上田 佳祐　　東京大学教養学部学際科学科2年
ueda-keisuke@g.ecc.u-tokyo.ac.jp

科目名：広域システム概論

関連講義：福永アレックス先生担当の自動計画、GAに関する講義

コードURL：https://github.com/u-keisuke/ga_reversi

2021 年 1 月 20 日

リバーシゲームの盤面評価関数の実数値GAによる最適化

1. 概要

本レポートでは、状況評価をスコア盤面関数ベースで行うリバーシゲームAIを実装し、その関数の内部パラメータを遺伝的アルゴリズム（以下GA）によって最適化することを目指した。

2. 手法

2.1. リバーシのアルゴリズム (reversi.py)

0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	1	-1	0	0	0
0	0	0	-1	1	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0

図1 Bij

リバーシの盤面は、自分の石があるマスを 1、相手の石があるマスを -1、どちらの石もないマスを 0 とすることによって数字を使って表現できる。例えばゲーム開始時点の盤面 B_{ij} は図１のように表せる。

次に本実験で作成したリバーシAIのアルゴリズムの概要を説明する。AIが打ち手を導くアルゴリズムにはMin-Max法を用いている。（Alpha-Beta法というMin-Max法の探索効率を枝刈りによって改善したものを採用する予定でコードもそのように書いていたがデバッグが終わらなかった。）またその際の先読み手数は3手先までとしている。

また盤面評価の方法として各マスにポイントを割り当て、そのマスに自分の石があれば評価値に加え、相手の石があれば評価値から引くとして、その総計を盤面評価関数eとする。そのような各マスのポイントを盤面上に並べたものをスコア盤面E_{ij}と呼ぶことにする。（例えばE_{ij}は図２のように表せる。）

100	-20	20	5	5	20	-20	100
-20	-40	-5	-5	-5	-5	-40	-20
20	-5	15	3	3	15	-5	20
5	-5	3	3	3	3	-5	5
5	-5	3	3	3	3	-5	5
20	-5	15	3	3	15	-5	20
-20	-40	-5	-5	-5	-5	-40	-20
100	-20	20	5	5	20	-20	100

図2 Eij

その際、盤面評価関数eは以下のように定義される。

$$e = \sum_{i=1}^{8} \sum_{j=1}^{8} B_{ij} E_{ij}$$

またスコア盤面E_{ij}はリバーシゲームの対称性から、図２で塗りつぶされている10個のマスの値が分かれば導くことができる。そこで、その10個のマスの値をベクトルのように並べたものをgとする。ただしGAを適用する過程でgの値が発散、あるいは0へと収束してしまわないように、gは平均0、最大値が常に100になるように正規化しておくものとする。

本レポートではこのgをGAアルゴリズムによって様々に生成、選択、交叉、変異を行い、最適化することを目指す。

2.2. GAによる最適化 (ga.py)

上記で述べた通り g を「遺伝子」としてGAによって最適化していく。GAでは一般に4つのステップに分けられる；初期集団の生成、集団からの選択（淘汰）、選択されたものの交叉、そして変異である。全体の流れとしては図３に示す通りである。

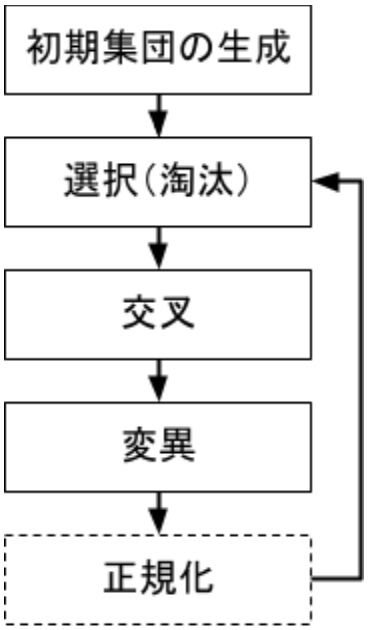


図3

2.2.1. 初期集団の生成

平均0、標準偏差10のガウス分布に基づいた乱数を10個生成し、それを並べたものを g とする。そのような g を個体数 n_pop だけ生成する。つまり１世代の遺伝子プール全体は $n\_pop \times 10$ の行列によって表される。

