FINAL REPORT

EDUCATIONAL RESEARCH MINI-GRANT

2016-2017

Student-Centered Dynamic Syllabus Development for Mathematical Programming

by

Dincer Konur, Ph.D.
Assistant Professor
Engineering Management and Systems Engineering
Missouri University of Science and Technology

Address: 206 Engineering Management, 600 W. 14th Street, Rolla, MO, 65409

Email : konurd@mst.edu

Phone : 573 341 7256

Submitted in December 2017 to Center for Educational Research and Teaching Innovation, Missouri University of Science and Technology.

SUMMARY

In this project, the PI investigated a methodology to prepare a student-centered syllabus and evaluated the students' reactions to the student-centered syllabus. Specifically, the main idea of a student-centered syllabus is to help students learn not only the learning objectives defined by the instructor but also the learning objectives individually defined by the students based on their own interests and/or needs. Throughout the project, the PI developed and tested a student-centered syllabus for Engineering Management 6412 – Mathematical Programming course taught in Spring 2017. The project consisted of three main phases:

- The first phase (***before-the-semester***) is to prepare an outline of the topics to be included in the syllabus. To do so, input from engineering faculty in the campus is collected. Then, the outline of the topics are based on the instructor-defined learning objectives as well as the input from the engineering faculty.
- The second phase (***during-the-semester***) is to acquire students' rankings on the instructor-defined topics and students' individual learning objectives. To do so, at the beginning of the semester, students are asked to rank their learning expectations from instructor defined topics as well as other topics and disclose the topics they would like to learn. Then, the common topics, which are the learning objectives common to every student, and individual topics, which are unique to each student, are finalized.
- The third phase (***after-the-semester***) is to evaluate the students' reactions to the student-centered syllabus. To do so, in addition to the student evaluations, students' satisfaction levels are acquired at the end of the semester. Then, satisfaction levels are compared to the expectation levels collected at the beginning of the semester.

This project proposes a method for a well-tailored syllabus, that accounts for students' individual learning objectives and the evaluation results indicate that the students react positively and enjoyed the idea of learning not only the topics defined by the instructor but also the topics they wanted to learn with assistance. The project outcomes are published as a conference proceeding paper and the PI is working on a full journal article, which will be finalized after collecting additional data in teaching the same course once again with student-centered syllabus approach.

PURPOSE OF THE PROJECT

Background: Graduate courses on concepts/tools/methods that are used by many disciplines naturally interest graduate students from different disciplines. Engineering Management (EMGT) 6412 – Mathematical Programming (co-listed with Mathematics (MATH) 6665) is one such course, offered to both on-campus and distance graduate students enrolled at the Missouri University of Science and Technology (Missouri S&T). Specifically, Mathematical Programming is a set of concepts/tools/methods used to mathematically model and solve many engineering, science, and business as well as social science problems [1]. Many engineering faculty at Missouri S&T use mathematical programming tools for their research; therefore, many graduate students enrolled in engineering (and science) programs need/want to learn the basic theory of and how to use these concepts/tools/methods.

EMGT 6412/MATH 6665 is the only course, offered to on-campus and distance students, that covers the basic foundations of mathematical programming independent of application and domain. Also, students from different disciplines have been enrolling in this course to learn about mathematical programming, mostly to help them in their research studies. However, mathematical programming discipline is a broad discipline and it is impossible to cover the details of all of the concepts within one course (typically, industrial engineering/optimization/operations research programs offer various courses on mathematical programming – separate courses on different tools such as linear programming, integer programming, nonlinear programming, stochastic optimization, network optimization, etc.). Therefore, in previous semesters, as the instructor of the course, the PI has focused on covering the main concepts throughout a semester and prepared the course syllabus accordingly. Specifically, the course syllabus included: introduction to mathematical programming (1 week); linear algebra and sets review (1 week); linear programming and simplex method (2 weeks); duality (1 week); decomposition principle and column generation (2 weeks); integer programming and mixed-integer programming (2 weeks); basics of branch-and-bound, branch-and-price, and Bender's decomposition (3 weeks); and other topics, if time allows, such as well-known heuristics and nonlinear programming (1 week). A sample course syllabus can be seen in [3]. Note that these topics are the basic topics covered in many mathematical programming courses and textbooks ([4] is used as the course textbook).

Problem Description: The problem with the fact that the topics to be covered in class are defined by the instructor is that some of the students will not be introduced to the concepts they need/want to learn considering their research needs. This, in turn, might result in students trying to learn the concepts on their own. The common problems the PI observed with students trying to learn mathematical programming concepts/tools without proper instruction are:

- Failing to learn the basic theory behind the concepts and tools: This leads to inability to properly apply, implement, or modify the concepts and tools to their research problems.
- Failing to choose the right concepts and tools to use: This is mainly because they are not aware of the existence of some concepts and tools.

Specifically, distance students might suffer from these issues more as they will mostly not get face-to-face discussion. Another problem is that the research advisors (who use optimization but not conduct research on optimization) might also fail to direct students in the right way (and not realize that) as mathematical programming is not their main expertize.

This project attempted to address the problem of syllabus development for a course that might have various learning expectations from a diverse set of students. Specifically, the project focused on developing a student-centered syllabus rather than developing a concept-centered syllabus. This project addressed a learning problem for mathematical programming concepts. Incomplete understanding of specific tools and not-knowing specific concepts definitely result in devaluation of the research studies that require mathematical programming tools and might negatively affect research-oriented students' professional development. However, as noted before, it is not possible to cover all tools within one course. This project tries to overcome these problems by focusing on developing student-centered syllabus for a graduate-level mathematical programming course.

Purpose of the Project: The main purposes of this project are as follows:

- ***Improve attainment of learning outcomes:*** The very basic learning outcome of the mathematical programming course is to teach the basics and main tools of mathematical programming so that the students can correctly utilize needed tools in their research studies. However, concept-centered learning outcomes will not achieve the ultimate learning outcome due to the broadness of the discipline. By developing student-centered learning outcomes, learning expectations of the students will be better satisfied (i.e. the learning outcomes will be tailored based on student needs).
- ***Enhance student professional development:*** The idea of student-centered syllabus development will enable the course cover the topics/tools/concepts that are most needed by the students considering their needs. Therefore, this project will enhance students' knowledge on the topics they study/research in their studies and contribute to their professional development.
- ***Promote active learning:*** Creating a student-centered syllabus will require input from the students. Therefore, the students will involve in activities such as reading related studies, discussing their research needs, and evaluation of the concepts for suitability for their research and learning needs.

The project focuses on the way of syllabus development for a campus-wide course, with a large set of concepts, taken by diverse set of students coming from different disciplines with different learning expectations. The main idea of the project is similar to "differentiated instruction" concept. In particular, the premise of differentiated instruction is to tailor content, process, product, and environment based on the individual learner [5]. However, differentiation concept is mostly understood as changing the subjects and/or increasing/decreasing the complexity of the subjects to be covered. In this project, the main goal is to create a framework that can be used to prepare a student-oriented syllabus that will focus on making the most for the student based on the student's individual learning expectations. Specifically, student-centered syllabus will have concepts driven from the students' learning expectations and include student-specific assignments based on the students' individual learning objectives.

METHODOLOGY

Preparation and evaluation of student-centered syllabus consisted of three main phases: before-the-semester, during-the-semester, and after-the-semester. Next, the details of the tasks carried out in each phase are explained. The results are later discussed in *Results* section.

Phase 1 - Before-the-Semester: The purpose of Phase 1 is to prepare an outline for the syllabus of the course. Specifically, the outline will include the topics to be discussed during the regular class times. Therefore, these topics will be common to all students. To do so, two tasks are carried out: (i) collecting engineering faculty input on mathematical programming and (ii) defining common topics and assignments for the course. The details of these two tasks are as follows.

