インタラクティブ多腕バンディットゲーム

二つのトレードオフと選択

Interactive Restless Multi-armed Bandit Game

Choices under two tarde-offs

吉田 俊介
Yoshida Shunsuke
北里大学大学院理学研究科
Graduate School of Science, Kitasato University
sp10144h@st.kitasato-u.ac.jp

守 真太郎
Mori Shintaro
北里大学・理学部・物理学科
Department of Physics, Kitasato University
mori@sci.kitasato-u.ac.jp, http://sharaku.sci.kitasato-u.ac.jp/mori

キーワード: social learning, rational herding, experiment, restless multi-armed bandit, interactive game

概要

多腕バンディット (restless multi-armed bandit) のインタラクティブゲームを用いてヒトの社会的学習について実験を行った。Rendell らの社会的学習戦略トーナメントに基づき、ゲームではバンディットのレバーを引く (Exploit)、新たなレバーを探す (Innovate) に加え、他者が引いたレバーの情報を獲得する (Observe) ことが可能である。被験者は、多数の単純な戦略で選択するエージェントプログラムと対戦し (インタラクティブゲーム)、ランキング上位に入ることでリターンを獲得する。エージェントの戦略は、レバー情報を探すときに Observe を用いる確率 p_{obs} と、Exploit を行うときのレバーに対する最低条件（閾値）を表す c の 2 個のパラメータのみで指定される。被験者数 I は 18 名，環境の変化の確率 p_c、新たなレバーを探すときの検索の範囲 k について $(p_c, k) \in \{A:(0.1,1), B:(0.1,10), C:(0.2,1), D:(0.2,10)\}$ の計 4 パターンで実験を行った。エージェントおよび被験者が情報獲得で Observe を用いた比率 r_{obs}, レバーを引いたときのそのレバーのコイン枚数の平均値 c_{avg} を評価し、獲得コイン枚数との相関を調べた。ケース B,D の場合、Innovate と Observe のトレードオフの状況となっていることが確認された。

1. はじめに

現在の情報を利用してリターンを得る Exploit と、探索することにより、より有用な情報を獲得する Explore。Explore によって、よりよい情報を得ることできれば、Exploit をしなかったことによる機会損失を挽回することができる。しかし、Explore でよりよい情報が獲得できると保証されているわけではない。このトレードオフはよく知られた問題であり、最適な選択のアルゴリズムについて、さまざまな状況での厳密解や近似解などが議論されてきた [?]。例えば、二つのレバーがあるスロットマシンがあり、それぞれのレバーが異なる未知の確率でコインを 1 枚出すとする。何度もレバーを引くときの獲得コイン枚数の最大化の問題もそのひとつである。こうした複数のレバーを持つスロットマシンを一般に多腕バンディット (multi-armed bandit) と呼び、レバーの選択のアルゴリズムを Bandit アルゴリズムと呼ぶ [?]。最初は両方のレバーを同じ割合で引いて、Explore と Exploit を同時に行い、出たコインの枚数で徐々にコインが出る確率が高いレバーの割合を高めて Explore の比率を下げ、最終的にコインの出る確率が高いレバーを選択する。このアルゴリズムは、コインの出る確率の大小関係が明確になるまで両方のレバーを引く A/B アルゴリズムに比べ、Explore の比率が低いため機会損失が少なく、また、大小関係の判定の検出力も高いことが知られている。 で

図 1 多数のスロットマシン@ラスヴェガスのカジノ (Wikipedia より)

は、スロットマシンのレバーの数が多数であり、かつコインの出方が時間とともに変化する、より複雑な状況で

のどうだろうか？こうした時間的に変化するスロットマシンを restless multi-armed bandit と呼び、選択を解析的に最適化することは困難であることが知られている [?]。この複雑な状況下で、もうひとつのトレードオフ：コピー（社会的学習）とトライ＆エラーのトレードオフでの最適な選択のアルゴリズムを解明するためのエージェントプログラムのトーナメントが行われた [?]。

