N-003

プログラミング導入教育と実践的なアルゴリズム設計
－非線形最適化アルゴリズム設計の試み－
Introductory Programming and Practical Algorithm Design
－Trial of an Algorithm Design for Non-linear Optimization－

辻　康孝†
Yasutaka Tsuji

1. はじめに

プログラミング導入教育において，現実的な問題への取り組みを通して，プログラミングの基本概念や文法を学習するアプローチが試みられている（例えば [1]等）．

本稿では，プログラミング導入教育の中で，多点探索に基づく非線形最適化アルゴリズムの実装を通して，実践的な配列操作とアルゴリズム設計を学習者に取り組ませる授業設計とその学習成果について報告する．プログラミング自体への関心度だけでなく将来選択する専門分野も多様な学習者を対象としたプログラミング導入教育の中で，プログラミング言語の基本概念に加え，特に学習意欲の高い学習者に，より実践的プログラミングスキルの習得を目指させる枠組みについて述べる．

2. 多様な学生に対するプログラミング導入教育

2.1 プログラミング導入教育の重要性

非情報系の専攻学科でもプログラミングの知識の習得が必要になりつつある．海外の事例ではあるが，インターンで IT 企業を目指す非情報系学生が共通言語としてプログラミングの知識を必要としている状況も指摘されている[2]．また「文理融合型」のように幅広い分野を学習し，その中から希望する専門分野に進んでいく場合，プログラミングの授業が情報系分野を進路として選択するかどうかに影響を与える場合も多い．

2.2 本実践の対象となる学生

著者は，2014 年度より九州大学経済学部経済工学科にてプログラミング導入教育に相当する「プログラミング演習（1 年前期）」を担当してきた．同学科の学生は，経済システム，経営政策分析，数理情報の三分野に関して幅広く学び，三年次以降自身の関心のある分野に進んでいく．「プログラミング演習」は選択必修科目として開講しており，ほとんどの学生が受講する．入学時から数理情報分野に関心を持つ学生も一定割合存在しており（20～25%），同程度，未定の学生も存在する．

3. 演習課題としての最適化アルゴリズム

3.1 多変数関数の最適化

多変数の実数値関数の最適化問題は数理情報分野では，さまざまな形式で頻出する基本的なトピックの一つである．数学的には，勾配（偏微分値）が 0 になる点を候補点として求めることになり，アルゴリズムの授業の中では，勾配情報を用いた手法（最急勾配法や準ニュートン法等）が取り上げられる．

3.2 確率的多点探索法

複雑な問題に対しては，勾配情報を用いない確率的多点探索法が適用されることが多い．これらは，一般に進化計算と総称され，基本的なアルゴリズムの枠組みとしては以下のようなものである．なお以降では，最小化問題を前提に説明を行う．

（a）ランダムに探索点群を生成し，各点の関数値を計算．

（b）点群の情報を用いて，次の新しい点群を新たに生成する．その際，以下のことを考慮する．
- b-1 今までで一番関数値が小さい点を残す（option）
- b-2 未探索領域に新しい点群を生成
- b-3 良い点（関数値が小さい）の周囲に点群を生成

（c）(b)の操作を多数回繰り返す．

一連の（a）～（c）の手続きを自然界の仕組み（例えば，生物の行動学）に関連付けて，自由な発想のもと数多くの手法が考案されている．有用な手法もあるが，一部の手法のアプローチには批判的な意見もある．

3.3 演習課題としての確率的探索法

勾配法に基づく手法は，数学理論や表記及びアルゴリズム実装において，一年生でプログラミング初学習者には難しい内容である．複雑な関数に対しては探索性能自体の問題もある．一方，前述のように確率的多点探索法の骨子部分においては，さほど高度な数学知識は必要としないため，数理情報分野にあまり関心がない学生でも説明は十分理解でき，全受講者に対し同分野のトピックを紹介できる．

確率的多点探索手法のアルゴリズム実装を演習課題として扱う場合，以下のような実行可能性及び教育上の利点があると考えられる．

- アルゴリズムの骨子部分の実装だけなら，制御構造（順次，反復，多分岐），二次元配列，乱数生成の知識だけで可能．また学習対象言語が C でも実装が可能である．
- 骨子部分の実装は，制御構造と配列演算の複雑な組合せになり，実践的なアルゴリズム実装を体験できる．また，ある程度動くプログラムができたならば，例え部分的に間違っていたとしても，学生にとっては自信に繋がる．
- 探索性能を改善するアイディアを柔軟に考え出すことができ，自身のアイディアを容易に骨子部分の改良あるいは機能の追加として，アルゴリズムに反映できる．この過程を通して実践的なアルゴリズム設計を体験できる．
- 対象とする多変数関数の定義は，教員側が予めユーザ定義関数で与えることができ，ユーザ定義関数が難しいと

† 九州大学大学院工学研究院
Faculty of Engineering, Kyushu University

考える学生もこのタイプのユーザ定義関数ならプログラム中で容易に使うことができる．

・変数の数が調整可能で，形状や難易度の異なる多数のベンチマーク問題が準備されており，自身が設計・実装したアルゴリズムの性能を客観的に評価できる．

・骨子部分だけなら，50 行程度のソースコードで記述でき，学生にとって過度な負担とならない．評価する教員にとっても比較的負担が少ない．

4. 実施のための授業設計

4.1 授業方針

本授業は，多様な学生に対するプログラミング導入教育であるため，授業内容はプログラミング言語（現在：Python,，2017 年以前：C）の基本文法の習得を目指した一般的なものである．毎回，その回のトピックを説明し，サンプルプログラムを実演し，学生のレベルに合った演習課題（Assignment）を選択させ，その問題に取り組ませている．また授業最後には，総合課題（Final Project）を課し，上位者向けの課題の一つとして，確率的多点探索法の設計と実装問題を取り上げている．

