情報理論 第 8 回 計算機演習

222C1021 今村優希

2024 年 6 月 8 日

演習 1

$P_{X,Y}(x,y)$		Y		$P_X(x)$
		風邪をひいている	風邪をひいていない	
X	熱がある	0.55	0.05	
	熱がない	0.15	0.25	
$P_Y(y)$				

図 1　2 つの情報源の発生確率

上記の図 1 の 2 つの情報源の値から相互情報量を求める．まずは，それぞれの周辺確率を求める．

$$P_X(0) = \sum_j P_{X,Y}(0, y_j) = P_{X,Y}(0,0) + P_{X,Y}(0,1) = 0.55 + 0.05 = 0.60$$
$$P_X(1) = \sum_j P_{X,Y}(1, y_j) = P_{X,Y}(1,0) + P_{X,Y}(1,1) = 0.15 + 0.25 = 0.40$$
$$P_Y(0) = \sum_i P_{X,Y}(x_i, 0) = P_{X,Y}(0,0) + P_{X,Y}(1,0) = 0.55 + 0.15 = 0.70$$
$$P_Y(1) = \sum_i P_{X,Y}(x_i, 1) = P_{X,Y}(0,1) + P_{X,Y}(1,1) = 0.05 + 0.25 = 0.30$$

この確率から X, Y それぞれのエントロピーを求める．

$$H(X) = -0.6 \log_2 0.6 - 0.4 \log_2 0.4 = 0.442 + 0.529 = 0.971$$
$$H(Y) = -0.7 \log_2 0.7 - 0.3 \log_2 0.3 = 0.881$$

さらに，周辺確率から条件確率 $P_{X|Y}(x_i|y_j) = \dfrac{P_{X,Y}(x_i, y_j)}{P_Y(y_j)}$ を求める．

$$P_{X|Y}(0|0) = \frac{P_{X,Y}(0,0)}{P_Y(0)} = \frac{0.55}{0.7} = 0.786$$
$$P_{X|Y}(0|1) = \frac{P_{X,Y}(0,1)}{P_Y(1)} = \frac{0.05}{0.3} = 0.167$$
$$P_{X|Y}(1|0) = \frac{P_{X,Y}(1,0)}{P_Y(0)} = \frac{0.15}{0.7} = 0.214$$
$$P_{X|Y}(0|0) = \frac{P_{X,Y}(1,1)}{P_Y(1)} = \frac{0.25}{0.3} = 0.833$$

これを用いて条件付きエントロピーを求める．先に，$H(X|Y=0)$ と $H(X|Y=1)$ を求める．

$$\begin{aligned} H(X|Y=0) &= -P_{X|Y}(0|0) \log_2 P_{X|Y}(0|0) - P_{X|Y}(1|0) \log_2 P_{X|Y}(1|0) \\ &= -0.786 \log_2 0.786 - 0.214 \log_2 0.214 \\ &= 0.749 \\ H(X|Y=1) &= -P_{X|Y}(0|1) \log_2 P_{X|Y}(0|1) - P_{X|Y}(1|1) \log_2 P_{X|Y}(1|1) \\ &= -0.167 \log_2 0.167 - 0.833 \log_2 0.833 \\ &= 0.651 \end{aligned}$$

$$\begin{aligned} H(X|Y) &= \sum_j P_Y(y_i) H(X|Y=y_i) \\ &= P_Y(0) H(X|Y=0) + P_Y(1) H(X|Y=1) \\ &= 0.7 \cdot 0.749 + 0.3 \cdot 0.651 \qquad = 0.720 \end{aligned}$$