社会的学習とは、動物やヒトなどの生物集団における情報収集の方法の一つで、他の個体の振る舞いの観察や他の個体との相互作用から情報を獲得する方法である [?, ?]。個体がトライ＆エラーで情報を獲得する場合、獲得した情報は確かでも、獲得コストは一般に高い。さらに、有用性の高い情報の獲得に限定すれば、そのコストはさらに高くなる。社会的学習の場合、基本的にコストは低い。しかし、獲得できる情報は、他の個体経由の情報であるため、多少の伝達エラーや情報の古さといった問題もある。このように、トライ＆エラーと社会的学習の選択はコストと精度のトレードオフとなっている。

では、どのように選択するのが適応的なのか？過去の研究から、いつコピーするのか (when strategy)、誰をコピーするのか (who strategy)、について明らかになってきた [?, ?]。それら以下にをまとめた。

When Strategy:いつコピーするか

- 確立した振る舞いが非生産的であるとき
- トライ＆エラーのコストが高いとき
- 不確かなとき
- 満足できないとき
- 環境が比較的安定しているとき

Who Strategy:誰の情報をコピーするか

- 多数派に属する個体
- パフォーマンスのよい個体
- コピーのパフォーマンスがよい個体
- 血縁関係
- 親しい個体
- 古い個体

これらの戦略はキタノトヨミ [?]、ドブネズミ [?] などの社会的学習を用いた実験などで検証されつつある。

一方、[?] でのトーナメントは、100 本のレバーを持ち、レバーを引いたときに獲得できるコインの枚数が指数分布の自乗に従い、ゲームの進行とともに、毎ターン確率 p_c でコインの枚数が変化する非定常多腕バンディット (restless multi-armed bandit) を用いてエージェントプログラム同士の対戦形式で行われた。エージェントプログラムは、各ターンで、記憶にあるレバーを引く (Exlploit) か、新しいレバーを探 (Innovate) してその情報を得るか、他のエージェントが前ターンで Exploit したレバーの情報を獲得 (Observe) するかの、3 種類のアクションのどれを行うかを指示するものである。ここでレバー情報とは、1 から 100 までのレバー番号と、そのレバーを引いたときに獲得できると考えられるコインの枚数である。エージェントは自分の選択とその結果のみを記憶し、また、Exploit できるレバーは Innovate か Observe で情報を得たレバーのみである。Innovate では自分が情報を持たないレバーから 1 本ランダムに選び、その情報を得ることができる。その情報は、次のターンでは、確率 p_c で変化する。Observe では、前ターンで Exploit されたレバーから、ランダムに n_{obs} 本選び、それらのレバー番号と獲得コイン枚数のレバー情報を得る。その際、レバー番号や獲得コイン枚数にはノイズが入り、さらに、次のターンで Exploit するときには、2 ターン分だけ確率 p_c で変化する。つまり、Innovate で得る情報より、1 ターン分より古くかつノイズが入った情報しか得ることができないが、n_{obs} が大きい場合には、低コストでレバー情報を得ることが出来る。自分が情報を持ったレバーの場合は、その情報がを更新する。

このトーナメントの目的は、従来の数理モデルの解析で扱わる限られたモデルではなく、多数・多様なアルゴリズムを集めて評価することにより社会的学習に関する一般的な知見を得ることにあった [?]。様々な分野の専門分野の専門家、学生のグループから 104 のエントリーがあった。p_c, n_{obs} や、ノイズの大きさなどを様々に変更した環境での総当たり、上位 10 個の戦略によるバトルロイヤル、1 ターンあたりのコイン獲得枚数に比例した確率で戦略をコピーするレプリケーターダイナミックスのフォーマットでトーナメントを行い、戦略の最終的な生存率でそのパフォーマンスを評価した。そこで得られたもっとも重要な結論はトライ＆エラーやコピーによる Explore でのコピーの実施比率 r_{obs} の高さがエージェントのパフォーマンスに直結したことである。その理由は、コピーで獲得するレバー情報は、他のエージェントが最適だと考えて Exploit したレバー情報であるというフィルタリングが機能しているからである。これはトライ＆エラーとコピーを比較した上で最適なほうを選択するのが適応的であるとｓした従来の考えとは異なる結論であった。

