AUDIO PROCESSING CHAIN RECOMMENDATION

Spyridon Stasis

Digital Media Technology Lab,
Birmingham City University
Birmingham, UK
spyridon.stasis@bcu.ac.uk

Nicholas Jillings

Digital Media Technology Lab,
Birmingham City University
Birmingham, UK
nicholas.jillings@bcu.ac.uk

Sean Enderby

Digital Media Technology Lab,
Birmingham City University
Birmingham, UK
sean.enderby@mail.bcu.ac.uk

Ryan Stables

Digital Media Technology Lab,
Birmingham City University
Birmingham, UK
ryan.stables@bcu.ac.uk

ABSTRACT

In sound production, engineers cascade processing modules at various points in a mix to apply audio effects to channels and busses. Previous studies have investigated the automation of parameter settings based on external semantic cues. In this study, we provide an analysis of the ways in which participants apply full processing chains to musical audio. We identify trends in audio effect usage as a function of instrument type and descriptive terms, and show that processing chain usage acts as an effective way of organising timbral adjectives in low-dimensional space. Finally, we present a model for full processing chain recommendation using a Markov Chain and show that the system's outputs are highly correlated with a dataset of user-generated processing chains.

1. INTRODUCTION

Mixing audio involves a range of complex processes that include balancing source amplitudes, applying audio effects, and positioning a sound source in a perceived space. These tasks can be time consuming and are often carried out for either corrective or creative reasons. Corrective tasks are straightforward but time consuming operations, including noise removal, temporal alignment, and level correction using equalisation and dynamics processing. Creative tasks on the other hand require artistic interpretation, and can involve perceptually mapping ideas and descriptions of audio to processing parameters.

During production, audio effect modules are often cascaded to create processing chains for each channel and bus in the mix, including the master. This allows combinations of linear and nonlinear systems to be able to apply processing to the audio signal at various points in the workflow, typically using a Digital Audio workstation (DAW). For corrective purposes, the effects included in each of the processing chains are selected by the engineer as a reaction to audible cues, such as artefacts in the mix. For creative purposes, they can be selected with a view to make a source more or less prominent in the mix, or to achieve a given aesthetic.

1.1. Intelligent Music Production

Intelligent Music Production aims to provide interfaces and algorithms to automate and facilitate the music production process [1]. This should effectively reduce the time spent by producers and engineers on time consuming, menial tasks, and allow them to focus on the creative aspects of music production. Previously, these systems have been developed for automatic mixing, whereby aspects of the production workflow such as balance [2, 3] and panning [4], can be optimised according to psychoacoustic principles.

Studies in the area of Intelligent Music Production have explored methods for the automation of parameter settings based on external semantic cues, such as descriptive language [5]. This has been applied to a range of audio effects such as equalisation [6, 7, 8], distortion [9], compression [10] and reverberation [11]. In each of these cases, parameter automation is applied within the context of a single effect, meaning the user of the system needs to be aware of the type of processing required to achieve the desired aesthetic.

The use of descriptive terms in music production can be shown to represent changes in musical timbre, which often requires the application of multiple audio effects [12, 13]. This suggests that in order to provide users with a flexible interface, combinations of effects need to be explored. This process is nontrivial, as individual audio effects are complex, multidimensional processing units [6], and the combination of linear and nonlinear systems are noncommutable, resulting in a large number of possible combinations. For example, Dynamic range compression placed before an equaliser (EQ) will provide a potentially different outcome than an EQ placed after the compressor [14], even when the settings of the two audio effects are retained and only the order of them is altered. This is confounded by additional contextual conventions, such as effects being used for specific instrument classes (e.g. drum compression or vocal equalisation).

1.2. Objectives

In this paper, we present a method for full processing chain recommendation, based on a dataset of empirically captured user data. We provide a comparative analysis of audio processing tools, based firstly on the frequency of individual effect usage within a chain, then on combinations thereof. Using this information, we can then identify commonalities and patterns in audio effects usage, with respect to contextual attributes such as timbral descriptions, audio effects and genre. We conclude by presenting a system which is able to recommend a number of processing modules based on the likelihood of an effect's position in a processing chain, weighted

by external factors. This method for processing chain recommendation can help lower the barrier to entry of novice engineers, and can reduce the time required for expert users, when incorporated into an intelligent mixing environment [15].

2. METHODOLOGY

In order to investigate the ways in which processing chains are constructed by audio engineers, we conducted an experiment in which participants were asked to apply audio processing to a number of predefined audio samples, to achieve a specified timbral transformation. Subjects were provided with a range of audio effects, with no restrictions placed on the number of instances of an effect, or the order in which they can be selected.

