電気電子計算工学及演習 課題 3

三軒家 佑將（さんげんや ゆうすけ）
3 回生
1026-26-5817
a0146089

1　プログラムの説明

1.1　概要

本レポートにおいては、プログラム言語として Ruby を採用した。

プログラムを実行する手順は、以下のとおりである。以下の手順に従うことで、課題 3.1, 3.2, 3.3, 3.4 の 4 つ全てに関して、結果をグラフにした画像が graphs ディレクトリ以下に出力される。また、各数値解法における $p - \log_2 E_r$ のグラフの傾きが、標準出力に表示される。

```
?> cd src
?> bundle install --path vendor/bin
?> bundle exec ruby main.rb 2> /dev/null
```

二行目で、依存ライブラリのインストールを行っている。また、三行目は、プログラムを実行するコマンドである。エラー出力を/dev/null にリダイレクトしているのは、線形回帰に用いたライブラリの警告メッセージを表示しないためである。リダイレクトを行わなくても、プログラムは問題なく実行される。

1.2　各機能・関数の説明

プログラムを作成するにあたって、見通しを良くするために、プログラムを複数のファイルに分割している。ここでは、各ファイルごとに、そのファイルの担う機能と、そのファイル内にある関数の機能などについて簡単に説明する。

各関数の詳しい使用方法などは、プログラム内のコメントにて示したので、そちらも参照されたい。

calculation.rb

各課題の数値計算を行なう部分のうち、共通する部分を切り出したものである。calculate 関数と all_calculations 関数を含む。

calculate 関数は、渡された各種パラメーターと、渡されたブロックで表されたアルゴリズムに基づいて、数値計算を行なう。

all_calculations 関数は、渡された各種パラメーターと、渡されたブロックで表されたアルゴリズムに基づいて、calculate 関数を内部で複数回呼び出し、課題 3.1 と 3.2 に示された各種数値計算を行なう。

vector.rb

一次元のベクトルを表す MyVector クラスを定義している。

MyVector クラスは、Ruby の組み込みクラスである Array クラスを継承して定義した。Array クラスの機能に加えて、ベクトル間の加算 (+)・減算 (-) と、ベクトル-スカラー間の乗算 (*)・除算 (/) を定義している。また、MyVector クラスには、ベクトルの大きさ (二乗和平方根) を求める norm メソッドと、要素の合計を求める sum メソッドを定義した。さらに、MyVector クラスのインスタンスを簡単に生成するために、Array クラスに、to_v メソッドを追加した。

plot.rb

グラフを描画し、ファイルに出力する機能を担う。gnuplot のラッパーを利用している。

draw_graphs 関数に各種パラメーターを渡すことで、graphs ディレクトリ以下にグラフの画像が出力される。save_graphs 関数は、draw_graphs 関数に呼び出され、実際にグラフを出力する処理を行なう。

least_square.rb

線形回帰を行って、一次関数の係数を求める機能を担う。statsample というライブラリを利用している。

least_square 関数が定義されており、x の配列と y の配列を与えると、その 2 つのデータの間に $y = a + bx$ の関係があると考え、a と b の値を求める。

main.rb

上記で述べた関数を利用して、実際にオイラー法・ホイン法・四次のルンゲ-クッタ法にて、微分方程式の数値解を求める。

関数 f は、与えられた微分方程式を関数にしたものである。すなわち、

$$\frac{d\mathbf{x}}{dt} = \mathbf{f} \tag{1}$$

である。

2 課題 3.1

与えられた微分方程式

$$\begin{aligned}\frac{d\mathbf{r}}{dt} &= \mathbf{v} \\ &= \begin{pmatrix} v_1 \\ v_2 \end{pmatrix} \\ \frac{d\mathbf{v}}{dt} &= \frac{q}{m}(\mathbf{v} \times \mathbf{B}) \\ &= \begin{pmatrix} v_2 \\ -v_1 \end{pmatrix}\end{aligned}$$

を、$\mathbf{x} = \begin{pmatrix} r_1 & r_2 & v_1 & v_2 \end{pmatrix}^T$ に対する微分方程式

$$\frac{d\mathbf{x}}{dt} = \mathbf{f}(\mathbf{x}) = \begin{pmatrix} v_1 \\ v_2 \\ v_2 \\ -v_1 \end{pmatrix} \tag{2}$$

