lot of last minute scrambling and headaches for certain departments to schedule the remaining courses. In rare cases, the classroom may still be unassigned at the time students choose classes.

Since taking ownership of the scheduling process, they have been trying to increase the transparency of the communications with department coordinators, increase the percentage of courses that are scheduled by the end of Phase 1, and apply evidence-based decision making. They are constantly analyzing the historical and current schedules to identify possible improvements.

1.2 Desirable Goals

Ideally, a course scheduling system should:

- Efficiently utilize the available space to schedule all courses and output a feasible schedule on time, well before registration starts. This is projected to become more difficult in the future as demand for Marshall courses is expected to rise, but the school does not have the resources to build additional classrooms due to land restrictions. If efficiency does not rise through optimization, then Marshall may be forced to increasingly schedule classes during early morning or late evening hours, which may be unpopular to the majority of students and faculty.
- Satisfy the preferences and needs of students as much as possible, so that they are able to successfully schedule all their required courses as well as the electives that they most desire. Students also have preferences for certain time of day that should be satisfied as much as possible.
- Satisfy the preferences of faculty as much as possible. For example, most faculty prefer to teach everything on the same day, so they do not have to teach every day of the week. Moreover, many faculty prefer not to teach in the evenings, although some actually prefer evenings to avoid traffic. It's best to match the times with faculty preferences, so as to improve the work environment for faculty and help Marshall to recruit and retain the best.
- Be transparent and fair in how it satisfies student and faculty preferences, so that certain groups are not discriminated against.
- Limit the administrative effort needed by Shannon's team and by department coordinators, who also have other roles to fill.
- Adapt to the changing needs of the students and faculty, as the set of courses and the demand for each course changes from one year to the next, and there may be changes in the composition of students as well as turnover among faculty.

1.3 Institutional Constraints

The department coordinators are currently overtaxed in general, so any recommendation that requires additional time and effort from them has a low chance of being adopted. On the contrast, anything that would save them significant time and effort is appealing.

Similarly, a recommendation that can be easily incorporated into the current work flow has a much better chance of being implemented than a recommendation that requires large changes. For example, it is conceivable that Shannon's team may adopt a decision support tool that recommends a schedule that they can tweak; but it's hard to imagine that they would accept a black-box system that allows no human feedback and simply outputs an "optimized" schedule.

While the school leadership is open to innovative ideas that improve the status quo, it also wishes to minimize the risk of disrupting normal operations. As a result, a series of gradual improvements is easier to accept than a sudden large-scale revamp. Furthermore, it is important to be able to quantify the potential for improvement in a rigorous way, so as to make sure that any changes will be worth the efforts. The school will not make any changes outright without convincing evidence of its benefit.

2. Data Description

Shannon and Hal have shared a wealth of data that may be relevant to classroom scheduling. All of the data are available on this Dropbox folder: https://www.dropbox.com/sh/o7ojknaqcobunsf/AAA_tyj_HS0R7PJ6s8c2V1wWa?dl=0 (https://www.dropbox.com/sh/o7ojknaqcobunsf/AAA_tyj_HS0R7PJ6s8c2V1wWa?dl=0)

You are not expected to make use of all of the data but can choose what you would like to focus on. (It is better to focus on a portion of the problem to perform a thoughtful and deep analysis, rather than try to do everything but not doing it well.) For your reference, here is a description of all of the available data.

2.1 Schdules

The file **schedule_2015_to_2019.csv** contains the complete schedules of all Marshall classes from Fall 2015 (Term 20153) through Summer 2019 (Term 20192), including classes that were scheduled at some point but cancelled for whatever reasons. The columns are:

- **cancelled**: whether the section was later cancelled. Reasons for cancellation may include data entry error, insufficient demand, or idiosyncratic reasons.
- **date_of_cancellation**: starting from 20173, we also have the time when the cancellation took place.
- **term**: i.e. 20153 is Fall 2015, 20161 is Spring 2016, and 20162 is Summer 2016.
- **course**: the course ID.
- **section**: the section ID.
- **title**: name of the course.
- **mode**: "C" standas for regular Class, "L" for Lab, and "D" for Discussion. Note that labs and discussions have zero units because they are associated with a certain class.
- **units**: the number of tuition units, and roughly corresponds to the number of class hours per week.
- **level**: "UG" stands for undergrad and "G" for graduate courses, but this data is not always available.
- **department**: the department or degree program that "owns" the course.
- **type**: for certain courses, this column contains additional information, such as whether it is a PhD class, a Core class, or an Elective.
- **session**: USC associates each section with a session ID, which correspond to the timing of the course and where it is held. The fields **session_name**, **session_more_info**, **classes_begin** and **classes_end** contain information about the session. The files **session_codes.csv** and **session_more_info.csv** contains the source of the underlying data within these columns.
- **dr_flag** and **link**: special markers for administrative purposes, which you probably would not need.

- **first_days**, **first_begin_time**, **first_end_time**, **first_room**: the day of the week, and time of the day for the section.
- **first_instructor**, **first_instructor_uid**: the instructor for the course.
- **second_days**, **second_begin_time**, **second_end_time**, **section_room**: a few sections meet at two different times in possibly different rooms. Such cases are relatively rare.
- **second_instructor**, **second_instructor_uid**: a few sections have multiple instructors. Such cases are rare.
- **seats_offered**: number of seats available according to the professor.
- **reg_count**: number of registered students.
- **adj_reg**: adjusting the reg_count for half-semester courses, for the purpose of computing tuition units.
- **wait_count**: number of students on the waitlist.
- **total_tuition_units**: the (adjusted) registration count multiplied by the number of units.
- **classroom_capacity**: estimated maximum capacity of the room.
- **cap_remaining_seats**: seats_offered minus reg_count.
- **classroom_remaining_seats**: classroom_capacity minus reg_count.

In [1]:
```
import pandas as pd
pd.set_option("display.max_columns",100)
schedule=pd.read_csv('data/schedule_2015_to_2019.csv')
schedule.head()
```

Out[1]:

	cancelled	date_of_cancellation	term	course	section	title	mode	units	level	depar
0	True	NaN	20153	NaN	14025	External Financial Reporting Issues	C	4.0	NaN	
1	False	NaN	20153	ACCT-370	14025	External Financial Reporting Issues	C	4.0	NaN	
2	True	NaN	20153	NaN	14026	External Financial Reporting Issues	C	4.0	NaN	
3	False	NaN	20153	ACCT-370	14026	External Financial Reporting Issues	C	4.0	NaN	
4	True	NaN	20153	NaN	14027	External Financial Reporting Issues	C	4.0	NaN	

2.2 Classroom Information

The file **room_capacity.csv** contains the estimated capacity of most Marshall classrooms. The file **general_use_classrooms.csv** contains information on USC classrooms that Marshall can book if needed. However, Marshall prefers to use its own classrooms as booking USC classrooms would incur certain administrative costs. The USC classrooms were used heavily before the building of JFF, which was completed in 20163.

2.3 Department Allocations

The files **department_allocations_20171.xlsx** contains the allocation of slots to departments in Spring 2017. These are the slots that each department coordinator fills in during Phase 1, and unused slots become centrally allocated at the end of Phase 1.

The files **department_allocations_20193.xlsx** and **department_allocations_20201.xlsx** contain similar data for Fall 2019 and Spring 2020. However, the data is now organized in a visual format.

2.4 Student Course Selection

The file **student_course_selection_2015to2019.xlsx** contains anonymized data for student course selection, which gives information about the role of each course in the curriculum as well as what courses are taken together. There are several sheets.

In [7]:
```
course_selection=pd.read_excel('data/student_course_selection_2015to20
course_selection.keys()
```

Out[7]:
```
odict_keys(['course_selection', 'by_degree', 'by_class', 'together_cou
```

The sheet **course_selection** is the raw data, which has the following columns:

- **anonymized_id**: a scrambled version of the student ID.
- **major**: the major of the student.
- **class**: whether the student is undergrad (U) and if so the year in undergraduate. Alternatively, whether the student is graduate (G), law (L) or other (O).
- **owner**: the program that the student belongs to.
- **degree**: the degree objective of the student.
- **term**: the term for which the student is selecting the course.
- **course**, **title**, **section**, **instructor**, **units**: information about the course and section that the student selected.
- **enroll**: information on whether the student completed the course (L), withdrew (W), or used another special provision.
- **withdraw**: Equal to "W" if the student withdrew, and nan otherwise.

