Bewley-Aiyagari モデルの解法

山田 知明
立正大学経済学部

2009年1月13日@神戸大学

はじめに

- Bewley-Aiyagari **モデルの解法**
  - **一部門最適成長モデルの拡張**
  - **モデルの特徴は**
    1. **固有リスク** (Idiosyncratic Riks)
    2. **推移関数** (Law of Motion) **と資産分布**
- **様々な数値計算ツールが必要**
  - **最適化** (Optimization)
  - **非線型方程式の解法** (Nonlinear Equation)
  - **近似法・内挿法** (Approximation, Interpolation)
  - **積分** (Integration)

はじめに

- **統計パッケージのようにコマンド一つでは出来ない**
  - **数値計算を「使える」ようにする**

[R]eading a book on computation techniques without actually using the compter is as foolish as reading a cook-book without ever entering the kitchen.(Marimon and Scott (1999), p,2)

- **講義ノート& プログラム**
  - http://homepage2.nifty.com/~tyamada/index.html
  - Matlab & Fortran Code (Password:hit-u_lecture)

テキスト選択

- Judd (1998)：**数値解析に関する一番基本的なテキスト**
- Miranda and Fackler (2002)：Matlab **との親和性が高い初心者向けテキスト**
- Adda and Cooper (2003)：**動学的マクロ経済学全般に関するテキストで、理論モデル、推計と数値計算を一つの本の中にコンパクトにまとめている**
- Marimon and Scott (1999)：**各トピックについて、その分野のスペシャリスト達が書いている**
- Press, Flannery, Teukolsky and Vetterling (1992)：**ニュートン法等、数値計算に必要なアルゴリズムの実際的な解説とコードを書いてくれているので、文字通り、レシピ感覚で使える**

ソフトウェア選択

- 実際に手を動かして数値計算を行う前に、ソフトウェアの選択を避けて通る事は出来ない
- ソフトウェアは、大雑把に分けて、次の3種類の選択肢がある

1. C言語やFortranのように数値計算専用ではない言語
2. MatlabやGaussのような数値計算(科学技術演算)用のソフトウェア
3. MapleやMathematicaのような数式処理ソフト

ソフトウェア選択

Fortran[FORTRAN]

- 数値計算に関しては一番優れた環境が整う
  - ただし、Matlab 等に比べると手間がかかる
  - I/O は極めて貧弱なので、グラフなどは別のツールが必要

1. Intel Visual Fortran (+IMSL)：統合環境。数値計算パッケージである IMSL がついている
2. NAG Fortran Builder (NAG)：学習用もあるのでお手ごろ感がある
3. GCC/G95：GNU プロジェクトによるコンパイラ
4. IMSL/NAG：有料のパッケージ(サブルーチンの集まり)
5. LAPACK/BLAS/MINPACK/HOMPACK：フリーのサブルーチン達(netlib)

ソフトウェア選択

Matlab/Scilab/Octave

- Fortran や C 言語と比べると敷居が低く、入門者用のソフトウェア達
- 扱いやすいが計算のスピード面では問題がある
- 大きなモデルを解きたい場合、プログラミングの手間がかかっても Fortran や C 言語で書いたほうが時間の節約に
- グラフィック面では、Matlab の方が優れている
- Matlab は正規版を購入すると高額だが、Matlab と似たような機能を持つフリーのソフトウェアも開発されている

1. Scilab: http://www.scilab.org/
2. Octave: http://www.gnu.org/software/octave/

MatlabからFortranへの移行

- 命令の対応関係を覚えれば、一週間もあれば移行できる
- コンパイル形式とインタープリタ形式では実行速度はかなり違う
- Fortranでは、「コンパイル」をして「リンク」をするという作業が必要になる
- が、統合環境だとワンクリックで済むので、あまり意識する必要はない
- コンパイルオプションを使いこなせるようになると、計算速度が急激に上昇する(20分が5分になったりする)

Ramseyから Bewleyへ

- 一部門最適成長モデルにおいて代表的個人を考えるのは、保険市場の完備性を仮定しているため
  - 社会計画者が解く配分問題と市場において達成された配分は一致
- 経済活動をしている主体(家計や企業)は固有リスク (Idiosyncratic Risk) に直面しているかもしれない
  - 不完備市場の一般均衡理論
  - Bewley モデル (Bewley,1977)

