Practical session

Musical applications of Pisano sequences and logistic difference equations

1 Introduction

This practical session uses software called Pure Data, which is a flexible programming environment for creating all sorts of real-time audio applications. Pure Data (or PD, for short) is free to use, share, and modify. There are PD versions for MacOS, Windows, and many Linux distributions – so you can take the software home to play with on your own time if you would like to.

The Apple computers in each booth are connected to M-Audio Ozonic keyboards with 3 octaves of piano keys, an audio interface (built into the Ozonic keyboard), and a pair of Genelec 8030 monitor speakers. It should be noted that the equipment in the booths has the potential to damage your hearing, please take care to make sure the volume is at a comfortable level! To adjust the volume, use the vertical sliders on the left of the Ozonic keyboard.

2 Pisano sequence software

The computers should already be logged with PureData running, with the Pisano sequence patch `01_MAIN_single.pd` already loaded. The display should look similar to Figure 1 below:

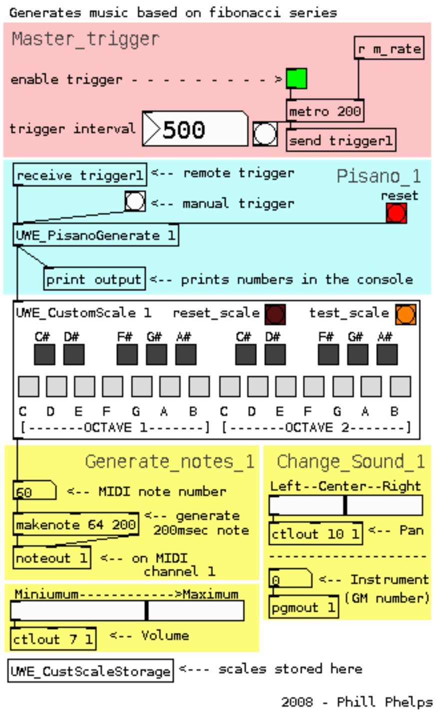


Figure 1 - Single channel Pisano patch

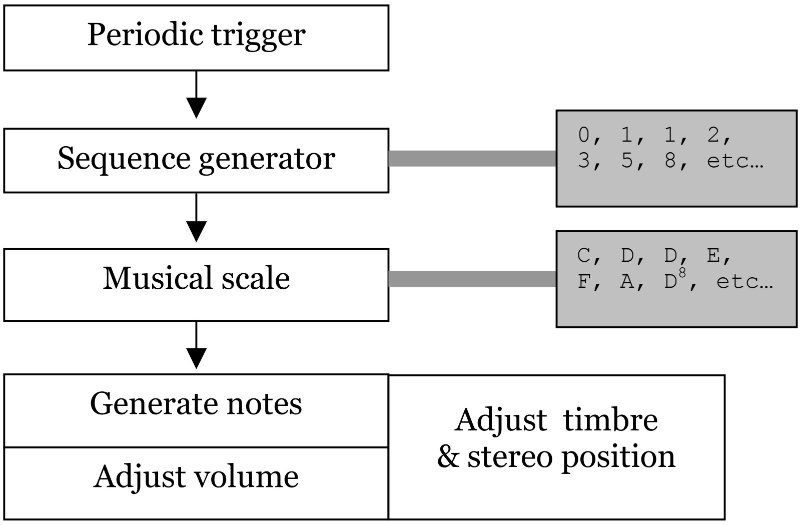

Periodic trigger
Sequence generator
0, 1, 1, 2, 3, 5, 8, etc…
Musical scale
C, D, D, E, F, A, D^8, etc…
Generate notes
Adjust volume
Adjust timbre & stereo position

Figure 2 - Block diagram

This patch works in the following way:

- Periodic triggers are generated in the red area at the top, labelled "`Master_Trigger`"
- These are sent to the sequence generator in the light blue area "`Fibonacci_1`"
- The white area with the keyboard maps the Fibonacci numbers onto notes in a musical scale. (E.G. by setting `C=0, E=1, G=2`, for a C major triad).
- The yellow areas allow you to specify the timbre, volume, and left-right stereo position of the notes produced.

