KTH ROYAL INSTITUTE
OF TECHNOLOGY

DT2470: Music Informatics

Bob L. Sturm (TMH)
André Holzapfel (MID)

Who are we?

Bob Sturm

bobs@kth.se

André Holzapfel

holzap@kth.se

What is music informatics?

How can we connect users with music and information about music

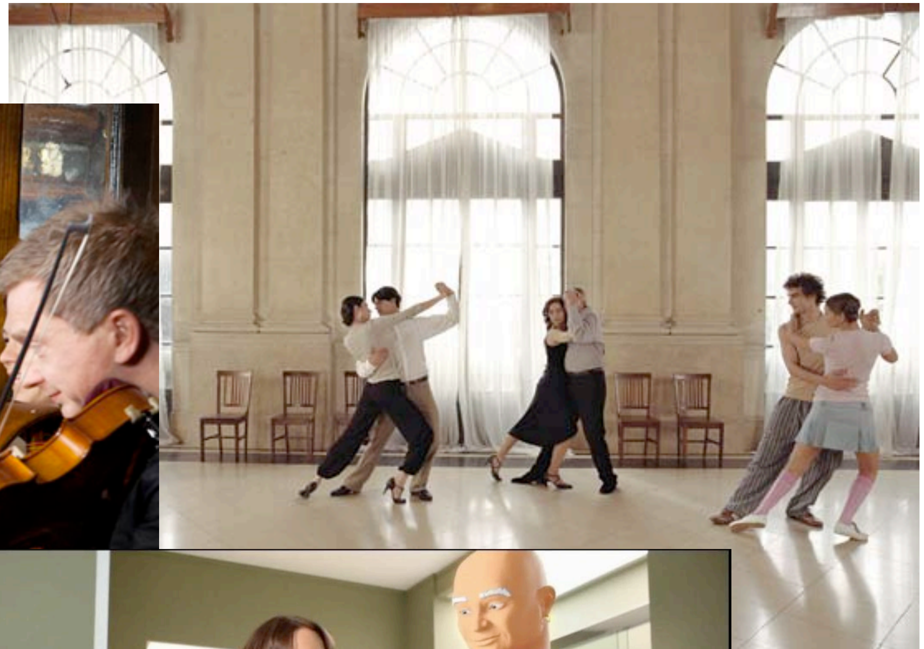
How can we help users make music and information about music

People, Scholars, Musicians, Schools, Organisations, Archives, Rights holders, Businesses, …

Michael Tompsett, *Musical Instruments Map Of The World Map* (http://fineartamerica.com)

Why do we want that?

Music is everywhere, and so is the need for music and information about music.



Music informatics:

Methods for making music data as accessible as text

Quick question

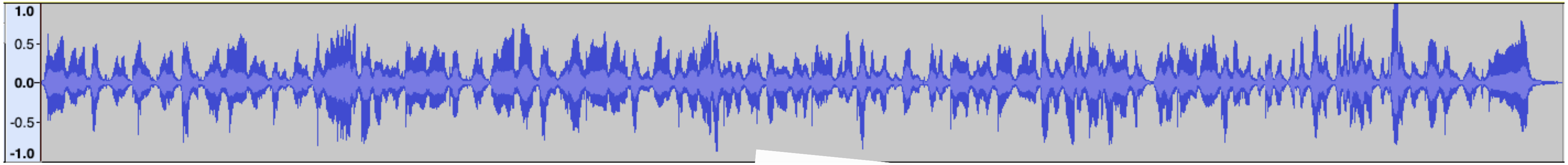
Who knows what music is?

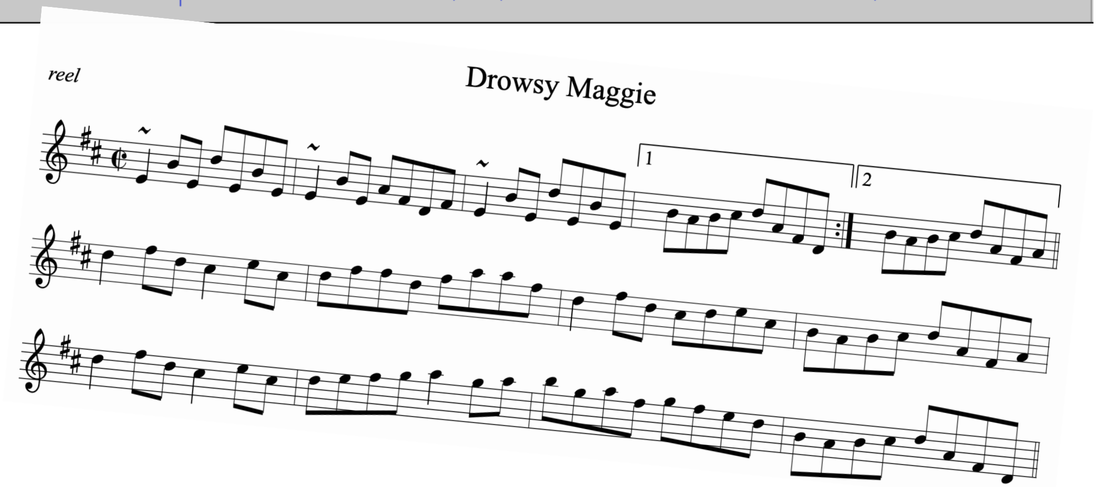
Who are you?

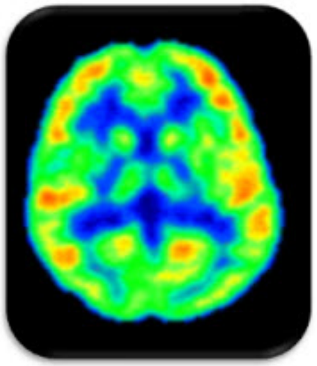
Tell us about yourself:

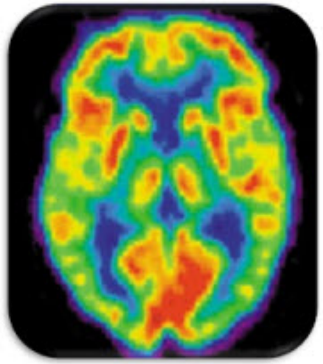
1. Name
2. Degree program
3. Interest in music informatics
4. Experience with music
5. Experience with python programming
6. Experience with signal processing
7. Experience with machine learning

Music data comes in many forms



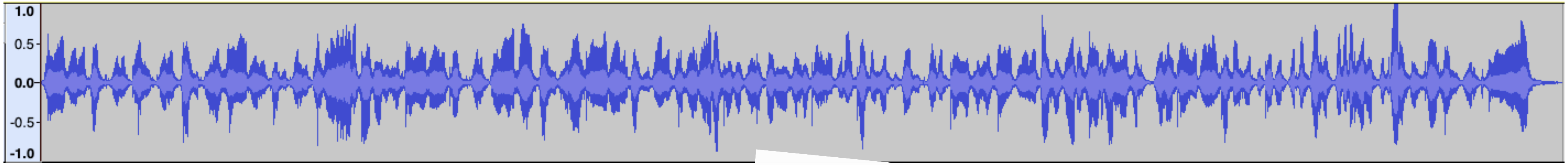


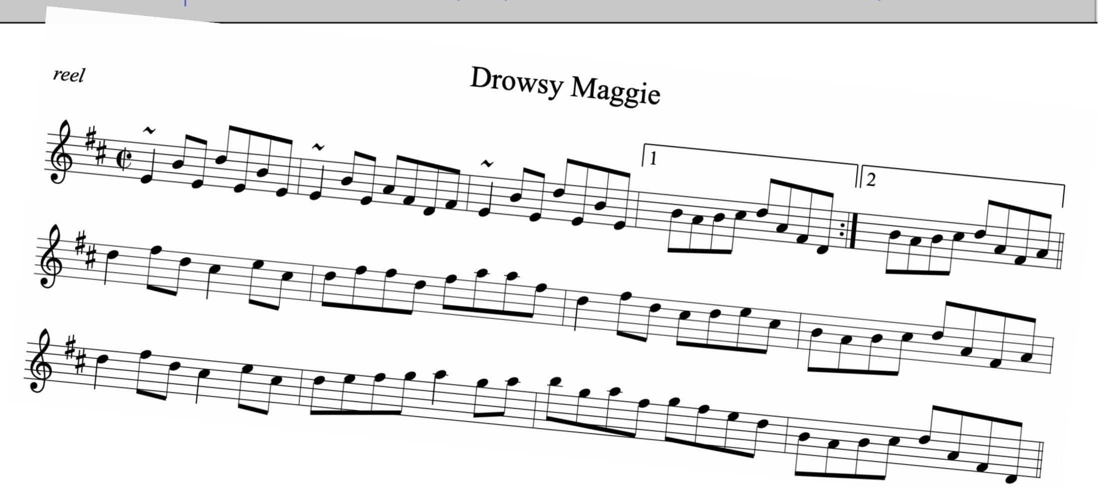


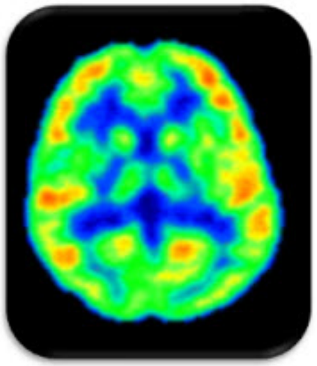


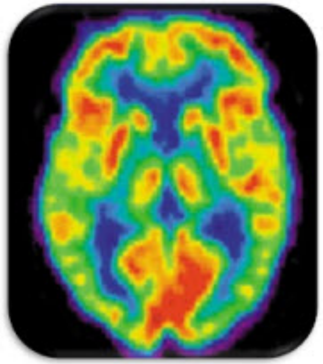
How can a computer analyze?