受講者数も非常に多いため，対象者向けの特別指導等は行っていない．基本的には，各回の演習問題への取り組みを通して，学生のプログラミングに対する関心の維持・向上を図り，特に数理情報分野に関心のある学生を含む上位グループの自己効力感を高めるような方針を取っている．

4.2 演習課題の構成と工夫

前章で示したように，確率的多点探索法のアルゴリズム設計と実装では，制御構造（順次，反復，多分岐），二次元配列，乱数生成と，それらを組み合わせることができる実装スキルが必要となる．各回の演習課題への取り組みを通して，徐々に，これらを用いた複雑な構造のプログラム作成に慣れさせていく．その際，中位グループでも十分取り組めるレベル設定とする必要がある．

そのため，本授業ではデータ処理・条件抽出問題を演習課題の一部に取り入れている．通常，まとまったデータを取り込む場合，ファイル操作および一連の文字列処理が必要となるが，これらは初学習者には難しい内容である．そこで本授業では，一様乱数生成を比較的早い段階で紹介し，直接コード内で模擬データを生成する方法を取っている．詳細については著者の既報[3][4]を参照されたい．

5. 実践結果

5.1 課題の提示

確率的多点探索法の設計・実装問題は，上位者向けの総合課題の一つとして提示している．特定の進化計算手法を取り上げることはせず，その中で用いられている探索プロセスの本質的な部分を簡略化して学生には提示する．課題の解説スライドの一部を図 1 に示す．図は探索アルゴリズムの計算手順（骨子部分）を示したものである．最小化の対象問題はベンチマークとしてよく利用される関数（多峰性，10～20 変数）を選んだ．その他，対象問題や探索の様子を図で表したスライドも準備している．

5.2 実践結果

過去 4 年間分の上位者向け最終課題に取り組んだ学生数を表 1 に示す（剽窃分は除外）．なお，確率的多点探索法とは別の上位者向け課題は，金利計算とローン返済に関する問題である．例年，8 名前後の学生が，本授業を通して，確率的多点探索法の課題に取り組むまでのプログラミング能力と学習意欲を高めている．また上位者向け課題に取り組んだ学生は，例年 2 割以下であった（2020 年度は除く）．一方で，各回の演習課題の合計が高いにも関わらず，上位者向け総合課題を敬遠していた学生も一定数存在していた．これは，同時期に他の科目の試験やレポート提出があり，そちらを優先させたためと考えられる．

表 1 上位者向け課題への参加人数（2017～2019：1 年次前期，2019：1 年次後期オンライン）

	2017 年	2018 年	2019 年	2020 年
学習言語	C	Python	Python	Python
受講者数	73	83	93	68
最適化課題	8	7	8	9
上位者用別課題	4	8	7	13

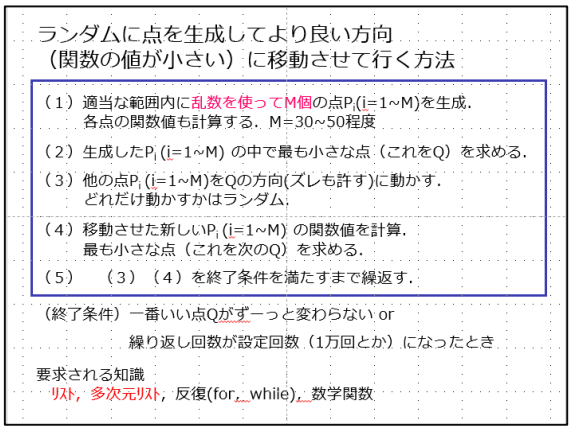

図 1 解説スライドの一部（アルゴリズムの骨子説明）

6. おわりに

本稿では，プログラミング導入教育の中で，より実践的なアルゴリズム設計を体験させる題材として，非線形関数に対する確率的多点探索法のアルゴリズム実装を取り上げ，そのための授業上の工夫について説明した．

今後は，探索アルゴリズム実装に取り組むことができた学生の学習特徴について調査を行う予定である．

参考文献

[1] Anderson,R., Ernst,M.D., Ordonez, R., Pham, P. and Wolfman, S.A, “Introductory programming meets the real world: Using real problems and data in CS1”, Proceedings of the 45th ACM Technical Symposium on Computer Science Education, 465-466 (2014)

[2] Chilana, P.K., Alcock, C., Dembla, S., Ho, A., Hurst, A., Armstrong, B., and Guo, Philip J., “Perceptions of non-CS majors in intro programming: The rise of the conversational programmer”, Proceedings of the IEEE Symposium on Visual Languages and Human-Centric Computing, 251-259 (2015).

[3] 辻 康孝, “Python によるプログラミング導入教育の実践とその学習評価”, 基幹教育紀要, Vol.5, 43-55 (2019).

[4] 辻 康孝, “学生の多様性に対応したプログラミング導入教育の実践”, FIT2019, Vol.4, 355-358 (2019).