The main components are also illustrated as a block diagram in Figure 2.

To familiarise yourself with the components in the patch, let's run through a few exercises, starting on the next page:

Exercise 1 – The master trigger section

In the red "Master_trigger" area at the top:

- Click the green box (called a PD [toggle]) above the [metro] object
- An X will appear in the toggle, indicating that it is switched on
- The box just below the [metro] with a circle inside (called a PD [bang] button) should begin flashing at periodic intervals

- Click once inside the box (called a PD [number] object) displaying the number "500"
- Type a different number, and press enter
- Depending on the number entered, you will see the [bang] flash slower or faster.
- You can also alter the value inside, by clicking and dragging the mouse up and down. This works for any [number] object

The [metro] object is essentially a metronome that generates pulses with a period specified in milliseconds. Note that one second is equal to 1,000 milliseconds.

Exercise 2 – The custom scale generator

This practical session requires a bit of musical knowledge, so the following section is included in order to explain a few things about musical scales and how to transpose them. It will also provide some example scales in a table.

Example Scales

The subject of musical scales is a vast topic; the table on the next page only lists a few scales. One of the things you will hopefully notice during today's session is that the difference between the same sequence of notes played in two different scales can be striking.

Each scale listed in the table has a repeated note at the octave, so the numbers in the interval column always add up to 12. This is not always the case however (and this is another vast and interesting area of music theory!) Some musical systems such as the Arab-tone system have 24 intervals in an octave, and the 72-TET system divides the octave up into 72 intervals.

An example image is provided for each scale to help you program the scale into the software. The images show the scale in the "root key" of C, but you can use the "interval" column to transpose this easily as in the example box below. Simply start from your new root note (for example, G#), and use the interval column in the table to work out the other notes in your scale.

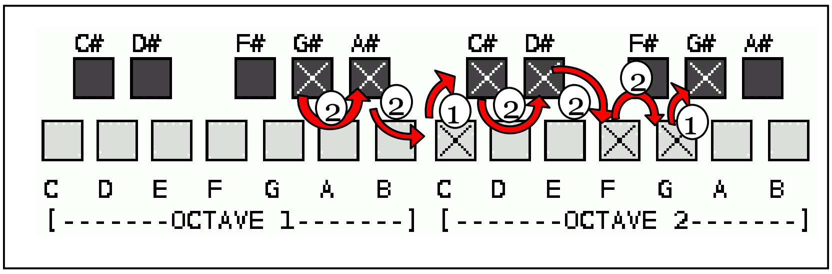

C# D# F# G# A# C# D# F# G# A#
2 2 1 2 2 2 1
C D E F G A B C D E F G A B
[-------OCTAVE 1-------] [-------OCTAVE 2-------]

Figure 3 - G# major (with 2,2,1,2,2,2,1 interval overlay)

