情報処理演習グループ課題1

役職	学籍番号	名前	備考
リーダー	BQ23107	窪田 大輝	
プログラムリーダー	BQ23008	脇家 優太	
企画係	BQ23071	平川 奨	
連絡係	BQ23060	塚田 水月	
書記１	BQ23110	信賀 晃	グループワークには参加したものの，担当分未提出
書記２	BQ23103	山田 泰我	欠席

ロジスティック写像とは

ロジスティック写像は人口の推移を表す単純なモデルである。$y = ax(1-x)$のような二次関数で表される。ここで、aは定数を意味し、パラメータと呼ばれる。変数xの値からyの値を計算し、得られたyの値を新しいxの値として式に代入することで新しいyを得られる。

`

C言語でのロジスティック写像プログラム

今回取り扱うロジスティック写像では、グラフの初期条件(最大繁殖率a)によっては一定間隔で値が大きな変化を起こす「カオス（初期条件鋭敏性）」が発生する。 そこで、C言語によるプログラムを用いてカオスを意図的に発生させる。そして、未知数にさまざまな値を代入して挙動の変化を確認することが今回の目的である。

実験において使用したソースコード(scombz内解説動画「プログラム説明1」より引用)は、

・xの計算結果のみを100回プロットするプログラム ・x・yの計算結果をそれぞれプロットするプログラム の2つである。

まずこれらのソースコードを使用し、使用する未知数の数の違いによる変化を観察した。

```
#include <stdio.h>

int main() {

    int n, i;
    double a, x0, x;
    a = 1.5;//ロジスティック写像のパラメータ
    x0 = 0.5;//初期条件 x0 € [0,1]

    printf("0 %f %f\n", x0);//（回数，値）

    n = 100;
```

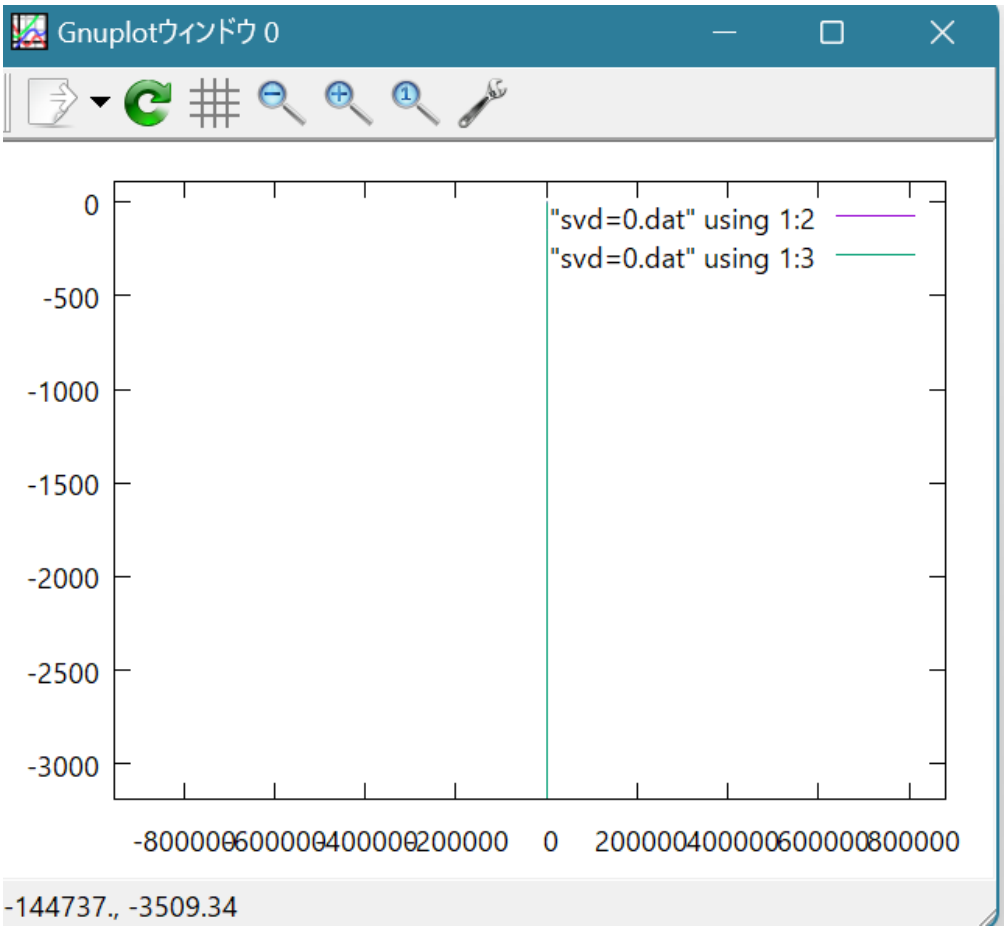
```
    x = x0;

    for (i = 1;i <= n;i++) {
        x = a * x * (1 - x);
        printf("%d %f\n", i, x);
    }

    }
```