2.2.2. 選択（淘汰）

トーナメント方式で各個体同士を対局させその勝敗によって集団の淘汰を行う。具体的にはトーナメント方式で勝った個体だけを残していき、残存個体の数が n_sel ($n\_sel < n\_pop$) になった時点で淘汰を終了する。

2.2.3. 交叉

遺伝子 g の表現形式が実数ベクトルである（実数値GA）ことを考慮すると、通常の０１で遺伝子が表されているGAで採用されている交叉方法（例えば一点交叉や一様交叉）とは異なる方法が必要と考えられる。実数値GAの交叉方法としては、$BLX-\alpha$ 交叉や SPX 交叉などが提案されている [樋口 2001]。本レポートでは $BLX-\alpha$ 交叉を採用した。 $BLX-\alpha$ 交叉では、2つの親個体の d 次元ベクトル（本レポートでは $d=10$）を対角点としてもつような d 次元超直方体 K を考える。そのような K の各辺の長さを $1+2\alpha$ 倍したものを K' とし、その内部において一様ランダムに生成した点を子個体とする。（図４：簡単のために $d=2$ の場合を図示している。）その際 α の値は0.3程にすることが提案されている [静岡理工科大学情報学部 online: lbx_spx.html]。

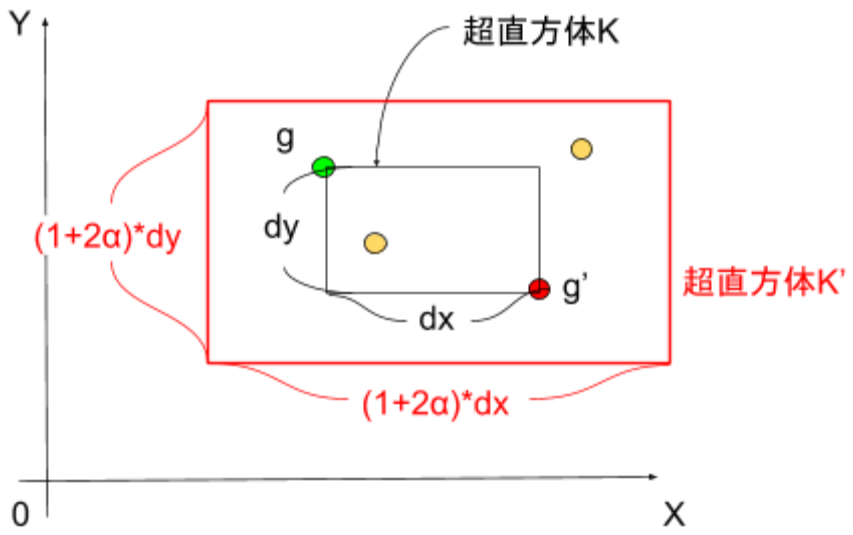


図4

2 つの親個体 1 組から生成される子個体の数を c（上記の例では $c=2$）として、淘汰された親個体（n_sel 体）の全組み合わせ ${}_{n\_sel}C_2$ 通りに対して $BLX-\alpha$ 交叉を行った場合、生成される子個体の数は $c \cdot {}_{n\_sel}C_2$ となる。

また本レポートでは、親個体 n_sel 体はそのままコピーして次の世代に受け継がれるとする。その場合、全子個体の数は $n\_sel + c \cdot {}_{n\_sel}C_2$ となる。したがって各世代で個体数が一定であるには以下の等式が成立している必要がある。

$$n\_pop = n\_sel + c \cdot {}_{n\_sel}C_2 \qquad \cdots (*)$$

また上記で遺伝子 g の正規化の必要性を述べたが、$BLX-\alpha$ 交叉はその特性としてスケールへの依存性をもつため [樋口 2001]、この観点からも g の正規化が必要であると考える。

