応用一般均衡分析入門
第 4 章：MCP（Mixed Complementarity Problem）の解法 *

武田 史郎 †

Date: 2024/01/28,
Version 4.0

目次

* このファイルの配布場所: https://shirotakeda.github.io/ja/research-ja/cge-howto.html

† 所属：京都産業大学経済学部．Website: https://shirotakeda.github.io/ja/

1 導入

今回の内容について。

- 第 3 章では GAMS で連立方程式を解く問題を扱った。第 3 章では、連立方程式を解く問題は GAMS では「MCP（mixed complementarity problem）」というタイプの問題になると説明した。
- 「MCP ＝連立方程式を解く問題」と考えても大きな間違いではないが、厳密には異なっている。GAMS で CGE モデルを解く手法を深く理解するには、MCP がどのような問題で、GAMS でその MCP がどのように扱われるかを正確に理解しておくのが望ましい。
- 第 4 章では MCP とはどのようなタイプの問題で、それを GAMS で解くにはどのようにプログラムを記述するかを説明する。

2 MCP の例：部分均衡モデル

2.1 直感的な説明

厳密な説明は後に廻すとして、まず直感的に MCP の説明をしておこう。例として、**「完全競争市場を仮定した部分均衡モデルにおいて均衡を求めるというケース」**を考える。完全競争市場＋部分均衡という仮定より、企業側（供給側）の行動は供給曲線、消費者側（需要側）の行動は需要曲線で表現される。これを表したのが図 1 である。供給曲線と需要曲線の交点において均衡価格と均衡の数量が決定される。数式では需要と供給が等しくなるように価格が決まるという条件として表現できる。これが部分均衡モデルにおいてごく普通に想定されるケースである。

しかし、実際には図 1 のように供給曲線と需要曲線がきれいに交わるとは限らず、図 2 のようなケースも考えられる。図 2 のパターン B は供給曲線が需要曲線よりも非常に高い位置にあるというケースであり、均衡価格は正になるが、均衡供給量・需要量はゼロとなる。パターン C は逆に供給曲線が需要曲線に比べ非常に低い位置にあるというケースであり、均衡価格が 0 となり、均衡において超過供給が発生する。もしこれらのパターンが実現するとしたら、単純な「需要=供給」という条件では均衡を求めることはできない。

事前に図 1 のパターン A のケースになることがはっきりしていれば簡単であるが、シミュレーションによって均衡を計算するような場合にはパターン B やパターン C となるケースを事前には排除できない場合がある。パターン B やパターン C のようなケースも可能性として含んだ形で均衡条件を記述し、解くのが MCP である。上のような状況は部分均衡モデルに限らず、一般均衡モデルにもよく現れることであり、CGE 分析でもこの MCP の表現が出てくる。以下では上の問題を数式で導出し、それを GAMS で解く手順について解説していく。

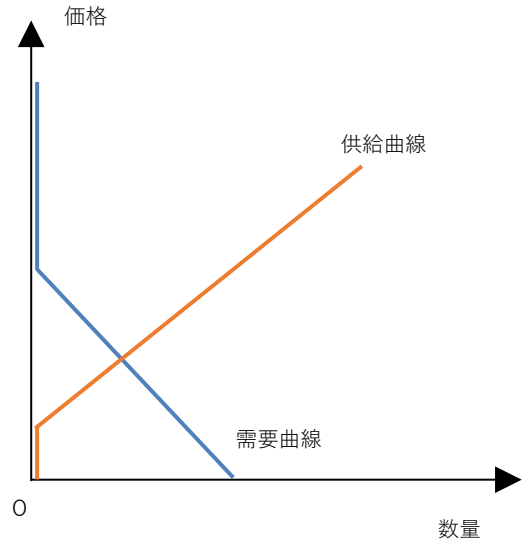

図 1：パターン A

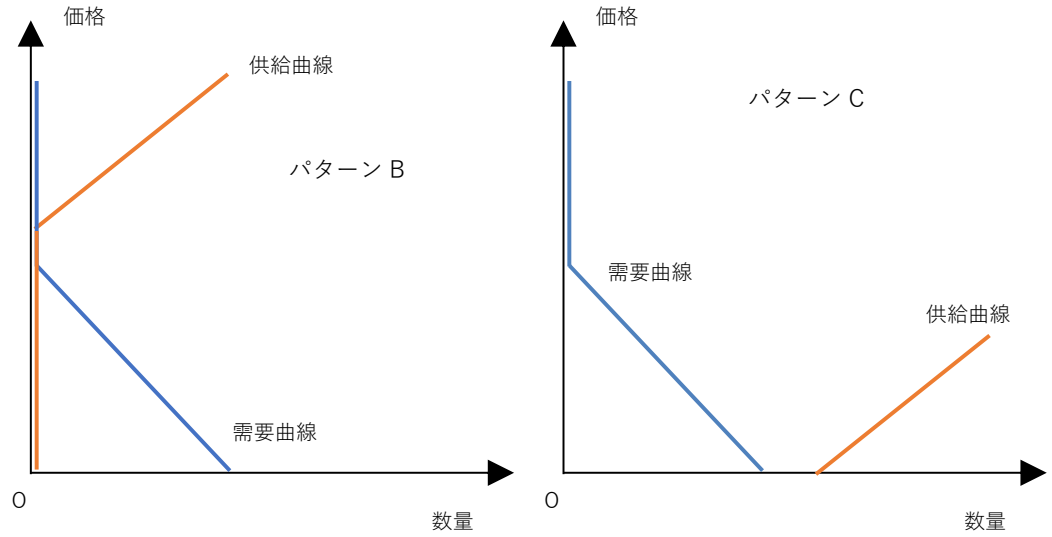

図 2：左側がパターン B、右側がパターン C

2.2 数式による説明

2.2.1 企業

まず企業側を考える。企業はプライステイカーとして行動し、利潤最大化を目指して生産量を決定すると仮定する。企業の費用は次のように表されているとする。

$$c(q) = \alpha q + \frac{1}{2}\beta q^2 \tag{1}$$

ただし $c(q)$ は総費用、q は生産量、$\alpha > 0$ と $\beta > 0$ は外生的なパラメータである。(1) 式は、総費用が生産量の二次関数で、かつ 凸関数であることを意味している[1)]。この費用を使うと企業の利潤 π は次式のように表現できる。

