AMPL：数理計画のためのモデリング言語

全22章シリーズ　第1章

生産モデル
利益の最大化

圧延機の生産計画から学ぶ、線形計画法とAMPLの基本

Robert Fourer, David M. Gay, Brian W. Kernighan 著
日本語版

AGENDA

本章の内容

1.1 **2変数の線形計画問題**
圧延機の生産計画を例にLPの基礎を学ぶ

1.2 **2変数の線形計画問題をAMPLで書く**
prod0.modとAMPLセッションの実行

1.3 **線形計画モデル**
代数形式による一般モデルの記述

1.4 **線形計画モデルをAMPLで書く**
集合・パラメータ・変数・目的・制約

1.5 **モデルに下限を加える**
commitパラメータによる下限制約

1.6 **モデルに資源制約を加える**
生産段階(STAGE)への拡張と限界値

1.7 **AMPLのインタフェース**
コマンド・GUI・Web/APIの選択肢

1.1 節

問題設定：圧延機とバンド・コイル

ある製鋼会社の圧延機は、鋼スラブを2種類の半製品に加工できる。

バンド（bands）とコイル（coils）
毎時の生産速度・トン当たり利益・週間の最大生産量が製品ごとに異なる。

今週の生産時間は40時間。総利益を最大にするには、バンドとコイルを何トンずつ生産すべきか？

製品	毎時トン数	トン当たり利益	最大週間トン数
バンド	200	$25	6,000
コイル	140	$30	4,000

40 時間
今週利用できる圧延機の稼働時間

1.1 節

決定変数とLPの定式化

決定変数

XB = 生産するバンドのトン数　　XC = 生産するコイルのトン数

総所要時間は40時間を超えられない：

$$(1/200)\ XB + (1/140)\ XC \leqq 40$$

生産量にはそれぞれ上限がある：

$$0 \leqq XB \leqq 6000 \qquad 0 \leqq XC \leqq 4000$$

LP 定式化

最大化:

```
25 XB + 30 XC
```

制約:

```
(1/200) XB + (1/140) XC <= 40
0 <= XB <= 6000
0 <= XC <= 4000
```

1.1 節

手計算による解法

トン当たり利益 × 毎時トン数 = 圧延機1時間当たりの利益

$5,000 /時
バンド生産の毎時利益

$4,200 /時
コイル生産の毎時利益

バンドが有利
→ 上限の6,000トンまで生産（30時間）

残り 10 時間はコイルの生産に充当 → 1,400 トン

利益 = $25 × 6,000トン + $30 × 1,400トン

$192,000

グラフによる解法（1）実行可能領域

- 横軸 XB、縦軸 XC が生産量の組合せ（解）を表す
- 垂直線：バンドの生産上限 6,000
- 水平線：コイルの生産上限 4,000
- 斜めの線：時間制約 (1/200)XB+(1/140)XC=40
- 網掛け領域＝すべての制約を満たす実行可能領域

