テスト駆動学習：名前付き loop の重要性

関西学院大学・理工学部　西谷滋人，河野大登, 内田 啓太郎
Shigeto R. Nishitani, Hiroto Kohno, Keitaro Uchida, Department of Informatics,
Kwansei Gakuin University
香川大学・創造工学部，福森聡
Satoshi Fukumori, Faculty of Engineering and Design, Kagawa University

1　はじめに

プログラミングの極意は,「数学の問題を解くようにして，コードを書けばいい」とよく言われます．ところが，今時の大学生は数学の問題をそれほど解いていません．あるいは，数学は解法を暗記するものと認識しています．違ったアプローチが必要なのですが，小学生に教えるようにして，亀を走らせたとしても大学生が喜ぶわけではありません．したがって，もっと違った「例え」が必要なのですが，適切なものが見当たりませんでした．

最新のソフトウェア開発手法であるテスト駆動開発を教えていて，今まで数式処理教育で開発してきた「ペア試験」という手法が，まさにその「例え」になると気が付いたので，それを報告します．キーは「名前付き loop」です．

2　数式処理の演習内容

2.1　今までのまとめ

数式処理を 3 年生に対する実習科目として教えています．この演習内容については今までに,

2010 「パターンとペアプロの数式処理ソフト学習への適用」[1]

2011 「数式処理演習でのペアプロの効果」[2]

2012 「Maple 版ルフィの仲間たちに試練を!! – ペア評価による数式処理ソフト教育」[3]

2013 「アクティブラーニングにおけるチーム評価の導入」[4]

2018 「数式処理を sympy で教えてみた (涙)」[5]

として報告してきました．ペアプログラミングのように行う「ペア作業」とペアで相談しながら解く「ペア試験」，そしてペアで点数を全く同一にする「ペア評価」が特徴です．

一昨年度までは数式処理専用ソフトの Maple を使用していたのですが，昨年度より python の sympy を使って，Jupyter Notebook で教えるように教材を変更しました．授業内容や，試験のレベルなどは揃えたつもりなのですが，昨年度の成績は有意に低下が見られました．ところが今年の学生は頑張ってくれて，例年並みあるいはそれ以上の成績をあげてくれました．図 1 の通り，19 年度の分布はほぼ理想的で，平均点は 18 年度より 27 点，それ以前よりも 10 点ほども上がっています．python に変更したことによって，学生たちの授業に対する食いつきが大きく変わりました．AI や画像認識などで python が広範囲で使われていることを学生たちは知っていて，そのような話題に関する興味から積極的に取り組んでくれたようです．

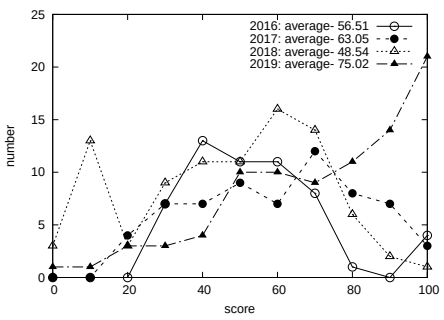


図 1: 試験の成績分布と平均.

2.2 学生さんのディスりの言い訳

ところが中には，文句をいう学生さんがいます．

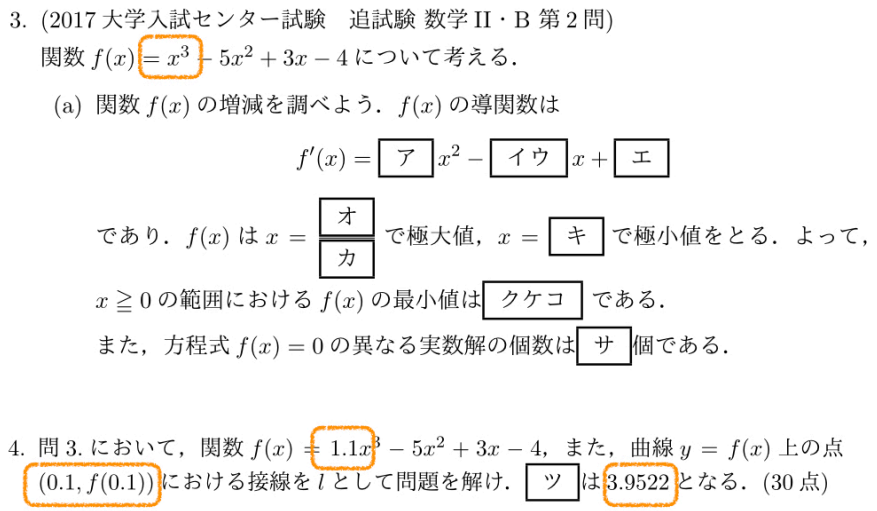
> テストの配点がおかしすぎる。
> 最後の数値のみを変えた問題が 30 点もあるところが特に。
> あのようなテストだと 70 点以下か、100 点のどちらかの点数になる。
> また、出題されるセンター試験の過去問が分析されやすい。
> あらかじめ問題の答えを作っておくなどの工作が可能になってしまう。