$$\pi = pq - c(q) = pq - \left[\alpha q + \frac{1}{2}\beta q^2\right] \tag{2}$$

もし生産量 q に何の制約もないならば利潤最大化の条件は次式となる。

$$\frac{d\pi}{dq} = p - [\alpha + \beta q] = 0$$

つまり、「価格＝限界費用」という条件である。しかし、この式によって生産量を決定する場合、パラメータと価格の値によっては生産量が負になってしまう場合がでてくる。例えば、$p = 1$、$\alpha = 2$、$\beta = 2$ のケースでは $q = -0.5$ となる。

生産量が負になるのはおかしいので、このタイプの問題を解くときには非負制約を課しておく必要がある。生産量が非負という制約付きで、(2) 式の利潤を最大化するという問題の条件は、Kuhn-Tucker 条件と呼ばれる次の式で与えられる[2)]。

$$\begin{aligned} &p - [\alpha + \beta q] \leq 0 \\ &q \geq 0 \\ &(p - [\alpha + \beta q])q = 0 \end{aligned}$$

最後の式は、$p - [\alpha + \beta q] < 0$、つまり限界費用が価格を上回るときには、$q = 0$、つまり生産量はゼロでなければならず、逆に $q > 0$、つまり生産量が正になるなら $p - [\alpha + \beta q] = 0$、つまり限界費用が価格に等しくなっていなければならないということを意味する。この 3 本目の式は**「complementary slackness 条件」**と呼ばれる。これにより生産量に非負制約を課した形での利潤最大化条件が表現できる。

---

1) $c''(q) > 0$ であるので凸関数である。凸関数であれば後の利潤最大化の二階の条件が自動的に満たされることになる。(1) 式で β の前に 1/2 がかけられているが、これは供給関数を単純な形にするためであり、それ以外に深い意味はない。1/2 をかけないでも実質的には何も変わらない。

2) 不等号制約付きの最適化問題について詳しくは、例えば、Simon and Blume (1994, p.439)、あるいは Mas-Colell et al. (1995, p.959) を参照されたい。

企業が決めるのは供給量（supply）であるので、以下では q の代わりに、s という変数を利用することにする。また、以下では上の式を少し変形した次の形式を用いることにする。

$$\alpha + \beta s - p \geq 0 \tag{3}$$

$$s \geq 0 \tag{4}$$

$$(\alpha + \beta s - p)s = 0 \tag{5}$$

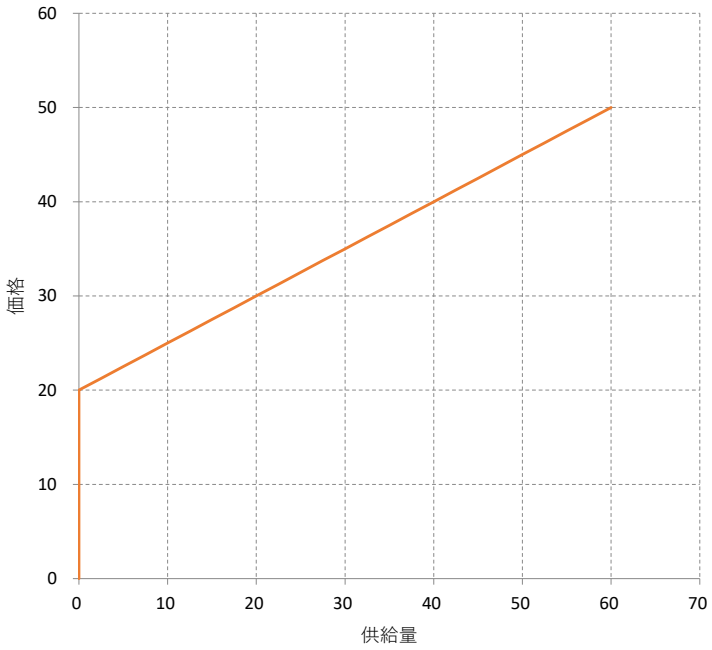

図 3：供給曲線

図 3 は $\alpha = 20$、$\beta = 0.5$ のときの、(3)〜(5) 式を満たす $\{p, s\}$ の組み合わせ、つまり供給曲線を表している[3)]。価格が 20 を下回るときには供給量は 0 となり、供給曲線は縦軸に張り付くような形状になる。費用関数を生産量について 2 次関数と仮定していることから、限界費用は生産量について 1 次関数になる。そのため供給曲線は直線となる。

2.2.2 消費者

次に消費者側を考える。消費者は q 財、v 財という 2 つの財を消費しており、次の効用関数を持っているとする。

$$u = u(q, v) = \gamma q - \frac{1}{2}\epsilon q^2 + v \tag{6}$$

ただし、u は効用、q は q 財の消費量、v は v 財の消費量、$\gamma > 0$、$\epsilon > 0$ は外生的なパラメータである[4)]。v 財をニュメレールとし、q の価格を p、所得を m と置くと、予算制約式は次のように表

3) (p, s) の組み合わせを求め、それを基に Excel で散布図を描いたグラフである。
4) (6) 式のように $u(q, v) = f(q) + v$ と書ける効用関数は準線形（quasi-linear）の効用関数と呼ばれる。