と考え、この数値解を、オイラー法を用いて求めた。また、$\mathbf{x}$ の初期値は、$\mathbf{x}_0 = \begin{pmatrix} -1 & 0 & 0 & 1 \end{pmatrix}^T$ とした。

ただし、数値解を求める範囲は $0 \leq t \leq 6.4 \times 5$ とし、微小時間は $\mathrm{dt} = 0.1$ 秒とした。これは、課題 3.3, 3.4 においても同様である。

2.1 オイラー法

オイラー法は、以下の手順で $\mathbf{x}_n$ を順次求める手法である。

1. $\mathbf{k} = \mathbf{f}(\mathbf{x}_{n-1})$
2. $\mathbf{x}_n = \mathbf{x}_{n-1} + \mathbf{k} \cdot \mathrm{dt}$

ただし、$\mathbf{f}$ は式 (2) による (課題 3.3, 3.4 でも同様)。

2.2 結果

図 1 は、式 (2) の解析解と数値解のそれぞれについて、$\mathbf{r}$ の軌跡をプロットしたものである。時間の経過に従って、数値解の差が大きくなっていることがわかる。

図 2 は、式 (2) の解析解と数値解の誤差を、時間に沿ってプロットしたものである。時間の経過に従って、誤差が二次関数的に増加しているように見える。

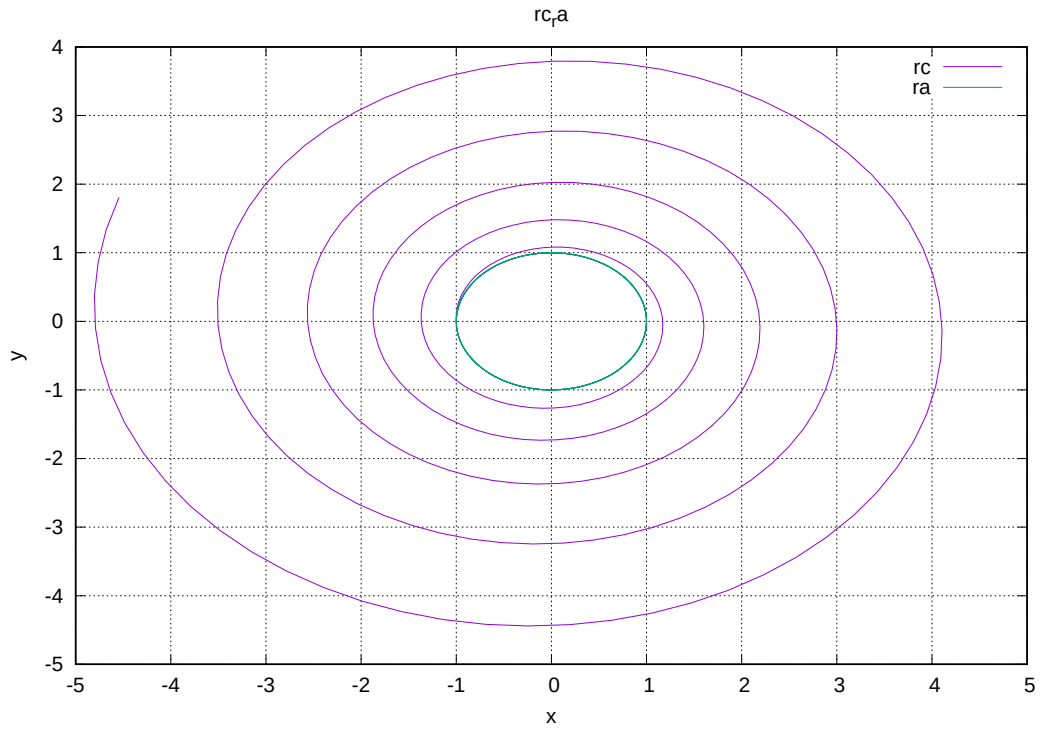

図 1　解析解と数値解の比較

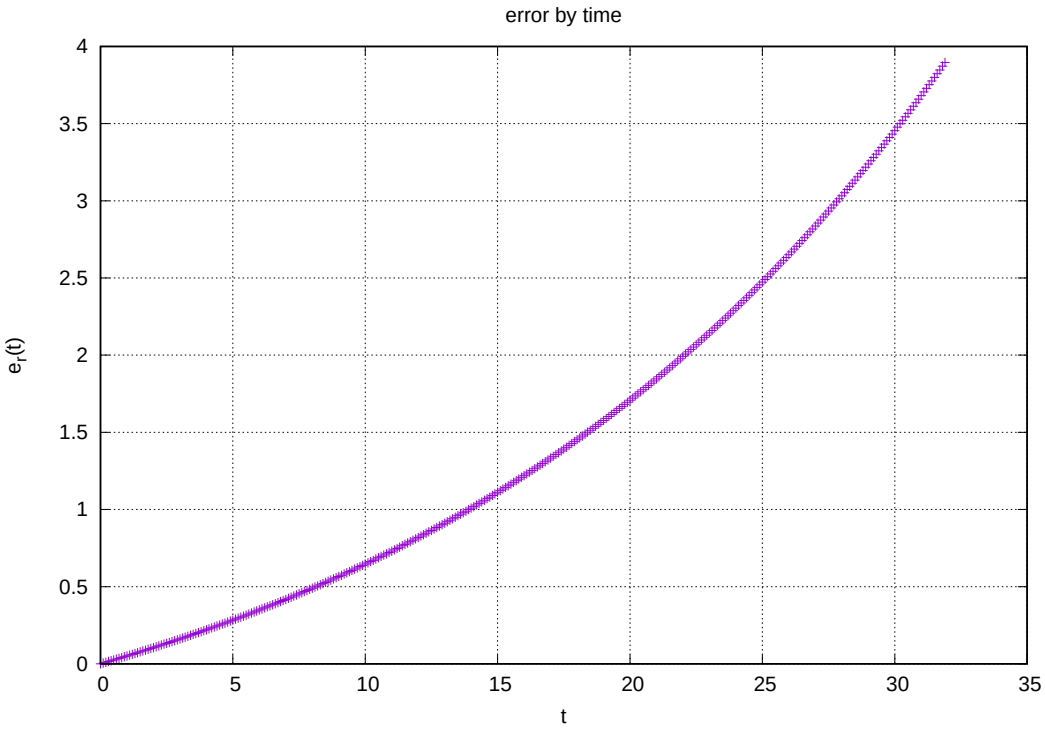

図 2　誤差の時間発展

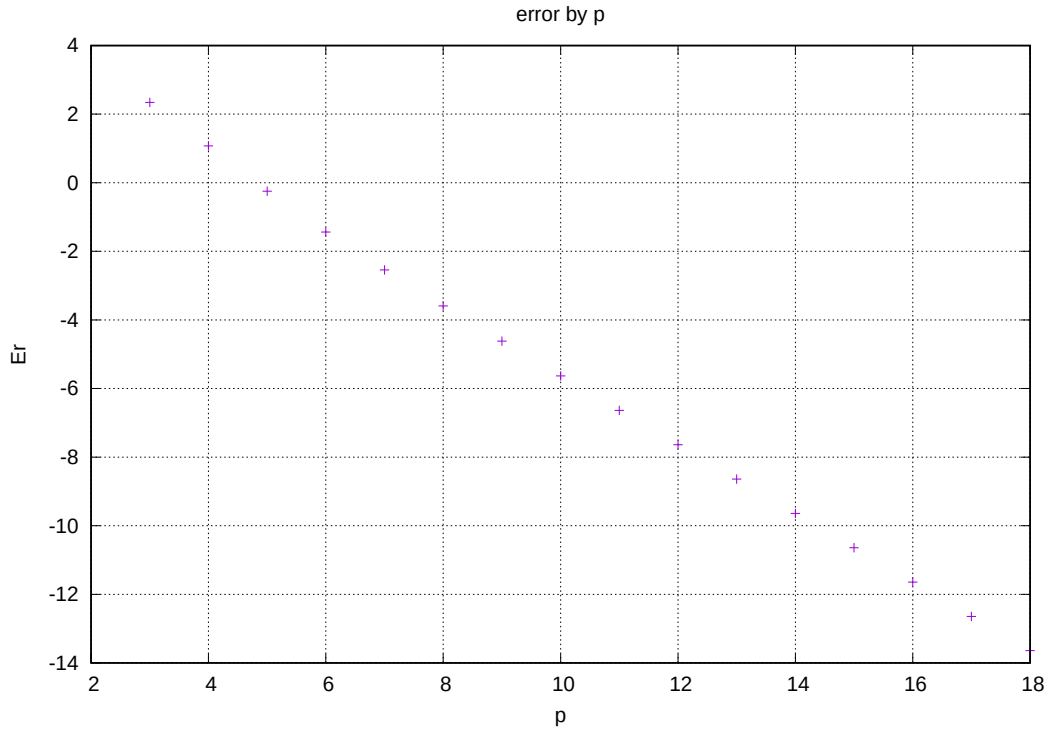

図 3　微小時間の大きさに対する誤差の大きさ

3　課題 3.2

$\mathrm{dt} = 6.4 * 2^{-p} (p = 3, 4, .., 18)$ として、$0 \leq t \leq 6.4$ の範囲でオイラー法による数値解を計算し、各 p に対して最大誤差 $E_r = \max |e_r(t)|$ を求めた。

