Sensitivity, IP

Fall 2021

Overview

Sensitivity analysis

Integer Linear Programming
- Review
- Writing Logic Statements

Examples
- Job Scheduling Problem
- Plan for a move

Terminology

- **Shadow price** y_i = Dual variable
  Change of the objective function from one unit **increase** in its right-hand side b_i
- **Reduced cost** r_j = Dual slack $= (\mathbf{c}^\top - \mathbf{y}^\top A)_j$
  Amount by which the cost coefficient of non-basic variable c_j must be **lowered** for that variable to become basic
- **Allowable increase/decrease**
  - Optimal solution x^* and objective $\sum_{j=1}^{n} c_j x_j^*$ may change
  - Whether a decision variable is basic or non-basic stays unchanged
  - Whether a constraint is binding or non-binding stays unchanged

Terminology

For a problem in symmetrical form, let $(\bar{\mathbf{x}}, \bar{\mathbf{y}})$ be primal-dual feasible point satisfying complementary slackness. Let $\bar{\mathbf{s}}$ be corresponding primal slack and $\bar{\mathbf{r}}$ be corresponding dual slack.

- Decision variable $\bar{x}_j$ is **basic** if $\bar{x}_j \neq 0$ ($\bar{r}_j = 0$ due to CS)
- Decision variable $\bar{x}_j$ is **non-basic** if $\bar{x}_j = 0$ ($\bar{r}_j < 0$ in general)
- Constraint $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$ is **binding** if $\sum_{j=1}^{n} a_{ij}\bar{x}_j = b_i$ ($\bar{s}_i = 0$)
- Constraint $\sum_{j=1}^{n} a_{ij}x_j \geq b_i$ is **not binding** if $\sum_{j=1}^{n} a_{ij}\bar{x}_j > b_i$ ($\bar{y}_i = 0$ due to CS)

Example: continuous knapsack

```
ampl: include cont_knapsack.run;
CPLEX 12.6.1.0: sensitivity
CPLEX 12.6.1.0: optimal solution; objective 85
1 dual simplex iterations (1 in phase I)

suffix up OUT;
suffix down OUT;
suffix current OUT;
:    x        x.rc       x.current   x.down     x.up      :=
1   2.5    -3.55271e-15     24          20     1e+20
2   5      -1.77636e-15      5      -1e+20        12
3   0      -0.4              2      -1e+20         2.4
4   0     -15                3      -1e+20        18
;

: _conname   _con _con.slack _con.current _con.down     _con.up      :=
1   volume    0      12.5          60        47.5      1e+20
2   weight    1.2     0           100        50          183.333
3   water    -7       0             5         0            6.92308
;
```

AMPL notation

- `x` – primal variable;
- `x.rc` – reduced cost or dual slack;
- `x.current` – objective coefficients ($\mathbf{c}_i$);
- `_conname` – shadow price or dual variable;
- `_con.slack` – primal slack;
- `_con.current` – right hand side ($\mathbf{b}_j$);
- `...down` and `...up` are the minimal and the maximal value of the corresponding parameter $\mathbf{c}_i$ or $\mathbf{b}_j$ such that the problem stays within the allowable increase/decrease range

How to derive sensitivity analysis: Key Idea

In order for a change to be withing the allowable range, both of these must be true at the solution point:

- Whether a decision variable is basic or non-basic stays unchanged.
- Whether a constraint is binding or non-binding stays unchanged.

Types of analysis

- Case 1: Change b_i
  - Case 1a: Change b_i of non-binding constraint
  - Case 1b: Change b_i of binding constraint
  - Case 1c: Find g if Case 1b.
- Case 2: Change c_j
  - Case 2a: Change c_j of non-basic variable
  - Case 2b: Change c_j of basic variable
- Case 3: Change a_{ij}
  - Case 3a: Change a_{ij} of non-basic variable
  - *Case 3b: Change a_{ij} of basic variable*
- Case 4: Add a new constraint
- Case 5: Add a new decision variable

Case 2a: Change c_j of non-basic variable

Change c_j of non-basic variable

- Reduced cost $r_j \neq 0$ [1]

$$r_j = c_j - \sum_{i=1}^{m} a_{ij} y_i$$

- Consider c_3 which has reduced cost $r_3 = -0.4$
  - Allowable increase: $-r_j = 0.4$
  - Allowable decrease: $+\infty$
- Consider changing c_3 from 2 to 2.1
  - New optimal solution: Unchanged
  - New optimal objective value: Unchanged
- It's possible to change several c_j for non-basics variables at the same time!