2.2.4. 変異

実数値GAにおける変異の方法は様々なものが提案されているが、本レポートでは以下の 3 つの変異を等確率で組み合わせて用いている。すなわち変異発生確率 p で変異が起こり、変異形式については以下の 3 つからランダムに選ばれる。

- 個体の遺伝子 1 箇所をランダムに選択し、平均 0、標準偏差 20 のガウス分布に基づいた乱数を足す。
- 個体の遺伝子 1 箇所をランダムに選択し、-1 倍する。
- 個体の遺伝子 2 箇所をランダムに選択し、それらの値を入れ替える。

3. 実験条件

本レポートで行った実験条件、各種パラメータの設定について述べる。実験は2回に渡って一部異なる設定で行われた。

3.1. 1回目

世代数： 56
パラメータ設定

- 個体数： $n\_pop = 64$
- 淘汰後の個体数： $n\_sel = 8$
- 親個体1組から生成される子個体の数： $c = 2$
- $BLX-\alpha$ 交叉法のパラメータ： $\alpha = 0.3$
- 変異確率： $p = 0.1$

ただし重要な留意点として、実験者の実装ミスにより正規化の方法と変異方法が異なっていることに注意。

正規化：最大値は常に100となっているが、平均は0になっていない。
変異方法：個体の遺伝子の**全箇所**に、平均0、標準偏差10のガウス分布に基づいた**同じ**乱数を足す、という操作1つになってしまっている。

3.2. 2回目

世代数： 23
パラメータ設定

- 個体数： $n\_pop = 64$
- 淘汰後の個体数： $n\_sel = 8$
- 親個体1組から生成される子個体の数： $c = 2$
- $BLX-\alpha$ 交叉法のパラメータ： $\alpha = 0.3$
- 変異確率： $p = 0.2$

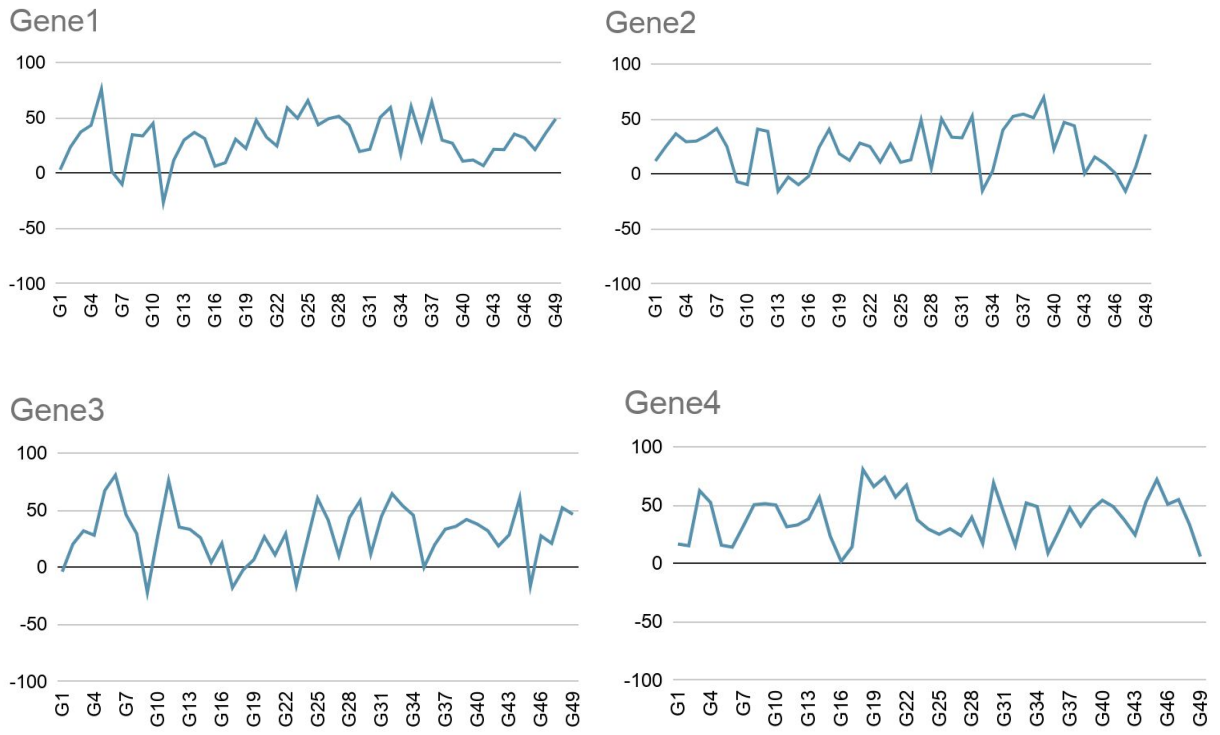
また上記のどちらの実験条件においても等式（∗）が成立しているため、各世代での個体数は一定になっていることに注意。