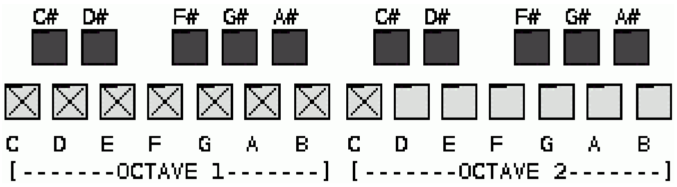
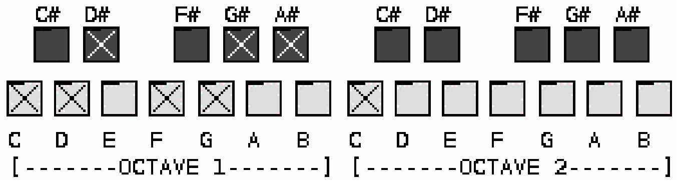
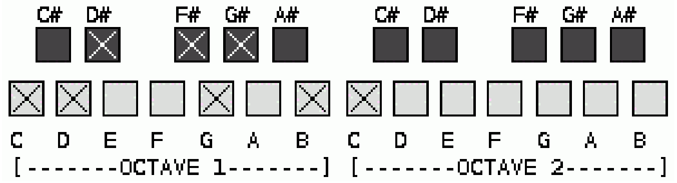
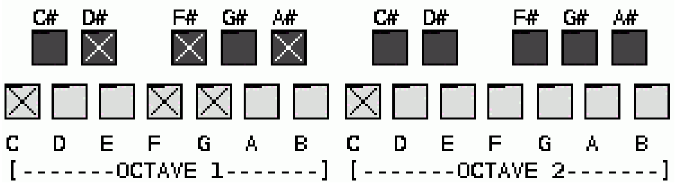
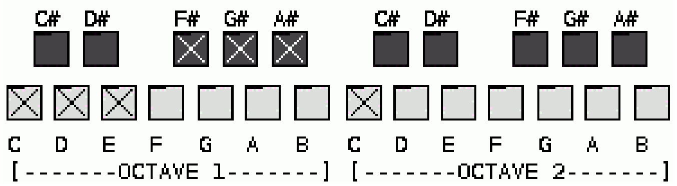
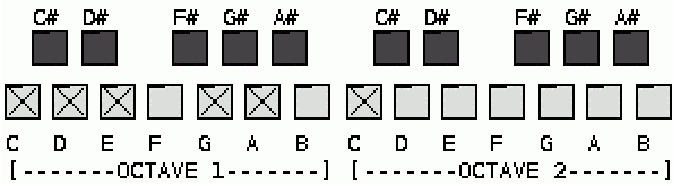
Scale name	No. of notes	Intervals	Scale Image
Major	8	2,2,1,2,2,2,1	
"Natural" Minor	8	2,1,2,2,1,2,2 *Note the same sequence of intervals as in a major scale but shifted.*	
"Hungarian" Minor	8	2,1,3,1,1,3,1,	
Blues Mode	7	3,2,1,1,3,2	
Wholetone	7	2,2,2,2,2,2	
Pentatonic	6	2,2,3,2,3	

Table 1 – Example scales

Exercise 2 – The custom scale generator

In the white area, third from the top:

- Push the red [bang] labelled "reset scale" (Get into the habit of resetting the scale each time you program a new one
- Program a C major scale (at the top of the table) by copying the scale image column
    - IMPORTANT: Make sure you click the notes from left-to-right! The software is easily confused and will produce strange results if you click the boxes in a different order.
    - If you think you've made a mistake, you should click the "reset scale" [bang] and start again.
- When you've finished, push the orange [bang] marked "test scale"
- Check that the sound comes out of the speakers at a comfortable volume
    - Remember that the equipment has the potential to damage your hearing
    - You can adjust the volume by using the sliders to the left of the Ozonic keyboard
- In the light blue "Pisano_1" area, second from the top:
    - Click the red "reset" [bang] (It is important to get into the habit of resetting the Fibonacci generator right after you have finished programming a scale)

- In the red “Master_trigger” area, at the top:
    - Set the trigger interval to a high number (try 1,000 for example)
    - Enable the the trigger [toggle]
    - The numbers from the Fibonacci generator are now being mapped to the C major scale and will repeat until you turn the trigger [toggle] off again.

Make a mental note of how a C major voicing sounds. How many notes are there before the sequence repeats?

- Stop the master trigger [toggle]
- Push the reset [bang] in the white custom scale area
- Program in a C minor scale (2nd scale from the top of the table)
    - Remember to program in left-to-right order!
- Push the reset [bang] in the light blue Pisano_1 area
- Run the generator (by enabling the master trigger [toggle] in the red area)

Did you notice that the number of notes in the major and minor scales is identical (8), so the Pisano series produced has the same period? **The number of notes in a scale determines how many notes are in a Pisano-generated phrase, before the musical phrase repeats.**

Using your knowledge of Pisano periods (or the handout which lists them all, depending on how lazy you are!), how many notes should a scale with 7 notes produce?