以上の結果から，相互情報量 $I(X;Y)$ を求めると，

$$I(X;Y) = H(X) - (X|Y) = 0.971 - 0.720 = 0.251$$

したがって，相互情報量 $I(X;Y)$ は，0.251 である．

演習 2

演習 2-1

作成した状態遷移図は図 2 である．ラーメンを S_1，そばを S_2，うどんを S_3 で作成を行った．

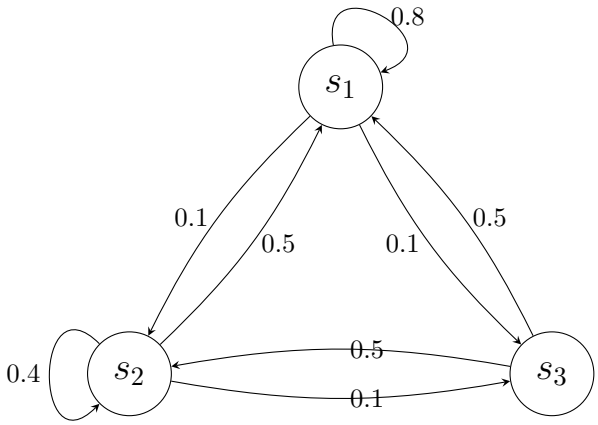

図 2　状態遷移図

演習 2-2

遷移確率行列を求めるために用いた $\prod$ は,

$$\prod = \begin{pmatrix} 0.8 & 0.1 & 0.1 \\ 0.5 & 0.4 & 0.1 \\ 0.5 & 0.5 & 0 \end{pmatrix}$$

である．$\prod^2, \prod^3, ..., \prod^{10}, \prod^{50}, \prod^{100}$ を算出するために以下のようなプログラムを作成し，matlab での実行を行った．5 行目で初期の行列の入力を行い，7 行目以下で行列の計算及び出力を行っている．

ソースコード 1　遷移確率行列を求めるプログラム

```
1  %情 報 理 論
2  %222C1021  今 村 優 希
3  %演 習 問 題2−2  Π の2~10,50,100 乗 を 求 め る 問 題
4
5  A = [0.8  0.1  0.1; 0.5  0.4  0.1; 0.5  0.5  0]; %行 列 の 入 力
6
7  %Π^1~ Π^10 ま で の 計 算 と 値 の 出 力
8  for c = 1:10
9      x_1_10 = A^c;
10     disp(c);
11     disp(x_1_10);
12 end
13
14 %Π^50 の 計 算 と 値 の 出 力
15 X_50 = A^50;
```

```
16  disp(50);
17  disp(X_50);
18  %Π^100の計算の値と出力
19  X_100 = A^100;
20  disp(100);
21  disp(X_100);
```

このプログラムを実行した結果がソースコード 2 である．1 行目の 1 が $\prod^1$ を表しており，その結果が 3 から 5 行目に行列として出力されている．

ソースコード 2　遷移確率行列を求めるプログラム

```
1   1
2
3   0.8000    0.1000    0.1000
4   0.5000    0.4000    0.1000
5   0.5000    0.5000         0
6
7    2
8
9   0.7400    0.1700    0.0900
10  0.6500    0.2600    0.0900
11  0.6500    0.2500    0.1000
12
13   3
14
15  0.7220    0.1870    0.0910
16  0.6950    0.2140    0.0910
17  0.6950    0.2150    0.0900
18
19   4
20
21  0.7166    0.1925    0.0909
22  0.7085    0.2006    0.0909
23  0.7085    0.2005    0.0910
24
25   5
26
27  0.7150    0.1941    0.0909
28  0.7126    0.1965    0.0909
29  0.7126    0.1966    0.0909
30
31   6
32
33  0.7145    0.1946    0.0909
34  0.7138    0.1953    0.0909
35  0.7138    0.1953    0.0909
36
37   7
38
39  0.7143    0.1947    0.0909
40  0.7141    0.1950    0.0909
41  0.7141    0.1950    0.0909
42
43   8
```

```
44
45    0.7143      0.1948      0.0909
46    0.7142      0.1949      0.0909
47    0.7142      0.1949      0.0909
48
49     9
50
51    0.7143      0.1948      0.0909
52    0.7143      0.1948      0.0909
53    0.7143      0.1948      0.0909
54
55    10
56
57    0.7143      0.1948      0.0909
58    0.7143      0.1948      0.0909
59    0.7143      0.1948      0.0909
60
61    50
62
63    0.7143      0.1948      0.0909
64    0.7143      0.1948      0.0909
65    0.7143      0.1948      0.0909
66
67   100
68
69    0.7143      0.1948      0.0909
70    0.7143      0.1948      0.0909
71    0.7143      0.1948      0.0909
```

