On symbolic representations and transformations of sound
The theory of sound-types

Carmine-Emanuele Cella

Università di Bologna/IRCAM Paris

December 1, 2011
Music, in its final stage of *performance*, can be described in many ways (time-varying signal, symbolic system, etc.).
Music, in its final stage of *performance*, can be described in many ways (time-varying signal, symbolic system, etc.).

Each approach selects a particular degree of abstraction: the *signal level*, the *symbolic level*, a fixed mixture of both (*mid-levels*).
Different representations (2)

- The signal level is efficient and invertible but has a low degree of abstraction.
Different representations (2)

- The signal level is efficient and invertible but has a low degree of abstraction.
- The symbolic level can define complex relationships between objects but is not invertible and is not physical.
The signal level is efficient and invertible but has a low degree of abstraction.

The symbolic level can define complex relationships between objects but is not invertible and is not *physical*.

Mid-levels are based on perceptual criteria related to hearing and are in between lower and higher levels.
The theory of sound-types

Different representations (2)

- The signal level is efficient and invertible but has a low degree of abstraction.
- The symbolic level can define complex relationships between objects but is not invertible and is not *physical*.
- Mid-levels are based on perceptual criteria related to hearing and are in between lower and higher levels.
- They have a *fixed degree of abstraction*. 
The signal level is efficient and invertible but has a low degree of abstraction.

The symbolic level can define complex relationships between objects but is not invertible and is not physical.

Mid-levels are based on perceptual criteria related to hearing and are in between lower and higher levels.

They have a fixed degree of abstraction.

They impose their own concepts onto the signal.
This research aims at creating a representation method for music (the \textit{theory of sound-types}) that fulfills \textit{by-design} the following requirements:

- \textit{Signal-dependent semantics}: the involved concepts of the representation should be inferred from the signal.
This research aims at creating a representation method for music (the *theory of sound-types*) that fulfills *by-design* the following requirements:

- **Signal-dependent semantics**: the involved concepts of the representation should be inferred from the signal.
- **Scalability**: it should be possible to change the *degree of abstraction* in the representation, ranging from the signal level to the symbolic level.
Variable abstraction representation (2)

- *Weak invertibility*: the representation method should be able to generate the represented signal; the generated signal must not be *waveform*-identical to the original one.
Variable abstraction representation (2)

- **Weak invertibility**: the representation method should be able to generate the represented signal; the generated signal must not be *waveform*-identical to the original one.

- **Generativity**: is the possibility to generate sounds *other* than the original one, according to some parameters in the domain of the representation.
A theory to represent sounds by means of *types* and *rules* inferred by some low-level descriptions of signals and subsequent learning stages.
1. A theory to represent sounds by means of types and rules inferred by some low-level descriptions of signals and subsequent learning stages.

2. It takes inspiration from the simple type theory and from \( \lambda \)-calculus.
A theory to represent sounds by means of *types* and *rules* inferred by some low-level descriptions of signals and subsequent learning stages.

It takes inspiration from the *simple type theory* and from \( \lambda \)-calculus.

The types represent classes of equivalences for sounds, while the rules represent transition probabilities that a type is followed by another type.
The decomposition of a signal $x[n]$ into expansion functions is a linear combination of the form:

$$\tilde{x} = \sum_{k=1}^{K} \alpha_k \ g_k[n].$$

The coefficients $\alpha_k$ are derived from the analysis stage, while the functions $g_k[n]$ can be determined by the analysis stage or fixed beforehand.
Mathematically, this theory tries to *translate* the original equation:

\[
x[n] = \sum_{k=1}^{K} \alpha_k g_k[n] \\
= \alpha_1 g_1[n] + \ldots + \alpha_k g_k[n] \\
= \beta_1 f_1[n] + \ldots + \beta_j f_j[n] \\
\vdots \\
= \omega_1 h_1[n] + \ldots + \omega_t h_t[n]
\]

where \( \alpha, \beta, \ldots, \omega \) are any kind of weighting coefficients, \( g_k, f_j, \ldots, h_t \) are variables belonging to different types. All the theory presented above can be realized by defining the sound-types transform (STT).
The sound-types transform (1)

