

The New StochGran: Expanded Stochastic Granular Synthesis Tools

Kieran McAuliffe

College-Conservatory of Music
University of Cincinnati
bkieranmkcauliffe@gmail.com

Mara Helmuth

College-Conservatory of Music
University of Cincinnati
mara.helmuth@uc.edu

ABSTRACT

*Kieran McAuliffe and Mara Helmuth have updated and expanded previous granular synthesis software instruments in the RTcmix music programming language and created MaxMSP externals for a new interface. The original non-real-time Cmix instruments by Helmuth allowed composers to create complex, changing timbres by specifying moving stochastic distributions of grains. The new instruments allow for much finer and real-time control of these distributions. This paper also introduces the multi-channel RTcmix instrument which generates stochastic grain clouds in up to 16 channels. In addition, two MaxMSP externals, based on the Cmix instruments, have been programmed for synthesis and sampling, with an expanded graphical user interface replacing the previous StochGran application. These tools provide expanded control options and facilitate more flexible and powerful performances and improvisations. Finally, compositions showcasing the new possibilities of the software are discussed, including McAuliffe's *Jealousy...guilt*, in eight channels, and Helmuth's *Slither*, *Bursty Splat* for laptop ensemble.*

1. INTRODUCTION

Stochastic granular synthesis is a powerful compositional and sound-generating technique for creating unique sounds and families of related sounds. Stochastic generation of grain parameters allows composers to create complex granular sounds using probabilities, without having to go through the laborious process of specifying the details for each individual grain. For dense granular clouds there may be hundreds of thousands of parameters for a sound that lasts a matter of seconds. Mara Helmuth originally developed the stochastic granular instruments `sgran()` and `stgran()` as part of the non-real time CMIX music programming language [1], and has used these programs in many of her compositions. To allow for graphical parameter control, and algorithmic sound generation, she created the StochGran application [2, 3, 4] to run those

instruments. Later, she programmed `SGRANR()` and `STGRANR()` in RTcmix and used them with live control in MaxMSP in two laptop ensemble works.

Our goal was to update the instruments to offer the control over grain parameters and their changes over time available in the non-real-time Cmix instruments, but in a real-time context. Other enhancements were added for multichannel grain panning, an adjustable limit on concurrent grains, and a buffer size control for `STGRANR2()` in RTcmix. Additionally, McAuliffe rewrote the instruments as MaxMSP externals and created useful interface tools for this environment.

2. BACKGROUND

2.1 Granular Synthesis

Iannis Xenakis envisioned granular synthesis [5] and the inspiration for this has been variously attributed to the theories of Einstein or work of mathematician Denis Gabor. In computer music, Barry Truax and Curtis Roads became pioneers in the technique with Truax developing the first real-time software for his piece *Riverrun* [6, 7], written in 1986. This piece and Xenakis's *Formalized Music* [8] inspired Helmuth to write her early stochastic instruments. Granular synthesis [9] is widely used today, and found in many plugins and applications.

2.2 RTcmix

RTcmix [10, 11] is a music programming language developed by Brad Garton, John Gibson, Dave Topper, Doug Scott and others as a real-time expansion of Paul Lansky's CMIX, which was written primarily in C. Instruments in RTcmix are inherited C++ classes, and scorefiles can be programmed in Minc, Perl or Python interfaces. Development continues to this day, including Brad Garton's embedded RTcmix plugins within MaxMSP (`rtcmix~`) and within Unity (`urtcmix`). The command line (standalone) RTcmix software currently only works on Linux and OSX, but the embedded versions work on any operating system the host software runs on.

2.3 Other Stochastic Granular Synthesizers

Dan Trueman and Luke Dubois included the granular external `munger~` as part of the Percolate MaxMSP package. Ico Bukvic later upgraded this to `munger1~` [12],

improving performance and adding additional features such as cross platform usage with Pure Data. Unlike our instruments, this is a sort of “swiss army knife” with a wide array of granular capabilities. It can work with fixed buffers or live input. For stochastic control of pitch `munger1`~ requires discrete bins, though a jitter control allows variability around these pitch centers.

Similarly, the CREATE team’s `EmissionControl2` [13] offers a wide array of granular controls, but here in the form of a standalone program, where all granular parameters can be controlled by low frequency oscillators. It works with a fixed soundfile buffer and cannot operate on live signals. Though extremely powerful and controllable, its use case will be different as a standalone software.

