第3章　整数规划建模技巧

整数规划的建模是非常有技巧性的。基础的关系，如和、差等，比较容易建模。但是一些复杂的关系，如互斥关系、并列关系等，就需要借助一些建模技巧才能完成。除此之外，还有一些非线性模型，也可以通过线性化的手段将其等价转化为线性模型，从而更高效地求解。建立正确、紧凑的模型是解决运筹优化问题非常重要和基础的部分，也是该领域从业者必须具备的能力。本章主要介绍逻辑约束和线性化两大部分。其中逻辑约束部分可以帮助读者准确地用数学语言刻画出实际问题，线性化部分可以用于模型重构和转化。

3.1　逻辑约束

在使用运筹学建模过程中，我们经常会遇到逻辑约束问题。那到底什么是逻辑约束呢？我们以投资组合为例进行说明。假如某商人考虑从若干种投资组合中选择投资，他目前对于每一项的投资的净收益和购买价格是已知的。在进行投资决策时，除了要满足总购买价格小于可用于投资的总资产外，该商人还要考虑一些约束，如：最多可以投资两项；如果选择投资 A，那么就必须选择投资 B；如果选择投资 C，那么就无法再选择投资 D；等等。这些约束就属于逻辑约束。对逻辑约束进行数学建模，通常需要灵活使用如下两个命题和两个条件。

3.1.1　两个命题

在对逻辑约束条件建模时，通常需要使用如下两个命题。

（1）如果 0-1 型变量 $y=0$, 那么变量 $x \leqslant 0$（x 不一定是 0-1 型变量，还可以是一个线性表达式）。

我们通常把该关系约束表示成：$x - My \leqslant 0$，其中 M 是变量 x 的一个上界。

（2）逆否命题。

如果满足约束 A，那么就需要满足约束 B。 $\Longleftrightarrow$ 如果不满足约束 B，那么就不满足约束 A。

3.1.2　二选一约束条件

在数学规划问题中，经常会出现如下情况：

$$f(x_1, x_2, \cdots, x_n) \leqslant 0 \tag{3.1}$$

$$g(x_1, x_2, \cdots, x_n) \leqslant 0 \tag{3.2}$$

如果我们希望保证至少满足 (3.1) 和 (3.2) 中的一个约束条件，则它被称作二选一约束条件（Either-Or）。

对于这种逻辑约束条件，通常引入 0-1 型变量。此逻辑约束条件可表示如下：

$$f(x_1, x_2, \cdots, x_n) \leqslant My \tag{3.3}$$

$$g(x_1, x_2, \cdots, x_n) \leqslant M(1-y) \tag{3.4}$$

在 (3.3) 和 (3.4) 中，y 是 0-1 型变量，M 是一个足够大的数。这两条约束可以至少满足 (3.1) 和 (3.2) 中的一个约束条件。

我们可以这样理解，如果 $y=0$, 那么 (3.3) 和 (3.4) 将变成 $f \leqslant 0$ 和 $g \leqslant M$，必定满足 (3.1)，也许还满足 (3.2)。类似地，如果 $y=1$，那么 (3.3) 和 (3.4) 将变成 $f \leqslant M$ 和 $g \leqslant 0$，必定满足 (3.2)，也许还满足 (3.1)。因此，无论 $y=0$ 还是 $y=1$, (3.3) 和 (3.4) 都将保证至少满足 (3.1) 和 (3.2) 中的一个约束条件。

在 Gurobi 中，这类约束可以通过构建 `And` 和 `Or` 约束实现。相应的函数为 `addGenConstrAnd (resvar, vars, name="")` 和 `addGenConstrOr(resvar, vars, name="")`。

