15.6 Basic Modeling Tricks - Using Binary Variables

In this section, we describe ways to model a variety of constraints that commonly appear in practice. The goal is changing constraints described in words to constraints defined by math.

Binary variables can allow you to model many types of constraints. We discuss here varios logical constraints where we assume that $x_i \in \{0,1\}$ for $i = 1,\dots,n$. We will take the meaning of the variable to be selecting an item.

1. If item i is selected, then item j is also selected.

$$x_i \leq x_j \tag{15.1}$$

   (a) If any of items $1,\dots,5$ are selected, then item 6 is selected.

$$x_1 + x_2 + \cdots + x_5 \leq 5 \cdot x_6 \tag{15.2}$$

   Alternatively!

$$x_i \leq x_6 \quad \text{for all } i = 1,\dots,5 \tag{15.3}$$

2. If item j is not selected, then item i is not selected.

$$x_i \leq x_j \tag{15.4}$$

   (a) If item j is not selected, then all items $1,\dots,i$ are not selected.

$$x_1 + x_2 + \cdots + x_i \leq i \cdot x_j \tag{15.5}$$

3. If item j is not selected, then item i is not selected.

$$x_i \leq x_j \tag{15.6}$$

4. Either item i is selected or item j is selected, but not both.

$$x_i + x_j = 1 \tag{15.7}$$

5. Item i is selected or item j is selected or both.

$$x_i + x_j \geq 1 \tag{15.8}$$

6. If item i is selected, then item j is not selected.

$$x_j \leq (1 - x_i) \tag{15.9}$$

7. At most one of items i, j, and k are selected.

$$x_i + x_j + x_k \leq 1 \tag{15.10}$$

8. At most two of items i, j, and k are selected.

$$x_i + x_j + x_k \leq 2 \tag{15.11}$$

9. Exactly one of items i, j, and k are selected.

$$x_i + x_j + x_k = 1 \tag{15.12}$$

These tricks can be connected to create different function values.

Example 15.2: Variable takes one of three values

Suppose that the variable x should take one of the three values $\{4, 8, 13\}$. This can be modeled using three binary variables as

$$\begin{aligned} x &= 4z_1 + 8z_2 + 13z_3 \\ z_1 + z_2 + z_3 &= 1 \\ z_i \in \{0,1\} \text{ for } i &= 1,2,3. \end{aligned}$$

As a convenient addition, if we want to add the possibility that it takes the value 0, then we can model this as

$$\begin{aligned} x &= 4z_1 + 8z_2 + 13z_3 \\ z_1 + z_2 + z_3 &\leq 1 \\ z_i \in \{0,1\} \text{ for } i &= 1,2,3. \end{aligned}$$

We can also model variable increases at different amounts.

Example 15.3: Discount for buying more

Suppose you can choose to buy 1, 2, or 3 units of a product, each with a decreasing cost. The first unit is \$10, the second is \$5, and the third unit is \$3.

$$\begin{aligned} x &= 10z_1 + 5z_2 + 3z_3 \\ z_1 \geq z_2 &\geq z_3 \\ z_i \in \{0,1\} \text{ for } i &= 1,2,3. \end{aligned}$$

Here, z_i represents if we buy the ith unit. The inequality constraints impose that if we buy unit j, then we must by all units i with $i < j$.

In this section, we describe ways to model a variety of constraints that commonly appear in practice. The goal is changing constraints described in words to constraints defined by math.

15.6.1. Big M constraints - Activating/Deactivating Inequalities

Big M comes again! It's extremely useful when trying to activate constrants based on a binary variable.

For intance, if we don't rent a bus, then we can have at most 3 passengers join us on our trip. Consider passengers A, B, C, D, E and let $x_i \in \{0,1\}$ be 1 if we take passenger i and 0 otherwise. We can model the constraint that we can have at most 5 passengers as

$$x_A + x_B + x_C + x_D + x_E \leq 3.$$

We want to be able to activate this constraint in the event that we don't rent a bus.

Let $\delta \in \{0,1\}$ be 1 if rent a bus, and 0 otherwise.