Inspiration for multichannel granular spatialization comes from in part from Michael Norris’s `spindrift`~ [14], which provides a variety of interesting capabilities including grains which move through multichannel space. It is also designed to work with a fixed buffer, and not for extremely dense grain clouds.

3. THE INSTRUMENTS

The instruments are two subclasses of the `RTcmix` Instrument class, and their shared libraries are loaded and run on the command-line or Terminal window with scripts or scorefiles containing parameter lists. We have also used them in both the Max and pd plugins, `rtcmix~`, which allows for `RTcmix` instruments to be controlled within those environments. `SGRAN2()` uses a wavetable for its source signal, as shown in Figure 1. `STGRAN2()` uses either a live input signal or sound file, shown in Figure 2.

3.1 Dense Stochastic Distributions

`SGRAN2()` and `STGRAN2()` stand out for allowing the user to precisely specify the stochastic distribution for each of their parameters, while other granular synthesizers with stochastic control generally rely on uniform distributions with equal probability of a parameter being chosen within the user provided bounds, shown in the middle example of Figure 3. This precision results in a variety of novel timbres especially with its dense polyphonic capabilities allowing tens of thousands of overlapping grains per second.

All iterations of the instrument have used the ‘`prob`’ function for stochastic control. This function can sample a variety of distribution shapes with four intuitive parameters. The user provides low and high values which are hard limits of the range, along with a middle point, which values will cluster around or avoid according to the tightness control being greater or less than one, as shown in Figure 4. `SGRAN2()` uses four different prob distributions for determining grain duration, grain rate (period between successive grain attack points), pitch and panning as shown in Figure 1. As it works with sampled sound, `STGRAN2()` replaces the pitch control with a transposition control. `STGRAN2()` also adds a parameter controlling the size of the input circular buffer, shown in Figure 2. A short buffer allow for granulations which preserve the transients of the original material, while a long buffer blur them.

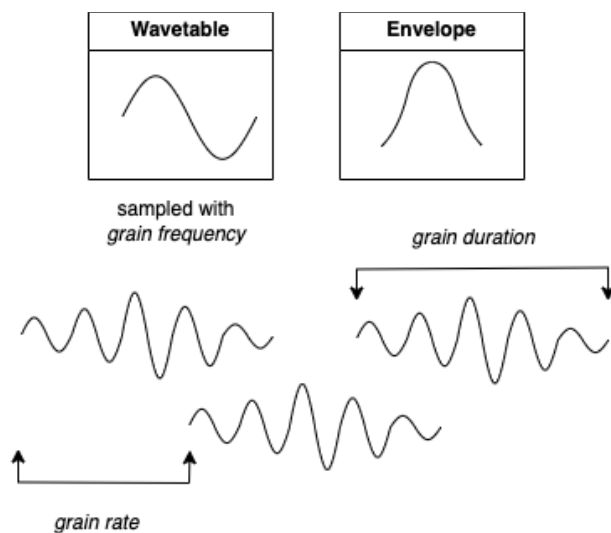


Figure 1. `SGRAN2()` algorithm: An envelope is applied to a waveform for each grain. Grain rate is time in seconds between grain starting points. (All graphs show time vs. amplitude.)

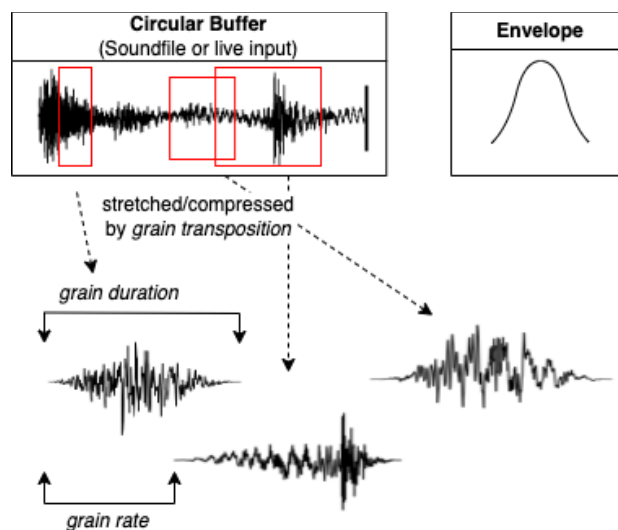


Figure 2. `STGRAN2()` algorithm: An envelope is applied to sound from the buffer for each grain. (All graphs show time vs. amplitude.)