首先来看 `And` 约束。假设有一系列 0-1 变量 $y, x_1, x_2, \cdots, x_n$，考虑下面的关系。

$$y = \text{and}\{x_1, x_2, \cdots, x_n\}$$

式中，当且仅当所有 $x_1, x_2, \cdots, x_n$ 的取值均为 1 时，$y=1$，否则 $y=0$。如果 $x_1, x_2, \cdots, x_n$ 为一般变量（即连续型变量），上面的约束依然可以运行成功，但是 Gurobi 会在内部自动将参与 `And` 约束的变量强制转换成 0-1 变量。下面为代码示例。

And constraints

```python
# define variables
x_1 = model.addVar(vtype = GRB.BINARY)
x_2 = model.addVar(vtype = GRB.BINARY)
x_3 = model.addVar(vtype = GRB.BINARY)
y = model.addVar(vtype = GRB.BINARY)
# 也可以将其定义为连续变量
# x_1 = model.addVar(lb = -GRB.INFINITY, ub = GRB.INFINITY, vtype = GRB.
    # CONTINUOUS)
# x_2 = model.addVar(lb = -GRB.INFINITY, ub = GRB.INFINITY, vtype = GRB.
    # CONTINUOUS)
# x_3 = model.addVar(lb = -GRB.INFINITY, ub = GRB.INFINITY, vtype = GRB.
    # CONTINUOUS)
# y = model.addVar(lb = 0, ub = GRB.INFINITY, vtype = GRB.CONTINUOUS)

# y = and(x_1, x_2, x_3)
model.addGenConstrAnd(y, [x_1, x_2, x_3], "andconstr")
# overloaded forms
```

```
model.addConstr(y == and_([x_1, x_2, x_3]), "andconstr")
model.addConstr(y == and_(x_1, x_2, x_3), "andconstr")
```

然后来看 Or 约束。假设有一系列 0-1 变量 $y, x_1, x_2, \cdots, x_n$，考虑下面的关系。

$$y = \text{or}\{x_1, x_2, \cdots, x_n\}$$

式中，当且仅当存在 $x_i, i = 1, 2, \cdots, n$ 的取值为 1 时，$y = 1$；当 $x_1, x_2, \cdots, x_n$ 的取值均为 0 时，$y = 0$。下面为代码示例。

And constraints

```
# define variables
x_1 = model.addVar(vtype = GRB.BINARY)
x_2 = model.addVar(vtype = GRB.BINARY)
x_3 = model.addVar(vtype = GRB.BINARY)
y = model.addVar(vtype = GRB.BINARY)

# y = or(x_1, x_2, x_3)
model.addGenConstrOr(y, [x_1, x_2, x_3], "orconstr")
# overloaded forms
model.addConstr(y == or_([x_1, x_2, x_3]), "orconstr")
model.addConstr(y == or_(x_1, x_2, x_3), "orconstr")
```

同样地，参与 Or 约束的所有决策变量将被自动强制转换成 0-1 变量。

3.1.3 指示约束条件

在很多情况下，我们希望保证，如果满足约束条件 $f(x_1, x_2, \cdots, x_n) > 0$，那么必须满足约束条件 $g(x_1, x_2, \cdots, x_n) \geqslant 0$；如果没有满足 $f(x_1, x_2, \cdots, x_n) > 0$，那么可以满足也可以不满足约束条件 $g(x_1, x_2, \cdots, x_n) \geqslant 0$。

为了保证这个约束，我们引入 0-1 型变量 y，加入如下约束条件：

$$-g(x_1, x_2, \cdots, x_n) \leqslant My \tag{3.5}$$

$$f(x_1, x_2, \cdots, x_n) \leqslant M(1 - y) \tag{3.6}$$

通常，M 是一个足够大的正数，使得 $f \leqslant M$ 和 $-g \leqslant M$ 对于满足问题中其他约束条件中的 $x_1, x_2, \cdots, x_n$ 所有值都成立。

可以看到，如果 $f > 0$，那么只有当 $y = 0$ 时才满足 (3.6)，当 $y = 0$ 时，由 (3.5) 可知，$g \geqslant 0$，这是我们需要满足的逻辑约束条件。此外，如果没有满足 $f > 0$，那么 (3.6) 允许 $y = 0$ 或 $y = 1$，$g < 0$ 和 $g \geqslant 0$ 的情况都可能出现。而选择 $y = 1$ 将满足 (3.5)。