Bewley-Aiyagari モデル：目的関数

- 基本的なセットアップは、Aiyagari (1994)
  - 無限期間の生産経済
  - 経済主体は測度 1 で $[0, 1]$ 区間の連続体上に無限人存在
  - 0 期には同質的
- 各家計の目的関数

$$\max E_0 \sum_{t=0}^{\infty} \beta^t \frac{c_t^{1-\gamma}}{1-\gamma}$$

予算制約

- 資産 a_t と労働所得 y_t を所与
- 予算制約

$$c_t + a_{t+1} \leq (1 + r_t)\, a_t + y_t$$

  - 利子率：r_t
  - 借入制約：$a_t \geq \varphi$

固有リスク(Idiosyncratic Risk)

- 勤労所得：$y_t = w_t e_t$
  - 労働保有量(Labor Endowment)：$\{e^1, \ldots, e^J\}$
  - 賃金：w_t
- 家計は無数に存在
  - 推移確率(マルコフ環)：$\pi(e^j | e^i) > 0$
  - 人口割合の推移も表している
  - 一意的な定常分布：π^*
- 自然な借入制約(Natural Borrowing Limit)

$$a_t \geq -\frac{we^1}{r} = \varphi$$

所得変動問題(Income Fluctuation Problem)

- **家計の最適化問題**

$$v(a,e) = \max\left\{\frac{c^{1-\gamma}}{1-\gamma} + \beta\sum_{e'}\pi(e'|e)\,v(a',e')\right\},$$

$$\text{s.t.}\ \ c + a' \leq (1+r)\,a + we,\ \ a' \geq 0.$$

  - **政策関数**：$g(a,e)$
- FOC：

$$u'(c) \geq \beta(1+r)\sum\pi(e'|e)\,u'(c').$$

推移確率と分布関数

- $x = (a, e)$
- 現在の状態が$x \in \mathcal{X}$である家計が次期に$B \in \mathcal{B}(\mathcal{X})$が実現する確率を推移関数$Q : \mathcal{X} \times \mathcal{B}(\mathcal{X}) \to [0, 1]$と書く

$$Q(x, B) = \pi\left(\left\{e' \in \mathcal{E} \middle| \left(g(a, e), e'\right) \in B\right\} \middle| e\right),$$

- $\Psi(B)$：$B \in \mathcal{B}(\mathcal{X})$上で定義された確率測度
  - Ψは家計の状態が(a, e)となる割合
  - あらゆる$B \in \mathcal{B}(\mathcal{X})$について、

$$\Psi_{t+1}(B) = \int_{\mathcal{X}} Q(x, B)\, d\Psi_t$$

マクロ経済(集計経済)

- 総資本供給
$$K = \int g\left(a, e\right) d\Psi$$
- 総労働供給：
$$N = \sum e\pi^{*}\left(e\right)$$
  - $\pi^{*}\left(e\right)$ は定常分布での e の実現確率
- 企業の生産関数
$$Y = K^{\theta} N^{1-\theta}$$

競争均衡の定義

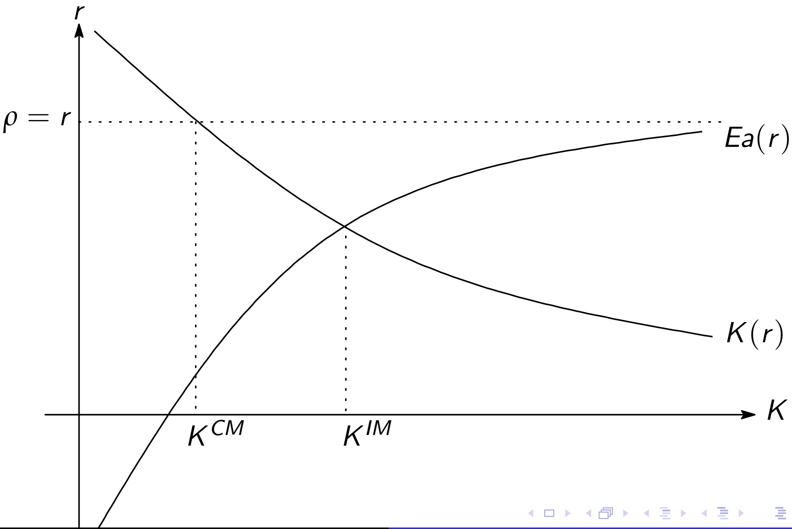
- 定常再帰的競争均衡とは、以下を満たす価値関数 v、政策関数 g、利子率 r、賃金 w 及び分布関数 Ψ^* である。