- Given a signal $\vec{x}$ of length $N$-samples and a window $\vec{h}$ of length $n$-samples, it is possible to define an **atom** as a windowed chunk of the signal of length $n$-samples:

$$\vec{a} = \vec{h} \cdot \vec{x}.$$
Given a signal $\mathbf{x}$ of length $N$-samples and a window $\mathbf{h}$ of length $n$-samples, it is possible to define an atom as a windowed chunk of the signal of length $n$-samples:

$$\mathbf{a} = \mathbf{h} \cdot \mathbf{x}.$$ 

A sound-cluster as a set of atoms that lie in a defined area of a feature space (ie. that share a similar set of features):

$$\mathbf{c}_r = \{\mathbf{a}_{r,1}, \ldots, \mathbf{a}_{r,k_r}\}.$$ 

The content of $\mathbf{c}_r$ is given by a statistical analysis applied on the feature space.
The sound-types transform (2)

A **model** $\mathcal{M}_{N}^{\vec{x}}$ of the signal $\vec{x}$ is defined as the set of the clusters discovered on it:

$$\mathcal{M}_{N}^{\vec{x}} = \{ \vec{c}_1, \ldots, \vec{c}_r \}.$$
A model $\mathcal{M}_N^\vec{x}$ of the signal $\vec{x}$ is defined as the set of the clusters discovered on it:

$$\mathcal{M}_N^\vec{x} = \{\vec{c}_1, \ldots, \vec{c}_r\}.$$ 

The cardinality $|\mathcal{M}_N^\vec{x}|$ of the model is also called the abstraction level of the analysis; since the number of atoms is $N/t$ it is evident that $1 \leq |\mathcal{M}_N^\vec{x}| \leq N/t$ with higher abstraction being 1 and lower abstraction being $N/t$ (where $t$ is the overlapping factor used to create atoms).
A sound-cluster has an associate **sound-type** $\vec{\tau}_r$, defined as the weighted sum of all the atoms in the sound-cluster where the weights $\vec{\omega}_r$ are the distances of each atom to the center of the cluster:

$$\vec{\tau}_r = \sum_{j=1}^{k_r} \vec{a}_{r,j} \cdot \omega_{r,j}$$

with $\omega_{r,j} \in \vec{\omega}_r$. 

The set of sound-types in the signal $\vec{x}$ is called **dictionary**:

$$D_{\vec{x}} = \{\vec{\tau}_1, ..., \vec{\tau}_{k_r}\}$$
The sound-types transform (3)

- A sound-cluster has an associate **sound-type** $\vec{\tau}_r$, defined as the weighted sum of all the atoms in the sound-cluster where the weights $\omega_{r,j}$ are the distances of each atom to the center of the cluster:

$$\vec{\tau}_r = \sum_{j=1}^{k_r} a_{r,j} \cdot \omega_{r,j}$$

with $\omega_{r,j} \in \omega_r$.

- The set of sound-types in the signal $\vec{X}$ is called **dictionary**:

$$\mathcal{D}_{\vec{X}} = \{ \vec{\tau}_1, \ldots, \vec{\tau}_r \}.$$
Finally, it is possible to define the *sound-types transform* as a function of time and frequency obtained by multiplying the sound-types in a given dictionary with complex sinusoids:

\[
\Phi_n^k = \frac{N}{t} \sum_{i=0}^{N} \tau_i, p \cdot e^{-j \cdot \frac{2 \pi}{n} \cdot k}
\]

where \( \vec{k} = \{f_1, \ldots, f_n\} \) is a vector of frequencies.
The extreme case for $|\mathcal{M}| = N/t$ is interesting: for that abstraction level, each sound-cluster is a singleton made of a single atom and consequently each sound-type reduces to that single atom scaled in amplitude:

$$|\mathcal{M}| = N/t \implies \vec{c}_r = \{\vec{a}_1\} \implies \vec{\tau}_r = \vec{a}_r \cdot \omega_{r,1}.$$
The sound-types transform (5)

- The extreme case for $|\mathcal{M}| = N/t$ is interesting: for that abstraction level, each sound-cluster is a singleton made of a single atom and consequently each sound-type reduces to that single atom scaled in amplitude:

$$|\mathcal{M}| = N/t \implies \frac{1}{n} \mathbf{c}_r = \{\mathbf{a}_1\} \implies \mathbf{\tau}_r = \mathbf{a}_r \cdot \omega_{r,1}.$$ 