指示约束条件在 Gurobi 中可以通过调用函数 `addGenConstrIndicator(binvar, binval, lhs, sense=None, rhs=None, name="")` 实现。该函数各个参数的含义如下。

- binvar：0-1 指示变量对象；
- binval：0-1 指示变量的取值（True 或者 False）；
- lhs (float, Var, LinExpr, or TempConstr)：与 0-1 指示变量相关联的线性约束的左端项表达式；
- sense (char)：线性约束的符号，可选值为 GRB.LESS_EQUAL，GRB.EQUAL，GRB.GREATER_EQUAL；
- rhs (float)：线性约束的右端项；
- name (string, optional)：约束的名称。

例如，我们想实现如果 $y=1$，则 $x_1+2x_2+x_3=1$，此类约束可以通过下面的代码实现。

Indicator constraints

```python
# y = 1 → x_1 + 2 x_2 + x_3 = 1
model.addGenConstrIndicator(y, True, x_1 + 2*x_2 + x_3, GRB.EQUAL, 1.0)
# alternative form
model.addGenConstrIndicator(y, True, x_1 + 2*x_2 + x_3 == 1.0)
# overloaded form
model.addConstr((y == 1) >> (x_1 + 2*x_2 + x_3 == 1.0))
```

3.2 线 性 化

3.2.1 分段线性函数线性化

下面将介绍如何使用 0-1 型变量建立涉及分段线性函数的最优化问题模型。

分段线性函数由几条直线段组成，分段线性函数斜率发生变化（或者函数的定义范围结束）的点称作函数的间断点。

如图 3.1 所示，该分段线性函数由 4 条直线段组成，横坐标 x 为 0、10、30、40、50 的点都是这个函数的间断点（Winston，2004）。

分段线性函数不是线性函数，一般不能使用线性规划直接对涉及线性分段函数的最优化问题进行求解。但是我们可以利用 0-1 型变量，把分段线性函数表示成线性形式。

假设一个分段线性函数 $f(x)$ 在 x 为 $b_1,b_2,\cdots,b_n$ 处有分段点。对于某个 $k(k=1,2,\cdots,n-1)$, 有 $b_k\leqslant x\leqslant b_{k+1}$。因此对于某个数 $z_k(0\leqslant z_k\leqslant 1)$, 可以把 x 记作

$$x=z_kb_k+(1-z_k)b_{k+1} \tag{3.7}$$

由于 $f(x)$ 对于 $b_k\leqslant x\leqslant b_{k+1}$ 是线性的，所以我们可以把 $f(x)$ 记作

$$f(x)=z_kf(b_k)+(1-z_k)f(b_{k+1}) \tag{3.8}$$

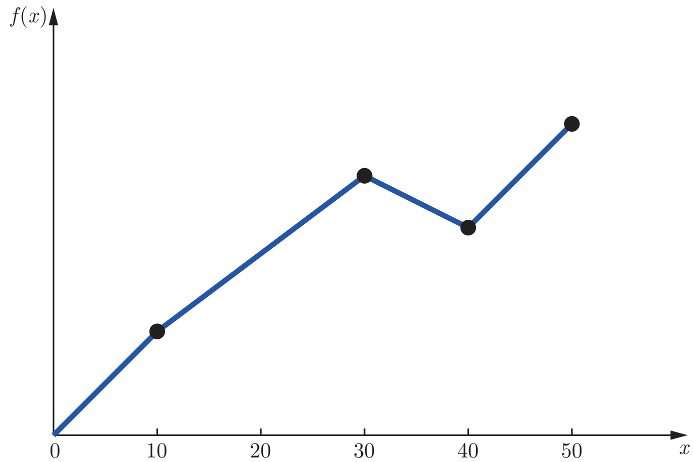

图 3.1　分段线性函数

接下来介绍利用线性约束条件和 0-1 型变量表示分段线性函数的方法。

（1）第 1 步：在最优化问题中出现 $f(x)$ 的地方，用 $z_1f(b_1)+z_2f(b_2)+\cdots+z_nf(b_n)$ 代替 $f(x)$。

