# 線形計画法（シンプレックス法）

生産量、コスト、人員などのデータが一次関数として与えられているとき、目的関数に対し最適解を得る方法を **線形計画法 (LP:Linear Programming)** と呼ぶ。

例えば、ある企業が $x_1, x_2$ という 2 種類の商品を生産しているものとする。生産に使われる資源は IC、トランジスタ、抵抗であり、$x_1, x_2$ 各 1 単位を生産するために必要な資源の量は以下であるとする。

| 資源 | $x_1$ | $x_2$ | 手持ちの資源 |
|---|---|---|---|
| IC | 1 個 | 2 個 | 14 千個 |
| トランジスタ | 1 個 | 1 個 | 8 千個 |
| 抵抗 | 3 個 | 1 個 | 18 千個 |
| (商品単価) | 2 万円 | 3 万円 | |

このとき、手持ちの資源の範囲内で売上を最大にするためには、$x_1, x_2$ をそれぞれ何個生産したらよいかを考えるのが線形計画法である。

上記の例の場合、

$$\begin{cases} x_1 + 2x_2 \leq 14 \\ x_1 + x_2 \leq 8 \\ 3x_1 + x_2 \leq 18 \\ x_1, x_2 \geq 0 \end{cases}$$

のもとで、

$$2x_1 + 3x_2 = z \quad (\text{売上高})$$

が最大となるような $x_1, x_2$ を求めればよい。

これを図式化すると以下のようになる。

塗りつぶされた領域が生産可能領域であり、目的関数の直線を $z$ が大きなる方向へ平行移動して領域から離れる瞬間の $z$ の値が求める売上高の最大値である。そして、そのときの接点 $A$ の $x_1, x_2$ の値が各商品の生産量である。

さて、変数が $x_1, x_2$ の2個だけなら図から容易に計算が可能であるが、変数が3つ以上になると困難となる。そこで、コンピュータでは**シンプレックス法**という解法を利用する。

制約式として、

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n \gtreqless b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n \gtreqless b_2 \\ \quad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n \gtreqless b_m \end{cases} \quad \left( \begin{array}{l} \gtreqless \text{は} >, =, < \\ \text{のどれかという意味} \end{array} \right)$$

目的関数として、

$$z = c_1x_1 + c_2x_2 + \cdots + c_nx_n$$

が与えられたとする。

しかし、$\gtreqless$ のままでは解けないので、制約式にスラック変数を入れて等式化する。

$$\begin{array}{lllll} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n + x_{n+1} & & & & = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n & + x_{n+2} & & & = b_2 \\ & \vdots & & & \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n & & & + x_{n+m} & = b_m \\ z - c_1x_1 - c_2x_2 - \cdots - c_nx_n & & & & = 0 \end{array}$$

($x_{n+1}$〜$x_{n+m}$がスラック変数)

この連立方程式から次のような係数配列を作り、

$$\begin{array}{ccccccccc} a_{11} & a_{12} & \ldots & a_{1n} & 1 & & & & b_1 \\ a_{21} & a_{22} & \ldots & a_{2n} & & 1 & & & b_2 \\ \vdots & \vdots & & \vdots & & & \searrow & & \vdots \\ a_{m1} & a_{m2} & \ldots & a_{mn} & & & & 1 & b_m \\ -c_1 & -c_2 & \ldots & -c_m & 0 & 0 & \cdots & 0 & 0 \end{array}$$

$x_1$〜$x_n$ の係数が1になるように掃き出し演算を行えばよい。しかし、完全に等式の方程式を解くのではなく、制約条件を満たし、目的関数値が最大（もしくは最小）になるような特殊な掃き出し演算を行う必要がある。

実際のアルゴリズムは以下のようになる。

1. 最下行（目的関数の係数）の中から最小のものがある列 $y$ を探す。（列選択）
2. 最小値 $\geq 0$ なら終了。この条件は最下行の係数が全て正になったことを意味し、これ以上掃き出し演算を行なっても目的関数の値は増加しないことを意味する。
3. 1. で求めた $y$ 列にある各行の要素で各行の右端要素を除算したものが最小となる行 $x$ を探す。（行選択）
4. $x$ 行 $y$ 列をピボットにして掃き出し演算を行う。

（ここでは、掃き出し演算については説明しない。）