演習 2-3

定常確率の計算を行う．演習 2-1 にて作成した状態遷移図から以下の連立方程式が得られる．

$$\begin{cases} P(S_1) = 0.8P(S_1) + 0.5P(S_2) + 0.5P(S_3) \\ P(S_2) = 0.1P(S_1) + 0.4P(S_2) + 0.5P(S_3) \\ P(S_3) = 0.1P(S_1) + 0.1P(S_2) \end{cases} \tag{1}$$

また，確率の総和は 1 なので，

$$P(S_1) + P(S_2) + P(S_3) = 1 \tag{2}$$

も成り立つ．数式 (1) と (2) を用いて連立方程式を解くと，以下の結果が得られた．

$$\begin{cases} P(S_1) = \dfrac{5}{7} = 0.714 \\ P(S_2) = \dfrac{15}{77} = 0.195 \\ P(S_3) = \dfrac{1}{11} = 0.091 \end{cases}$$

この結果から，$\prod^{100}$ の値における 1 列目が $P(S_1)$，2 列目が $P(S_2)$，3 列目が $P(S_3)$ とほとんど同じ値を出力していることがわかった．それぞれの列が定常確率を表していると思われる．このことから，$\prod$ の乗数を大きくすれば，それぞれの列の値が定常確率に近づいていることが確認できる．

演習 2-4

演習 2-3 で求めた定常確率からエントロピーを計算する.

$$\begin{aligned} H(X) &= -\frac{5}{7}\log_2\frac{5}{7} - \frac{15}{77}\log_2\frac{15}{77} - \frac{1}{11}\log_2\frac{1}{11} \\ &= 0.347 + 0.460 + 0.314 \\ &= 1.120 \end{aligned}$$

と求めた.

演習 3

演習 3-1

50 回遷移するように設計したプログラムは，ソースコード 3 である．11 行目の StudentID を 021 に設定した．また，50 回遷移するから，NIteration を「50」に，さらに 62 行目も「1:50」に変更した．このプログラムを matlab で実行した結果が，ソースコード 4 である．また，遷移した状態の回数を棒グラフで表したものが図 3 である．

ソースコード 3　状態変化を見るプログラム

```
1   % 九州工業大学情報工学部情報・通信工学科3年
2  % 情報理論 演習課題1 演習3
3  % 2024/05/10 黒崎正行
4  % 2024/05/10 今村優希一部変更
5
6  clear;
7  clearvars -global;
8  close all;
9  clc;
10 %% Parameter setting
11 StudentID = 021;              % 学籍番号の下3桁に変更すること
12 seed = StudentID;             % 学籍番号を乱数のシード値に設定
13 rng(seed);                    % 乱数のシード値の設定
14
15 CurrentState = 1;             % 現在の状態 (初期状態1)
16 NextState = 1;                % 次の状態
17
18 NIteration = 50;            % 繰返し回数
19
20 StateHist = zeros(1, NIteration);
21
22 %% Start simulation
23 for ite = 1: NIteration
24
25     Transition = rand;                                    % ランダム値の生成
26     StateHist(ite) = CurrentState;                        % 現在の状態の番号を格納
27
28     switch CurrentState
29         case 1                                            % 状態1における動作を記述
30             if Transition < 0.8                           % 0.8の確率で状態1に遷移
31                 NextState = 1;
32             elseif Transition >= 0.8 && Transition < 0.9  % 0.9-0.8=0.1の確率で状態2に
                   遷移
33                 NextState = 2;
34             else                                          % 0.1の確率で状態3に遷移
35                 NextState = 3;
36             end
37         case 2                                            % 状態2における動作を記述
38             if Transition < 0.5                           % 0.5の確率で状態1に遷移
39                 NextState = 1;
40             elseif Transition >= 0.5 && Transition < 0.9  % 0.9-0.5=0.4確率で状態2に遷
                   移
41                 NextState = 2;
```

```
42              else                                                    % 0.1の確率で状態2に遷移
43                  NextState = 3;
44              end
45          case 3                                                      % 状態3における動作を記述
46              if Transition < 0.5                                     % 0.3の確率で状態1に遷移
47                  NextState = 1;
48              elseif Transition >= 0.5 && Transition < 1.0  % 1.0-0.5=0.5の確率で状態2に
                    遷移
49                  NextState = 2;
50              else                                                    % 0の確率で状態2に遷移
51                  NextState = 3;
52              end
53          otherwise
54              NextState=1;
55      end
56      CurrentState = NextState;
57  end
58
59  %% Evaluation
60  % 最初から50回の遷移を表示
61  disp('State History');
62  disp(StateHist(1:50));
63
64  % ヒストグラムのグラフの表示
65  h = histogram(StateHist);
66
67  % 頻度の表示
68  %
69   disp('State Frequency');
70   disp('State    Count    Percent');
71   fprintf('    S1, %6d, %8.3f %%\n', h.Values(1),h.Values(1)/NIteration*100);
72   fprintf('    S2, %6d, %8.3f %%\n', h.Values(2),h.Values(2)/NIteration*100);
73   fprintf('    S3, %6d, %8.3f %%\n', h.Values(3),h.Values(3)/NIteration*100);
```