Music data comes in many forms



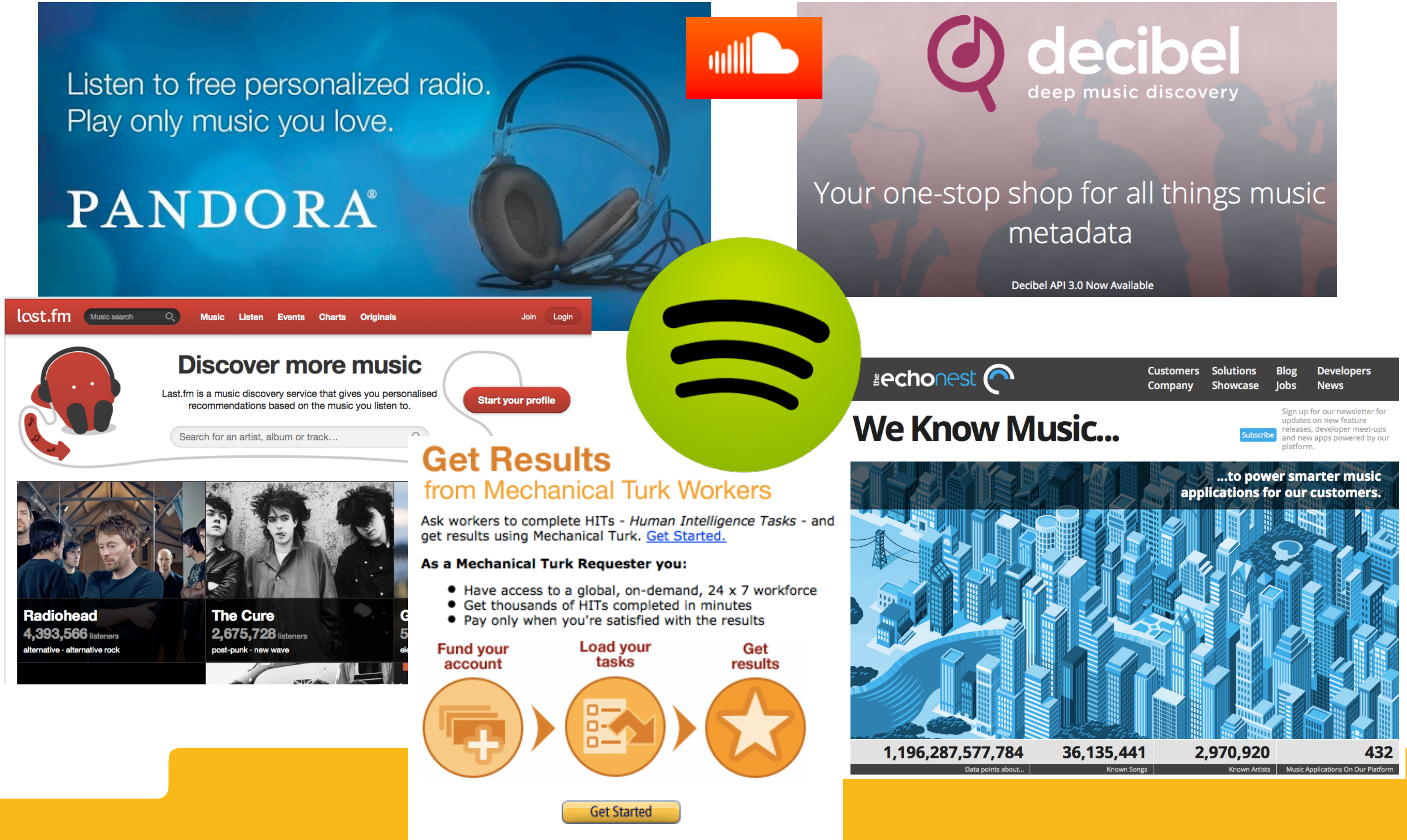






How can a computer analyze?

Music informatics is profitable!



Music informatics is profitable!

Aiva Technologies

AI.MUSIC

melodrive

amper

magenta

How might we search music?

Query by metadata

Google

ABBA Da

abba dan**cing queen**
abba dan**cing queen lyrics**
abba dan**cing queen chords**
abba da**bba jabba**
abba da**bba**
abba dan**cing queen piano sheet music**
abba da**bba chabba**

How might we search music?

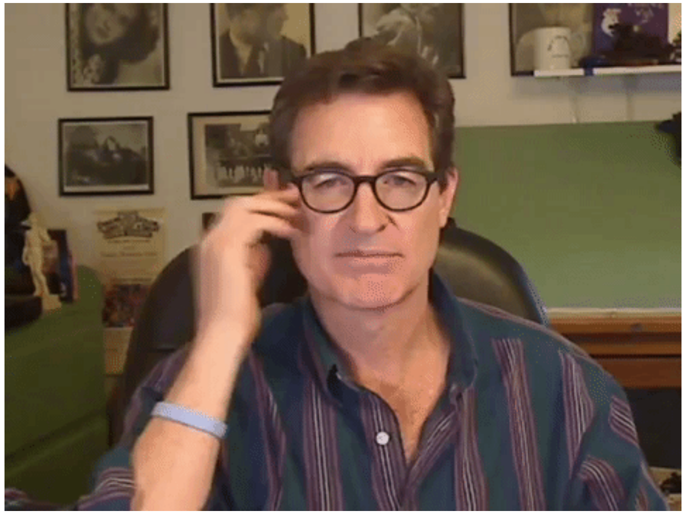
Query by onomatopoeia

Google

Mmmna me me me me mena mena me ma ma ne ne mana

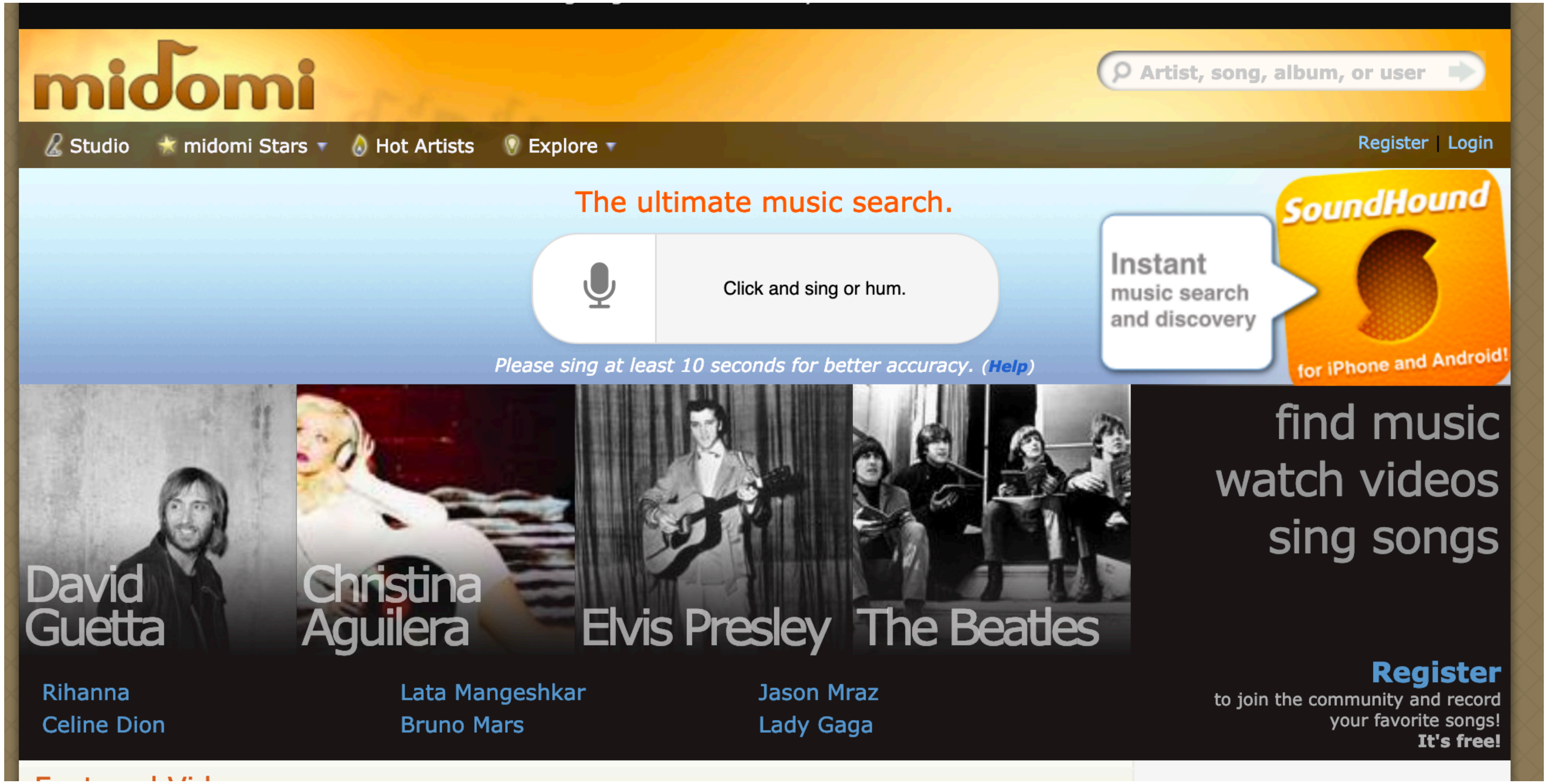
How might we search music?