Audio samples were taken from the *Mixing Secrets* library[1] and were selected to span a range of instruments and genres. The instruments, selected for their popularity and availability in the dataset were *acoustic guitar*, *bass guitar*, *drums* (mixed), *electric guitar*, *piano*, *saxophone*, *violin*, and *vocals*. In order to evaluate the effects of the channel type, *mixed* signals were also used. These covered 5 genres: *Reggae*, *Folk*, *Hip Hop*, *Rock* and *Jazz*. From the multi-track recordings 30 seconds long excerpts were selected, in which all the instruments were active.

To describe the transformations requested from the users, a range of timbral descriptors were obtained from the SAFE Project [12]. Firstly, the ten most frequent terms were chosen based on the number of entries into the dataset, and then the ten terms with the highest generality across audio effects. This was to ensure that both terms which are associated with a single audio effect and terms that are associated with multiple audio effects were used [12]. The terms were: *air*, *boom*, *bright*, *close*, *cream*, *crisp*, *crunch*, *damp*, *deep*, *dream*, *fuzz*, *punch*, *room*, *sharp*, *smooth*, *thick*, *thin*, and *warm*. The total number of terms was 18, as two of the most frequently used terms also displayed a high generality score. These were then stored in a relational database, resulting in 450 possible combinations of instrument, genre, and descriptor.

The tests were deployed using a web interface, in which subjects were given a URL[2] and asked to participate in their home studios. Whilst distributed tests like these contain more ambiguity due to uncontrolled variables such as listening environment and participant experience, we were able to collect larger amounts of data. This is a common practice in audio research [8, 16] provided suitable screening of participants takes place [17]. Participants were asked to follow the test instructions for a predetermined period of time. The duration of the tests was set to 5, 10, 15 or 30 minutes, depending on their availability. During this time, audio samples were presented along with a descriptor and a set of available audio effects.

The four audio effects available to test participants were (1) a parametric equaliser, (2) a dynamic range compressor, (3) a nonlinear distortion, and (4) an algorithmic reverberation. These were chosen to reflect the plugins available through the SAFE Project[3] [5], and were built using the JSAP web audio plugin framework [18]. Each time a processing chain is submitted, the plugins and their parameters are transmitted to a server along with an extensive set of differential audio features for each node in the chain, using the JS-Xtract feature extraction library. A full list of the audio features can be found in [19].

[1] Available at `http://www.cambridge-mt.com/ms-mtk.htm`

[2] Available at `http://dmtlab.bcu.ac.uk/nickjillings/safe-AMT/`

[3] Available at `http://www.semanticaudio.co.uk`

3. SINGLE EFFECTS

A total of 178 submissions were made over a two week period by 47 participants. Of the four available plugins, 124 equalisation, 72 compression, 57 reverb and 40 distortion effects were used. 90 of the 178 entries only use a single plugin (50.6%) with a further 64 only using two plugins (36.0%). The longest processing chain created is 4, giving 256 possible permutations. As there were no bounds on the number of plugins in a chain, this suggests that long processing chains are unnecessary for the given task. Tables 1, 2 and 3 show the number of entries per instrument, descriptor and genre respectively, distributed across the audio effect type.

Inst.	#	EQ	Comp.	Dist.	Revrb.
Ac. Gtr	8	4 (0.43)	3 (0.29)	1 (0.00)	4 (0.43)
El. Gtr.	14	9 (0.73)	4 (0.14)	5 (0.36)	5 (0.22)
Saxo.	4	4 (1.00)	2 (0.33)	2 (0.33)	1 (0.00)
Mix	36	22 (0.68)	17 (0.40)	6 (0.15)	15 (0.48)
Bs. Gtr.	31	22 (0.65)	12 (0.36)	7 (0.08)	8 (0.27)
Vocals	21	15 (0.81)	6 (0.16)	3 (0.04)	5 (0.27)
Drums	30	21 (0.57)	13 (0.53)	9 (0.19)	8 (0.27)
Violin	17	12 (0.70)	7 (0.26)	4 (0.27)	8 (0.36)
Piano	17	15 (0.81)	6 (0.53)	3 (0.07)	3 (0.20)

Table 1: *Number of entries for each instrument with the number of plugins applied and generality g_i across all descriptors in braces.*