（2）第 2 步：在问题中添加如下约束条件：

$$
\begin{aligned}
& z_1 \leqslant y_1 \\
& z_2 \leqslant y_1 + y_2 \\
& z_3 \leqslant y_2 + y_3 \\
& \vdots \\
& z_{n-1} \leqslant y_{n-2} + y_{n-1} \\
& z_n \leqslant y_{n-1} \\
& y_1 + y_2 + \cdots + y_{n-1} = 1 \\
& z_1 + z_2 + \cdots + z_n = 1 \\
& x = z_1b_1 + z_2b_2 + \cdots + z_nb_n \\
& y_i \in \{0,1\}, \qquad \forall i = 1,2,\cdots,n-1 \\
& z_i \geqslant 0, \qquad \forall i = 1,2,\cdots,n
\end{aligned}
$$

针对分段函数线性化，我们还有另一种形式的描述。

假如有如图 3.2 所示的分段线性函数。

该分段线性函数 x 轴和 y 轴的分段区间为

$$
\begin{aligned}
& x\text{ 轴：}[0,a_1),[a_1,a_2),[a_2,a_3] \\
& y\text{ 轴：}[0,b_1),[b_1,b_2),[b_2,b_3]
\end{aligned}
$$

间断点则为

$$(0,0)\,,(a_1,b_1)\,,(a_2,b_2)\,,(a_3,b_3)$$

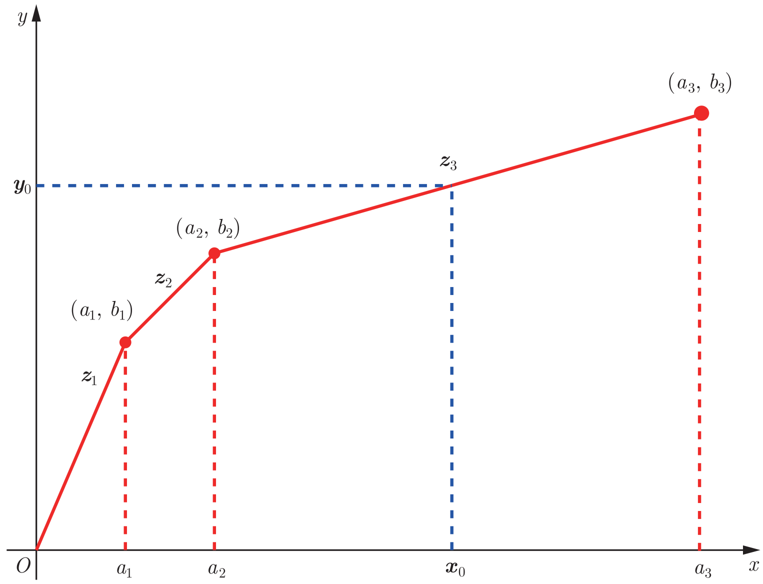

图 3.2 分段线性函数线性化

这里的目的是给定一个 x_0，通过约束，使其获得相应的 y_0。为此，引入一个辅助变量 $z \in \{0,1\}$，该变量的取值与 x 所在的横轴区间有关，即

$$z_i = \begin{cases} 1, & x \text{ 落在第 } i \text{ 个分段区间内} \\ 0, & \text{其他} \end{cases}$$