ソースコード 4　ソースコード 3 を実行したときの結果

```
1    State History
2    1 列から 20 列
3
4       1     1     1     1     1     1     1     1     1     1     1     1     2     1
              1     1     2     2     3     2
5
6    21 列から 40 列
7
8       1     1     1     1     2     3     2     2     1     1     1     1     1     1
              1     1     1     2     1     1
9
10   41 列から 50 列
11
12      1     1     1     2     2     2     1     2     2     2
13
14  State Frequency
15  State    Count    Percent
16     S1,       34,    68.000 %
```

```
    S2,      14,    28.000 %
    S3,       2,     4.000 %
```

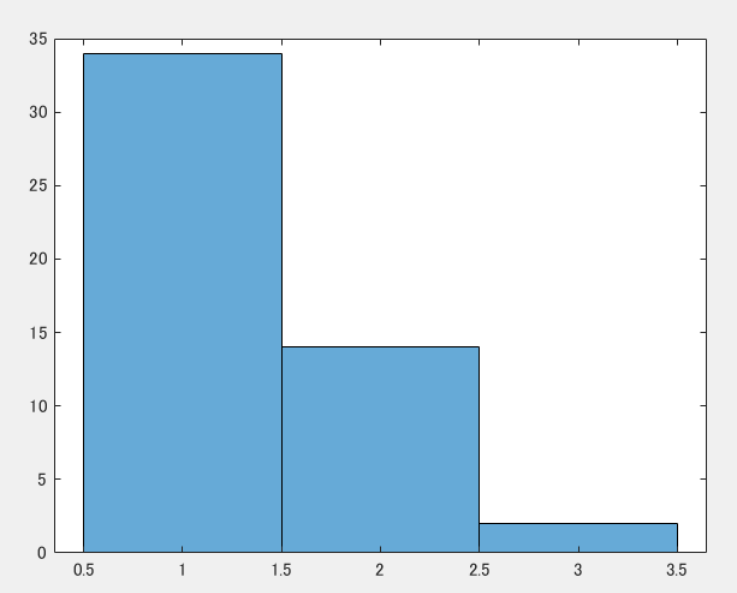

図 3　50 回遷移させたときの各状態に遷移した回数

演習 3-2

利用したプログラムは演習 3-1 で使用したソースコード 3 を一部変更したものである．1000 回遷移するから，NIteration を「1000」に，さらに 62 行目も「1:1000」に変更した．出力はソースコード 5 で表しており，遷移した状態の回数を棒グラフで表したものが図 4 である．