Desc.	#	EQ	Comp.	Dist.	Revrb.
air	7	4 (0.60)	3 (0.40)	0 (0.00)	3 (0.40)
boom	7	7 (0.86)	4 (0.30)	0 (0.00)	0 (0.00)
bright	10	10 (0.73)	3 (0.33)	1 (0.00)	3 (0.33)
close	13	11 (0.64)	5 (0.40)	1 (0.00)	5 (0.67)
cream	9	8 (0.81)	4 (0.38)	0 (0.00)	3 (0.17)
crisp	9	9 (0.58)	3 (0.44)	0 (0.00)	1 (0.00)
crunch	15	5 (0.34)	7 (0.61)	14 (0.78)	2 (0.14)
damp	8	4 (0.75)	1 (0.00)	1 (0.00)	9 (0.83)
deep	9	9 (0.82)	3 (0.11)	0 (0.00)	2 (0.17)
dream	9	4 (0.50)	1 (0.00)	1 (0.00)	9 (0.82)
fuzz	11	2 (0.25)	0 (0.00)	11 (0.64)	1 (0.00)
punch	9	7 (0.86)	9 (0.71)	1 (0.00)	0 (0.00)
room	13	4 (0.50)	2 (0.17)	1 (0.00)	13 (0.72)
sharp	7	7 (0.86)	5 (0.53)	1 (0.00)	0 (0.00)
smooth	9	6 (0.67)	5 (0.48)	2 (0.20)	3 (0.40)
thick	9	8 (0.56)	5 (0.20)	3 (0.50)	1 (0.00)
thin	11	11 (0.73)	1 (0.00)	1 (0.00)	2 (0.17)
warm	13	11 (0.82)	5 (0.40)	2 (0.17)	3 (0.11)

Table 2: *Number of entries for each descriptor with the number of plugins applied and generality g_d across all descriptors in braces.*

3.1. Effect Generality

A plugin or transform can be considered *general* if the likelihood of it occurring is not bound by the measured term (instrument, genre or descriptor). If an effect is only applied to a single instance, it has a low generality score; if an effect is applied to every

Genre	#	EQ	Comp.	Dist.	Reverb.
Reggae	32	22 (0.61)	13 (0.43)	7 (0.21)	9 (0.33)
Jazz	33	24 (0.66)	20 (0.65)	9 (0.10)	13 (0.49)
Hip Hop	38	25 (0.64)	10 (0.39)	12 (0.21)	12 (0.34)
Folk	22	14 (0.684)	10 (0.31)	2 (0.07)	10 (0.31)
Rock	53	39 (0.60)	19 (0.47)	10 (0.20)	13 (0.20)

Table 3: *Number of entries for each genre with the number of plugins applied and generality g_{genre} across all descriptors in braces.*

Term	EQ	Comp.	Dist.	Reverb.
Genre	0.786	0.799	0.713	0.904
Instrument	0.692	0.625	0.656	0.640
Descriptor	0.766	0.631	0.291	0.464

Table 4: *Generality of plugins against the type of term (genre, instrument, descriptor).*

instance, it has a high generality score. Equations 1 and 2 calculate a single generality score from the data for a given contextual term (instrument, genre or descriptor). If a plugin only occurs for a small number of the terms, then the g will be low. These were adapted from [12]. Table 1 gives the generality of each plugin according to the source instrument.

$$g_i\left(p\right) = \frac{2}{D-1}\sum_{d=0}^{D-1} d\text{sort}\left(x\left(d,p\right)_i\right) \tag{1}$$

$$x\left(d,p\right)_i = \frac{n_p\left(d,i\right)}{\sum_{d=0}^{D-1} n_p\left(d,i\right)} \tag{2}$$

Here, $g_i\left(p\right)$ is the generality of a plugin p on instrument i. $n_p\left(d,i\right)$ is the number of plugin occurrences on descriptor d and instrument i. These measurements refer to the range of plugins are used to process a given instrument.

Table 1 shows that distortion is the least general effect whilst the EQ exhibits a consistently high generality score. The compressor is most general on Piano, Drums and full Mixes, suggesting that in these cases, the effects are applied irrespective of the genre and timbral descriptor. Table 2 measures the generality of descriptors across instrument classes. The genre, like the instrument, has little impact on the choice of plugin with all plugins attaining similar generality scores across the various genres, shown in table 4. Distortion is significantly less general when used on *Folk* samples, indicating that it must only be used in very specific use cases, with only 2 instances when a distortion was used from all 22 responses for *Folk*. The reverb effect followed similarly low generality scores.

Table 4 shows the cumulative generality scores for an effect being used across contexts: *genre*, *instrument* and *descriptor*. A low score here indicates the term has a high impact on whether a plugin appears. This suggests the genre and instrument play a relatively small role in the selection of effects in a processing chain. However the *descriptor* has an impact on whether distortion or reverb is used in the chain, indicating these only appear when a specific descriptor is used. EQ and compressor appear to be universally more general and can appear in any chain.

