プログラミング演習 第2課題

滝本亘（学籍番号 1029-33-1175）

2022 年 5 月 9 日

問題 1

(1)

作成したプログラムをコード 1 に示す。

授業資料の Algorithm 1 において、$a = 0, b = 1$ を代入している。
また、被積分関数 f(x) は 4〜8 行目で、$f(x) = \dfrac{4}{1+x^2}$ と定義している。
分割数 n は 13〜14 行目で、任意の整数に設定できる。

コード 1: 数値積分 1（台形公式）

```
1  #include <stdio.h>
2
3  //被積分関数の定義
4  double f(double x)
5  {
6      double y = 4/(1+x*x);
7      return y;
8  }
9
10 int main(void)
11 {
12     //分割数n
13     double n;
14     scanf("%lf", &n);
15
16     //面積の初期値を0に設定
17     double sum = 0;
18
19     //小台形の面積の和
20     for(double i=1; i<n; i++)
21     {
22         double t = i/n;
23         sum += f(t);
24     }
25
26     sum = (sum+(f(0)+f(1))/2)/n;
27
28     //結果をターミナルに出力
29     printf("%.16f\n", sum);
30 }
```

(2)

(1) で作成したプログラムを用いて、各分割数 n における計算結果と厳密値との差を表 1 に示す。ここで、厳密値は $\pi \approx 3.1415926535897932$ である。

表 1: 各分割数 n における計算結果と厳密値との差

n	計算結果	厳密値との差
10	3.1399259889071587	-0.0016666646826344
10^2	3.1415759869231290	-0.0000166666666641
10^3	3.1415924869231238	-0.0000001666666694
10^4	3.1415926519231396	-0.0000000016666535
10^5	3.1415926535731526	-0.0000000000166405

(3)

作成したプログラムをコード 2 に示す。
授業資料の Algorithm 1 において、$n = 10^4$ を代入している。
また、被積分関数 f(x) は 5〜9 行目で、$f(x) = \frac{e^{-\frac{x^2}{2}}}{\sqrt{2\pi}}$ と定義している。
積分区間は [a,b] は $a = -w, b = w$ とし、13 行目で $w = 1, 2, 3$ としている。

コード 2: 数値積分 2 （台形公式）

```
1  #include <stdio.h>
2  #include <math.h>
3
4  //被積分関数の定義
5  double f(double x)
6  {
7      double y = exp(-x*x/2)/sqrt(2*M_PI);
8      return y;
9  }
10
11 int main(void)
12 {
13     for(double w=1; w<=3; w++)
14     {
15         //分割数
16         double n = 10000;
17
18         //面積の初期値
19         double sum = 0;
20
21         //小台形の面積の和
22         for(double i=1; i<n; i++)
23         {
24             double t = -w + 2*w*i/n;
25             sum += f(t);
26         }
27
28         sum = 2*w*(sum+(f(-w)+f(w))/2)/n;
29
30         //結果をターミナルに出力
31         printf("%.16f\n", sum);
32     }
33 }
```

計算結果を表 2 に示す。

表 2: 各積分区間 [-w,w] における計算結果

w	計算結果
1	0.6826894905239459
2	0.9544997332241233
3	0.9973002031390054

問題 2

(1)

x_{i+1} の決め方より、

$$\begin{aligned} f(x_i) &= f'(x_i) \times (x_i - x_{i+1}) \\ \frac{f(x_i)}{f'(x_i)} &= x_i - x_{i+1} \\ x_{i+1} &= x_i - \frac{f(x_i)}{f'(x_i)} \end{aligned}$$

以上より、ニュートン法の反復公式が導出された。

(2)

作成したプログラムをコード 3 に示す。

授業資料の Algorithm 2 において、ε $= 10^{-12}$ を代入している。

コード 3: ニュートン法 1

```c
#include <stdio.h>
#include <math.h>

//実数関数 f(x)の定義
double f(double x);

//f(x)の導関数の定義
double df(double x);

int main(void)
{
    //初期値 x
    double x;
    scanf("%lf", &x);

    //ニュートン法の反復公式によって得られる x_
    double x_ = x - f(x)/df(x);

    //2 数の差の絶対値が指定の値未満になるまでループ
    while(fabs((x_-x)/x_)>=10e-12)
    {
        //2 数の値を更新
        x = x_;
```

```
        x_ -= f(x)/df(x);
    }

    //結果をターミナルに出力
    printf("%.16f\n",x_);
}
```