3.1　結果

図 3 は、各 p に対する $\log_2 E_r$ をグラフにプロットしたものである。p の増加に伴い、一次関数的に最大誤差が減少していることがわかる。

このグラフの傾きは、標準出力の表示によると、

$$b = -1.0461609925464084$$

であった。

4　課題 3.3

式 (2) の数値解を、ホイン法を用いて計算した。また、課題 3.2 と同様に、各 p に対して最大誤差 $E_r = \max |e_r(t)|$ を求めた。

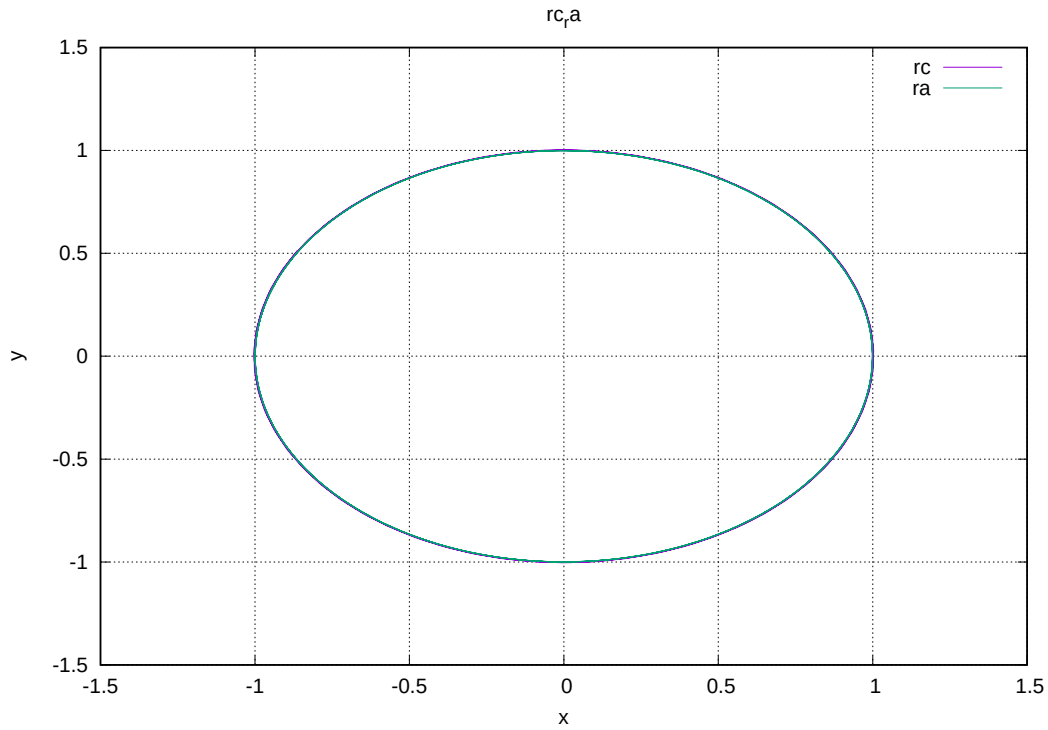

図 4　解析解と数値解の比較

4.1　ホイン法

ホイン法は、以下の手順で $\mathbf{x}_n$ を順次求める手法である。

1. $\mathbf{k}_1 = \mathbf{f}(\mathbf{x}_{n-1})$
2. $\mathbf{k}_2 = \mathbf{f}(\mathbf{x}_{n-1} + \mathbf{k}_1 \cdot \mathrm{dt})$
3. $\mathbf{k} = \frac{\mathbf{k}_1 + \mathbf{k}_2}{2}$
4. $\mathbf{x}_n = \mathbf{x}_{n-1} + \mathbf{k} \cdot \mathrm{dt}$