Query by tapping



How might we search music?

Query by humming or singing



https://www.midomi.com

Demo: Consider this “music document”

How could one find information about what I just played?

- Name
- Notes
- History
- Who else has played it
- …

How might we search music?

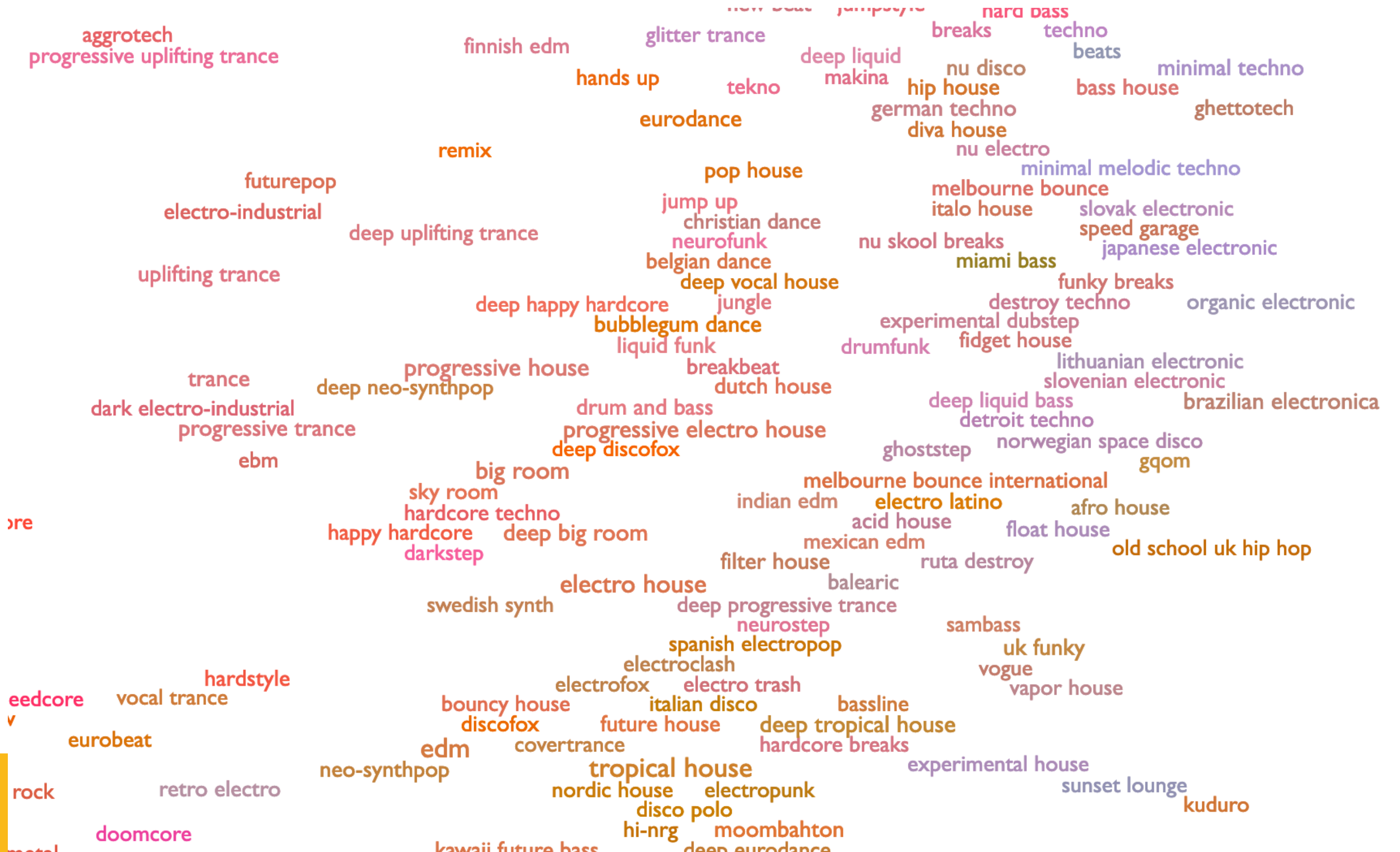
Query by example

How might we search music?