えっと，学生さんがディスってるとこを意図して問題を作っているので，そこに異議を唱えられても，意図通りにみなさんが行動してくれているというのがわかって，逆に喜んでしまいます．少し，問題と試験内容について説明します．

試験の内容は図 2 に示した通りです．大学入試センター試験から取ってきた問題をそのまま使っています．大学入試センター試験の問題は，数式処理を教えるのに最適です．

1. 複雑な問題解答手順の適切な誘導
2. ほぼ整数になる気持ちのいい解答
3. 詳しい解答・解説が手軽に見られる

などが数式処理で教える内容の理解を試すのに最適です．一時期，試験準備としてセンター試験の過去問を全てあらかじめ手元に用意するというのが流行ったことがあります．そこに問題の本質はないので，解答例はこちらで用意するようにしました．ところが，学生たちの中には，適当にその数値が出るようにスクリプトをでっち上げる，というのが出てきました．解答者（学生）もちゃんとできているのか，いないのか判断できないでしょうね．採点者 (私) にはわかりません．そこで，数式の数値を少し変えて解答が



3. (2017 大学入試センター試験　追試験 数学 II・B 第 2 問)
関数 $f(x) = x^3 - 5x^2 + 3x - 4$ について考える．

(a) 関数 $f(x)$ の増減を調べよう．$f(x)$ の導関数は

$$f'(x) = \boxed{ア}x^2 - \boxed{イウ}x + \boxed{エ}$$

であり，$f(x)$ は $x = \dfrac{\boxed{オ}}{\boxed{カ}}$ で極大値，$x = \boxed{キ}$ で極小値をとる．よって，

$x \geqq 0$ の範囲における $f(x)$ の最小値は $\boxed{クケコ}$ である．

また，方程式 $f(x) = 0$ の異なる実数解の個数は $\boxed{サ}$ 個である．

4. 問 3. において，関数 $f(x) = 1.1x^3 - 5x^2 + 3x - 4$，また，曲線 $y = f(x)$ 上の点 $(0.1, f(0.1))$ における接線を l として問題を解け．$\boxed{ツ}$ は 3.9522 となる．(30 点)

図 2: 試験の抜粋.

整数ではなく，浮動小数点数になるように変えました．こうすると，解答スクリプトがちゃんとできてないと意図した答えになりません．この解答の手順は我々数式処理の熟練者がやる

　　解ける問題でスクリプトを書いて，複雑な問題に修正する

というテクニックそのものになっています．ただ，点数を均等割にするといい加減なスクリプトの判定が面倒なので，数値変更前を 10 点，変更後を 30 点としています．

　試験は，学期の中間にペアで相談しながら解くというペア試験を実施しています．最終は個別の試験です．そしてそれらの点数を等分してペアの成績とする,「ペア評価」としています．そうすると，組んだペアによっては実力差があって，試験前に予想問題を立てて，その解答を準備して，相方に教え込むという輩がいるようです．教えあい，学びあいを助長している学院にとっては理想的です．その結果が先ほどのアンケートの文句となってきます．

3　loop に名前をつける意義

　多くの学生さんからは逆に集中したという反応が返ってきました．これが考察の原点なんです．つまり，なんで,

　　学生さんたちは喜んでやるんだろう

というのがわからなかったのです．その集中の理由は，相方の点数に足を引っ張られるかもという「ペア評価」の不安だけではなさそうです．python を教える時の使用環

境は Jupyter Notebook で，Maple や Mathematica と同じような面構えです．これを Interactive computing 環境と呼ぶそうですが，環境を変えただけなのに，Maple の方が圧倒的に信頼性が高いのに，なんで？