Task 1.1. Acquire engineering faculty input on mathematical programming: The goal of this task is to learn the mathematical programming concepts and the tools, which are seen by the engineering faculty as the most necessary for research oriented graduate students. To do so, a

survey is conducted among the engineering faculty, from different departments (the PI tried to collect input from the science programs as well, however, faculty in science programs did not participate in the survey). The College of Engineering and Computing (CEC) in Missouri S&T has 9 engineering departments [6] and the faculty in each of these departments are emailed and asked to voluntarily participate in a one-minute simple survey. The survey is still currently available at [7] and can be seen in Appendix 1. At the beginning of the survey, there is short explanation for its intention. This survey has the following 6 parts:

1. Please select your department (there is a list of departments to select from).
2. Do you use mathematical programming/operations research/optimization concepts and/or tools in your research? Yes or No (the participant picks yes or no)
3. Please rate the necessity of each of the main topics listed below for your students' research needs (1: unnecessary, 2: less necessary, 3: necessary, 4: very necessary): Mathematical modeling (formulating problems), linear programming concepts and tools, integer/mixed-integer programming concepts and tools, nonlinear programming concepts and tools, network optimization, stochastic optimization, robust optimization, multi-objective optimization, multi-level optimization, game theory, heuristics. That is, 11 topics are asked to be ranked.
4. Please list any other mathematical programming/optimization concepts that your students might need (this an open comment question to capture the other topics faculty can list).
5. What software do you or your students use for optimization? C/C++/C#, Matlab, GAMS, Excel, Other (here the participant can select multiple of these and specify other software).
6. Any comments (to get open comments from the participating faculty).

The output of this task is used to identify the topics to be discussed during regular class hours for each student enrolled in the course. Furthermore, with this task, the common types of software used for optimization by the faculty are also identified.

Task 1.2. Define common topics and assignments for the course: This task defines the topics that will be discussed during the lectures. Each student will learn about these topics and class assignments will be given to the students (i.e., the students will complete the same assignments on these common topics). These common topics are identified using the faculty input results as well as the instructor's teaching goals based on the instructor's expertise.

Phase 2 - During-the-Semester: The purpose of Phase 2 is to finalize the syllabus based on student input. To do so, two tasks are carried out: (i) collecting student's learning expectations on various mathematical programming topics and (ii) defining individual topics and assignments for each student. The details of these two tasks are as follows.

Task 2.1. Collecting student's learning expectations on mathematical programming topics: This task aims at determining each student's individual learning expectations. Here, the students have a chance to rank the common topics in terms of how much they would like to learn as well as specify other topics that are not covered by the common topics. To do so, first, a list of common topics is already in place and the students are asked to rank them. Then, in

the same list, there is a wide list of topics, from which the student can pick an individual topic. This wide list is prepared using the taxonomy suggested in [2] as well as the instructor's knowledge. Table 1 presents the list of topics grouped under 9 main categories.

Table 1. *List of topics and main categories*

Category 1:	FORMULATING MODELS (FOR)
	Formulations steps and modeling
	Continuous vs. Integer variables
	Conditional constraints
	Overall learning
Category 2:	LINEAR PROGRAMMING (LP)
	Properties of LP models
	Simpex method
	LP Duality
	Decomposition principles
	Column generation
	Overall learning
Category 3:	INTEGER PROGRAMMING (IP)
	Properties of IP models
	Linear relaxation relations
	Branch and bound method
	Branch and cut method
	Overall learning
Category 4:	MIXED-INTEGER PROGRAMMING (MIP)
	Properties of MIP models
	Relation to IP models
	Branch and Price
	Bender's Decomposition
	Overall learning
Category 5:	NON-LINEAR PROGRAMMING (NLP)
	Properties of NLP models
	Global/unconstrainted optimization
	KKT Conditions
	Lagrangean duality
	Overall learning
Category 6:	MULTI-OBJECTIVE OPTIMIZATION (MOP)
	Properties of MOP models
	Basic solution concepts
	Detailed analysis
	Overall learning
Category 7:	MULTI-LEVEL OPTIMIZATION (MLP)
	Properties of MLP models
	Basic solution concepts
	Detailed analysis
	Overall learning
Category 8:	STOCHASTIC OPTIMIZATION (SOP)
	Properties of SOP models
	Basic solution concepts
	Detailed analysis
	Overall learning
Category 9:	ROBUST OPTIMIZATION (ROP)
	Properties of ROP models
	Basic solution concepts
	Detailed analysis
	Overall learning

For each category, overall learning of that category is included as a topic. The ranking of the common topics might be used in later semesters for modifying the common topics and common assignments (Phase 1, in following years, can include input from faculty, instructor, and students). This task is completed by emailing the students the list. The full list is shown in Appendix 2. It asks for students' expectation level on each listed topic as well as the students can list any additional topic that is not in the list, for their individual topic as shown in Appendix 2. The expectation level is measured using a scale of 1 to 5 as described in Figure 1 below (also, see Appendix 2).

Expectation scale: 1 to 5
1: is not aware of the concept
2: is aware but was not expecting to learn
3: is expecting at least an introduction
4: is expecting more than an introduction
5: is expecting to learn

Figure 1. *Expectation level scale used to collect student learning objectives on common topics*

In the case, a student cannot identify a topic, the instructor can discuss with the student and they can identify an individual topic, which will not be a common topic, to best benefit the student, for instance, considering research studies or professional career goals.

Task 2.2. Defining individual topics and assignments for each student: After students' inputs are collected, the instructor prepares individual assignments for each student based on the topic selected by the student. Recall that the students have the option to determine topics they want to learn that are not in the list. The assignments are designed so that the student can understand the basic theoretical background, current state-of-the-art, some applications, and simple implementation of the topic. Specifically, each student are assigned 4 or 5 individual assignments consisting of: problem description and mathematical formulation, theoretical analyses and properties of the problem/model, review of solution concepts and focusing on a solution method, coding and implementation of a solution method. These assignments assure that the student understands the basics of the individual topic as well as gets on-hand experience with some solution methods.

Phase 3 - After-the-Semester: The purpose of Phase 3 is to evaluate the student-centered syllabus from the students' perspective. To do so, three tasks are carried out: (i) collecting students learning satisfactions on various mathematical programming topics (ii) comparing satisfaction vs. expectation levels, and (iii) investigating the students' comments on student evaluation forms. The details of these two tasks are as follows.

Task 3.1. Collecting student's learning satisfactions on mathematical programming topics: This task aims at determining each student's individual learning satisfaction on the various topics included in the list sent out to the students at the beginning of the semester. Here, at the end of the semester, each student is sent out the list they filled at the beginning of semester with their expectation levels and each student is asked to rank their learning satisfaction on the mathematical programming topics included in the list. A sample list with learning satisfaction

ranking is shown in Appendix 3. The learning satisfaction level is also measured using a scale of 1 to 5 as depicted in Figure 2 below (also, see Appendix 3).

Satisfaction scale: 1 to 5		
1: was not satisfied at all		
2: below my expectation		
3: ok but would not mind more		
4: fairly satisfied		
5: satisfied		

Figure 2. *Expectation level scale used to collect student learning objectives on common topics*

Task 3.2. Comparing satisfaction vs. expectation levels: This task compares the students' expectation levels to satisfaction levels. To do so, for each student for each topic included in the list, the difference between the satisfaction and expectation levels is calculated as the comparison level. That is,

$$Com_{ij} = Sat_{ij} - Exp_{ij}$$

where Com_{ij}, Sat_{ij}, and Exp_{ij} denote the comparison level, satisfaction level, and expectation level of student i for topic j. Note that the comparison level can vary between -4 and 4. Negative values would imply that the student is not satisfied (not learned as much as expected) while positive values would imply that the student is satisfied (learned at least how much expected) on a particular topic. The scale presented in Table 2 is used to evaluate the results.

Table 2. *Implications of comparison levels*

Comparison Level	Implication
[4,1)	Exceeds expectations
[1,-1]	Meets expectations
(-1,-4]	Fails expectations

For each main category (see Table 1), each student's average comparison level is calculated using the topics under that category. Then, the average comparison levels are analyzed. As noted in Table 2, if the average comparison level is between 4 and 1, it is accepted that the student's expectation on the category is exceeded; if the average comparison level is between 1 and -1, it is accepted that the student's expectation on the category is met; and if the average comparison level is between -1 and -4, it is accepted that the student's expectation on the category is failed.