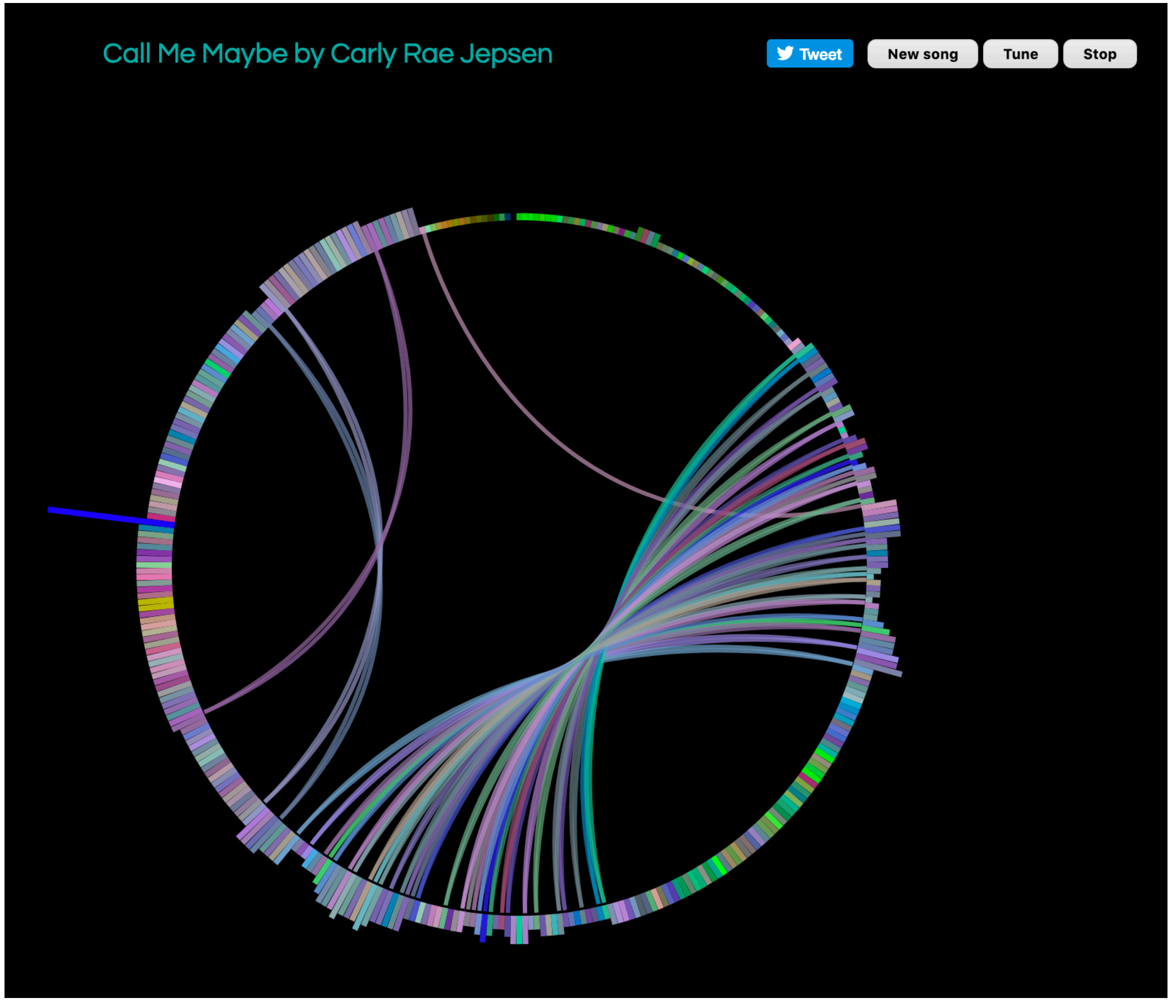
Query by playing

Fun Applications: Every Noise at Once

http://everynoise.com

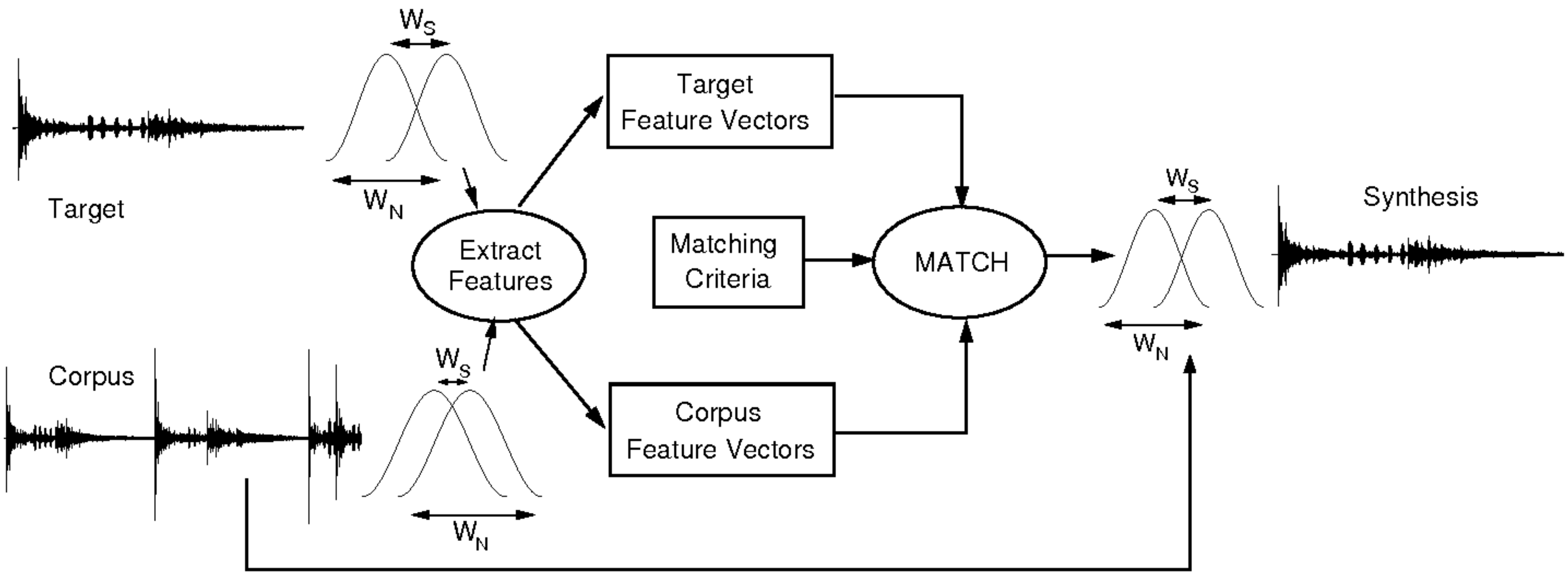


Fun Applications: The Eternal Jukebox



https://eternalbox.dev/jukebox_index.html

Fun Applications: Concatenative music synthesis

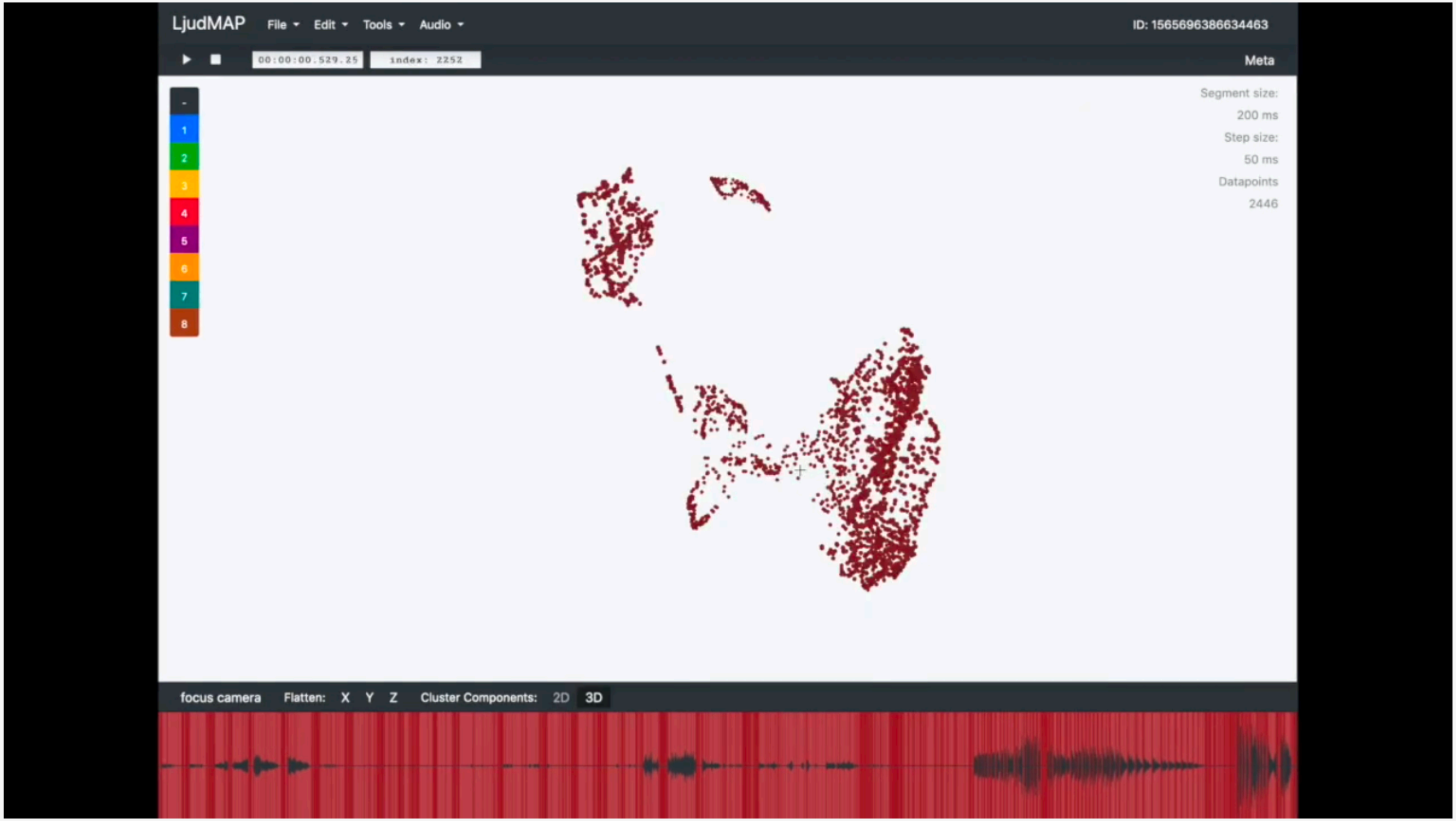


Original

Resynthesized with saxophone

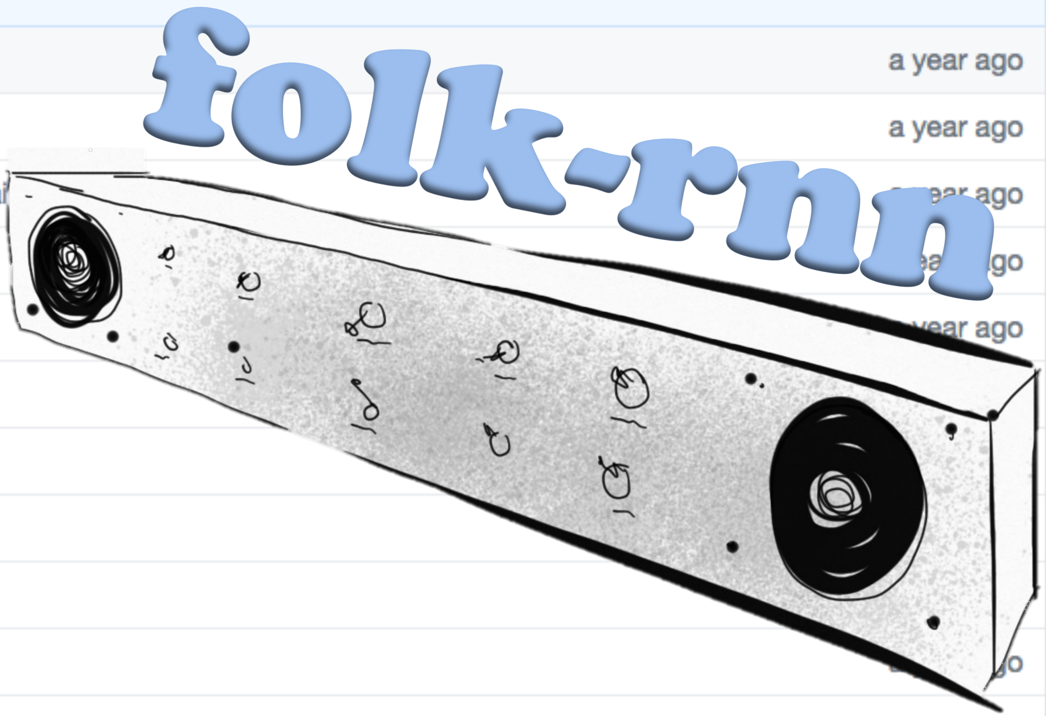
http://spectrum.mat.ucsb.edu/~b.sturm/CMJ2006/MATConcat.html

LjudMAP



https://youtu.be/AqyCXSfKRfs

Fun Applications: Music generation



boblsturm committed on GitHub Update README.md — Latest commit 52a7d37 22 hours ago

configurations	bugfix	a year ago
data	change to readme	a year ago
metadata	Updated metadata with newer trai	ear ago
samples	add metadata and samples	go
soundexamples	Update README.md	year ago
.gitignore	clean before checkout	
LICENSE	license	
README.md	Update README.md	
data_iter.py	added 1hot option	
logger.py	bugfix	o
sample_rnn.py	added terminal output, fixed initialisation	10 months ago
train_rnn.py	bugfix	a year ago

