

音楽・音声制作演習

第4回 マルコフ連鎖による自動作曲

高岡 明

1. The MidiBusのダウンロードとインストール

The MidiBusはProcessingでMIDIを扱うためのライブラリである：

<http://www.smallbutdigital.com/themidibus.php>

Processingを起動し、“Tools”メニューから“Add Tool...” → “Libraries” → “Sound” を選択するとサウンド関係のライブラリのリストが表示されるので、その中から“The MidiBus”を選択し、“Install”ボタンを押してThe MidiBusをダウンロード、インストールする。

2. コンピュータ・プログラムによる楽曲の自動生成

```
byte data[] = new byte[2];
data[0] = (byte)0xC0; // C0hex = MIDI program change
data[1] = (byte)0; // 0 = Acoustic Grand Piano
mBus.sendMessage(data);

int channel = 0; // 0 <= channel <= 15
int pitch = 64; // MIDI note number: E4 = 64
int velocity = 127; // 0 <= velocity <= 127

for(int start = 0; start < 20; start++) {
    float r = random(0.0, 1.0);

    if(0.0 <= r && r < 0.125) // 0.125 = 1/8
        pitch = 60;
    else if(0.125 <= r && r < 0.25) // 0.25 = 2/8
        pitch = 62;
    else if(0.25 <= r && r < 0.375) // 0.375 = 3/8
        pitch = 64;
    else if(0.375 <= r && r < 0.5) // 0.5 = 4/8
        pitch = 65;
    else if(0.5 <= r && r < 0.625) // 0.625 = 5/8
        pitch = 67;
    else if(0.625 <= r && r < 0.75) // 0.75 = 6/8
        pitch = 69;
    else if(0.75 <= r && r < 0.875) // 0.875 = 7/8
        pitch = 71;
    else
        pitch = 72;

    mBus.sendNoteOn(channel, pitch, velocity); // Send a Midi noteOn
    delay(200);
    mBus.sendNoteOff(channel, pitch, velocity); // Send a Midi nodeOff
    delay(100);
}
}
```

プログラム 1：乱数を使ったピッチ列の生成

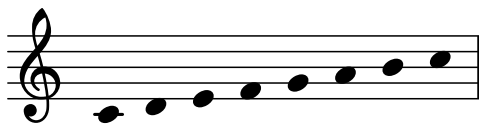
ピッチ	確率
C4	0.125
D4	0.125
E4	0.125
F4	0.125
G4	0.125
A4	0.125
B4	0.125
C5	0.125