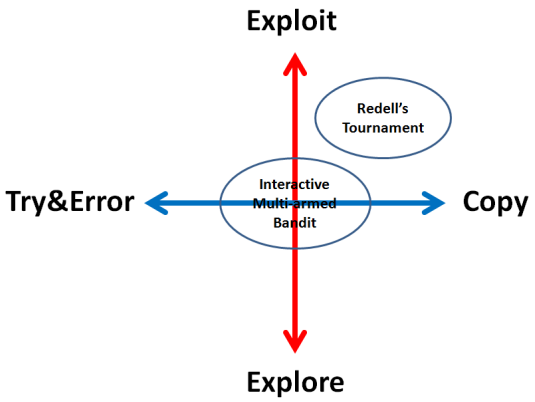

図 2 社会的学習のある非定常多腕 Bandit ゲームでの 2 つのトレードオフ

トーナメントの結果はトライ＆エラーとコピーの間のトレードオフは存在しないことを意味する。総当たり、上位10個でのバトルロイヤルの双方で1位となったdiscount machineというエージェントプログラムはInnovateの評価をすることなく、p_cを推定し、各レバーの期待リターンをそのp_cで割り引いて、ベストなレバーをExploitするか、それともObserveするかを選択するものであった。Innovateを行わないため、自己と同じエージェントプログラムしか存在しないときのパフォーマンス（コイン獲得枚数）は低いが、他のエージェントが存在する場合、そのエージェントの獲得したレバー情報をうまく取り込んで、圧倒的なパフォーマンスを示した。このトーナメントでは、Innovateは、エージェントの知らないレバーの情報を1本だけ獲得できるのに対し、Observeでは、他のエージェントの用いていたレバー情報を最低でも1本($n_{obs} \geq 1$)獲得できる。もともと、多数のコインの出るレバーが非常に少ない環境においては、Innovateすることにほとんど意味がない状況に設定されていたと考えられる。

では、トレードオフの状況にするためには、Innovateの価値を高めるのが簡単である。そこで、Innovateで情報が獲得できるレバーの数kをパラメータとて導入することを考える。このkをコントロールすることにより、非定常多腕バンディットを二つのトレードオフを持つ環境とした(**??**)。本研究の目的は、この環境での最適な戦略、レバー情報の伝達、ヒトの選択、そしてゲームとしての面白さを明かにすることにある。今回の発表は、その第一歩として、簡単な戦略で選択する多数のエージェントとヒト1人が対戦し、順位を競うインタラクティブ多腕バンディットゲームを開発し、少数の被験者で行った実験の結果を解析した結果の報告である。本文の構成は以下の通りである。セクション2では、実験環境である多腕バンディットとエージェントプログラムについて解説する。セクション3では、エージェントのパフォーマンスと戦略に用いているパラメータの関係の解析結果を述べる。セクション4では、被験者のパフォーマンスとデータから推定したパラメータの間の関係を解析する。セクション5はまとめと今後の課題について述べた。

2. 実験環境：インタラクティブ多腕バンディットゲーム

実験環境はヒト1人と120のエージェントプログラムが1つの多腕バンディットを舞台にコインの獲得枚数を競うゲームである。ヒトとエージェントプログラムに可能な選択は、Exploit,Innovate, Observeの3種類、1ゲームは100ターンで構成され、ヒトとエージェントそれぞれがこの3種類のどれかを選択することによりゲームは1ターン進行する。多腕バンディット（スロットマシン）は100本のレバーを持ち、1から100までの番号$n \in \{1, 2, \cdots, 100\}$をつける。レバー毎に、レバーを引く(Exploit)ことにより獲得できるコインの枚数は異なり、指数分布に従う乱数を自乗し整数値に丸めたものを用いる。その枚数は毎ターンある確率p_cで変化し(restless)、変化するときは再度指数分布を用いてコイン枚数を決定する。