とっかかりは”loop”です．プログラミング環境での loop をピックアップして，なぜ loop が学生さんたちの琴線に触れたかの分析から始めます．

3.1 REPL read - eval - print loop

まずは loop という単語で有名なのが REPL です．語源は，lisp で書くと

```
(loop (print (eval (read))))
```

となることから，その頭文字で呼ばれています．ruby や python が用意している，irb, ipython などの interactive 環境を指します．新しいプログラミング言語を始める人にとっては，動作や用法を対話的に確認する手軽な環境です．

REPL でやっている作業を日本語にすると

入力して，実行して，出力をみる

になります．Jupyter notebook でも，その環境を提供していて，なんの下準備もいらずに，命令を打ち込んで，実行させると，すぐに綺麗なプロットで結果を返してくれます．あるいは，MathJax によって Latex 並みの数式を出してくれます．複雑な数学を相手にしているのにとても簡単で，Maple とか Mathmatica で初めて因数分解させた時の感動と同じなんですよ．この手軽さは．迅速なフィードバック loop が集中のとっかかりなんでしょう．

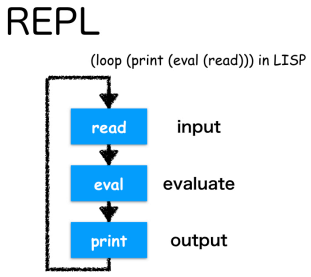


図 3: read, evaluate, print loop (REPL).

3.2 red, green, refactoring

ソフトウェア開発での有名な loop に，red, green, refactoring を合言葉にした loop があります．テスト駆動開発 (test driven development, TDD) で使われる用語です．まず，coding の最終目標を

動く綺麗なコード (clean code that works)

としています．TDD では，まず動くことを優先させ，その後綺麗にするように勧めています [6]．アーキテクチャ駆動とは取り組む順序が全く逆です．それを実践するために，それぞれの開発工程を

1. write a test (red),

2. make it run (green),

3. make it right (refactoring)

とニックネームをつけて細かく分けて指導しています．loop でしょ？

なぜ，細かく名前をつけるかというと，暗黙知を形式知に変えるためです．名前が付いてないと，どういう動作か指し示せないからです [7]．例えば，チャート式というやり方は，数学者はよく理解できると思いますが，文系出身の技術者に「チャート式で勉強会を進めましょう」と言ってもちんぷんかんぷんでしょうね．学生にゼミ形式でと言って，不思議な顔をするのも同じです．

loop に名前をつけるソフト開発でのこのような取り組みは，Design Pattern というのが契機になっています．その源流は数学科で教育を受けた建築家のクリストファー・アレグザンダーの「パターン・ランゲージ」です [8]．これが浅田彰に繋がるそうですが，それはまた別の機会に [9]．

3.3 フロー状態

では，名前付き loop だけで，学生さんたちはなぜ集中できるのでしょうか？スポーツや音楽などで,「ゾーン」というのをご存知でしょうか？バスケでは,「黒子のバスケ」以降この言葉で語られます．なに，数学者なら問題に向かっている時がゾーンなんですが，あるいは，禅の境地でしょうか．チクセントミハイがその著書で「フロー状態」と呼んで詳しく報告しています [10][11]．

> 集中した精神的，情緒的，身体的活動を通じてもたらされる，世界との完全な一体化の状態

と定義しています．そこでは，

> 集中が焦点を結び，散漫さは消滅し，時の経過と自我の感覚を失う．その代わり，われわれは行動をコントロールできているという感覚を得，世界に全面的に一体化していると感じる．

という状態にいるとしています．その状態があたかも何か大きな流れの中に漂っているという感覚からフロー状態という名前をつけたそうです．その状態に入るためには，

- 目標が明確で，
- 迅速なフィードバックがあり，
- スキル (技術) とチャレンジ (挑戦) のバランスが取れたギリギリのところで活動している

時，われわれの意識は変わり始めるとしています．さらに活動自身に創造性や価値があることも大切なようです．

これって，数学の問題を解いている時だと思いませんか？TDD とか coding の最中はこのようになってほしいのですが，そんな状態に今時の学生さんはなるんでしょうか？きっと，ゲーム中の状態はそうなのかもしれません．実際に体験している以上，他の活動においても，その状態へ行くことができるはずです．その loop に価値を見出せば．

