

AMPL：数理計画のためのモデリング言語

第19章　相補性問題

complements 演算子で経済均衡・ゲーム理論の相補性条件をモデル化する

目 本章の内容

19.1 相補性の発生源

経済均衡モデル、生産経済学の相補性形式、上下限・価格依存需要への拡張、他の応用

19.2 相補性制約の形式

complements 演算子の一般定義（単一不等式の対／二重不等式と式）

19.3 相補性制約を扱う

関連する解の値（接尾辞）、前処理（presolve）、制約の総称的な同義語

相補性問題とは

19.1　相補性の発生源

- さまざまな物理的・経済的現象は、2つの不等式制約が「相補的」——少なくとも一方が等号で成り立つ——であることを要求する形で最も自然にモデル化される。
- 最適化は相補性の特殊な場合とみなせる。線形・非線形計画の標準的な最適性条件そのものが相補性問題である。
- しかし他の相補性問題は最適化から生じるとは限らず、最適化問題として便利に定式化できないこともある。
- **AMPL の complements 演算子は、制約宣言の中で相補性条件を直接記述できるようにする。**

相補性条件

$$x \geq 0$$

complements

$$f(x) \geq 0$$

→ 両方成立、かつ少なくとも一方は等号で成立

x = 0 または f(x) = 0（あるいは両方）

生産経済学の最小費用モデル

19.1　生産経済学の相補性モデル

- 決定変数 Level[j]：活動 j の生産水準
- 目的関数：総生産費用 Total_Cost の最小化
- 制約 Demand：各製品の総産出が需要を満たす

図19-1　econmin.mod（生産費用最小化モデル）

```
set PROD;  set ACT;
param cost {ACT} > 0;
param demand {PROD} >= 0;
param io {PROD,ACT} >= 0;
var Level {j in ACT} >= 0;

minimize Total_Cost:
   sum {j in ACT} cost[j]*Level[j];
subject to Demand {i in PROD}:
   sum {j in ACT} io[i,j]*Level[j]
      >= demand[i];
```

実行結果（CPLEXで solve）

```
ampl: model econmin.mod;
ampl: data econ.dat;
ampl: solve;
CPLEX 8.0.0: optimal solution;
   objective 6808640.553
ampl: display Level;
Level [*] :=
P1a  1555.3   P3   147.465
P3c  1889.4   (他は 0)
```

双対値（制約 i の「価格」）

```
ampl: display Demand.dual;
AA1 16.7051   AC1  5.44585
BC1 57.818    BC2  0
NA2 0         NA3  0
```

最適化から均衡へ：2つの要件

19.1　変数 Level[j] に加え変数 Price[i] を導入

① 価格と需給の相補性

各製品について、総産出は需要を満たさなければならず、価格は非負でなければならない。生産が需要を上回るところでは価格はゼロ、価格が正であれば産出は需要に等しい。

② 水準と価値の相補性

各活動について、産出の総価値は単位費用を超えられない。費用が総価値を上回るところでは活動はゼロ、活動が正であれば総価値は費用に等しい。

この2組の相補性条件をまとめると「正方」相補性問題になる（変数数＝相補性関係数）。

① complements 構文：価格と需給

各製品 *i* について *Pri_Compl* 制約を定義

AMPL 記述

```
subject to Pri_Compl {i in PROD}:
   Price[i] >= 0 complements
      sum {j in ACT} io[i,j] * Level[j] >= demand[i];
```

- 2つの不等式が complements で結ばれるとき、両方が成り立ち、少なくとも一方は等号で成り立つ。
- この例は集合 PROD で添字づけられているため、製品ごとにこの相補性関係が設定される。

② complements 構文：水準と総価値

各活動j についてLev_Compl 制約を定義

AMPL 記述

```
subject to Lev_Compl {j in ACT}:
   Level[j] >= 0 complements
      sum {i in PROD} Price[i] * io[i,j] <= cost[j];
```

活動 j の1単位あたりの総価値 = sum{i in PROD} Price[i]*io[i,j] が費用 cost[j] を超えない。

正方（square）相補性問題

変数の数 ＝ 相補性関係の数（＝活動数＋製品数）

均衡モデルの実行（PATH ソルバー）

図19-3　*econ.mod*

econ.mod（Pri_Compl と Lev_Compl）

```
var Price {i in PROD};
var Level {j in ACT};

subject to Pri_Compl {i in PROD}:
   Price[i] >= 0 complements
      sum {j in ACT} io[i,j]*Level[j]
         >= demand[i];

subject to Lev_Compl {j in ACT}:
   Level[j] >= 0 complements
      sum {i in PROD} Price[i]*io[i,j]
         <= cost[j];
```

実行結果

```
ampl: model econ.mod;
ampl: data econ.dat;
ampl: option solver path;
ampl: solve;
Path v4.5: Solution found.
7 iterations; 33 pivots.

ampl: display
   sum{j in ACT} cost[j]*Level[j];
  = 6808640

→ LP最小費用解と一致。
Price は Demand.dual と一致。
```

上下限をもつ変数への拡張

level_min[j] <= Level[j] <= level_max[j]

混合相補性条件（二重不等式）

```
subject to Lev_Compl {j in ACT}:
   level_min[j] <= Level[j] <= level_max[j] complements
      cost[j] - sum {i in PROD} Price[i] * io[i,j];
```