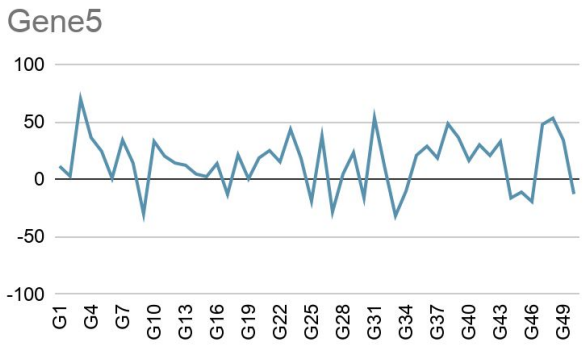
4. 結果・考察

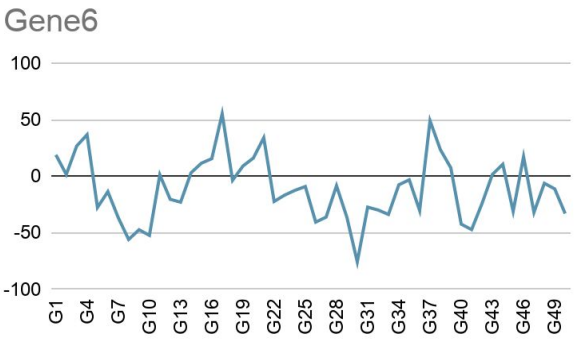
4.1. 1回目

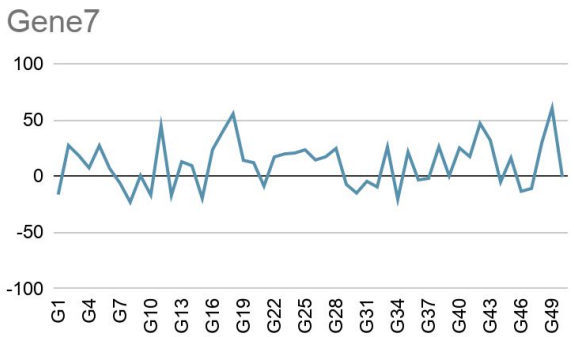
- 実行環境
  - プロセッサ：2.3 GHz デュアルコアIntel Core i5
  - メモリ：16 GB 2133 MHz LPDDR3
- 純実行時間：約14時間（約1,000秒 / 世代）（CPUの使用状況により変化）
- 各世代の遺伝子データは次のファイルを参照：data.json