現できる。

$$pq + v = m$$

これにより (6) 式から v を消去できる。

$$u = \gamma q - \frac{1}{2}\epsilon q^2 + m - pq$$

ここでもし q 財の消費量に何ら制約がないのなら、効用を最大化する条件は次式で与えられる[5)]。

$$\frac{du}{dq} = \gamma - \epsilon q - p = 0 \tag{7}$$

しかし、企業側でも問題になったように、(7) 式では消費量が負になる可能性が生じてしまう。そこで消費についても非負制約を課して最適化問題を解く必要がある。そのための Kuhn-Tucker 条件は次式となる。

$$\gamma - \epsilon q - p \leq 0$$
$$q \geq 0$$
$$(\gamma - \epsilon q - p)q = 0$$

最後の式は先程と同じように解釈すればよい。すなわち、$\gamma - \epsilon q - p < 0$、つまり限界効用が価格（消費者にとっての限界費用）よりも小さいなら、$q = 0$、つまり消費量はゼロとなっていなければならず、逆に $q > 0$、つまり消費量が正なら、$\gamma - \epsilon q - p = 0$、つまり限界効用が価格に等しくなっていなければならないということである。この 3 式によって、消費量が 0 になる可能性も含めた形での効用最大化条件が表現される。

以下では、需要量（消費量）を d という変数で表現する。また、ここでも不等号が左側を向くように表現しなおす。

$$p - [\gamma - \epsilon d] \geq 0 \tag{8}$$
$$d \geq 0 \tag{9}$$
$$(p - [\gamma - \epsilon d])d = 0 \tag{10}$$

図 4 は $\gamma = 40$、$\epsilon = 2$ のときの (8) 〜(10) 式を満たす $\{p, q\}$ の組み合わせ、つまり需要曲線を描いたものである。 $p \geq 40$ のときには需要量は 0 となり、供給曲線と同様に縦軸に張り付く部分が生じる。

2.2.3 市場均衡

ここまでで、主体的均衡、つまり利潤最大化と効用最大化を考えた。その利潤最大化と効用最大化から供給量と需要量が決まってくる。均衡を考えるにはさらに市場均衡を考える必要がある。も

5) 消費者についてもプライステイカーと仮定している。

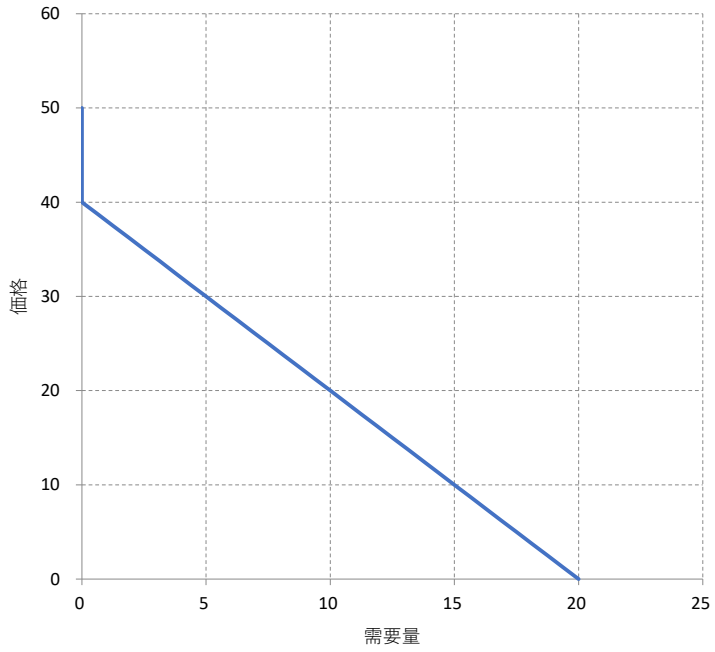

図 4：需要曲線

し、価格が必ず正になるとわかっているのなら、市場均衡条件は

$$s = d$$

ですむ。つまり、供給量と需要量が等しくなるという条件である。しかし、価格が必ずしも正にならない、つまり価格が 0 になっても供給が需要を上回ってしまうというケースを事前には排除できないかもしれない。そのような場合には、市場均衡条件は次のように修正される。

$$s - d \geq 0 \tag{11}$$

$$p \geq 0 \tag{12}$$

$$(s - d)p = 0 \tag{13}$$

ここでもやはり 3 番目の complementary slackness 条件が入ってくる。これは、$s > d$、つまり供給が需要を上回っているなら、$p = 0$、つまり価格が 0 になっていなければならず、逆に $p > 0$、つまり価格が正なら、$s - d = 0$、つまり供給は需要と等しくなっていなければならないということである。

2.2.4 均衡条件（まとめ）

以上より、供給量、需要量、価格に対して非負制約を課した形での均衡条件は (3)〜(5) 式、(8)〜(13) 式の 9 本で表される。ここまで利用した表現では一つの変数を決定するのにそれぞれ 3 本ずつの条件が必要になった。このままでは表現が非常に煩雑になるので、次のように簡略化した表

現を用いる。すなわち

$$f(x) \geq 0 \qquad x \geq 0 \qquad f(x)x = 0$$

という式を以下では

$$f(x) \geq 0 \perp x \geq 0$$

というように表現する。この表記を使う 9 本の均衡条件式は次のよう表せる。

$$\alpha + \beta s - p \geq 0 \perp s \geq 0 \tag{14}$$

$$p - [\gamma - \epsilon d] \geq 0 \perp d \geq 0 \tag{15}$$

$$s - d \geq 0 \perp p \geq 0 \tag{16}$$

以上のように

- 等号だけではなく、不等号も条件式に入ってくる
- Complementary slackness 条件が入ってくる

というような問題を mixed complementarity problem（MCP）という[6)]。普通の等号のみを含む連立方程式は MCP の特殊なケースとして扱える。GAMS ではこの MCP を解くための専用のソルバー（MCP ソルバー）が存在している。

