Practical session

Musical scales triggered by rotating disks, Pisano sequences and a Whitney music box

1 Introduction

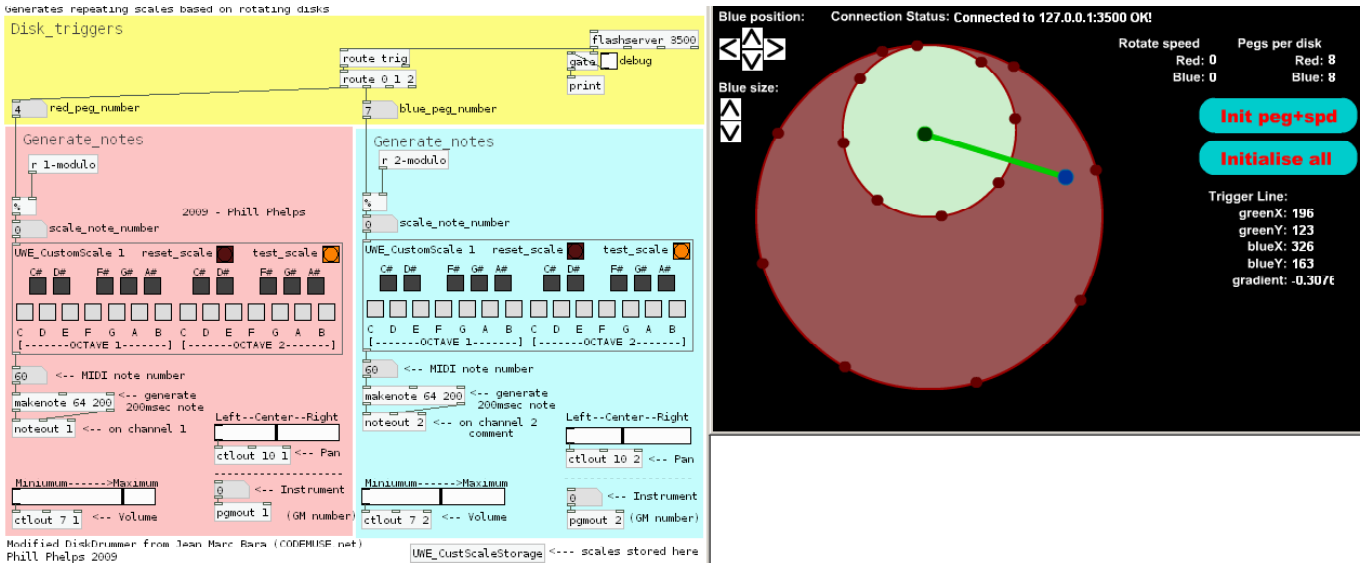
The object of today's session is to explore the idea of algorithmic music by programming a computer to generate a series of musical notes "all by itself". Algorithmic music involves designing a set of rules (i.e. the algorithm) and instructing the computer to follow these rules in order to generate music from scratch. It is the blending of mathematics and music.

One of the things you will see for yourself today is that the rules do not have to be very complex before it becomes difficult for the ear to pick out the underlying structure. Visual cues are often very useful when designing or adjusting the various parameters. For this reason; a common theme in this session will be to use rotating disks with pegs around the circumference representing individual musical notes.

Before we begin, please note 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!

2 Rotating disks and musical scales

The computers should already be logged in. The display should look similar to Figure below:



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. The PD patch is on the left, and an interactive Flash animation is on the right.

Here's a brief explanation of the way the patch works:

- Periodic triggers are generated in Flash when pegs on the rotating disks cross the green line
- Each peg on each disk is numbered from zero, Flash communicates these numbers to PD
- The peg numbers can be mapped onto notes in a scale (E.G. by setting C=0, D=1, E=2, etc)

Obviously, there's more to it than just this simple explanation, so to familiarise yourself with the various components, let's look at some exercises.

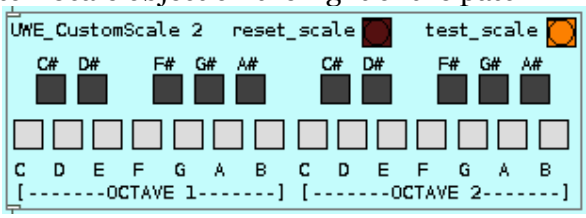
Exercise 1 – Numbering notes in scales
This practical session requires a bit of musical knowledge, so this first exercise is a little wordy in order to explain a few things about musical scales. Even if you are comfortable with these musical structures, it is worth reading though this exercise to understand how the software represents, stores, and manipulates scales.