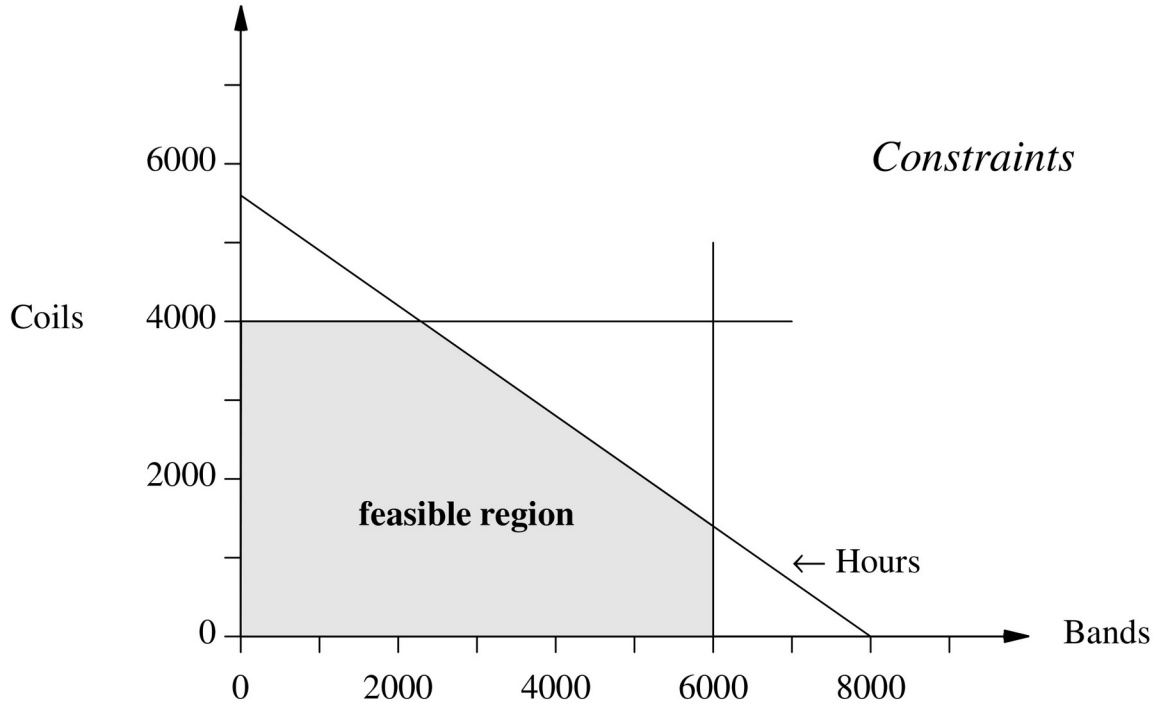

図1-1　制約条件と実行可能領域（網掛け部分）

1.1 節

グラフによる解法（2）等利益線

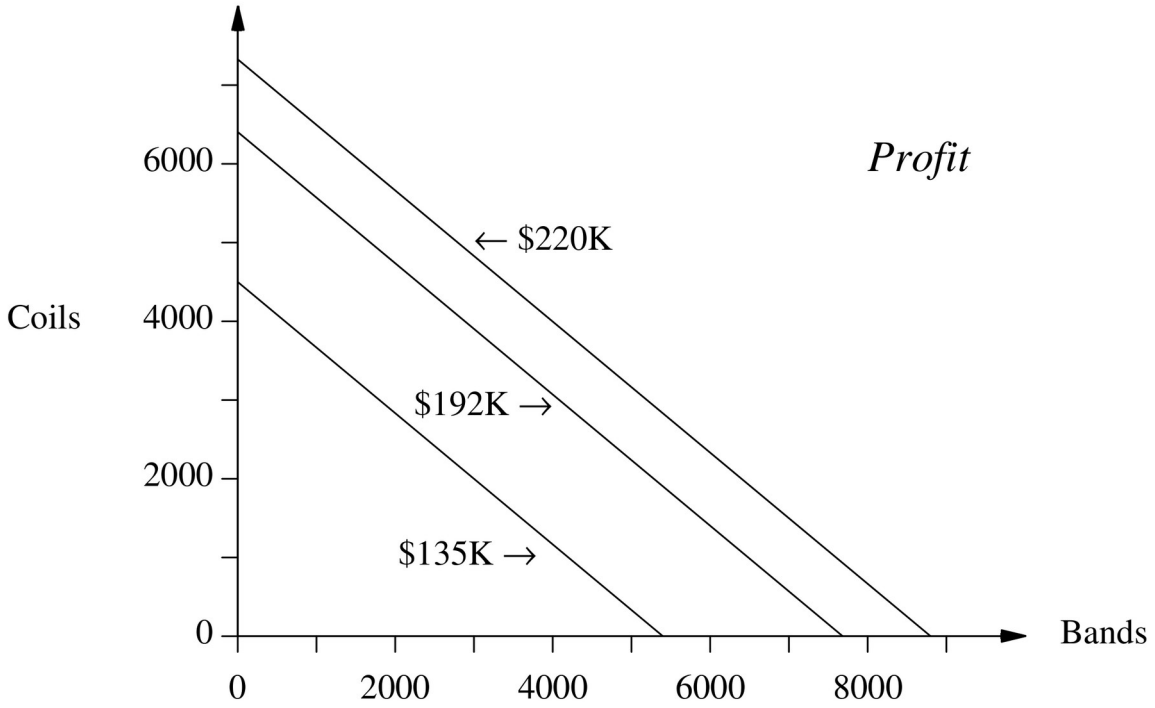

図1-2　利益（Profit）を表す等利益線

- 傾き -25/30 の平行線が、同じ利益をもたらす組合せを表す
- 例：(0, 4500) から (5400, 0) の直線 → 利益 $135,000
- 利益が高いほど、直線は右上に位置する
- 実行可能領域と交わる直線のうち、最も右上にあるものが最大利益