Exercise 2.1 - Changing the instrument

Before moving on, and whilst the C minor Pisano sequence is still playing, try using the number box in the yellow area shown in Figure 4. Click then type a number, or drag up and down to select different synthesised instruments for the notes to play back on. Table 2 is also included to show how Instruments in the General MIDI (GM) specification are grouped into families by their “program change number”:

PGM	Family	PGM	Family
1-8	Piano	65-72	Reed
9-16	Chromatic Percussion	73-80	Pipe
17-24	Organ	81-88	Synth Lead
25-32	Guitar	89-96	Synth Pad
33-40	Bass	97-104	Synth Effects
41-48	Strings	105-112	Ethnic
49-56	Ensemble	113-120	Percussive
57-64	Brass	121-128	Sound Effects

Table 2 – GM instrument list (from http://www.midi.org/about-midi/gm/gm1sound.shtml)

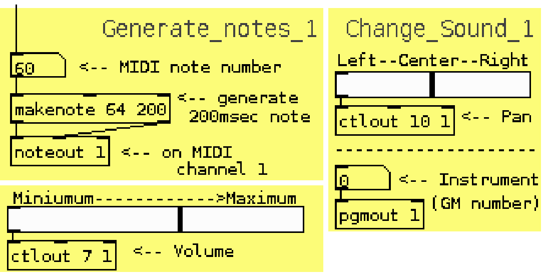


Figure 4 – Changing sound parameters

Exercise 3 – Transposing scales

The scale table includes an interval column, this can be very useful for programming scales in different tonalities to that pictured in the “scale image” column.

The general approach to transposing a scale using the table is to use the intervals to calculate each note. If you make a mistake, you should click the red “reset scale” `[bang]` and start again. It may also help to look at Figure 3 (on page 2).

Try transposing a C blues scale into an A blues scale:

- Click the “A” `[toggle]` in octave 1
- The first interval is a “3” – so move up 3 semitones to click on the “C” in octave 2
- The next interval should be a “2” – so move up 2 semitones to the “D” in octave 2
- …carry on in this manner until you reach the last note (which should be an A in octave 2)
- Test the scale
- Reset the Fibonacci generator in the light blue area
- Enable the master trigger to begin listening

How many notes are there in this sequence? Was your prediction correct - about the Pisano period for a scale with 6 notes (like a blues scale)?

On the next page, we will experiment with using two generators side-by-side – to hear combinations of scales with different periods.

3 Investigation – Two scales, side-by-side

For this part of the session, the objective is to use your knowledge of Pisano periods and musical scales to investigate how two or more sequences interact in a musical way (either in a pleasing way, or deliberately dissonant; the choice is yours!).

To begin experimenting with more than one Pisano sequence simultaneously:

- Load the `02_MAIN_dual.pd` patch
  - Close the “`01_MAIN_single.pd`” patch
    - If a message asks you about discarding changes, click “Yes” to discard.
  - Use the menus to select File->Open
  - Select the `02_MAIN_dual.pd`
  - A new patch (as in Figure 5 below) should appear

This investigation is not like the previous guided exercises: it is completely up to you what direction you take in your exploration. That said, here are a few suggestions for starting points and reminders about how the software works.