In [3]:
```
course_selection['course_selection'].head()
```

Out[3]:

	anonymized_id	major	class	owner	degree	term	course	title	section	instru
0	14681.019386	PSYC	U3	LASS	BA	20153	WRIT-340	Advanced Writing	66793	Wa Na I
1	14681.019386	PSYC	U3	LASS	BA	20153	ECON-351	Microeconomics for Business	26308	Akb Ral
2	28361.710711	IRGB	U3	LASS	BA	20153	BUAD-285A	Accounting Fundamentals, Financial and Manager...	14511	Da Ru /
3	28361.710711	IRGB	U3	LASS	BA	20153	BAEP-452	Feasibility Analysis	14380	Orla
4	28361.710711	IRGB	U3	LASS	BA	20153	BAEP-423	Management of Small Businesses	14374	A (

The sheet **by_degree** aggregates the above by grouping on term, course, and the first item listed in the degree field. This sheet gives information on what type of students typically take a class. For example, in the examples below, we see that both ACCT-370 and ACCT-371 are exclusively undergraduate classes in Fall 2015.

In [4]:
```
course_selection['by_degree'].head()
```

Out[4]:

	term	course	first_degree	count
0	20153	ACCT-370	BA	1
1	20153	ACCT-370	BS	105
2	20153	ACCT-371	BS	110
3	20153	ACCT-372	BA	1
4	20153	ACCT-372	BS	158

The sheet **by_class** is similar to the above, except we have the class information instead of first_degree. In the following example, we see that the majority of students taking ACCT-370 do so in their 4th years, although there are some that take it in their 3rd years.

In [5]:
```
course_selection['by_class'].head()
```

Out[5]:

	term	course	class	count
0	20153	ACCT-370	U3	27
1	20153	ACCT-370	U4	79
2	20153	ACCT-371	U3	30
3	20153	ACCT-371	U4	80
4	20153	ACCT-372	U3	11

The sheet **together_count** gives information on what courses are typically taken together. For example, in the following table, we see that all 220 students who took GSBA-542 in Fall 2018 also took GSBA-599. However, there are 600 students who took GSBA-599, and not all of them took GSBA-542.

- **together_count** is the number of students that term who selected both **course1** and **course2** at the beginning of the semester.
- **total_course1** is the total enrollment of course1 that semester.
- **total_course2** is the total enrollment of course2 that semester.
- **together_proportion1** is together_count divided by total_course1.
- **together_proportion2** is together_count divided by total_course2.
- **together_proportion_max** is the maximum of the two previous columns.

The sheet is sorted in descending order of together_proportion_max, total_course1 and total_course2.

In [8]:
```
course_selection['together_count'].head()
```

Out[8]:

	term	course1	course2	together_count	total_course1	total_course2	together_proportion1
0	20183	GSBA-542	GSBA-599	220	600	220	0.366667
1	20183	GSBA-542	GSBA-544	203	600	203	0.338333
2	20183	GSBA-542	GSBA-550A	203	600	203	0.338333
3	20173	GSBA-542	GSBA-552	222	539	222	0.411874
4	20173	GSBA-510	GSBA-522A	179	457	179	0.391685

2.5 MBA Student Preferences

In Spring 2018, USC Marshall experimented with using a new course selection system called CourseMatch, in which students indicated their preference over all courses, and a special algorithm determined the assignment. For various reasons, USC did not continue using the system for subsequent semesters, but the data collected for this semester gives insights on student preferences. (In other semesters, we only observe which courses students enrolled in, but not which courses they would like to enroll in and how much they like each course.) However, this system was only used by full-time MBA, part-time MBA and iBEAR students. The columns are:

- **student_id**: a scrambled version of the student ID.
- **units**: the total number of units the student is expected to enroll in that semester.
- **year**: whether the student is a year 1 MBA or year 2 MBA student.
- **ranking_class**: 4 denotes that the course is the student's "Favorite," 3 denotes "Great," 2 denotes "Good" and 1 denotes "acceptable."
- **total_rank**: 1 denotes the course being the student's top choice, 2 denotes the second choice and so on.
- **class_rank**: ranking of course within the same "ranking_class." For example, if a student selected three courses as "Great" (ranking_class=3), then 1 denotes the course being the student's favorite among the three, and 2 denotes the second favorite among the three, and so on.
- **course_id**, **slug**, **longname**: informaiton about the course and section. Some sections have been split into two virtual sections to control the relative proportion of students from different programs.
- **night**, **meetingday1**, **meetingtime1**, **meetinglocation1**: information about the timing of the course and section: whether the course meets in the evening, what day of the week it meets, time of meeting, and room.
- **allocated**: binary indicator for whether the student got the course/section.
- **price**: the CourseMatch system is based on simulating a virtual auction, and the price gives information on how many virtual points a course costs. Higher prices means that the demand is high compared to the supply.
- **count**, **max_capacity**: the number of students enrolled in the course, and the number of seats offered.

In [11]:
```
course_match=pd.read_csv('data/course_match_2018.csv')
course_match.head()
```

Out[11]:

	student_id	units	year	ranking_class	total_rank	class_rank	course_id	slug	lor
0	30139	15.0	2	4	1	1	MKT-533	MKT-533 (033-16530)-NOTEVNG	E Strate Ful
1	30139	15.0	2	3	2	1	DSO-583	DSO-583 (033-16292)	Op Consulti
2	30139	15.0	2	3	3	2	BAEP-554	BAEP-554 (033-14447)	Venture I (0
3	30139	15.0	2	3	4	3	MOR-569	MOR-569 (033-16717)-NOTFT1	Negotia Deal- (033, Ful
4	30139	15.0	2	3	5	4	BAEP-562	BAEP-562 (033-14425)	Entreprer in eCo (0

2.6 Faculty Survey

The file **faculty_survey_raw.csv** contains the result of a Qualtrics survey conducted in Spring 2017 about professor's preferences about teaching. The file **faculty_survey_report.pdf** contains a summary report listing the survey questions and the overall distribution of responses.

The file **professor_information_2018.csv** contains the result of a web crawl of the Marshall website in Spring 2017 to retrieve additional information regarding professors, including their rank and position. Clinical faculty are teaching only, while Tenure Track faculty are responsible for doing both research and teaching. Lecturers and Assistant Professors are the lowest ranks, and such professors can be promoted to Associate Professors and then to Full Professors.

3. Deliverables

The following 8 files are due on Blackboard by **Tuesday, May 5** at **6pm Pacific Time**.

3.1 Project report (.docx or .pdf file)

A 10-15 page report in Word or PDF format, with the main body targeted to a senior administrator at USC, and a technical appendix targeted to an data analyst (both parts are included in the page count, but an optional title page is excluded). The report should answer the main questions: **what are the biggest opportunities for improvement and how much are the potential gains from optimization?**

The **main body** of the report should be written in language appropriate for a non-technical but business saavy executive, with minimal mathematical notation or technical jargon. It must contain the following five components:

1. Executive Summary: Concise overview of the main insights of your analysis. This includes the biggest opportunity for improvement, the estimated potential gain from your optimization, and final recommendations. Be brief and clear.

2. Opportunity for Improvement: What is the main weakness or inefficiency in the current process of classroom scheduling that you choose to focus on in your project? You must illustrate this weakness or inefficiency either by analyzing existing data, or by conducting a simulation analysis with reasonable assumptions. (Either data analysis or simulation can suffice, but you cannot simply claim the weakness without justificaiton.)