グラフによる解法（3）最適解

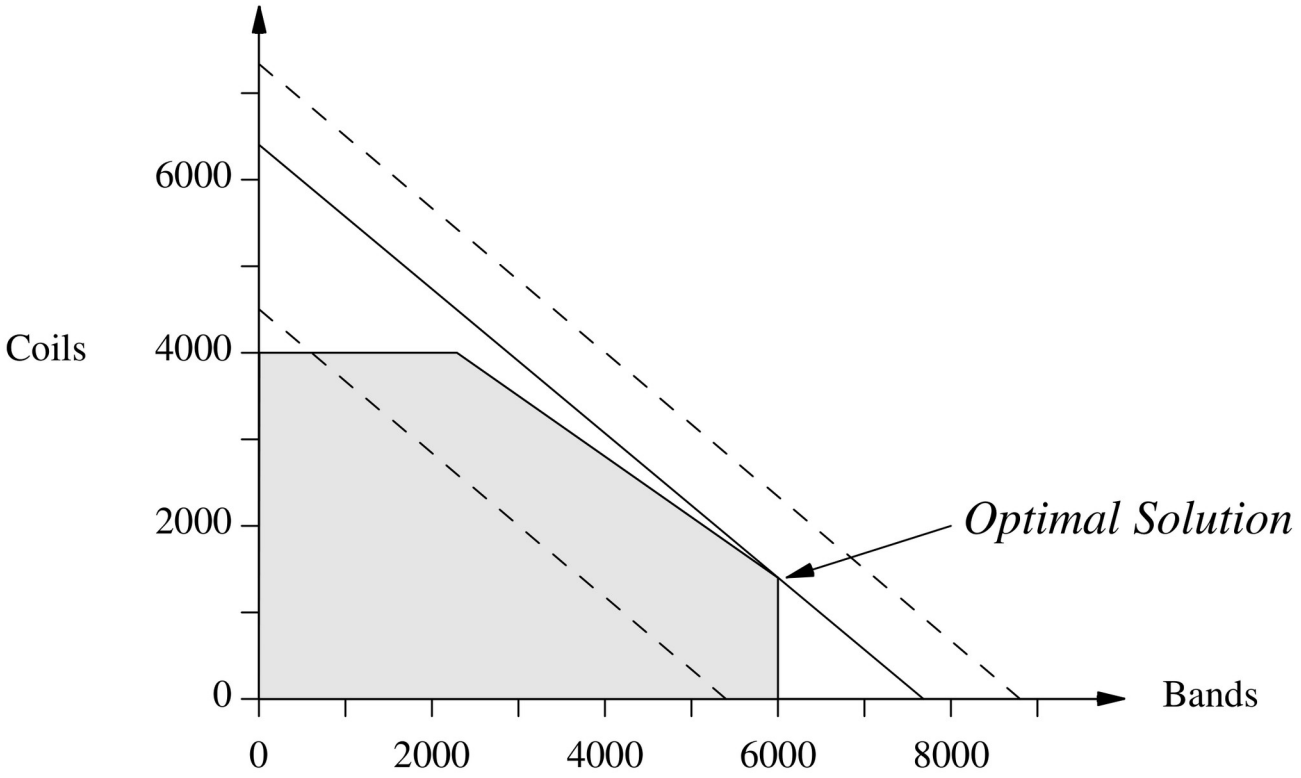

図1-3　最適解（Optimal Solution）

$220,000の等利益線は実行可能領域と交わらない。領域の隅の一点で接する直線が最適解。

6,000
XB（バンド, トン）

1,400
XC（コイル, トン）

最大利益

$192,000

1.2 節

AMPLで2変数LPを記述する

prod0.mod

```
var XB;
var XC;

maximize Profit: 25 * XB + 30 * XC;

subject to Time:
   (1/200) * XB + (1/140) * XC <= 40;
subject to B_limit: 0 <= XB <= 6000;
subject to C_limit: 0 <= XC <= 4000;
```

- var 文で各変数に名前を与える
- subject to の後に制約名（Time, B_limit, C_limit）を書く
- 掛け算には * 演算子が必須
- ≦ は <= と書く
- 代数的な記述にきわめて近い構文

1.2 節

AMPLセッションを実行する

ampl session

```
ampl: model prod0.mod;
ampl: solve;
MINOS 5.5: optimal solution found.
2 iterations, objective 192000
ampl: display XB, XC;
XB = 6000
XC = 1400
ampl: quit;
```

model
prod0.mod を読み込む

solve
ソルバーを呼び出して解く

display
変数の最適値を表示する

quit
AMPLセッションを終了する

1.3 節

一般化：線形計画モデル（代数形式）

基本の生産モデル

```
与えられるもの:
  P = 製品の集合
  a[j] = 製品jの毎時トン数
  b    = 利用可能な時間
  c[j] = 製品jのトン当たり利益
  u[j] = 製品jの最大トン数

変数: X[j] = 製品jの生産トン数

最大化: sum j in P  c[j] * X[j]
制約:  sum j in P (1/a[j]) X[j] <= b
       0 <= X[j] <= u[j]  (各 j)
```