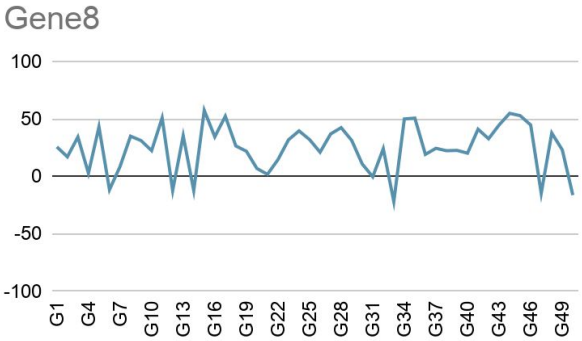
以下のグラフは、各世代で選択された8個体の遺伝子に対して各次元(1〜10)ごとで平均をとった値が、世代を経るごとにどのように遷移したかを表したものである。仮に上手く最適化できていたならば、1〜10の全てのパラメータにおいてそれぞれある値に収束しているはずである。

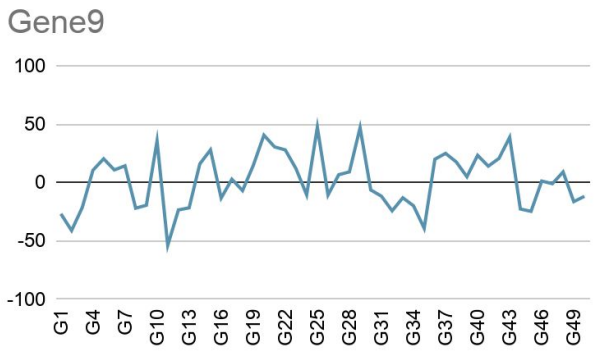