3. Optimization Methodology: What does the optimization tool (see 3.3) you created do and how does it interact with the course scheduling process? This section should describe the optimization tool in enough detail and precision so another analyst can understand your objectives and constraints, and be able to run it on different data of the same format. In particular, you should clearly describe

- **i)** At a conceptual level, what are the input data needed for your optimization? (This should match the types of data in the sample input that you provide, see Section 3.4.)
- **ii)** What are the outputs of the optimization? (This should match types of data in the sample output that you provide, see Section 3.5.)
- **iii)** What are your decision variables, objectives, and constraints in words? (You should defer the precise formulation using mathematical notation to Appendix A1.)
- **iv)** How do you envision your tool being used by the end user? Who would be the intended end user (Shannon/Hal, department coordinator, individual faculty, or someone else)? You should describe both the command needed to execute your tool, as well as how you imagine the user interacting with it.

4. Optimization Results: Use easy to understand tables or graphs (not large printouts or messy plots that are hard to read) to illustrate the result of using your optimization tool. You need to quantify the potential gains from applying your optimization methodology. Furthermore, you need to provide qualitative intuition on where the gains come from and why your outputs make sense. You should also illustrate what qualitative insights can be learned from using your optimization. For any claims you make, you must rigorously justify them based on sound reasoning and reference to supporting analysis. It is important not to oversell your result, and not to make bold claims without sufficient justification. You would be graded on the appropriateness of your presentation, as well as the rigour of your analysis.

5. Discussion: This section should discuss the following

- **i) Appropriateness of Methodology:** Why do you think your proposed methodology is the best fit for the situation? In particular, you need to explain the rational behind your choice of input and output data in designing the tool, as well as your choice of objective and constraints. You should also summarize the main assumptions in words, and acknowledge the main weaknesses of your analysis. (Detailed discussion of technical issues are in Appendix A2.)

- **ii) Final Recommendation:** In light of your analysis, what do you suggest Shannon and Hal to pursue as next steps? Beside stating your recommendations, you must justify them using sound reasoning, while exercising business sense, which involves taking into account other important practical issues outside of your mathematical formulation.

The **technical appendix** should be written in language appropriate for a technical audience, using correct mathematical notation and precise technical terms. It should contain the following two components.

A1. Mathematical Formulation: A precise statement of your optimization formulation using correct mathematical notation. You should define the data and decision variables, and use them to express the objectives and constraints.

A2. Discussion of Technical Details: Imagine the reader is a saavy analyst who is sceptical of the technical rigour of your results. Anticipate the issues that the reader may see, and address them to the best of your ability. You should critically examine all of the assumptions underlying your analysis, explain your reasoning behind them, and acknowledge what are the weakest points and the biggest gaps, which you would like to fix if you had more time or resources.

3.2 Documentation of all tables and plots (.ipynb file)

A Jupyter notebook which provides sufficient information so that a trained analyst can understand what exactly went into each numerical estimate, table or plot in your project report, and be able to reproduce them if needed. If you conducted this analyst in Python, your Jupyter notebook should show the code that created them, and explain any technical details or underlying assumptions. You should make this notebook easy to navigate using headers in Markdown cells, and clearly indicate which block of code corresponds to which plot. If your report contained figures or plots created using another program (i.e. R or Excel), you should explain how exactly you created the figure or plot or made your calculations.

3.3 Optimization tool (.py file)

A Python module containing your optimization tool, which implements the optimization formulation in your project report, and can be used to reproduce your results. This file should contain a function called "optimize" with at least two input parameters:

- inputFile: the path to the input file.
- outputFile: the path to write the output.

If your code requires multiple input files, then you can update the parameters appropriately. Having this function would allow a user to import your module from Jupyter notebook and run your code without understanding your code.

In addition, the module need to be able to be directly executed from command line. The following example teaches you how to do this. Suppose you write a module called "optimize.py" and you include the above code at the end:

```
if __name__=='__main__':
    import sys
    print (f'Running {sys.argv[0]} using argument list {sys.argv
```

Then one can run it in command line as follows.

```
ipython optimize.py input.xslx output.xlsx
```

This would yield the output

```
Running optimize.py using argument list ['optimize.py','input.x
slx','output.xlsx']
```

Instead of printing the arguments supplied by the command line, you can simply call your optimization function.