3.4 abduction loop

こういう loop は他にもたくさんあります．文章術の指南書でもよく似た loop を紹介しています．図 4 は世界的コンサル会社で文章術を指南していたバーバラ・ミントの第 2 版から付録に載ってきた絵です [12]．どこから問題を start するかによって，用いるべき考え方が決まるとした図です．演繹法，帰納法というよく知られた論理法を

演繹法は 「ルール」があって,「ケース」があれば「結果」は決まる．

帰納法は 「ケース」があって,「結果」があれば,「ルール」が求められる．

とわかりやすくまとめています．さらに，不明推測法 (abduction) というあまり知られていない論理法を解説しています．曰く，

	例	形式
結果	売上が低下した	A という予期しなかった事実を観察する．
ルール	売上が低下する理由の一つは 価格が高すぎるから	B が本当なら，A となるかもしれない．
ケース	実際に価格が高すぎるか チェックしてみよう	もし B が本当ならば，C が当然起こる． C が本当に起こるか確かめよう．

という，予期せぬ「結果」があって，それを引き起こす「ルール」を見つけるために,「ケース」を想定して，新たな「結果」を確かめて行くという loop を紹介しています．

ミントはこの不明推測法の，ルールを「作業仮説」，ケースを「実験」と読み替えて，科学的手法の特質を「仮説を創り出し実験を考案する」ことにあるとしています．米盛はもう少し詳しく科学的手法との比較を掘り下げています [13]．どう分類するかは難しそうですが，次々と仮説を組み立てる loop が科学的手法の本質であることは間違いなさそうです．

3.5 IMRAD loop

そうしてみると理系の論文作成にも有名な loop があります．IMRAD というのをご存知でしょうか？論文のフォーマットをまとめて Introduction, Method, Results and Discussions とすることから命名されています．まあ，米国の研究者でもこの名称では，意外と知られてないという報告があります [14]．

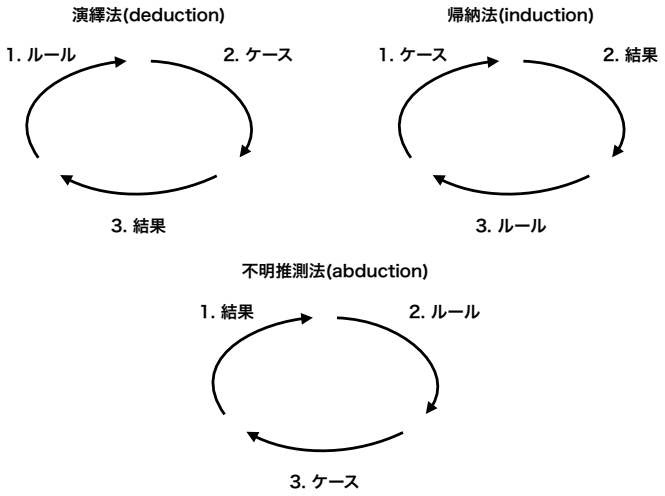


図 4: abduction loop.

これも，図 5 に示したように，先ほどの不明推測法と同じ loop になっています．loop というよりもステップですね．でも，我々研究者はそれを何度も何度も繰り返して，学問しています．一方で，学生さんたちは，教育の最終段階で行う卒業研究で一度だけすればいいと思い込んでいます．う〜〜ん，会社に入ったら「ほうれんそう (報告，連絡，相談)」として，一生繰り返す作業なんですが...

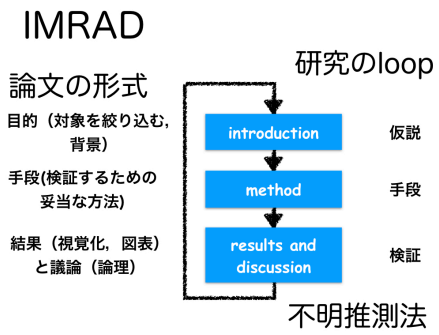


図 5: IMRAD loop.

