MODELING WITH BINARY VARIABLES

Class 3 – September 30, 2024

Context

- You have several projects available
- You choose **which projects to fund**
- For each project, we have a binary variable indicating if it's chosen
- A=1 **if and only if** project A is funded

If you fund **A**, you should also fund **E**

- What are the feasible values for A, E?
  - Recall that A, E are **binary**
  - We want: *if A=1, must have E=1*

ALL OPTIONS:

A	**E**	
0	0	✓
0	1	✓
~~1~~	~~0~~	
1	1	✓

- How about: **A** $\leq$ **E**
  - If A=1, the only option is E=1
  - If A=0, can set any value for E

- **Remember!** "If you fund **A**, then you should fund **B**": **A** $\leq$ **B**

- **Q:** "If you do **not** fund **A**, then you should fund **B**"
  - Add a constraint: $1 - A \leq B$
    - "Not selecting A" is same as $1 - A = 1$, so this is just like **Q5 !**

Logical Implications with Binary Variables

- **Q**. If you fund project A, then you should fund projects E **and** H.
  - Same as: “If you fund A, then fund E” and “If you fund A, then fund H”
  - A <= E, A <= H
  - Also possible to do this with **one** constraint: A <= (E+H)/2

  Q. Why not A <= E+H?

- **Q**. If you fund anything from **A/B/C**, then also fund **H**.
  - Same as: “If you fund A, then fund H” and “If you fund B, then fund H”, …
  - A <= H, B <= H, C <= H
  - Also possible to do this with **one** constraint: (A+B+C)/3 <= H

  Q. Why not A + B + C <= H?

Modeling Complex Logical Constraints

- Suppose we would like to model a constraint of the form:

$$Y = 1 \textit{ if and only if } a_1 X_1 + \dots + a_n X_n + b \geq 0$$

  - Y is a binary decision variable
  - $X_1, \dots, X_n$ are continuous or discrete decision variables
  - $a_1, \dots, a_n, b$ are parameters/data

- This has two implications

(1): *If* $Y = 1$ *then* $a_1 X_1 + \dots + a_n X_n + b \geq 0$

(2): *If* $Y = 0$ *then* $a_1 X_1 + \dots + a_n X_n + b < 0$ ← can't have a strict inequality

(2'): *If* $Y = 0$ *then* $a_1 X_1 + \dots + a_n X_n + b + \varepsilon \leq 0$ ← instead, do this!

'ε' is a **parameter** with a small, positive value (e.g., 0.00000001)

if X_1 are discrete, you can typically set a precise value here

- The two implications can be implemented with the constraints:

(1): $a_1 X_1 + \dots + a_n X_n + b \geq m \cdot (1 - Y)$

'**m**' is a parameter = smallest value $a_1 X_1 + \dots + a_n X_n + b$ can take (with any X)

(2'): $a_1 X_1 + \dots + a_n X_n + b + \varepsilon \leq (M + \varepsilon) Y$

'**M**' is a parameter = largest value $a_1 X_1 + \dots + a_n X_n + b$ can take (with any X)

"Cheat-Sheet"

(1): *If* $Y = 1$ *then* $a_1 X_1 + \dots + a_n X_n + b \geq 0$

(1): $a_1 X_1 + \dots + a_n X_n + b \geq m \cdot (1 - Y)$

'**m**' is a parameter = smallest value $a_1 X_1 + \dots + a_n X_n + b$ can take (with any X)

(2'): *If* $Y = 0$ *then* $a_1 X_1 + \dots + a_n X_n + b + \varepsilon \leq 0$

(2'): $a_1 X_1 + \dots + a_n X_n + b + \varepsilon \leq (M + \varepsilon)\, Y$

'**M**' is a parameter = largest value $a_1 X_1 + \dots + a_n X_n + b$ can take (with any X)

EXAMPLE. Y=1 *if and only if* A+B+C ≥ 2.

First direction. *If* Y=1 *then* A+B+C ≥ 2.