3 MCP の定義

前節で MCP がどのような問題かおおざっぱに説明したが、ここで一応、厳密な定義を紹介しておく。

3.1 非線形の連立方程式

まず、MCP ではなく普通の非線形連立方程式（system of nonlinear equations）は次のような形式で表現される。

$$f_i(x_1, \cdots, x_n) = 0 \qquad i = \{1, \cdots, n\} \tag{17}$$

連立方程式を解くとは (17) 式を満たす $\mathbf{x} = \{x_1, \cdots, x_n\}$ を求めるという問題である。

3.2 MCP の一般的な定義

注： MCP については、『PATH のマニュアル』が詳しい。

一般的な MCP とは、$\forall i$ について、次の 3 つの条件のどれかを満たす $\mathbf{x} = \{x_1, \cdots, x_n\}$ を見つけ

6) MCP は実際にはもっと一般的な形で表現される。(14)〜(16) 式は MCP の特殊ケースである。

るという問題である。

$$f_i(x_1, \cdots, x_n) = 0 \qquad l_i < x_i < u_i \tag{18}$$

$$f_i(x_1, \cdots, x_n) > 0 \qquad x_i - l_i = 0 \tag{19}$$

$$f_i(x_1, \cdots, x_n) < 0 \qquad u_i - x_i = 0 \tag{20}$$

ただし、$f_i(\mathbf{x})$ は $\mathrm{R}^n \to \mathrm{R}$ の関数である。

3.3 変数に非負制約だけあるケース

一般的な MCP は 3.2 節のような問題であるが、実際によく見られるのはその特殊ケースである。特によくあるのが変数に対する制約としては非負制約のみが存在するケースである。これは、一般的な定義において、$l_i = 0$、$u_i = +\infty$ と設定したケースである。

$$f_i(x_1, \cdots, x_n) \geq 0 \qquad x_i \geq 0 \qquad f_i(x_1, \cdots, x_n) x_i = 0$$

4 GAMS のコード

以下で GAMS でこの問題を解くためのコード（mcp_sample.gms というプログラム）を説明しよう

4.1 コードの解説

それでは以下でコードを解説していく。やはりここでも自分でコードを書き、少しずつ実行、計算結果を確認しながら説明を読むようにして欲しい。

```
$title MCP (Mixed complementarity problem) の解き方
$ontext
Time-stamp:     <2022-01-12 15:37:40 st>
First-written:  <2013/10/11>
$offtext
```

まず、ここはプログラムの説明部分である。ここについては第 3 章で説明した。

```
parameter
    alpha       Parameters for cost function
    beta        Parameters for cost function
    gamma       Parameters for utility function
    epsilon     Parameters for utility function;

alpha = 0;
beta = 0.5;
```

```
gamma = 10;
epsilon = 2;
display alpha, beta, gamma, epsilon;
```

ここでは費用関数、効用関数のパラメータを設定している。これらは本来費用関数、効用関数のパラメータであるが、各パラメータは結局

- alpha は供給曲線の切片
- beta は供給曲線の傾き
- gamma は需要曲線の切片
- (-)epsilon は需要曲線の傾き

というように解釈し直せる。

```
*       ---------------------------------------------------------------
*       内生変数の宣言
*       内生変数は variables 命令で宣言
variables
    sup         Supply
    dem         Demand
    pri         Price
;
```

ここではモデルの内生変数、つまり供給量（sup)、需要量（dem)、価格（pri）にあたる変数を宣言している。

```
equations
    e_sup   "FOC for profit maximization"
    e_dem   "FOC for utility maximization"
    e_pri   "Market clearing condition"
;
```

ここでは式を宣言している。式については後に

- 利潤最大化条件（供給関数）に対しては供給量（sup）
- 効用最大化条件（需要関数）に対しては需要量（dem）
- 市場均衡条件に対しては価格（pri）

を対応させることになる。対応関係がわかりやすくなるように、式の名前は「e_ + その式に対応

する変数名」としている。

```
e_sup .. alpha + beta * sup - pri =g= 0;

e_dem .. pri - (gamma - epsilon * dem) =g= 0;

e_pri .. sup - dem =g= 0;

model mcp_sample MCP / e_sup.sup, e_dem.dem, e_pri.pri /;
```

ここではまず式を定義し、その後、モデルを宣言、定義している。式は上から順に均衡条件の(14)〜(16) 式に対応している。ここの部分が MCP タイプのモデルを記述する際のポイントになるので、その記述方法をよく理解する必要がある。

例えば、供給量 s については次の 3 つの式が決定式であった。

$$\alpha + \beta s - p \geq 0$$
$$s \geq 0$$
$$(\alpha + \beta s - p)s = 0$$

GAMS でこのような MCP タイプ問題を記述する際には条件式として直接この 3 つの式を書くということはしない。記述は次のようにおこなう。

- 1 番目の条件式は普通に式として記述する。
- 2 番目の条件式は、一本の式として記述するのではなく、変数に対する下限値を設定するという形で導入する。つまり、`sup.lo = 0` という形で記述する。
- 3 番目の条件式は明示的に式としては記述しない。その代わりにモデルの定義の部分で `e_sup.sup` のように式 `e_sup` に対し、変数 `sup` を対応させることで 3 番目の関係を組み込むことができる。
  - つまり、モデルの定義の部分の `e_sup.sup` という指定は、式 `e_sup` が等号では満たされないなら `sup` は下限値（この場合 0）をとり、`sup` が下限値より大（つまり正）なら、`e_sup` は等号で満たされるという条件を指定するのと同じになるということである。

1 番目の式の書き方についての注

上で述べたように 1 番目の式はそのまま式として記述する。それを $\alpha + \beta s - p \geq 0$ という形で書いているが、これは数学的には $p - (\alpha + \beta s) \leq 0$ と同値である。よって、この式をプログラムで記述する際に

```
e_sup .. alpha + beta * sup - pri =g= 0;
```

ではなく、

```
e_sup .. pri - (alpha + beta * sup) =l= 0;
```

という書き方をしても同じではないかと思うかもしれない。しかし、この書き方をすると GAMS は間違いとみなし、次のようなエラーを出す（実際試してみて欲しい）。

```
**** Lower Bound and =L= illegal
     Sup
```