ソースコード 5　1000 回遷移させるためのプログラム

```
1   % 九州工業大学情報工学部情報・通信工学科3年
2  % 情報理論 演習課題1 演習3
3  % 2024/05/10 黒崎正行
4  % 2024/05/10 今村優希改変
5
6  clear;
7  clearvars -global;
8  close all;
9  clc;
10 %% Parameter setting
11 StudentID = 021;              % 学籍番号の下3桁に変更すること
12 seed = StudentID;             % 学籍番号を乱数のシード値に設定
13 rng(seed);                    % 乱数のシード値の設定
14
15 CurrentState = 1;             % 現在の状態 (初期状態1)
16 NextState = 1;                % 次の状態
17
18 NIteration = 1000;            % 繰返し回数
19
20 StateHist = zeros(1, NIteration);
21
```

```
22 %% Start simulation
23 for ite = 1: NIteration
24
25     Transition = rand;                                          % ランダム値の生成
26     StateHist(ite) = CurrentState;                              % 現在の状態の番号を格納
27
28     switch CurrentState
29         case 1                                                  % 状態1における動作を記述
30             if Transition < 0.8                                 % 0.8の確率で状態1に遷移
31                 NextState = 1;
32             elseif Transition >= 0.8 && Transition < 0.9         % 0.9-0.8=0.1の確率で状態2
                   に遷移
33                 NextState = 2;
34             else                                                % 0.1の確率で状態3に遷移
35                 NextState = 3;
36             end
37         case 2                                                  % 状態2における動作を記述
38             if Transition < 0.5                                 % 0.5の確率で状態1に遷移
39                 NextState = 1;
40             elseif Transition >= 0.5 && Transition < 0.9         % 0.9-0.5=0.4確率で状態2に
                   遷移
41                 NextState = 2;
42             else                                                % 0.1の確率で状態2に遷移
43                 NextState = 3;
44             end
45         case 3                                                  % 状態3における動作を記述
46             if Transition < 0.5                                 % 0.3の確率で状態1に遷移
47                 NextState = 1;
48             elseif Transition >= 0.5 && Transition < 1.0         % 1.0-0.5=0.5の確率で状態2
                   に遷移
49                 NextState = 2;
50             else                                                % 0の確率で状態2に遷移
51                 NextState = 3;
52             end
53         otherwise
54             NextState=1;
55     end
56     CurrentState = NextState;
57 end
58
59 %% Evaluation
60 % 最初から1000回の遷移を表示
61 disp('State History');
62 disp(StateHist(1:1000));
63
64 % ヒストグラムのグラフの表示
65 h = histogram(StateHist);
66
67 % 頻度の表示
68 %
69  disp('State Frequency');
70  disp('State   Count   Percent');
71  fprintf('   S1, %6d, %8.3f %%\n', h.Values(1),h.Values(1)/NIteration*100);
72  fprintf('   S2, %6d, %8.3f %%\n', h.Values(2),h.Values(2)/NIteration*100);
```

```
73  fprintf('    S3, %6d, %8.3f %%\n', h.Values(3),h.Values(3)/NIteration*100);
```

ソースコード 6　ソースコード 5 を実行した際の結果

```
1   State History
2   1 列から 20 列
3
4      1     1     1     1     1     1     1     1     1     1     1     1     2     1
           1     1     2     2     3     2
5
6   21 列から 40 列
7
8      1     1     1     1     2     3     2     2     1     1     1     1     1     1
           1     1     1     2     1     1
9
10  % 一部省略 %
11
12  941 列から 960 列
13
14     1     2     2     1     1     1     1     2     2     2     1     3     2     1
           1     2     2     2     1     1
15
16  961 列から 980 列
17
18     1     3     2     1     1     2     2     1     1     2     2     1     3     2
           2     2     2     1     1     2
19
20  981 列から 1,000 列
21
22     2     1     2     2     1     1     1     1     1     1     3     1     1     1
           1     1     1     1     2     2
23
24  State Frequency
25  State    Count    Percent
26     S1,      713,    71.300 %
27     S2,      195,    19.500 %
28     S3,       92,     9.200 %
```

