Fictitious play

辻 裕一郎

2013/6/28

はじめに

- Fictitious play の説明
- コードの説明
- まとめ

Fictitious play の説明

- 2 人プレイヤー、戦略が 2 個の標準形ゲームを考える。

  各プレイヤーを 0、1 とし、戦略も 0、1 とする。

  このゲームを何回も繰り返しプレーするものとする。

- 各 t 期において、プレイヤー $i(i = 0, 1)$ は信念 $x_i(t)$ を持っており、「プレイヤー $j(\neq i)$ は確率 $x_i(t)$ で戦略 1 を、$1 - x_i(t)$ で戦略 0 をとる」と考えている。

  各プレイヤーはこの信念に基づいて期待利得を計算し、期待利得が最大になるように毎期行動するとする。この時の最適反応を $a_i(t)$ のように表記する。

Fictitious play の説明

- プレイヤーの信念は以下のように形成されるとする。

$$x_i(t+1) = \frac{x_i(0) + a_j(0) + a_j(1) + ... + a_j(t)}{t+2}$$

このとき、$x_0(t)$ は、

$$x_0(t+1) = x_0(t) + \frac{1}{t+2}(a_1(t) - x_0(t))$$

と再帰的に書くことができる。

これを使い、Python で信念の形成のされ方をシュミレーションしてみる。

コードの説明

- Matching pennie game のコード

```
import matplotlib.pyplot as plt
from random import uniform

game_length = 2000 #ゲームの長さを指定
current_belief0 = uniform(0,1) #最初の信念の指定
current_belief1 = uniform(0,1)

def subplots(): #軸、目盛りを設定
    fig, ax = plt.subplots()
    ax.set_yticks([0, 0.25, 0.5, 0.75, 1])
    ax.set_title('Fititious play')
    return (fig, ax)
fig, ax = subplots()
```

コードの説明

```
 belief0 = [current_belief0] #信念のリストを作成
 belief1 = [current_belief1]

for i in range(game_length):
    if current_belief0 > 0.5: #プレイヤー0の行動を指定
        player0_play = 1
    else:
        player0_play = 0

    if current_belief1 > 0.5: #プレイヤー1の行動を指定
        player1_play = 0
    else:
        player1_play = 1
```

コードの説明

```
#プレイヤーの信念の変更、リストへ追加
    current_belief0
      = (current_belief0)
       + ((player1_play - current_belief0)/(i + 2))
    current_belief1
      = (current_belief1)
       + ((player0_play - current_belief1)/(i + 2))
    belief0.append(current_belief0)
    belief1.append(current_belief1)

#信念のリストをプロットする
 ax.plot(belief0, label = 'x0(t)')
 ax.plot(belief1, label = 'x1(t)')
 ax.legend()
 plt.show()
```

図 1

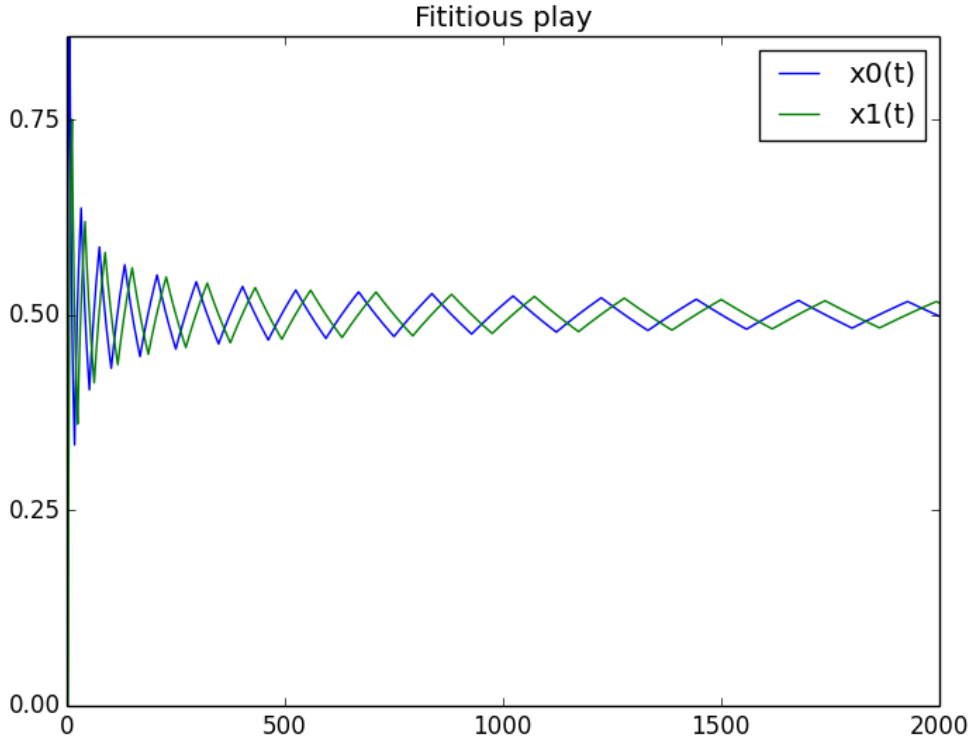

Figure : Matching pennie game

まとめ

- 実際にプログラムを書いて信念の動きをプロットすることで、matching pennie game ではお互いの信念が 0.5 付近に収束し、coordination game では最初の信念 $x_i(0)$ によってお互いの信念が 0 か 1 に収束するということがよく分かった。
- 自分のプログラムの一番の欠点は、汎用性が全くないことである。利得行列から最適反応が変わる点を計算しなければならない、また戦略が 3 個以上になっても対応できない。numpy による計算にもう少しなれる必要があるな、と痛切に感じた。