3.2 Real-Time Control

Inn `RTcmix`, all parameters previously controlled by functions which could not be changed after initiating the scorefile, can now be programmed to accept input values, allowing them to be modified in real time by envelopes, LFOs or user control. This allows for a variety of complex timbral gestures from changing distribution shapes in real time, such as the sound produced by the script below in Figure 3.

```

rtsetparams(44100, 2)
load("../libSGRAN2.so")

outskip = 0
dur = 25

amp = maketable("line", 1000, 0, 0, 8, 0.8, 16, 1, 17, 0)

ratelo = 0.00004
ratemid = 0.00005
ratehi = 0.00007
rateti = 3

durlo = 0.000001
durmid = 0.000005
durhi = 0.0008
durti = 0.4

freqlo = 220
freqmid = 440
freqhi = 480
freqti = maketable("line", "nonorm", 200, 0, 6, 1, 1)

panlo = 0
panmid = 0.5
panhi = 1
panti = 0.4

wave = maketable("wave", 1000, "square")
env = maketable("window", 1000, "hanning")

SGRAN2(outskip, dur, 800 * amp, ratelo, ratemid, ratehi, rateti,
durlo, durmid, durhi, durti, freqlo, freqmid, freqhi, freqti,
panlo, panmid, panhi, panti, wave, env)

```

Figure 3. RTcmix script showing an SGRAN2() instance which starts as a narrow frequency band around 440 Hz. and spreads out into a wide band between 220 and 880 Hz. by use of the tightness control (freqti).

3.3 Max Externals

Although these instruments are accessible in the MaxMSP programming environment via the `rtcmix~` external, McAuliffe also programmed MaxMSP externals, `sgran~` and `stgran~`. The `maketable()` functions required for choosing waveforms and envelopes for `sgran~` are replaced by user-specified `buffer~` tables. Similarly, instead of using an internal buffer for storing granular source material, `stgran~` relies on an external MaxMSP buffer for the source signal. `stgran~` involves a fifth prob function controlled set of parameters for GrainHead, which gives more detailed stochastic control of the start point for grains in the buffer. In the absence of RTcmix pfield controls, the externals require the user to send new messages to live update the distributions, which is shown in the help patches. This may seem to go against MaxMSP design principles, and we considered the use of audio rate inlets for every stochastic parameter, but this would result in unwieldy objects with 12+ inlets. Both externals can be compiled for OSX or Windows.

3.4 Max Display Scripts

Additionally in MaxMSP, a Javascript script `prob-visual.js` can allow users to visualize the probability density of a set of parameters by attaching the script to a `plot~` object to help understand the intuition behind the prob function, shown in Figure 4. Another script, `prob-transition.js`, allows users to linearly interpolate between the parameters of two prob distributions allowing for the easy creation of a moving distribution or sonically exploring timbral possibilities. Both of these are demonstrated in the help patches for the externals.

4. STOCHGRAN DISPLAY

Mara Helmuth previously developed a standalone soft-

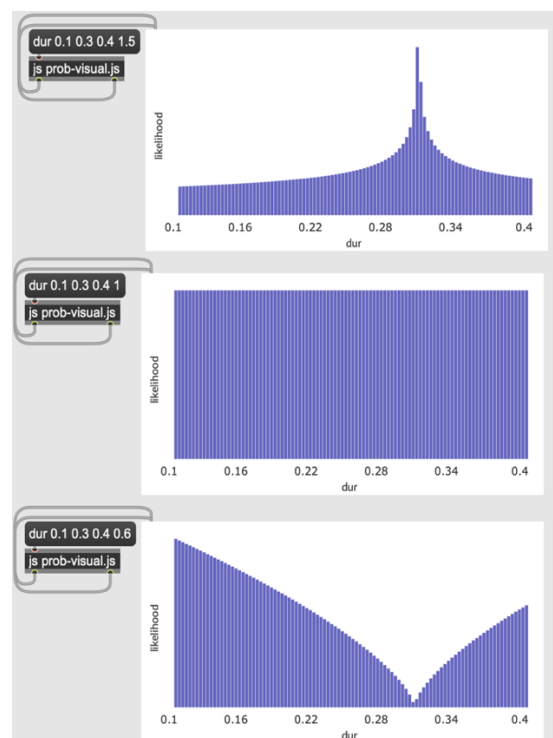


Figure 4. StochGran Panel. Various probability densities functions for grain duration created with the prob function with 0.1 low, 0.3 mid, 0.4 high and varying tightness. Higher vertical bars show a greater probability of choosing that duration value. Visualized using the `prob-visual.js` script and `plot~` in MaxMSP.

