Composing, Performing, and Listening to Experimental Music

**Lab sections** There will be weekly lab sections. These sections will meet in small groups of four or five. Scheduling will be completed at the second class.

**Attendance:** Attendance at both classes and lab sections is a requirement of the course. If you are unable to attend a class, you must email me an explanation for your absence prior to the class you will miss. Failure to do so will be deemed an unexcused absence. Unexcused absences will result in a 5 point reduction in your final grade.

**Assignments:**
The first half of this course has daily homework assignments. They must be completed by the following class session. I will end each class with a brief look at the assignment. It is strongly recommended that you actually undertake the assignment immediately after class while the material is fresh in your mind.

**Tests**
There is a quiz and a take-home midterm. Each test will be taken using the SuperCollider application, so you will have full access to the help facilities of the language.

**Grading**
assignments: 50%, tests: 25%, final project: 25% of final grade.

### Part I: SuperCollider basic concepts

**Day 1** 1/23  **Introduction to the SuperCollider 3 environment**
**Server and Language**
Reading:
SC tutorial: Getting Started With SC: sections 1 – 5
HW 0 (due 1/28): Go shopping!
Go to sccode.org. Try out at least 10 sound examples you find there. Select one that you would like to understand better and submit it on moodle

**Day 2** 1/28  **Objects and Classes, Methods and Messages**
Reading:
SC tutorial: Getting Started With SC: sections 6 - 8
Ron Kuivila, 1. Notes on Message Passing
HW 1 (due 1/30): identify classes, selectors, functions, arrays

**Day 3** 1/30  **Numbers and Operators**
**UGens and the numbers they produce**
Reading:
SC tutorial: Getting Started With SC: sections 6 - 9
SC3 help: Operators
notes 2. A Note on Numerical Ranges in SuperCollider
notes 2a. A note on the overall range of hearing
HW 2 (due 2/4) Using the range message, writing synthesis functions from verbal instructions

**Day 4** 2/4  **Overview of the Server, Arrays and how they represent multi-channel audio**
Reading:
SC tutorial: Getting Started With SC: sections 10 – 13
Notes 3: Server Architecture
HW 3 (due 2/6): Multi-channel expansion and declaring controls for synthesis functions
Quiz 1
Day 5  2/6  Using Ndef to investigate SynthDef parameters
How Ndef “acts like an array”
Using ControlSpecs to create GUI controls for an Ndef
Reading:
Help files on NodePoxy, Ndef, and ControlSpec
HW 4: (due 2/11): using Ndefs to create GUIs for synthesis functions

Day 6  2/11  Using Clocks, Routines, Arrays and iteration to play an Ndef
Reading:
SC tutorial: Getting Started With SC: sections 14 - 16
HW5: make a routine that “Shuffle plays” a set of sounds through an Ndef. Make a routine
that plays a single sound in an Ndef by changing its controls.

Part 2: The UGen Library

Day 7  2/13  Some UGens through SC Tweets
Reading:
Notes 7: SC tweets background tips
HW 6: SC Tweets

Day 8  2/18  SoundFile playback with Buffer and Info UGens
Reading:
Notes 8. Buffers
Notes 8 Sample Rate and Bit Depth
Buffer, PlayBuf and BufRd Help files
HW7: variations of an SC tweet recording using sample playback

Day 9  2/20  Generators and Envelopes
Reading:
Help files for:
SinOsc, Pulse, Saw, LFPulse, LFSaw, LFTri Blip, Klang, Klank. Impulse, PMOsc, SinOscFB,
SyncSaw, VarSaw
Linen, Env, EnvGen, IEnvGen
HW8: make a one minute long piece that combines as many different generators as possible
while maintaining clarity. The piece has succeeded if members of the class can correctly count all the
different generators used.

Day 10  2/25  Filters, Delays, and Dynamics
Reading:
Help files for:
Combander, Limiter
CombC, AllpassC, Pluck
RLPF, RHPF, BPF, BRF, BMoog, BPeakEQ
FreqShift, PitchShift
HW9: use filtering, delay and dynamics processing to make a collection of completely different
variations of the same original. The goal is for all of the variations to have no obvious relation even
though they are all derived from the same sound. Produce a set of six different sounds, each rendered as
a soundfile.

Day 11  2/27  Analysis UGens and Review
Reading
Help files for:
Amplitude, PeakFollower,
Qitch, ZeroCrossing, Onsets
Day 12 3/4  Triggers and OSCdef
Reading
Helpfiles for:
SendTrig, SendReply, SendPeakRMS
OSCdef, OSCfunc
HW10: Make a "listener" that sends data from the server to the language using Amplitude, Qiitch, and Onsets, to an OSCdef. The OSCdef displays those values through an Ndef GUI

Day 13 3/6  FFT based UGens
Helpfiles for:
FFTOverview

Midsemester break!
Day 14 3/25  Highlights of the forthcoming SEAMUS conference
HW11: A review of three events in SEAMUS
a. One workshop or paper
b. One specific concert piece
c. One specific installation

Day 15, 3/27  No class, attend the conference
HW12: Make a “final project fantasy”, a proposal for your final project unconcerned with practicality.

Day 16 4/1  Synthdefs and Events
HW 13:

1. Write an Ndef that plays three SinOsc UGens with independent frequencies controlled as MIDI key values. Use a Lag UGen to allow them to make glissandi between different frequencies. The oscillators will look something like: "SinOsc.ar(Lag.kr(pitch1, lag1).midicps)"

2. 2. Write a set of 5 events that change the tunings of your Ndef. An Event will look something like: " ~e = { play: { ~ndef.set(pitch1, ~pitch1, pitch2, ~pitch2) } pitch1: 50, pitch2: 60, ndef: myNdef };"

3. 3. Write a routine that plays these events. It should first play each event in a fixed order and then randomly shuffle the collection of events and replay them. (In this way each event is heard before any other event is repeated.)

Day 17 4/3  Patterns and Events
HW 14: Make a piece that is implemented completely with Patterns and that involves at least three SynthDefs producing:

a. steady tones

b. impulses

c. noise

Make a variant by chaining the pattern to another that changes some critical aspect of the piece.

Day 18 4/8  Translating musical notation into patterns
HW 15: Write a pattern that presents the theme music to "the Simpsons". See the associated homework files.
Alternate instrument topologies and controls

HW 16: Attached is an example of a pattern producing a drum beat using a four voice synth to simulate different drums. Notice how this example uses Pseg to gradually alter the envelope characteristics of some sounds.

In the second example the four Pseqs that specify the rhythmic patterns are replaced with Prouts, some of which create patterns dynamically.

1. Copy the second pattern into a second file and write comments that explain what the four Prouts are doing.

2. Create your own patterns that create the rhythmic patterns played dynamically. Your pattern should last for precisely 16 16 beat repetitions.

Day 20 4/15  MIDI and Open Sound Control

Day 21 4/17 Using Responders to process External Control input

HW 17: write a Pattern that generates sound based on the spectrum of the incoming sound using the program provided for the sound analysis.

Day 22 4/22 Machine Listening

HW 18: Final project proposal. The proposal should include a prose description and sound examples written in SuperCollider.

Day 23 4/24 Final Project starting points

In class discussion and review of the final project proposals. No more homework! All work should focus on the final project

Day 24 4/29 Special topic: Conlon Nancarrow and James Tenney

Day 25 5/1 Special topic: “Systematic” composition and Tom Johnson

Day 26 5/6 Review of the semester

5/13: FINAL PROJECTS PRESENTED