- Programs in PureData are defined by linking objects together with virtual cables
- A collection of these objects linked together is called a "patch"

Most objects in PD are plain rectangles with their name inside, the [UWE_CustomScale] object is different in that it has a strip of controls arranged like a piano keyboard inside. We will use this to map peg numbers to specific notes in a musical scale.

Let's try programming a scale (using the table on the next page):

- Use the blue UWE_CustomScale object on the right of the patch



- Push the dark red "reset_scale" to begin
  - Try to get into the habit of resetting the scale before programming a new one

- Program a major scale (using the table on the next page)
  - IMPORTANT: Make sure you click the notes from left-to-right! The software is easily confused and will not number the notes correctly if you click the boxes in a different order
  - If the test_scale button plays back the notes in a strange order, you've made a mistake and should reset the scale and start again

- Push the orange "test_scale" button
- Check that the sound from the speakers is at a comfortable level
  - Remember that the equipment has the potential to damage your hearing
  - Adjust the level using the rotary controls on each speaker

The important thing to remember here is that the order in which the notes are clicked is significant. Missing out a note, or swapping two notes can produce different musical results.

This may be important in later exercises, so if you think you make a mistake when inputting a scale – simply use the dark red "reset_scale" button, and start again.

Example Scales
The subject of musical scales is a vast topic, with potentially confusing terminology:

- A note with twice the pitch of another (a ratio of 1:2) is said to be separated by an "octave"
- Western music divides the octave into twelve sub-divisions known as "semitones"
- Scales are constructed by simply selecting a sub-set of these twelve notes
    - In loose terms A scale can have as few as two notes, and at most twelve notes
    - A scale with all twelve notes in it is called a "chromatic scale"

If you've studied music theory, you may have heard scale "degrees referred in terms of "Tonic", "Sub-dominant", etc… We won't use those terms. We simply number the notes in the scale.

One of the things you will hopefully notice during today's session is that the difference between the same sequence played in two different scales can be striking. The table below lists some common scales.

We will come back to this table later – so don't spend too long looking at it now!

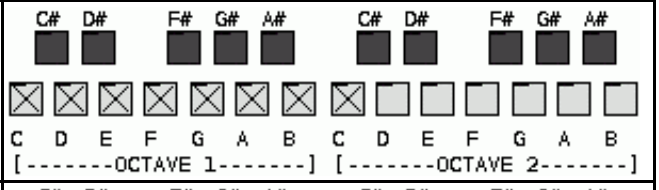
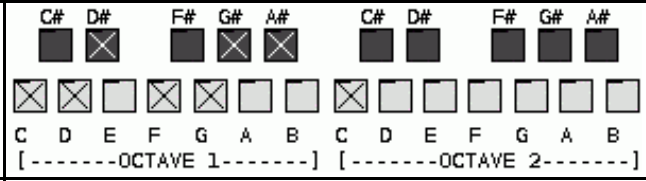
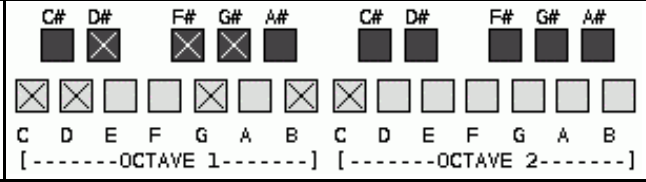
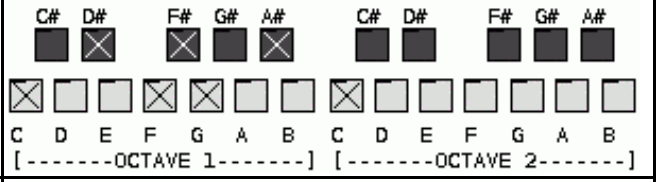
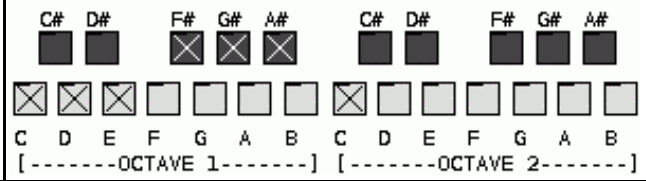
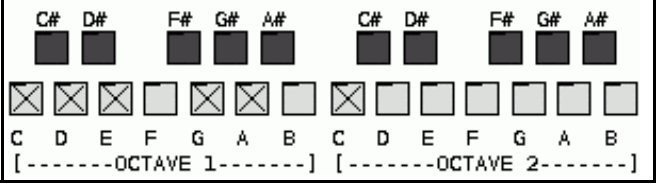
Scale name	No. of notes	Intervals	Scale Image
Major	8	2,2,1,2,2,2,1	
"Natural" Minor	8	2,1,2,2,1,2,2 *Note the same sequence of intervals as in a major scale but shifted.*	
"Hungarian" Minor	8	2,1,3,1,1,3,1,	
Blues Mode	7	3,2,1,1,3,2	
Wholetone	7	2,2,2,2,2,2	
Pentatonic	6	2,2,3,2,3	