ware, StochGran to run the original pair of instruments. This graphical interface for creating moving granular textures with the original RTcmix instruments [5]. We have created a reimaging of this software in the form of a Max patch using the new Max externals which we call Stochgran Display, partially shown in Figure 4. Users can either craft or randomly generate two probability distributions for each parameter and edit a breakpoint function controlling interpolation between the two. Now with real time operation, StochGran can be used as either a tool for crafting fixed pieces, or an interactive patch for live improvisation.

4.1 Integration with NPAN()

The new SGRAN2MULTI and STGRAN2MULTI RTcmix instruments borrow John Gibson's code developed for the multichannel panning instrument NPAN [15] and apply it to each individual grain to create circular clouds or swarms of grains. Gibson based NPAN itself off a scheme described in F. R. Moore's *Elements of Computer Music* [16]. Instead of using `prob()` values to stereo pan each grain, with these instruments the user provides a location for a cloud center via polar or cartesian coordinates, and a radius, with which the instrument will use to randomly place grains as shown in Figure 5. These parameters can also be controlled in real time, allowing moving clouds which may change in shape.

4.2 Performance

The RTcmix SGRAN2() and STGRAN2() instruments

Figure 3. RTcmix script showing an SGRAN2() instance which starts as a narrow frequency band around 440 Hz. and spreads out into a wide band between 220 and 880 Hz. by use of the tightness control (freqti).

both performed well up to 1500 concurrent grains on a single core of a 2019 iMac with an Intel Core i9 processor, while with sgran~ and stgran~ MaxMSP externals

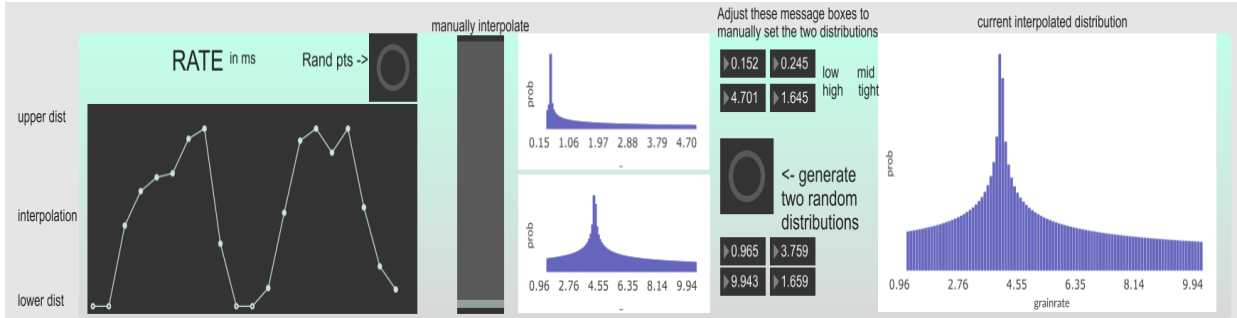


Figure 5. RTcmix script showing an SGRAN2() instance which starts as a narrow frequency band around 440 Hz. and spreads out into a wide band between 220 and 880 Hz. by use of the tightness control (freqti).

up to 1000 concurrent grains. In both software environments the user can specify a parameter for the maximum allowed overlapping grains, so they can freely experiment with parameters and distributions without having to worry about gapping artifacts from overloading CPU usage. Due to the instrument not allowing multiple grains to be started on the same sample frame, the sample rate is the current upper bound for the maximum grains per second. Using multiple copies of the instrument and a multithread enabled RTcmix build, or using the poly~ object with multithreading enabled in MaxMSP, the user can obtain densities and rates far higher than these.