費用 ＞ 総価値	水準は下限に達する（level_min）
費用 ＝ 総価値	水準は上下限の間にある
費用 ＜ 総価値	水準は上限に達する（level_max）

上下限付きモデルの実行結果

図19-4　econ2.mod を PATH で解く

実行結果（display level_min, Level, level_max）

```
ampl: model econ2.mod;  ampl: data econ2.dat;
ampl: option solver path;
ampl: solve;
Path v4.5: Solution found.  9 iterations; 8 pivots.

ampl: display level_min, Level, level_max;
:   level_min Level level_max :=
P1      240      240    1000
P1a     270     1000    1000
P2      220      220    1000
P2a     260      680    1000
P2b     200      200    1000
P3      260      260    1000
P3c     220     1000    1000
P4      240      240    1000
;
```

結果は、図19-1の変数に同じ上下限を課したLPの解と一致する。

価格に依存する需要への拡張

19.1　LP に等価でない相補性問題

- 需要を固定とせず、価格の減少関数とみなす方が現実的。
- 線形の場合：demzero[i] − demrate[i]*Price[i]
- **Price[i] が complements の両側に現れるため、等価な LP は存在しない。**

図19-5　econnl.mod（抜粋）

```
subject to Pri_Compl {i in PROD}:
  Price[i] >= 0 complements
   sum{j in ACT} io[i,j]*Level[j] >=
     demzero[i] - demrate[i]*Price[i];
```

実行結果（PATH）

```
ampl: model econnl.mod;
ampl: data econnl.dat;
ampl: option solver path;
ampl: solve;
Path v4.5: Solution found.
11 iterations; 11 pivots.

ampl: display Level;
Level [*] :=
  P1   240    P2a  260
P1a   710.156 P2b  200
  P2   220    P3   260
P3c   939.063  P4  240
;
```

→ 需給均衡が生産水準を
押し下げる。

他の相補性モデルと応用

19.1　経済均衡以外の相補性問題

MPEC（均衡制約付き数理計画）

目的関数を追加し、等式・不等式・相補性制約を混在させる。解法はまだ実験的段階。

物理系の均衡

力の均衡条件のモデル。相補性制約は2物体の接触／非接触の離散化を表す。

ゲーム理論（ナッシュ均衡）

バイマトリクスゲームのナッシュ均衡は相補性条件で特徴づけられる（各手の確率と期待利得）。

相補性制約の一般形式

19.2　complements 演算子の構文

① 単一不等式どうし

```
single-inequality complements single-inequality ;
```

両方の不等式が成り立ち、少なくとも一方が等号で成り立つときに満たされる。

② 二重不等式と式（混合相補性）

```
double-inequality complements expression ;
expression complements double-inequality ;
```

- 下側 <= または右側 >= が等号 → 式は 0 以上
- 上側 <= または左側 >= が等号 → 式は 0 以下
- どちらの側も等号でない → 式は 0 に等しい

解に関連する値：接尾辞

19.3　cname.Lsuf ／ cname.Rsuf ／ cname.slack

- cname.Lsuf と cname.Rsuf は complements の左オペランド・右オペランドに対応。
- cname.slack は、単一不等式の対では Lslack と Rslack の小さい方。両方満たされていれば非負、厳密に相補性が成り立てば 0。
- 接尾辞なしで cname が現れると cname.slack と解釈される。

display Lev_Compl.Llb, .Lbody, .Rbody, .Rslack

```
ampl: display Lev_Compl.Llb,
ampl?  Lev_Compl.Lbody,
ampl?  Lev_Compl.Rbody,
ampl?  Lev_Compl.Rslack;
:  Llb   Lbody   Rbody   Rslack :=
P1   240    240  1392.86  Infinity
P1a  270   1000  -824.286 Infinity
P2   220    220   264.286 Infinity
P2a  260    680  5.00e-12 Infinity
;
```

前処理（presolve）による簡単化

19.3　相補性制約特有の簡単化

option show_stats 1 の出力

```
Presolve eliminates 16 constraints
   and 2 variables.
Presolve resolves 2 of 14
   complementarity conditions.
Adjusted problem:
12 variables, all linear
12 constraints, all linear
12 complementarity conditions
```

28
_ncons

14
_nccons

12
_sncons

12
_snccons

需要が最小産出量を常に下回らない製品（NA2・NA3）では Price を 0 に固定し、問題から除去できる。

制約の総称的な同義語

19.3　con → ccon への置き換え

モデル作成者の視点（前処理前）

```
_nccons    相補性制約の数
_cconname  相補性制約の名前
_ccon      相補性制約の同義語

各相補性制約 cname は
cname.L と cname.R の2つの
通常制約に展開される。
```

ソルバーの視点（前処理後・標準形）

```
expr complements
   lbound <= var <= ubound
expr <= 0 complements var <= ubound
expr >= 0 complements lbound <= var

solexpand で送信形を確認できる
```

expand Pri_Compl['AA1'] の結果（例）

```
subject to Pri_Compl['AA1']:
    Price['AA1'] >= 0
 complements
    60*Level['P1'] + 8*Level['P1a'] + ... >= 70000;

→ _con[1] = "Pri_Compl['AA1'].L"、_con[2] = "...R"
```

✓ まとめ

1. 相補性条件は「2つの不等式のうち少なくとも一方が等号で成立」を表す自然なモデル化手段であり、最適化の最適性条件もその特殊例である。

2. AMPL の complements 演算子は、単一不等式どうし、あるいは二重不等式と式という2つの構文で相補性制約を宣言できる。

3. 経済均衡モデルでは、価格と需給、活動水準と総価値のあいだの相補性が「正方」相補性問題を構成し、PATH ソルバーで解ける。

4. 上下限をもつ変数や価格依存の需要関数への拡張により、LP に等価でない相補性問題も自然に表現できる。

5. 接尾辞（.Lbody, .Rbody, .slack）、前処理、総称的な同義語（_ccon 等）により、相補性制約も通常の制約と同様に扱える。

次章：第20章へ続く