

# *SANTIAGO* - A Real-time Biological Neural Network Environment for Generative Music Creation

Hernán Kerlleñevich, Pablo E. Riera, Manuel C. Eguía

Laboratorio de Acústica y Percepción Sonora, Universidad Nacional de Quilmes  
R. S. Peña 352. Bernal. 1876. Buenos Aires. Argentina.

{hk,pr,me}@lapso.org

<http://lapso.org>

**Abstract.** In this work we present a novel approach for interactive music generation based on the dynamics of biological neural networks. We develop *SANTIAGO*, a real-time environment built in Pd-Gem, which allows to assemble networks of realistic neuron models and map the activity of individual neurons to sound events (notes) and to modulations of the sound event parameters (duration, pitch, intensity, spectral content). The rich behavior exhibited by this type of networks gives rise to complex rhythmic patterns, melodies and textures that are neither too random nor too uniform, and that can be modified by the user in an interactive way.

**Keywords:** generative music, biological neural networks, real-time processing

## 1 Introduction

Complex systems offer an outstanding opportunity to work in generative music. One important message that came out from the study of complex systems is that very simple rules can lead to complex behavior, starting both from an ordered or from a completely disordered state. In this way, complex systems emerge as a new state that supersedes the dichotomy between order and disorder, being regular and predictable in the short term but unpredictable in the long run [6].

It is interesting to trace an analogy with what happens in generative music, where the key element is the system to which the artist cedes partial or total subsequent control [5]. Incorporating complex systems into generative music potentially allows the emergence of a variety of stable and unstable time structures, that can create expectancies in the listener and also deceive them, in a similar way to what happens with tension-distension curves in composed music [8].

Among complex systems, biological neural networks are probably the ones that exhibit the most rich behavior. In fact, the brain is the most complex device that we can think of. In its simplest realization a neural network consists of units (neurons) connected by unidirectional links (synapses). The units are characterized by its (voltage) level that can be continuous or discrete. We distinguish

between artificial neural networks, where units are highly simplified models and the complexity resides in the network, and biological (or biophysical) neural networks where the model neuron is fitted to adjust the records obtained in real biological neurons and has a rich intrinsic dynamics.

There are some previous works that incorporate the behavior of neural networks into rules for generative music [3]. The most disseminated tool has been artificial neural networks, that have been applied to music in many ways, ranging from pitch detection [17], musical style analysis [16] and melody training [4] to composition purposes [14]. Most of these works were based on the idea that complex behavior emerges from the network only, disregarding the intrinsic dynamics of each neuron unit. In fact, in artificial neural networks the units are modeled with a single continuous or two-state variable.

However recent studies on the behavior of neural networks have shown that the intrinsic dynamics of the neurons plays a major role in the overall behavior of the system [11] and the connectivity of the network [2]. In this work we take into account the complexity emerging from the intrinsic dynamics of the units and from the connectivity of network as well. In order to do this, we adopt biologically inspired models for the neurons, that produce spikes, which are considered the unit of information in the neural systems. In simple terms, a spike is a short-lasting event in which the voltage of the neuron rapidly rises and falls. A review of some biological neuron models can be found in [10].

Also, previous works treated the networks as input/output systems, biased with the task-oriented view of artificial neural networks. In contrast, our work includes all the events (spikes) from the network as relevant information, as small neuron networks are supposed to do. Among the many realizations of small neural networks in the animal realm, we take those that are able to generate rhythmic behavior, with central pattern generators (CPGs) as a prominent example. CPGs are small but highly sophisticated neural networks virtually present in any animal endowed with a neural system, and are responsible for generating rhythmic behavior in locomotion, synchronized movement of limbs, breathing, peristaltic movements in the digestive tract, among others functions [7]. In this CPG view, the individual spikes can be assigned to events in our musical generative environment (such as notes) that can lead to complex rhythmic behavior.

In addition to rhythm generation, spike events can trigger complex responses in neural systems [13], and modulate dynamical behavior. In this spirit, we also use biological neural networks to modify the pitch, spectral content, duration and intensity of the generated events.

Finally, we adopt excitatory-inhibitory (EI) networks as the paradigmatic network. EI networks arise in many regions throughout the brain and display the most complex behavior. Neurons can be either excitatory or inhibitory. When a spike is produced in an excitatory (inhibitory) neuron, this facilitates (inhibits) the production of spikes in the neurons that receive the signal of that neuron through a synapse.