[1] $r_j = 0$ for non-basic variable means multiple optimal solutions

Case 2b: Change c_j of basic variable

Change c_j of basic variable

$$
\begin{array}{lrrrrcl}
\max & 24x_1 & +(5+\delta)x_2 & +2x_3 & +3x_4 & & \\
s.t. & 3x_1 & +8x_2 & +14x_3 & +6x_4 & \leq & 60 \\
 & 20x_1 & +10x_2 & +2x_3 & +15x_4 & \leq & 100 \\
 & & x_2 & & & \geq & 5 \\
 & x_1 & & & & \geq & 0 \\
 & & x_2 & & & \geq & 0 \\
 & & & x_3 & & \geq & 0 \\
 & & & & x_4 & \geq & 0
\end{array}
$$

Case 2b: Change c_j of basic variable

Consider the Dual problem:

$$
\begin{array}{lcccccr}
\min & 60y_1 & +100y_2 & +5y_3 & & & \\
s.t. & 3y_1 & +20y_2 & & \geq & 24 & (1) \\
 & 8y_1 & +10y_2 & +y_3 & \geq & 5+\delta & (2) \\
 & 14y_1 & +2y_2 & & \geq & 2 & (3) \\
 & 6y_1 & +15y_2 & & \geq & 3 & (4) \\
 & y_1 & & & \geq & 0 & (5) \\
 & & y_2 & & \geq & 0 & (6) \\
 & & & y_3 & \leq & 0 & (7)
\end{array}
$$

Optimal solution $y_1 = 0, y_2 = 1.2, y_3 = -7$

Case 2b: Change c_j of basic variable

$$
\begin{array}{lrrcll}
\min & +100y_2 & +5y_3 & & & \\
s.t. & +20y_2 & & = & 24 & (1) \\
 & +10y_2 & +y_3 & = & 5+\delta & (2) \\
 & +2y_2 & & \geq & 2 & (3) \\
 & +15y_2 & & \geq & 3 & (4) \\
 & y_2 & & \geq & 0 & (6) \\
 & & y_3 & \leq & 0 & (7)
\end{array}
$$

Case 2b: Change c_j of basic variable

From (1) we get $y_2 = \frac{24}{20}$ (satisfies (3), (4), (6)), substitute in (2)

$$y_3 = -7 + \delta.$$

From (7)

$$\delta \leq 7$$

Case 2b: Change c_j of basic variable

Change c_j of basic variable

- Reduced cost $r_j = 0$
- Consider c_2
  - Allowable increase: 7
  - Allowable decrease: $+\infty$
- Consider changing c_2 from $5 \to 10$
  - New optimal solution: Unchanged
  - New optimal objective value:

$$\sum_{j=1}^{n} c_j^{new} x_j^* = \sum_{j=1}^{n} c_j x_j^* + \delta x_2^* = 110$$

Case 4: Add a new constraint

Add a new constraint

- If current solution satisfies the new constraint
  - New optimal solution: Unchanged
  - New optimal objective value: Unchanged
- If current solution does not satisfy the new constraint
  - Dual simplex method (but don't worry about this for now)

Note: the problem might become infeasible

Note

Sensitivity analysis lets you simultaniously think about a continious set of instances of LP for which δ is within the range. The other instances still have to be considered individually.

Review

- ILP stands for Integer Linear Programming and MILP for Mixed Integer Linear Programming (notation may change depending on the reference).
- Both ILP and MILP can be seen as:

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{n} c_i x_i \\
\text{s.t.} \quad & \sum_{i=1}^{n} a_{ij} x_i \geq, \leq, = b_j \quad \forall j \in \{1, \dots m\} \\
& x_i \text{ integer} \quad \forall i \in \{1, \dots, n\} \quad \text{(ILP)} \\
& x_i \text{ integer or real} \quad \forall i \in \{1, \dots, n\} \quad \text{(MILP)}
\end{aligned}
$$

- A special case of integer is binary. Notice that x binary can also be written as $0 \leq x \leq 1$, integer.

Review

Integer variables, and in particular binary variables, are well suited to write logical statements. Usually we think of $x = 0$ as false, and $x = 1$ as true.

Super Important: Your reformulation should express no more and no less than what you are trying to express. For example, if you are trying to express an implication for one direction, you don't want to also obligate the implication on the other way or extra implications.

Logical Statements

For the following exercises assume all variables to be binary.

Logical Statements

For the following exercises assume all variables to be binary.