(3)

（2）で作成したプログラムにおいて、$f(x) = x^2 - 3$, $df(x) = 2x$ を代入する。
$f(x) = 0$ の解は $x = \pm\sqrt{3}$ であるため、ニュートン法の初期値を $\sqrt{3}$ に十分近い値に設定することで、$\sqrt{3}$ の近似値を得ることができる。

f(x), df(x) をコード 4 に示す。

コード 4: ニュートン法２

```
//実数関数 f(x)の定義
double f(double x)
{
    double y = x*x-3;
    return y;
}

//f(x)の導関数の定義
double df(double x)
{
    double y = 2*x;
    return y;
}
```

近似解の収束の様子を図に示す。
ここでは、初期値を $x = 10$ とした。

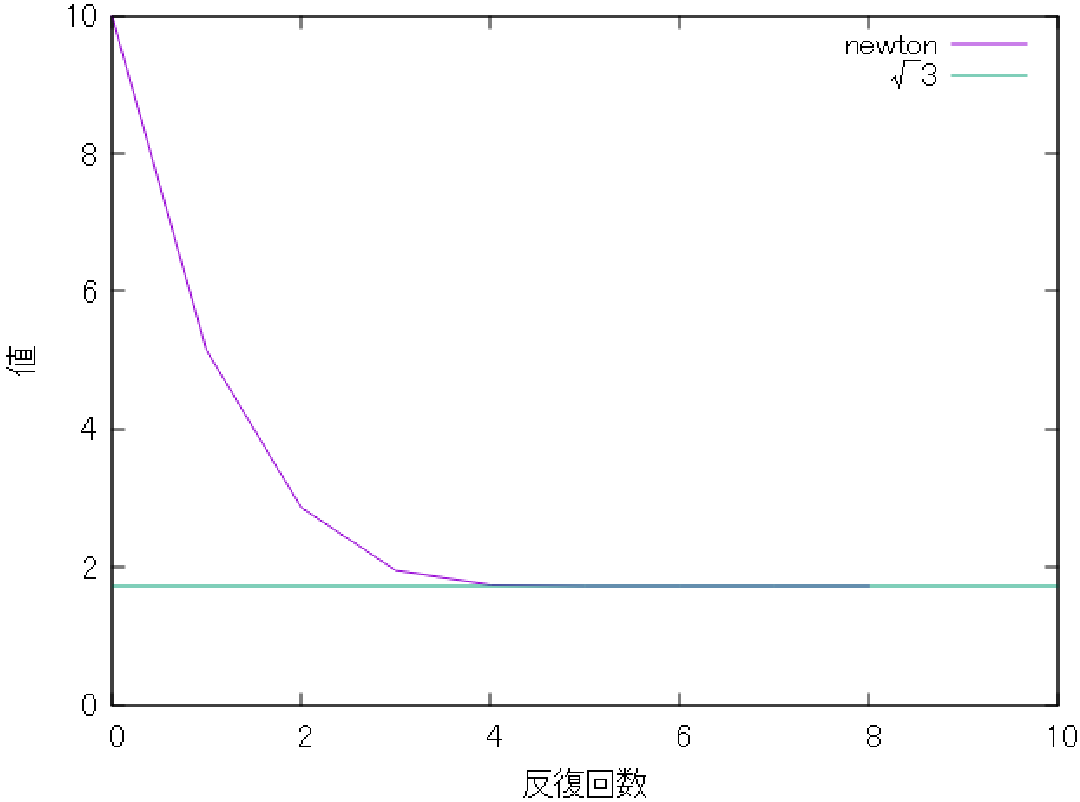

図 1: $\sqrt{3}$ の近似解の収束の様子

問題 3

(1)

ロジスティック方程式より、

$$\begin{aligned}
\frac{dN}{dt} &= rN\left(1-\frac{N}{K}\right) \\
\frac{K}{N(K-N)}dN &= rdt \\
\int\left(\frac{1}{N}+\frac{1}{K-N}\right)dN &= \int rdt \\
\log\left|\frac{K-N}{N}\right| &= -rt+C \ (C\text{ は積分定数}) \\
\frac{K-N}{N} &= \pm e^{-rt}e^{C}
\end{aligned}$$