4.2　結果

図 4 は、式 (2) の解析解と数値解のそれぞれについて、$\mathbf{r}$ の軌跡をプロットしたものである。目視では違いが見られないほど、高い精度で数値解が求められていることがわかる。

図 5 は、式 (2) の解析解と数値解の誤差を、時間に沿ってプロットしたものである。時間の経過に従って、誤差が一次関数的に増加していることがわかる。

図 6 は、各 p に対する $\log_2 E_r$ をグラフにプロットしたものである。p の増加に伴い、一次関数的に最大誤差が減少していることがわかる。

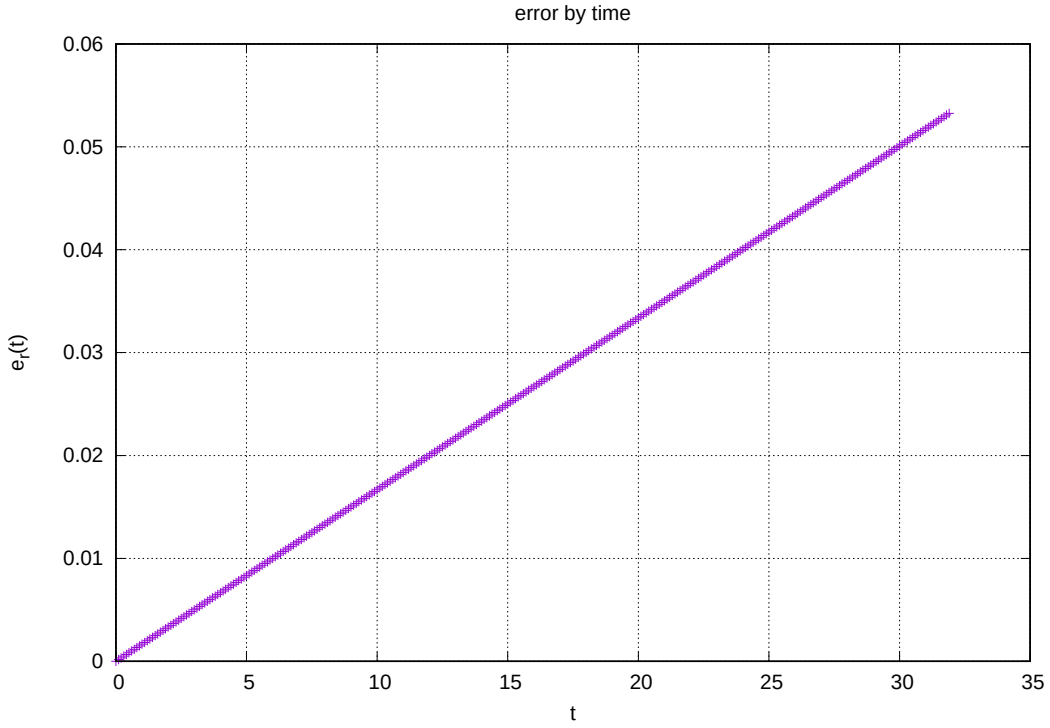

図 5　誤差の時間発展

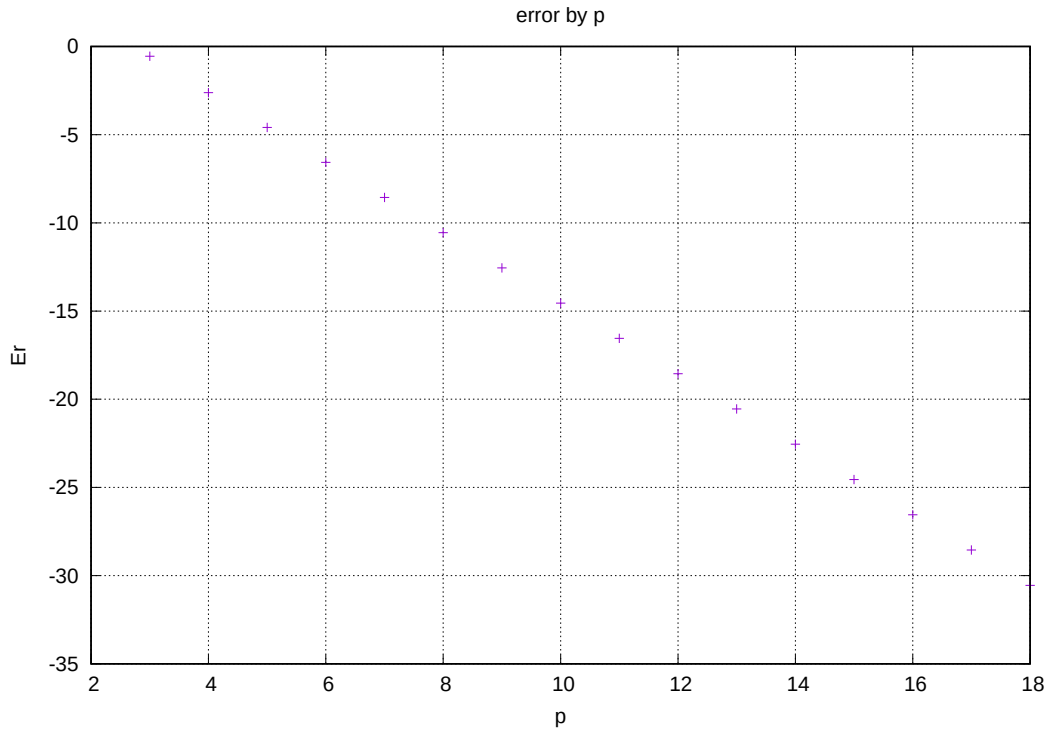

図 6　微小時間の大きさに対する誤差の大きさ

このグラフの傾きは、標準出力の表示によると、

$$b = -1.9974708704503414$$

であった。

5 課題 3.4

式 (2) の数値解を、4 次のルンゲ-クッタ法を用いて計算した。また、課題 3.2 と同様に、各 p に対して最大誤差 $E_r = \max|e_r(t)|$ を求めた。

5.1 4 次のルンゲ-クッタ法

4 次のルンゲ-クッタ法は、以下の手順で $\mathbf{x}_n$ を順次求める手法である。

1. $\mathbf{k}_1 = \mathbf{f}(\mathbf{x}_{n-1})$