The realization of the ideas expressed above is *SANTIAGO*, a real-time biological neural network environment for interactive music generation. It is named

after Santiago Ramón y Cajal (1852-1934) the spanish Nobel laureate, histologist and psychologist, considered by many to be the father of modern neuroscience.

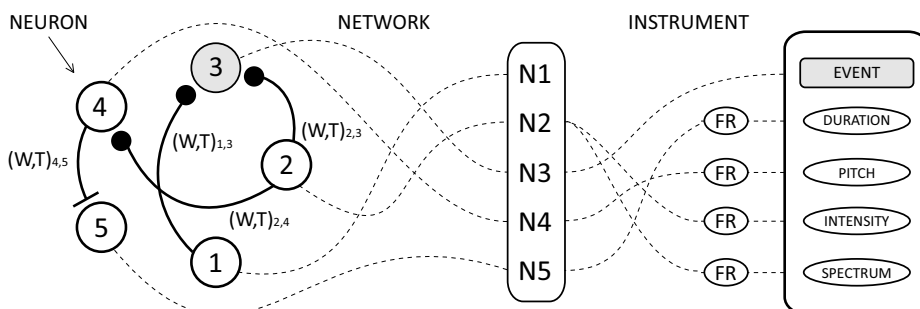
The article is organized as follows. In section 2 we expose the concept and design of *SANTIAGO*. In section 3 we show some results. In section 4 we present our conclusions.

## 2 *SANTIAGO* description

### 2.1 Overview

Starting with a simple yet biologically relevant spiking neuron model, our environment allows to build up neural networks that generates a musical stream, creating events and controlling the different parameters of the stream via the action of individual spikes or some time average of them (firing rate).

*SANTIAGO* is designed with a highly flexible and modular structure. It is possible to build small networks that only creates rhythmic patterns, where each neuron generates events (spikes) of a certain pitch, intensity and timbre (see the example in Section 3), and progressively add more neurons or other small networks that control the pitch, intensity or spectral content of the events generated by the first network. Also, different temporal scales can be used in different sub-networks, thus allowing the spike events to control from long term structures to granular synthesis.



**Fig. 1.** Diagram of a simple realization of *SANTIAGO*. Five neurons are connected forming a small EI network (left). Neuron 1 excites neuron 3, neuron 2 excites neuron 3 and 4, and neuron 4 inhibits neuron 5. Each synapse is characterized by its synaptic weight  $W$  and time delay  $T$ . Instrument terminals N1-N5 map the output of each neuron (dotted lines) to events or event parameters (right). The notes (events) of this instrument will occur when N3 fires a spike, the pitch of the note will be determined by the averaged firing rate (FR) of N4, the durations of the notes by the FR of N5, and the intensity and the spectral content will be controlled by the FR of N2.

In fig. 1 we display a diagram of a very basic realization of our environment. Excitatory neurons project excitatory synapses (black dots) to other neurons

and inhibitory neurons project inhibitory synapses (bars) to other neurons. Each synapse is characterized by its weight  $W$  (how strong is its action) and some time delay  $T$ , which in real synapses corresponds to the physical length of the link (axon). The EI network will have a complex behavior as a result of the intrinsic dynamics of the neurons, which depends on its internal parameters, and the connectivity of the network (connectivity matrix). The user can control all these parameters in real time in order to obtain a desired result or explore a wide diversity of behaviors.

The outputs of this network are the spikes generated by all the neurons (N1-N5). These events are injected into the Instrument which has the main responsible to map the spike events to sound.

As we mentioned in the previous section, neurons of the network can act as event generators or modulation signals. In the first case, every spike triggers an event or note. In the second case an averaged quantity, the firing rate (number of spikes in a fixed time window), is calculated and mapped to a certain sound event parameter.

## 2.2 Implementation

*SANTIAGO* is basically built as a set of abstractions for Pd-Extended (version 0.41.1) [15]. It is completely modular and works under different platforms. It also uses the Gem external for neural activity visualization[1].

The current version implements the neuronal model with the *fexpr~* object [18], which permits access to individual signal samples, and by entering a difference equation as part of the creation argument of the object the output signal gives the integrated dynamical system. The environment runs flawlessly with 10 neurons, using 60% of CPU load on a Intel Core2Duo 2.2 Ghz machine. In the present we are implementing the model with external objects developed specially for this application that will increase the number of possible units.