Then we want to say

If $\delta = 0$, then
$$x_A + x_B + x_C + x_D + x_E \leq 3.$$

We can formulate this using a big-M constraint as

$$x_A + x_B + x_C + x_D + x_E \leq 3 + M\delta. \tag{15.13}$$

Notice the two case
$$\begin{cases} x_A + x_B + x_C + x_D + x_E \leq 3 & \text{if } \delta = 0 \\ x_A + x_B + x_C + x_D + x_E \leq 3 + M & \text{if } \delta = 1 \end{cases}$$

In the second case, we choose M to be so large, that the second case inequality is vacuous. That said, choosing smaller M values (that are still valid) will help the computer program solve the problem faster. In this case, it suffices to let $M = 2$.

We can speak about this technique more generally as

Big-M: If then:

We aim to model the relationship

$$\text{If} \quad \delta = 0, \quad \text{then} \quad a^\top x \leq b. \tag{15.14}$$

By letting M be an upper bound on the quantity $a^\top x - b$, we can model this condition as

$$\begin{aligned} a^\top x - b &\leq M\delta \\ \delta &\in \{0,1\} \end{aligned} \tag{15.15}$$

15.6.2. Either Or Constraints

"At least one of these constraints holds" is what we would like to model. Equivalently, we can phrase this as an *inclusive or* constraint. This can be molded with a pair of Big-M constraints.

Either Or:

$$\text{Either} \quad a^\top x \leq b \quad \text{or} \quad c^\top x \leq d \quad \text{holds} \tag{15.16}$$

can be modeled as

$$\begin{aligned} a^\top x - b &\leq M_1 \delta \\ c^\top x - d &\leq M_2(1-\delta) \\ \delta &\in \{0,1\}, \end{aligned} \tag{15.17}$$

where M_1 is an upper bound on $a^\top x - b$ and M_2 is an upper bound on $c^\top x - d$.

Example 15.4

Either 2 buses or 10 cars are needed shuttle students to the football game.

- *Let x be the number of buses we have and*
- *let y be the number of cars that we have.*

Suppose that there are at most $M_1 = 5$ buses that could be rented an at most $M_2 = 20$ cars that could be available.
This constraint can be modeled as

$$\begin{aligned} x - 2 &\leq 5\delta \\ y - 10 &\leq 20(1-\delta) \\ \delta &\in \{0,1\}, \end{aligned} \tag{15.18}$$

15.6.3. If then implications - opposite direction

Suppose that we want to model the fact that if we have at most 10 students attending this course, then we must switch to a smaller classroom.

Let $x_i \in \{0,1\}$ be 1 if student i is in the course or not. Let $\delta \in \{0,1\}$ be 1 if we need to switch to a smaller classroom.

Thus, we want to model

If

$$\sum_{i \in I} x_i \leq 10$$

then

$$\delta = 1.$$

We can model this as

$$\sum_{i \in I} x_i \geq 10 + 1 + M\delta. \tag{15.19}$$

If inequality, then indicator:

W

e let m be a lower bound on the quantity $a^\top x - b$ and we let ε be a tiny number that is an error bound in verifying if an inequality is violated. **If the data a, b are integer and x is an integer, then we can take $\varepsilon = 1$.**

Now

$$\text{If} \quad a^\top x \leq b \quad \text{then} \quad \delta = 1 \tag{15.20}$$

can be modeled as

$$a^\top x - b \geq \varepsilon(1 - \delta) + m\delta. \tag{15.21}$$

Proof. We not justify the statement above.

A simple way to understand this constraint is to consider the *contrapositive* of the if then statement that we want to model. The contrapositive says that

$$\text{If } \delta = 0, \text{ then } a^\top x - b > 0. \tag{15.22}$$

To show the contrapositive, we set $\delta = 0$. Then the inequality becomes

$$a^\top x - b \geq \varepsilon(1 - 0) + m0 = \varepsilon > 0.$$

Thus, the contrapositive holds.