4.3 Availability

All of the new instruments are open source. The RTcmix commandline versions and library for use with MaxMSP can be found at:

<https://github.com/trian-gles/SGRAN>

The multichannel instruments can be found at:

<https://github.com/trian-gles/SGRANMULTI>

The MaxMSP externals, prob-visual.js script, and StochGran remake can be found at:

<https://github.com/trian-gles/stochgran-tilde>

5. MUSIC

Kieran McAuliffe and Mara Helmuth have created several pieces using these newly-developed instruments.

5.1 Jealousy...guilt

Kieran McAuliffe's piece *Jealousy...guilt* uses the new SGRAN2MULTI() sent to an 8 channel speaker array. It uses 16 copies of the instrument within rtmix~, spread out

across cores with a multithreaded poly~. Each voice generates a granular cloud which fades in and out while orbiting the listener. The center of the frequency distribution for each cloud corresponds to a partial of the harmonic series, however for much of the piece this series is not heard. The piece heavily relies on various distortions of the harmonic spectrum, as shown in equation (1).

$$Partial_i = Fundamental \times i + shift \quad (1)$$

The extreme uses of granular parameters further disguises

the spectrum. For example, the effect of raising the frequency tightness for each harmonic cloud is a gradual transition from pure harmonicity to scattered granular chaos. Additionally, a mean duration control allows individual grains to range from coherent pitches to noisy clicks. In the space between these textures lies a pitched yet transparent wind-like timbre. The compositional and recording process involved mapping all of these controls to physical MIDI controller sliders, and improvising with the sounds.

Combinations of parameters all yield different sound worlds which constitute the different sections of the piece. It opens with minimum grain duration combined with a super low fundamental, resulting in nothing more than a faint crackle. As the duration rises, a bubbling timbre emerges which rises in pitch and becomes the audible harmonic series. Halfway through the piece, the LFOs controlling the fading in and out of the various clouds jump to their fastest rates, turning into chaotic rhythmic chatter, heightened in intensity by rapidly changing granular parameters.

The piece can be either presented as a fixed piece or performed live with the MIDI controller operating the MaxMSP patch. With the intuitive nature of the controls, the patch can also function well in improvisatory settings.

5.2 Burren Wind

Mara Helmuth collaborated with Esther Lamneck to create together a piece for tarogato and fixed stereo audio in 2022-23, based largely on granular sampling of Lamneck's multiphonics played on the instrument. The piece was inspired by their experience of the rocky, sparse Burren landscapes on the Eastern coast of Ireland. Most of the processing was done with the new STGRAN2() instrument either in standalone RTcmix or in MaxMSP rtmix~. A new version incorporates interactive video by composer Alfonso Belfiore.

5.3 Slither, Bursty Splat

Slither, Bursty Splat was created for the Cincinnati Composers Laptop Orchestra Project (CiCLOP) in a Live Electronic Music class at CCM by Mara Helmuth. Sonic gestures with identifying names such as slither, bursty, gloom, needle and splat and their combinations and fragments interact in various ways throughout the piece.

The compositional process for Slither, Bursty Splat was to choose three collections of descriptive words. The group of words describes the basic, unique, and variable sound gestures of the piece. Slither, bursty, splat, needle and gloom are examples of these sound gesture names. The second collection of words describes the extending processes which were applied to basic gestures, and includes corrugate, splinter, undulate, shiver, sway, twitch, and convulse. A final collection of one-word descriptors applies to the combination or breaking apart of multiple sound gesture types, including aggregate, amalgamate, squeeze, conspire, sunder and rupture. The various named sound gestures and their extensions were synthesized primarily with the new RTcmix granular synthesis instrument SGRAN2().

The piece can be improvised by laptop performers with MaxMSP patches, or played in an 8-channel fixed media format.

6. FUTURE DEVELOPMENT

6.1 LUAGRAN

Through the programming of these instruments, we realized that the most computationally expensive aspect was the polyphonic handling of many concurrent grains, and not the stochastic selection of parameters for new grains. We are currently developing a new instrument/external LUAGRAN()/luagran~ which takes advantage of this realization by delegating grain parameter generation to user provided Lua scripts, while leaving the CPU intensive polyphonic synthesis to C/C++ code. Code savvy users will get even finer granular control, and those not interested in learning Lua will still be able to use the instrument by modifying the many provided example scripts.

