Models and Algorithms for Music Generated by Physiological Processes

Jaime Alonso Lobato-Cardoso and Pablo Padilla-Longoria

This science [mathematics] is the easiest. This is clearly proved by the fact that mathematics is not beyond the intellectual grasp of anyone. For the people at large and those wholly illiterate know how to draw figures and compute and sing, all of which are mathematical operations.

Roger Bacon, c.1265

Abstract Generative art emphasizes processes. On the other hand, mathematical models are used to understand underlying biological, economic, physical or social phenomena (among others). The outcome of such models can be considered as processes in their own right. For instance, physiological processes give rise to a wide variety of signals which can, in turn, be detected by changes in pressure, temperature, electrical potential and so on. When measured and converted with an appropriate transducer, they constitute the raw material which algorithms and models may translate into sound. In this paper we explore a mathematical model of the human circulatory system based on differential equations. We then use this model as a generator of melodic and rhythmic structures in a compositional multimedia context.

1 Introduction

Generative art is focused on processes and not only on the outcome. This is probably the reason why an algorithmic approach in creative disciplines has become more and more a subject of interest. On the other hand, the possibility of controlling in a very precise way the context and parameters with which such algorithms are

J. Alonso Lobato-Cardoso (✉)
SEMIMUTICAS-IIMAS, UNAM, Cd. Universitaria, Cto. Escolar 3000, Coyoacán, D.F., Mexico
e-mail: jaimelobatocardoso@gmail.com

P. Padilla-Longoria
IIMAS, UNAM, Cd. Universitaria, Cto. Escolar 3000, Coyoacán, D.F., Mexico
e-mail: pabpad@gmail.com

G. Pareyon et al. (eds.), *The Musical-Mathematical Mind*,
Computational Music Science, DOI 10.1007/978-3-319-47337-6_16

being implemented provides flexibility in manipulating the basic material an artist has at his disposal. Here is not the appropriate place to discuss in detail the trends and most representative positions in this matter (for such a discussion, including an introduction about the circulatory system as an endosymbiotic context for music, see [2]). However, we would like to discuss here what in our opinion might constitute a significant and interesting line of creative work. More specifically, we would like to discuss the role mathematical models might have as process generators in their own right. It is rather natural to think that the resulting description of a real process by means of a mathematical model, in the form of a simulation for instance, can be used in the same way as other algorithms.

In recent years the use of biosensors in the context of multimedia art has attracted lots of attention. Rather than obtaining data or information in general from direct measurements, we propose to explore the use of mathematical models of physiological processes as sources of data for creating compositional tools complementing or interacting with information obtained by means of such sensors. In this paper we limit ourselves to a relatively simple mathematical model of the human circulatory system, which is given by ordinary differential equations. Then we solve these equations numerically obtaining two time series, for the circulatory pressure and for the blood flow respectively. Later on we transfer these data generated in MATLAB to Supercollider, in order to manipulate them and construct rhythmic and melodic structures. We also automatically generate a score with this material. We would like to point out that what we are presenting here is not a final work or piece of electroacoustic music, but rather a compositional tool that can be used to generate sound material in different compositional and sonification contexts.

2 The Model

Dynamical systems models have traditionally been used to study and understand natural processes. In particular, they constitute a suitable methodological framework to deal with the evolution of systems with time. Quite often the model leads to a system of differential equations. In a physiological model this is rather complicated and involves not only the dynamical understanding of the variables, but also stability, robustness, control and feedback aspects. In order to simplify our presentation we consider a model for the blood pressure and blood flow in the human circulatory system. It is derived and explained in detail in [1], Sects. 1.11 and 1.12. Here we write down the corresponding equations

$$C_{sa}\dot{P}_{sa} = Q_{A_0} - P_{sa}/R_s, \tag{1}$$

where $P_{sa}(t)$ is the arterial pressure, C_{sa} is the systemic arterial compliance, Q_{A_0} is the outflow from the left ventricle through the aortic valve into the systemic arterial tree and is a given periodic function of time, each period being a heart beat (see Fig. 1). Finally R_s is the systemic resistance.