OSC communication has been implemented with *mrpeach* objects, enabling live music creation across local networks and the internet. On the other hand, the network capability can be used for parallel distributed computing, when the number of neurons is high or the desired output demands too much load for a single machine. For this purpose, when the main patch is loaded, the user can establish a personal Machine ID number, which will be sent as a signature for every message and event generated from that computer. Finally, all streamed data is easily identifiable and capable of being routed and mapped dynamically.

In this version, the inputs of the environment are: keyboard, mouse and MIDI controllers. The outputs include real-time MIDI controllers and OSC messages, multichannel audio and visuals.

A single-neuron version of *SANTIAGO* (SOLO) was also created, for the user to explore the possibilities of one neuron only, and its different firing modes.

### 2.3 Architecture

The implementation of *SANTIAGO* is still in development. Yet it is very functional even for complex interactions. The environment is built upon a modular structure, which is described next. The core of the system consists of three main structures: neurons, network and instruments. Each structure has a window, accessible from the Main Panel.

**Neurons** This structure allows the creation of *neuron modules*, which constitute the minimal units of the system. The implemented neuron model it was originally proposed by Izhikevich[9], as a canonical model capable of display a large variety of behaviors. It is described by a system of two differential equations (Eqs. 1a and 1b) and one resetting rule (Eq 1c)

$$\frac{dv}{dt} = 0.04v^2 + 5v + 140 - u + I(t) \quad (1a)$$

$$\frac{du}{dt} = a(bv - u) \quad (1b)$$

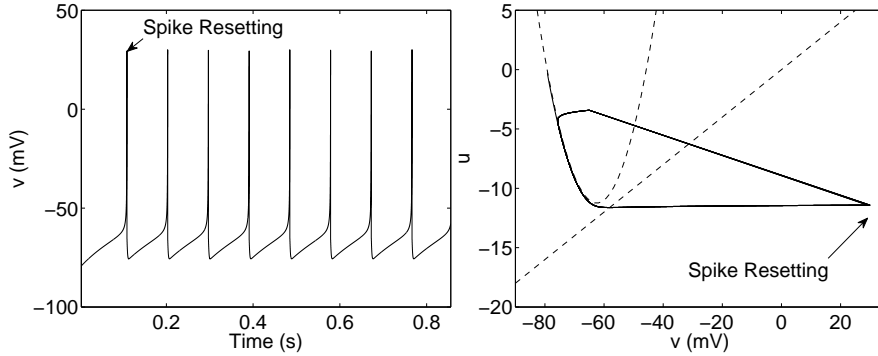
$$if v \geq 30 \text{ mV}, \text{ then } \begin{cases} v \leftarrow c \\ u \leftarrow u + d \end{cases} \quad (1c)$$

The dynamical variables correspond to the voltage membrane of the neuron  $v$  and a recovery variable  $u$ . There are four dimensionless parameters ( $a$ ,  $b$ ,  $c$ ,  $d$ ) and an input current  $I(t)$ . The spike mechanism works by resetting variables  $v$  and  $u$  when the voltage reaches some fixed value. Figure 2 shows the neuron spikes in the  $v$  variable and the phase portrait of the two dynamical variables. The second plot also presents the nullclines of the system, *i. e.* the curves where only one of the differential equations is equal to zero.

The current  $I(t)$  includes all the synaptic currents. The neuron may receive synaptic inputs from other neurons. When one of these neuron fires a spike, after some time delay  $T$ , a post-synaptic current pulse is added to  $I(t)$ : for an excitatory (inhibitory) synapse a step increase (decrease) of size  $W$  is followed by an exponential decay towards the previous current value with a time constant  $t_s$  (which is also an adjustable parameter). In addition to the synaptic currents,  $I(t)$  includes other currents that can be controlled by the user: a DC current, a zero-average noisy current, and excitatory or inhibitory pulses.

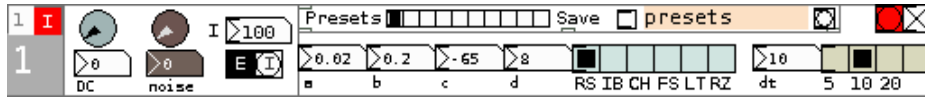
This model has the advantage of being both computationally cheap and versatile. With simple adjustment of its parameters, it can reproduce several known neuronal spiking patterns such as bursting, chattering, or regular spiking. For a complete description of the possible behaviors see [10].