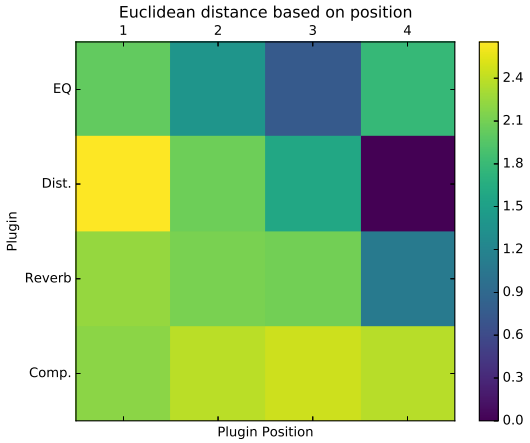

Figure 1: *Euclidean distances of the features according to plugin type (SAFE distortion, SAFE equaliser, SAFE compressor and SAFE reverb) and position in the chain*

3.2. Effect Salience

Each processor in the chain has a varying number of parameters, each set empirically by the participant. We quantify this by extracting audio features before and after each effect in the chain. Over 30 temporal and spectral features are averaged over each audio sample, extracted using JS-Xtract [19], in line with the features extracted by the SAFE project [5]. The impact of each plugin can then be characterised by the change in feature space before and after processing. We measure this using the Euclidean distance over each feature dimension, defined in Eq. 3.

$$d(\mathbf{p},\mathbf{q}) = \sqrt{\sum_{i=1}^{N}\left(q_i - p_i\right)^2} \tag{3}$$

Each feature is vector normalised against all other instances of that same feature within a processing chain, thus capturing relative salience in the chain. Fig. 1 shows the feature distance as a function of position in the chain across all entries into the dataset. This indicates the first plugin generally has the greatest impact, irrespective of plugin type. As the plugin index increases, the feature differences decrease. The mean effect chain length is 1.64, where the probability of an effect being selected for a given position in the chain is presented in Table 5.

Effect	1st	2nd	3rd	4th
EQ (E)	0.44	0.43	0.21	0.33
Compressor (C)	0.22	0.28	0.25	0.33
Distortion (D)	0.15	0.10	0.16	0.00
Reverb (R)	0.17	0.18	0.38	0.33

Table 5: *Probability of effects per chain position*

For the first position, which includes chains of single effects, the EQ is the most popular, appearing in 44.9% of the total instances, followed by the compressor (22%). The first and second positions retain the same structure, and the effects in order of descending popularity are EQ, compressor, reverb and distortion.

This aspect shifts when moving to the third position, were the most popular effect is the reverb (37.5%), followed by the compressor (25%), EQ (20%) and distortion (16%). Finally the fourth position is split equally between EQ, compressor and reverb with the distortion never appearing in that position.

3.3. Plugin Order

We consider each processing chain to be a multi-dimensional vector, where each dimension represents a plugin instance. Each index in the vector is then considered to have a finite state. The likelihood of a plugin appearing at position k, given the state at positions $\{0, \ldots, k-1\}$ can be evaluated using a Markov chain [20, 21]. Eq. 4 and 5 give the state transition matrix for the chain, highlighting the probability of the next plugin type given the previous plugin, defined in equation 6. A fifth state is entered which represents a blank plugin state. The chain must start at this *empty* plugin and the chain terminates once this state is re-entered. The state vector v comprises equalisation (E), compression (C), distortion (D), and reverb(R).

$$v = [\text{'None'}, \text{'E'}, \text{'C'}, \text{'D'}, \text{'R'}] \tag{4}$$

$$P = \begin{bmatrix} 0.000 & 0.645 & 0.555 & 0.675 & 0.544 \\ 0.449 & 0.000 & 0.250 & 0.200 & 0.316 \\ 0.191 & 0.250 & 0.013 & 0.025 & 0.088 \\ 0.124 & 0.056 & 0.111 & 0.000 & 0.053 \\ 0.235 & 0.048 & 0.069 & 0.100 & 0.000 \end{bmatrix} \tag{5}$$

$$\Pr\left(A_n = p_i | A_{n-1} = p_j\right) = P_{i,i-1} = P_{i,j} \tag{6}$$

$$\Pr\left(A_n = p_i | A_{n-1} = p_j, ..., A_0 = p_0\right) = \prod_{n=1}^{N} P\left(A_n, A_{n-1}\right) \tag{7}$$

Here, the probability that the k^{th} plugin is the last plugin in the chain is given by the first row, whilst the probability of the first plugin in the chain is given by the first column. The transition matrix can be used to generate all possible outcomes with their probabilities. Eq. 7 provides a formal definition, showing nodes in the chain be represented a probabilistic series of states. Using a Markov Chain, the most likely sequences from the model are: 1) EQ (29.0%), 2) reverb (12.8%), 3) compressor (10.6%), 4) distortion (8.3%), 5) compressor-EQ (6.2%) and 6) EQ-reverb (4.8%).