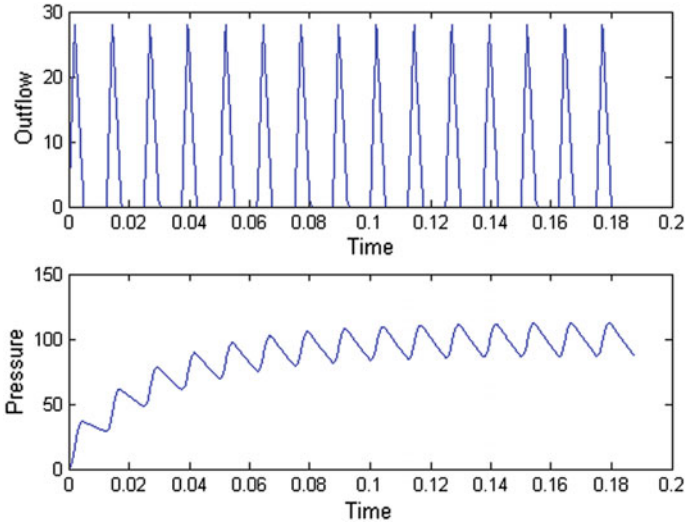

Fig. 1 Typical simulation

3 Numerical Implementation

The previous equation is solved by the Euler's method. Let us observe that in other applications a more accurate implementation—for instance the use of Runge–Kutta methods—would be desirable, but for the present purposes our choice suffices. For the sake of completeness we include the MATLAB code below. It is essentially the same as that presented in the above mentioned reference and the modifications are related to the interface and sending data to Supercollider.

```
%filename: sa.m
clear all %clear all variables
clf      %clear all figures
global T TS TMAX QMAX;
global Rs Csa dt;
in_sa %initialization
for klok=1:klokmax
    t=klok*dt;
    QAo=QAo_now(t);
    Psa=Psa_new(Psa,QAo), %new Psa overwrites old
    %Store values in arrays for future plotting:
    t_plot(klok)=t;
    QAo_plot(klok)=QAo;
    Psa_plot(klok)=Psa;
end
```

```
%Plot results in one figure
%with QAo(t) in upper frame
%and Psa(t) in lower frame
subplot(2,1,1), plot(t_plot,QAo_plot)
subplot(2,1,2), plot(t_plot,Psa_plot)

function Psa=Psa_new(Psa_old, QAo)
%filename Psa_new.m

global Rs Csa dt;
Psa=(Psa_old+dt*QAo/Csa)/(1+dt/(Rs*Csa));
end

function Q=QAo_now(t)
%filename: QAo_now.m

global T TS TMAX QMAX;
tc=rem(t,T); %time elapsed since
% the beginning of the current cycle
%rem(t,T) is the remainder when t is divided by T
if (tc<TS)
    %SYSTOLE:
    if (tc<TMAX)
        %BEFORE TIME OF MAXIMUM FLOW:
        Q=QMAX*tc/TMAX;
    else
        %AFTER TIME OF PEAK FLOW:
        Q=QMAX*(TS-tc)/(TS-TMAX);
    end
else
    %DIASTOLE:
    Q=0;
end


end


% filename: in_sa.m (initialization for the script sa)
T=0.0125     %Duration of the heartbeat (minutes)
TS=0.0050    %Duration of the syastole  (minutes)
TMAX=0.0020  %Time at which flow is max (minutes)
QMAX=28.0    %flow through aortic valve (liters/minute)
Rs=17.86     %Systemic resistance (mmHg/(liter/minute))
Csa=0.00175  %Systemic arterial compliance (liters/(mmHg))
```

```
%This value of Csa is approximate and will need adjustment
%to make the blood pressure be 120/80.
dt=0.01*T        %Time step duration (minutes)
%This choice implies 100 timesteps per cardiac cycle.
klokmax=15*T/dt  %Total number of timesteps
%This choice implies simulation of 15 cardia cycles.
Psa=0
%Any intial value is OK here.
%Initialize arrays to store data for plotting:
t_plot=zeros(1,klokmax);
QAo_plot=zeros(1,klokmax);
Psa_plot=zeros(1,klokmax);
Psa_plot_red=zeros(1,klokmax);
```