2. $\mathbf{k}_2 = \mathbf{f}(\mathbf{x}_{n-1} + \mathbf{k}_1 \cdot (\mathrm{dt}/2))$
3. $\mathbf{k}_3 = \mathbf{f}(\mathbf{x}_{n-1} + \mathbf{k}_2 \cdot (\mathrm{dt}/2))$
4. $\mathbf{k}_4 = \mathbf{f}(\mathbf{x}_{n-1} + \mathbf{k}_3 \cdot \mathrm{dt})$
5. $\mathbf{k} = \frac{\mathbf{k}_1 + 2\cdot\mathbf{k}_2 + 2\cdot\mathbf{k}_3 + \mathbf{k}_4}{6}$
6. $\mathbf{x}_n = \mathbf{x}_{n-1} + \mathbf{k} \cdot \mathrm{dt}$

5.2 結果

図 7 は、式 (2) の解析解と数値解のそれぞれについて、$\mathbf{r}$ の軌跡をプロットしたものである。目視では違いが見られないほど、高い精度で数値解が求められていることがわかる。

図 8 は、式 (2) の解析解と数値解の誤差を、時間に沿ってプロットしたものである。時間の経過に従って、誤差が一次関数的に増加していることがわかる。

図 9 は、各 p に対する $\log_2 E_r$ をグラフにプロットしたものである。p の増加に伴い、途中までは一次関数的に最大誤差が減少しているが、$p = 13$ 周辺から最大誤差の減少が止まり、$E_r = -48$ 程度で頭打ちになったことがわかる。