Table 1 – Example scales

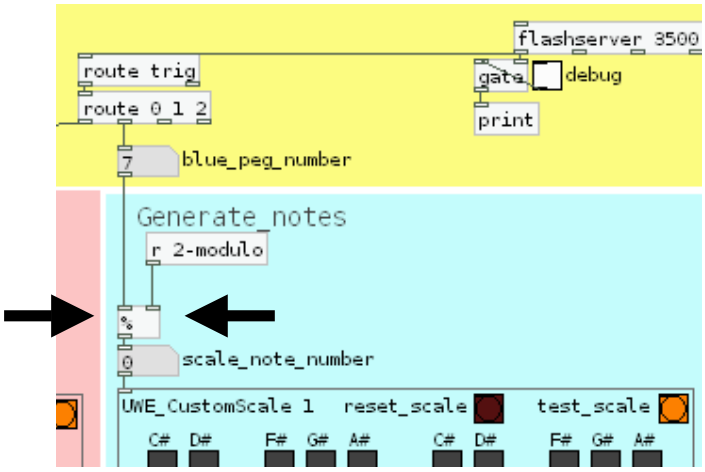
Exercise 2 – The rotating disks

Next, look at the window with the two disks, this is actually a flash animation, but it is not animating at the moment:

In the flash window:

- Change the zero next to "rotate speed blue" to a one
    - Let's leave the red rotation speed alone for now to save confusion!
- When a peg crosses the green line, Flash sends a message to PD
- PD can interpret these messages to generate audio

Notice that each disk has a number of pegs around its circumference. The number labelled "blue_peg_number" feeds into the [%] modulo object (see arrows in figure below). The modulo divisor directly corresponds with the number of notes in the scale and updates dynamically as you click more notes.



The important thing here is that the number produced after the modulo operation used to pull a note out from the scale. This ensures that the scale wraps around if the number of pegs is greater than the number of the notes in the scale.

- Make a mental note of how a C major scale (with 8 notes) sounds when triggered by a disk with 8 pegs.
- Next, increase the number of pegs on the blue disk from 8 to 9.
- **Where does the scale repeat now?**

Exercise 2.1 continued

Now try increasing the number of pegs on the blue disk from 8 to 80:

- Notice that the rate at which notes are generated increases by a factor of 10
- Notice also that the sequence of notes is repeating due to the modulo operation.
- Reduce the rotation speed of the blue disk by a factor of 10 to compensate
    - Position the cursor before the 1 and type "0." (zero point) to change the value to "0.1"

On the next page, we'll change the sound of the instrument, and experiment with cross-rhythms.

Exercise 2.2 - Changing the instrument

Before moving on**, and whilst the 80-peg C major scale is still playing**, try using the number box (with arrows) in the area shown in the figure below.

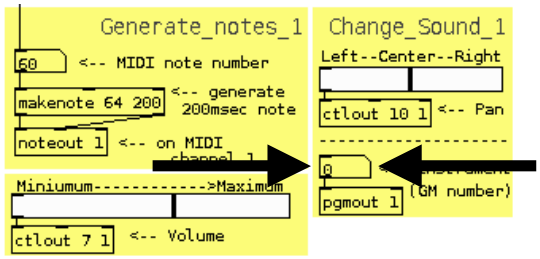
To select different synthesised instruments for the notes to play back on:

- Click then type a number
- Or click, then drag up and down

Table 2 is 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)



Changing sound parameters

It is also possible to change the length of the note produced. The default note length is 200 milliseconds. For some instruments (string instruments and synth-pads for example) this may not be long enough.