プレイヤーおよびエージェントプログラムは、レバー情報を格納するレパートリーを持ち、その中にあるレバーしか引くことが出来ない。ここでレバー情報とは、レバー番号nと、そのレバーに対するコイン枚数xの組(n, x)のことである。もちろん、レバー情報は毎ターン確率p_cで変化しているため、レパートリーのレバー情報にあるのと同じコイン枚数を獲得できるとは限らない。レパートリーに格納可能なレバーの数は最大で3とした。3つの選択肢は次のとおりである。

(1) Innovate 100本の中からランダムにk本が選ばれ、その中から獲得コイン枚数の最も多いレバー情報が得られる。そのレバー情報はレパートリーに保存される。レパートリーにすでに3本のレバー情報が存在する場合、古いものを捨てる。kの値はゲームが終了するまで変化しないものとする。

(2) Exploitレパートリーから1つレバーを選択し、そのレバーを引き、レバーを引いたターンでのコイン枚数を獲得できる。獲得できるコイン枚数がレパートリーの情報と異なる場合、更新する。

(3) Observe

前ターンでExploitしたプレイヤーの中からランダムに1人選ばれ($n_{obs} = 1$)、そのプレイヤーが何番のレバーでコインを何枚獲得したかの情報が得られる。そのレバー情報はレパートリーに保存される。すでにレパートリーにそのレバー情報が存在する場合は、Observeで得た情報で更新する。前のターンにExploitしたプレイヤーがいなかった場合はからぶりとし、何の情報も得れない。

2·1 エージェントプログラムのパラメータ c, p_{obs}

プレイヤーと対戦するエージェントプログラムについて説明する。[**?**]のトーナメントの結果から、エージェントのパフォーマンスに直結する重要なファクターとして、Explore(Innovate or Observe)のうちのObserveする率r_{obs}があった。また、Observeが適応的な理由は、他のエージェントがExploitしたレバーの情報にコピーでのフィルタリング効果であった。そこで、この二つのみを取り入れた単純なエージェントプログラムを採用する。

(1) 閾値c

レパートリーの中のレバーで閾値cを越えるコイン枚数のレバーがない場合、InnovateかObserveを行うものとする。

(2) オブザーブ確率p_{obs}

InnovateかObserveを行う場合、確率p_{obs}でOb-

serve を選ぶ。

2·2 ゲーム環境

ゲーム環境では、閾値 c として 1 から 12 の 12 個の整数値, オブザーブ確率 p_{obs} として、0.1 から 0.9 までの 0.1 刻みで 10 個の値を組み合わせた合計 120 個の組み合わせ (c, p_{obs}) の値に設定したエージェントプログラムを用意し、ヒトはこれらのエージェントの集団と闘うことになる。ただし、対戦形式はリアルタイムにエージェントとヒトが闘うものではない。エージェント集団ですでに 999 ターンのプレイをシミュレートし、そのデータのうちからランダムに 100 ターン選んでヒトが参加する。そのため、ヒトはエージェントのレバー情報を Observe で獲得することができるが、エージェント側はヒトの情報を見ることはできない。ヒトは 100 ターンのゲームを行う前に、レパートリー情報を準備するステップとして Innovate または Observe を 3 ターン分実行できる（??）。ヒトがゲームに参加した時点でエージェントのコイン獲得枚数をゼロにリセットし、ヒト参加後のコイン獲得枚数でヒトおよびエージェント集団の順位を計算し、ゲーム画面に表示する（??）。

図 3 最初の 3 ターンの画面

図 4 ゲームプレイ中画面

バンディットのパラメータは (p_c, k) の二つである。実験では、(p_c, k) の次の 4 つの組み合わせを採用した。

- A モード:$(0.1, 1)$
  環境の変化は比較的ゆっくりで、Innovate でもいいレバーがなかなか見つからない状況
- B モード:$(0.1, 10)$
  環境の変化は比較的ゆっくりで、Innovate でいいレバーが見つかる状況
- C モード:$(0.2, 1)$
  環境の変化は激しく、Innovate でもいいレバーがなかなか見つからない状況
- D モード:$(0.2, 10)$
  環境の変化は激しく、Innovate でいいレバーが見つかる状況