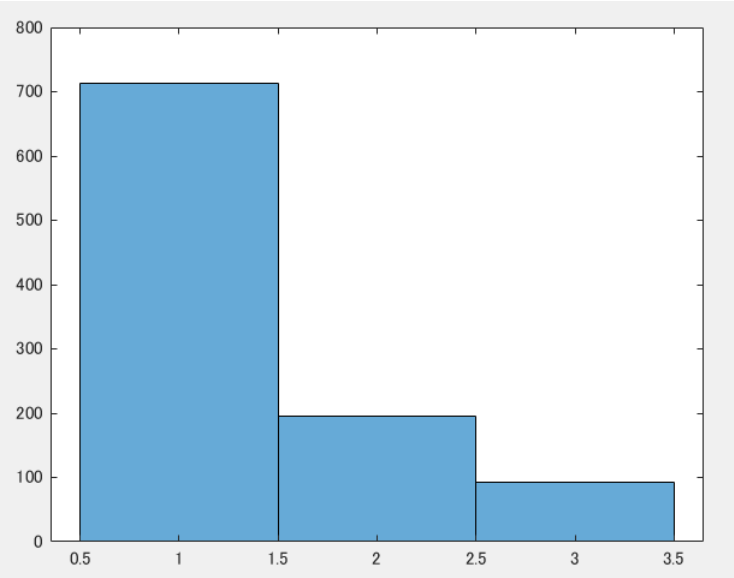

図 4　1000 回遷移させたときの各状態に遷移した回数

演習 3-3

演習 3-2 の遷移する回数と初期状態を変更した以下の条件で実行した際の各確率に関して調べた.

■初期状態 S_2 で 50 回遷移させたとき

$$P(S_1) = 66.0\%$$
$$P(S_2) = 30.0\%$$
$$P(S_3) = 4.0\%$$

■初期状態 S_2 で 1000 回遷移させたとき

$$P(S_1) = 71.2\%$$
$$P(S_2) = 19.6\%$$
$$P(S_3) = 9.2\%$$

■初期状態 S_3 で 50 回遷移させたとき

$$P(S_1) = 66.0\%$$
$$P(S_2) = 28.0\%$$
$$P(S_3) = 6.0\%$$

■初期状態 S_3 で 1000 回遷移させたとき

$$P(S_1) = 71.2\%$$
$$P(S_2) = 19.5\%$$
$$P(S_3) = 9.3\%$$

演習 3-4

演習 2 と演習 3 の結果を見比べてみると，状態遷移を 1000 回遷移させたときの S_1 から S_3 それぞれに遷移する確率が演習 2 で求めた定常確率とほぼ同じである事がわかる．したがって，定常確率 $P(S_1)$ はある時間において，状態 S_1 の確率も表しているとも言えると考える．

また，初期状態を変更したとしても各状態の発生確率はさほど変わらなかったことから，初期状態は発生確率に関係ないことがわかる．

演習 4

演習 4-1

ZIP のアルゴリズムに関して説明する．データ圧縮は一般に「2 段ロケット方式」が用いられている．「2 段ロケット方式」とはその名の通り，1 段目の圧縮と 2 段目の圧縮が存在し，その 2 つをうまく組み合わせることで効率的な圧縮を実現している．

ZIP に関しては，1 段目に「スライド辞書」方式を使用している．辞書と呼ばれるエリアを確保して，その中から事前に出現した文字を探し出し，「その文字列と同じであること」を出力することで文字列を圧縮することができる．1 段目で文字列を圧縮できたので，2 段目には「ハフマン符号」を用いてビット列にする．ハフマン符号とは，可変長のビットを割り当てるアルゴリズムである．出現頻度が多ければ多いほど短い符号に割り当て，その逆の場合は比較的長い符号が割り当てられる．この事によって，文字列をビットに変換し圧縮することが可能である．

ただ，上記のハフマン符号の処理を実現するためには複数の支援機能が必要である．まずは，符号として処理するためにビット単位に変換するための「ビットストリーム」，単位にデータを溜め込んで頻度表を作る「バッファブロック」，そのバッファブロック単位にビット長テーブル (頻度表) を入出力する処理も必要である．