Ask one of the supervising lecturers to help you change the note length if you're particularly interested in slowly developing strings/pad sounds.

Exercise 2.3 – Moving things around

The buttons along the left side of the Flash animation allow the size of the blue disk and it's centre point to be changed

- The rate at which pegs move around the disk is constant regardless of disk size

Also, try dragging the end points of the green line:

- The green line causes note triggers regardless of the disk rotation speed.
- Try setting **both** disk rotation speeds to zero, then drag the green line around to produce flurries of notes.

Certain configurations will cause two pegs to simultaneously collide at different points of the line:

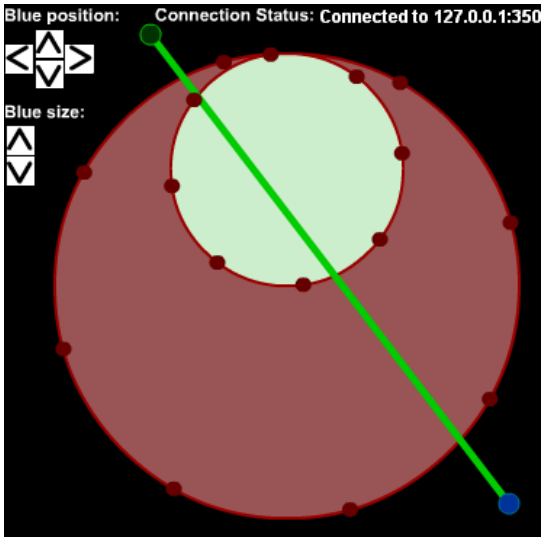
- Try arranging the line so that it bisects the diameter of a disk exactly
- The PD patch should generate a note event when pegs on either side of the disk cross the line resulting in two notes simultaneously (a "dyad" in music theory terms).

Other configurations will cause two pegs to ALMOST collide simultaneously with a slight triplet swing.

Exercise 2.4 – Two disks generating notes
It is possible to generate chords with more notes by involving pegs from the red disk.

To generate 4 note chords:

- Program both [UWE_CustomScale] objects
- Change both the red and blue disk rotation speed to "1"
- Position the green line and disks such that it intersects both sides of both disks (see figure below)



Exercise 2.5 – Interlocking melodies
The modulo operation means we can produce some interesting interlocking melodies when two scales "wrap around" in overlapping periodic ways.

To see this in action:

- Program just 3 notes into both sides of the patch (for example the first 3 notes of C-major: C,D,E)
- Set the peg numbers for the red disk to 8 pegs
- Set the peg numbers for the blue disk to 14 pegs

Notice that for the first revolution of the red disk the notes produced are identical, but after this point, interesting interlocking melodies occur. The notes produced by the red disk will harmonise with the notes generated by the blue disk in ways that repeat in a predictable (but complex) way.

This is due to the modulo operation "reseting" the separate sequences back to scale note zero at different points, displacing the two patterns. It may help to visualise this in a table/diagram below:

Blue Pegs	00	01	02	03	04	05	06	07	08	09	10	11	12	13	00	01	02	03	04
Modulo 3	0	1	2	0	1	2	0	1	2	0	1	2	0	1	0	1	2	0	1
Note name	C	D	E	C	D	E	C	D	E	C	D	E	C	D	C	D	E	C	D
Red Pegs	00	01	02	03	04	05	06	07	00	01	02	03	04	05	06	07	00	01	02
Modulo 3	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0	1	2	0
Note name	C	D	E	C	D	E	C	D	C	D	E	C	D	E	C	D	C	D	E
Notes from Red	C	D	E	C	D	E	C	D	C	D	E	C	D	E	C	D	C	D	E
Notes from Blue	C	D	E	C	D	E	C	D	E	C	D	E	C	D	C	D	E	C	D

Time along this axis →

Displacement of note sequences

This idea of periodic sequences is explored in much more detail in the next section.

3 Pisano sequences - Introduction

This exercise explores the use of Pisano sequences as note generators:

- In this exercise, peg numbers are ignored
- Instead, Pisano sequences are used to pick notes from the scale
- A peg that crosses the green line generates the NEXT Pisano number
- There are separate scales and Pisano generators, one for each disk

To begin, we'll need to close down some windows, and load a new PureData patch:

- Close the "01_MAIN_DiskScales.pd" patch
- **DO NOT CLOSE THE FLASH ANIMATION**

- A PD dialog may appear asking you about discarding changes, click "Yes" to discard.
- Use the menus to select File->Open
- Navigate to the 02_DiskPisano folder
- Open the "02_MAIN_DiskPisano.pd" file
- IMPORTANT: Arrange your windows such that the flash and puredata patch are both clearly visible simultaneously

- Next, in the Flash animation - press the big blue button marked "Initialise all".