Task 3.3. Investigating the students' comments on student evaluation forms: This task aims at determining if there are common comments given by the students, especially, related to the student-centered approach adopted. To do so, the comments sections of the student evaluation forms are carefully read to observe any common points highlighted by the students.

Next section presents the results of the tasks carried under each phase of the project.

RESULTS

In this section, the results of the tasks carried out under each phase of the project are presented.

Phase 1 Results: Recall that the purpose of Phase 1 was to prepare an outline for the syllabus of the course and to do so, two tasks were described: Task 1.1. and Task 1.2.

Task 1.1. Outcomes: Task 1.1 focused on acquiring engineering faculty input on mathematical programming concepts. As explained previously, a survey given in Appendix 1 is conducted among the engineering faculty of Missouri S&T. There were 22 responses and 21 of these said they use mathematical programming/operations research/optimization concepts and/or tools in their research. Figure 3 below shows the departments of the respondents.

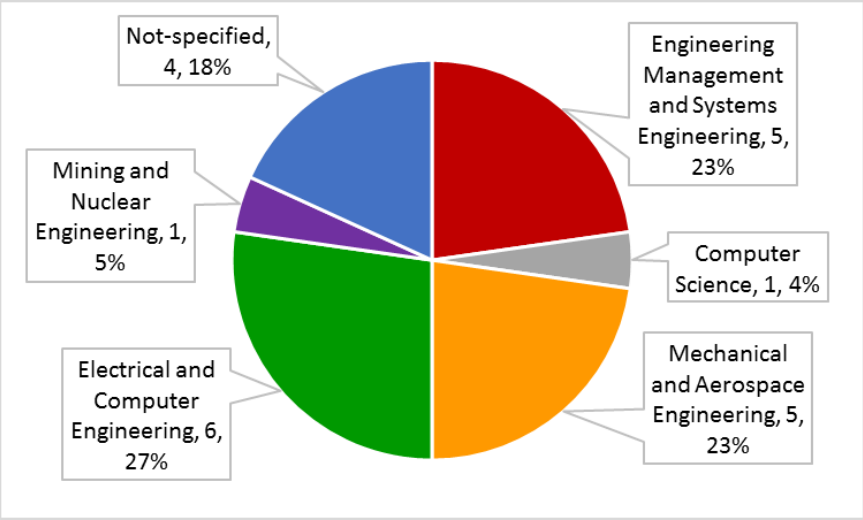

Figure 3. *Distribution of the responses by department*

Next, we summarize the results of the responses to part 3 (rating the necessity of the main topics listed) of the survey. Figure 4 shows the results. The average ranking of these main topics are as follows: 3.73 for Mathematical modeling (formulating problems); 2.95 for linear programming concepts and tools; 2.59 for integer/mixed-integer programming concepts and tools; 2.73 for nonlinear programming concepts and tools; 2.52 for network optimization; 2.68 for stochastic optimization; 2.32 for robust optimization; 2.91 for multi-objective optimization; 2.45 for multi-level optimization; 2.36 for game theory; and 2.77 for heuristics. For instance, multi-objective optimization, which is not typically covered in mathematical programming courses, is noted to be an important topic by faculty. This is due to the fact that, in many practical engineering problems, multiple objectives are considered in the decision making process. A few faculty also noted other topics of interest such as dynamic programming, data fitting, and distributed optimization. Based on these results, the PI decided to include lectures on formulating problems (previously, it was briefly discussed) and multi-objective optimization (previously, it was assigned as an option students can pick for term project).

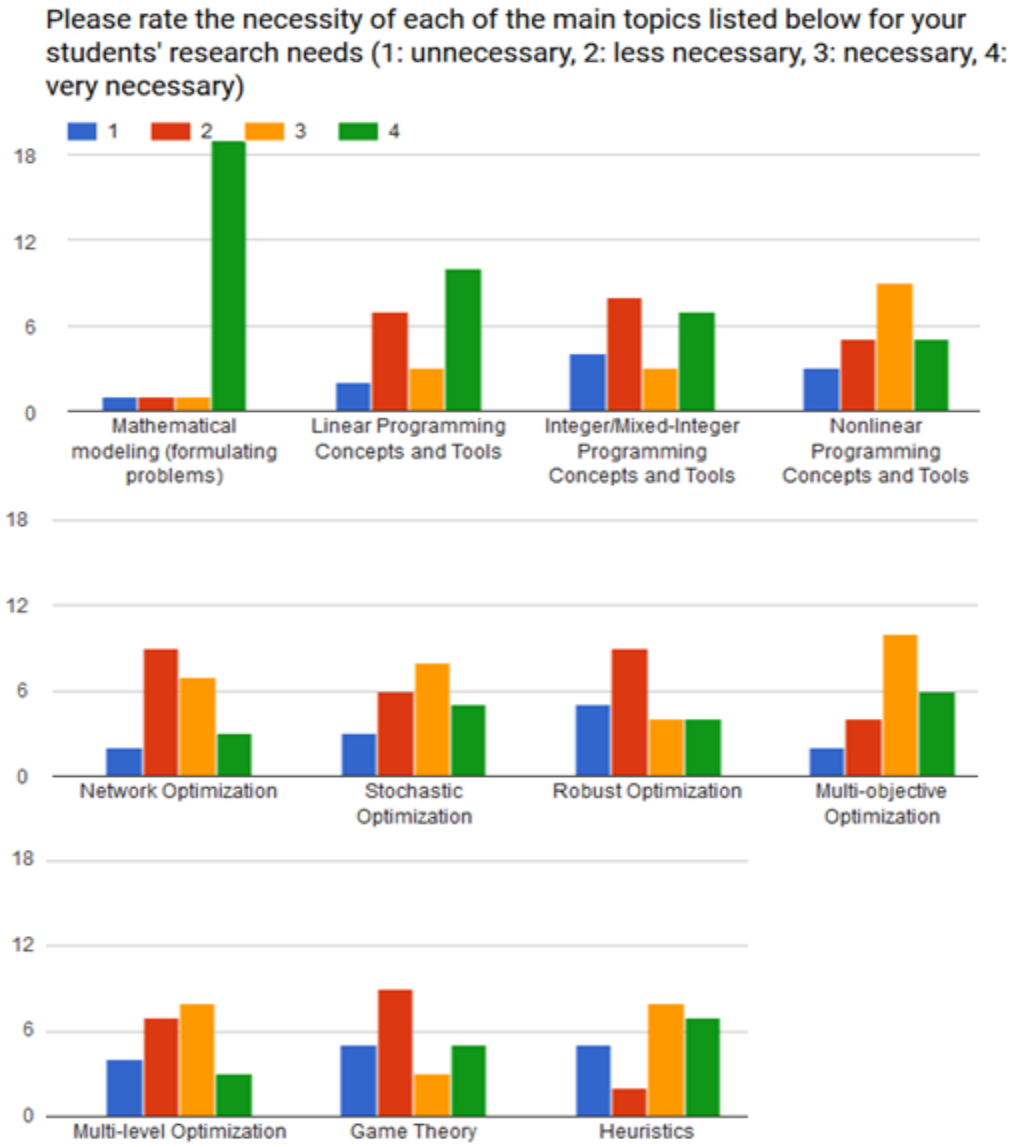

Figure 4. *Distribution of the responses to part 3 of the survey*

Finally, the summary of the responses to part 5 of the survey is given in Figure 5.

