# 2025年度計算物理学B 中間レポート問題

2025年11月18日出題

**諸連絡:**

1. 以下の九つの課題のうち、アルゴリズム部門とコーディング部門からそれぞれ少なくとも二つずつを選んで、合計四つ解答してください。
2. レポートは学務情報システムから提出してください。
3. アルゴリズム部門の問題は PDF ファイルで提出してください。手書きの人は、紙に書いたのを携帯/デジカメで読めるように撮るかスキャナを使うかして画像で送るかしてください。画像ファイルは一頁一ファイルでなく、ひとつのファイルにまとめてください。やりかたが分からない人は、Google で “jpg を pdf に変換” で調べてみてください。
4. 手書きの人は、書き殴るのでなくて、答案を読む教員と TA の方を哀れんで、なるべく綺麗な字で書いてください。
5. コーディング部門の問題は、毎回の実習と同様.ipynb 形式でアップロードしてください。ローカルの環境を使用しても構いませんが、課題提出の際は、仮想環境の requirements.txt を一緒に提出して下さい。
6. 締切は**2025/12/9(火)、日本時間23時59分まで**とします。
7. 問題文の意味をなさないところ、おかしいと思うところがあったら、適宜正しいと思うように修正して解答してください。野垣か藤本にメールで連絡してくれたら、この問題も修正します。
8. 生成 AI の使用を許可します。ただし、出力をそのまま貼り付けるのではなく、自分で内容を咀嚼、取捨選択すること。また、生成 AI を用いてレポート・コードを書いた場合は、その旨を記載すること。採点の際、生成 AI の使用による不利益はありません。

# アルゴリズム部門

## 課題 1：マシンイプシロン

第6回目の授業で、計算機での浮動小数点の表現を解説した。特に、有限の桁で打ち切って計算機に保持する都合上、ほとんどすべての実数を正確に表現することはできず、本来稠密である実数とは対照的に、離散的に分布する性質を持つ。上記の離散性に関して、仮数部の最小の桁(一番右の桁)を反転させると、有限の大きさの変化が起こる。この変化の大きさについて考察しよう。まず、小数 1.0 を二進数表示すると、$(0.1)_2 \times 2^1$ である。 1.0 の64 bit浮動小数点表示において、一番右のbitを反転させると、計算機上で最も 1.0 に近い小数 $1.0 + \varepsilon$ が得られるが、この $\varepsilon$ を2の冪乗を用いた形と10の冪乗を用いた形についてそれぞれ導出せよ。 $\varepsilon$ は、計算機で表現できる最小の「相対誤差」であり、マシンイプシロンと呼ばれている。問題を解く際には、64bit 浮動小数点の仮数部は52 bitであることを用いよ。また、仮数部 $(0.b_{-1}b_{-2}b_{-3}\cdots)_2$ において、$b_{-1}$ は必ず 1 であるため、 $b_{-1}$ は保持せず、$b_{-2}$ 以降のデータを保持することに注意せよ (けち表現)。 Pythonでは、以下のコードでマシンイプシロンを確認できるので、検算に用いるとよい。

```
import sys
print(sys.float_info.epsilon)
```

## 課題 2： Kahan のアルゴリズム

授業において、0.01 を 10000000 回足し上げると、$0.01 \times 10000000 = 100000$ からずれることを紹介した。これは、大きさの異なる 2 つの実数の和において、桁落ちが発生するためであった。この桁落ちを回避して、精度良く総和を計算するアルゴリズムが知られており、Kahanのアルゴリズムと呼ばれている。 Kahanのアルゴリズムについて調べ、その仕組を解説せよ。

## 課題 3：高次の差分公式

授業で数値微分として、関数 $f(x)$ の以下の 2 点前方差分、中心差分の公式を紹介した。

$$f'(x_0) = \frac{f(x_1) - f(x_0)}{h} + \mathcal{O}(h)\,, \qquad f'(x_0) = \frac{f(x_1) - f(x_{-1})}{2h} + \mathcal{O}(h^2)\,. \tag{3.1}$$

ただし、$x_n = x_0 + nh$ である。式 (3.1) の1つ目は

$$\frac{f(x_1) - f(x_0)}{h} = f'(x_0) + \frac{1}{2}f''(x_0)h + \cdots \tag{3.2}$$

という Taylor 展開の表式からも分かるとおり $\mathcal{O}(h)$ に余計な項が出てくる。つまり、 $f'(x_0)$ の表式としては $\mathcal{O}(h)$ の精度まで正しいということである。

関数 $f(x_n)$ の $x_0$ 周りでの Taylor 展開を元に、余計な項を消去して、以下の高次精度の数値微分公式を導出せよ。

$$f'(x_0) = \frac{-\frac{1}{2}f(x_2) + 2f(x_1) - \frac{3}{2}f(x_0)}{h} + \mathcal{O}(h^2)\,, \tag{3.3}$$

$$f'(x_0) = \frac{-\frac{1}{12}f(x_2) + \frac{2}{3}f(x_1) - \frac{2}{3}f(x_{-1}) + \frac{1}{12}f(x_{-2})}{h} + \mathcal{O}(h^4)\,. \tag{3.4}$$