表1：“プログラム1”によって出現するピッチの確率

3. コンピュータによる音楽の様式模倣

3.1. 規則の集合による様式の規定： 沖縄風音楽の生成

長音階 Major scale



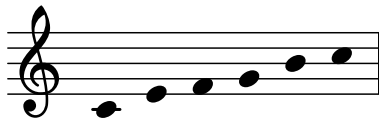
C D E F G A B C
0 2 4 5 7 9 11 0

短音階 Minor scale



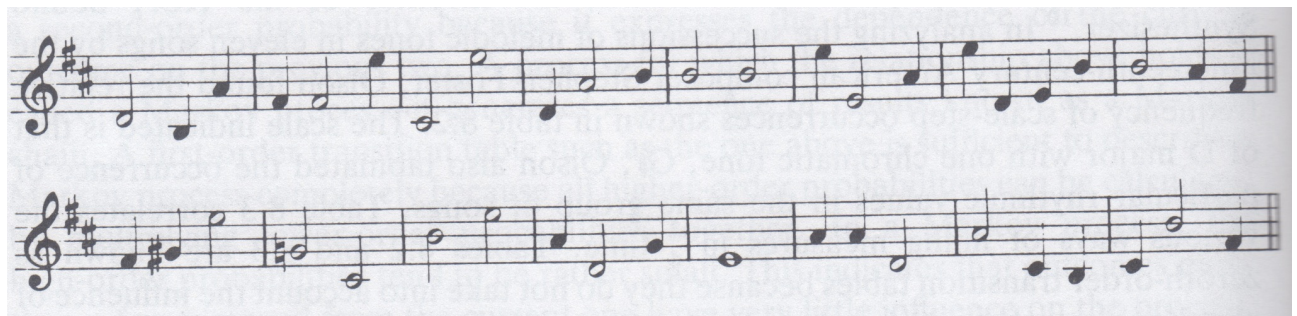
C D E F G A B C
0 2 3 5 7 8 11 0

沖縄音階



C E F G B C
0 4 5 7 11 0

3.2. マルコフ連鎖 Markov chain による様式の規定



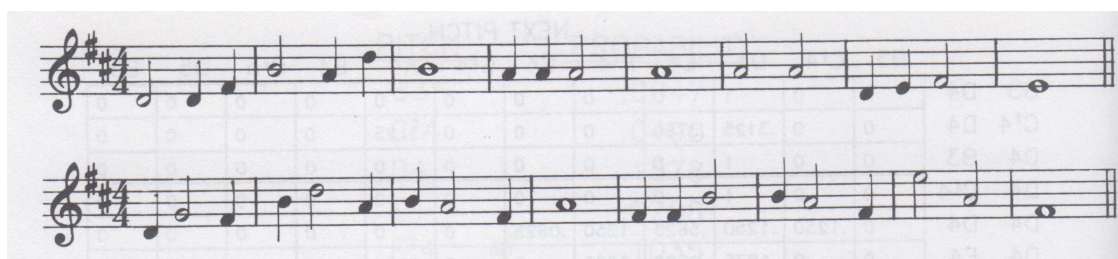
“表2”の出現確率に従うピッチによって自動生成された二つの旋律

ピッチ	確率
B3	0.0047
C#4	0.0490
D4	0.1578
E4	0.0708
F#4	0.1035
G4	0.0626
G#4	0.0463
A4	0.1824
B4	0.1143
C#5	0.0789
D5	0.0816
E5	0.0481

表2：ステファン・フォスターの曲中のピッチの出現確率

		LAST PITCH											
		B3	C#4	D4	E4	F#4	G4	G#4	A4	B4	C#5	D5	E5
NEXT PITCH	B3	0	0	.0625	0	0	0	0	0	0	0	0	0
	C#4	0	0	.0625	.0625	0	0	0	0	0	0	0	0
	D4	1	1	.1250	.3750	.1250	0	0	.0625	.0625	0	0	0
	E4	0	0	.3125	.1875	.2500	0	0	0	0	0	0	0
	F#4	0	0	.1875	.2500	.3125	.2500	0	.3125	.0625	0	0	0
	G4	0	0	.0625	0	.1250	.1875	0	.0625	.0625	0	0	0
	G#4	0	0	0	0	0	0	0	.0625	0	0	0	0
	A4	0	0	.0625	.0625	.1250	.3750	1	.2500	.5625	0	.2500	.3750
	B4	0	0	0	0	.0625	.1875	0	.1875	.1250	.5000	.4375	0
	C#5	0	0	.0625	0	0	0	0	0	0	0	.1875	.6250
	D5	0	0	.0625	.0625	0	0	0	.0625	.1250	.5000	.0625	0
	E5	0	0	0	0	0	0	0	0	0	0	0	0