README.md

https://github.com/IraKorshunova/folk-rnn

Folk music style modelling using LSTMs

https://folkrnn.org/

*folk*RNN

generate a folk tune with a recurrent neural network

PRESS TO GENERATE TUNE

Compose

MODEL

thesession.org (w/ :|

TEMPERATURE | SEED

1 | 359615

METER | MODE

4/4 | C Major

INITIAL ABC

Enter start of tune in ABC notation

FOLK RNN TUNE №1580

enter the folkRNN composition competition

```
X:1580
M:4/4
K:Cmaj
cGGE GFEG|(3CCCCD EGEG|cAdc ecdB|ACDE G3A|
(3ccccG EGGC|EDCD (3EEEG2|cGGG e2dc|(3ABcdB c3d:|
|:e3d cege|f3d Bdga|gfeg ecde|fegc A2GF|
EGG2 cdeg|(3fffaf dafd|e2eg fdBc|dedB c3d:|
```

*The RNN properties were **thesession_with_repeats** with seed **441885** and temperature **1**.*

*The prime tokens were **M:4/4 K:Cmaj**.*

*Generated on **14/06/2018, 14:46:35**.*

HEAR IT

0:00 Download MIDI

SEE IT

https://themachinefolksession.org/

THE *machine folk* SESSION

enter the folkRNN composition competition

TUNES RECORDINGS EVENTS

Bob: log out; submit a tune; tune of the month. Help

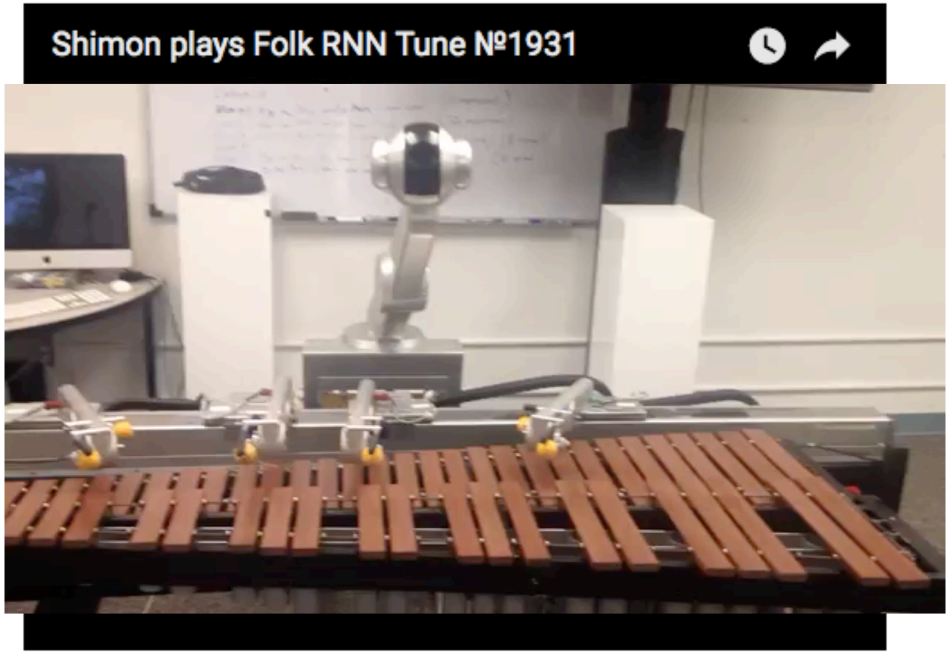
In beta! Launching soon!

Hello Bob

The Machine Folk Session is a community website dedicated to folk music generated by, or co-created with, machines.

You can find tunes to play, recordings of them, or events where they're played.