- Express: $x_1 = 0 \rightarrow x_2 = 1$.

Logical Statements

For the following exercises assume all variables to be binary.

- Express: $x_1 = 0 \to x_2 = 1$.
- Solution: $x_2 \geq 1 - x_1$.

Logical Statements

Remember that $A \to B$ also implies that $\neg B \to \neg A$. For the following exercises assume all variables to be binary.

Logical Statements

Remember that $A \rightarrow B$ also implies that $\neg B \rightarrow \neg A$. For the following exercises assume all variables to be binary.

- Express: (x_1 false, x_2 true) implies x_3 false.

Logical Statements

Remember that $A \to B$ also implies that $\neg B \to \neg A$. For the following exercises assume all variables to be binary.

- Express: (x_1 false, x_2 true) implies x_3 false.
- **Solution:** $x_3 \leq 2 - (1 - x_1) - x_2$

Logical Statements

Remember that $A \rightarrow B$ also implies that $\neg B \rightarrow \neg A$. For the following exercises assume all variables to be binary.

- Express: (x_1 false, x_2 true) implies x_3 false.
- **Solution:** $x_3 \leq 2 - (1 - x_1) - x_2$
- Express: (x_1 true, x_2 false) implies x_3 true.

Logical Statements

Remember that $A \rightarrow B$ also implies that $\neg B \rightarrow \neg A$. For the following exercises assume all variables to be binary.

- Express: (x_1 false, x_2 true) implies x_3 false.
- **Solution:** $x_3 \leq 2 - (1 - x_1) - x_2$
- Express: (x_1 true, x_2 false) implies x_3 true.
- **Solution:** $x_3 \geq -1 + x_1 + (1 - x_2)$

Logical Statements

Remember that $A \rightarrow B$ also implies that $\neg B \rightarrow \neg A$. For the following exercises assume all variables to be binary.

- Express: (x_1 false, x_2 true) implies x_3 false.
- **Solution:** $x_3 \leq 2 - (1 - x_1) - x_2$
- Express: (x_1 true, x_2 false) implies x_3 true.
- **Solution:** $x_3 \geq -1 + x_1 + (1 - x_2)$
- Express: $\sum_{i=1}^{n} a_i x_i > b$ implies y true . For this problem assume $\sum_{i=1}^{n} a_i x_i \leq M$ for any choice of $(x_1, \ldots, x_n)$ feasible. In this question we must think of a constraint that is always feasible.

Logical Statements

Remember that $A \rightarrow B$ also implies that $\neg B \rightarrow \neg A$. For the following exercises assume all variables to be binary.

- Express: (x_1 false, x_2 true) implies x_3 false.
- **Solution:** $x_3 \leq 2 - (1 - x_1) - x_2$
- Express: (x_1 true, x_2 false) implies x_3 true.
- **Solution:** $x_3 \geq -1 + x_1 + (1 - x_2)$
- Express: $\sum_{i=1}^{n} a_i x_i > b$ implies y true . For this problem assume $\sum_{i=1}^{n} a_i x_i \leq M$ for any choice of $(x_1, \ldots, x_n)$ feasible. In this question we must think of a constraint that is always feasible.
- **Solution:** $\sum_{i=1}^{n} a_i x_i \leq (1 - y)b + yM$

Logical Statements

- $\sum_{i=1}^{n} a_i x_i \neq b$ implies z true. Here you can assume

$$-M \leq \sum_{i=1}^{n} a_i x_i \leq M.$$

Logical Statements

- $\sum_{i=1}^{n} a_i x_i \neq b$ implies z true. Here you can assume

$$-M \leq \sum_{i=1}^{n} a_i x_i \leq M.$$

- **Solution:**

$$\begin{aligned} \sum_{i=1}^{n} a_i x_i &\leq (1-y_1)b + y_1 M & \left(\sum_{i=1}^{n} a_i x_i > b \rightarrow y_1 = 1\right) \\ \sum_{i=1}^{n} a_i x_i &\geq (1-y_2)b - y_2 M & \left(\sum_{i=1}^{n} a_i x_i < b \rightarrow y_2 = 1\right) \\ z &\geq \frac{y_1 + y_2}{2} & \end{aligned}$$

Logical Statements

Let's say we have two constraints $\sum_{i=1}^{n} a_{i1}x_i \geq b_1$ and $\sum_{i=1}^{n} a_{i2}x_i \geq b_2$ where the x's and a'are all ≥ 0. Write a constraint or set of constraints to enforce that at least one of the two inequalities is enforced at all times.