被験者には、実験前に説明を行い、k, p_c の値の意味を説明した。そして、画面での 4 つのモードはこれらの値がそれぞれ異なることを説明し、すきなモードからゲームを始めてもらった（??）。ただし、どのモードがどういった値に設定されているかは教えていない。

図 5 mode の選択画面

3. エージェントプログラムのパフォーマンス

エージェントのコインの合計獲得枚数の p_{obs} 依存性と c 依存性を調べた。ここで扱ったデータはエージェント同士で 100000 ターン対戦させたときのものである。

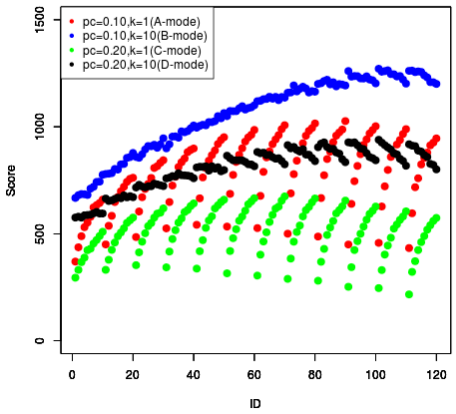

図 6 横軸:エージェントの ID、縦軸:コイン獲得枚数の合計
各モードでの 100 ターンあたり平均コイン獲得枚数

??は各モードにおける各エージェントの 100 ターンあたりの平均コイン獲得枚数を表したものである。横軸がエージェントの ID、縦軸が平均コイン獲得枚数である。赤が A モード、青が B モード、緑が C モード、黒が D モードを表す。エージェント ID とパラメータ (c, p_{obs}) の関係は、ID が 1 から 10 は $c = 1$、ID:11 から 20 は $c = 2$、$\cdots$ と c が 1 から 12 の 12 グループに分けられ、各グループの ID が低い順に 1 人目が $p_{obs} = 0.0$、2 人目が $p_{obs} = 0.1$、10 人目が 0.9 と設定されている。この?? から、A-mode(赤) と C-mode(緑) は p_{obs} が高いと獲得枚数が高いことがいえる。A-mode(赤) と C-mode(緑) の共通する点は $k = 1$ であり、Innovate で地道で真面目に良いレバーを探すエージェントよりも Observe でその真面目なエージェントたちが Exploit している良いレバーを得たが方が得であるからと考えられる。しかし、B-mode(青) と D-mode(黒) の場合の $k = 10$、すなわち Innovate の価値が高い時、地道で真面目に Innovate して

いる c が高めなエージェントの場合は p_{obs} が高いエージェントの方が獲得枚数が低くなっている。これは、$k=10$ により Observe するエージェントよりも先に Innovate している真面目なエージェントが良いレバーを見つけ出し、その後に Observe で情報を得るため、その差が獲得枚数に影響しているものだと考えられる。さらに、D-mode(黒) の場合には $p_c=0.2$ と環境が激しく変化するため、Observe で情報を得てもその情報は古く、Exploit するときには得れるはずのコインが得れないことが多い。そのため、p_{obs} が高いエージェントは真面目に Innovate するエージェントに比べて獲得枚数が低い、あるいはイーブンであるという傾向がみられた。