暇な人は、上の公式をさらに高次の精度や、2階以上の微分の場合についても調べてみよ。

## 課題 4： Gauss-Legendre 求積法

講義で紹介した Gauss-Legendre の求積法について考えよう。この方法は、被積分関数を直交多項式を用いて、全区間を単一の近似式で補間する方法である。特に、この方法では直交多項式として Legendre 多項式 $P_n(x)$ を用いる：

$$P_n(x) = \frac{1}{2^n n!} \frac{d^n}{dx^n} \left[ (x^2 - 1)^n \right] . \tag{4.1}$$

Gauss-Legendre の求積法は以下の公式として与えられる：

$$\int_{-1}^{1} dx\, f(x) \approx \sum_{i=1}^{n} w_i f(x_i) . \tag{4.2}$$

ただし、$n$ は積分点の数、積分点 $x_i$ は $n$ 次の Legendre 多項式 $P_n(x)$ の $i$ 番目の零点 ($P_n(x_i) = 0$ となる点のこと) であり、重み $w_i$ は以下の式で与えられる：

$$w_i = \frac{2}{(1 - x_i^2) \left[ P_n'(x_i) \right]^2} . \tag{4.3}$$

ここでは、被積分関数として、$(2n - 1)$ 次の多項式関数を考えよう。

$$f(x) = \sum_{k=0}^{2n-1} a_k x^k . \tag{4.4}$$

この多項式は $n$ 次の Legendre 多項式を用いて、

$$f(x) = P_n(x) g_{n-1}(x) + r_{n-1}(x) , \tag{4.5}$$

と書ける。ただし、$g_{n-1}(x)$ と $r_{n-1}(x)$ はそれぞれ $n - 1$ 次の多項式関数である。

このとき、$f(x)$ に対して $n$ 点の厳密な求積を可能とするような積分点 $x_i$ の満たすべき条件 $P_n(x_i) = 0$ を導出せよ。ただし、Legendre 多項式の直交性 $\int_{-1}^{1} P_n(x) P_m(x) dx = [2/(2n + 1)] \delta_{nm}$ は既知のものとして良い。

さらに余裕のある人は、重み (4.3) を導出してみよう。

# コーディング部門

## 課題 5：整数問題

以下の2025年京大理系数学の問題をプログラミングで解いてみよう。

正の整数$x, y, z$を用いて、$N = 9z^2 = x^6 + y^4$で表される正の整数 $N$ の最小値を求めよ。

ただし、$0 < x \leq 20,\ 0 < y \leq 50$の範囲には、解が存在するとして良い。また、必要であれば、整数 $n$ に対して $\sqrt{n}$ 以下の最大整数を求めるPythonの関数 math.isqrt を用いて良い。その際には、import math を実行する必要がある。

## 課題 6：漸化式の解法

漸化式

$$3x_{i+2} + 14x_{i+1} - 5x_i = 0 \tag{6.1}$$

は、解析解 $x_i = (1/3)^i$ を持つ。数値的に漸化式を解いて得られる数列の振る舞いを調べよう。

(1) 初期値 $x_0 = 1$、$x_1 = 1/3$ とし、漸化式を数値的に解いてみよう。数値解と解析解 $x_i = (1/3)^i$ を比較すると、項数が進むにつれて、数値解の精度が極めて低くなることを確認せよ。

実は、上記の漸化式は $x_i = (-5)^i$ という解も持つため、その一般解は、$x_i = A(1/3)^i + B(-5)^i$ である。初期値 $x_0 = 1$、$x_1 = 1/3$ の条件から $A = 1$、$B = 0$ となるが、浮動小数点の精度の問題により、計算機では $B = 0$ が完全に満たされない。項数を進むにつれて、誤差が $-5$ 倍されて増大し、最終的に全く出鱈目な解が得られる（誤差の爆発）。
この問題への対処法として、漸化式を逆進する方法が知られる。

(2) $n = 30$ 程度に設定して、$y_{n+1} = 0$、$y_n = 1$ の下で漸化式を逆向きに解け。数値的に得られた数列 $y_{n+1}, y_n, \cdots y_0$ を $a = x_0/y_0$ 倍することで、$x_0 = 1$ の初期条件を満たす解を構成すると、$x_i = (1/3)^i$ を十分精度良く再現することを確認せよ。

暇な人は、$y_{n+1} = 0$、$y_n = 1$ について、手計算で$A, B$ の値を求め、全体を$a$ 倍し、逆進した場合の$x_i$の表式を求めよ。極限 $n \to \infty$ で $x_i = (1/3)^i$ を再現することを確認せよ。ただし、同じノートブックのテキストブロックに解答せよ。

以上は、漸化式の解として、減少する成分と増大する成分が共存する場合に、精度良く数値解を得るために必要な手続きとなっている。減少する成分を得たい場合には逆進、増大する成分を得たい場合には前進解法を用いるべきである。つまり、計算機には誤差がつきものであるが、誤差を減少させる方向に計算を進めると良い。これらの技法は、ベッセル関数の計算等に用いられている。