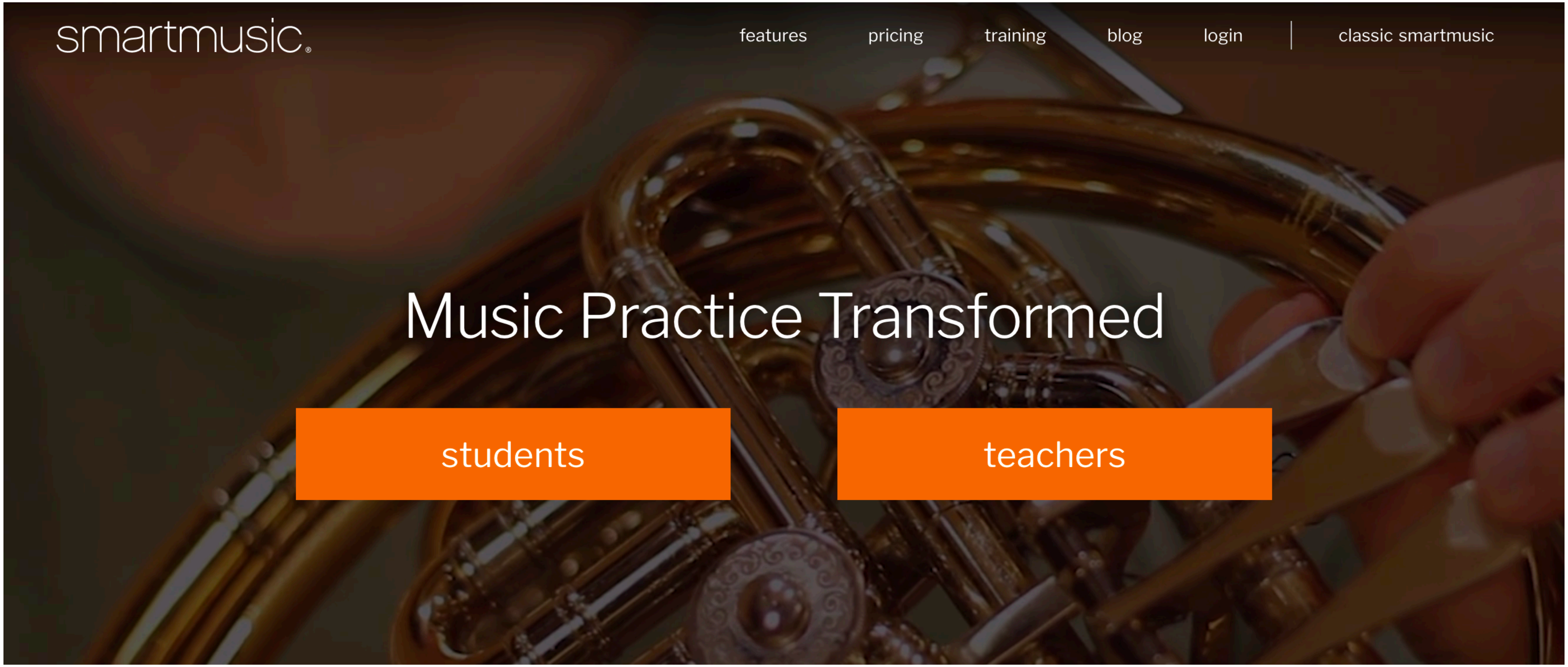
machine folk, live



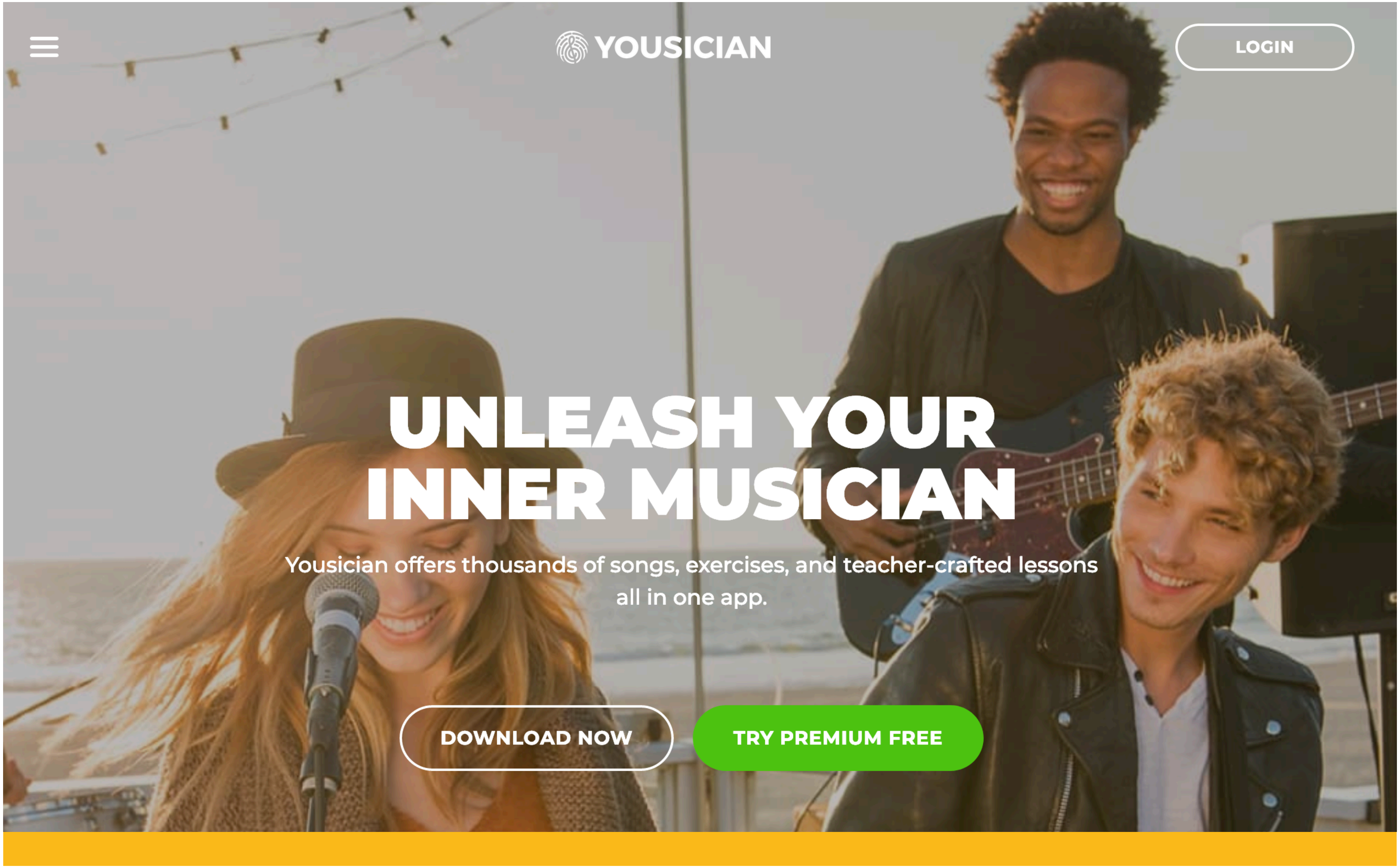
Popular tunes

Folk RNN Tune №1931

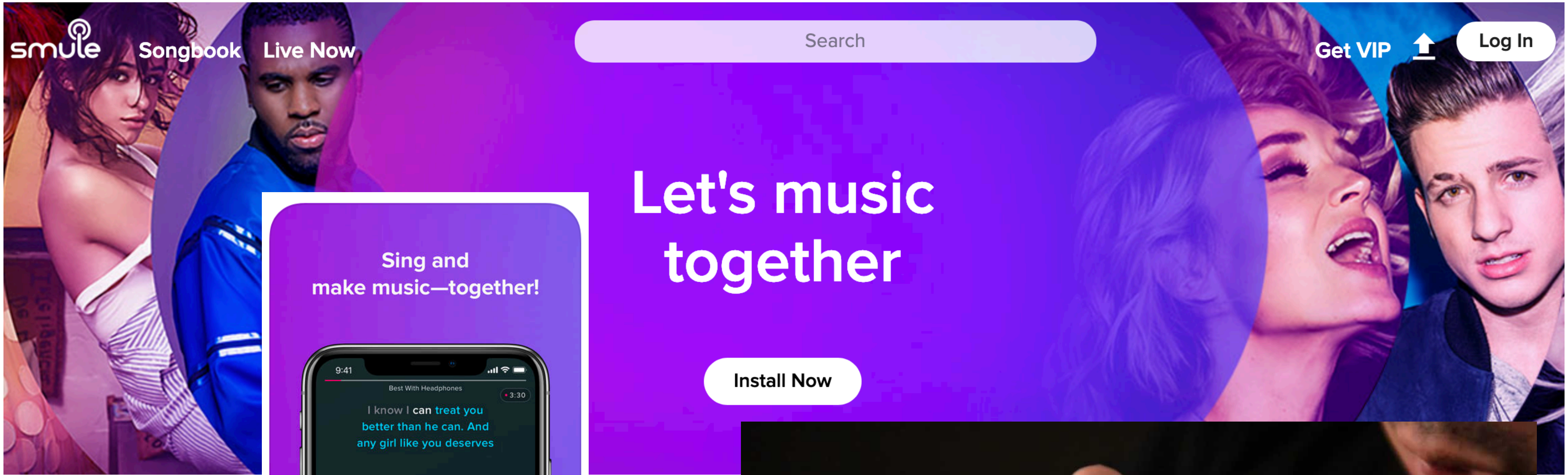
Application: smartmusic

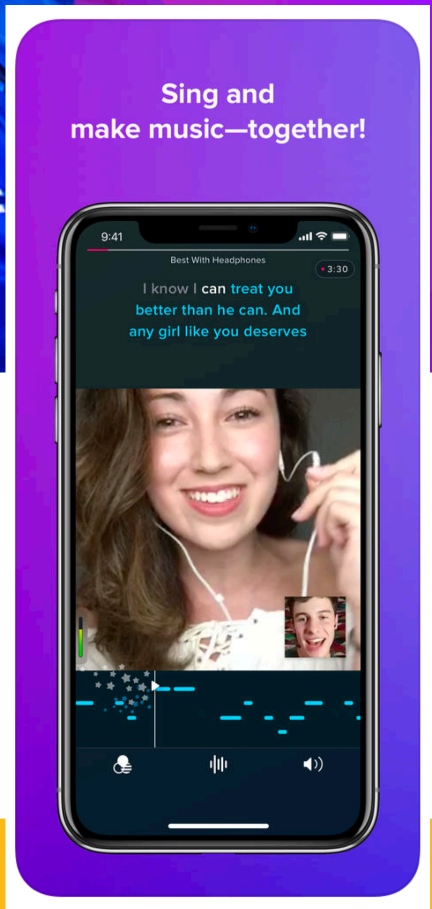


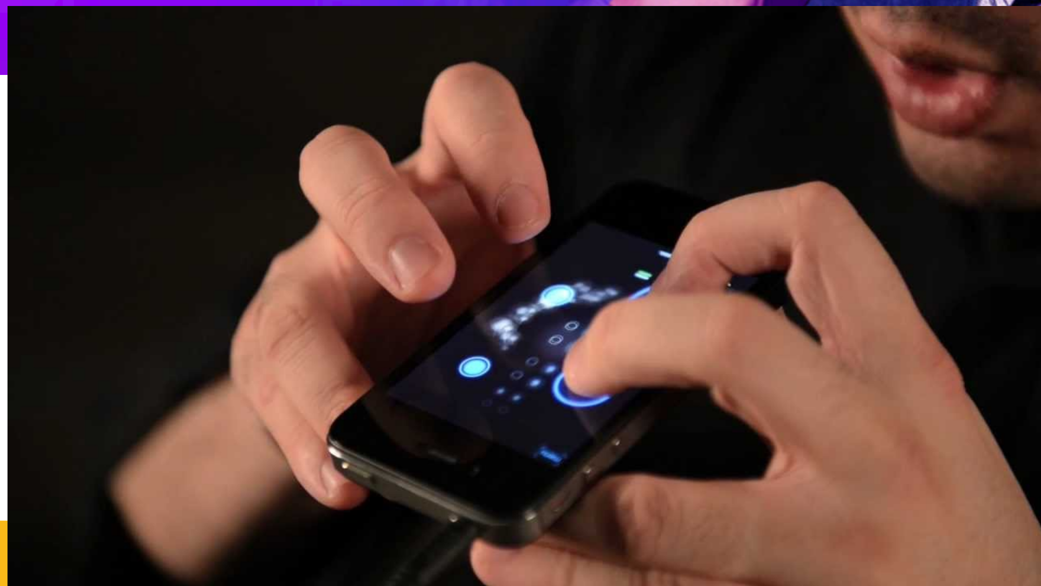
Application: From Finland!