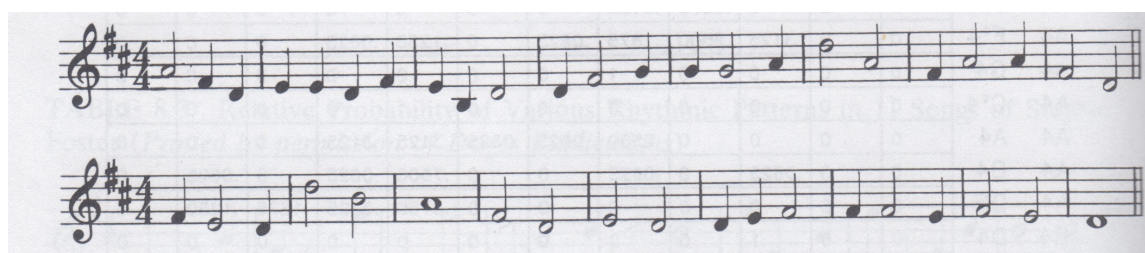
表3：フォスターの旋律の第1階マルコフ連鎖の推移表



“表3”の推移表に従って生成された二つの旋律

		NEXT PITCH											
		B3	C#4	D4	E4	F#4	G4	G#4	A4	B4	C#5	D5	E5
PREVIOUS PAIR OF PITCHES	B3 D4	0	0	1	0	0	0	0	0	0	0	0	0
	C#4 D4	0	0	.3125	.3750	0	0	0	.3125	0	0	0	0
	D4 B3	0	0	1	0	0	0	0	0	0	0	0	0
	D4 C#4	0	0	1	0	0	0	0	0	0	0	0	0
	D4 D4	0	.1250	.1250	.5625	.1250	.0625	0	0	0	0	0	0
	D4 E4	0	0	.1875	.2500	.5000	0	0	.0625	0	0	0	0
	D4 F#4	0	0	0	.4375	.1875	.1250	0	.2500	0	0	0	0
	D4 G4	0	0	0	0	.6875	0	0	0	.3125	0	0	0
	D4 A4	0	0	0	0	.2500	0	0	.7500	0	0	0	0
	D4 C#5	0	0	0	0	0	0	0	0	0	0	1	0
	D4 D5	0	0	0	0	0	0	0	.1250	.6875	.1875	0	0
	E4 C#4	0	0	1	0	0	0	0	0	0	0	0	0
	E4 D4	.0625	0	.0625	.2500	.3125	0	0	.0625	0	.0625	.1875	0
	E4 E4	0	.0625	.7500	.0625	.1250	0	0	0	0	0	0	0
	E4 F#4	0	0	.0625	.1875	.3750	.2500	0	.0625	.0625	0	0	0
	E4 A4	0	0	0	0	0	0	0	.8125	.1875	0	0	0
	E4 D5	0	0	0	0	0	0	0	0	0	1	0	0
	F#4 D4	0	0	0	.7500	.1875	.0625	0	0	0	0	0	0
	F#4 E4	0	.1250	.4375	.1875	.1250	0	0	.0625	0	0	.0625	0
	F#4 F#4	0	0	.1875	.2500	.3750	.1250	0	.0625	0	0	0	0
	F#4 G4	0	0	0	0	.2500	.1875	0	.3750	.1875	0	0	0
	F#4 A4	0	0	0	0	.1250	0	0	.6250	.1875	0	.0625	0
	F#4 B5	0	0	0	0	0	0	0	1	0	0	0	0
	G4 F#4	0	0	0	.5000	0	.5000	0	0	0	0	0	0
	G4 G4	0	0	0	0	0	.5000	0	.5000	0	0	0	0
	G4 A4	0	0	.1250	0	0	0	0	.6250	0	0	.2500	0
	G4 B4	0	0	0	0	0	0	0	1	0	0	0	0
	G#4 A4	0	0	0	0	0	0	0	0	1	0	0	0
	A4 D4	0	0	0	.6875	.3125	0	0	0	0	0	0	0
	A4 F#4	0	0	.3125	.2500	.1875	.0625	0	.1250	.0625	0	0	0
	A4 G4	0	0	0	0	1	0	0	0	0	0	0	0
	A4 G#4	0	0	0	0	0	0	0	1	0	0	0	0
	A4 A4	0	0	0	0	.2500	.0625	.0625	.3125	.3125	0	0	0
	A4 B4	0	0	.0625	0	.0625	0	0	.7500	.0625	0	.0625	0
	A4 D5	0	0	0	0	0	0	0	.3750	.3125	.1875	.1250	0
	B4 D4	0	0	1	0	0	0	0	0	0	0	0	0
	B4 F#4	0	0	0	.6875	.3125	0	0	0	0	0	0	0
	B4 G4	0	0	0	0	0	0	0	0	1	0	0	0
	B4 A4	0	0	.0625	0	.5625	.0625	0	.1250	.0625	0	.1250	0
	B4 B4	0	0	0	0	.1250	0	0	.7500	0	0	.1250	0
	B4 D5	0	0	0	0	0	0	0	.5625	.1250	.3125	0	0
	C#5 B4	0	0	0	0	0	0	0	1	0	0	0	0
	C#5 D5	0	0	0	0	0	0	0	0	.3750	0	0	.6250
	D5 A4	0	0	0	0	.8750	0	0	.1250	0	0	0	0
	D5 B5	0	0	0	0	0	.0625	0	.3125	.3750	0	.2500	0
	D5 C#5	0	0	0	0	0	0	0	0	.7500	0	.2500	0
	D5 D5	0	0	0	0	0	0	0	0	1	0	0	0
	D5 E5	0	0	0	0	0	0	0	.3125	0	.6875	0	0
	E5 A4	0	0	0	0	0	0	0	1	0	0	0	0
	E5 C#5	0	0	0	0	0	0	0	0	0	0	1	0

表4：フォスターの旋律の第2階マルコフ連鎖の推移表



“表4”の推移表に従って生成された二つの旋律

4. マルコフ連鎖の推移表の作成

1) 出現ピッチを書き出す：C4, G4, A4, F4, E4, D4, 四分休符



2) 表を作成する：

現在のピッチ

次のピッチ

	C4	G4	A4	F4	E4	D4	休符
C4							
G4							
A4							
F4							
E4							
D4							
休符							

3) 曲を構成する各ピッチについて、曲の最初から順番に現在のピッチ（Last pitch）が次に進行するピッチ（Next pitch）の出現回数を表に記入する：

現在のピッチ

次のピッチ

	C4	G4	A4	F4	E4	D4	休符
C4	2	0	0	0	0	2	1
G4	2	4	2	0	0	0	2
A4	0	2	2	0	0	0	0
F4	0	2	0	4	0	0	2
E4	0	0	0	4	4	0	0
D4	0	0	0	0	4	2	0
休符	1	2	0	0	0	2	0

4) 各ピッチと休符の出現回数の合計を各ピッチの列の最下段に記入する：

現在のピッチ

次のピッチ

	C4	G4	A4	F4	E4	D4	休符
C4	2	0	0	0	0	2	1
G4	2	4	2	0	0	0	2
A4	0	2	2	0	0	0	0
F4	0	2	0	4	0	0	2
E4	0	0	0	4	4	0	0
D4	0	0	0	0	4	2	0
休符	1	2	0	0		2	0
出現総数	5	10	4	8	8	6	5