下面不加证明地给出一个结论。

点 $X^{(1)}(x_1,y_1)$ 和 $X^{(2)}(x_2,y_2)$ 为平面内两点，则 $X^{(1)}$ 和 $X^{(2)}$ 连线上任意一点 X 可以表示为

$$X = (1-\alpha)X^{(1)} + \alpha X^{(2)} \tag{3.9}$$

且 $X^{(1)}X = \alpha X^{(1)}X^{(2)}$。

这一点很容易得到，因为只看横坐标, 当 $X^{(1)}X = \alpha X^{(1)}X^{(2)}$ 时，有

$$x = x_1 + \alpha(x_2 - x_1) = (1-\alpha)x_1 + \alpha x_2$$

因此，对于曲线上任何一个点 (x,y)，均可被表示为如下的形式：

$$x = \beta_0 a_0 + \beta_1 a_1 + \beta_2 a_2 + \beta_3 a_3 = \sum_{k \in K} \beta_k a_k \tag{3.10}$$

$$y=\beta_0 b_0+\beta_1 b_1+\beta_2 b_2+\beta_3 b_3=\sum_{k\in K}\beta_k b_k \tag{3.11}$$

并且，$\beta_0,\beta_1,\beta_2,\beta_3$ 中只能有 1 个或者 2 个大于 0，其余均等于 0，并且满足

$$\beta_0+\beta_1+\beta_2+\beta_3=1 \tag{3.12}$$

下面来看如何将分段线性函数转化成约束（这里假设 $a_0=0$，但是 β_0 是不能忽略的，必须有，也就是 β 比 z 多 1 个），假设一共有 $K(K\geqslant 1)$ 个区间段。

（1）对于第 1 个区间段，因为都是左闭右开区间，如果 $z_1=1$，则必然有 $\beta_0>0,\beta_1\geqslant 0$。因为是左闭右开区间, 左边一定能取到，也有可能取到左边区间端点。但是右边断点一定取不到，因此 $\beta_1<1$，故有 $\beta_0>0$，并且 β_0 也可能取到 1，取到 1 时，说明刚好落在区间左端点。如果 $z_1=0$，则 $\beta_0\geqslant 0,\beta_1\geqslant 0$。因此，$z$ 的取值在这个层面没有必然联系。

（2）对于前 $K-1$ 个区间，均与上述同理。

（3）先看第一段，$z_1=0$，必有 $\beta_0=0,\beta_1\geqslant 0\Rightarrow\beta_0-Mz_1\leqslant 0$。

（4）再看最后一段，如果 $z_K=0$, 必有 $\beta_K=0,\beta_{K-1}\geqslant 0\Rightarrow\beta_K-Mz_K\leqslant 0$。

（5）最后看中间的区间段，当 $z_{k-1}=0$ 且 $z_k=0$ 时，必有 $\beta_{k-1}=0\Rightarrow\beta_{k-1}-M\left(z_{k-1}+z_k\right)\leqslant 0$。

由于 M 是 z 的一个上界，这里取 $M=1$，因此只需要添加如下的约束：

$$x=\sum_{k=0}^{K}\beta_k a_k \tag{3.13}$$

$$y=\sum_{k=0}^{K}\beta_k b_k \tag{3.14}$$

$$\sum_{k=0}^{K}\beta_k=1 \tag{3.15}$$

$$\sum_{k=0}^{K}z_k=1 \tag{3.16}$$

$$\beta_0-z_1\leqslant 0 \tag{3.17}$$

$$\beta_K-z_K\leqslant 0 \tag{3.18}$$

$$\beta_{k-1}-(z_{k-1}+z_k)\leqslant 0,\quad \forall k\in K,k\neq 0,k\neq K \tag{3.19}$$

$$z_0=0 \tag{3.20}$$

$$0\leqslant\beta_k\leqslant 1,z_k\in\{0,1\},\quad \forall k\in K \tag{3.21}$$

此外，对于 $\ln x,\log x,\sin x,\cos x,\tan x,\mathrm{e}^x,a^x,x^a$ 等形式的非线性函数，均可通过分段线性近似，进而用分段函数线性化的方法将其线性化。Gurobi 中也提供了相应的函数，可以很方便地实现这些非线性函数的分段函数线性近似，具体介绍见第 7 章。