Your submitted Python module will be graded on correctness and ease of being understood. For the latter, you should include appropriate docstrings to all functions, write comments to explain the logic, and use intuitive variable names.

3.4 Sample input (.xlsx or .csv file)

One or more sample input files that can be fed into your optimization code. Your optimization code should be flexible enough such that one can use it simply by changing the input file, and not have to alter anything in the code. You should avoid hardcoding any parameters into the code, but try to read all parameters from the input files.

3.5 Sample output (.xlsx or .csv file)

The result of running your optimization code given the input files you provided. This needs to be reproducible in that running your code again with the input files you submitted should yield the same output file on another computer.

3.6 Documentation of optimization tool (.txt, .pdf or .docx)

A text file describing how to use your optimization module. You should explain how a user can run your code from the command line, as well as what functions to import into Jupyter notebook. This needs to be sufficiently clear so someone with very little knowledge of Python can use your tool.

In addition, you should explain in detail the format and significance of every part of your sample input and output data. This needs to be in sufficient detail so someone else can transform their own data into the correct format to use your tool, and interpret the results correctly.

3.7 Presentation slides (.pptx or .pdf)

The slide deck that you will use during the final presentation. You must submit your slides with the report and you will not be able to change your slides on the day of presentation.

3.8 Contributions of team members (.docx or .pdf file)

A one page description of the individual contribution of each team member to the project, as well as the percentage that each contributed. This may be used to adjust individual grades if certain team members did not contribute their fair share.

4. Grading Rubrics

4.1 Project report (20% of course grade)

The report is out of a maximum of 30 points, with 3 points being distributed in each of the following 10 categories. What is needed to obtain a perfect score in each category is described below. Within each category, there is a 0.5 point deduction for every minor issue and a 1 point deduction for every major issue.

1. Executive summary: The executive summary clearly and concisely summarizes the main findings of the report, including the inefficiency found, the potential gain from optimization, and the final recommendation. The writing is polished and free from spelling or grammar issues, with appropriate words and well-formed paragraphs. The overall look is professional and the formatting should be clean and consistent. The main message of the report can be quickly grasped by a busy executive.

2. Opportunity for improvement: Clear and precise description of what is the inefficiency in the status quo you found. The description of the opportunity for improvement should be compelling, but the magnitude of the inefficiency should not be oversold.

3. Rigour in justifying opportunity for improvement: The identified opportunity for improvement is convincingly illustrated using either data analysis or simulation. Moreover, the analysis is rigorous and free from errors and statistical fallacies. The way you conducted the data analysis or set up the simulation is clearly explained so the overall idea is clear to readers.

4. Clarity of methodology: The required inputs, outputs, as well as the decision variables, objective, and constraints are clearly described. The language should target a non-technical but business saavy executive, with minimal mathematical notation or technical jargon. (The technical details should be deferred to the Appendix.)

5. Clarity of results: The optimization results and insights are clearly presented. If tables, figures or plots are used, they should be easy to read, easy to understand, fully and appropriately labeled, and clearly described. There should not be unnecessary, distracting, or unprofessional looking visuals. Furthermore, the potential gain from using your optimization tool is clearly articulated, as well as qualitative intuition on where the gains may come from and why they make sense.

6. Rigour of results: The interpretation of the numerical results is correct and any claim of potential gain or insight found is convincingly justified by the presented supporting evidence. The result is not oversold, and the logic of the analysis and its explanation is correct.

7. Discussion of methodology: The reason behind the chocie of input and output data, as well as the decision variables, objectives, and constraints are clearly described and the logic is sound. The main assumptions are clearly and thoroughly articulated. The weaknesses of the sound. The main assumptions are clearly and thoroughly articulated. The weaknesses of the analysis are clearly acknowledged.

8. Discussion of recommendation: The finall recommendation is clearly stated and explained. The issues that are not captured by your model are accounted for, and your recommendation and explanation make business sense (showing street smarts).