このグラフの傾きは、標準出力の表示によると、

$$b = -3.9836971175101317$$

であった。

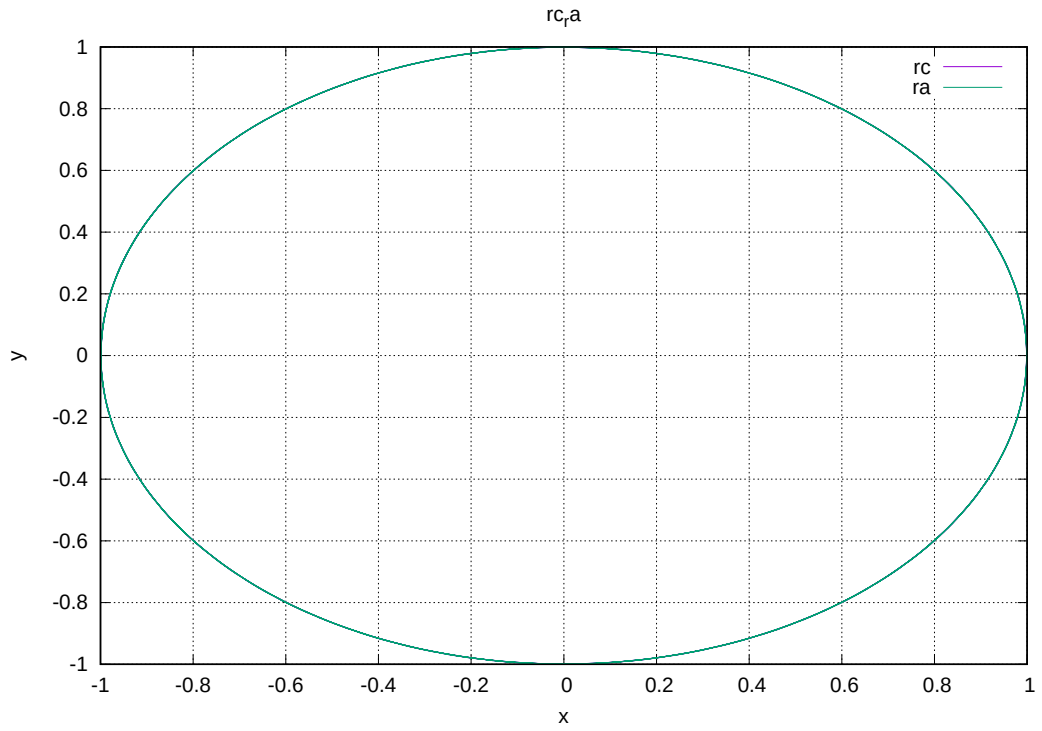

図 7　解析解と数値解の比較

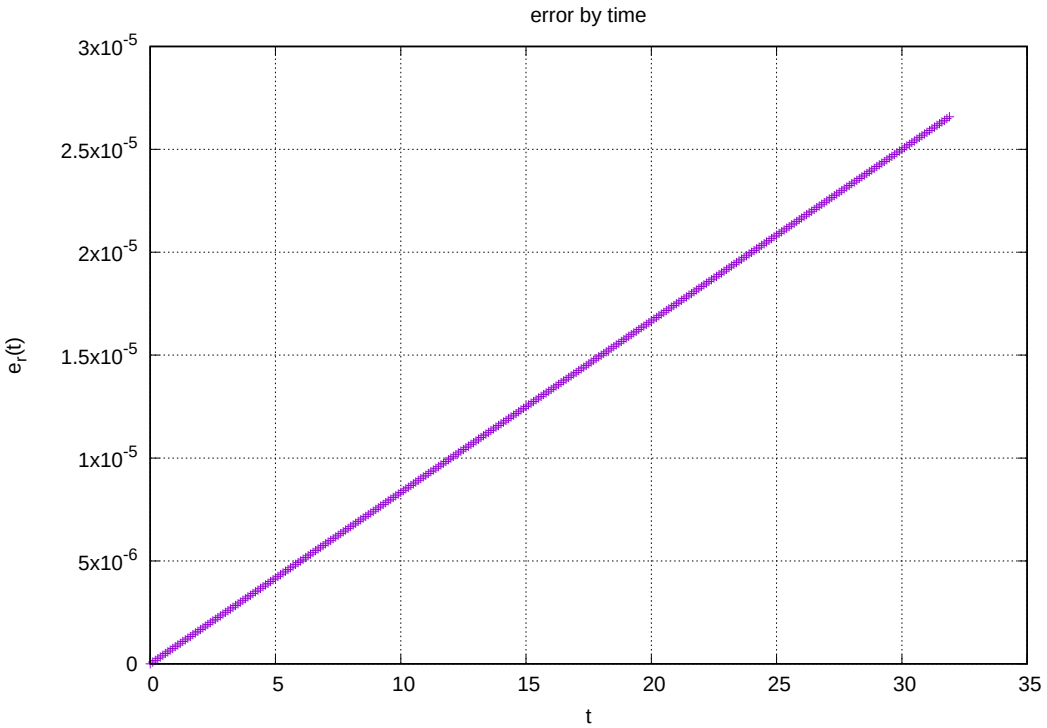

図 8　誤差の時間発展

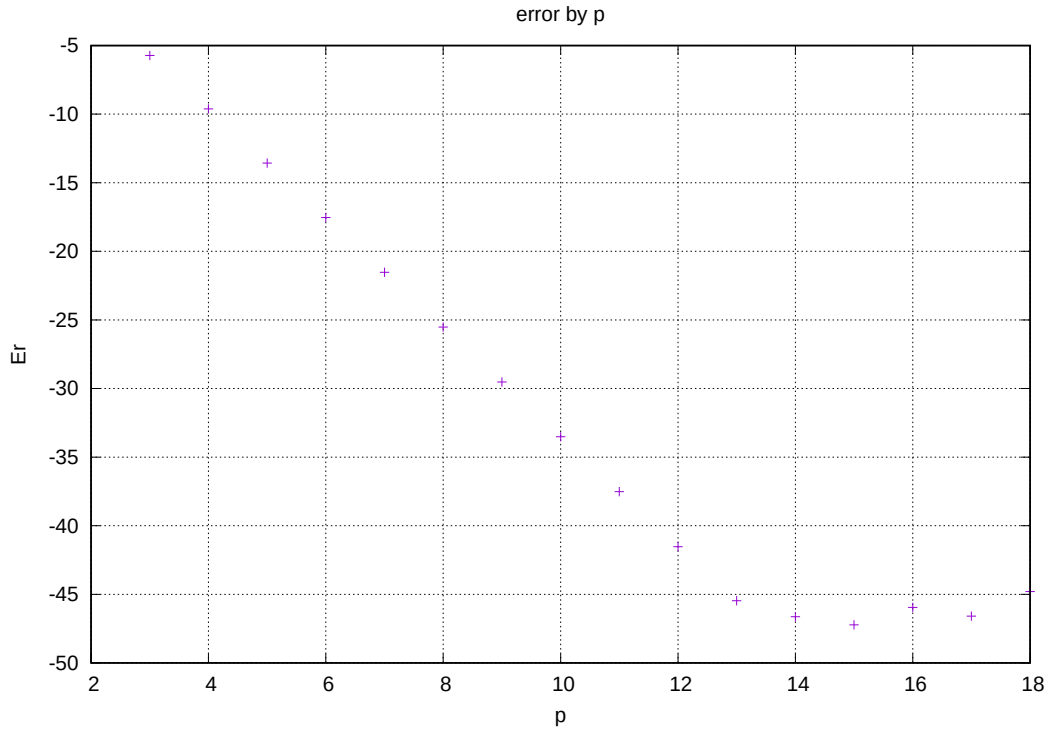

図 9　微小時間の大きさに対する誤差の大きさ

6　考察

6.1　ルンゲ-クッタ法の $p > 11$ での精度について

図 9 を見ると、$p > 13$ から、p を大きく、すなわち dt を小さくしても、精度が上がらなくなっている。これは、Ruby の Float クラスの精度が原因だと考えられる。

Ruby の Float クラスの精度は 15 桁ほどである。従って、15 桁程度まで数値解の精度が上がった場合、それ以上の精度は見込めない。実際、図 9 で E_r の値が減少しなくなったとき、$\log_2 E_r \approx -48$ であるから、$E_r \approx 10^{-14}$ である。解析解と数値解の差を取ったときに情報落ちが発生したと考えれば矛盾はない。

6.2　p - $\log_2 E_r$ の傾きについて

どの手法においても（精度の限界に至るまでは)、p - $\log_2 Er$ のグラフは直線である。すなわち、

$$\begin{aligned} \log_2 E_r &= p \cdot b + c \\ \therefore E_r &= 2^c \cdot (2^p)^b \\ \therefore E_r &\propto \mathrm{dt}^{-b} \\ (\because \mathrm{dt} &= \mathrm{T} \cdot 2^{-p}) \end{aligned}$$

である。

結果から、$-b$ の値は、オイラー法において 1 程度、ホイン法において 2 程度、4 次のルンゲ-クッタ法は 4 程度であった。これらの手法において、$\Delta\mathbf{x}$ の誤差は、

7 付録

8 参考文献

1. Euler 法 - [物理のかぎしっぽ] (https://goo.gl/ZV2wd6)
2. Heun 法 - [物理のかぎしっぽ] (https://goo.gl/0DH44q)
3. Runge-Kutta 法 - [物理のかぎしっぽ] (https://goo.gl/raIx64)
4. 2 数値計算法 - [物理のかぎしっぽ] (https://goo.gl/mGdShj)