- This leads to the important consequence that STT is a generalization of STFT:

$$\mathbf{\vec{\tau}}_r = \mathbf{\vec{a}}_r = \mathbf{h} \cdot \mathbf{x} \implies \sum_{i=0}^{N/t} \mathbf{\vec{\tau}}_{r,p} \cdot e^{-j \cdot \frac{2 \cdot \pi}{n} \cdot \mathbf{k}} = \sum_{i=0}^{N/t} \mathbf{h} \cdot \mathbf{\chi}_{i,t} \cdot e^{-j \cdot \frac{2 \cdot \pi}{n} \cdot \mathbf{k}}$$

with $p$ defined as above.
Computational criteria

- Some criteria are needed to define types and rules
Some criteria are needed to define types and rules.
They can be obtained by means of a twofold process:

- **Types inference**: first, the types involved in the representations are discovered by looking for common entities in a sound.
- **Rules inference**: a second stage is needed to discover the rules that link one type to another by means of a sequential analysis.
A possible realization is based on low-level descriptors plus classification for types inference and Markov models for rules inference:

- **(atomic decomposition):** subdivide a sound into small grains of approximately 40 ms called *atoms* or *0-types* overlapping in time and frequency
Creation of sound-types (1)

A possible realization is based on low-level descriptors plus classification for types inference and Markov models for rules inference:

- **(atomic decomposition):** subdivide a sound into small grains of approximately 40 ms called *atoms* or *0-types* overlapping in time and frequency

- **(1-types inference):** compute a set of low-level descriptors on the atoms obtained in the previous step, project the descriptors in a multi-dimensional space and compute the *clusters*; each cluster will represent a *1-type*

- **(1-rules inference):** implement a Markov model to describe the sequences of types present in the analysed sound (*1-rules*)
Creation of sound-types (2)

- **(n-types inference):** compute a set of low-level descriptors on the whole sequences found in the previous step; project again the descriptors and compute the clusters: each cluster will represent a *n-type*

- **(n-rules inference):** repeat the Markov model until there are no more sequences (*n-rules*).
The number of iterations represents the degrees of abstraction
The number of iterations represents the degrees of abstraction

Discovered types are defined in the time-frequency plane and have increasing time scale
Sound-types: properties (1)

1. The number of iterations represents the degrees of abstraction.
2. Discovered types are defined in the time-frequency plane and have increasing time scale.
3. The higher the level of a type, the more will be expressive (the less generic).
Sound-types: properties (2)

- *Signal-dependent semantics*: the atoms and the + relation are derived from the signal.

- *Scalability*: the possibility to scale over abstraction is implicit to theories of types; we showed how it is possible to translate a representation to another by changing the involved elements and operators.

- *Weak invertibility* and *generativity*: there are many possibilities to create a signal back from sound types: pick up randomly an element of each cluster used, pick up the element closest to the center of the cluster or to generate a weighted sum of all the elements of a cluster, etc.
A generalized framework

Back-end

SIGNAL LEVEL

Atomic decomposition
Onsets
Windowing

CLUSTERS AND SEQUENCES

GMM, K-means, etc.
HMM, Factor oracle, etc.

Classes and probabilities (n-types, n-rules)

SYMBOlIC LEVEL

Strings
Lambda-like calculus

Front-end
A tool called **Clusters** has been implemented in C++ to analyze sounds and produce quasi-symbolic representations:

- Low-level features analysis of sounds (spectral and temporal) with dimensionality reduction (PCA).
A tool called Clusters has been implemented in C++ to analyze sounds and produce quasi-symbolic representations:

- Low-level features analysis of sounds (spectral and temporal) with dimensionality reduction (PCA).
- Clusters estimation with $K$-means and GMM (types).
A tool called **Clusters** has been implemented in C++ to analyze sounds and produce quasi-symbolic representations:

- Low-level features analysis of sounds (spectral and temporal) with dimensionality reduction (PCA).
- Clusters estimation with $K$-means and GMM (types).
- Auto-estimation of number of clusters (gap statistic, BIC).
A tool called **Clusters** has been implemented in C++ to analyze sounds and produce quasi-symbolic representations:

- Low-level features analysis of sounds (spectral and temporal) with dimensionality reduction (PCA).
- Clusters estimation with $K$-means and GMM (types).
- Auto-estimation of number of clusters (gap statistic, BIC).
- Within-cluster dispersion for outliers detection.
A tool called Clusters has been implemented in C++ to analyze sounds and produce quasi-symbolic representations:

- Low-level features analysis of sounds (spectral and temporal) with dimensionality reduction (PCA).
- Clusters estimation with $K$-means and GMM (types).
- Auto-estimation of number of clusters (gap statistic, BIC).
- Within-cluster dispersion for outliers detection.
- Inverse transformations (original sound reconstruction, probabilistic generation and hybridization).
A tool called **Clusters** has been implemented in C++ to analyze sounds and produce quasi-symbolic representations:

- Low-level features analysis of sounds (spectral and temporal) with dimensionality reduction (PCA).
- Clusters estimation with $K$-means and GMM (types).
- Auto-estimation of number of clusters (gap statistic, BIC).
- Whitin-cluster dispersion for outliers detection.
- Inverse transformations (original sound reconstruction, probabilistic generation and hybridization).
- Transition probabilities with a Markov model.
Clusters (1)

A tool called **Clusters** has been implemented in C++ to analyze sounds and produce quasi-symbolic representations:

- Low-level features analysis of sounds (spectral and temporal) with dimensionality reduction (PCA).
- Clusters estimation with $K$-means and GMM (types).
- Auto-estimation of number of clusters (gap statistic, BIC).
- Within-cluster dispersion for outliers detection.
- Inverse transformations (original sound reconstruction, probabilistic generation and hybridization).
- Transition probabilities with a Markov model.
- Symbolic representation with a simple string.
Clusters (2): types representation (3D case)
Reconstruction

- Original sample 1 (rock band).
Clusters (4): inverse transformation examples 1

Reconstruction

- Original sample 1 (rock band).
- 170 types (10%), 11 mixed features, weighted, taxicab.
Reconstruction

- Original sample 1 (rock band).
- 170 types (10%), 11 mixed features, weighted, taxicab.
- 860 types (50%), 4 spectral features, weighted, taxicab.
Reconstruction

- Original sample 1 (rock band).
- 170 types (10%), 11 mixed features, weighted, taxicab.
- 860 types (50%), 4 spectral features, weighted, taxicab.
- 1600 types (95%), 11 spectral features, weighted, cosim.
Reconstruction

- Original sample 1 (rock band).
- 170 types (10%), 11 mixed features, weighted, taxicab.
- 860 types (50%), 4 spectral features, weighted, taxicab.
- 1600 types (95%), 11 spectral features, weighted, cosim.
- Original sample 2 (orchestra) and 100% reconstruction.
Clusters (5): bad signal reconstruction

Under a given ratio between types and atoms, the signals are not reconstructed correctly. The following example shows such a situation:
Transformations and generations

- Original sample 3 (lute) and two times slower resynthesis.
Clusters (6): inverse transformation examples 2

Transformations and generations

- Original sample 3 (lute) and two times slower resynthesis.
- Original sample 4 (voice) and one fifth up resynthesis.
Clusters (6): inverse transformation examples 2

Transformations and generations

- Original sample 3 (lute) and two times slower resynthesis.
- Original sample 4 (voice) and one fifth up resynthesis.
- Original sample 5 (bass) and its probabilistic generation.
Transformations and generations

- Original sample 3 (lute) and two times slower resynthesis.
- Original sample 4 (voice) and one fifth up resynthesis.
- Original sample 5 (bass) and its probabilistic generation.
- Original sample 6 (piano) and its probabilistic generation.
Transformations and generations

- Original sample 3 (lute) and two times slower resynthesis.
- Original sample 4 (voice) and one fifth up resynthesis.
- Original sample 5 (bass) and its probabilistic generation.
- Original sample 6 (piano) and its probabilistic generation.
- A probabilistic generation of sample 2 (orchestra).
Clusters (6): inverse transformation examples 2

Transformations and generations

- Original sample 3 (lute) and two times slower resynthesis.
- Original sample 4 (voice) and one fifth up resynthesis.
- Original sample 5 (bass) and its probabilistic generation.
- Original sample 6 (piano) and its probabilistic generation.
- A probabilistic generation of sample 2 (orchestra).
- Hybridization between sample 2 (orchestra) and sample 6 (piano) [mixing].
Clusters (6): inverse transformation examples 2