## 課題 7： Richardson 加速

円周率を与える次の無限級数を計算しよう[1]。

$$\frac{\pi^2}{6} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \cdots = \sum_{k=1}^{\infty} \frac{1}{k^2} = 1.6449340668482264\ldots \tag{7.1}$$

級数の部分和を $S_n = \sum_{k=1}^{n} \frac{1}{k^2}$ とおき、級数の極限を $S_\infty = \lim_{n\to\infty} S_n$ とする。

(1) まずは、単純にこの級数の部分和を数値的に計算せよ。具体的には $n$ を 1 から $10^7$ 程度まで変化させて、収束の遅さを確認せよ。マシンイプシロンの問題があるので、級数の和をとる際には $k = 1$ からではなく、逆順に $k = n$ から 1 まで足し上げるよう注意せよ。

今、$S_n = S_\infty + c/n + \mathcal{O}(n^{-2})$ ($c$ は定数) であることが示せる。同様の関係式 $S_{n+1} = S_\infty + c/(n+1) + \mathcal{O}(n^{-2})$ から $c$ を消去すると、$S_\infty = (n+1)S_{n+1} - nS_n + \mathcal{O}(n^{-2})$ となる。ここから、新しい級数

$$S_n^{(1)} = (n+1)S_{n+1}^{(0)} - nS_n^{(0)} \tag{7.2}$$

[1]現代では、円周率の計算には Chudnovsky 級数が用いられるようである：

$$\frac{1}{\pi} = 12\sum_{n=0}^{\infty} \frac{(-1)^n (6n)!}{(3n)!(n!)^3} \frac{13591409 + 545140134n}{640320^{3n+3/2}}.$$

が定義できる。ただし、元の級数 $S_n$ を $S_n^{(0)}$ と表した。新しい級数について $S_n^{(1)} = S_\infty + c^{(1)}/n^2 + \mathcal{O}(n^{-3})$ ($c^{(1)}$ は定数) であることが示せて、$S_{n+1}^{(1)} = S_\infty + c^{(1)}/(n+1)^2 + \mathcal{O}(n^{-3})$ から $c^{(1)}$ を消去すると、さらに新しい級数

$$S_n^{(2)} = \frac{\left(1+\frac{1}{n}\right)^2 S_{n+1}^{(1)} - S_n^{(1)}}{\left(1+\frac{1}{n}\right)^2 - 1} \tag{7.3}$$

が定義できる。同様の操作を $m$ 回繰り返し、数式 $S_n^{(m)}$ が得られる。

$$S_n^{(m)} = \frac{\left(1+\frac{1}{n}\right)^m S_{n+1}^{(m-1)} - S_n^{(m-1)}}{\left(1+\frac{1}{n}\right)^m - 1} \tag{7.4}$$

このように、$\mathcal{O}(n^{-1}), \mathcal{O}(n^{-2}), \ldots, \mathcal{O}(n^{-m})$ の誤差を逐次相殺させてゆき、$S_n^{(m)}$ を定義することで、極限への収束が加速される (Richardson 加速)。

(2) $S_n^{(m)}$ を数値的に計算し、元の級数 $S_n^{(0)}$ と比較して収束が加速されていることを確認せよ。

## 課題 8： Romberg 積分

数値積分を級数としてみたとき、課題 7 で扱った Richardson 加速を適用することができる。これを **Romberg 積分**という。講義で見た通り、適応求積法の $n$ ステップ目での積分区間の分割の幅を $h_n$ とした、合成台形則の数値積分 $I_{\rm trap}^n$ を考えよう。 $n+1$ ステップ目の幅を $h_{n+1} = h_n/2$ とする。このとき、$n+1$ ステップ目の台形則の数値積分 $I_{\rm trap}^{n+1}$ の誤差 $\epsilon_{n+1}$ は、$\epsilon_{n+1} \simeq (I_{\rm trap}^{n+1} - I_{\rm trap}^n)/3 + \mathcal{O}($ と表される。したがって、$n+1$ ステップ目の台形則の数値積分の値自体は $I_{\rm trap}^{n+1} + (I_{\rm trap}^{n+1} - I_{\rm trap}^n)/3$ と表される。ここから、新しい級数

$$R_n^{(1)} = R_{n+1}^{(0)} + \frac{R_{n+1}^{(0)} - R_n^{(0)}}{3}, \tag{8.1}$$

が定義できる。ただし、元の台形則の数値積分を $R_n^{(0)} \equiv I_{\rm trap}^n$ と再定義した。これはまさに課題 7 の式 (7.2) そのものである。同様の操作を $m$ 回繰り返すと、

$$R_n^{(m)} = \frac{4^m R_{n+1}^{(m-1)} - R_n^{(m-1)}}{4^m - 1}. \tag{8.2}$$

このように、得られた $R_n^{(m)}$ は少ない回数の被積分関数の評価によって高精度の結果を与える。

以上の Romberg 積分のコードを実装し、適当な被積分関数に適用してみよ。

## 課題 9：自由課題

そのほかにも、講義で気になったことや、発展的な内容について自分で調べた結果を PDF にまとめたり、コードを実装したりしてもよい。