3.6 テスト駆動学習

研究にも価値を見出して，フロー状態で loop に没入して欲しいのですが，数式処理の学習の分析に戻りましょう．数式処理のスキル習得としてペア試験で，やっていることを再度見直すと，

- 手計算ができる，解答がある（目標が明確）
- input に即座に output が返る（迅速なフィードバック，loop）
- それほど簡単な問題ではない（挑戦と能力のバランス）

と，フロー状態に入るのに必要な条件を満たしています．さらに必要とされる価値と創造性をみると，

1. センター試験（高校時代での価値）
2. python である（AI なんかからの価値）

3. 手計算ではしんどい部分の自動化 (創造性)

です．Maple では 2 番目の価値を説得するのに言葉だけになってしまいますが，python では社会的に十分な説得力があるのでしょうね．

さらに形式知にするために，この作業あるいは形態を名前付きにする必要があります．ソフト開発に倣って，数式処理で行なっている学習方法を「テスト駆動学習」と今は名付けて，学生に教えています．べったベタやね．いいのがあったら教えてください．

4 まとめ

情報科学科の数式処理演習で行っているテスト駆動学習について，手法と意義をまとめました．ペア学習とか，ペア試験，ペア評価よりさらに進んで，知識習得の loop を回すために必要な工夫です．この学習法は，ソフトウェア開発で使われている，テスト駆動開発や REPL と同じように，作業 loop に没入することを目標にしています．loop の手順の明確化と，名前付きの大切さを認識して，今後も布教していきます．さらには，この学習 loop を体験した学生さんたちが,「適切な例え」として，coding や卒論，研究に適用してくれることを願います．

参考文献

[1] 西谷滋人, 廣岡愛未. ”パターンとペアプロの数式処理ソフト学習への適用”. 清水克彦, 高遠節夫（編）, 数学ソフトウェアと教育－数学ソフトウェアの効果的利用に関する研究, 京都大学数理解析研究所講究録 (ISSN 1880-2818), No. 1735, pp. 127–39, Apr. 2011.

[2] 西谷滋人. ”数式処理教育でのペアプロの効果”. 清水克彦, 高遠節夫（編）, 数学ソフトウェアと教育－数学ソフトウェアの効果的利用に関する研究, 京都大学数理解析研究所講究録 (ISSN 1880-2818), No. 1780, pp. 40–9, Apr. 2012.

[3] 西谷滋人. Maple 版ルフィの仲間たちに試練を!! –ペア評価による数式処理ソフト教育–. 清水克彦, 中村泰之（編）, 数学ソフトウェアと教育－数学ソフトウェアの効果的利用に関する研究, 京都大学数理解析研究所講究録 (ISSN 1880-2818), No. 1865, pp. 62–71, Nov. 2013.

[4] 西谷滋人. アクティブラーニングにおけるチーム評価の導入. 数学ソフトウェアとその効果的教育利用に関する研究, 数理解析研究所講究録 (ISSN 1880-2818), No. 1909, pp. 223–32. 数理解析研究所, 2014.

[5] 西谷滋人. ”数式処理を sympy で教えてみた (涙)”. 中村泰之, 金子真隆（編）, 数学ソフトウェアとその効果的教育利用に関する研究, 京都大学数理解析研究所講究録 (ISSN 1880-2818), No. 2105, pp. 52–58, Feb. 2019.

[6] Kent Beck. テスト駆動開発. オーム社; 新訳版, 2017.

[7] まつもとゆきひろ. 語彙と共通言語の重要性. まつもとゆきひろコードの世界. 日経 BP, 2009.

[8] クリストファー・アレグザンダー. パタン・ランゲージ – 環境設計の手引. 鹿島出版会, 1984.

[9] 加藤弘一. 『パターン、wiki、xp ～時を超えた創造の原則～』 江渡浩一郎 (技術評論社), 書評空間::紀伊國屋書店 kinokuniya::booklog. https://booklog.kinokuniya.co.jp/kato/archives/2010/07/post_206.html.

[10] M. チクセントミハイ. フロー体験 喜びの現象学. 世界思想社, 1996.

[11] M. チクセントミハイ. フロー体験入門 – 楽しみと創造の心理学. 世界思想社, 2010.

[12] バーバラミント. 考える技術・書く技術 – 問題解決力を伸ばすピラミッド原則. ダイヤモンド社, 1999.

[13] 米盛裕二. アブダクション – 仮説と発見の論理. 勁草書房, 2007.

[14] ランディ・オルソン. なぜ科学はストーリーを必要としているのか – ハリウッドに学んだ伝える技術. 慶應義塾大学出版会, 2018.