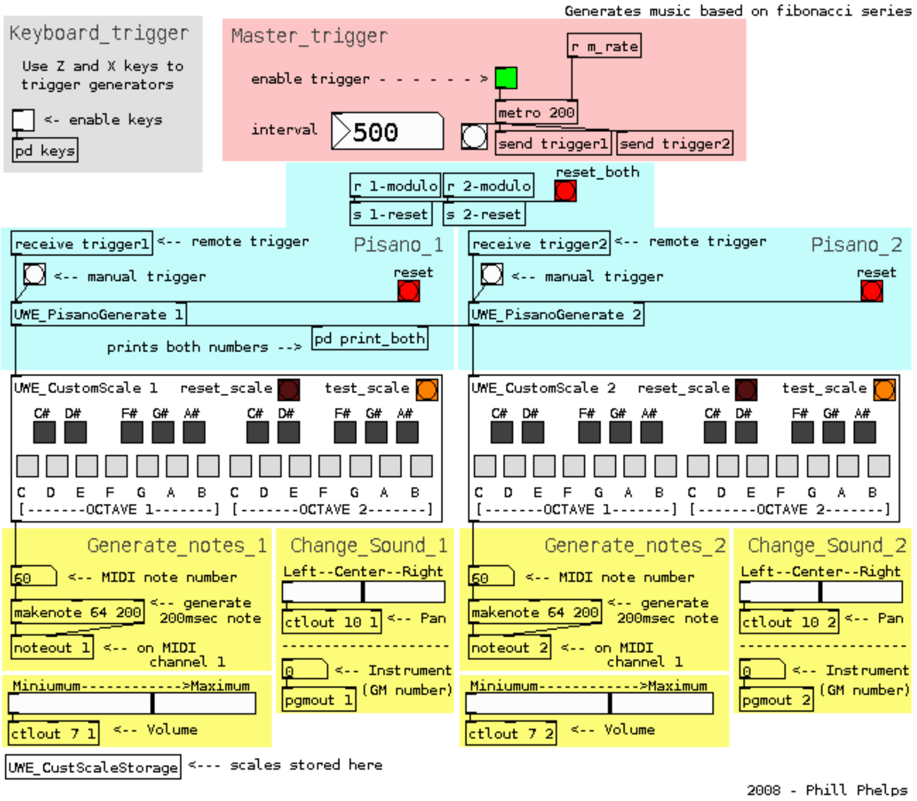


Figure 5 – Dual channel Pisano sequence patch

The dual channel patch has a few new features, the red “reset both” `[bang]` above the two light blue Pisano areas, and the `Keyboard_trigger` area, which allows you to press the Z and X keys on to trigger the left and right Pisano generators respectively.

The “reset both” `[bang]` is for resetting both generators, should they become out of sync due to triggering one more frequently than the other using the keyboard trigger facility.

3.1 Investigation – pointers, reminders, and hints

- Remember to program notes in the scale area from left to right
  - If you make a mistake, you must click the “reset scale” [bang] and start again
- Remember to reset the Pisano generators after the scale has been programmed
  - There are separate reset buttons if you deliberately want to make the sequence generators go out of sync, and a “reset both” button to re-sync both at the same time

- Try experimenting with a pair of scales which will produce the same sequence length, to see which musical scales go nicely together.
  - You can program any scale, they don’t have to be listed in the table
  - Remember that the number of notes in a scale directly determines the sequence length
  - Notice that the interval sequence of minor and major scales are transposed – which is where the terms “relative major” and “relative minor” come from. (Hint!)
- You can change the instrument sounds and stereo position to help aurally separate two sequences that are closely related

- Try experimenting with scales with radically different periods too

- If you get really adventurous, you can try the `03_MAIN_ AltRhythm.pd` patch (similar to Figure 5 – but with new section Figure 6)

- Try experimenting with different periods, scales, and rhythms all at once!

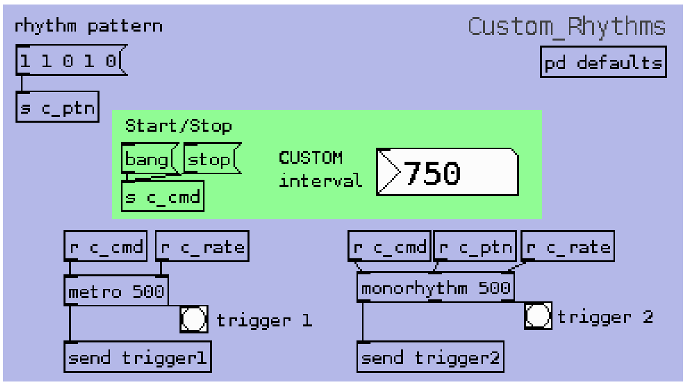


Figure 6 – Additional rhythm generator

This additional rhythm generator allows you to specify an alternative trigger pattern for the second Pisano sequence on the right. By default, it plays a triplet pattern (specified by the message box with the numbers “`1 1 1`”, indicating three beats in a row) but you can edit this to whatever you like:

For example “`1 1 0`” would play a triplet rhythm with a silent third beat, and “`0 1 0 1`” would play a double time beat with syncopated accents on the off beats.

- Remember that the keyboard triggers can work in combination with the automatic triggers, you can create interesting accents by triggering notes with your keyboard.

- The `04_MAIN_triple.pd` patch allows you to freely experiment with wildly complex combinations of rhythms, periods, scales, and chords (by using the keyboard trigger to strike all three generators simultaneously).

Don’t forget to take home your CD ROM! – it contains Mac and Windows versions of PureData, the patches you’ve been using today, videos, and audio examples. If you have any problems, please email zenpho@zenpho.co.uk with your questions or feedback.

4 Logistic difference equation investigation

The patch `01_MAIN_LogisticEquation.pd`, in the `Logistic` directory is a patch that converts the numbers generated by a form of the logistic difference equation, into audible tones and MIDI note triggers. Figure 7 shows a screenshot of the patch.

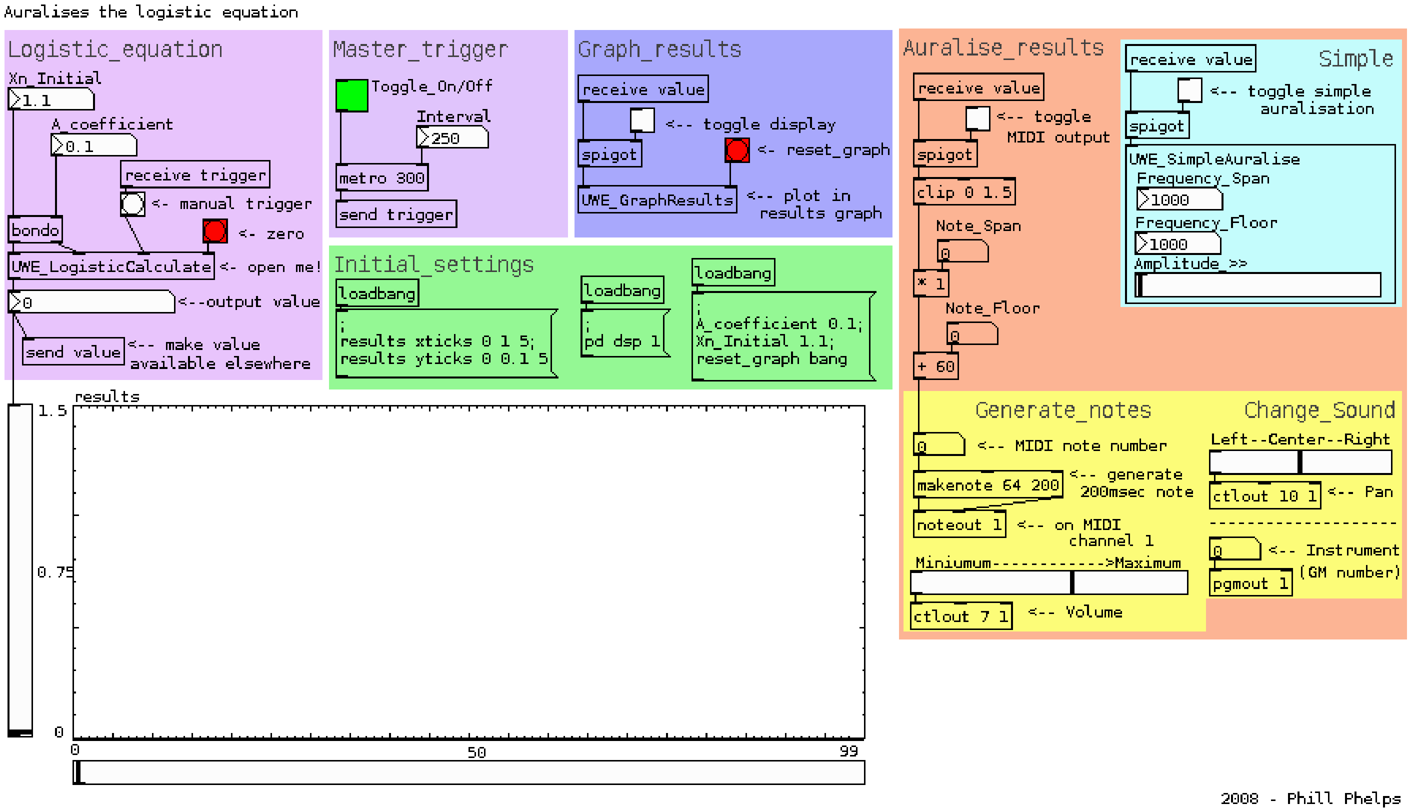


Figure 7 – Logistic equation patch

The `master_trigger` section should be familiar, as should the yellow `generate_notes`, and `change_sound` sections near the bottom right.

Some hints to get you started:

- Click the green master trigger `[toggle]`, and the `[number]` object labled “`output value`” should start updating.
- Next, click the `[toggle]` in the dark blue `graph_results` area to begin updating the large results graph in the lower left of the patch.
- Without turning off the automatic trigger, while the graph is still updating; gradually increase the value of the `[number]` object labled “`A_coefficient`” either by dragging up and down or by clicking and entering numbers
  - Note down the point where the behaviour exhibits period doubling (i.e. the output begins to oscillate between two values)

It can be difficult to see clearly on the graph when the first period doubling point occurs. This is partly due to the limited resolution of the screen, but your ears can be much more sensitive than an LCD monitor. Follow these steps to hear the numbers converted into tones:

- In the light blue `simple_auralise` area, click the `[toggle]` labled "`toggle simple auralisation`"
- Drag the amplitude `[slider]` object to the right to begin hearing the tones
- Some people's ears are more sensitive to differences in pitch than others, it may help to play around with the frequency span and floor values
    - These values scale the logistic difference equation numbers (ranging from zero to ~1.5) into frequencies in Hz by shifting the zero point up to a range within human hearing and by scaling very small differences between successive numbers into larger jumps.

The `simple_auralise` area might not sound very musical. This is because converting into absolute frequencies produces the tones, and these converted frequencies will not (often) lie on any western musical scale. One way to make the output sound more musical is to convert the numbers from the logistic difference equation into MIDI note-numbers in the western 12-TET system. The yellow areas should be familiar components for achieving this goal, to begin hearing the numbers converted to MIDI:

- First turn down the amplitude slider in the light blue `simple_auralise` area
- Turn on the `[toggle]` object labelled "`toggle MIDI output`" in the red area
- The numbers from the difference equation often have a decimal point, whilst the MIDI system uses integers between zero and 127.
    - Try using the `note_span` and `note_floor [number]` objects to scale the numbers so that they fit inside the MIDI note number range
- Try combining the sound of the MIDI instruments with the `simple_auralise` tones
- Remember you can alter the GM instrument number that the notes are played back on

If you feel really adventurous, you can open the `UWE_LogisticCalculate` object by double clicking it. Inside, you can re-program the difference equation to have a different structure. You can do this whilst the program is running, and whilst all tones are being generated; since PureData is a completely real-time modular programming language.

Don't forget to take home your CD ROM! – it contains Mac and Windows versions of PureData, the patches you've been using today, videos, and audio examples. If you have any problems, please email zenpho@zenpho.co.uk with your questions or feedback.