1. 利子率 r 及び賃金 w を所与としたとき、v は家計のベルマン方程式を満たし、$g(\cdot)$ はそれに伴なう政策関数になる。
2. 企業は利潤を最大化している。すなわち、

$$r = \theta K^{\theta-1} N^{1-\theta} - \delta, \; w = (1-\theta) K^{\theta} N^{-\theta}.$$

3. 財、労働及び資本市場は均衡している。
4. 資産分布は時間を通じて不変である。

競争均衡の存在



Bewley-Aiyagari モデルの数値計算

1. パラメターを設定する (Calibration)
2. 利子率 r を任意に与える。総労働供給 N は決まっているため、企業の一階条件から (w, K) を逆算
3. 要素価格 (r, w) が一つ与えられたので、家計の最適化問題を解く
4. 各家計の意思決定関数が得られたので、推移関数に基づいて家計分布の推移を計算し、定常分布 Ψ_r^* を導出
5. 定常分布 Ψ_r^* を使って家計行動を積分して、総資本供給 $Ea(r)$ を計算
6. 総資本需要 $K(r)$ と総資本供給 $Ea(r)$ が一致する r を見つけるまで、$2-5$ を繰り返す。

価値関数繰り返し法(Value Function Iteration; VFI)

VFI を選択する理由は、

- 価値関数を繰り返し計算するときに縮小写像の性質 (Contraction Mapping Property) があるので、
  - 安全 (Safe)
  - 信頼性 (Reliable)
  - 収束 (Convergence Property)
  - DP の形にすれば、何にでも応用可能

価値関数繰り返し法(2)

動的最適化問題を解く際に用いられる方法は、大きく分けて二種類の方法

1. **状態変数や操作変数を離散化して解く方法**(Discretization Approach)
2. **解きたい関数(政策関数)を有限個のパラメータで表現する方法**(Parametric Approximation Approach)

- **モデルを理論的に解く場合、連続性を要請**
- **連続区間をコンピュータで扱うのは不可能**
- **何らかの方法で有限個に離散化する必要がある**

Discretized Approach

状態空間と操作変数が共に離散的な場合の動的計画法の解法

- 非常に遅くコンピュータのメモリも大量に消費
- 安定しており、強い非線形性があるモデルでもOK

1. 状態変数の上限$\bar{a}$及び下限$\underline{a}$を設定
2. 状態変数aおよび操作変数a'をそれぞれ離散n個に区切る ($\{a_i, a'_j\}_{i,j=1}^n$)
   - 離散個に区切った状態変数の各点の事を、グリッド(Grid)、ノード(Node)あるいはメッシュ(Mesh)などと呼ぶ
3. 資本の状態は離散個に区切った値しかとらない

ベルマン方程式は、

$$v(a_i, e) = \max\{u((1+r)a_i + we - a_j) + \beta v(a_j)\}.$$

Discretized Approach：アルゴリズム

1. **グリッド生成：状態空間及び操作変数を有限個のグリッドに区切る**
    - **この有限個のグリッド上における価値関数 v^i の初期値を推測** (Initial Guess)
    - **価値関数は収束するので、始めは 0 からスタートしても問題ない**
2. **収束の基準：収束の基準になるパラメーター ε を設定**
3. **効用関数：効用関数 $u((1+r)a_i + we - a_j)$ を各グリッド上で評価する**
    - $n \times n$ **の行列になる**

Discretized Approach：アルゴリズム(続き)

4. **価値関数を繰り返し計算：**
   1. **各 a_i について**

      $$v\left(a_i, e\right) = \left\{u((1+r)a_i + we - a_j) + \beta \hat{v}(a_j'),\right.$$

      を計算。右辺第2項の $v(a_j')$ は、初期値では上で与えたもの(例えばゼロ行列)を使う
   2. **各 a_i について、$v\left(a_i\right)$ を最大にする a_j を探す(最適政策)**
   3. **古い価値関数 $\hat{v}$ と新しい価値関数 v が十分に近ければストップ**
   4. **そうでなければ、$v\left(a_i\right)$ を $\hat{v}\left(a_j\right)$ に代入して、同じ計算を繰り返す**
   5. **繰り返していくと、縮小写像の性質があるため、いずれ収束**