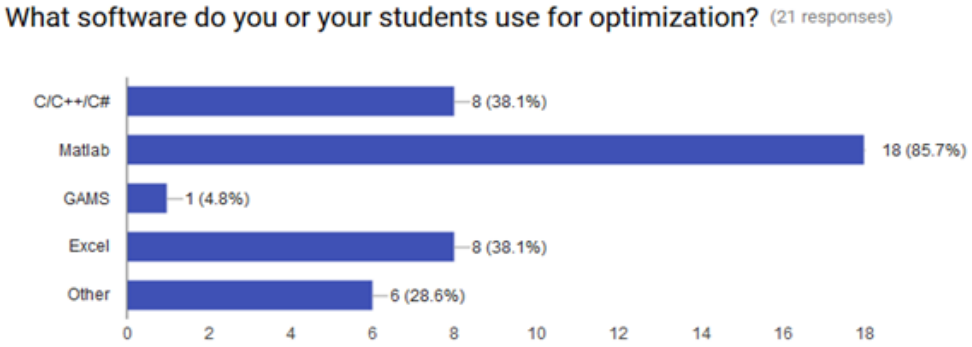

Figure 5. *Distribution of the responses to part 5 of the survey*

It can be seen that Matlab is the dominating software used for optimization, followed by Excel and C/C++/C#. In class, Matlab's basic optimization tools and Excel Solver are being demonstrated already. Other software noted to be used are Mathematica, R, and Python.

Task 1.2. Outcomes: This task focused on defining common topics and assignments for the course. Based on the instructor input and the faculty survey results, a set of topics and assignments are determined. Figure 6 shows the schedule used in the syllabus in Spring 2017.

Date		Topic	Assignment
23-Jan-17	Monday	Introduction to Mathematical Programming	
30-Jan-17	Monday	Mathematical Programming Formulation	
6-Feb-17	Monday	Linear Algebra Review and Simplex Method	HW1 Due (5%)
13-Feb-17	Monday	Simplex Method	Paper Review 1 Due (1%)
20-Feb-17	Monday	Duality	
27-Feb-17	Monday	Decomposition Principle	HW2 Due (5%)
6-Mar-17	Monday	Column Generation	Paper Review 2 Due (1%)
13-Mar-17	Monday	Integer Programming	Exam 1 Due (20%)
20-Mar-17	Monday	Branch and Bound	HW3 Due (5%)and Paper Review 3 Due (1%)
27-Mar-17	Monday	spring break	
3-Apr-17	Monday	Branch and Cut	
10-Apr-17	Monday	Mixed-Intger-Programming	Paper Review 4 Due (1%)
17-Apr-17	Monday	Bender's Decomposition	
24-Apr-17	Monday	Branch and Price	HW4 Due (5%) and Paper Review 5 Due (1%)
1-May-17	Monday	Other Topics	Project and Presentation (10%)
8-May-17	Monday	No class	Exam 2 Due (20%)

Figure 6. *Common topics and assignments determined*

The other topics included multi-objective optimization, which was suggested by the output of the faculty survey. Similarly, two weeks are allocated to formulating problems based on the faculty survey. The common assignments included exams and homework on the topics covered in class, paper reviews of various papers to maximize exposure of students to various topics that are not necessarily covered in class, and projects heavily focused on coding. 25% of the final grade based on individual assignments. These assignments described in the syllabus as follows:

- ***40% Exams*:** There will be 2 take home exams, each is 20%. Students will be given one week to analyze different mathematical programming problems. Exams will be done individually. You will submit exams on canvas. Pre-scheduled exam dates are noted on the tentative schedule below (note that these dates might change depending on the materials covered).
- ***20% Homework:*** Students will be given 4 homework sets throughout the semester, for which they need to submit their solutions. The instructor will grade one or more of the questions out of the given set for each homework. The submission dates for the homework are noted on the tentative schedule below (note that these dates might change depending on the materials covered).
- ***10% Project and Presentation:*** Students can form teams of 2 or work individually. Each project will have two parts: a coding part and a research part. The coding part will

be coding of the methods learned in class. The research part will be about learning some well-known methods, which are not discussed in class. A code for the coding part, a report (5 pages max) and a presentation (15 minutes max) for the research part will be returned by the students. The presentation will be given in class on the last day of the classes. The submission dates for the projects are noted on the tentative schedule below (note that these dates might change depending on the materials covered).

- ***5% Paper Reviews:*** Students will be assigned scholarly articles related to application of the methods discussed in class and/or articles that analyze applied problems using mathematical programming tools. For each paper reviewed, the students will write at most 2 pages review report explaining the problem analyzed and the solution methods adopted. The submission dates for the paper reviews are noted on the tentative schedule below (note that these dates might change depending on the materials covered).
- ***25% Individual Assignments:*** This course aims to help students to learn as much as possible on basic mathematical programming concepts as well as the topics of interest to the individual students. At the beginning of the semester, each student will discuss with the instructor to determine a topic, which will not be covered in class in detail. Then, the instructor will assign individual assignments on the topic to help the student learn the topic correctly and assess the student's knowledge on the selected topic. The topic will be an optimization-related topic and it can be related to the student's research, learning interest, future plans, professional needs, etc. In case the student does not have a specific topic of interest, the instructor and the student will decide on a topic together for further investigation.

Prior to posting the final syllabus, the PI of the project announced the course throughout the campus using a course flyer. This flyer included information about the concept of student-centered syllabus so that the students knew the basic idea of individual topics before enrolling in the course. Appendix 4 shows the flyer used.

There were 9 students in total enrolled in the class, and 3 of them were distance students. One student did not return satisfaction results at the end of the semester (Task 3.1), therefore, that student is excluded from the analyses in the rest of the project report.

Phase 2 Results: Recall that the purpose of Phase 2 was to finalize the syllabus based on student input and to do so, two tasks are carried out: Task 2.1. and Task 2.2.

Task 2.1. Outcomes: This task focused on collecting students' input on their learning expectations and individual topics. Appendix 5 gives the learning expectations of the students on the topics included in the list (pink cells under the expectation columns).

Task 2.2. Outcomes: After students' inputs are collected at the beginning of the semester, Task 2.2 focused on identifying the individual learning topics for each student. Specifically, each student was able to individually determine a further refined topic as his/her learning objective. Table 3 gives the individual topics defined by each student.

Table 3. *Individual topics determined by the students*

Student	Individual Topic
1	Bi-objective optimization
2	Multi-commodity min-cost flow problem
3	Robust linear programming
4	Node selection in branch and bound
5	Minimum spanning tree problem
6	Particle swarm optimization
7	curve fitting optimization
8	k-means clustering optimization

As noted before, each student were assigned 4 or 5 individual assignments consisting of: problem description and mathematical formulation, theoretical analyses and properties of the problem. Depending on the topic, the due dates for the assignments are determined for each student individually. Typically, problem description of formulation is followed by theoretical analyses, which is followed by coding and implementation. Please note that these individual assignments consisted of the 25% of the final grade as noted before. The PI assisted each student for each individual assignment as the semester progressed.

Phase 3 Results: Recall that the purpose of Phase 3 was to evaluate the student-centered syllabus from the students' perspective. To do so, three tasks are carried out: Task 3.1, Task 3.2, Task 3.3.

Task 3.1. Outcomes: This task focused on collecting students' input on their learning satisfactions. Appendix 5 gives the learning satisfactions of the students on the topics included in the list (pink cells under the satisfaction columns).

Task 3.2. Outcomes: This task compares the students' expectation levels to satisfaction levels. Appendix 5 gives the learning satisfactions of the students on the topics included in the list (blue cells under the comparison columns). Furthermore, for each main category, average comparison result is calculated by taking the average of the comparison results of the topics under that category (purple cells under the comparison columns). These average comparison values are used in further analyses. In particular, Table 4 documents the average comparison values for each main category and each student.