4. PROCESSING CHAINS

In total, 30 unique processing chains were constructed during the experiment. In order to compare the various combinations, plugin chains that were implemented only once in our dataset were excluded, leaving 19 unique entries. The mean usage of a processing chain is 8.78 times, and the most popular processing chains are EQ (27.5%), reverb (12.5%), compressor-EQ (11.9%), distortion (8.9%), EQ-compressor (8.9%) and EQ-reverb (5.3%). This correlates with effect transitions generated by the Markov Chain in Section 3.3.

To measure processing chain similarity, a matrix of descriptor occurrences per chain is constructed, using a distance measure based on the coexistence of terms in each pair of processing chains. We then compute pairwise distances to perform multidimensional scaling [22], followed by agglomerative clustering to establish a hierarchy of the plugin chains in the dataset, presented in Fig. 2. Plugin chains that are used to achieve similar terms, as is the case with EQ and EQ-compressor, are placed close in the hierarchy, whereas chains that do not share any descriptors, as is expected with distortion and reverb are placed further apart. Similarly, we can identify the relationship between transform descriptions based on the frequency of use across processing chains. In this case we represent the descriptive terms in multidimensional space, where each of the dimensions is the frequency of use for a given processing chain. A matrix D, with dimensions $M \times N$ is constructed, where M is the descriptor and N the unique plugin chain, and entry $D(i, j)$ is the amount of times plugin chain j was used to achieve descriptor i. This process allows us to perform hierarchical clustering, this time on the descriptive terms, presented in Fig. 3. The results show that the terms are organised in three predominant groups: a group that uses mainly distortion (*punch*, *fuzz*, *crunch*), a group that uses mainly reverb (*room*, *dream*, *damp*), and a group with high generality, distributed across a range of plugin

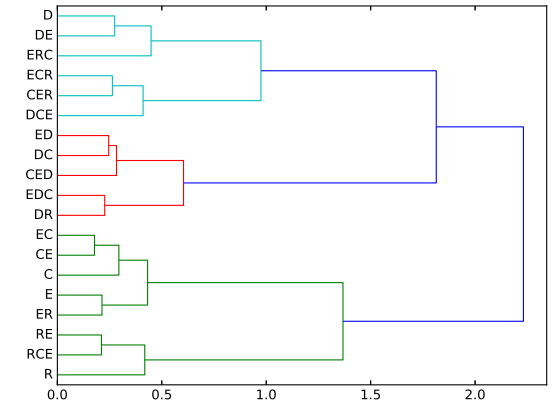

Figure 2: *Hierarchical clustering of unique chains based on term usage*

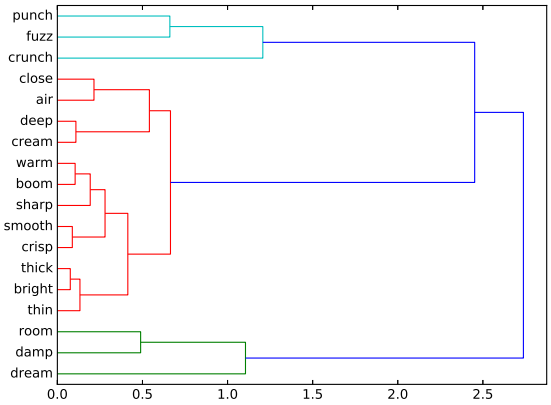

Figure 3: *Hierarchical clustering of unique terms based on processing chain usage.*

chains. For example *warm* is a descriptor that can be achieved by 8 different unique plugin chains, using 42% of the unique chains in our dataset, while *fuzz* makes use of just 15%.

4.1. Transform Similarity

An audio effect can have a more significant effect on the audio signal than others in the chain, based on its respective parameter space. In order to quantify this, audio feature differences across each effect in the chain are captured using Euclidean distance. This is represented as a matrix P with dimensions $M \times E$, where M represents the number of descriptors and E is the number of base effects (EQ, compressor, distortion, reverb). We can then apply dimensionality reduction to this matrix using PCA and project the audio effect classes into the low-dimensional space, as shown in Fig. 4.

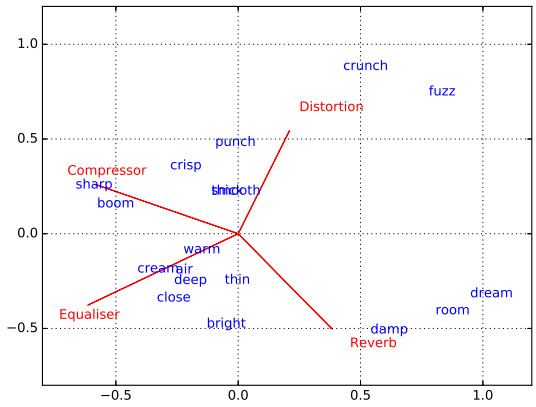

Figure 4: *Low-dimensional descriptor mapping with prevalent dimensions*

The figure shows that terms associated with specific effects, i.e. those with low generality scores, are highly correlated with the effect dimensions. *Fuzz* and *crunch* for example are correlated with distortion, *damp* and *room* with the reverb, and *sharp* with the compressor. Terms with a more general representation, such as *warm* tend to exhibit lower correlation scores.