The user can generate any number of neuron modules (figure 3), only limited by CPU capability. Each module has local input currents, and can also be affected by a global current and noise level sent from the main panel. The user can choose between six presets of firing modes or manually set the model parameters. Every neuron module has a particular integration time, allowing different



**Fig. 2.** Neuron membrane voltage trace  $v$  (left) and phase portrait  $u, v$  (right). The dashed lines represent the nullclines of the dynamical system and the full line is the periodic spiking orbit showing a limit cycle.

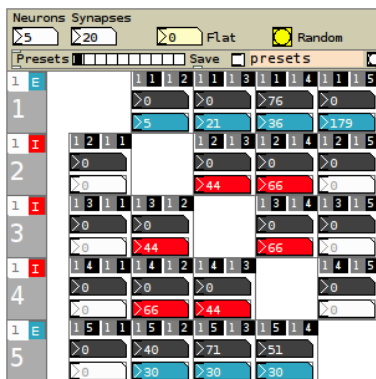
*tempos* for equal firing modes and parameter settings. There is also a presets manager for saving and loading stored combinations of parameters.



**Fig. 3.** Neuron Module. The numbers in the left indicate MachineID and Instrument number respectively. There are two sub-modules: local input current and neuron parameters with a preset manager.

**Network** This structure allows designing and configuring the network by selecting synaptic coupling between neurons, and the delays. In the left column the user can select the neuron type, switching between excitatory (E) and inhibitory (I) post-synaptic pulses. The central module of this structure is the *connectivity matrix* (figure 4). Delays are set in the dark gray number boxes and synaptic weight have color depending on the E/I status, so at a glance is easy to grasp the network status. A synaptic weight equal to zero means that there is no synaptic connection. There is number box (flat) for setting the same synaptic weights for all the connections, and also a random set up generator.

**Instruments** In this structure the user can map the neural activity to sound event generation and sound event parameters. Each instrument has basic parameters the user may expect to control. In the example shown in figure 5, Instrument 1 of Machine (ID) 1 has four sub-modules mapping: rhythm, duration, pitch and intensity. Every one of this has a Machine ID number and a Neuron

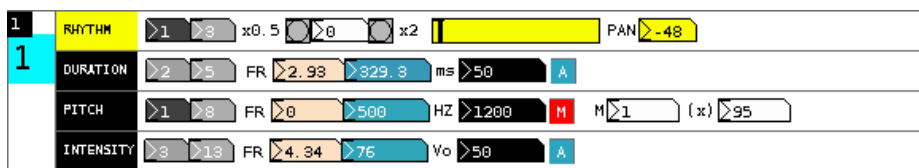


**Fig. 4.** Network connectivity matrix. This structure allows designing the network by selecting synaptic gains (blue and red) and delays (dark gray) between neurons. In the left column the user switches between excitatory and inhibitory postsynaptic pulse for every neuron.

number from which spikes and firing rates will be mapped. Both variables are by default initialized with the same Machine ID and Instrument number. (A spatialization sub-module (not shown) allows the instrument to control the distribution of events in a multichannel system.) The initial setup this instrument module works with a basic synth oscillator or sends MIDI messages.

When the parameters are switched to manual (M) the user can fix the sound event parameters to any desired value. When switched to automatic (A), it maps the firing rate of the input neuron to a modulation of the corresponding sound event parameter. In the pitch case, the firing rate value is used for frequency modulation, with a movable offset and configurable depth. Intensity varies between 0 and 1 for each instrument, and the overall volume is set from the main panel.

Santiago has currently two visualization panels where network activity is depicted in real-time: spike view and event view. In spike view all the individual traces of  $v$  are plotted in rows. The event view shows vertical lines in rows (one for each neuron), corresponding to spikes, discarding sub-threshold activity.