If instead we wanted a direct proof:

Case 1: Suppose $a^\top x \leq b$. Then $0 \geq a^\top x - b$, which implies that

$$\delta(a^\top x - b) \geq a^\top x - b$$

Therefore

$$\delta(a^\top x - b) \geq \varepsilon(1 - \delta) + m\delta$$

After rearranging

$$\delta(a^\top x - b - m) \geq \varepsilon(1 - \delta)$$

Implication	Constraint
If $\delta = 0$, then $a^\top x \leq b$	$a^\top x \leq b + M\delta$
If $a^\top x \leq b$, then $\delta = 1$	$a^\top x \geq m\delta + \varepsilon(1-\delta)$

Table 15.1: Short list: If/then models with a constraint and a binary variable. Here M and m are upper and lower bounds on $a^\top x - b$ and ε is a small number such that if $a^\top x > b$, then $a^\top x \geq b + \varepsilon$.

Implication	Constraint
If $\delta = 0$, then $a^\top x \leq b$	$a^\top x \leq b + M\delta$
If $\delta = 0$, then $a^\top x \geq b$	$a^\top x \geq b + m\delta$
If $\delta = 1$, then $a^\top x \leq b$	$a^\top x \leq b + M(1-\delta)$
If $\delta = 1$, then $a^\top x \geq b$	$\mathrm{a}^\top x \geq b + m(1-\delta)$
If $a^\top x \leq b$, then $\delta = 1$	$a^\top x \geq b + m\delta + \varepsilon(1-\delta)$
If $a^\top x \geq b$, then $\delta = 1$	$a^\top x \leq b + M\delta - \varepsilon(1-\delta)$
If $a^\top x \leq b$, then $\delta = 0$	$a^\top x \geq b + m(1-\delta) + \varepsilon\delta$
If $a^\top x \geq b$, then $\delta = 0$	$a^\top x \geq b + m(1-\delta) - \varepsilon\delta$

Table 15.2: Long list: If/then models with a constraint and a binary variable. Here M and m are upper and lower bounds on $a^\top x - b$ and ε is a small number such that if $a^\top x > b$, then $a^\top x \geq b + \varepsilon$.

Since $a^\top x - b - m \geq 0$ and $\varepsilon > 0$, the only feasible choice is $\delta = 1$.

Case 2: Suppose $a^\top x > b$. Then $a^\top x - b \geq \varepsilon$. Since $a^\top x - b \geq m$, both choices $\delta = 0$ and $\delta = 1$ are feasible.

By the choice of ε, we know that $a^\top x - b > 0$ implies that $a^\top x - b \geq \varepsilon$.

Since we don't like strict inequalities, we write the strict inequality as $a^\top x - b \geq \varepsilon$ where ε is a small positive number that is a smallest difference between $a^\top x - b$ and 0 that we would typically observe. As mentioned above, if a, b, x are all integer, then we can use $\varepsilon = 1$.

Now we want an inequality with left hand side $a^\top x - b \geq$ and right hand side to take the value

- ε if $\delta = 0$,
- m if $\delta = 1$.

This is accomplished with right hand side $\varepsilon(1-\delta) + m\delta$. ♠

Many other combinations of if then statements are summarized in the following table: These two implications can be used to derive the following longer list of implications.

Lastly, if you insist on having exact correspondance, that is, "$\delta = 0$ if and only if $a^\top x \leq b$" you can simply include both constraints for "if $\delta = 0$, then $a^\top x \leq b$" and if "$a^\top x \leq b$, then $\delta = 0$". Although many problems may be phrased in a way that suggests you need "if and only if", it is often not necessary to use both constraints due to the objectives in the problem that naturally prevent one of these from happening.

For example, if we want to add a binary variable δ that means

$$\begin{cases} \delta = 0 \text{ implies } a^\top x \le b \\ \delta = 1 \text{ Otherwise} \end{cases}$$

If $\delta = 1$ does not effect the rest of the optimization problem, then adding the constraint regarding $\delta = 1$ is not necessary. Hence, typically, in this scenario, we only need to add the constraint $a^\top x \le b + M\delta$.

15.6.4. Binary reformulation of integer variables

If an integer variable has small upper and lower bounds, it can sometimes be advantageous to recast it as a sequence of binary variables - for either modeling, the solver, or both. Although there are technically many ways to do this, here are the two most common ways.