GAMS でこの MCP タイプのモデルを記述する際には不等号の向きにルールがあり、下限値を設定する変数を対応させる式は $\geq$ （=g=）を使って記述しなければならないことになっている。よって、下側のような「=l=」を使った記述はできない。(3) 式と (8) 式において不等号が左開きになるようにわざわざ書き直したのは、GAMS **のプログラムで記述するときに左開きを使う必要がある**ためである。

以上の説明より、MCP **タイプのモデルでは式と変数の対応関係が重要**であることがわかる。e_sup.sup の場合、e_sup が binding にならないときに 0 になるのは sup という変数であり、sup が正のときに binding にならなければいけないのは e_sup という式である。もし、e_sup.dem となっていれば全く違う条件になってしまうことになる。よって、式と変数を適切に対応させる必要がある。ただし、経済学のモデルの場合には式と変数の対応関係は非常にわかりやすいことが多いので、どう対応させるかで迷うようなケースは少ないと考えられる。

[注]

MCP タイプ以外のモデル、例えば NLP のモデルでは model 命令でモデルを定義するときに式を指定するだけでよく、式に変数を対応させる必要はない。モデルの定義の際に、式に変数を対応させるのは MCP タイプのモデルに特有の要件である。この点については第 4 節で補足説明をおこなう。

それでは続きを考える。

```
sup.lo = 0;
dem.lo = 0;
pri.lo = 0;

sup.l = 10;
dem.l = 10;
pri.l = 10;
```

ここは変数に対する下限値と変数の初期値を設定している部分である。全て、下限値を 0 としている。10 という初期値は適当に選んだ値で、10 であることにたいした根拠はない。

```
solve mcp_sample using mcp;
```

ここでモデル mcp_sample を解いている。第 3 章で説明したように「using mcp」という指定によって MCP ソルバーを利用して解くことになる。実際にここまでのモデル解いてみて欲しい。

```
               S O L V E      S U M M A R Y

     MODEL   mcp_sample
     TYPE    MCP
     SOLVER  PATH                FROM LINE  100

**** SOLVER STATUS     1 Normal Completion
**** MODEL STATUS      1 Optimal

 RESOURCE USAGE, LIMIT          0.000 10000000000.000
 ITERATION COUNT, LIMIT         0    2147483647
 EVALUATION ERRORS              0             0
3 row/cols, 6 non-zeros, 66.67% dense.
```

モデルが正常解けていたら、上のような結果が listing ファイルに出力されるはずである。

```
                        LOWER     LEVEL     UPPER    MARGINAL

---- VAR sup              .        4.000     +INF       .
---- VAR dem              .        4.000     +INF       .
---- VAR pri              .        2.000     +INF       .
```

さらに、変数の解は上のような値になるはずである。均衡価格は 2、均衡供給量、需要量は 4 となっている（同じ結果かどうか確認して欲しい）。

上の solve 命令のすぐ後に $exit 命令があるのでプログラムの実行はここで終了する（$exit 命令はそこでプログラムの実行を終了せよという命令）。この後の部分ではパラメータを変更するシミュレーションを行っている。費用関数のパラメータ α（供給曲線の切片）の値を変更することで、均衡価格、均衡数量がどう変化するかを見るというシミュレーションである。実行する場合に

は、$exit 命令を消せばよい。

以下では、その部分のプログラムの解説をする。

```
set     itr     Index of iteration / itr1*itr21 /;
display itr;
```

ここでは集合 itr を宣言・定義している。これまで集合を定義する際には、スラッシュで囲んだ中にその要素を書くという形で記述してきたが、ここでは「itr1*itr21」という特殊な記法を利用している。この「itr1*itr21」という記法は、「itr1, itr2, itr3,…,itr20, itr21」のように記述するのと同じ効果を持つ。よって、上の定義によって itr は itr1 から itr21 という 21 個の要素を持つことになる。実際に要素がそのように定義されているか出力結果で確認して欲しい。

```
parameter
    alpha_(itr)         Value of alpha
    alpha_i             Initial value of alpha
    alpha_f             Final value of alpha;
```

ここでは alpha の値を変更するシミュレーションのための追加的なパラメータを宣言している。alpha_i は最初の alpha の値を設定するパラメータ、alpha_f は最後の alpha の値を設定するパラメータである。

```
alpha_i = 20;
alpha_f = -20;

alpha_(itr) =
    alpha_i + (ord(itr) - 1) * (alpha_f - alpha_i)/(card(itr) - 1);
display alpha_;
```

ここで上で宣言したパラメータに値を代入している。最初の alpha の値は 20、最後の alpha の値は -20 と置いている。つまり、供給曲線の切片を 20 から -20 の範囲で動かすことになる（これは供給曲線を上から下にシフトさせていくということ）。

その下で各 itr に対する alpha_ の値を設定している。ここでは繰り返しの回数に応じて alpha_i から alpha_f までの長さを等分して、1 回毎に少しずつ変えていくという方法をとる。繰り返しの数が 3 回なら 2 等分、4 回なら 3 等分するという形になるので、「繰り返しの数 − 1」で「alpha_f − alpha_i」の長さを割り、その値を 1 回毎に変えていく。ここで「card(itr)」は集合 itr の要素の数を返す命令である。itr は 21 個の要素を持つので、card(itr) = 21 となる。また、「ord(itr)」というのはその時点での itr の要素が何番目にあたるかを返す命令である。具