5. FULL CHAIN RECOMMENDATION

5.1. Descriptor-based chain recommendation

The Markov chain approach to processing chain generation uses a matrix of conditional probabilities, based solely on the effect in the chain at position $k-1$. This method will be inherently biassed to favour plugins with a high generality (tables 1 and 2). For example, there is a high probability that the matrix P (Eq. 5) will generate a chain consisting of a single effect, given the likelihood of 60.7%. To improve the recommendation, we use a state transition matrix, based on the specific probabilities for each descriptive term P_d. Sequentially generating states using P_d will then produce a set of chains for the specified descriptor d. For instance, *fuzz* will generate a chain of just distortion with a likelihood of 74.38%, and an EQ-distortion chain with a likelihood of 16.53%.

By implementing an independent matrix for each timbral descriptor, we are also able to compare the way in which terms are organised in our generated processing chains, to the way the terms are organised in our dataset. By generating the same number of entries per transform, we construct a similarity matrix of terms, based on their frequency of plugin chain usage. We then apply dimensionality reduction and the resulting mapping is presented in Fig. 5. Here, descriptors that generate similar chains are placed together such as *room* and *damp*, or *sharp* and *punch*. This behaviour adds a level of versatility to the system, given that similar or identical chains, which can be used for achieving neighbouring descriptors, have a high probability of co-occurring.

To demonstrate the similarity of the original and generated descriptor mappings, the two spaces can be assessed using the *trustworthiness* (T_k) and *continuity* (C_k) metrics [23, 24], shown in Eq. 8 and 9.

$$T_k = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in U_i^{(k)}} (r(i,j) - k) \tag{8}$$

$$C_k = 1 - \frac{2}{nk(2n-3k-1)} \sum_{i=1}^{n} \sum_{j \in V_i^{(k)}} (\hat{r}(i,j) - k) \tag{9}$$

Here, the distances of the n entries in two spaces (U and V) are converted to ranks (r and $\hat{r}$) between points i and j. The measurements then evaluate the distributions of datapoint in the respective spaces over a number of neighbouring datapoint (k).

The low-dimensional space generated by the descriptor-wise state transition matrices (presented in Fig. 5) achieves a *trustworthiness* score of 0.78 for the original structure of unique terms (the space used to generate the clusters in Fig. 3). This suggests that the organisation of terms is preserved when generating processing chains using the Markov Chain approach. Similarly, for *continuity*, the structure of the probability matrix is retained with a score of 0.782.

For terms with low generality, i.e. those that have very specific plugin usage patterns such as *fuzz* and *dream* (see table 2), very specific plugin chains will appear from P_d. However, for entries which consist of more general effects, *warm*, *smooth* and *cream*, more chains can appear which have lower probabilistic scores. This can be interpreted as a low confidence score, thus producing more variance in the results. P_{smooth} generates the following chains: EQ (16.67%), EQ-compressor (13.33%), reverb (11.11%) and distortion (11.11%). These all have low and relatively equal probability of occurring, highlighting how unspecified this matrix is.

This can be improved by weighting the transition matrix probabilities to penalise plugins which are not related to the term. This is done by incorporating weights which indicate the plugin's prevalence in a chain. A weight w_p is found using Eq. 10 to 12.

$$\mathrm{w}_p = \frac{\sum_{n=0}^{N-1} \sum_{l=0}^{L-1} f(\mathrm{d}, l) g(x(l), p)}{\sum_{n=0}^{N-1} n} \tag{10}$$

$$f(\mathrm{d}, l) = \begin{cases} 1, & \text{if } l = \mathrm{argmax}(\mathrm{d}) \\ 0, & \text{otherwise} \end{cases} \tag{11}$$

$$g(x, p) = \begin{cases} 1, & \text{if } x = p \\ 0, & \text{otherwise} \end{cases} \tag{12}$$