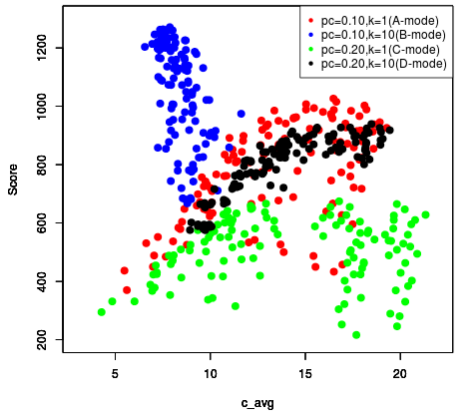

図 7　横軸:c、縦軸:平均コイン獲得枚数

次に、エージェントの実際の選択のデータから平均の閾値 c_{avg} とオブザーブ r_{obs} を計算し、平均コイン獲得枚数の相関を調べた。c_{avg} は、エージェントが Innovate,Observe から Exploit に選択を変更したときのレバーのコイン枚数の平均値とする。また、r_{obs} は、Innovate,Observe のうちの Observe の比率とする。ともに、エージェントの選択データから計算可能である。c, p_{obs} の代わりに、c_{avg}, r_{obs} を用いる理由は、エージェントの場合は c, p_{obs} はパラメータとして設定出来るが、ヒトの場合はミクロな選択モデルが不明であり、データから推定するしかないからである。

??は c_{avg} と平均コイン獲得枚数の散布図である。横軸が c_{avg}、縦軸が平均コイン獲得枚数である。A モード (赤) と D モード (黒) は相関係数が 0.55、0.83 であり、c_{avg} がエージェントのパフォーマンスに寄与していることが分かる。A モード (赤) の下側の領域にいるエージェントは、**??**から p_{obs} が小さい地道で真面目に Innovate するエージェントであると考えられ、c が高いと Exploit できず、パフォーマンスが低下したと考えられる。それに比べて D モード (黒) は $k=10$ より c が高くてもレバーを探すことが可能で、また、$p_c=0.2$ で環境が激しく変化する状況にあるので、真面目に innovate しているエージェントも Observe しているエージェントもあまり関係なく単純に、c が高いエージェントが多くのコインを獲得し、強い相関がみられたと考えられる。C モード (緑) は相関係数が 0.10 となった。これは、D モード同様に環境が激しく変化しかつ $k=1$ とかなり貧困な状況であるため c の大きさがコインの獲得とは関係ないためであろう。B モード (青) は相関係数が-0.58 と負の相関がみられた。これは、$p_c=0.1$ かつ $k=10$ と豊かで穏やかな環境であるため、ゆえに、c が低いエージェントたちがレバーを容易に見つけてコイン枚数を稼げる一方で、c が高いエージェントたちのうち p_{obs} が高いものは、Observe で低いコイン枚数のレバーしか見つけられず、Exploit の機会を失ったためと考えられる。

下の各図はエージェントのオブザーブ率 r_{obs} と平均コイン獲得枚数の散布図である。横軸が r_{obs} で縦軸が平均コイン獲得枚数となっている。

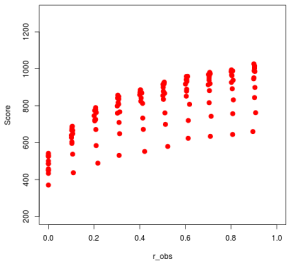

図 8　A モード

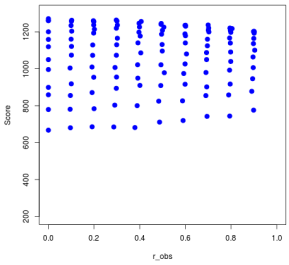

図 9　B モード

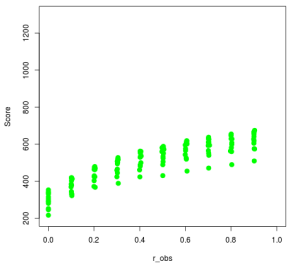

図 10　C モード

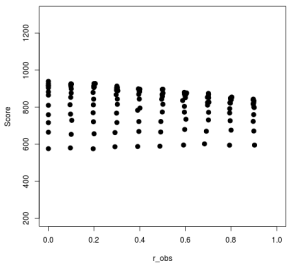

図 11　D モード

A モードと C モード、B モードと D モードとで同じ傾向がみられた。A モードと C モードはそれぞれ相関係数が 0.83 と 0.88 と非常に高い相関がみられた。これは、$k=1$ なので環境の変化によらず Innovate で地道で真面目にレバーを探すよりも Observe するエージェントが良いレバーを見つけ出すのでこのような結果がみられたと考えられる。次に、B モードと D モードの相関係数は 0.04 と-0.13 とほぼ相関がなかった。ゆえに、地道にレバーを探しているエージェントとひたすら Observe しているエージェントに差がないことを意味していて、これは、$k=10$ なのでひたすら Innovate するエージェ