体的には、ord("itr1")=1、ord("itr2")=2、ord("itr21")=21 という値をとる。

実際に「alpha_」にどのような値が代入されているか出力結果で確認して欲しい。GAMS においてパラメータを少しずつ変更して何度もモデルを解くというシミュレーションをおこなうときにはこのような集合、パラメータの使い方をすればよい。

```
loop(itr,

    alpha = alpha_(itr);

*       モデルを解く
    solve mcp_sample using mcp;

*       計算結果をパラメータに代入.
    results(itr,"alpha") = alpha + eps;
    results(itr,"price") = pri.l + eps;
    results(itr,"supply") = sup.l + eps;
    results(itr,"demand") = dem.l + eps;
    results(itr,"s-d") = sup.l - dem.l + eps;
    results(itr,"pri.m") = pri.m + eps;
    results(itr,"sup.m") = sup.m + eps;
    results(itr,"dem.m") = dem.m + eps;

);
```

この部分では loop 命令を利用して繰り返しモデルを解き、その計算結果をパラメータに代入している。

```
loop(itr,

    loop 内のコード

);
```

この表現によって集合 itr の要素の回数だけ loop 内のコードが繰り返し実行されることになる。

```
alpha = alpha_(itr);
```

この部分で itr の値に応じて、alpha の値を変更している。

```
results(itr,"alpha") = alpha + eps;
```

ここではパラメータ alpha の値をパラメータ results に代入している。**eps は GAMS においてゼロに近い小さい値を表す特殊な記号である。**eps を足しているのは、出力結果を後に Excel に出力するためである[7)]。

```
*       results パラメータを mcp_sample.gdx ファイルに出力する.
execute_unload "mcp_sample.gdx", results;
```

これはパラメータ results の値を mcp_sampl.gdx ファイルに出力するという命令である。GDX ファイルというのは GAMS 独自のデータ形式であり、GAMS でデータのやりとりをするときにはこの形式でデータを扱うのが標準となっている。例えば、エクセルからデータを読み込むような場合でも

Excel ファイル　→　GDX ファイル　→　GAMS

というように一度 GDX ファイルにしてから読み込む手法をとる。同様に、Excel ファイルにデータを出力するときは

GAMS　→　GDX ファイル　→　Excel ファイル

というようにやはり GDX ファイルを通じてエクスポートが行われる。ここでも後に結果を Excel ファイルに出力をするため、まず GDX ファイルに出力をしている。

```
*       mcp_sample.gdx ファイルの中身を mcp_sample.xlsx に出力する.
execute 'gdxxrw i=mcp_sample.gdx o=mcp_sample.xlsx epsout=0 par=results rng=results!A1 rdim=1 cdim=1';
```

これは GDX ファイルの中身を Excel ファイルにエクスポートする命令である。execute というのは GAMS から外部プログラムを呼び出す命令であり、GDX ファイルから Excel ファイルへの出力には gdxxrw.exe という GAMS に付属の外部プログラムを利用する。上の命令は、「mcp_sample.gdx」内の results というパラメータを「mcp_sample.xlsx」という Excel ファイルの results というシートに A1 セルを起点にして出力せよという意味になる。gdxxrw について詳しくは “GDXXRW” を見て欲しい。

[注]

Excel は Windows だけではなく Mac でも動作するが、**gdxxrw というプログラムは Windows でしか動作しない**ので、以上のプログラムの一番最後の部分（Excel に出力される部分）は Windows 以外では実行されない。

7) 後に計算結果を GDX ファイルに出力し、さらに Excel ファイルに出力する。このとき 0 の値をとる項目が全く出力されないことになってしまう。0 の項目は 0 という値をとるものとして、出力したいので、そのために eps を入れてある。

4.2 シミュレーション結果

前節のプログラムを実行した結果が表 1 である。ここから α（供給曲線の切片）の値を 20 から -20 に変化させたときに、均衡の価格、供給量、需要量、超過供給量がどう変化していくかがわかる。

均衡は第 2 節で見た 3 つのパターンに分かれる。まず、1 番上の青い部分がパターン B にあたる均衡である。このパターンでは、供給曲線が非常に高い位置にあるため、価格は正であるが、供給量、需要量ともゼロになるという均衡が成立している。2 番目の白い部分がパターン A にあたる均衡で、このパターンでは価格も正であるし、供給量、需要量も正である。最後の赤い部分がパターン C にあたる均衡で、均衡価格が 0 となる。価格がゼロになっているのに供給量 $>$ 需要量が成り立つため、この均衡では超過供給が発生する。

A というパターンがいわゆる普通の均衡であるが、MCP の形式でモデルを記述すれば、B や C という供給量、需要量、価格がゼロになってしまうような均衡を一つのケースとして考慮することができるということである。

表 1：計算結果

α	価格	供給量	需要量	超過供給量	pri.m
20	18.0	0.0	0.0	0.0	0.0
18	18.0	0.0	0.0	0.0	0.0
16	16.0	0.0	0.0	0.0	0.0
14	14.0	0.0	0.0	0.0	0.0
12	12.0	0.0	0.0	0.0	0.0
10	10.0	0.0	0.0	0.0	0.0
8	8.4	0.8	0.8	0.0	0.0
6	6.8	1.6	1.6	0.0	0.0
4	5.2	2.4	2.4	0.0	0.0
2	3.6	3.2	3.2	0.0	0.0
0	2.0	4.0	4.0	0.0	0.0
-2	0.4	4.8	4.8	0.0	0.0
-4	0.0	8.0	5.0	3.0	3.0
-6	0.0	12.0	5.0	7.0	7.0
-8	0.0	16.0	5.0	11.0	11.0
-10	0.0	20.0	5.0	15.0	15.0
-12	0.0	24.0	5.0	19.0	19.0
-14	0.0	28.0	5.0	23.0	23.0
-16	0.0	32.0	5.0	27.0	27.0
-18	0.0	36.0	5.0	31.0	31.0
-20	0.0	40.0	5.0	35.0	35.0

$\beta = 0.5$、$\gamma = 10$、$\epsilon = 2$ という設定。

4.3 変数の Marginal 値の意味

第 3 章において、**「MCP モデルにおける変数の marginal 値は、その変数が対応づけられた式の乖離幅を表している」**と説明した。表 1 の pri.m は変数 prim の marginal 値を表している。第 4.1 節で見たように、変数 pri は 式 e_pri に対応づけられている。その式 e_pri は次のように定義されていた。

```
e_pri .. sup - dem =g= 0;
```