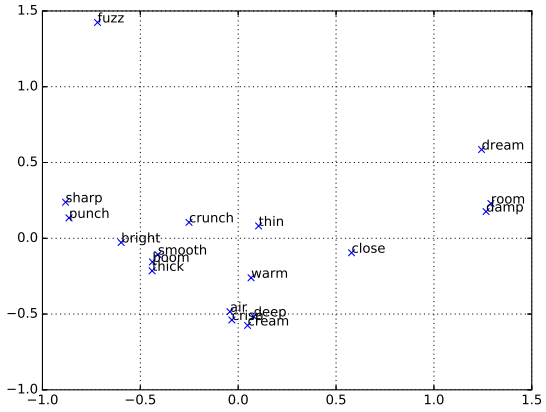

Figure 5:
Low-Dimensional Mapping for unweighted Markov Chains recommender

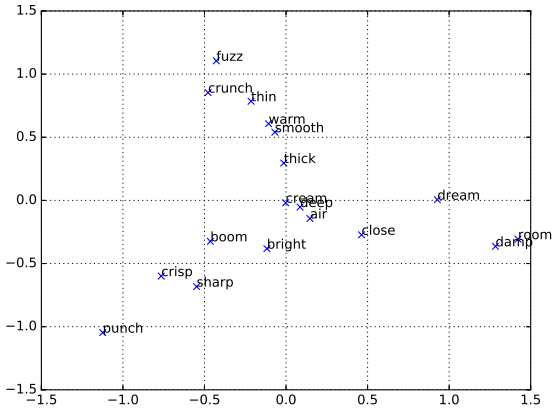

Figure 6: *Low-Dimensional Mapping for weighted Markov Chains recommender*

Here, N is the number of chains, L the length of chain n, d is a vector of the plugin Euclidean distances (Eq. 3) and x a vector of the plugin codes. Function f (Eq. 11) returns 1 if the plugin at position l is the most prevalent effect and function g (Eq. 12) returns a 1 if the plugin at position l is the same as plugin p. The weights are then multiplied with the corresponding row of the descriptor's transition matrix.

Applying weights to the probability increases the possibility of the prevalent effect(s) appearing at any position in the chain. In this manner the system gains an additional level of adaptivity, being able to recommend chains that might not exist in the original dataset, but concurrently takes into account the most important plugin in the chain. For example, using the weighted transition matrix, we are able to predict chains for the the *smooth* descriptor with higher accuracy, predicting compressor (28.87%), EQ (21.66%) and EQ-compressor (15.28%).

The distribution of terms using the weighted Markov chain approach are presented in Fig. 6. Using the *trustworthiness* and *continuity* metrics, the original structure of the descriptors is retained at a value of 0.75, and the *continuity* between the transformed space to the original data has increased to 0.86.

5.2. Instrument-based chain recommendation

As the source instrument also proves to be a salient attribute plugin selection, we can apply the same weighting method to the instrument classes P_i. This will allow the effects, which are specific to an instrument to be favoured in the processing chain. For instance, for the *Mix* samples, the Markov chain method generates EQ (19.69%), reverb (17.78%) and compressor (13.73%). However, applying the weighting w_i based on the most prevalent effect does not improve the model. With the weights, the generated processing chains for *Mix* are compressor (30.05%), EQ (25.69%) and reverb (21.44%). Thus, applying the weights reduces the likelihood of complex chains occurring. The weights for the *Mix*, w_{Mix} are 0.250, 0.333, 0.138 and 0.278 for the EQ, compressor, distortion and reverb respectively. Conversely, the weights for the *crunch*, w_{crunch} are 0.066, 0.266, 0.667 and 0. This shows that whilst certain plugins are applied more generally to a given instrument class, the most prevalent effect in a chain is actually driven by the descriptor. The weights for the *mix*ed samples are all relatively similar, except the distortion, which means after scaling and normalising the matrix, a very similar transition matrix is created.

6. CONCLUSION

We have introduced a method for audio processing chain recommendation, based on a dataset of user-inputs. We captured information regarding the instrument, genre and descriptor and used them to weight a state transition matrix. To evaluate the output of the model, we measured the similarity of descriptor mappings in low-dimensional space using *trustworthiness* and *continuity*, and we showed that a descriptor-based Markov chain method achieves a score of $T(k) = .78$, $C(k) = .782$ and the weighted descriptor-based model achieves a score of $T(k) = .75$, $C(k) = .86$.

We provide an evaluation of plugin usage in processing chains and show that the role of genre is negligible in plugin selection, whilst the descriptor heavily influences this decision making process. The EQ and compressor plugins both exhibit high generality, which suggests they are selected for most processing chains irrespective of instrument or descriptor. The distortion and reverb plugins are very specific, which means they are more frequently used when a specific timbral transformation is required.

7. ACKNOWLEDGMENTS

The work of the first author is supported by The Alexander S. Onassis Public Benefit Foundation.

8. REFERENCES

[1] Joshua Reiss, "Intelligent systems for mixing multichannel audio," in *17th International Conference on Digital Signal Processing*, July 2011, pp. 1–6.