Solution: Let's introduce "y" binary variable. A set of constraints that solves the problem is:

$$\begin{aligned} \sum_{i=1}^{n} a_{i1}x_i &\geq y \cdot b_1 \\ \sum_{i=1}^{n} a_{i2}x_i &\geq (1-y)b_2 \end{aligned}$$

Because both left sides are always ≥ 0.

Logical Statements

Let x be a non-negative real variable, and assume that if $x > 0$ then we always have that $x > \epsilon$. Let z be a binary variable and assume that $x \leq M$.

Logical Statements

Let x be a non-negative real variable, and assume that if $x > 0$ then we always have that $x > \epsilon$. Let z be a binary variable and assume that $x \leq M$.

- $x = 0$ if and only if z false.

Logical Statements

Let x be a non-negative real variable, and assume that if $x > 0$ then we always have that $x > \epsilon$. Let z be a binary variable and assume that $x \leq M$.

- $x = 0$ if and only if z false.
- **Solution:** The two next inequalities do the job.

$$\begin{aligned} x &\leq z \cdot M \quad (x > 0 \rightarrow z = 1) \\ z &\leq \frac{x}{\epsilon} \quad (x = 0 \rightarrow z = 0) \end{aligned}$$

Logical Statements

For the typical production problem, where z could represent the decision of activating a machine and $K > 0$ could be the cost of activating it, we only need to write:

$$\begin{aligned} \min \quad & (\dots) + K \cdot z \\ \text{s.t.} \quad & \text{production} \leq M \cdot z \\ & (\dots),\ z \text{ binary.} \end{aligned}$$

Because the part $z = 0 \rightarrow \textit{production} = 0$ is implied directly by the constraint, and $\textit{production} = 0 \rightarrow z = 0$ is obtained by the fact that we are minimizing.

Job Scheduling Problem

(Exercise 10.7 from Introduction to Linear Programming, Bertsimas & Tsitklis) We consider the production of a single product over T periods. If we decide to produce at period t, a setup cost of c_t is incurred. For $t = 1, \ldots, T$ let d_t be the demand for this product in period t, and let p_t, h_t be the unit production and storage cost resp. for period t.

1. Formulate a MILP in order to minimize the total cost of production, storage, and setup.

Job Scheduling Problem

Variables:

Job Scheduling Problem

Variables:

- z_t : binary variable that indicates if we produce in month t.

Job Scheduling Problem

Variables:

- z_t : binary variable that indicates if we produce in month t.
- x_t : production on month t.

Job Scheduling Problem

Variables:

- z_t : binary variable that indicates if we produce in month t.
- x_t : production on month t.
- I_t : inventory on month t (Also I_0 is included).

Job Scheduling Problem

Variables:

- z_t : binary variable that indicates if we produce in month t.
- x_t : production on month t.
- I_t : inventory on month t (Also I_0 is included).

Formulation:

Job Scheduling Problem

Variables:

- z_t : binary variable that indicates if we produce in month t.
- x_t : production on month t.
- I_t : inventory on month t (Also I_0 is included).

Formulation:

$$
\begin{aligned}
\min \quad & \sum_{t=1}^{T} (z_t c_t + x_t p_t + I_t h_t) \\
\text{s.t.} \quad & x_t \leq z_t \left(\sum_{i=1}^{T} d_i \right), \\
& I_t = I_{t-1} + x_t - d_t, \quad \forall t \in \{1, \ldots, T\} \\
& I_0 = 0,\ z_t \text{ binary}, \quad \forall t \in \{1, \ldots, T\} \\
& x_t \geq 0, \quad \forall t \in \{1, \ldots, T\} \\
& I_t \geq 0, \quad \forall t \in \{1, \ldots, T\}
\end{aligned}
$$

Job Scheduling Problem

2. Suppose we allow demand to be lost in every period except for period T, at a cost of b_t per unit lost of demand. Show how to modify the model to handle this option.

Job Scheduling Problem

2. Suppose we allow demand to be lost in every period except for period T, at a cost of b_t per unit lost of demand. Show how to modify the model to handle this option.
3. **Solution:** we need to add a new variable ℓ_t that is the demand lost in period t. We have to take into account the fact that it may be optimal to not satisfy demand in period t even if we could in order to use the saved storage for the next period.