従って、pri の marginal 値（pri.m）は次の値を表現することになる。

```
pri.m = sup - dem
```

つまり、超過供給の値となる。実際、pri.m の値は「供給量 − 需要量（= sup − dem）」によって計算された超過供給量の値に等しくなる。sup や dem の marginal 値も同様に解釈できる。

MCP モデルでは「変数の marginal 値＝その変数が対応づけられた式の乖離幅」であり、式が等号で成立するならゼロとなるが、上の例のように等号では成立しない場合にはどれだけ右辺左辺が乖離しているかを表すことになる。

5 補足説明

MCP のモデルではモデルを指定する際に式と変数を対応させる形をとると説明した。しかし、MCP のモデルであっても必ずしも変数を対応させる必要があるとは限らない。以下では、**MCP のモデルにおいてどのようなときに変数を式に対応させる必要があるのか（あるいは、ないのか）**をもう少し詳しく説明する。

5.1 式に変数を対応させない場合

まず、モデルの定義において、式に変数を対応させなかったらどうなるかを確認してみよう。mcp_sample_alt_a.gms ではモデルの定義を次のようにおこなっている。

```
model mcp_sample MCP / e_sup, e_dem, e_pri /;
```

この書き方では式だけを指定し、変数を対応させていない。このようにモデルを定義した場合に GAMS を実行すると、次のようなエラーが生じる。

```
97  *       ---------------------------------------------------------
98  *       モデルを解く
```

```
  99  option mcp = path;
 100  solve mcp_sample using mcp;
****                            $484,256
**** 256  Error(s) in analyzing solve statement.
**** 484  Unmapped MCP equations cannot be =g= =l= or =n=
**** The following MCP errors were detected in model mcp_sample:
**** 484 e_sup unmapped equ =l= =g= or =n= in mcp
**** 484 e_dem unmapped equ =l= =g= or =n= in mcp
**** 484 e_pri unmapped equ =l= =g= or =n= in mcp
```

solve 命令のところで 484 という番号のエラーが生じている。エラー番号$484 のメッセージ（Unmapped MCP equations cannot be =g= =l= or =n=）が示すように、これは式の中で「≥」や「≤」が使われているにもかかわらず、式に対して変数が対応（マップ）されていないというエラーである。このエラーが示すように、MCP **のモデルにおいて式の中で「≥」や「≤」を利用した場合には、必ず式に対して変数を対応させる必要**がある。

5.2 式で不等号を利用しない場合

MCP のモデルの式の中で「≥」や「≤」を利用しているときには式に対して変数を対応させる必要があった。それでは式の中で等号しか使われていなければどうであろうか？

第 4.2 節のシミュレーション結果によれば、デフォールトのパラメータの設定の下で均衡では供給量＝需要量＝ 4、価格＝ 2 となった。供給量、需要量、価格のどれも正であるため、これは均衡条件式が均衡において等号で成立しているということを意味する。均衡状態において結局等号として成立することから、あらかじめ式を全て等号で書き直したのが mcp_sample_alt_b.gms である。これを実行すると再びエラーが生じる。

```
**** Unmatched variable not free or fixed
     sup

**** Unmatched variable not free or fixed
     dem

**** Unmatched variable not free or fixed
     pri

**** Counts do not match
     Single equations in unmatched =E= blocks        3
     Unmatched single free variables                 0
```

「Unmatched variable not free or fixed」とは、式に対応させられていない（マッチされていない）変数が固定されている、あるいは制約が付いているというメッセージである。プログラムでは

sup、dem、pri とも全て 0 以上という制約が付けられている[8)]。MCP のモデルにおいては、このように制約を付けられた変数（下限、上限をつけられた変数）は必ず式に対応させなければならない。それが上のメッセージの意味である。

第 3 章で例に利用した効用最大化問題の式では等号しか含まれていなかった。しかし、その場合でも変数に非負制約を課していたので、やはりモデルの定義において式に変数を対応させる必要があるということである。

5.3 変数への制約を外した場合

式において不等号を利用していないとしても、変数に制約が付けられている場合にはやはり式に変数を対応させる必要があることを見た。それでは変数に対する制約をはずすとどうなるのか？

mcp_sample_alt_c.gms は mcp_sample_alt_b.gms から変数に対する下限の設定を除去したプログラムである。これを実行するとエラーも出ず、mcp_sample.gms と全く同じように解ける。つまり、MCP のモデルであっても

- 式の定義において等号しか使われていない
- 変数に対する制約がない

という設定であれば、モデルの定義の際に式に変数を対応させる必要はないということである。このような設定が可能なモデルでは単に式を指定するだけでよい。

それでは実際にそのようなケースは多いだろうか？ 1 つ目の設定については満たされることが多い、つまり式の定義において等号しか利用しないで済むケースは多い。しかし、2 つ目の設定にはできないケースが多い。というのは経済学のモデルでは変数に非負制約を課す必要があるケースが多いからである。ただ、場合によっては均衡において変数が必ず正の値をとることがわかっているケースもある。しかし、その場合であっても GAMS では非負制約を設定しなければ上手く解けないケースが多い。それは GAMS の計算過程において解に辿り着くまでに 0 や負の値をとることでエラーが生じてしまうことがよくあるからである[9)]。仮に解が正の値になることはわかっていたとしても、計算過程で変数にゼロの値が入ってしまい、その結果、division by zero などのような execution error が生じてしまうことがよくある。

以上のように、場合によっては、MCP のモデルであってもモデルの定義の際に式のみを指定すればよいときもあるが、経済学のモデル、CGE モデルでは変数に非負制約を課す必要があることが多いため式の指定と同時に変数を式に対応させる必要がある。

6 終わりに

GAMS で CGE モデルを扱う際には MCP というタイプの問題として扱うことが多い。この章ではその MCP という問題の解き方について説明した。第 5 章で実際に CGE モデルを解くが、そ

8) sup.lo=0 というような形で下限を設定した。

9) mcp_sample_alt_c.gms は変数に非負制約を課していないが特に問題なく解けている。このように非負制約を課していなくても正常に解けてしまうケースもあるが、そうならないケースも多い。

こではこの章で説明した MCP という形式の問題として解くことになる。ただし、CGE モデルを MCP というタイプの問題ではなく、NLP として扱うアプローチもある。これについては、第 11 章で説明する。

参考文献

Mas-Colell, A., M. D. Whinston, and J. R. Green (1995). *Microeconomic Theory*. New York: Oxford University Press (cit. on p. 4).