Table 4. *Average comparison levels for each category and each student*

		STUDENT								**Overall Avg. for each category**
		1	2	3	4	5	6	7	8	
CATEGORY	FORMULATING MODELS (FOR)	1	0	0	0	0.5	2	2	1.67	**0.90**
	LINEAR PROGRAMMING (LP)	1.5	0.5	0	-0.17	1.83	0.83	1	-1.2	**0.54**
	INTEGER PROGRAMMING (IP)	2	0.2	4	0	1.6	3.2	2.6	2	**1.95**
	MIXED-INTEGER PROGRAMMING (MIP)	3	-0.4	4	-1.2	2.2	1.6	1	2	**1.53**
	NON-LINEAR PROGRAMMING (NLP)	2	0.2	-1	-2	1	1.4	1	-3	**-0.05**
	MULTI-OBJECTIVE OPTIMIZATION (MOP)	0.25	1.25	4	-2	0.75	1.25	1.25	1	**0.97**
	MULTI-LEVEL OPTIMIZATION (MLP)	1	-0.5	3	-2.25	-0.5	1	2	1	**0.59**
	STOCHASTIC OPTIMIZATION (SOP)	-1.75	-2.25	3	-2	-0.5	2	-1	-1	**-0.44**
	ROBUST OPTIMIZATION (ROP)	0.25	-0.5	4	-2.25	-0.5	2	-1	-2	**0**
	Overall Avg. for each student	**1.03**	**-0.17**	**2.33**	**-1.32**	**0.71**	**1.70**	**0.98**	**0.05**	**0.66**

These results are evaluated using the implication table given in Table 2. Based on the implications, one can note that:

- Overall averages for each category imply that learning expectations of the students for each category are either met or exceeded. Specifically, the values under the overall avg. for each category column in Table 4 exceed 1 for 2 categories and fall between 1 and -1 for the other categories.
- Overall averages for each student imply that overall learning expectations of most of the students are either met or exceeded. Specifically, the values in the overall avg. for each student row in Table 4 exceed 1 for 3 students and fall between 1 and -1 for 4 students. Only for one student, the overall learning expectations are failed.

To further evaluate, the number of students for each implication level under each main category and the number of categories for each implication level under each student are determined. Tables 5 and 6 present these results.

Table 5. *Number of students whose learning expectations are exceeded, met, and failed for each category based on the average comparison levels*

		Number of Students		
		EXCEED	MEET	FAIL
CATEGORY	FORMULATING MODELS (FOR)	4	4	0
	LINEAR PROGRAMMING (LP)	2	5	1
	INTEGER PROGRAMMING (IP)	6	2	0
	MIXED-INTEGER PROGRAMMING (MIP)	5	2	1
	NON-LINEAR PROGRAMMING (NLP)	2	3	3
	MULTI-OBJECTIVE OPTIMIZATION (MOP)	4	3	1
	MULTI-LEVEL OPTIMIZATION (MLP)	2	5	1
	STOCHASTIC OPTIMIZATION (SOP)	2	3	3
	ROBUST OPTIMIZATION (ROP)	2	4	2

Table 6. *Number of categories for which students' learning expectations are exceeded, met, and failed for each student based on the average comparison levels*

	Number of Categories		
	EXCEED	MEET	FAIL
Student 1	4	4	1
Student 2	1	7	1
Student 3	6	3	0
Student 4	0	3	6
Student 5	3	6	0
Student 6	7	2	0
Student 7	4	4	0
Student 8	3	3	3

Based on the Tables 5 and 6, one can note the following observations:

- For most of the categories, most of the students' learning expectations are exceeded or met. There are two categories with 3 students' expectations failed and these categories are non-linear programming and stochastic optimization, which are not covered in class. In following semesters, the instructor plans to include introductions to non-linear programming concepts and tools.
- For most of the students, their learning expectations are exceeded or met for most of the categories. There are two students whose expectations are failed with more categories that the categories with exceeded or met expectations. Particularly, one student's expectations have not been satisfied with 6 categories, which is an outlier. The PI believes that this is more for an individual student rather than being a common for each student, as supported by the results in Tables 4-6.

Finally, in Figure 7, the graph of the data presented in Table 4 is shown.

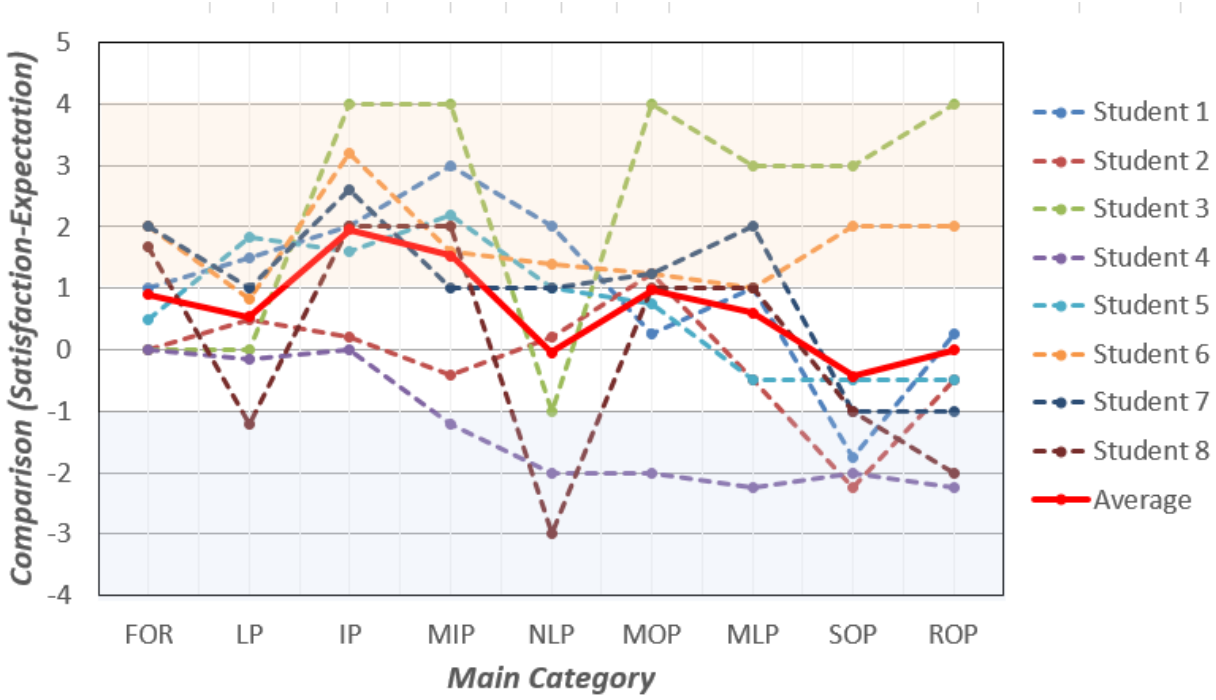

Figure 7. *Average comparison levels for each student and each category*

As can be seen in Figure 7, most of the category points for each student fall within the white (meet expectations) or pink (exceed expectations) regions. This figure highlights the categories that need attention. Particularly, for non-linear programming (NLP), stochastic optimization (SOP), and maybe robust optimization (ROP), a significant number of students' expectations are not met. Therefore, the PI plans to include introductions on these topics in the next semesters he will be teaching EMGT 6412/MATH 6665 – Mathematical Programming course.

Task 3.3. Outcomes: This task aimed at determining if there are common comments given by the students, especially, related to the student-centered approach adopted. Looking at the

teaching evaluation comments, there were only three specific comments related to student-centered syllabus approach:

- Comment 1: "… I think the assignment due date for individual assignment and research project could be set that one is during the midterm and one is by the end…"
- Comment 2: "… individual assignment is more interesting that the research project.."
- Comment 3: "…Strength: let students dictate important material according to their research.."

Based on Comment 1, which about scheduling the individual assignments, the deadlines can be planned better. Comments 2 and 3 are positive comments. Unfortunately, due to the size of the class, there was not any other comment related to the student-centered syllabus approach.

CONCLUSION/FUTURE IMPLICATIONS/PLANS FOR FURTHER DISSEMINATION

The project focused on developing a methodology to design a student-centered syllabus and evaluate the students' reactions to such a syllabus. The main motivation of the project was to help students learn individually defined objectives in addition to the learning objectives defined by the instructor. This approach is especially important for courses that might include a wide range of concepts, which cannot be taught in one semester, and for courses, which are taken by students from different programs with different learning objectives. Particularly, the mathematical programming course is a good example of such a course as mathematical programming concepts, as much as they are interdisciplinary, they are too broad to teach in one semester. And students trying to learn concepts on their own can have problems. Therefore, a student-centered syllabus can help an instructor overcome these issues.