4 Generation of Musical Structures

This numerical implementation in Matlab gives us as a result 1500 numbers describing a circulatory process. We send these numbers from Matlab to Supercollider via User Datagram Protocol (UDP) and keep them in a matrix. Then we apply mathematical and logical operations to obtain different melodic contours. In fact we use only a few specific transformations, but we could have applied any function to the elements of the array. Below we only give a few of these functions as typical examples:

Mapping using a linear transformation: The original data range 0.12450207–110.27082 is transformed into an audible range in MIDI numbers 60–72, using a subset of this array by

Selection of elements:

(a) Selecting elements whose indices lie within a pre-established range, for example from 100 to 250, 150 elements of the array, which constitute 10.
(b) Discard elements with even or odd indices (50).
(c) Discard a percentage of elements of the array randomly.

Tuning:

(d) Consider only the integer part of each element in the matrix.
(e) Select some ranges taking into account the decimal part, which enables us to generate a microtonal pitch profile.

Transformation range:

(f) Allow only a certain number repeated values.

Representing the contrapuntal derivations:

(g) Melodic inversion, retrograde, retrograde inversion, augmentation, diminution.

Articulation:

(h) Take some items as rests.

5 Compositional Application

This implementation is a compositional tool applied to the creation of melodic contours but it also can be used to structure a piece more focused on timbral developments or on spacialization parameters. The way in which we map the matrix to musical or sonic values is going to suggest more adequate musical forms relating the material generated by the system. Here is the corresponding code in Supercollider which receives the data generated by the model and computed using MATLAB.

```
~laLista=[];
(
~lis = OSCresponder(n, "/test", {|...msg|
~dato = msg[2][1].postln;
~laLista = ~laLista.add(~dato);
if(~laLista.size == 1500, {
~lista10 = ~laLista.collect({arg item, i; (item / 5) + 60});
~lista11 = ~lista10.collect({arg item, i; item.asInteger});
~itembuf = 0;
~lista12 = ~lista11.collect({arg item, i;
if(item == ~itembuf, {item = 0}, {item; ~itembuf = item;});
});
~lista13 = ~lista12.reject({arg item, i; item == 0});
~lista14 = ~lista13.collect({arg item, i;
if(i.odd, {~sel = [0,1].choose; if(~sel == 1, {\r}, {item})}, {item});

});
p = Pdef(\parti,
Pbind(
\chan,  0,
\midinote, Pseq(~lista14, inf),
    \dur, Pwrand([0.25, Pn(0.125, 2)], #[0.8, 0.1], inf),
    \legato, sin(Ptime(inf) * 0.5).linexp(-1, 1, 1/3, 3),
\amp, Pwrand([1, 0.5, 0.25, 0.125], [0.1, 0.8, 0.05, 0.05], inf)
)
).play;
m = SimpleMIDIFile( "~/Sangre/Midis/Prueba06.mid" );
m.init1( 2, 120, "4/4" );
m.fromPattern( p );
m.write;
});
}
).add;
)
```

6 Conclusions

In this paper we have presented a compositional platform based on mathematical models. The solutions of the dynamical systems and their corresponding simulations have been used to generate different musical structures. We hope it may be useful as

a starting point example in the development of musical compositions in which real time interaction and setting of the context via adjustment of the parameters of the model, can be included as an important part of the compositional process.

References

1. Hoppensteadt, F.C., Peskin, Ch.S.: Modeling and Simulation in Medicine and the Life Sciences. Texts in App. Math. Springer, New York (2004)
2. Pareyon, G.: On Musical Self-Similarity. The International Semiotics Institute, Imatra-Helsinki (2011)