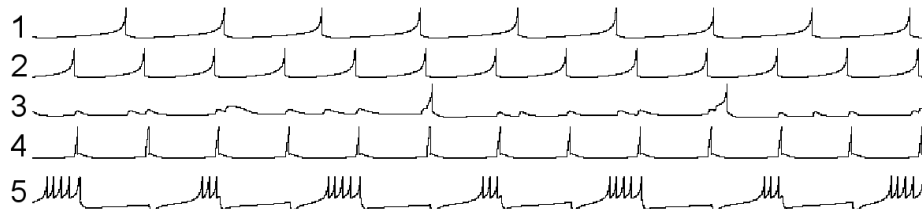


**Fig. 5.** Instrument module. There are four sub-modules that map rhythm, duration, pitch and intensity from neurons.

### 3 Network Activity

There are several ways to set up a network, depending on the desired goal. One possible way is to build simple feed-forward circuits that perform basic logic operations, such as coincidence detection (integrator neuron) or filtering intervals (resonator neuron). Other ways are to use mutual inhibition or excitation, or closed loops in order to achieve collective oscillations, like in CPG networks or coupled oscillators. Here we present three different examples: a simple network with a coincidence detector, an elementary CPG and a small network more oriented to obtain musical output.

As our basic example we used the same network displayed in figure 1. In this circuit, neuron 3 detects concurrent spikes from 1 and 2. Neuron 4 acts as an inhibitory inter-neuron for neuron 2 and inhibits the repetitive bursting pattern from neuron 5. The outcome of this network is the repetitive pattern displayed in figure 6). The length of the cycle varies depending on the parameters used.

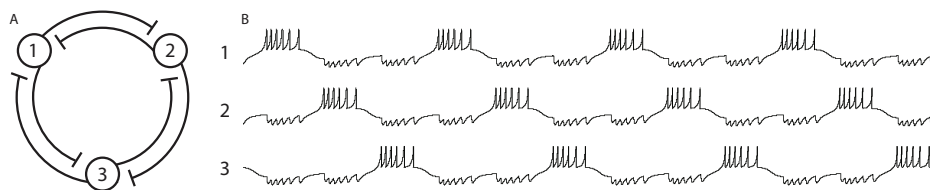


**Fig. 6.** Spike view of the network displayed in figure 1. Neurons 1 and 2 are Regular Spiking (RS) neurons, and both excite neuron 3, also RS. When their spikes are close enough in time, they make neuron 3 spike, acting as a coincidence detector. Neuron 4 is a Low Threshold neuron excited by neuron 2 and spikes every time neuron 2 does. At the same time, it inhibits 5 which has a chattering behavior.

We also tested whether our environment could reproduce some elementary CPGs, such as the inhibition network studied in [12]. *SANTIAGO* was able to reproduce the characteristic sustained oscillations patterns of the model. In figure 7 we show the burst of three chattering neurons with mutual inhibition. This example is particularly interesting to use with a spatialization module, because we can have travelling patterns in space.

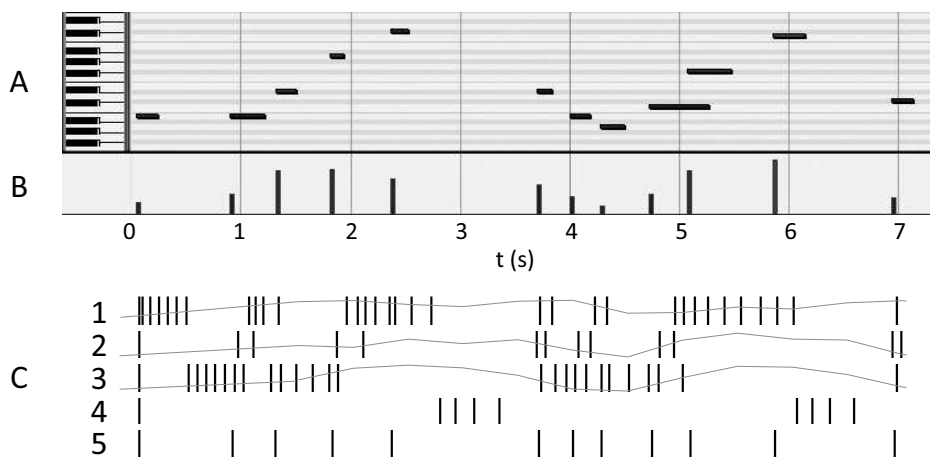
In our last example, we designed a network intended to depict a complex behavior in the sense mentioned in the introduction. Further, we were interested in obtaining, as an output of a single instrument associated to that network, a simple musical phrase with melodic and dynamic plasticity. For this purpose, we devised a small, but densely connected network of three chattering inhibitory neurons and two excitatory neurons (one chattering and one regular spiking) which produced a nearly recurring temporal pattern that shows up with slight variations on each cycle, as displayed in figure 8. The notes were triggered by neuron 5. The length of the notes was controlled by the firing rate of neuron 1,





**Fig. 7.** Network diagram (A), and spike view (B) of a fully connected inhibition network working as a CPG. Notice the rhythm and periodicity of the pattern.

the intensity with the firing rate of neuron 2 and the pitch with neuron 4. The firing rate is computed with a time window of two seconds, inserting a delay between the events onset time and the corresponding firing rate computation.