Full reformulation:

u many binary variables

For a non-negative integer variable x with upper bound u, modeled as

$$0 \le x \le u, \quad x \in \mathbb{Z}, \tag{15.23}$$

this can be reformulated with u binary variables $z_1, \dots, z_u$ as

$$\begin{aligned} x &= \sum_{i=1}^{u} i z_i = z_1 + 2z_2 + \cdots + u z_u \\ 1 &\ge \sum_{i=1}^{u} z_i = z_1 + z_2 + \cdots + z_u \\ z_i &\in \{0,1\} \quad \text{for } i = 1, \dots, u \end{aligned} \tag{15.24}$$

We call this the *full reformulation* because there is a binary variable z_i associated with every value i that x could take. That is, if $z_3 = 1$, then the second constraint forces $z_i = 0$ for all $i \neq 3$ (that is, z_3 is the only non-zero binary variable), and hence by the first constraint, $x = 3$.

Binary reformulation:

$O(\log u)$ many binary variables

For a non-negative integer variable x with upper bound u, modeled as

$$0 \le x \le u, \quad x \in \mathbb{Z}, \tag{15.25}$$

this can be reformulated with u binary variables $z_1, \ldots, z_{\log(\lfloor u \rfloor)+1}$ as

$$x = \sum_{i=0}^{\log(\lfloor u \rfloor)+1} 2^i z_i = z_0 + 2z_1 + 4z_2 + 8z_3 + \cdots + 2^{\log(\lfloor u \rfloor)+1} z_{\log(\lfloor u \rfloor)+1} \tag{15.26}$$
$$z_i \in \{0,1\} \quad \text{for } i = 1, \ldots, \log(\lfloor u \rfloor)+1$$

We call this the *log reformulation* because this requires only logarithmically many binary variables in terms of the upper bound u. This reformulation is particularly better than the full reformulation when the upper bound u is a “larger" number, although we will leave it ambiguous as to how larger a number need to be in order to be described as a “larger" number.

15.6.5. SOS1 Constraints

Definition 15.5: Special Ordered Sets of Type 1 (SOS1)

Special Ordered Sets of type 1 (SOS1) constraint on a vector indicates that at most one element of the vector can non-zero.

We next give an example of how to use binary variables to model this and then show how much simpler it can be coded using the SOS1 constraint.

Example: SOS1 Constraints Gurobipy

Solve the following optimization problem:

$$\begin{aligned} \text{maximize} \quad & 3x_1 + 4x_2 + x_3 + 5x_4 \\ \text{subject to} \quad & 0 \leq x_i \leq 5 \\ & \text{at most one of the } x_i \text{ can be nonzero} \end{aligned}$$

15.6.6. SOS2 Constraints

Definition 15.6: Special Ordered Sets of Type 2 (SOS2)

A Special Ordered Set of Type 2 (SOS2) *constraint on a vector indicates that* at most two elements of the vector can non-zero AND the non-zero elements must appear consecutively.

We next give an example of how to use binary variables to model this and then show how much simpler it can be coded using the SOS2 constraint.

Example: SOS2 Gurobipy

Solve the following optimization problem:

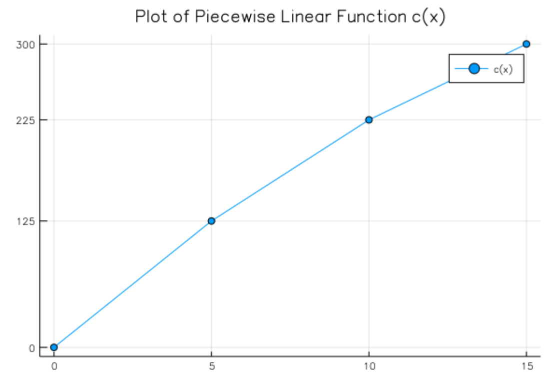
$$\begin{array}{rl} \text{maximize} & 3x_1 + 4x_2 + x_3 + 5x_4 \\ \text{subject to} & 0 \leq x_i \leq 5 \\ & \text{at most two of the } x_i \text{ can be nonzero} \\ & \text{and the nonzero } x_i \text{ must be consecutive} \end{array}$$

15.6.7. Piecewise linear functions with SOS2 constraint

Example: Piecewise Linear Function Gurobipy

Consider the piecewise linear function $c(x)$ given by

$$c(x) = \begin{cases} 25x & \text{if } 0 \leq x \leq 5 \\ 20x + 25 & \text{if } 5 \leq x \leq 10 \\ 15x + 75 & \text{if } 10 \leq x \leq 15 \end{cases}$$