$\pm e^{C}=A$ と置くと、

$$\begin{aligned}
\frac{K-N}{N} &= Ae^{-rt} \\
N &= \frac{K}{1+Ae^{-rt}}
\end{aligned}$$

以上より、ロジスティック方程式の一般解が導出された。

(2)

各初期値 N_0 について、ロジスティック写像による $N(t)$ の軌道と元のロジスティック方程式の解を比較した様子を図 2〜17 に示す。
いずれの場合も、Δt が小さくなるにつれ、$N(t)$ が正確な値となっている。

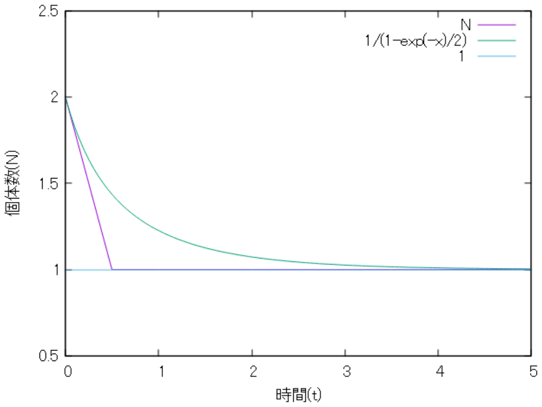

図 2: $\Delta t = \dfrac{1}{2}$, $N(0) = 2.0$

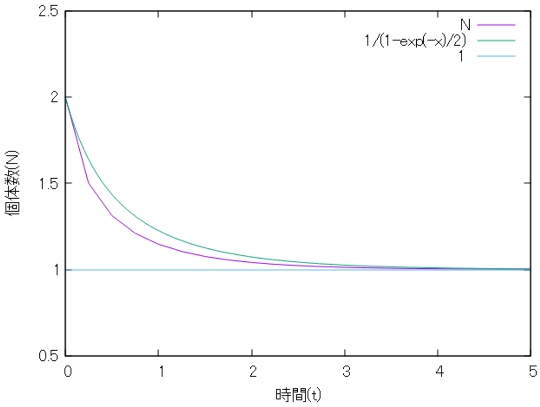

図 3: $\Delta t = \dfrac{1}{4}$, $N(0) = 2.0$

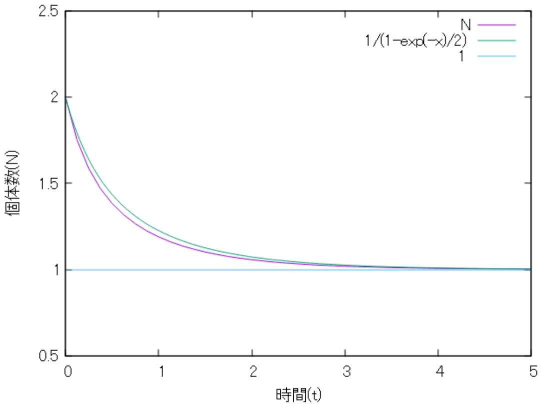

図 4: $\Delta t = \dfrac{1}{8}$, $N(0) = 2.0$

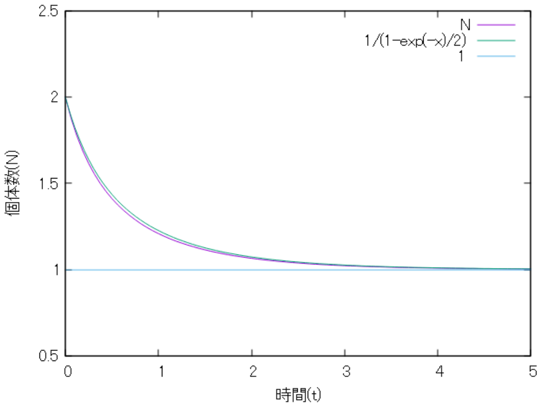

図 5: $\Delta t = \dfrac{1}{16}$, $N(0) = 2.0$