在 Gurobi 中，分段函数的线性化是可以自动完成的，不用像上面描述的那样复杂。如果是 Python 调用 Gurobi，则对应的函数为 `addGenConstrPWL(xvar, yvar, xpts, ypts, name="")`。其中各个参数的意义如下。

- xvar：决策变量 x；
- yvar：决策变量 y；
- xpts (list of float)：分段函数 $y=f(x)$ 的 x 轴分段点；
- ypts (list of float)：分段函数 $y=f(x)$ 的 y 轴分段点；
- name (string, optional)：该约束的名称。

接下来展示一个具体例子。

Piecewise-linear function

```
# define Piecewise-linear constraints
PWL_cons = model.addGenConstrPWL(x, y, [0, 2, 3], [1, 4, 3], "PWLConstr")
```

上文中提到的 $\ln x, \log x, \sin x, \cos x, \tan x, \mathrm{e}^x, a^x, x^a$ 等形式的非线性函数，虽然不能直接调用 addGenConstrPWL() 函数完成分段函数线性化，但是可以通过调用下面的函数间接完成线性化。

- addGenConstrExp(xvar, yvar, name="", options="")：$y=\mathrm{e}^x$；
- addGenConstrExpA(xvar, yvar, a, name="", options="")：$y=a^x$；
- addGenConstrLog(xvar, yvar, name="", options="")：$y=\ln x$；
- addGenConstrLogA(xvar, yvar, a, name="", options="")：$y=\log_a x$；
- addGenConstrPow(xvar, yvar, a, name="", options="")：$y=x^a$；
- addGenConstrSin(xvar, yvar, name="", options="")：$y=\sin(x)$；
- addGenConstrCos(xvar, yvar, name="", options="")：$y=\cos(x)$；
- addGenConstrTan(xvar, yvar, name="", options="")：$y=\tan(x)$。

3.2.2 含绝对值形式的线性化

对于目标函数含有绝对值的情况，如 $\min |x|$，可以通过引入辅助变量进行线性化。考虑下面的含有绝对值的非线性规划。

$$\min \quad |x_1|+|x_2|$$
$$x_1, x_2 \in \mathbb{R}$$

首先我们引入辅助变量，令

$$x_1^+ = \max\{0, x_1\}, \quad x_1^- = \max\{0, -x_1\}$$

$$x_2^+ = \max\{0, x_2\}, \quad x_2^- = \max\{0, -x_2\}$$

可以得出：

$$
\begin{aligned}
|x_1| = x_1^+ + x_1^-, \quad x_1 = x_1^+ - x_1^- \\
|x_2| = x_2^+ + x_2^-, \quad x_2 = x_2^+ - x_2^-
\end{aligned}
$$

因此，原问题等价于：

$$
\begin{aligned}
\min \quad & x_1^+ + x_1^- + x_2^+ + x_2^- \\
& x_1 = x_1^+ - x_1^- \\
& x_2 = x_2^+ - x_2^- \\
& x_1, x_2 \in \mathbb{R}, x_1^+, x_1^-, x_2^+, x_2^- \geqslant 0
\end{aligned}
$$

这就完成了线性化。

在 Gurobi 中，可以通过调用函数 `addGenConstrAbs(resvar, argvar, name="")` 或者函数 `abs_()` 很容易实现绝对值的线性化。比如，我们要实现上述模型的线性化，只需要编写下面的代码即可。