© **pwl-plot.png**[5]

We will use integer programming to describe this function. We will fix $x = a$ and then the integer program will set the value y to $c(a)$.

$$
\begin{aligned}
\min \quad & 0 \\
\text{Subject to} \quad & x - 5z_2 - 10z_3 - 15z_4 = 0 \\
& y - 125z_2 - 225z_3 - 300z_4 = 0 \\
& z_1 + z_2 + z_3 + z_4 = 1 \\
& SOS2: \{z_1, z_2, z_3, z_4\} \\
& 0 \leq z_i \leq 1 \quad \forall i \in \{1,2,3,4\} \\
& x = a
\end{aligned}
$$

Example: Piecewise Linear Function Application Gurobipy

Consider the following optimization problem where the objective function includes the term $c(x)$, where $c(x)$ is the piecewise linear function described in Example 23:

$$\max z = 12x_{11} + 12x_{21} + 14x_{12} + 14x_{22} - c(x) \tag{15.27}$$
$$\text{s.t.} x_{11} + x_{12} \leq x + 5 \tag{15.28}$$
$$x_{21} + x_{22} \leq 10 \tag{15.29}$$
$$0.5x_{11} - 0.5x_{21} \geq 0 \tag{15.30}$$
$$0.4x_{12} - 0.6x_{22} \geq 0 \tag{15.31}$$
$$x_{ij} \geq 0 \tag{15.32}$$
$$0 \leq x \leq 15 \tag{15.33}$$

Given the piecewise linear, we can model the whole problem explicitly as a mixed-integer linear program.

[5]**pwl-plot.png**, from **pwl-plot.png**. **pwl-plot.png**, **pwl-plot.png**.

$$
\begin{aligned}
\max \quad & 12X_{1,1} + 12X_{2,1} + 14X_{1,2} + 14X_{2,2} - y \\
\text{Subject to} \quad & x - 5z_2 - 10z_3 - 15z_4 &= 0 \\
& y - 125z_2 - 225z_3 - 300z_4 &= 0 \\
& z_1 + z_2 + z_3 + z_4 &= 1 \\
& X_{1,1} + X_{1,2} - x &\leq 5 \\
& X_{2,1} + X_{2,2} \leq 10 \\
& 0.5X_{1,1} - 0.5X_{2,1} &\geq 0 \\
& 0.4X_{1,2} - 0.6X_{2,2} &\geq 0 \\
& SOS2 : \{z_1, z_2, z_3, z_4\} \\
& X_{i,j} &\geq 0 \quad \forall i \in \{1,2\}, j \in \{1,2\} \\
& 0 \leq z_i &\leq 1 \quad \forall i \in \{1,2,3,4\} \\
& 0 \leq x &\leq 15 \\
& y & \text{free}
\end{aligned}
\tag{15.34}
$$

15.6.7.1. SOS2 with binary variables

Modeling Piecewise linear function

- Write down pairs of breakpoints and functions values $(a_i, f(a_i))$.
- Define a binary variable z_i indicating if x is in the interval $[a_i, a_{i+1}]$.
- Define multipliers λ_i such that x is a combination of the a_i's and therefore the output $y = f(x)$ is a combination of the $f(a_i)$'s.
- Restrict that at most 2 $\lambda_i's$ are non-zero and that those 2 are consecutive.

$$
\begin{aligned}
\min \quad & \sum_{i=1}^{k} \lambda_i f(a_i) \\
\text{s.t.} \quad & \sum_{i=1}^{k} \lambda_i = 1 \\
& x = \sum_{i=1}^{k} \lambda_i a_i \\
& \lambda_1 \leq z_1 \\
& \lambda_i \leq z_{i-1} + z_i && \text{for } i = 2, \ldots, k-1, \\
& \lambda_k \leq z_{k-1} \\
& \lambda_i \geq 0, y_i \in \{0,1\}.
\end{aligned}
$$

15.6.8. Maximizing a minimum

When the constraints could be general, we will write $x \in X$ to define general constraints. For instance, we could have $X = \{x \in \mathbb{R}^n : Ax \leq b\}$ of $X = \{x \in \mathbb{R}^n : Ax \leq b, x \in \mathbb{Z}^n\}$ or many other possibilities.