If the text at the top of the flash window does **not** say: "Connected to 127.0.0.1:3500 OK!" just ask a supervising lecturer for help

The display should look similar to the figure below:

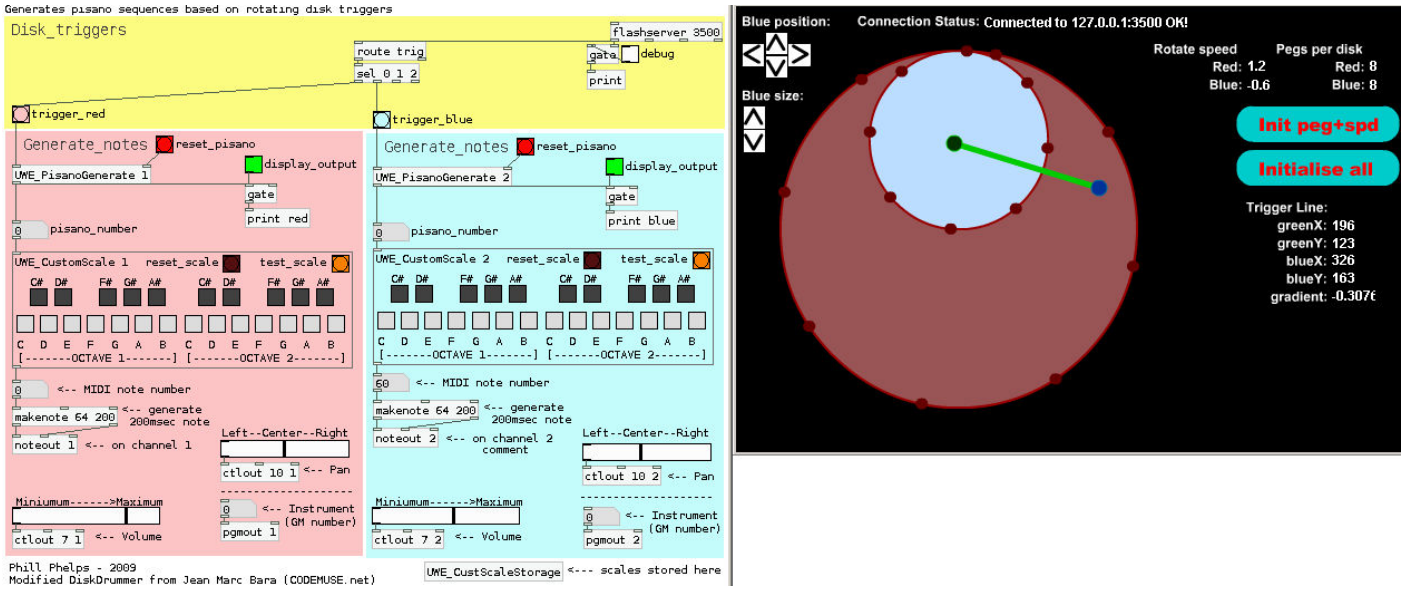


Figure Pisano PD patch for two disks

Exercise 3.1 – Pisano sequences

To make things simpler in the beginning, let's concentrate on just one disk for now.

In the Flash window:

- Position the green line so that it is not intersecting any pegs
  - This is temporary – just whilst we're setting up the PD patch
- Next, set the number of pegs for the blue disk to 8
- Start the blue disk spinning by setting the rotation speed to one, set the red disk rotation to zero

Next, in the PD patch:

- On the side for the blue disk - push the dark red “reset scale” button
- Also on the side for the blue disk - program a C major scale
- When you’ve finished, push the orange [bang] marked “test scale”
    - Remember to program the notes from left-to-right!
    - If the "test_scale" button plays back the notes in a strange order, you’ve made a mistake and should reset the scale and start again

- When the scale is correct, reset the Pisano generator on the side for the blue disk
    - This ensures the Pisano generator will start from the beginning of the sequence
    - Try to get into the habit of resetting the Pisano generator after inputting a new scale

Now let’s start the process going – in the flash window:

- Position the green line so that it intersects just ONE side of the blue disk
    - This is to ensure there are no simultaneous triggers
    - We only want to generate one note at a time at this early stage
- The numbers from the Pisano generator are now being mapped to the C major scale

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

It may help to count the Pisano numbers directly, in the PureData patch click one of the bright green "display output" toggle buttons. The numbers will appear in the PD console.