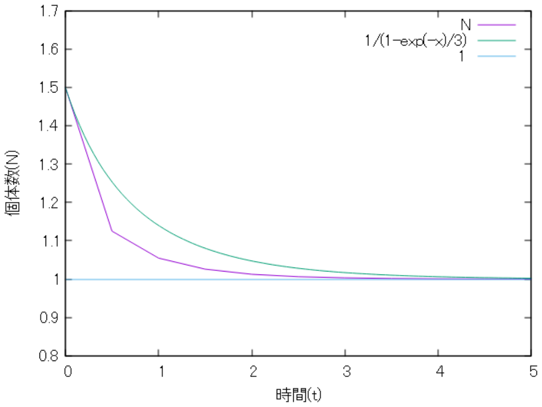

図 6: $\Delta t = \dfrac{1}{2}$, $N(0) = 1.5$

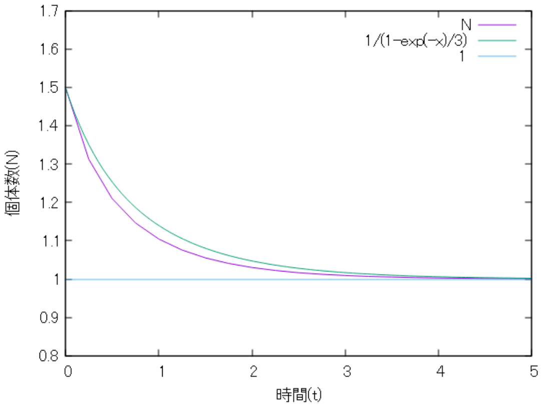

図 7: $\Delta t = \dfrac{1}{4}$, $N(0) = 1.5$

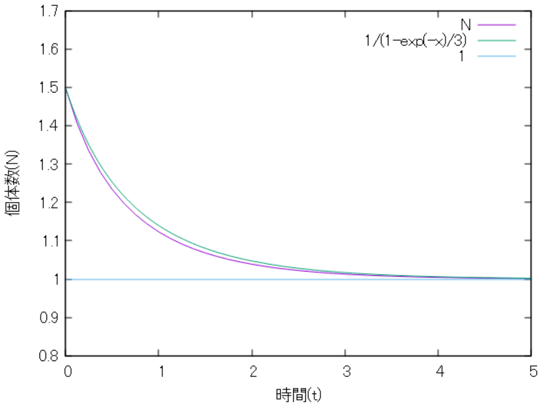

図 8: $\Delta t = \dfrac{1}{8}$, $N(0) = 1.5$

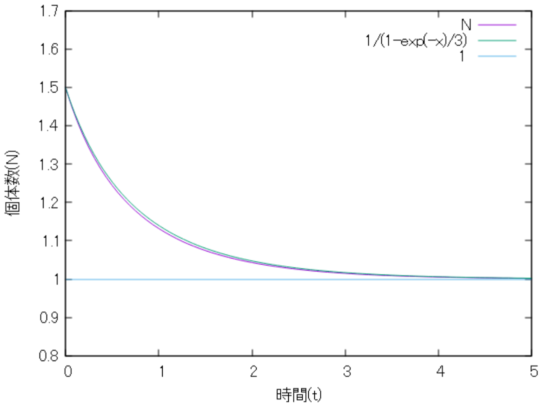

図 9: $\Delta t = \dfrac{1}{16}$, $N(0) = 1.5$

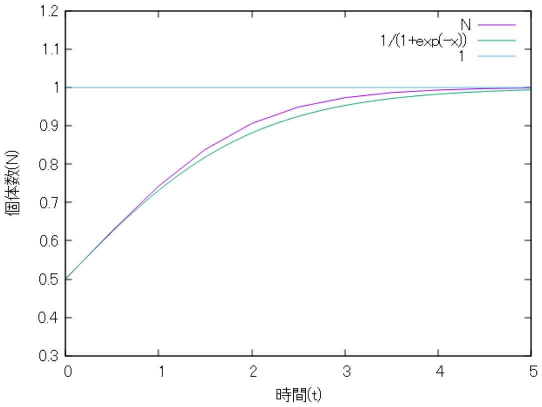

図 10: $\Delta t = \dfrac{1}{2}$, $N(0) = 0.5$

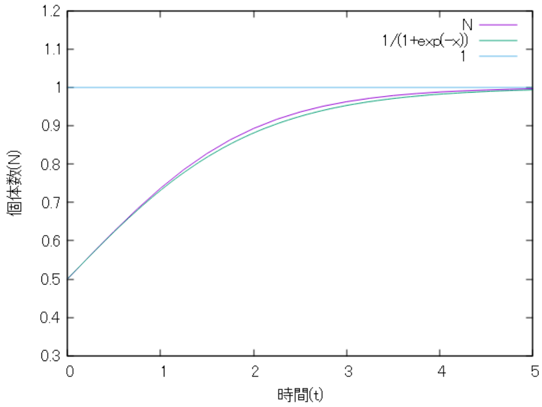

図 11: $\Delta t = \dfrac{1}{4}$, $N(0) = 0.5$

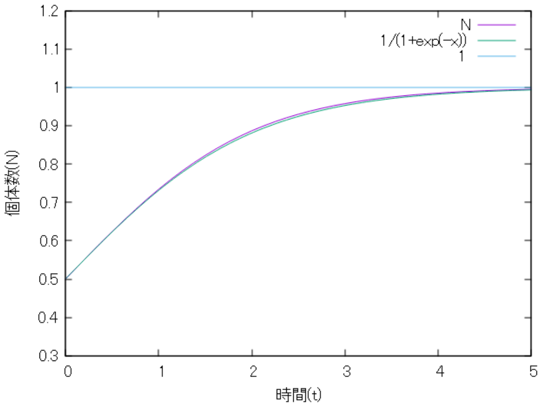

図 12: $\Delta t = \dfrac{1}{8}$, $N(0) = 0.5$