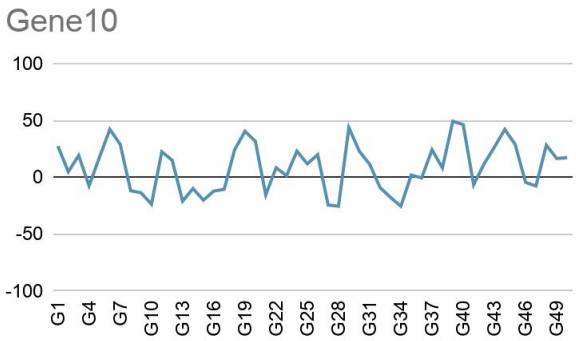










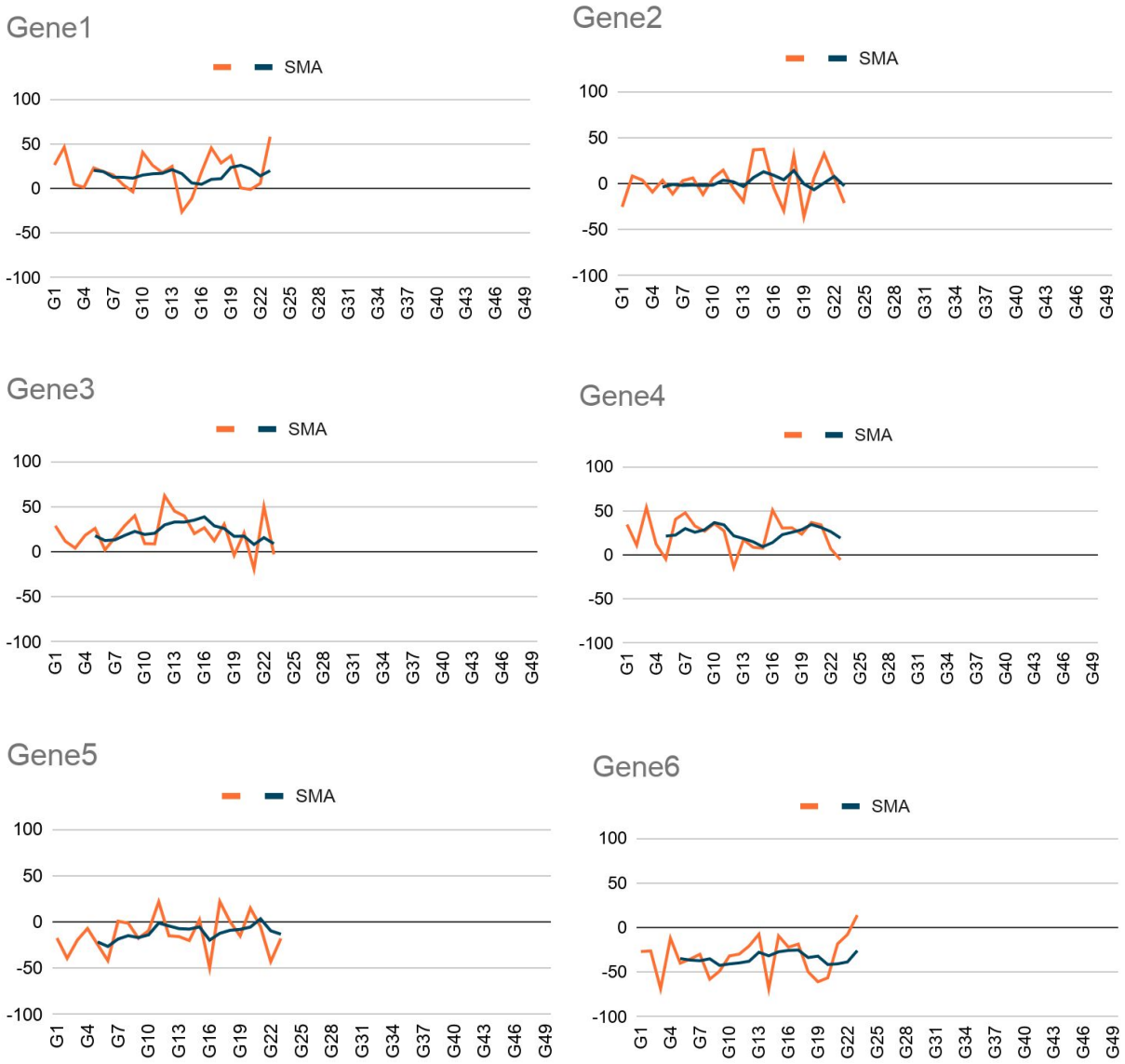


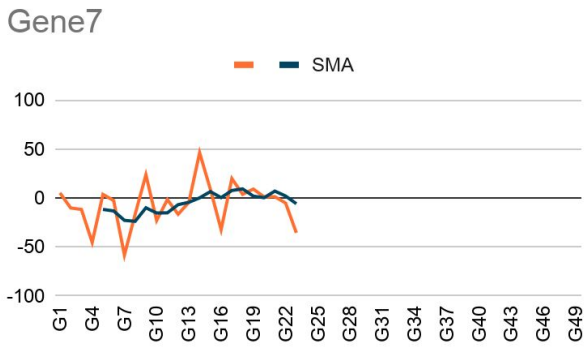
上記のグラフから分かる通り遺伝子の各パラメータは上手く収束できておらず、値0の周辺をランダムウォークしているだけのように見える。どのパラメータも最適化に失敗していると考えられる。