[Matlab Code 1]

Parametric Approximation Approach

政策関数を多項式などによってパラメトリックに表現できれば、ある状態変数 a **の下での** a' **を、任意の** $a \in [\underline{a}, \overline{a}]$ **で計算できる**

- **状態空間の評価点**(Evaluation Point)**を離散個** $\{a_i\}$ **だけ選んできて価値関数及び政策関数** $\{v(a_i), g(a_i)\}$ **を計算**
- **評価点以外の個所については、何らかの方法で近似**
    - **線形近似**
    - **多項式近似**(Chebyshev etc.)
    - **スプライン近似**(Cubic-Spline, Shape-Preserving Spline)
    - etc.

PAA：アルゴリズム

Johnson et al. (1993) や Judd and Solnick (1994) を参照

1. グリッド生成：状態空間を有限個のグリッドに区切る
   - グリッドは等分である必要はなく、流動性制約に直面しやすいゼロ近辺を多めにする方が精度が高くなる
   - 有限個のグリッド上における価値関数の値 $\hat{v}(a'_j)$ の初期値を推測
2. 収束の基準：収束の基準になるパラメター ε を設定
3. 近似・評価：近似点 a_i 上にない価値関数の値については、線形近似や多項式近似、スプライン補間などを使って近似する必要があるため、その係数を計算
   - $\hat{v}(a, \mathbf{b})$ をパラメター $\mathbf{b}$ を使って近似した時の、a 上での価値関数の値とする

PAA：アルゴリズム(続き)

4. 最適化：各a_iについて、次の式を計算

$$v\left(a_i\right)=\max\{u\left(\left(1+r\right)a_i+y-a'\right)+\beta\hat{v}\left(a';\mathbf{b}\right)\},$$

- 最適解を得るためには、ニュートン法などの手法が必要
- ここをどう工夫するかで計算精度やスピード、安定性が変わってくる
- このステップで新しい価値関数$\{v\left(a_i\right)\}$及び政策関数$\{g\left(a_i\right)\}$を得る

5. 新しいデータ$\{v\}$を使って、価値関数を求めるためにステップ3に戻る

- ステップ3と4を繰り返していくことによって価値関数は収束

[Matlab Code 2]

オイラー方程式を使ったアプローチ

最適化モデルの一階条件であるオイラー方程式を利用

- **代表的なものは、政策関数を多項式などで近似して残差をする射影法**(Projection Method)
- **射影法は**2**種類に分かれる**
  1. **スペクトラル法**(Spectoral Method)：Judd (1992)
  2. **有限要素法**(Finite Element Method; FEM)：McGrattan (1996,1999)

オイラー方程式を使ったアプローチ(続き)

- 基本的なアイディアは、オイラー方程式の誤差を最小にする意思決定関数を探す事
- オイラー方程式は、

$$u'(h(a,e)) \geq (1+r)\beta u'(h(a',e')),$$

  - 先行研究にならって貯蓄関数 $a' = g(a,e)$ ではなく消費関数 $c = h(a,e)$ を政策関数としている

スペクトラル法(1)

- 我々が知りたいのは消費関数である $h(a)$
- 状態空間 $a \in [\underline{a}, \overline{a}]$ 上のスムーズな関数 $h(a)$ を何らかのパラメター $\{\alpha_i\}$ で近似

$$\hat{h}(a; \alpha_i) = \sum_{i=0}^{n} \alpha_i \phi_i(a)$$

  - α は近似のパラメターである多項式の係数
  - $\phi_i(a)$ は基底 (Basis)

スペクトラル法(2)

- 最終的な目標はオイラー方程式の残差関数(Residual Function)をゼロにすること

$$R(a;\alpha) = \beta(1+r)\frac{u'((1+r)a + y - \hat{h}(a,\alpha))}{u'(\hat{h}(a,\alpha))} - 1,$$

  - 残差関数は状態空間$[\underline{a}, \overline{a}]$上で定義
  - 連続空間上のあらゆる点で残差がゼロになるのが理論的には正しい
  - 数値計算ではそのような表現は出来ない
  - 「最小にする」事の数値計算上の定義が必要になる

スペクトラル法(3)

