# 段階的 GP による数式の生成

日大生産工 (院)　○山野井裕基
日大生産工　　松田　聖

## 1　はじめに

本研究では, 与えられたデータの相関関係を表す数式を遺伝的プログラミング (GP) による自動探索で導く手法として, 段階的 GP を提案する. 単純な GP では探索効率が悪い事が知られているが, このモデルでは段階的にデータを抽出しては GP で探索する事により効率を改善するのがねらいである.

## 2　研究の背景

GP とは, 生物が進化して行くように構造的なデータを進化させて問題の解を得るヒューリスティクスである. そして, GP による数式の探索とは, 相関関係を求めたいデータの変動値と出力値の組み合わせを任意の件数与え, それら全てを満たす, またはより近似する数式を導くというものである. 単純な GP (以下 SGP と呼ぶ) によって数式を導く手順の例を Fig.1 に示す.

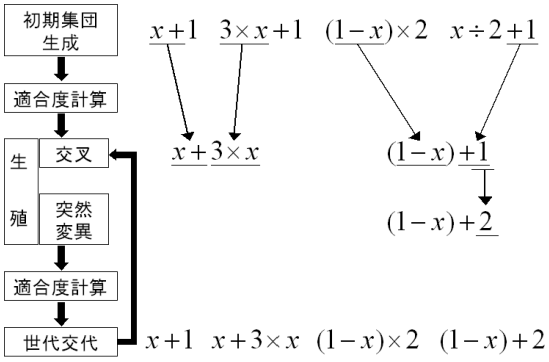


Fig.1　SGP による数式探索の例

(1) ランダムに数式をいくつか作る. 数式は木構造で表現される. 数式の作成数を集団の大きさと呼び, 数式各々を個体と呼ぶ. また, 個体の集合を集団と呼び, 特に初めに作った個体の集合を初期集団と呼ぶ.

(2) 生成した初期集団の個体について, どれだけ正解に近いかを示す適合度を計算する. 個体を $I$, データ件数を $n$, データの変動値を $x_1, \cdots, x_j$, データの出力値を $y$ とすると, 適合度は以下のように絶対誤差の相加平均で求められる.

$$F = \sum_{i=1}^{n} |I(x_{i1}, x_{i2}, \cdots, x_{ij}) - y_i|$$

したがって, 適合度は常に 0 以上で, 0 に近付くほど優良な解となり, 0 ならば正解となる,

(3) 適合度の良い個体を二つずつ何組か選び, 双方の部分木を入れ替えて新たな数式を作る. この操作を交叉と呼ぶ.

(4) (3) で生成した個体の中からランダムにいくつか選択し, 部分木を新たな部分木に置き換える. この操作を突然変異と呼ぶ.

(5) (3),(4) で生成した個体について, (2) と同様に適合度を計算する.

(6) 初期集団と (3),(4) で生成した個体を混ぜて適合度の良い個体を集団の大きさ分だけ選び, 次の世代の集団とする.

(7) 以降, 初期集団ではなく (6) で生成した集団について (3) ～ (6) を繰り返す.

(8) 正解, すなわち適合度が 0 となる個体が得られた場合はそこで終了する. 設定された最大世代に達した場合, その世代で最も適合度の良い個体を近似解とし終了する.

このような SGP でも簡単な相関関係を導くことはできる [1]. しかし, 例えば Two-Boxes 問題[*1] のように, 多少問題が複雑になるだけで正解を得る

---

Generating Mathematical Equations with Gradual Genetic Programming

Hiroki YAMANOI, Satoshi MATSUDA

[*1] 二つの箱について体積の差を表す数式を求める問題.

のが難しくなる. その原因は, スキーマの破壊にあると考えられる.

スキーマとは個体の持つ部分構造の事である. 例えば, ある個体が求めたい相関関係の特徴を捉えた部分木を含んでいるならば, その部分木は有効なスキーマであると考えられる. しかし, 生殖によってそのスキーマが破壊されてしまう可能性があるため, 有効なスキーマを活かしきれるとは限らない.

そこで, 集団を意図的に効率良く進化させるために, 有効なスキーマを生成してそれを元に個体を組み立てて行く手法がいくつも考案されており, その代表的なものとして ADFs (Automatically Defined Functions)[2] が挙げられる. これは数式を導く進化過程とは別のランダムな探索によってスキーマを生成し, 個体を構成する要素として生成したスキーマを使用するというもので, 探索効率の大きな改善が見られる事がわかっている. しかし, ADFs は予めスキーマの構造を決めておかなければならず, 問題に対してどのようなスキーマを生成すれば効率が良いかという知識が必要になるため, 汎用性があるとは言い難い.

## 3 段階的 GP の概要

本研究で提案する段階的 GP (以下 GGP と呼ぶ) とは, 問題を小問題に分解し小問題を徐々に大きくして行く事で解を得る手法である. Fig.2 に GGP の大まかな手順を示す.

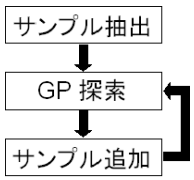


Fig.2 GGP による数式探索の流れ

(1) データの中からランダムに 2 件選ぶ. 抽出したデータをサンプル集合とする.

(2) サンプル集合について SGP による探索を行う.

(3) データの中からランダムに 1 件選び, サンプル集合に追加する.

(4) サンプル集合について SGP による探索を行う. このとき, 前フェイズ ((3).(4) の操作を合わせて 1 フェイズとする) で得た解 (P スキーマと呼ぶ) を数式の要素に用いる.

(5) 以降, 全てのデータを網羅するまで (3),(4) の操作を繰り返す.

GGP の最も大きな特徴は (4) で, 前フェイズで得た解をスキーマとして数式に組み込む事ができる. 前フェイズの解がそのままスキーマとなるため, GGP はスキーマの生成に問題への知識を必要としない汎用性のある手法である.

## 4 GGP の効果

サンプル集合が大きくなると, サンプルの追加によってサンプル集合全体の性質が受ける影響は小さくなる. そのため, P スキーマが有効なスキーマとして機能するものと考えられる.

逆に, 以下の欠点も考えられる.

(1) ランダムにサンプルを抽出するので, 常に有効な P スキーマが生成されるとは限らない.

(2) サンプル集合が小さい初期のフェイズでは, サンプル追加によってサンプル集合全体の性質に大きな影響が生じる. そのため, P スキーマを用いる事によって探索効率が落ちる. または P スキーマが数式中に出現しない可能性が高くなる.

## 5 今後

現在 Linear GP [3] を取り入れた GGP の探索プログラムを作成しており, これを用いてシミュレーションを行う. シュミレーションの内容は, ノイズを含まないものと, ノイズを含んだ実際の統計データ双方について行う予定である.

## 参考文献

(1) 山野井裕基, GP による数式の自動生成, 平成 18 年度 数理情報工学科卒業研究発表会, 2007, p.173.

(2) John R. Koza, Genetic Programming II: Automatic Discovery of Reusable Programs, MIT Press, 1994.

(3) 伊庭斉志, 人工知能学会, 進化論的計算手法, オーム社, 2005, p.78~83.