ントとひたすら Observe するエージェントどちらのエージェントも最終的に良いレバー情報が得ることができるからであると考えられる。また、D モードの若干の負の相関は前でも述べたように激しい $p_c = 0.2$ と激しく変化する環境なので Observe するエージェントがそのレバーを Exploit しようとしたときにはコイン枚数は変化しているためであると考えれる。

以上の結果から、$k = 1$ の A モード、C モードでは、Innovate にあまり価値がないためコピーとトライ＆エラーのトレードオフは存在せず、p_{obs}, r_{obs} とパフォーマンスは正の相関を示した。一方、$k = 10$ の B モード、D モードでは、相関がほとんどなく、二つの情報の獲得にはトレードオフが成立していることが分かる。

4. 被験者データの解析結果

今回の実験は被験者により真剣に実験に取り組んでもらうために、各モードで 20 位以内に入ったらクオカード 300 円分というリターンのみで実験を行った。参加費などの報酬は存在しない。被験者は北里大学理学部物理学科の学生 20 人弱。獲得コイン枚数と c_{avg}, r_{obs} 依存性を解析した。下の各図は A から D モードまでの被験者の c_{avg} とコイン獲得枚数の合計の散布図を表したものである。横軸が c_{avg} で縦軸がコイン獲得枚数となっている。

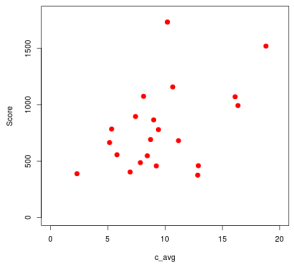

図 12 A モード

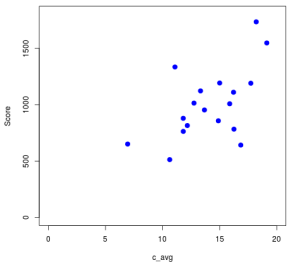

図 13 B モード

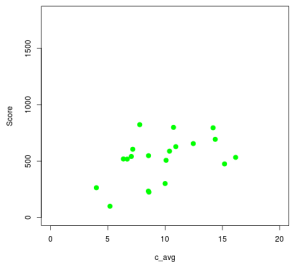

図 14 C モード

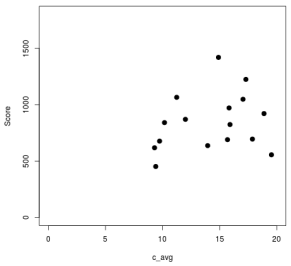

図 15 D モード

まず、A から D モードのそれぞれの相関係数は 0.42、0.52、0.44、0.25 という結果であった。$k = 1$ の場合の A モードと C モードと $k = 10$ の場合の B モードと D モードの c_{avg} を比較すると $k = 10$ の場合の方が全体的に高い傾向がみられた。これは、$k = 10$ により全員が Innovate しても枚数の多いレバーを引き当てることができてしまいかつ、より高い c をもって Exploit していかないと順位が上がらなかった状況であったと考えられるので全体的に c_{avg} が高めであったといえる。C モードでは $p_c = 0.2, k = 1$ と貧困な環境なのでなかなか良いレバーを見つけることができず地道にレバーを Exploit しつづけるのとたまたま良いレバーを見つけて Exploit するのはあまり変わらないが少しでも c が大きくないと順位が上がらないためシミュレーションよりは相関がみられた。さらに、D モードにおいては、シミュレーションでは一番高い相関がみられたが実験結果は 4 つの内一番低い相関であった。これは、$p_c = 0.2$ と環境が激しく変化してしまうため、あまり大きい c を持たず少ない獲得枚数でも Exploit して順位を悪くしないようにしていたのではないかと考えられる。全体的には、c が高いと獲得合計枚数も高いという結果であったが、B モードに関しては、シミュレーションと真逆の相関となってしまった。これは、エージェントは順位をみていない状況に対して被験者は順位を確認しながら c を変化させているとのでそれが原因だと考えられる。