Application: smule



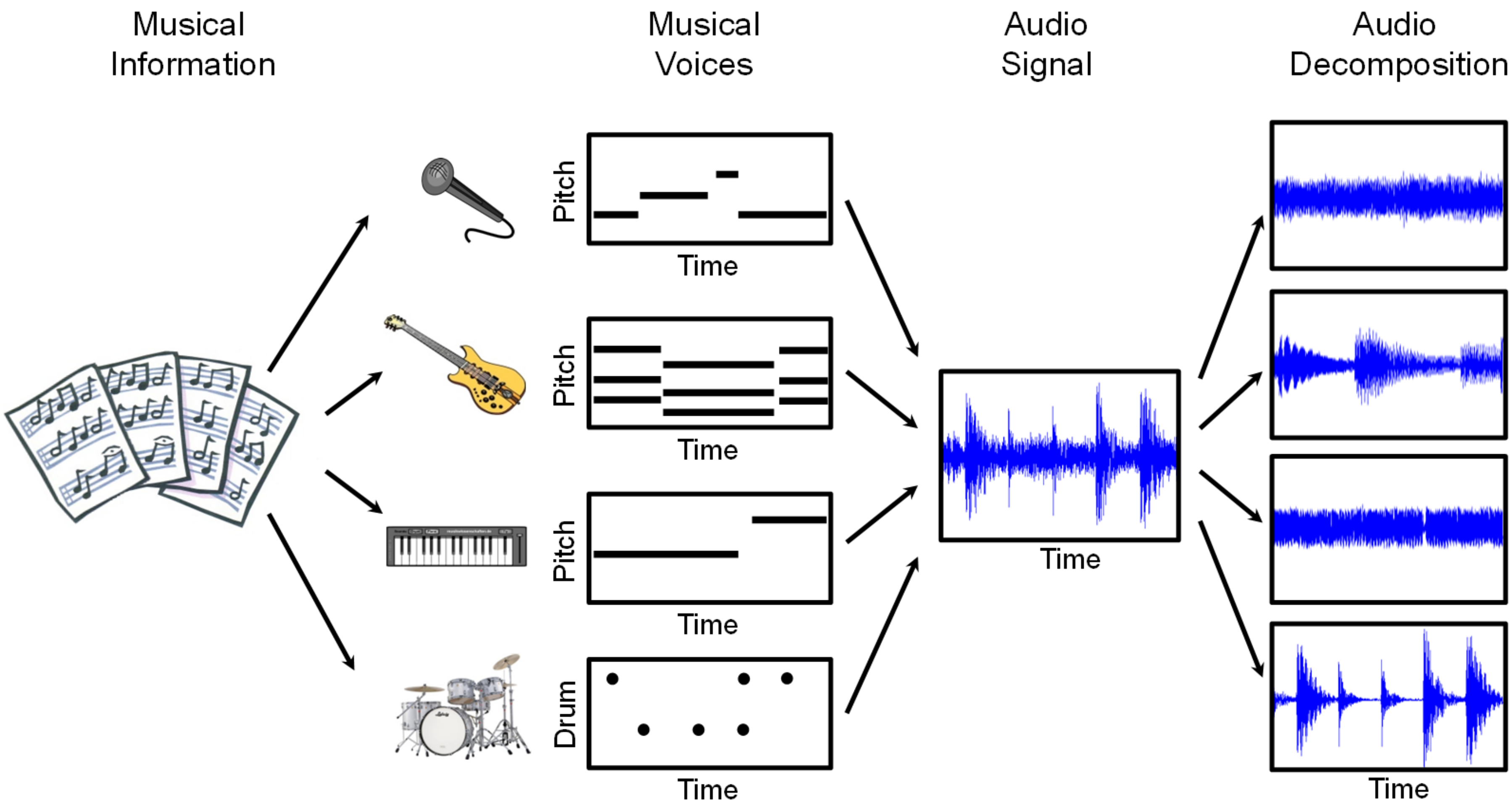




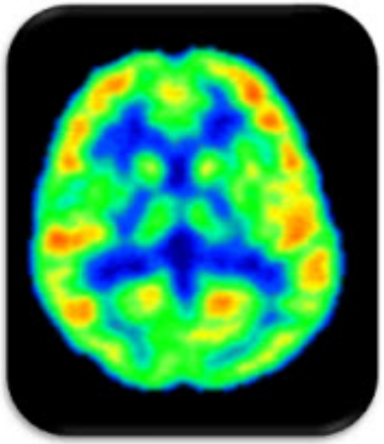
There are some difficult problems!

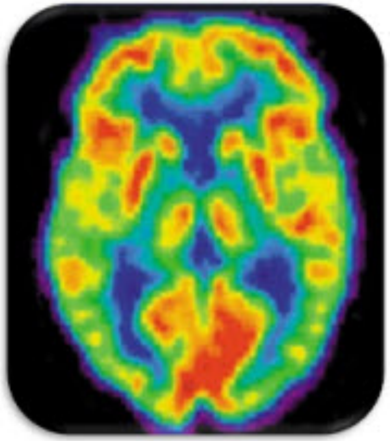
Simultaneous “voices”
Some voices are quite different!

Automated source separation



There are some difficult problems!





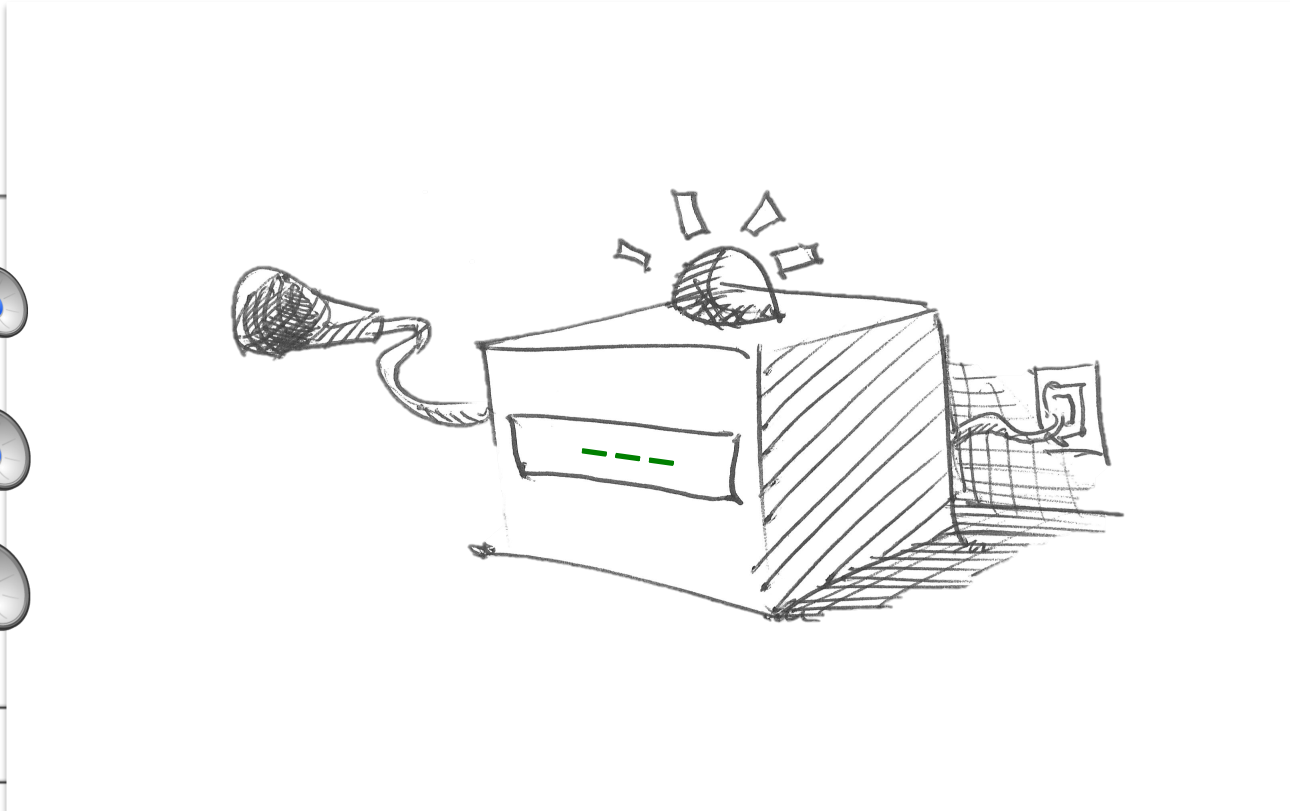
Music happens between the ears

There are some difficult problems!

Apply labels to these to make them discoverable
(t ex style, mood, instrumentation, lyrics, …)

There are some difficult problems!

Label
ChaCha
Jive
Quickstep
Rumba
Samba
Tango
Waltz
Total

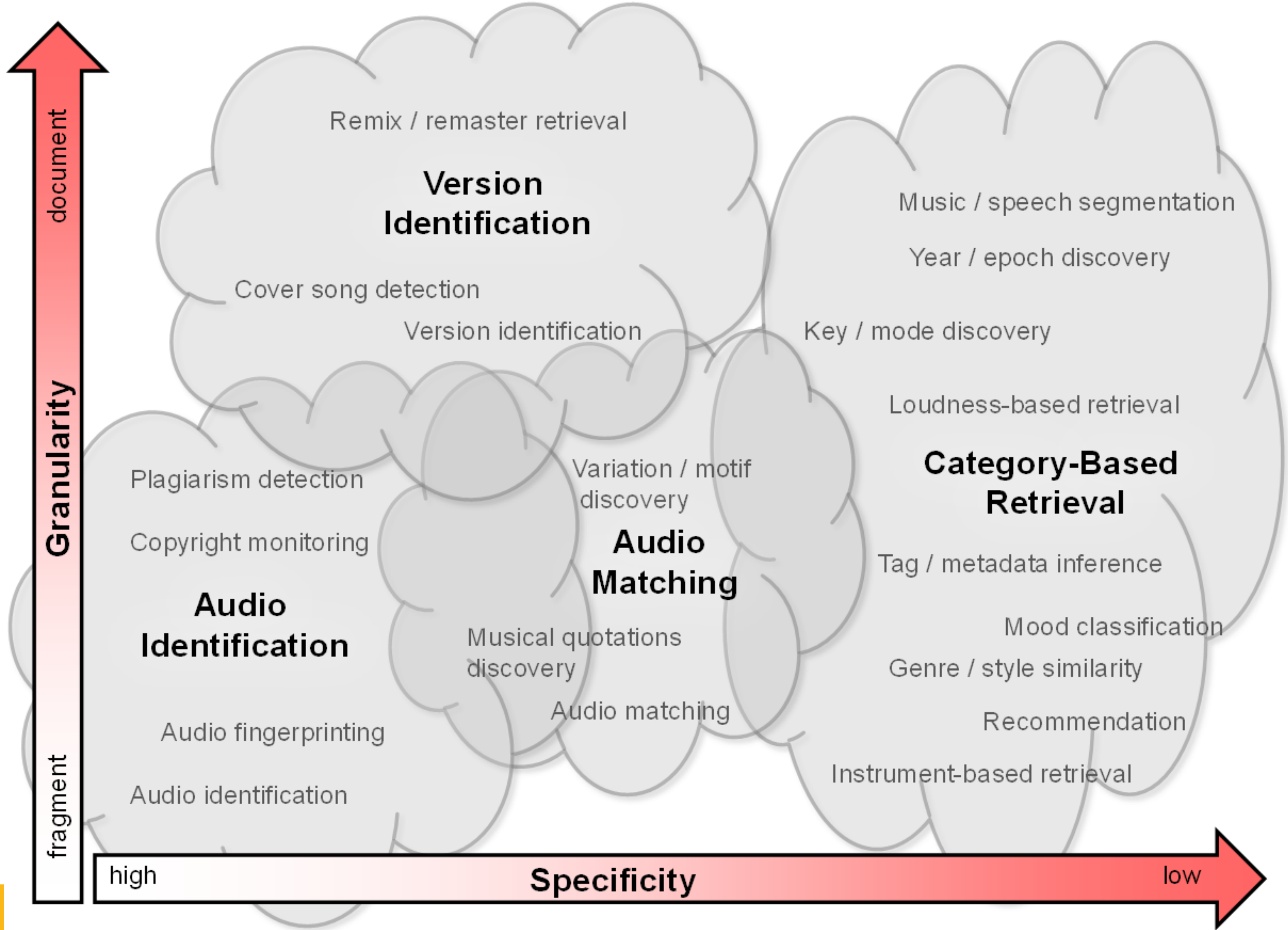


There are some difficult problems!