9. Correctness of mathematical formulation: The input data, decision variables, objectives, and constraints are clearly expressed by appropriate, precise and correct mathematical language. All formula and mathematical notation should be rendered nicely. Furthermore, the mathematical formulation correctly implements the English description in the main body.

10. Discussion of technical details: The potentially problematic technical details in the analysis are identified and precisely described. The reasoning behind these technical choices are clearly presented and the remaining weak points and gaps are acknowledged. There are no large weaknesses or gaps in the analysis that are not mentioned.

4.2 Python code and documentation (5% of course grade)

The optimization tool (3.3), documentation (3.2 and 3.6), and data files (3.4 and 3.5) are graded together using the following rubric. There is a maximum of 15 points, with 3 points for each of the following categories. What is needed to obtain a perfect score in each category is described below. Within each category, there is an one point deduction for every minor issue, and a two point deduction for every major issue.

1. Correctness: Code implements what is claimed without syntax or logical errors. The code can be run on another computer without error. All the features described in the documentation (see 3.6) works. Code correctly runs on the sample input (3.4) and produces the sample output (3.5).

2. Scalability: Code is able to be scaled up to the size of inputs that is needed for the actual business context, within a reasonable runtime, while obtaining a good solution. The optimization formulation is reasonably efficient (cannot do the same thing with many fewer decision variables or constraints.)

3. Usability: The inputs needed to run the code can be plausibly obtained by the end user within a reasonable amount of effort. The functionality of the code matches the need of the end user. While the expectations of the code is that it is only a prototype, it should be something that USC Marshall would be interested to develop further.

4. Readability: The code includes appropriate docstrings for all functions, and enough comments so a trained analyst can understand it easily. The variable names should be appropriate and there should not be extraneous logic that has little functionality but may confuse the reader.

5. Completeness of documentation and data files: The documentation of plots and figures (see 3.2) and the documentation of the optimization tool (see 3.6) contain all of the required components: explanation with enough detail to reproduce every numerical estimate, table or plot in the main body; instructions with examples of how to use the optimization tool both from Jupyter notebook and from the command line; explanation of the format and significance of every piece of the sample input and output data. Moreover, all of the input files needed to run the code are attached to the submission. Moreover, the sample output that would be created by the code is also given. All of these should match the high-level description in the project report and the detailed description in the documentation (see 3.6).

4.3 Final presentation (5% of course grade)

The presentation is out of 18 points, with 3 points for each of the below categories. What is needed to obtain a perfect score in each category is described below. Within each category, there is a 0.5 point deduction for every minor issue and a 1 point deduction for every major issue.

1. Business insights: Presentation provides clear insights that may be valuable to the stakeholders. Moreover, the insights are effectively communicated, and the reasonings are compelling, having sufficient supporting evidence or arguments.

2. Structure and timing: Present information in a logical, interesting, orderly sequence. The main message of the talk and the outline is clear from the start. Use of signposting language to guide the audience through the presentation. The presentation is well timed and completed within the time allotted, spending enough time on the important points not rushing through too many things. (When time is short, you should skip less important points rather than trying to rush through and lose the audience.)

3. Visual aids: Slides have few words and can be easily read by audience sitting far away; visuals improve the presentation and coordinate with the points of speech; effective use of highlighting techniques; no extraneous graphics, animations and distracting visuals; avoid complex looking math formula.

4. Engaging the audience: Engage audience members from all sections of the room through eye contact, body language and enthusiasm; relaxed and confident poise, with appropriate gestures and movement to enhance articulation, but no distracting mannerisms or fidgeting; clearly spoken and engaging voice with appropriate variation in tone, emphasis and pauses.

5. Accessibility to non-technical audience: Explain content in such a way that a senior executive without much technical training can follow along the overall logic. Whenever you use a technical term, you should also explain or reword it in a non-technical way to aid understanding.

6. Correct technical content: Whenever technical terms or mathematical notation is used, they are used correctly; the logic of arguments presented is rigorous and free from fallacies. There should not be significant fallacies or blunders that a technically minded skeptic can latch on to.