6.2 New Distributions

We plan on implementing additional distributions. The normal/Gaussian distribution has audibly different kurtosis, with higher density around the center, and lower density around the wings.

We are also exploring correlation between parameter distributions. For example, higher pitched grains tending to have shorter duration creates a different timbre than the inverse. A major challenge is making the required covariance matrix intuitive and controllable in real time.

Finally, we are researching options for non-independent calls for a particular distribution, such as random walks.

6.3 More Parameters

Though we do not intend to expand the capabilities of the instrument to the “swiss army knife” munger~ or

EmissionControl2, we would like to try adding more parameters for user control. None of the updated instruments allow for stochastic control of grain amplitude, a feature of one of the original Cmix instruments, which should be included. STGRAN2() currently only produces grains that play forward, and the option to mix between forward and backward grains, such as in munger~, could create interesting textures. Finally, we would like to make a version of SGRAN2() which works with frequency modulated grains, offering stochastic control for carrier frequency, modulator frequency and modulator depth.

7. CONCLUSIONS

The new stochastic granular synthesis instruments SGRAN2() and STGRAN2() bring the parameter control and sound quality of their original Cmix versions into real time, encouraging new gestural exploration. The multi-channel instruments and expanded interface implemented in MaxMSP also provide a powerful StochGran environment for original composition and improvisation.

Acknowledgments

We would like to acknowledge the RTcmix developer community for inspiration and responsiveness to questions.

8. REFERENCES

- [1] P. Lansky, “The Architectural and Musical Logic of Cmix,” *Proceedings of the 1990 International Computer Music Conference*, ICMA, 1990.
- [2] M. Helmuth, “Patchmix and StochGran: Two Graphical Interfaces,” *Proceedings of the 1991 International Computer Music Conference*, pp. 563-566, 1991.
- [3] M. Helmuth, “Granular Synthesis with Cmix and Max” *Proceedings of the 1993 International Computer Music Conf. ICMA, Tokyo*, 1993, pp. 449-452.
- [4] M. Helmuth, “StochGran on OSX” *Proceedings of the 2002 International Computer Music Conference*. ICMA, Gothenburg, 2002, pp. 77-78.
- [5] M. Solomos, The Granular Connection (Xenakis, Vaggione, di Scipio...), Symposium: The Creative and Scientific Legacies of Iannis Xenakis International Symposium, 2006, Canada. fhal-00770088.
- [6] B. Truax, “Riverrun,” *Digital Soundscapes*. Cambridge Street Records, 1987. Compact disk.
- [7] M. Helmuth, “Barry Truax’s Riverrun,” *Analytical Methods of Electroacoustic Music*. Routledge: Taylor and Francis, 2006, pp. 186-238.
- [8] I. Xenakis, *Formalized Music: Thought and Mathematics in Composition*, Bloomington and London, Indiana University Press, 1971.
- [9] J. Haas, “Granular Synthesis and Granulation of Sampled Sound” in *Introduction to Computer Music*. cited 2023, Available from <https://cmtext.indiana.edu/>

- [10] B. Garton, D. Topper, “RTcmix – Using CMIX in Real Time” *Proceedings of the 1997 International Computer Music Conference*, ICMA, 1997.
- [11] RTcmix, “an open-source digital signal processing and sound synthesis language,” cited 2023, Available from <http://rtcmix.org/>
- [12] I. Bukvic, “munger1~: TOWARDS A CROSS-PLATFORM SWISSARMY KNIFE OF REAL-TIME GRANULAR SYNTHESIS,” in *Proc. Int. Computer Music Conf. (ICMC 2007)*, Copenhagen, 2007.
- [13] EmissionControl2, cited 2023, Available from <https://github.com/EmissionControl2/EmissionControl2>
- [14] M. Norris, “spindrift~ granular synthesis Max external,” cited 2023, Available from <https://www.michaelnorris.info/software/spindrift>
- [15] J. Gibson, “NPAN”, cited 2023, Available from <https://github.com/RTcmix/RTcmix/blob/1b04fd3f121a1c65743fde8ea37eb5d65f2cf35c/in-sts/jg/NPAN/NPAN.cpp>
- [16] F. R. Moore, *Elements of Computer Music*. Pearsons, 1990, pp. 353-359.