[2] Enrique Perez-Gonzalez and Joshua D. Reiss, "Automatic gain and fader control for live mixing," in *2009 IEEE Work-*

shop on Applications of Signal Processing to Audio and Acoustics, Oct 2009, pp. 1–4.

[3] Stuart Mansbridge, Saorise Finn, and Joshua D. Reiss, "Implementation and evaluation of autonomous multi-track fader control," in *Audio Engineering Society Convention 132*, 2012.

[4] Stuart Mansbridge, Saorise Finn, and Joshua D. Reiss, "An autonomous system for multitrack stereo pan positioning," in *Audio Engineering Society Convention 133*, 2012.

[5] Ryan Stables, Sean Enderby, Brecht De Man, György Fazekas, and Joshua D. Reiss, "SAFE: A system for the extraction and retrieval of semantic audio descriptors," in *15th International Society for Music Information Retrieval Conference*, 2014.

[6] Spyridon Stasis, Ryan Stables, and Jason Hockman, "Semantically controlled adaptive equalisation in reduced dimensionality parameter space," *Applied Sciences*, vol. 6, no. 4, 2016.

[7] Spyridon Stasis, Jason Hockman, and Ryan Stables, "Navigating descriptive sub-representations of musical timbre," in *Proceedings of the International Conference on New Interfaces for Musical Expression*, Copenhagen, Denmark, 2017.

[8] Mark Cartwright and Bryan Pardo, "Social-EQ: Crowdsourcing an equalization descriptor map," in *Proceedings of the International Society of Music Information Retrieval*, 2013, pp. 395–400.

[9] Sean Enderby and Ryan Stables, "A nonlinear method for manipulating warmth and brightness," in *Proceedings of the 20th International Conference on Digital Audio Effects*, Edinburgh, UK, 2017.

[10] Jacob A Maddams, Saorise Finn, and Joshua D Reiss, "An autonomous method for multi-track dynamic range compression," in *Proceedings of the 15th International Conference on Digital Audio Effects*, 2012.

[11] Prem Seetharaman and Bryan Pardo, "Reverbalize: a crowdsourced reverberation controller," in *Proceedings of the 22nd ACM international conference on Multimedia*, 2014, pp. 739–740.

[12] Ryan Stables, Brecht De Man, Sean Enderby, Joshua D. Reiss, György Fazekas, and Thomas Wilmering, "Semantic description of timbral transformations in music production," in *Proceedings of the 2016 ACM on Multimedia Conference*, 2016, pp. 337–341.

[13] Taylor Zheng, Prem Seetharaman, and Bryan Pardo, "SocialFX: Studying a crowdsourced folksonomy of audio effects terms," in *Proceedings of the 2016 ACM on Multimedia Conference*. ACM, 2016, pp. 182–186.

[14] Roey Izhaki, *Mixing audio: concepts, practices and tools*, Taylor & Francis, 2013.

[15] Nicholas Jillings and Ryan Stables, "Investigating music production using a semantically powered digital audio workstation in the browser," in *Audio Engineering Society Conference on Semantic Audio*, Erlangen, Germany, June 2017.

[16] Michael H. Birnbaum, "Human research and data collection via the internet," *Annual Review of Psychology*, vol. 55, pp. 803–832, October 2003.

[17] Mark Cartwright, Bryan Pardo, Gautham J. Mysore, and Matt Hoffman, "Fast and easy crowdsourced perceptual audio evaluation," in *2016 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, March 2016, pp. 619–623.

[18] Nicholas Jillings, Yonghao Wang, Joshua D. Reiss, and Ryan Stables, "JSAP: A plugin standard for the web audio api with intelligent functionality," in *Audio Engineering Society Convention 141*, 2016.

[19] Nicholas Jillings, Jamie Bullock, and Ryan Stables, "JSXtract: A realtime audio feature extraction library for the web," in *17th International Society for Music Information Retrieval Conference*, 2016.

[20] Andrey Markov, "Extension of the limit theorems of probability theory to a sum of variables connected in a chain," *Dynamic Probabilistic Systems*, vol. 1, 1971.

[21] George Tauchen, "Finite state markov-chain approximations to univariate and vector autoregressions," *Economics Letters*, vol. 20, no. 2, pp. 177 – 181, 1986.

[22] Warren S. Torgerson, *Theory and methods of scaling.*, Wiley, 1958.

[23] Jarkko Venna and Samuel Kaski, "Visualizing gene interaction graphs with local multidimensional scaling.," in *ESANN*, 2006, vol. 6, pp. 557–562.

[24] Laurens Van Der Maaten, Eric Postma, and Jaap Van den Herik, "Dimensionality reduction: a comparative," *J Mach Learn Res*, vol. 10, pp. 66–71, 2009.