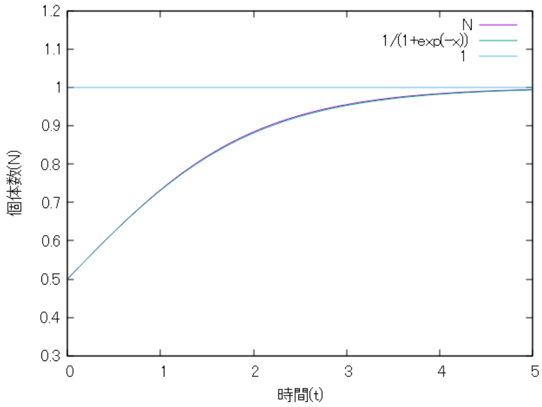

図 13: $\Delta t = \dfrac{1}{16}$, $N(0) = 0.5$

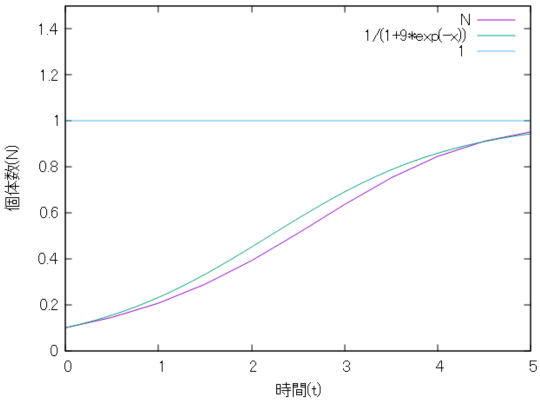

図 14: $\Delta t = \dfrac{1}{2}$, $N(0) = 0.1$

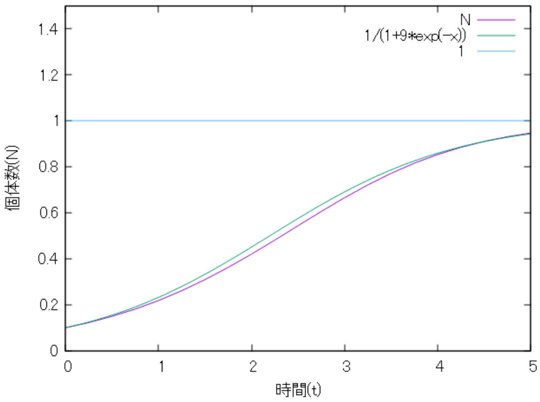

図 15: $\Delta t = \dfrac{1}{4}$, $N(0) = 0.1$

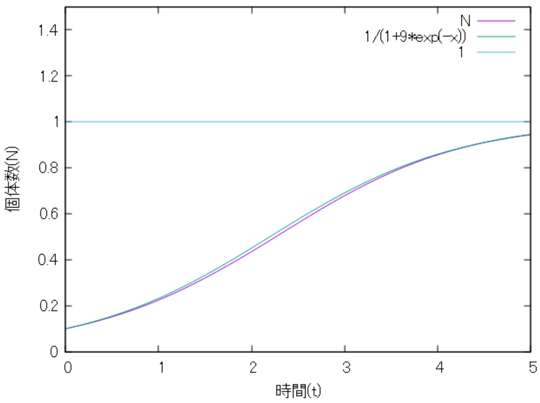

図 16: $\Delta t = \dfrac{1}{8}$, $N(0) = 0.1$

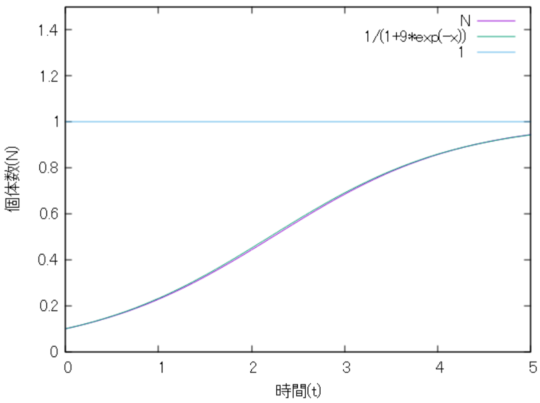

図 17: $\Delta t = \dfrac{1}{16}$, $N(0) = 0.1$

(3)

式 (4) を式 (5) に代入して、

$$x_1 = a^2 x_1(1 - x_1)\{1 - a x_1(1 - x_1)\}$$

を得る。
よって、問題２（２）で作成したプログラムにおいて、

$$\begin{aligned} f(x) &= a^3x^4 - 2a^3x^3 + (a^3 + a^2)x^2 + (1 - a^2)x \\ df(x) &= 4a^3x^3 - 6a^3x^2 + 2(a^3 + a^2)x + 1 - a^2 \end{aligned}$$

を代入する。

f(x), df(x) をコード 5 に示す。

コード 5: ロジスティック写像

```
double f(double x, double a)
{
    double y = a*a*a*x*x*x*x-2*a*a*a*x*x*x+(a*a*a+a*a)*x*x+(1-a*a)*x;
    return y;
}

double df(double x, double a)
{
    double y = 4*a*a*a*x*x*x-6*a*a*a*x*x+2*(a*a*a+a*a)*x+(1-a*a);
    return y;
}
```

ここで、初期値を $x = 1$ とすると、

$$x_1 = 0.8236032832060688$$

これを式 (4) に代入して、

$$x_2 = 0.4794270198242342$$

(4)

$a = 01.0, 1.5, 2.0, 2.5, 3.0, 3.3, 3.5, 4.0$ としたときの軌道 $\{x_i\}$ をそれぞれ図 18〜25 に示す。
$a = 3.5$ くらいからカオスに到っている。