Fig.1 題名：NC1.c（ｘの計算結果を１００回プロットするプログラム）

```
#include <stdio.h>


int main() {

    int n, i;
    double a, x0, x,y0,y,e;
    a = 1.5;//ロジスティック写像のパラメータ
    x0 = 0.5;//初期条件 x0 € [0,1]
    e = 0.0001;//初期条件の差
    y0 = x0 + e;//もう一つの初期条件;


    printf("0 %f %f\n", x0, y0);//（回数，値）

    n = 100;

    x = x0;
    y = y0;


    for (i = 1;i <= n;i++) {
        x = a * x * (1 - x);
        y = a * y * (1 - y);
        printf("%d %f %f\n", i, x,y);
    }

}
```

Fig.2 NC1′.c（ｘ・ｙの計算結果をそれぞれ１００回プロットするプログラム）

Fig.1・2にて示された計算結果をgunplotにて同時にプロットしたものは，以下の通りである．



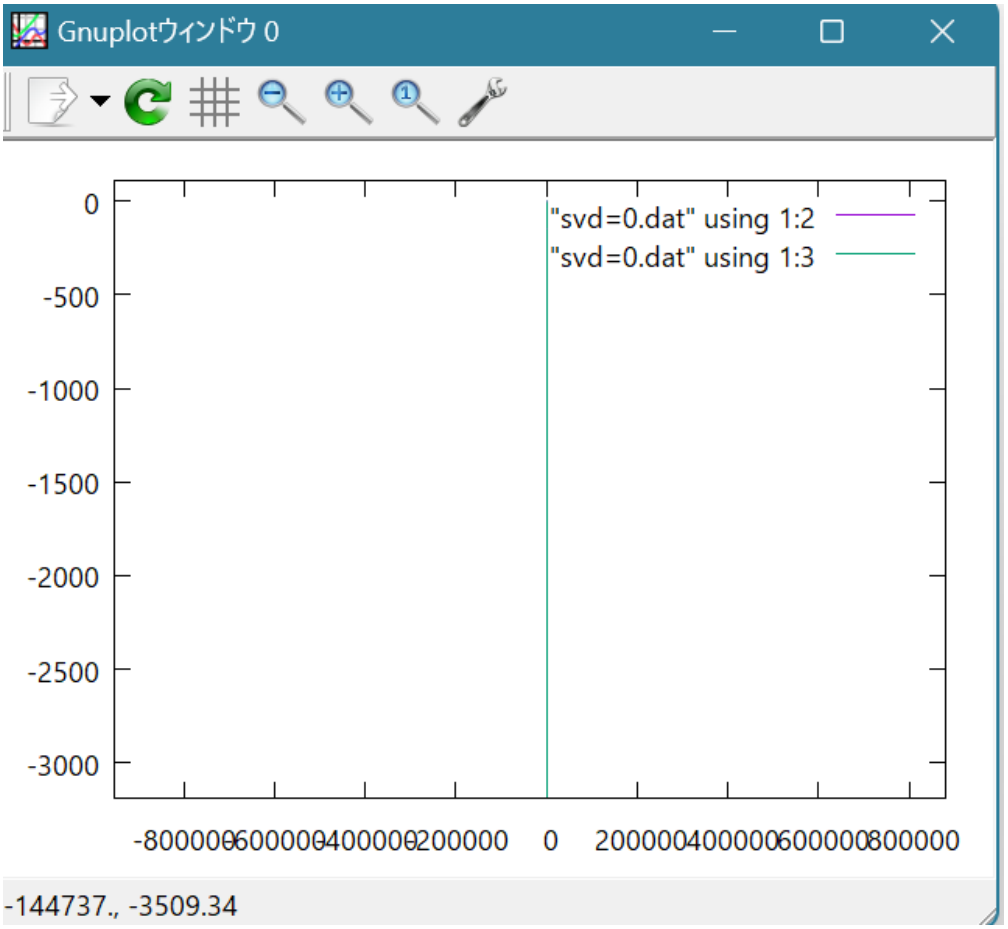
このように，初期条件の一つである開始位置が異なるだけである．つまり，グラフが重なった理由は，それ以外の初期条件の一つであるaによるものであることが．後述する結果からもうかがえる．