Consider the problem

$$\begin{array}{rl} \max & \min\{x_1, \dots, x_n\} \\ \text{such that} & x \in X \end{array}$$

Having the minimum on the inside is inconvenient. To remove this, we just define a new variable y and enforce that $y \leq x_i$ and then we maximize y. Since we are maximizing y, it will take the value of the smallest x_i. Thus, we can recast the problem as

$$\begin{array}{rl} \max & y \\ \text{such that} & y \leq x_i \ \text{ for } \ i = 1, \dots, n \\ & x \in X \end{array}$$

15.6.9. Relaxing (nonlinear) equality constraints

There are a number of scenarios where the constraints can be relaxed without sacrificing optimal solutions to your problem. In a similar vein of the maximizing a minimum, if because of the objective we know that certain constraints will be tight at optimal solutions, we can relax the equality to an inequality. For example,

$$\begin{array}{rl} \max & x_1 + x_2 + \cdots + x_n \\ \text{such that} & x_i = y_i^2 + z_i^2 \text{ for } i = 1, \dots, n \end{array}$$

15.6.10. Connecting to continuous variables

Let $x_i \geq 0$ and $y_i \in \{0, 1\}$ for all $i = 1, \dots, n$.

1. If $x_i > 0$, then $y_i = 1$.
$$x_i \leq M y_i \tag{15.35}$$
where M is a suffiently large upper bound on the variable x_i.

2. If $x_i = 0$, then $y_i = 0$.
This is harder to model! Alternatively, we try modeling "if x_i is sufficiently small, then $y_i = 0$. For instance, if $x_i \leq 0.0000001$, then $y_i = 0$. This can be modeled as
$$x_i - 0.0000001 \geq y_i - 1. \tag{15.36}$$

3. If $y_i = 1$, then $x_i \geq 5$

$$5y_i \leq x_i. \tag{15.37}$$

15.6.11. Exact absolute value

Suppose we need to model an exact equality

$$|x| = t$$

It defines a non-convex set, hence it is not conic representable. If we split x into positive and negative part $x = x^+ - x^-$, where $x^+, x^- \geq 0$, then $|x| = x^+ + x^-$ as long as either $x^+ = 0$ or $x^- = 0$. That last alternative can be modeled with a binary variable, and we get a model of :

$$\begin{aligned} x &= x^+ - x^- \\ t &= x^+ + x^- \\ 0 &\leq x^+, x^- \\ x^+ &\leq Mz \\ x^- &\leq M(1-z) \\ z &\in \{0,1\} \end{aligned}$$

where the constant M is an a priori known upper bound on $|x|$ in the problem.

15.6.11.1. Exact 1 -norm

We can use the technique above to model the exact ℓ_1-norm equality constraint

$$\sum_{i=1}^{n} |x_i| = c$$

where $x \in \mathbb{R}^n$ is a decision variable and c is a constant. Such constraints arise for instance in fully invested portfolio optimizations scenarios (with short-selling). As before, we split x into a positive and negative part, using a sequence of binary variables to guarantee that at most one of them is nonzero:

$$\begin{aligned} x &= x^+ - x^- \\ 0 &\leq x^+, x^-, \\ x^+ &\leq cz \\ x^- &\leq c(e-z), \\ \sum_i x_i^+ + \sum_i x_i^- &= c, \\ z &\in \{0,1\}^n, x^+, x^- \in \mathbb{R}^n \end{aligned}$$

15.6.11.2. Maximum

The exact equality $t = \max\{x_1, \ldots, x_n\}$ can be expressed by introducing a sequence of mutually exclusive indicator variables $z_1, \ldots, z_n$, with the intention that $z_i = 1$ picks the variable x_i which actually achieves maximum. Choosing a safe bound M we get a model:

$$\begin{aligned} x_i \leq t \leq x_i + M(1 - z_i), i = 1, \ldots, n \\ z_1 + \cdots + z_n = 1, \\ z \in \{0,1\}^n \end{aligned}$$