- **"aX"** is A+B+C. **"b"** is -2. **"m"** is -2.
  (m is smallest value of A+B+C-2. Because A,B,C all take 0/1 values, smallest value is achieved when they are 0.)
- Add the constraint: A+B+C **–** 2 ≥ (-2) * (1–Y) which is equivalent to A+B+C ≥ 2*Y

"Cheat-Sheet"

(1): *If* $Y = 1$ *then* $a_1 X_1 + \dots + a_n X_n + b \geq 0$

(1): $a_1 X_1 + \dots + a_n X_n + b \geq m \cdot (1 - Y)$

'**m**' is a parameter = smallest value $a_1 X_1 + \dots + a_n X_n + b$ can take (with any X)

(2'): *If* $Y = 0$ *then* $a_1 X_1 + \dots + a_n X_n + b + \varepsilon \leq 0$

(2'): $a_1 X_1 + \dots + a_n X_n + b + \varepsilon \leq (M + \varepsilon)\, Y$

'**M**' is a parameter = largest value $a_1 X_1 + \dots + a_n X_n + b$ can take (with any X)

EXAMPLE. Y=1 *if and only if* A+B+C ≥ 2.

Second direction. *If* Y=0 *then* A+B+C ≤ 1.

- **"aX"** is A+B+C. **"b"** is -1. "**ε**" is 0. **"M"** is 2.
  (M is largest value of A+B+C-1. Because A,B,C all take 0/1 values, largest value is achieved when they are 1.)
- Add the constraint: A+B+C **–** 1 ≤ 2*Y which is equivalent to A+B+C ≤ 2*Y + 1

"Cheat-Sheet"

X and **Y** are decisions; **a**, **b** are parameters/data; **a X** denotes any linear expression in **X**

1. (X,Y bin) ***"If X = 1 then Y = 1"*** → add constraint: $X \leq Y$
2. (X,Y bin) ***"If X = 1 then Y = 1, and vice-versa"*** → add constraint: $X = Y$
3. (Y bin) ***"If Y = 1 then a X + b ≥ 0"*** → add constraint: $a X + b \geq m \cdot (1-Y)$
   - 'm' is the *smallest* value *a X + b* can take
4. (Y bin) ***"If Y = 1 then a X ≥ b"*** → add constraint: $a X - b \geq m \cdot (1-Y)$
   - 'm' is the *smallest* value *(a X – b)* can take
5. (Y bin) ***"If Y = 1 then a X ≤ b "*** → add constraint: $a X - b \leq M \cdot (1-Y)$
   - 'M' is the *largest* value *(a X - b)* can take
6. (Y bin) ***"If Y = 1 then a X + b ≤ 0 "*** → add constraint: $a X + b \leq M \cdot (1-Y)$
   - 'M' is *largest* value (*a X + b*) can take

3-6 are all "the same"! Use whichever you like!

7. (Y bin) ***"If Y = 1 then a X + b > 0 "*** → **CAN'T DO > 0.**
   - Instead, do ***"If Y = 1 then a X + b ≥ ε" for a very small number ε > 0***
   - To implement, add the constraint: $a X + b - \varepsilon \geq (m - \varepsilon)(1-Y)$, where 'm' is the smallest value (*a X + b*) can take
   - **If (*a X + b*) takes integer values, this is the same as "a X + b ≥ 1" and can be expressed exactly (no need for ε)**

8. If you need ***"If Y = 0 then …"***, replace *Y* in the constraint with *1-Y*
9. If you need ***"If a X + b ≤ 0 then Y = 1"***, replace this with ***"If Y=0, then a X + b > 0"***

10. (Y bin) Need **"X * Y"** → add new variable **Z** ("= X * Y") and constraints:

$$Z \leq M \cdot Y \qquad Z \geq m \cdot Y \qquad Z \leq X - m \cdot (1 - Y) \qquad Z \geq X - M \cdot (1 - Y)$$

   - m/M are smallest/largest value that X can take