ここで，前述のパラメータaを変更したグラフを

また， Fig.2にて示したプログラム内の定数$a（最大繁殖率）$を　1.5>6　に変更したプログラムおよび結果をプロットした表は以下の通りである．

```c
#include <stdio.h>

int main(){

    int n,i;
    double a,e,x0,x,y0,y;

    a=6;//ロジスティック写像のパラメータ a € [0,4]
    x0=0.5;//初期条件 x0 € [0,1]
    y0=x0+e;//追加の初期条件
    e=0.0001;//初期条件の差

    printf("0 %f\n",x0);//(回数,値)，回数と初期位置

    n=100;

    x=x0;
    y=y0;
```

```
    for(i=1;i<=n;i++){//100回分の入力
    x=a*x*(1-x);
    y=a*y*(1-y);
    printf("%d %f %f\n",i,x,y);//回数とその時の位置を表示
    }

}
```



以上のように，今回は一本の線状になってしまったが，初期条件を変更しただけでもグラフの形状が大きく変化したということがわかる．

Pythonでのロジスティック写像プログラム

前述の通り，C言語でのロジステック写像プログラムは示した．ここではPythonでのロジスティック写像プログラムを示す。以下の図の通り，pythonでロジスティクス

```
import matplotlib.pyplot as plt #グラフを描画するためにMatplotlibのpyplotモジュールをpltとしてインポート

def logistic_map(a, x):  #ロジスティクス写像のパラメータをa、現在の状態をxとおく
    return a * x * (1 - x)

def generate_logistic_map_sequence(a, x0, num_steps): #ロジスティック写像グラフを生成する関数を定義。x0は初期条件、num_stepsは繰り返し回数
```

```
    result = [] #ロジスティクス写像の結果を格納するためのリスト
    current_x = x0  #current_xを初期条件x0に設定
    for _ in range(num_steps):
        result.append(current_x) #現在のcurrent_xの値を結果リストに追加
        current_x = logistic_map(a, current_x) #current_xを更新
    return result

def plot_logistic_map(a, x0, num_steps): #ロジスティック写像のシーケンスをプロットする関数を定義
    sequence = generate_logistic_map_sequence(a, x0, num_steps) #ロジスティック写像のシーケンスを生成

    plt.plot(sequence, 'o-', markersize=2) #Matplotlibを使用してシーケンスをプロット,'o-'は点と線でつながるスタイルを指定。
    plt.title(f'Logistic Map (a={a}, x0={x0})')
    plt.xlabel('Steps')
    plt.ylabel('x')
    plt.show()

#パラメータの設定
a_value=3.8  #ロジスティック写像のパラメータ
x0_value=0.5  #初期値
num_steps_value=100  #ステップ数

#ロジスティック写像のプロット
plot_logistic_map(a_value, x0_value, num_steps_value)
```