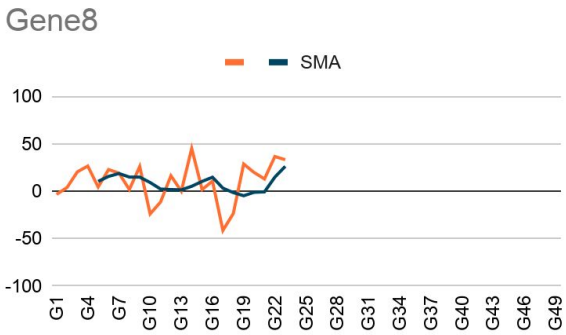
4.2. 2回目

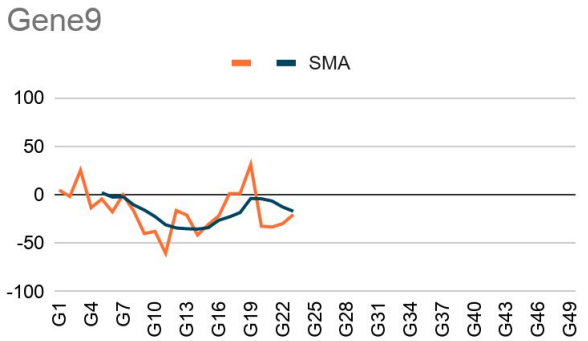
- 実行環境：（同上）
- 純実行時間：約7時間（約1,000秒 / 世代）
- 各世代の遺伝子データは次のファイルを参照：data_v2.json

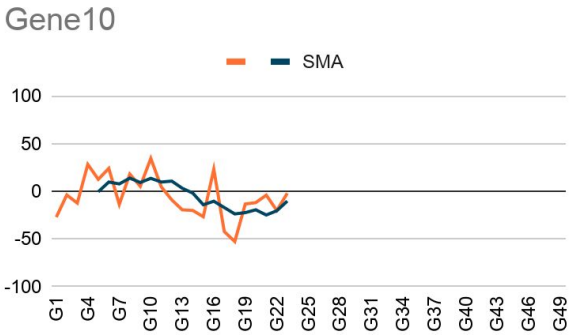
上記と同様に遺伝子の各パラメータの平均の変化を見てみる。SMAと表記されているのは世代間での区間5の移動平均を取ったものである。











上記のグラフから分かる通りこちらの実験でも遺伝子の各パラメータ平均は上手く収束できておらず、また第一世代の初期値に非常に依存した値を第二世代以降も取っていることが分かる。どのパラメータも最適化に失敗していると考えられる。

5. 最適化失敗の原因考察

何故上記のように最適化に失敗してしまったのか考えられる原因の一つはGAの選択（淘汰）方法である。今回採用した方法は個体同士を互いに対局させて、その勝敗によって残存個体を決めるものであった。この方法のメリットは計算量の小ささである。具体的には一世代当たりの総対局回数を高々 n_pop に抑えることができる。

しかしこの方法では、個体の適応度（性能の良さ）を絶対的な指標（あるいはその近似）で測ることができず、他の個体がパラメータ空間にどのように分布しているかに大きく依存してしまう。つまり世代ごとに性能の指標が大きく変わってしまう可能性がある。

あるいは別の方法として考えられるのは、ランダムに生成した仮想相手に対する勝率によってその個体の性能を測る方法である。この方法のメリットは、個体の性能を世代によらず等しい指標で測れることである。したがってGAの他のステップで問題がな

く、かつそもそもの遺伝子の表現力が真の解空間を近似できないほど小さくなければ、問題なく最適化できるはずである。

　しかしこの方法のデメリットはやはり計算量である。具体的には一世代当たりの総対局回数が少なくとも *n_pop* の数十倍になってしまう。この問題はリバーシの一回当たりの対局計算の計算量を減らすことができれば解決すると考えられる。

6.　注釈

本レポートの実験に使ったプログラムの実装に関して参照した記事を記す。

- 「numpyオブジェクトをjson.dumpできるようにエンコーダーを拡張する方法」

Works Cited（引用文献）

[樋口 2001] 樋口, 筒井, 山村: 実数値GAおけるシンプレックス交叉の提案, 人工知能学会論文誌, Vol. 16, No.1, pp.147-155 (2001).

[静岡理工科大学情報学部 online: lbx_spx.html] 実数型GAに於ける交叉法の改良, https://www.sist.ac.jp/~kanakubo/research/evolutionary_computing/lbx_spx.html (Retrieved on January 20, 2021).