The methodology to develop and evaluate a student-centered syllabus consisted of three phases. The first phase focused on creating an outline, with the topics to discuss in class, for the syllabus considering the instructor's expertize on the subject as well as the input from faculty. The second phased focused on collecting students' input and individual learning objectives, and use these to finalize the syllabus by determining common assignments on common topics as well as individual topics and individual assignments. The last phase focused on evaluating the students' reactions by comparing their learning expectations and learning satisfactions on various concepts related to the subject.

In conclusion, the PI can note that the students enjoyed the student-centered syllabus approach as most of their learning expectations on various categories have been exceeded or met. It is important to note that some of the categories were not discussed in class by the instructor. The students' positive reactions have been reflected on the teaching evaluation scores, which are 3.80/4 for on-campus section and 4/4 for distance section and both sections had 100% response rate.

The future work includes to use the student-centered syllabus approach for the same course in future semesters in order to collect more data. Also, the PI will work on improving the methodology by better scheduling the individual assignments.

The project outcomes so far are published as one conference proceedings paper [8], presented in one international conference, and presented in the TLT conference in 2017. After collecting further data, the PI plans to convert the research in this project to a full journal article.

The PI acknowledges the support of the Center for Educational Research and Teaching Innovation and the Engineering Management and Systems Engineering Department at the Missouri University of Science and Technology.

REFERENCES

1. "mathematical programming." A Dictionary of Computing. Encyclopedia.com. 12 January 2017, url: http://www.encyclopedia.com
2. NEOS, 2017, "Optimization Taxonomy", url: https://neos-guide.org/content/optimization-taxonomy
3. http://web.mst.edu/~konurd/6412_files/EMGT%206412_Spring%202016.pdf
4. Bazarra, M.S., Jarvis, J.J., and Sherali, H.D., 2009, Linear Programming and Network Flows, 4th Edition, John Wiley & Sons, Hoboken, New Jersey.
5. Dosch, M. and Zidon, M., 2014, "The Course Fit Us: Differentiated Instruction in the College Classroom," International Journal of Teaching and Learning in Higher Education, 26 (3), 343-357.
6. College of Engineering and Computing, Missouri University of Science & Technology, http://cec.mst.edu/
7. https://docs.google.com/forms/d/e/1FAIpQLSdQcEVOaUFNtU4LUNFrBTjJnC3gAyXN_Rb_7mc0dCVpi7JwSQ/viewform
8. Konur, Dinçer. "Towards Developing a Student-Centered Optimization Course Syllabus." In *IIE Annual Conference. Proceedings*, pp. 476-481. Institute of Industrial and Systems Engineers (IISE), 2017.

APPENDIX

Appendix 1: Survey to acquire faculty input on mathematical programming

Mathematical Programming Concepts

This scale-based survey is designed to get input from Missouri S&T faculty for determining the most crucial mathematical programming/optimization concepts that might be of interest to the research needs of the graduate students. The survey results will be used in developing the syllabus for EMGT 6412/MATH 6665 - Mathematical Programming course. The survey is conducted as part of the project "Student-centered Dynamic Syllabus Development for Mathematical Programming" sponsored by the Center for Educational Research and Teaching Innovation (CERTI), Missouri S&T. The main purpose of the project is to help graduate students learn the mathematical programming concepts that they need the most rather than concepts defined by the instructor. If you have any questions, please contact me at konurd@mst.edu or office phone 573-341-7256.

Please select your department.

Choose

Do you use mathematical programming/operations research/optimization concepts and/or tools in your research?

- ○ Yes
- ○ No

Please rate the necessity of each of the main topics listed below for your students' research needs (1: unnecessary, 2: less necessary, 3: necessary, 4: very necessary)

	1	2	3	4
Mathematical modeling (formulating problems)	○	○	○	○
Linear Programming Concepts and Tools	○	○	○	○
Integer/Mixed-Integer Programming Concepts and Tools	○	○	○	○
Nonlinear Programming Concepts and Tools	○	○	○	○
Network Optimization	○	○	○	○
Stochastic Optimization	○	○	○	○
Robust Optimization	○	○	○	○
Multi-objective Optimization	○	○	○	○
Multi-level Optimization	○	○	○	○
Game Theory	○	○	○	○
Heuristics	○	○	○	○

Please list any other mathematical programming/optimization concepts that your students might need.

Your answer

What software do you or your students use for optimization?

- [ ] C/C++/C#
- [ ] Matlab
- [ ] GAMS
- [ ] Excel
- [ ] Other: _____

Any comments:

Your answer

SUBMIT

Appendix 2: List used to collect students' learning expectations on various topics

Topic	Expectation
FORMULATING MODELS	1 to 5
Formulations steps and modeling	
Continuous vs. Integer variables	
Conditional constraints	
Overall learning	
LINEAR PROGRAMMING (LP)	1 to 5
Properties of LP models	
Simpex method	
LP Duality	
Decomposition principles	
Column generation	
Overall learning	
INTEGER PROGRAMMING (IP)	1 to 5
Properties of IP models	
Linear relaxation relations	
Branch and bound method	
Branch and cut method	
Overall learning	
MIXED-INTEGER PROGRAMMING (MIP)	1 to 5
Properties of MIP models	
Relation to IP models	
Branch and Price	
Bender's Decomposition	
Overall learning	
NON-LINEAR PROGRAMMING (NLP)	1 to 5
Properties of NLP models	
Global/unconstrainted optimization	
KKT Conditions	
Lagrangean duality	
Overall learning	
MULTI-OBJECTIVE OPTIMIZATION (MOP)	1 to 5
Properties of MOP models	
Basic solution concepts	
Detailed analysis	
Overall learning	
MULTI-LEVEL OPTIMIZATION (MLP)	1 to 5
Properties of MLP models	
Basic solution concepts	
Detailed analysis	
Overall learning	
STOCHASTIC OPTIMIZATION (SOP)	1 to 5
Properties of SOP models	
Basic solution concepts	
Detailed analysis	
Overall learning	
ROBUST OPTIMIZATION (ROP)	1 to 5
Properties of ROP models	
Basic solution concepts	
Detailed analysis	
Overall learning	

Expectation scale: 1 to 5

1: is not aware of the concept
2: is aware but was not expecting to learn
3: is expecting at least an introduction
4: is expecting more than an introduction
5: is expecting to learn

Please give comments on topics, which are not listed bu you would like to learn

*Please use more space if needed.

Appendix 3: A sample list with learning satisfaction ranking

Topic	Expectation	Satisfaction
FORMULATING MODELS	1 to 5	1 to 5
Formulations steps and modeling	4	5
Continuous vs. Integer variables	4	5
Conditional constraints	4	5
Overall learning	4	5
LINEAR PROGRAMMING (LP)	1 to 5	1 to 5
Properties of LP models	4	5
Simpex method	4	5
LP Duality	3	5
Decomposition principles	3	5
Column generation	3	5
Overall learning	4	5
INTEGER PROGRAMMING (IP)	1 to 5	1 to 5
Properties of IP models	3	5
Linear relaxation relations	3	5
Branch and bound method	3	5
Branch and cut method	3	5
Overall learning	3	5
MIXED-INTEGER PROGRAMMING (MIP)	1 to 5	1 to 5
Properties of MIP models	2	5
Relation to IP models	2	5
Branch and Price	2	5
Bender's Decomposition	2	5
Overall learning	2	5
NON-LINEAR PROGRAMMING (NLP)	1 to 5	1 to 5
Properties of NLP models	2	4
Global/unconstrainted optimization	2	4
KKT Conditions	2	4
Lagrangean duality	2	4
Overall learning	2	4
MULTI-OBJECTIVE OPTIMIZATION (MOP)	1 to 5	1 to 5
Properties of MOP models	5	5
Basic solution concepts	5	5
Detailed analysis	4	5
Overall learning	5	5
MULTI-LEVEL OPTIMIZATION (MLP)	1 to 5	1 to 5
Properties of MLP models	2	3
Basic solution concepts	2	3
Detailed analysis	2	3
Overall learning	2	3
STOCHASTIC OPTIMIZATION (SOP)	1 to 5	1 to 5
Properties of SOP models	5	3
Basic solution concepts	5	3
Detailed analysis	4	3
Overall learning	5	3
ROBUST OPTIMIZATION (ROP)	1 to 5	1 to 5
Properties of ROP models	3	3
Basic solution concepts	3	3
Detailed analysis	2	3
Overall learning	3	3

Expectation scale: 1 to 5

1: is not aware of the concept
2: is aware but was not expecting to learn
3: is expecting at least an introduction
4: is expecting more than an introduction
5: is expecting to learn

Satisfaction scale: 1 to 5

1: was not satisfied at all
2: below my expectation
3: ok but would not mind more
4: fairly satisfied
5: satisfied

Please give comments on individual topics you wanted to learn

*Please use more space if needed.