Simon, C. P. and L. Blume (1994). *Mathematics for Economists*. New York: W. W. Norton Company (cit. on p. 4).

7 問題

問題 1：リカードモデル

国際貿易の理論モデルの一つとして「リカードモデル」と呼ばれるモデルがある。以下で単純なリカードモデルを用いて MCP タイプのシミュレーションをする[10)]。

モデルの説明

- 財（部門）は 2 財（$i = 1, 2$）
- 生産要素は労働のみ（1 生産要素）
- 各財の生産は次の生産関数に従っておこなわれる

$$q_i = l_i / a_i$$

  ただし、q_i は部門 i の生産量、l_i は部門 y の労働投入量、a_i は 1 単位の生産をおこなうのに必要となる労働の量を表すパラメータ（労働投入係数）である。
- この国の労働賦存量を $\bar{l}$ とする。
- 生産要素が労働だけで、その限界生産性（上の記号では $1/a_i$）が常に一定という点がリカードモデルの大きな特徴である。

各部門は利潤最大化行動に基づいて労働投入量を決定すると仮定する。各部門の利潤 π_i は

$$\pi_i = p_i \frac{l_i}{a_i} - w l_i$$

と表現できる。ただし、p_i は財 i の価格、w は賃金率であり、第 1 項目が収入、第 2 項目が費用を表している。

10) リカードモデルは非常に有名なモデルで、理論分析ではよく利用されるが、生産要素を労働のみとするという非常に特殊（で、非現実的）な仮定を用いるモデルであるので、CGE 分析用のモデルとしては利用されることはない。ここではあくまで計算問題用の例として用いているだけである。

各部門はプライステイカーとして行動する。また、各部門の労働投入量は非負の値をとる。このときの利潤最大化のための Kuhn-Tucker 条件は次式で与えられる。

$$\frac{\partial \pi_i}{\partial l_i} = \frac{p_i}{a_i} - w \leq 0 \qquad l_i \geq 0 \qquad \left[w - \frac{p_i}{a_i}\right] l_i = 0$$

これは MCP の形式になっている。

さらに、労働市場の均衡条件は次式となる。

$$\bar{l} - \sum_i l_i = 0$$

このモデルでは労働供給も労働需要も必ず正となり、その結果必ず賃金も正となるので、労働市場の均衡条件は「労働供給＝労働需要」のように表現している。

本来のリカードモデルでは、ここに財の需要側（財市場）を加え、さらに複数の国を想定し、貿易をおこなうことの効果を考えるのであるが、ここでは財市場は考慮せず、財の価格が外生的に与えられるとする。そして、財 1 をニュメレールと仮定するすると、以上のモデルは次のように表現できる。

$$w - \frac{p_i}{a_i} \geq 0 \perp l_i \geq 0 \qquad i = 1, 2$$
$$\bar{l} - \sum_i l_i = 0$$

ただし、財 1 がニュメレールであることから、$p_1 = 1$ である。

このモデルを GAMS において MCP の形式で記述し、財 2 の価格 p_2 を 0.5 から 4 まで変更していくシミュレーションをおこない、それによって各財の生産量がどう変化するかを説明しなさい。また、価格がどのような値のときに不完全特化（二つの財が同時に生産される状況）が生じるか求めなさい。ただし、パラメータ、労働賦存量は次のように設定すること。

- $a_1 = 1$
- $a_2 = 2$
- $\bar{l} = 100$

図 5 は上のシミュレーションの結果をもとに財 2 の供給曲線を Excel で描いたグラフである。横軸の q(2) が財 2 の供給量、縦軸の p(2) が財 2 の価格（相対価格）である。

8 履歴

- 2023-04-28: 誤植の修正。
- 2022-01-12: 説明の追加・修正。
- 2018-07-20: 説明の追加・修正。
- 2017-03-15: 説明の修正。

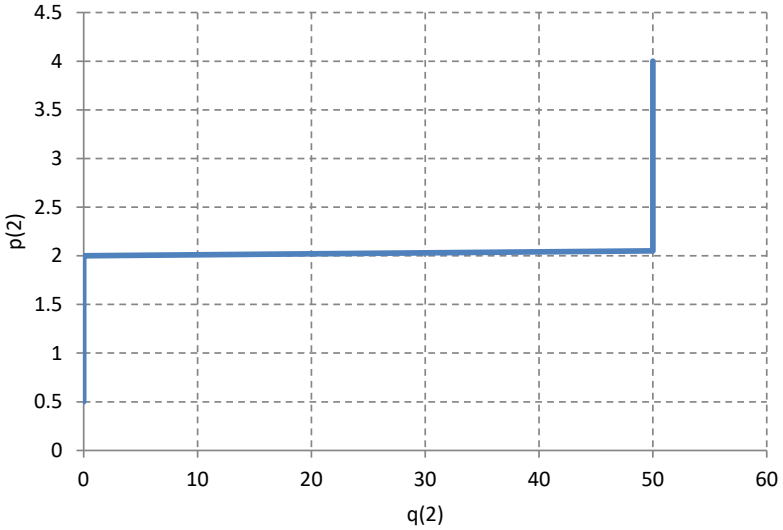

図 5：リカードモデルにおける供給曲線