次に下の各図は A から D モードでの被験者のオブザーブ率 $_{obs}$ とコイン獲得枚数の散布図である。横軸が r_{obs} で縦軸がコイン獲得枚数の合計となっている。

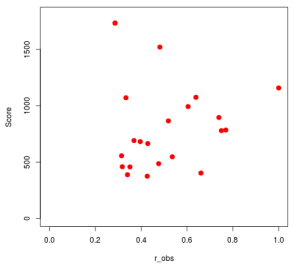

図 16 A モード

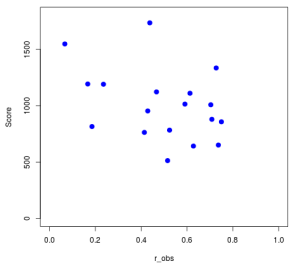

図 17 B モード

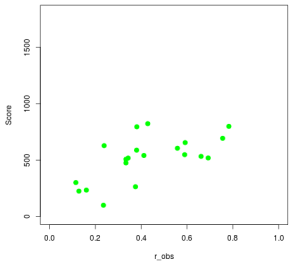

図 18 C モード

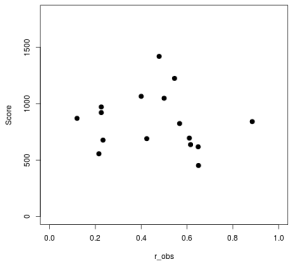

図 19 D モード

まず、A から D モードそれぞれでの相関係数は 0.01、-

0.37、0.6、-0.09 であった。シミュレーションでは A モードと C モードで高い相関がみられたが、この結果によると、A モードに相関がなかった。これは、たまたま良いレバーを得てしまった (すばぬけて獲得合計枚数が多い) 被験者がいてそのデータが効いていると考えられる。次に、B モードと D モードでは負の弱い相関がみられた。これはエージェント集団と同様に、Observe と Innovate の間にトレードオフの関係が起きていることを示唆する。また、それぞれの mode においての r_{obs} と c の重回帰分析を行った。結果をまとめたのが下記の表である。

表 1 p 値

mode	r_{obs}	c	$r_{obs} : c$
A-mode	0.837	0.0709	0.9455
B-mode	0.7678	0.0216	0.0925
C-mode	0.00256	0.0367	0.05538
D-mode	0.792	0.351	0.347

表 2 回帰係数

mode	r_{obs}	c	$r_{obs} : c$
A-mode	0.044528	0.426259	-0.020835
B-mode	-0.0664	0.5583	0.3254
C-mode	0.6007	0.3297	-0.3964
D-mode	0.08137	0.27467	0.25832

5. 考察と今後の課題

エージェント集団および被験者のデータ解析の結果、$k = 10$ の環境では r_{obs} とパフォーマンスの間には正の相関はなく、Innovate と Observe の間にはトレードオフの関係が成立していることが分かった。ただ、被験者数が限られていたため、ヒトの選択の詳細まで踏み込むことはできなかったが、ヒト対ヒトの集団実験を目指すにあたっての最初の一歩にはなったと考えている。

また、エージェントプログラムのアルゴリズムが単純すぎるため、その点は改良したゲーム環境を整備する必要があるとも考えている。ランキング情報をエージェントに与えて、最適化を行った場合、エージェントは c, p_{obs} を調整しながらゲームに参加するようになる。その際、ヒトはどう選択し、集団挙動に反映されるのか。また、エージェントとヒトはどのように異なるのか。今後の課題である。

謝　　辞

本研究は科研費 25610109（挑戦的萌芽研究）の助成を受けました。