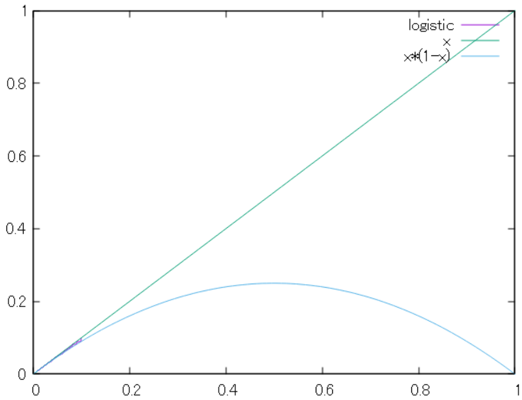

図 18: $a = 0.5$

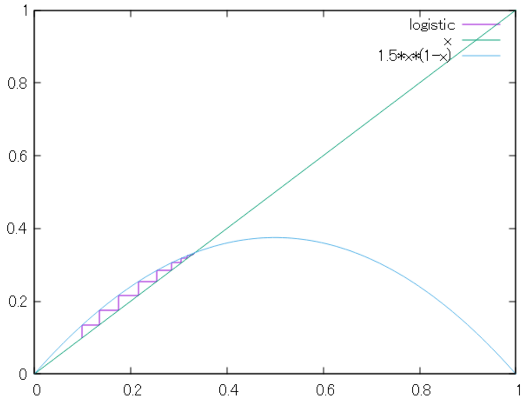

図 19: $a = 1.0$

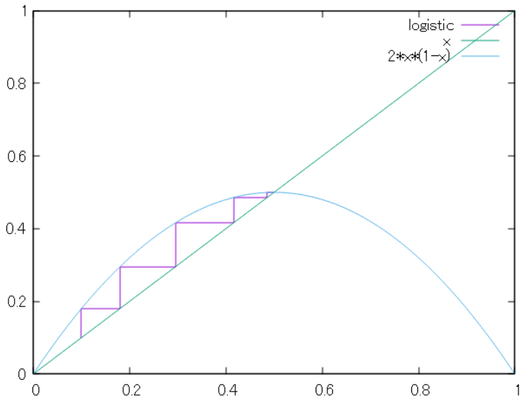

図 20: $a = 1.5$

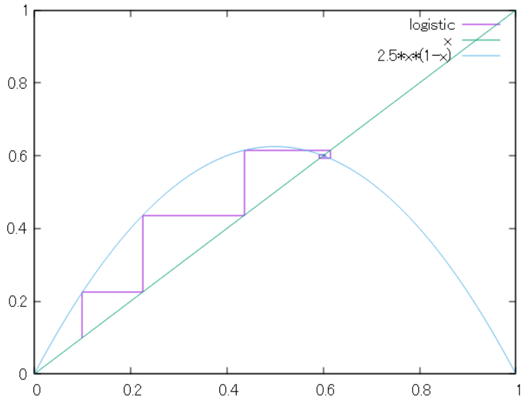

図 21: $a = 2.0$

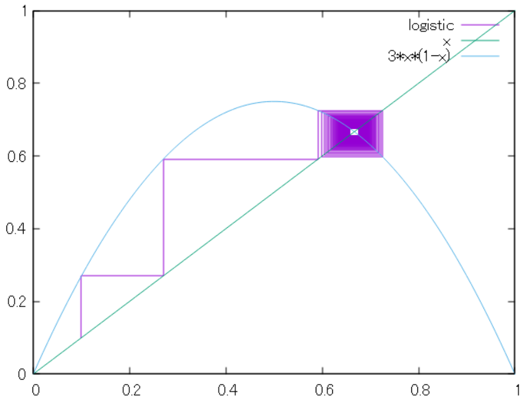

図 22: $a = 2.5$

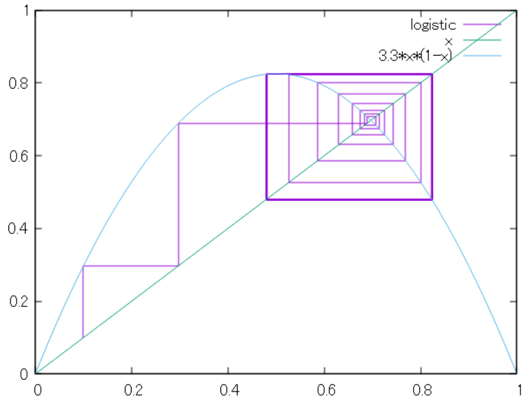

図 23: $a = 3.0$

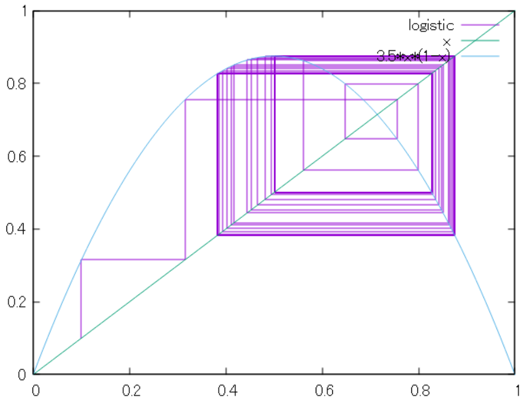

図 24: $a = 3.5$

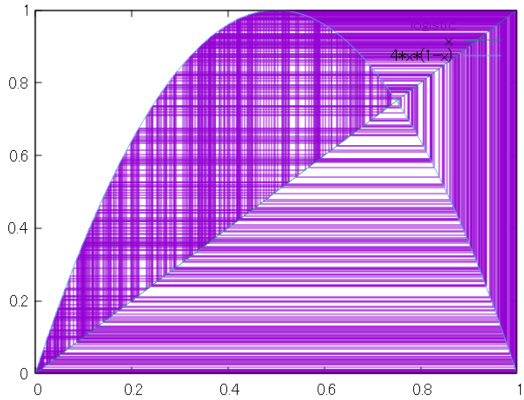

図 25: $a = 4.0$

参考文献

[1] ”プログラミング演習”、岩崎淳、根本孝裕、原田健自、2022