Appendix 4: Course flyer used to summarize the outline of the syllabus

ENG MGT 6412 – MATH 6665 MATHEMATICAL PROGRAMMING

SPRING 2017 by Dr. Dincer Konur

(On-campus and Distance sections, Mondays 7:00pm-9:30pm)

with student-centered syllabus (see below)

(sponsored by the Center for Educational Research and Teaching Innovation)

Course Introduction:

- Mathematical Programming (operations research-optimization-management science) has a very wide range of applications in engineering and sciences. Mathematical programming tools enable modeling and solving large and complex engineering and science optimization problems.
- This course will review: the basic steps of developing mathematical programming models, the basic theoretical foundations of mathematical programing, and the basic solutions methods available.

Learning Objectives:

- *General learning objectives*: Formulating optimization models, linear programming analysis, integer programming analysis, mixed-integer-programming analysis, main solution concepts and methods, and basic software use for optimization (Matlab mostly).
- *Individual learning objectives:* Set by the student based on his/her research needs/interests and/or professional goals/plans for learning other optimization topics such as nonlinear programming, stochastic optimization, robust optimization, multi-objective/multi-level optimization, heuristic methods, specific optimization problems analysis.

In-class topics to be covered:

- Formulating mathematical programming models
- Linear Programming
  - Linear algebra review
  - Simplex method
  - Duality
  - Decomposition principle overview
  - Column generation overview
- Integer Programming
  - Properties and integrality
  - Branch & bound, branch & cut
- Mixed-integer linear programming
  - Basic properties, branch & price
  - Bender's decomposition, heuristics

Student-defined topics might include but not limited to:

- Basic properties and solution concepts for:
  - Non-linear programming
  - Multi-objective/multi-level optimization
  - Stochastic optimization
  - Robust optimization
- Heuristic methods
- Specific optimization problem analysis:
  - Formulation and solution alternatives

Student-Centered Syllabus: Mathematical programming (optimization) is a very broad discipline and all concepts and tools cannot be taught in one semester. The idea of student-centered syllabus is that the instructor and the student jointly determine the learning objectives individual to each student. This way, in addition to the basic knowledge of mathematical programming (see in-class topics), a student will set additional learning objectives for other mathematical programming concepts (see possible student-defined topics) based on his/her or his/her advisor's research needs, interests, and goals. In addition to general assignments, each student will have individual assignments based on their topics of choice. In case a student does not have individual learning goals, the instructor will define additional topics.

Course Requirements: 4 homework (20%), 2 take-home exams (40%), term project report and presentation (10%), five paper-reviews (5%), and individual-assignments (25%).

Text-book: Linear Programming and Network Flows (4th Ed.) by Bazaraa, Jarvis, and Sherali (WILEY).

Instructor: Dr. Dincer Konur is an Assistant Professor in Engineering Management and Systems Engineering. His research area is optimization and game theory in supply chain, logistics, and systems design. His holds PHD and MS degrees in Industrial and Systems Engineering from University of Florida.

Webpage: http://web.mst.edu/~konurd/ *Email:* konurd@mst.edu *Phone:* 573-341-7256 *Office:* 206 EMAN

Appendix 5 (part 1): Student Expectation, Satisfaction, and Comparison Results: Students 1-4

		Student 1			Student 2			Student 3			Student 4		
	Topic	Expectation	Satisfaction	Comparison	Expectation	Satisfaction	Comparison	Expectation	Satisfaction	Comparison	Expectation	Satisfaction	Comparison
Category 1:	FORMULATING MODELS	1 to 5	1 to 5	**1**	1 to 5	1 to 5	**0**	1 to 5	1 to 5	**0**	1 to 5	1 to 5	**0**
	Formulations steps and modeling	4	5	1	5	5	0	5	5	0	5	5	0
	Continuous vs. Integer variables	4	5	1	5	5	0	5	5	0	5	5	0
	Conditional constraints	4	5	1	5	5	0	5	5	0	5	5	0
	Overall learning	4	5	1	5	5	0	5	5	0	5	5	0
Category 2:	LINEAR PROGRAMMING (LP)	1 to 5	1 to 5	**1.5**	1 to 5	1 to 5	**0.5**	5-Jan	1 to 5	**0**	1 to 5	1 to 5	**-0.17**
	Properties of LP models	4	5	1	5	5	0	5	5	0	5	5	0
	Simpex method	4	5	1	5	5	0	5	5	0	5	5	0
	LP Duality	3	5	2	4	5	1	5	5	0	5	5	0
	Decomposition principles	3	5	2	4	5	1	5	5	0	5	4	-1
	Column generation	3	5	2	4	5	1	5	5	0	5	5	0
	Overall learning	4	5	1	5	5	0	5	5	0	5	5	0
Category 3:	INTEGER PROGRAMMING (IP)	1 to 5	1 to 5	**2**	1 to 5	1 to 5	**0.2**	1 to 5	1 to 5	**4**	1 to 5	1 to 5	**0**
	Properties of IP models	3	5	2	5	5	0	1	5	4	5	5	0
	Linear relaxation relations	3	5	2	4	5	1	1	5	4	4	5	1
	Branch and bound method	3	5	2	4	5	1	1	5	4	4	4	0
	Branch and cut method	3	5	2	4	3	-1	1	5	4	4	3	-1
	Overall learning	3	5	2	4	4	0	1	5	4	4	4	0
Category 4:	MIXED-INTEGER PROGRAMMING (MIP)	1 to 5	1 to 5	**3**	1 to 5	1 to 5	**-0.4**	1 to 5	1 to 5	**4**	1 to 5	1 to 5	**-1.2**
	Properties of MIP models	2	5	3	5	4	-1	1	5	4	4	4	0
	Relation to IP models	2	5	3	5	5	0	1	5	4	4	3	-1
	Branch and Price	2	5	3	4	4	0	1	5	4	4	2	-2
	Bender's Decomposition	2	5	3	4	3	-1	1	5	4	4	2	-2
	Overall learning	2	5	3	4	4	0	1	5	4	4	3	-1
Category 5:	NON-LINEAR PROGRAMMING (NLP)	1 to 5	1 to 5	**2**	1 to 5	1 to 5	**0.2**	1 to 5	1 to 5	**-1**	1 to 5	1 to 5	**-2**
	Properties of NLP models	2	4	2	5	4	-1	3	3	0	4	2	-2
	Global/unconstrainted optimization	2	4	2	4	4	0	5	3	-2	4	2	-2
	KKT Conditions	2	4	2	3	4	1	5	5	0	4	2	-2
	Lagrangean duality	2	4	2	3	4	1	5	3	-2	4	2	-2
	Overall learning	2	4	2	4	4	0	5	4	-1	4	2	-2
Category 6:	MULTI-OBJECTIVE OPTIMIZATION (MOP)	1 to 5	1 to 5	**0.25**	1 to 5	1 to 5	**1.25**	1 to 5	1 to 5	**4**	1 to 5	1 to 5	**-2**
	Properties of MOP models	5	5	0	4	5	1	1	5	4	4	2	-2
	Basic solution concepts	5	5	0	4	5	1	1	5	4	4	3	-1
	Detailed analysis	4	5	1	3	5	2	1	5	4	4	1	-3
	Overall learning	5	5	0	4	5	1	1	5	4	4	2	-2
Category 7:	MULTI-LEVEL OPTIMIZATION (MLP)	1 to 5	1 to 5	**1**	1 to 5	1 to 5	**-0.5**	1 to 5	1 to 5	**3**	1 to 5	1 to 5	**-2.25**
	Properties of MLP models	2	3	1	5	4	-1	1	4	3	4	2	-2
	Basic solution concepts	2	3	1	5	4	-1	1	4	3	4	1	-3
	Detailed analysis	2	3	1	3	3	0	1	4	3	3	1	-2
	Overall learning	2	3	1	4	4	0	1	4	3	3	1	-2
Category 8:	STOCHASTIC OPTIMIZATION (SOP)	1 to 5	1 to 5	**-1.75**	1 to 5	1 to 5	**-2.25**	1 to 5	1 to 5	**3**	1 to 5	1 to 5	**-2**
	Properties of SOP models	5	3	-2	5	2	-3	1	4	3	4	2	-2
	Basic solution concepts	5	3	-2	4	2	-2	1	4	3	3	1	-2
	Detailed analysis	4	3	-1	4	2	-2	1	4	3	3	1	-2
	Overall learning	5	3	-2	4	2	-2	1	4	3	3	1	-2
Category 9:	ROBUST OPTIMIZATION (ROP)	1 to 5	1 to 5	**0.25**	1 to 5	1 to 5	**-0.5**	1 to 5	1 to 5	**4**	1 to 5	1 to 5	**-2.25**
	Properties of ROP models	3	3	0	4	3	-1	1	5	4	4	1	-3
	Basic solution concepts	3	3	0	4	3	-1	1	5	4	3	1	-2
	Detailed analysis	2	3	1	3	3	0	1	5	4	3	1	-2
	Overall learning	3	3	0	3	3	0	1	5	4	3	1	-2