**Fig. 8.** An example of a musical score generated by a five-neuron network. Piano roll (A) and note velocity (B) show data received through a virtual MIDI port into an external sequencer. Event view (C) of the five-neuron network. Events from neuron 5 were selected to trigger the notes. The duration, intensity and pitch were modulated with the firing rate of neurons 1, 2 and 3 respectively. The delayed firing rates are plotted as thin lines in the corresponding row

## 4 Conclusions

We presented a real-time environment for generative music creation, based on the intrinsic dynamics of biological neural networks. From this perspective, several mapping strategies were presented towards a composition-performance technique. *SANTIAGO* is a system that allows the user to compose and perform

music in which every sound event and parameter may interact with the others. The presence of an event may alter the whole network and a change in a bigger scale may be reflected in a micro level too. This brings together the essence of complex systems and music performance. We can think of small musical elements with simple rules which scale to another level when they interact. For more references, project updates, and audible examples go to <http://lapso.org/index.php?group=07santiago>.

## References

1. Danks, M.: Real-time image and video processing in Gem. In: Proceedings of the International Computer Music Conference, pp. 220223. International Computer Music Association (1997)
2. Destexhe, A. and Marder, E. Plasticity in single neuron and circuit computations *Nature* 431, 789-795 (2004)
3. Eldridge, A. C.: Neural Oscillator Synthesis: generating adaptative signals with a continuous-time neural model.
4. Franklin, J.A.: Recurrent neural networks for music computation. In: *INFORMS Journal on Computing*, vol. 18, num. 3, pp. 312. (2006)
5. Galanter, P.: What is generative art? Complexity theory as a context for art theory. In: GA2003–6th Generative Art Conference, (2003)
6. Gribbin, J. *Deep Simplicity: Bringing Order to Chaos and Complexity*, Random House (2005).
7. Hooper, S. L. : *Central Pattern Generators. Embryonic ELS* (1999).
8. Huron, D. *Sweet anticipation: Music and the psychology of expectation*. MIT Press (2008).
9. Izhikevich, E. M.: Simple model of spiking neurons. In: *IEEE Transactions on Neural Networks*, vol. 14, num. 6, pp. 1569-1572. IEEE (2004)
10. Izhikevich, E. M.: *Dynamical systems in neuroscience: The geometry of excitability and bursting*. The MIT press (2007)
11. Kocho, K. and Segev, I. The role of single neurons in information processing. *Nature Neuroscience* 3, 1171 - 1177 (2000)
12. Matsuoka, K.: Sustained oscillations generated by mutually inhibiting neurons with adaptation. In: *Biological Cybernetics*, vol. 52, num. 6, pp. 367-376. Springer (1985)
13. Molnár, G. et al. Complex Events Initiated by Individual Spikes in the Human Cerebral Cortex. *PLoS Biol* 6(9): e222 (2008).
14. Mozer, M.C.: Neural network music composition by prediction: Exploring the benefits of psychoacoustic constraints and multi-scale processing. In: *Connection Science*, vol. 6, num. 2, pp. 247-280 (1994)
15. Puckette, M. S. *Pure Data*. Proceedings, International Computer Music Conference. San Francisco: International Computer Music Association, pp. 269272 (1996)
16. Soltau, H. and Schultz, T. and Westphal, M. and Waibel, A.: Recognition of music types. In: Proceedings of the 1998 IEEE International, vol. 2, pp. 1137-1140. IEEE (2002)
17. Taylor, I. and Greenhough, M.: Modelling pitch perception with adaptive resonance theory artificial neural networks. In *Connection Science*, vol. 6, num. 6, pp. 135-154 (1994)
18. Yadegari, S.: Chaotic Signal Synthesis with Real-Time Control - Solving Differential Equations in Pd, Max-MSP, and JMax. In: Proceedings of the 6th International Conference on Digital Audio Effects (DAFx-03). London (2003)