Exercise 3.2 – The effect of scale lengths on the Pisano period

In the Flash window:

- Temporarily move the green line so that it is not intersecting any pegs.

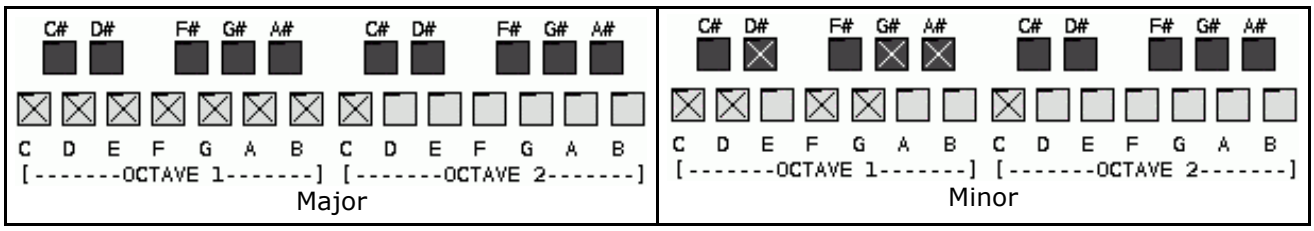
In the PD patch:

- On the side for the blue disk - reset the scale
- Also on the blue side - program in a C minor scale (2nd scale from the top of the table)
    - Remember to program in left-to-right order
- Again on the blue side - reset the Pisano generator
    - Remember to do this after programming each scale

Finally, back in the flash window:

- Position the green line so that it intersects one side of the blue disk

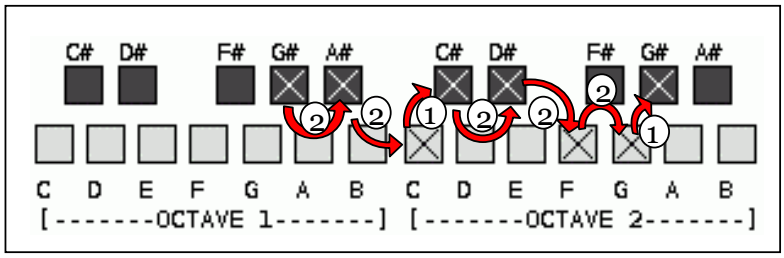
Did you notice that the number of notes in the major and minor scales is identical (8), so the Pisano series produced has the same period? **The number of notes in a scale determines how many notes are in a Pisano-generated phrase, before the musical phrase repeats.**



Major and Minor scale images repeated for convenience

Exercise 3.3 – Transposing scales
The scale table from earlier includes "scale images" to help you program scales in the key of C. The images are all in the "root key" of C, but you can use the "interval" column to transpose this easily as in the example below.

The general approach to transposing a scale using the table is to use the intervals to calculate each note. 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. It may also help to look at the figure below with the intervals for G# major overlaid on the scale image.



G# major (with 2,2,1,2,2,2,1 interval overlay)

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

In the flash animation:
- Temporarily move the green line (or set rotation speeds to zero) so that no pegs trigger

In the PD patch:
- 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)
- Use the "test_scale" button, if the scale plays back in a strange order, reset and start again
- Lastly, reset the Pisano generator to ensure the sequence starts from the correct place

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 6 notes (like a blues scale) produce?**

Test your prediction:
- Set the rotation speed of disks (or move the green line) so that pegs are triggered again
- How many notes are there in this sequence?
- Were you correct about the Pisano period for modulo 6?

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

If you've reached this point, now might be a good opportunity to take a break.

3 **Investigation –** Investigation – pointers, reminders, and hints

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!).

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.

- 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

- Try experimenting with a pair of scales which will produce the same Pisano period, to see which musical scales go nicely together.
    - Major and minor scales have the same number of notes (8)
    - Look carefully at the intervals for both a major and minor scale
    - Notice that the interval sequence of minor and major scales are transposed – this is partly where the terms "relative major" and "relative minor" come from
- Try a major scale on one disk, and it's relative minor on the other
    - A minor is the relative minor of C major, can you work out any others?