モデルの5つの基本構成要素

集合 (sets)
製品のような要素の集まり

パラメータ (parameters)
生産速度や利益率などの定数

変数 (variables)
ソルバーが値を決定する量

目的 (objective)
最大化・最小化すべき式

制約 (constraints)
解が満たすべき条件

1.3-1.4 節

AMPLモデルとデータの分離

prod.mod

```
set P;
param a {j in P};
param b;
param c {j in P};
param u {j in P};
var X {j in P};

maximize Total_Profit:
  sum {j in P} c[j] * X[j];
subject to Time:
  sum {j in P} (1/a[j]) * X[j] <= b;
subject to Limit {j in P}:
  0 <= X[j] <= u[j];
```

prod.dat

```
set P := bands coils;
param:   a    c    u :=
 bands  200  25  6000
 coils  140  30  4000 ;
param b := 40;
```

モデルとデータの分離が鍵

製品が2種類から42種類に増えても、モデル（prod.mod）は一切変更不要。データファイルを差し替えるだけで新しい問題例を解ける。

改良モデル：読みやすさの工夫

steel.mod

```
set PROD;                  # products
param rate {PROD} > 0;     # tons/hour
param avail >= 0;          # hours/week
param profit {PROD};       # profit/ton
param market {PROD} >= 0; # sales limit

var Make {p in PROD} >= 0,
                    <= market[p];

maximize Total_Profit:
  sum {p in PROD} profit[p]*Make[p];

subject to Time:
  sum {p in PROD} (1/rate[p])*Make[p]
     <= avail;
```

- 意味の分かる名前：PROD, rate, avail, profit, market
- 変数の上下限は var 宣言の中で直接指定
- # から行末までがコメント
- 同じ2変数LPを解いても結果は同じ（利益 $192,000）

1.5 節

モデル拡張①：生産量の下限を加える

鋼板（plate）を追加すると、コイルの生産量が0になってしまう。
現実には販売の確約があるため、生産量には下限が必要。

steel3.mod（抜粋）

```
param commit {PROD} >= 0;
var Make {p in PROD} >= commit[p],
                     <= market[p];
```

製品	commit	Make	market
bands	1000	6000	6000
coils	500	500	4000
plate	750	1028.57	3500

$194,828.57

steel3.mod の最適解の利益（bands 6000, coils 500, plate 1028.57）

1.6 節

モデル拡張②：資源制約（生産段階）を加える

生産を再加熱（reheat, 35時間）と圧延（roll, 40時間）の2段階に分割する。

steel4.mod（抜粋）

```
set STAGE;  # reheat, roll
param rate {PROD,STAGE} > 0;
param avail {STAGE} >= 0;

subject to Time {s in STAGE}:
  sum {p in PROD} (1/rate[p,s])*Make[p]
     <= avail[s];
```

$190,071
最適解の総利益

$1,800 /h
reheat の限界値

限界値（双対値）

制約を1時間緩めたときの利益改善分。reheatは+1時間ごとに$1,800、rollは+1時間ごとに$3,200の追加利益。

被約費用

coilsの下限を1トン増やすと利益は約$1.86減少。bands・plateは上下限の間にあり被約費用は実質0。

1.4 節

誤りの検出と expand コマンド

エラー例

```
steel.mod, line 8:
        syntax error
context: var Make {p in PROD}
  >>> 0 <<< <= Make[p] <=
  market[p];

steel.mod, line 11:
        make is not defined
context: maximize Total_Profit:
  sum {p in PROD} profit[p] *
  >>> make[p] <<< ;
```

expand コマンド

```
ampl: expand Total_Profit;
maximize Total_Profit:
  25*Make['bands']
  + 30*Make['coils'];

ampl: expand Time;
subject to Time:
  0.005*Make['bands']
  + 0.00714286*Make['coils']
  <= 40;
```

1.7 節

AMPLの多様なインタフェース

コマンドインタフェース

文字コマンドを打ち込む方式。全機能にアクセスでき、どの環境でも共通。本書の例で使用。

グラフィカルインタフェース

Java／Tcl-Tk／Visual Basicベースなど。ボタンやメニューで標準操作を行える。

Web・APIインタフェース

クライアント・サーバー型のアクセスや、他プログラムから呼び出すAPIも利用可能。

まとめ

1. 線形計画問題（LP）は、目的と制約を変数の線形な式で記述する。
2. 変数が2個の小規模LPなら、手計算やグラフでも最適解を求められる。
3. AMPLは代数的な記述にきわめて近い構文でLPをそのまま表現できる。
4. モデルとデータを分離することが、複雑な問題を簡潔に扱う鍵となる。
5. 下限・複数段階の資源制約なども、モデルへ自然に組み込める。

次章：第2章「食餌問題：費用の最小化」へつづく