Appendix 5 (part 2): Student Expectation, Satisfaction, and Comparison Results: Students 5-8

		Student 5			Student 6			Student 7			Student 8		
	Topic	Expectation	Satisfaction	Comparison	Expectation	Satisfaction	Comparison	Expectation	Satisfaction	Comparison	Expectation	Satisfaction	Comparison
Category 1:	FORMULATING MODELS	1 to 5	1 to 5	**0.5**	1 to 5	1 to 5	**2**	1 to 5	1 to 5	**2**	1 to 5	1 to 5	**1.67**
	Formulations steps and modeling	5	5	0	3	5	2	3	5	2	2	3	1
	Continuous vs. Integer variables	4	5	1	3	5	2	3	5	2	2	4	2
	Conditional constraints	4	4	0	3	5	2	3	5	2	2	4	2
	Overall learning	4	5	1	3	5	2	3	5	2	2	4	1.67
Category 2:	LINEAR PROGRAMMING (LP)	1 to 5	1 to 5	**1.83**	1 to 5	1 to 5	**0.83**	1 to 5	1 to 5	**1**	1 to 5	1 to 5	**-1.20**
	Properties of LP models	4	5	1	5	5	0	4	4	0	5	4	-1
	Simpex method	1	4	3	5	5	0	5	4	-1	5	4	-1
	LP Duality	1	4	3	4	5	1	1	4	3	5	4	-1
	Decomposition principles	1	3	2	3	4	1	1	3	2	5	3	-2
	Column generation	1	3	2	3	4	1	3	4	1	5	4	-1
	Overall learning	4	4	0	3	5	2	3	4	1	5	4	-1.2
Category 3:	INTEGER PROGRAMMING (IP)	1 to 5	1 to 5	**1.6**	1 to 5	1 to 5	**3.2**	1 to 5	1 to 5	**2.6**	1 to 5	1 to 5	**2**
	Properties of IP models	4	4	0	3	5	2	4	5	1	2	4	2
	Linear relaxation relations	1	3	2	1	5	4	1	5	4	2	4	2
	Branch and bound method	1	5	4	1	5	4	1	5	4	2	4	2
	Branch and cut method	1	3	2	1	4	3	1	3	2	2	4	2
	Overall learning	4	4	0	2	5	3	3	5	2	2	4	2
Category 4:	MIXED-INTEGER PROGRAMMING (MIP)	1 to 5	1 to 5	**2.2**	1 to 5	1 to 5	**1.6**	1 to 5	1 to 5	**1**	1 to 5	1 to 5	**2**
	Properties of MIP models	1	3	2	3	4	1	3	4	1	2	4	2
	Relation to IP models	1	4	3	3	4	1	3	3	0	2	4	2
	Branch and Price	1	4	3	1	4	3	1	3	2	2	4	2
	Bender's Decomposition	1	3	2	1	3	2	1	3	2	2	4	2
	Overall learning	3	4	1	3	4	1	3	3	0	2	4	2
Category 5:	NON-LINEAR PROGRAMMING (NLP)	1 to 5	1 to 5	**1**	1 to 5	1 to 5	**1.4**	1 to 5	1 to 5	**1**	1 to 5	1 to 5	**-3**
	Properties of NLP models	3	3	0	2	3	1	2	3	1	5	2	-3
	Global/unconstrainted optimization	3	3	0	2	3	1	2	3	1	5	2	-3
	KKT Conditions	1	4	3	1	3	2	2	3	1	5	2	-3
	Lagrangean duality	1	3	2	1	3	2	2	3	1	5	2	-3
	Overall learning	3	3	0	2	3	1	2	3	1	5	2	-3
Category 6:	MULTI-OBJECTIVE OPTIMIZATION (MOP)	1 to 5	1 to 5	**0.75**	1 to 5	1 to 5	**1.25**	1 to 5	1 to 5	**1.25**	1 to 5	1 to 5	**1**
	Properties of MOP models	1	4	3	2	4	2	2	4	2	2	3	1
	Basic solution concepts	3	3	0	2	3	1	2	3	1	2	3	1
	Detailed analysis	3	3	0	2	3	1	2	3	1	2	3	1
	Overall learning	3	3	0	2	3	1	2	3	1	2	3	1
Category 7:	MULTI-LEVEL OPTIMIZATION (MLP)	1 to 5	1 to 5	**-0.5**	1 to 5	1 to 5	**1**	1 to 5	1 to 5	**2**	1 to 5	1 to 5	**1**
	Properties of MLP models	1	1	0	2	3	1	1	3	2	2	3	1
	Basic solution concepts	1	1	0	2	3	1	1	3	2	2	3	1
	Detailed analysis	1	1	0	2	3	1	1	3	2	2	3	1
	Overall learning	3	1	-2	2	3	1	1	3	2	2	3	1
Category 8:	STOCHASTIC OPTIMIZATION (SOP)	1 to 5	1 to 5	**-0.5**	1 to 5	1 to 5	**2**	1 to 5	1 to 5	**-1**	1 to 5	1 to 5	**-1**
	Properties of SOP models	1	1	0	1	3	2	3	2	-1	5	4	-1
	Basic solution concepts	1	1	0	1	3	2	3	2	-1	5	4	-1
	Detailed analysis	1	1	0	1	3	2	3	2	-1	5	4	-1
	Overall learning	3	1	-2	1	3	2	3	2	-1	5	4	-1
Category 9:	ROBUST OPTIMIZATION (ROP)	1 to 5	1 to 5	**-0.5**	1 to 5	1 to 5	**2**	1 to 5	1 to 5	**-1**	1 to 5	1 to 5	**-2**
	Properties of ROP models	1	1	0	1	3	2	3	2	-1	5	3	-2
	Basic solution concepts	1	1	0	1	3	2	3	2	-1	5	3	-2
	Detailed analysis	1	1	0	1	3	2	3	2	-1	5	3	-2
	Overall learning	3	1	-2	1	3	2	3	2	-1	5	3	-2