- Try experimenting with scales with radically different periods:
    - How about a scale that produces a short Pisano period with a scale that produces a long period

- You can program any scale, they don't have to be listed in the table
    - Try leaving out some notes in a scale, or inserting extra notes
    - When constructing your own scales, it may be useful to remember that interval steps should add up to 12
    - Try extending a scale into the second octave (it's okay to break the "add up to 12" rule!)
    - Remember that the number of notes in a scale directly determines the Pisano period

- Remember you can change the instrument sounds and stereo position to help aurally separate two sequences if you're having trouble hearing the period of each sequence

Exercise 4.1 - Extra stuff
If you're feeling particularly adventurous, you could try loading the "three disk" versions of the Scale and Pisano exercises. The 3 disk version is helpful if you want to scatter a scale across all three generators. Try programming the first 2 notes on the red disk, the next 3 on the blue disk, and the last 3 on the green disk.

You can also re-wire the pure-data patch to automatically change the instrument sound, shift the notes down into lower octaves for a bass line, change the note lengths, and play back drum-kit/percussion sounds. Ask a lecturer for help with this.

5 Whitney music box - Introduction

Earlier in the week, you may have seen the Whitney Musicbox (named after John Whitney, a pioneer in early computer graphics and animation).

The Whitney music box is very similar to terms of the rotating disk triggers we've been using.

It is a system of 30 rotating disks with just one peg per disk, but the radius of each disk is slightly smaller than the previous disk. Each disk is black, pegs are coloured in a rainbow effect, and the rotation speed of each disk is strongly related to adjacent disks.

To begin, we'll need to load a different set of Flash and PD files:

- Close any flash animations and PD patches that may be open
- load the PD patch "03_MAIN_WhitneyPisano.pd"
- Load the flash animation "03_Whitney.swf"
- IMPORTANT: Arrange your windows such that the flash and puredata patch are both clearly visible simultaneously
- Don't forget to check the "Connected to 127.0.0.1:3500 OK!" message

The display should then look similar to the Figure below:

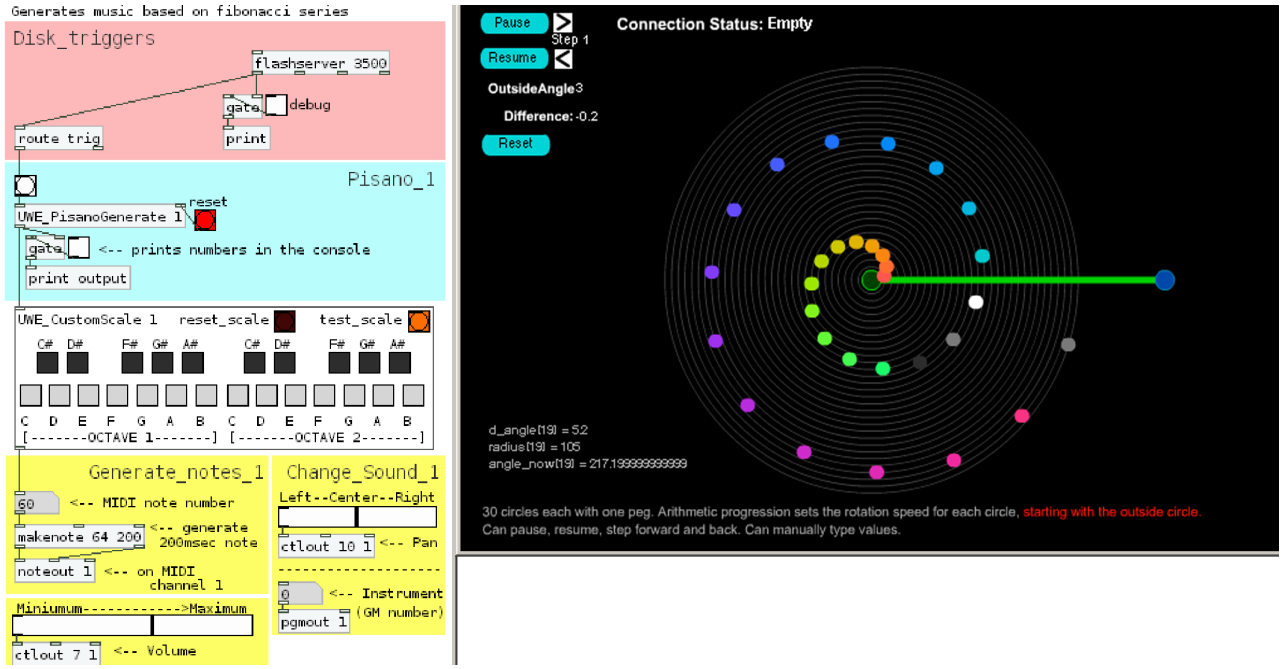


Figure – WhitneyPisano

Exercise 5.1 – Whitney & Pisano

This exercise is also completely free for you to choose the direction you take to explore the possibilities. Here are some suggestions:

- Try setting the GM instrument number to one within the percussion family (You can also change the MIDI channel to output sounds from a drum kit – ask a supervising lecturer for help with this)
- Try resetting the Pisano generator manually at specific points during the sequence (i.e. always reset at points when there is a clear pattern, or always when there is no clear pattern)
- Try moving the green trigger line to intersect only a subset of the full 30 disks
- Try setting unusual values for the outside angle & angle difference that cause the some disks to rotate in reverse whilst the others rotate normally (try "3" for the outside angle and "+0.4" for the difference)

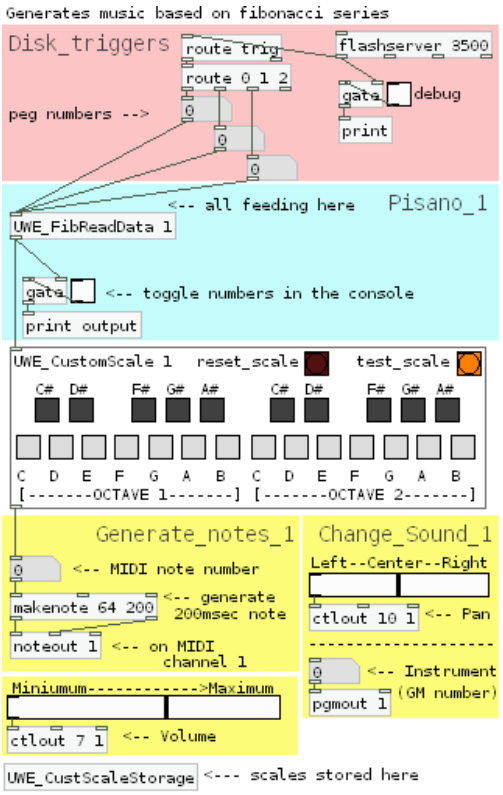
Exercise 5.2 – Alternative versions

There is also a PD patch that uses the Whitney disk number as a direct index to the scale (bypassing the Pisano sequence generator as in the first exercise). It is also possible to run the WhitneyPisano and WhitneyScale versions are side-by-side (allowing for some very intricate crossovers between the two patterns).

In addition to the above Flash and PD files there are "EXTRAS" provided – this table explains a few:

Flash File	Comments
EXTRA_thirtyDiskGeometricWhitney	• A geometric progression sets the rotation speeds for each disk. • Visual patterns and resultant rhythmic triggers that result from this are quite different from the patterns seen with the arithmetic progression.
EXTRA_whitneyGamelan	• Five disks with rotation speed set using a Geometric progression. • Use with "EXTRA_WhitneyGamelan" PD patch for an authentic Indonesian Gamelan ensemble experience!

Exercise 5.3 the Pth' Pisano number



This patch (see Figure to the left) is designed to use the DiskTrigger flash files not the Whitney animation. This is because it uses the peg number to select a specific number from the Pisano sequence, and the Whitney animation only has 1 peg per disk.

Here's how it works:

- You can program a scale up to 24 notes
- The scale size affects the Pisano period
- Each rotating disk has a number of pegs
- The peg number is used to jump to the particular index point in the Pisano sequence for each peg
- This Pisano number is then mapped inside the scale
- Simultaneous peg events result in chords as before

There are two levels of abstraction at work here, which can result in some really unexpected rhythms and melodies.

Up to three disks are recognised by default, but this could also be used with the Whitney music box with 30 disks if the PD patch were tweaked to register the disk number instead of the peg numbers.

6 Taking it further

The exercises today have been set up in a fixed way (with various parameters left open for your investigation) but the editable nature of PureData means that you can literally rewire the behaviour if you wish. A CD-ROM is provided so you can take the PD patches and Flash animations home to play with on your own time if you would like to.

If you have any questions or comments (positive or negative!) about today's session or about setting up your CD-ROM at home - please email philip3.phelps@uwe.ac.uk