Abs constraints

```
model = Model()

x_1 = model.addVar(lb = -GRB.INFINITY, ub = GRB.INFINITY, vtype = GRB.CONTINUOUS)
x_2 = model.addVar(lb = -GRB.INFINITY, ub = GRB.INFINITY, vtype = GRB.CONTINUOUS)
y_1 = model.addVar(lb = 0, ub = GRB.INFINITY, vtype = GRB.CONTINUOUS)
y_2 = model.addVar(lb = 0, ub = GRB.INFINITY, vtype = GRB.CONTINUOUS)

# method 1
model.addGenConstrAbs(y_1, x_1)  # y_1 = abs(x_1)
model.addGenConstrAbs(y_2, x_2)  # y_2 = abs(x_2)

# method 2 : 使用重载过的函数abs_()   (overloaded form)
# model.addConstr(y_1 == abs_(x_1) )  # y_1 = abs(x_1)
# model.addConstr(y_2 == abs_(x_2) )  # y_2 = abs(x_2)

model.setObjective(y_1 + y_2, GRB.MINIMIZE)  # min abs(x_1) + abs(x_2)
```

上述例子中，我们通过引入两个非负的辅助变量 y_1, y_2，并且令 $y_1 = |x_1|, y_2 = |x_2|$，然后将目标函数设置为 $\min \ y_1 + y_2$，从而实现了线性化。

之后的内容部分参考自 Gurobi 的教程和用户手册。

3.2.3 含乘积形式的线性化

若目标函数含有两个变量乘积形式，则可以通过引入变量 y 进行线性化。由于变量有可能是 0-1 变量或者整数变量，所以分如下 4 种情况进行讨论。

第 1 种情况：

$$\begin{aligned} \min \quad & x_1x_2 \\ \text{s.t.} \quad & x_1 \in \{0,1\} \\ & x_2 \in \{0,1\} \end{aligned}$$

令 $y = x_1x_2$，则上述非线性项可以通过添加下面的约束实现线性化：

$$\begin{aligned} \min \quad & y \\ \text{s.t.} \quad & y \leqslant x_1 \\ & y \leqslant x_2 \\ & y \geqslant x_1 + x_2 - 1 \\ & x_1, x_2, y \in \{0,1\} \end{aligned}$$

第 2 种情况：

$$\begin{aligned} \min \quad & x_1x_2 \\ \text{s.t.} \quad & x_1 \in \{0,1\} \\ & x_2 \in [0,u] \end{aligned}$$

同样地，令 $y = x_1x_2$，则上述非线性项可以通过添加下面的约束实现线性化：

$$\begin{aligned} \min \quad & y \\ \text{s.t.} \quad & y \leqslant ux_1 \\ & y \leqslant x_2 \\ & y \geqslant x_2 - u(1 - x_1) \\ & x_1 \in \{0,1\}, x_2, y \in [0,u] \end{aligned}$$

第 3 种情况：

$$\begin{aligned} \min \quad & x_1x_2 \\ \text{s.t.} \quad & x_1 \in \{0,1\} \\ & x_2 \in [l,u] \end{aligned}$$

令 $y = x_1x_2$，则上述非线性项可以通过添加下面的约束实现线性化：

$$\min \quad y$$

$$
\begin{aligned}
\text{s.t.} \quad & y \leqslant x_2 \\
& y \geqslant x_2 - u(1 - x_1) \\
& l x_1 \leqslant y \leqslant u x_1 \\
& x_1 \in \{0, 1\} \\
& x_2 \in [l, u] \\
& y \in [0, u]
\end{aligned}
$$

第 4 种情况：

当 x_1 和 x_2 都是连续变量时，是不能等价线性化的，只能给出一个较紧的近似。对于具体细节，读者可以参考文献（Sherali and Alameddine, 1992）。

3.2.4　含分式形式的线性化

本部分参考自 gurobi 的教学文档。若目标函数是如下分式形式：

$$
\begin{aligned}
\min \quad & \frac{\sum\limits_i (c_i x_i + \alpha)}{\sum\limits_i (d_i x_i + \beta)} && (3.22)\\
\text{s.t.} \quad & \sum_i a_{ij} x_i \leqslant b_j, \qquad \forall j \in J \\
& \sum_i d_i x_i + \beta > 0 \\
& x_i \geqslant 0, \qquad \forall i \in I
\end{aligned}
$$