Transformations and generations

- Original sample 3 (lute) and two times slower resynthesis.
- Original sample 4 (voice) and one fifth up resynthesis.
- Original sample 5 (bass) and its probabilistic generation.
- Original sample 6 (piano) and its probabilistic generation.
- A probabilistic generation of sample 2 (orchestra).
- Hybridization between sample 2 (orchestra) and sample 6 (piano) [mixing].
- Hybridization between sample 2 (orchestra) and sample 6 (piano) [merging].
Symbolic description: analysing the created string of labels, it is possible to acquire information of salient properties of the sound and represent it in a meaningful way:

\[ x[n] = 0 + 1 + 2 + 3 + \ldots + \ldots + \ldots + \ldots + n \]
\[ = \alpha_1 + \ldots + \ldots + \ldots + \alpha_k \]
\[ = \beta_1 + \ldots + \ldots + \beta_j \]
\[ \vdots \]
\[ = \omega_1 + \ldots + \omega_t \]

where \( n > k > j > t \) and the types increase their amount of information from one level to another.
Audio compression: while not being the main purpose of the approach, it is possible to compress a sound by a given ratio.
Audio compression: while not being the main purpose of the approach, it is possible to compress a sound by a given ratio.

Time and frequency transformations: it is possible to perform various transformations such as time-stretch and pitch-shift.
Clusters (8): current applications II/II

- **Audio compression**: while not being the main purpose of the approach, it is possible to compress a sound by a given ratio.

- **Time and frequency transformations**: it is possible to perform various transformations such as time-stretch and pitch-shift.

- **Probabilistic generation**: using the types and their probabilities it’s possible to generate sounds related to the original, but different.
Audio compression: while not being the main purpose of the approach, it is possible to compress a sound by a given ratio.

Time and frequency transformations: it is possible to perform various transformations such as time-stretch and pitch-shift.

Probabilistic generation: using the types and their probabilities it’s possible to generate sounds related to the original, but different.

Hybridization (still experimental): using the types of one sound and the probabilities of another one it’s possible to create hybrid sounds.
The smaller the number of clusters (meaning that we reduce the number of clusters, grouping more entities in the same sound-type) the better will be the representation but the worst will be the sound resynthesis.
The smaller the number of clusters (meaning that we reduce the number of clusters, grouping more entities in the same sound-type) the better will be the representation but the worst will be the sound resynthesis.

Sound quality is directly linked to the number of clusters, but if we augment this number we loose the possibility of having a compact representation analysis.
Is the *within-cluster dispersion* is a measure for quality?
Future applications (1)

- **Selective transformations**: it should be possible to perform various transformations only on some *selected* types (let’s suppose that some discovered types represent the vowels of a singing voice: it should be then possible to operate only on those types selectively).
Future applications (1)

- **Selective transformations**: it should be possible to perform various transformations only on some selected types (let’s suppose that some discovered types represent the vowels of a singing voice: it should be then possible to operate only on those types selectively).

- **Pattern recognition on the representation**: the tool represents a sound using a string of labels; many algorithms could be applied on that string to discover patterns; for example if at a given level the representation is $\alpha\beta\beta\ldots\gamma\delta\delta$, it could be possible to rewrite this with a function of two variables such as $\phi(x, y) = xyy$ thus having $\alpha\beta\beta\ldots\gamma\delta\delta = \phi(\alpha, \beta)\ldots\phi(\gamma, \delta)$. 
Affective classification and generation of spectra: when a sound has been represented in term of sound-types and probabilities, a supervised labelling could be applied on the discovered elements in order to classify them by means an affective model: some types could be called, for example, rough or sad. It could be possible, then, to ask the machine to generate similar sounds by means of the discovered probabilities.
The theory of sound-types

Introduction
The theory of sound-types
Implementation
Conclusions and perspectives

Outline of some possible expansions

Audio signal

Temporal segmentation (windows, onsets, atoms, etc.)

LLD computation (spectral, temporal, etc.)

Clustering (k-means, gmm, etc.)

Sequential analysis (HMM, factor oracle, etc.)

Pattern recognition on labels
DNA sequences alignment
Symbolic representation (labels, formulas, graphs, etc.)

Sound-types (classes and probabilities)

Probabilistic generation
Calculus-based generation (lambda-like)
Affective classification and generation of spectra
Selective transformations
Sound hybridization
Defective reconstructions
Any questions?

- C. E. Cella, *Towards a Symbolic Approach to Sound Analysis*, MCM 2009, Yale University - New Haven (CT), Springer.