- コロケーション法：評価点$\{a_i\}$を任意に導出して、

$$R(a_i;\alpha) = 0, \quad i = 1, \ldots n,$$

  - 有限個であるため計算が容易
  - 評価点には関数近似のためのグリッドを使えばよい
- ガラーキン法：残差に重み(Weight)$\Psi(a)$を付けて

$$\int_{\underline{a}}^{\overline{a}} \Psi(a) R(a;\alpha) dk = 0,$$

  - 重みには、多項式の基底などを用いる
  - 数値計算において積分をそのまま計算する事は出来ない
  - 積分をさらに何らかの方法で近似をする必要がある

Endogenous Gridpoint Method

- Carroll (2006,EL)：安定性& 計算速度
  - 一番のポイントは、次期の資本 a' の方を有限個に区切る点
  - 最適化は必要ない
  - 近似法は必要
- 家計問題を考える

$$
\begin{aligned}
v(a) &= \max\{u(c) + \beta v(a')\}, \\
\text{s.t. } c + a' &= (1+r)a + y
\end{aligned}
$$

Endogenous Gridpoint Method(続き)

- 価値関数の右辺第2項を次のように定義

$$\Omega(a') = \beta v(a'),$$

  - 現金保有高 $x \equiv (1+r)a + w$ 上の消費関数を $\tilde{h}(x)$ とする
- 包絡線定理から

$$\begin{aligned} c^{-\gamma} &= \Omega'(a') = \beta(1+r)(c')^{-\gamma}, \\ &= \beta(1+r)\tilde{h}(x')^{-\gamma}. \end{aligned}$$

  - 消費関数 $\tilde{h}(x)$ の初期値を(例えば線形ルールで)与えれば、各資産水準 $\{a^1, \ldots, a^n\}$ 上で $\Omega'(a')$ を容易に計算出来

Endogenous Gridpoint Method (続き)

- CRRA 型の限界効用関数は逆関数を計算する事が可能
- $c^i = [\Omega'(a^i)]^{-\frac{1}{\gamma}}$ が各グリッド a^i 上で計算出来る
- 現金保有高は $x_t^i = c_t^i + a^i$ で求められるので、現金保有高上の新しい政策関数 $\tilde{h}(x)$ を導出する事が可能
- 繰り返していき、収束したものがオイラー方程式を満たす政策関数
    - Carroll (2006) が使用した Matlab(&Mathematica) コードは Carroll の HP に置いてある

分布関数の計算(1)：シミュレーションベース

- Monte Carlo シミュレーションを使って計算
  - 政策関数 $g(a, e)$ は既知なので、$\{a_t^i\}_{t=1}^T$ をシミュレーション
  - 例えば、5000 人分の第 i 家計について、11000 期間 ($T = 11000$) 計算する
  - $a_{t+1}^i = g(a_t^i, e_t^i)$、($t = 1, \ldots, 11000$)
  - 5000 人分を集計して平均値を計算すれば、総資本になる
- 特徴
  - 統計データと同じように扱えるので、平均だけでなく分散やジニ係数も簡単に計算
  - シミュレーション誤差が大きい

分布関数の計算(2)：前向きに解く

- 分布関数を離散的に近似：グリッドを設定する
- 分布関数 $\psi_0\left(a^i, e\right)$ の初期値を設定
  - 分布は収束することが知られているため、一様分布などからスタートしてOK
- 現時点で (a, e) である家計が次期に $\hat{a}'$ だけ貯蓄
  - しかし、必ずしも $\hat{a}'$ がグリッド上にあるとは限らない
  - 定数 ω を次のように設定
  - $[a_\ell, a_h]$ は $\hat{a}'$ を挟むグリッド値

$$\omega = \frac{\hat{a}' - a_\ell}{a_h - a_\ell}, \ \hat{a}' \in [a_\ell, a_h]$$

- (a, e) である家計は次期に a_ℓ と a_h に振り分けられる

$$\begin{cases} \pi(e'|e)(1-\omega)\psi_0\left(a, e\right) \\ \pi(e'|e)\omega\psi_0\left(a, e\right) \end{cases}$$

さらに進んだ話

- **マクロショック**
  - Krusell and Smith (1998,JPE)
- **世代重複モデル**
  - Storesletten, Telmer and Yaron (2004,JME)
- **構造推計**
  - Gourinchas and Parker (2002)
  - Fernández-Villaverde and Rubio-Ramírez (2004,2006,2007)