可以通过引入如下变量 y 进行线性化。令

$$
y = \frac{1}{\sum\limits_i (d_i x_i + \beta)} > 0
$$

代入式 (3.22) 得

$$
\begin{aligned}
\min \quad & \sum_i (c_i x_i y + \alpha y) \\
\text{s.t.} \quad & \sum_i a_{ij} x_i \leqslant b_j, \qquad \forall j \in J \\
& \sum_i d_i x_i y + \beta y = 1 \\
& y > 0, x_i \geqslant 0, \qquad \forall i \in I
\end{aligned}
$$

令

$$
z_i = x_i y
$$

得线性化最终形式为

$$
\begin{aligned}
\min \quad & \sum_i (c_i z_i + \alpha y) \\
\text{s.t.} \quad & \sum_i a_{ij} z_i \leqslant b_j y, \qquad \forall j \in J \\
& \sum_i d_i z_i + \beta y = 1 \\
& y > 0, z_i \geqslant 0, \qquad \forall i \in I
\end{aligned}
$$

3.2.5 含 max/min 形式的线性化

对于含有 max 形式的约束，如下式所示：

$$z = \max\{x, y, 3\}$$

可以通过如下方式进行线性化：

$$
\begin{aligned}
x \leqslant z, y \leqslant z, 3 \leqslant z \\
x \geqslant z - M(1 - u_1) \\
y \geqslant z - M(1 - u_2) \\
3 \geqslant z - M(1 - u_3) \\
u_1 + u_2 + u_3 \geqslant 1 \\
u_1, u_2, u_3 \in \{0, 1\}
\end{aligned}
$$

对于含有 min 形式的约束，如下式所示：

$$z = \min\{x, y, 3\}$$

可以通过如下方式进行线性化：

$$
\begin{aligned}
x \geqslant z, y \geqslant z, 3 \geqslant z \\
x \leqslant z - M(1 - u_1) \\
y \leqslant z - M(1 - u_2) \\
3 \leqslant z - M(1 - u_3) \\
u_1 + u_2 + u_3 \geqslant 1 \\
u_1, u_2, u_3 \in \{0, 1\}
\end{aligned}
$$

对于含有 max 和 min 的情形，在 Gurobi 中可以通过调用函数 `addGenConstrMax(resvar, vars, constant=None, name="")` 和 `addGenConstrMin(resvar, vars, constant=None, name="")` 实现。

例如，我们考虑下面的表达式：

$$y = \max\{x_1, x_2, \cdots, x_n, c\}$$

其中，$x_1, x_2, \cdots, x_n$ 为决策变量，c 为常数。则相应的代码如下：

Max constraints

```
# y = max(x_1, x_2, x_3, 10.0)
model.addGenConstrMax(y, [x_1, x_2, x_3], 10.0, "maxconstr")

# alternative form
model.addGenConstrMax(y, [x_1, x_2, x_3, 10.0], name="maxconstr")

# overloaded forms
model.addConstr(y == max_([x_1, x_2, x_3, 10.0]), name="maxconstr")
model.addConstr(y == max_(x_1, x_2, x_3, 10.0), name="maxconstr")
```

对于 min 的情形，处理方法类似。考虑下面的表达式。

$$y = \min\{x_1, x_2, \cdots, x_n, c\}$$

其中，$x_1, x_2, \cdots, x_n$ 为决策变量，c 为常数。相应的代码如下：

Min constraints

```
# y = min(x_1, x_2, x_3, 10.0)
model.addGenConstrMin(y, [x_1, x_2, x_3], 10.0, "minconstr")

# alternative form
model.addGenConstrMin(y, [x_1, x_2, x_3, 10.0], name="minconstr")

# overloaded forms
model.addConstr(y == min_([x_1, x_2, x_3, 10.0]), name="minconstr")
model.addConstr(y == min_(x_1, x_2, x_3, 10.0), name="minconstr")
```