ZIP 圧縮の流れとしては，まずはスライド辞書に読み込まれ，一致文字列と不一致文字列の大きく 2 種類のコードに分解され，バッファにブロックに溜め込まれる．その後，一定量溜め込まれたコードはハフマン符号期によってビットパターンに変換される．そのビットパターンは可変長なので，ビットストリームを使ってバイト単位に出力することで圧縮が完了する．これらの流れを図示したものが図 5 である．

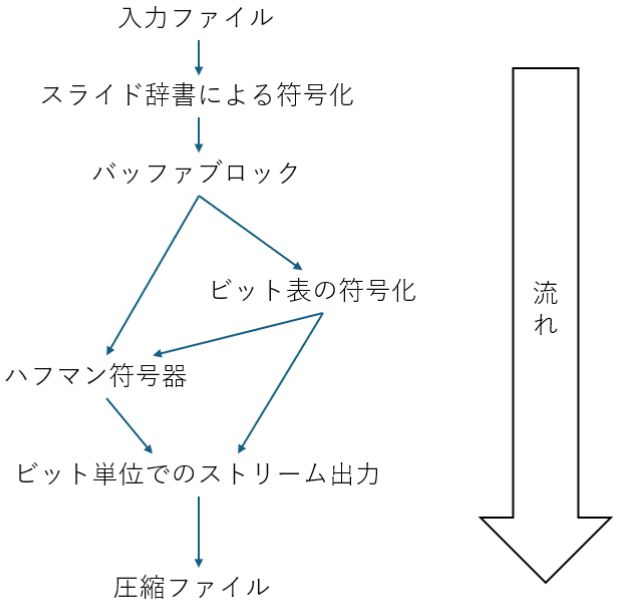

図 5　ZIP 圧縮アルゴリズムのモデル

演習 4-2

用意したファイルとその容量は以下の表 1 の通りである．テキストファイルの文字コードは UTF-8 である．

表 1　用意したファイルとその容量

ファイルの種類	ファイルの容量
.txt	3,204,504 バイト
.jpeg	149,984 バイト
.mp3	74,196,385 バイト

演習 4-3

zip 圧縮は windows 標準搭載のものを使用した．圧縮後の容量をまとめたものが表 2 である．

表 2　用意したファイルとその容量

ファイルの種類	圧縮ファイルの容量
.txt	3,240 バイト
.jpeg	137,581 バイト
.mp3	73,670,682 バイト

演習 4-4

圧縮率の比較を行う．比較した結果を表 3 にまとめた．圧縮率は

$$\text{圧縮率} = \frac{\text{圧縮前のファイル容量}}{\text{圧縮後のファイル容量}}$$

で計算を行った.

表 3　それぞれのファイルの圧縮率

ファイルの種類	圧縮率
.txt	1.011×10^{-3}
.jpeg	0.917
.mp3	0.993

演習 4-5

演習 4-2 から 4 をもとに，それぞれのファイル圧縮率に関して考察を行う．演習 4-1 の ZIP 圧縮のアルゴリズムから，ZIP 圧縮は文字列に対して非常に有効であると考えられる．したがって，テキストファイルの圧縮率が非常に小さいことから，効率的に圧縮が行われていることが伺える．一方，文字列に対して jpg や mp3 の圧縮率は非常に大きく，効率的に圧縮が行われてないと思われる．jpeg も mp3 もすでに圧縮されたファイルであり，文字列ではないので ZIP での圧縮が効率的に行われなかったと考えられる．

追記 (一回消えたのでバックアップからの出力で一部間違いがあるかも)

参考文献

[1] 奥村晴彦 山崎敏,LHA と ZIP：圧縮アルゴリズム×プログラミング入門, ソフトバンク パブリッシング株式会社, 東京, 2003.

[2] SHARP, デジタルカメラで撮影した画像を ZIP 形式に圧縮してもファイルサイズが小さくならない Q＆A情報（文書番号：107790）
https://cs.sharp.co.jp/faq/qa?qid=107790, 参照:20245/19.