5) 各ピッチについて、次のピッチへの進行回数を出現総数で割って次のピッチへの確率を計算する：

現在のピッチ

次のピッチ

	C4	G4	A4	F4	E4	D4	休符
C4	2/5	0	0	0	0	2/6	1/5
G4	2/5	4/10	2/4	0	0	0	2/5
A4	0	2/10	2/4	0	0	0	0
F4	0	2/10	0	4/8	0	0	2/5
E4	0	0	0	4/8	4/8	0	0
D4	0	0	0	0	4/8	2/6	0
休符	1/5	2/10	0	0		2/6	0
出現総数	5	10	4	8	8	6	5

6) この推移表をプログラムに書く：

```
import themidibus.*; //Import the library

MidiBus mBus; // The MidiBus object
int instrument = 0; // MIDI program change: 0 = Acoustic Grand Piano

void setup() {
  size(200,200);
  background(0);
  MidiBus.list(); // lists available synthesizers
  mBus = new MidiBus(this, -1, "Microsoft GS Wavetable Synth");
  noLoop();
}

void draw() {
  background(0);

  byte data[] = new byte[2]; // status byte + data byte
  data[0] = (byte)0xC0; // status byte: C0hex = MIDI program change for channel 0
  data[1] = (byte)0; // data byte: 0 = Acoustic Grand Piano
  // data[1] = (byte)55; // 55 = Orchestra Hit
  mBus.sendMessage(data); // sends the MIDI program change message
  print("\n");

  int channel = 0; // 0 <= MIDI channel <= 15
  int nextPitch = 60; // MIDI note number: C4 = 60
  int velocity = 96; // 0 <= MIDI velocity <= 127

  for(int start = 0; start < 48; start++) {
    print(nextPitch + " ");
    mBus.sendNoteOn(channel, nextPitch, velocity); // Send a Midi noteOn
    delay(200);
    mBus.sendNoteOff(channel, nextPitch, velocity); // Send a Midi noteOff
    delay(100);

    int lastPitch = nextPitch;
    float r = random(0.0, 1.0); // random numbers between 0.0 and 1.0

    if(lastPitch == 60) { // 60 = C4
      if(0.0 <= r && r < 0.4) { // 2/5
        nextPitch = 60;
        velocity = 96;
      } else if (0.4 <= r && r < 0.8) { // 2/5
        nextPitch = 67;
        velocity = 96;
      } else { // 1/5
        velocity = 0;
      }
    } else if(lastPitch == 67) { // 67 = G4
      if(0.0 <= r && r < 0.4) { // 4/10
        nextPitch = 67;
        velocity = 96;
      } else if(0.4 <= r && r < 0.6) { // 2/10
        nextPitch = 69;
        velocity = 96;
      } else if(0.6 <= r && r < 0.8) { // 2/10
        nextPitch = 65;
        velocity = 96;
      } else {
        velocity = 0;
      }
    }
  }
}
```

```

    }
    } else if(lastPitch == 69) { // 69 = A4
        if(0.0 <= r && r < 0.5) { // 2/4
            nextPitch = 67;
            velocity = 96;
        } else { // 2/4
            nextPitch = 69;
            velocity = 96;
        }
    } else if(lastPitch == 65) { // 65 = F5
        if(0.0 <= r && r < 0.5) { // 4/8
            nextPitch = 65; // 65 = F4
            velocity = 96;
        } else { // 4/8
            nextPitch = 64; // 64 = E4
            velocity = 96;
        }
    } else if (lastPitch == 64) { // 64 = E4
        if(0.0 <= r && r < 0.5) { // 4/8
            nextPitch = 64; // 64 = E4
            velocity = 96;
        } else { // 4/8
            nextPitch = 62; // 62 = D4
            velocity = 96;
        }
    } else if (lastPitch == 62) { // 62 = D4
        if(0.0 <= r && r < 0.3333) { // 2/6
            nextPitch = 60; // 60 = C4
            velocity = 96;
        } else if (0.3333 <= r && r < 0.6667) { // 2/6
            nextPitch = 62; // 62 = D4
            velocity = 96;
        } else { // 2/6
            velocity = 0;
        }
    } else { // a rest
        if(0.0 <= r && r < 0.2) { // 1/5
            nextPitch = 60; // 60 = C4
            velocity = 96;
        } else if (0.2 <= r && r < 0.6) { // 2/5
            nextPitch = 67; // 67 = G4
            velocity = 96;
        } else { // 2/5
            nextPitch = 65; // 65 = F4
            velocity = 96;
        }
    }
}
print("\n");
}

```