Chapter 1
Linear Programming

Paragraph 7
Standard Formats MPS, LP, and the CPLEX callable Library

What we did so far

- We studied algorithms for solving linear programs
  - Simplex (primal, dual, and primal-dual)
  - Ellipsoid Method
  - Interior Point Algorithms
- Why is it called: linear programming?
  - You do not need to implement these algorithms anymore, there exists standard solvers!
  - Therefore, to “solve” a linear programming problem, all you need is to formulate the LP and hand it over to a solver.

“Languages for Linear Programming”

- Keeping the general terminology, the “languages” of linear programming are modeling languages in which we can express linear programs.
- Probably one of the oldest formats in which LPs can be stated is the MPS format from IBM.
- A more intuitive and also widely used format is the LP-Format.
- In order to reduce the overhead, solvers like CPLEX specify their own callable library interface.

The MPS Format

- Developed during the punch card era, MPS is column oriented:

	Field 1	Field 2	Field 3	Field 4	Field 5	Field 6
Columns	2-3	5-12	15-22	25-36	40-47	50-61
Contents	Indicator	Name	Name	Value	Name	Value

The MPS Format

- The input is further segmented horizontally by
  - a ROWS section and a COLUMNS section.
  - It may also contain optional NAME, RHS, RANGES, and BOUNDS sections.
  - These keywords are the only parts of the MPS-file that start in column 1.
  - Each section in an MPS format input must contain at least one data line, with the exception of the NAME section.
- All MPS files must terminate with an ENDATA line.

The MPS Format

1	2-3	4	5-12	15-22	25-36	40-47	50-61
N	AM	E					
R	OW	S					
C	OL	U	MNS				
R	HS						
E	ND	D	ATA				

The MPS Format

- The NAME section
  - Keyword NAME in columns 1-4
  - Title of the LP is columns 15-22
- The ROWS section
  - Purpose: label each row and assign a type
  - Row-type in columns 2-3:
    - E - =
    - L - <=
    - G - >=
    - N - first such row: objective, otherwise no restriction
  - Assign a name in columns 5-12

The MPS Format

1	2-3	4	5-12	15-22	25-36	40-47	50-61
N	AM	E		example			
R	OW	S					
	N		obj				
	L		r1				
	G		r2				
C	OL	U	MNS				
R	HS						
E	ND	D	ATA				

The MPS Format

- The COLUMNS section
  - Purpose: Specify the tableau column by column
  - Label each column (for example with the name of the corresponding variable) in columns 5-12
  - For each non-zero entry of the current tableau-column, give the entry by first giving the row label and then the numerical value.
  - At most two entries can be made per row in the MPS-file. For more entries, continue the specification of the current column in the next rows.

The MPS Format

1	2-3	4	5-12	15-22	25-36	40-47	50-61
N	AM	E		example			
R	OW	S					
	N		obj				
	L		r1				
	G		r2				
C	OL	U	MNS				
			x	obj	1	r1	1
			x	r2	2		
			y	obj	-2.3	r1	-1
			z	obj	0.5	r2	-1
			s	r2	-1	r1	1
R	HS						
E	ND	D	ATA				

The MPS Format

- The RHS section
  - Purpose: Specify the ride hand side of the tableau
  - Like every other column and row, also the ride hand side gets a label in columns 5-12
  - For each non-zero entry of the right hand side, the entry is specified by first giving the row label and then the numerical value.
  - At most two entries can be made per row in the MPS-file. For more entries, continue the specification of the right hand side in the next rows.

The MPS Format

1	2-3	4	5-12	15-22	25-36	40-47	50-61
N	AM	E		example			
R	OW	S					
	N		obj				
	L		r1				
	G		r2				
C	OL	U	MNS				
			x	obj	1	r1	1
			x	r2	2		
			y	obj	-2.3	r1	-1
			z	obj	0.5	r2	-1
			s	r2	-1	r1	1
R	HS						
			rhs	r1	10.75	r2	-100
E	ND	D	ATA				

The MPS Format

- The format is old and strange.
  - The punch-card organization of the data is outdated and annoying.
  - All variables are implicitly assumed to be non-negative (this could be changed in the BOUNDS section, though).
  - The objective direction is not specified!
- However, it is the industry standard, the standard benchmark MIPlib uses MPS, and most solvers accept input in this form. If a solver does not, there is a good chance that there exists a converter from MPS to the expected input format.

The LP Format

- Like MPS, a linear program in LP format is also segmented in sections:
  - Problem
  - Objective
  - Constraints and
  - Bounds (optional)
- The model starts with the keyword ‘Problem’ and is terminated with ‘End’

The LP Format

Problem
End

The LP Format

- The Problem section
  - Provide a name of the LP in the following line
- The Objective section
  - Specify the objective direction (Maximize or Minimize)
  - In the next line, specify a label for the objective followed by a colon.
  - Then give the objective.

The LP Format

Problem
example
Maximize
obj: x – 2.3y + 0.5z
End

The LP Format

- The Constraint section
  - Start with the keywords Subject To (or ST).
  - In the following lines, start adding constraints by labeling them (name plus colon) and then giving the equation or inequality.
- The Bounds section
  - Start with the keyword Bounds.
  - State the bounds of variables
    - inf, -inf, and free are keywords
    - Default is $x \geq 0$.

The LP Format

```
Problem
  example
Maximize
  obj: x – 2.3y + 0.5z
Subject To
  r1: x – y + s <= 10.75
  r2: -z + 2x – s >= -100
End
```

The CPLEX Callable Library

- The library was written in C.
- Instead of writing out an MPS or LP file from an application and having the solver read it back in, the data is better passed over within the RAM.
- Instead of passing over the entire mxn-matrix, CPLEX expects a sparse representation where only the non-zero entries are handed over.
- We only discuss the sparse vector data structure here. For more details check /com/cplex/doc/pdf/refcallablelibrary.pdf.

The CPLEX Callable Library

	0	1	2	3
0	3	0	5	9
1	6	2	0	1
2	0	0	1	3

- We would like to compress this matrix to:

#nz	0	1	2	3	4	5	6	7
rmatval	3	5	9	6	2	1	1	3

The CPLEX Callable Library

	0	1	2	3
0	3	0	5	9
1	6	2	0	1
2	0	0	1	3

- Since we lost this info, we need to add the column indices:

#nz	0	1	2	3	4	5	6	7
rmatval	3	5	9	6	2	1	1	3
rmatind	0	2	3	0	1	3	2	3

The CPLEX Callable Library

	0	1	2	3
0	3	0	5	9
1	6	2	0	1
2	0	0	1	3

rmatcnt	3	3	2
rmatbeg	0	3	6

- Finally, we need to add information where new rows start:

#nz	0	1	2	3	4	5	6	7
rmatval	3	5	9	6	2	1	1	3
rmatind	0	2	3	0	1	3	2	3

Thank you!