Job Scheduling Problem

Model:

$$
\begin{aligned}
\min \quad & \sum_{t=1}^{T} (z_t c_t + x_t p_t + I_t h_t) + \sum_{t=1}^{T-1} \ell_t b_t \\
\text{s.t.} \quad & x_t \le z_t \left(\sum_{i=1}^{T} d_i \right), \\
& I_t = I_{t-1} + x_t - d_t + \ell_t, \quad \forall t \in \{1, \ldots, T-1\} \\
& I_T = I_{T-1} + x_T - d_T \\
& \ell_t \le d_t, \quad \forall t \in \{1, \ldots, T-1\} \\
& I_0 = 0,\ \ell_t \ge 0, \quad \forall t \in \{1, \ldots, T-1\} \\
& z_t \text{ binary}, \quad \forall t \in \{1, \ldots, T\} \\
& x_t \ge 0, \quad \forall t \in \{1, \ldots, T\},\ I_t \ge 0, \quad \forall t \in \{1, \ldots, T\}.
\end{aligned}
$$

Job Scheduling Problem

3. Suppose that production capacity can occur in at most five periods, but no two such periods can be consecutive. Show how to modify the model to handle this option.

Job Scheduling Problem

3. Suppose that production capacity can occur in at most five periods, but no two such periods can be consecutive. Show how to modify the model to handle this option.
4. **Solution:** We can just add the following constraints:
   - $\sum_{t=1}^{T} z_t \leq 5.$
   - $z_j + z_{j+1} \leq 1 \quad \forall j \in \{1, \ldots, T-1\}.$

Plan for a move: problem

(Exercise 10.5 from Introduction to Linear Programming, Bertsimas & Tsitklis.) Suppose you are planning to move your new house. You have n items of size a_j, $j = 1, \ldots, n$ that need to be moved. You have rented a truck that has size Q and you have bought m boxes. Box i has size b_i, $i = 1, \ldots, m$. **Formulate an integer programming problem in order to decide if the move is possible**.

Plan for a move: solution

This problem needs extra assumptions, i will assume the following two: First, Let's imagine that the truck deliverers are so good at Tetris that for any combination of boxes with total volume less than Q they are able load the truck. Second, we can put as many objects in a box as long we do not surpass its volume (but of course let's imagine we can not divide the objects).

Plan for a move: solution

Variables:

- z_i : Binary variable that represents if box $i \in \{1, \ldots, m\}$ is taken in the truck or not.

Plan for a move: solution

Variables:

- z_i : Binary variable that represents if box $i \in \{1, \ldots, m\}$ is taken in the truck or not.
- x_j^i : Binary variable that represents if item $j \in \{1, \ldots, n\}$ is stored in box i or not.

Plan for a move: solution

Formulation:

$$\min \quad \sum_{i=1}^{m} z_i \text{ (Could have been anything)}$$

$$\text{s.t.} \quad \sum_{i=1}^{m} x_j^i = 1, \quad \forall j \in \{1, \dots, n\} \quad \text{(Also} \geq \text{works)}$$

$$\sum_{j=1}^{n} a_j x_j^i \leq z_i b_i, \quad \forall i \in \{1, \dots, m\} \quad \text{(Box Capacity)}$$

$$\sum_{j=1}^{n} z_i b_i \leq Q, \quad \forall i \in \{1, \dots, m\} \quad \text{(Truck Capacity)}$$

$$z_j \text{ binary}, \quad \forall i \in \{1, \dots, m\}$$

$$x_j^i \text{ binary}, \quad \forall i \in \{1, \dots, n\},\ j \in \{1, \dots, m\}$$

AMPL

- A restriction phrase for a parameter declaration may be the word *integer* or *binary* or a comparison operator followed by an arithmetic expression.

AMPL

- A restriction phrase for a parameter declaration may be the word *integer* or *binary* or a comparison operator followed by an arithmetic expression.
- While *integer* restricts a parameter to integral (whole-number) values, *binary* restricts it to zero or one.

AMPL

- A restriction phrase for a parameter declaration may be the word *integer* or *binary* or a comparison operator followed by an arithmetic expression.
- While *integer* restricts a parameter to integral (whole-number) values, *binary* restricts it to zero or one.
- **Example:**

$$param \quad promote\{1..T, 1..T\}\ binary;$$

AMPL

- A restriction phrase for a parameter declaration may be the word *integer* or *binary* or a comparison operator followed by an arithmetic expression.
- While *integer* restricts a parameter to integral (whole-number) values, *binary* restricts it to zero or one.
- **Example:**

$$param \quad promote\{1..T, 1..T\} \ binary;$$

- **Example:**

$$param \quad promote\{1..T, 1..T\} \ integer;$$

Thank you for your attention !