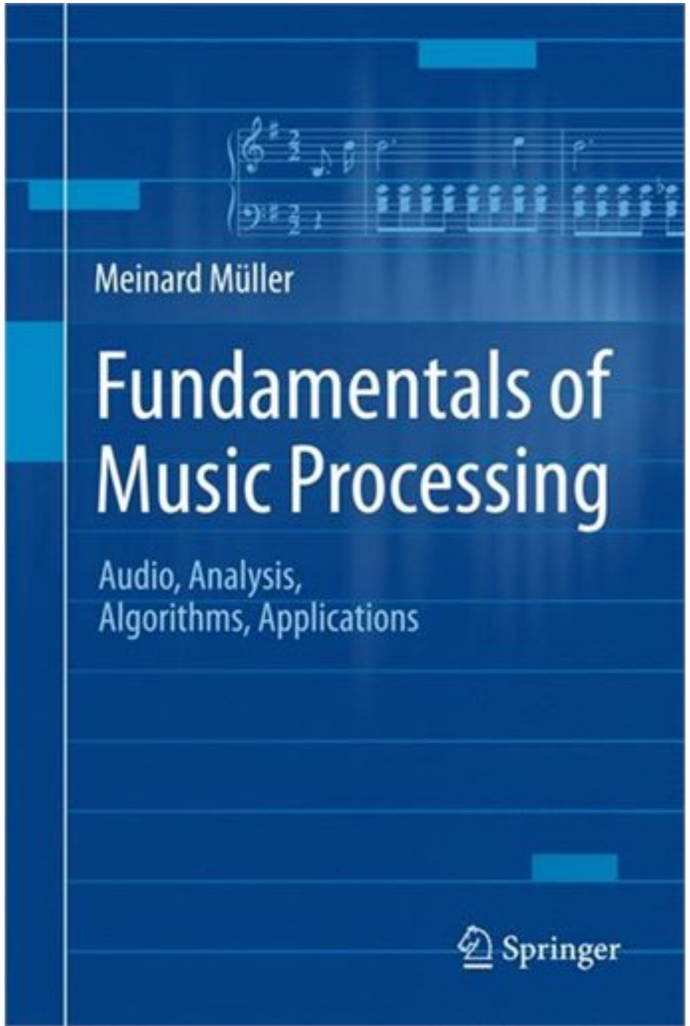
Label
ChaCha
Jive
Quickstep
Rumba
Samba
Tango
Waltz
Total

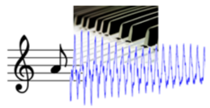
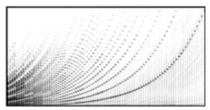
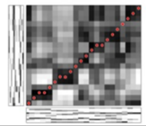
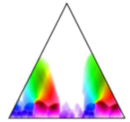
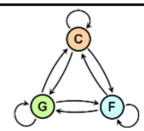
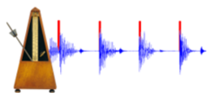
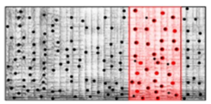
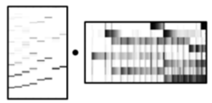
Reference	Highest Acc. Reported (%)
Dixon et al. [2004]	96
Gouyon et al. [2004]	90.1
ISMIR2004	82
Gouyon and Dixon [2004],Gouyon [2005]	82.1
Lidy and Rauber [2005]	84.24
Peeters [2005]	90.4
Flexer et al. [2006]	66.9
Lidy [2006]	82
Lidy et al. [2007]	90.4
Lidy and Rauber [2008]	90.0
Holzapfel and Stylianou [2008]	85.5
Holzapfel and Stylianou [2009]	86.9
Pohle et al. [2009]	89.2
Lidy et al. [2010]	87.97
Mayer et al. [2010]	88
Seyerlehner [2010],Seyerlehner et al. [2010]	~ 90
Peeters [2011]	96.1
Tsunoo et al. [2011]	77.2
Schindler and Rauber [2012]	67.3
Pikrakis [2013]	~ 85
Sturm et al. [2014]	88.8

Music Informatic application levels



DT2470 Required Text



Chapter		Music Processing Scenario	Notions, Techniques & Algorithms
1		**Music Representations**	Music notation, MIDI, audio signal, waveform, pitch, loudness, timbre
2		**Fourier Analysis of Signals**	Discrete/analog signal, sinusoid, exponential, Fourier transform, Fourier representation, DFT, FFT, STFT
3		**Music Synchronization**	Chroma feature, dyamic programming, dyamic time warping (DTW), alignment, user interface
4		**Music Structure Analysis**	Similarity matrix, repetition, thumbnail, homogeneity, novelty, evaluation, precision, recall, F-measure, visualization, scape plot
5		**Chord Recognition**	Harmony, music theory, chords, scales, templates, hidden Markov model (HMM), evaluation
6		**Tempo and Beat Tracking**	Onset, novelty, tempo, tempogram, beat, periodicity, Fourier analysis, autocorrelation
7		**Content-Based Audio Retrieval**	Identification, fingerprint, indexing, inverted list, matching, version, cover song
8		**Musically Informed Audio Decomposition**	Harmonic/percussive component, signal reconstruction, instantaneous frequency, fundamental frequency (F0), trajectory, nonnegative matrix factorization (NMF)

Outline with reference to book

Part I

Aug 24: Intro day!

Aug 27: Music representations (*Chapter 1*)

Aug 31: Signal processing overview (*Chapter 2*)

Sep 3: Machine learning overview (Bishop ch 1)

Sep 3: **Lab 1**: “Signal Processing for Music”

Outline with reference to book

Part II

Sep 7: Machine learning, cont. (Bishop ch 1)

Sep 10: Music structure (*Chapter 4*)

Sep 14: Pitch and chord recognition (*Chapter 5*)

Sep 17: Lab 1 due (before Lab 2 starts)

Sep 17: **Lab 2**: “Machine learning for music”

Outline with reference to book

Part III

Sep 21: Synchronization (*Chapter 3*)

Sep 24: Beat detection and tempo (*Chapter 6*)

Sep 28: Beat detection and tempo (*Chapter 6*)

Oct 1: Lab 2 due (before lab 3 starts)

Oct 1: **Lab 3**: “Beat detection and tempo”

Outline with reference to book

Part IV

Oct 5: Content-based Retrieval (*Chapter 7*)

Oct 8: Content-based Retrieval (*Chapter 7*)

Oct 12: Lab 3 due (before lab 4 starts)

Oct 12: **Lab 4**: “Music identification”

Oct 22: Project preliminary presentations

?: Project report due

?: Lab 4 due

Laboratories: Self-driven

Python based, but you can use whatever

DT2470 Lab 01: Teh Signal Processings for Music

by Bob L. T. Sturm

In this lab you will practice some fundamental concepts of signal processing and their application to music data.

In the first part, you will analyse a chosen sampled sound in the time-, frequency-, and time-frequency domains. You will write something intelligent about your analysis, observing things like periodicity, frequency content, harmonicity, etc.

In any case, the lab report you submit should be a testament to your intelligence, as well as a reflection of your willingness to be a part of this module. You are free to use whatever software you want, e.g., MATLAB, Processing, etc. But below I give tips for python. To be completely transparent, I didn't know much about python before writing this lab. So I learned as I went, making frequent use of the WWW for help. Here's some helpful links I used quite a bit:

- Numpy API
- Scikit-learn API
- MatPlotlib API
- Numpy Cheat Sheet
- Pydub API

I also include some images so you can confirm whether you are on the right track, or just to have a brief pause to laugh at how far your answer is from being correct.

Part 1 Teh Signal Processing

1. Choose an audio file to work with from http://bbcsfx.acropolis.org.uk. Load it and plot the resulting waveform with the appropriate axes labeled "Amplitude" and "Time (s)". The time axis **must be** in seconds. (Use the sample rate of your soundfile to find that.) If your audio file has more than one channel, just look at one channel.

Project information

1. *Build and test your own music informatics algorithm:*

- Classification (genre, mood, instrument)
- Cover song identification
- Beat tracking, tempo estimation
- Key detection
- Music/speech discrimination
- etc

2. *Write a brief summary of algorithm and results (2 pages)*
3. *Use github and include link in report*

See: https://www.music-ir.org/mirex/wiki/MIREX_HOME

Intended Learning Outcomes

After passing the course, the student should be able to:

- explain how music can be represented in reality and in the computer:

E: can describe a few representations (e.g., sampled audio, MIDI, feature vectors, piano roll, ...), and a few applications (e.g., search and retrieval, recommendation, transcription ...)

C: also, can compare and contrast representations and explain applications

A: also, can extend representations and applications

Assessed in Labs and Project Report

Intended Learning Outcomes

After passing the course, the student should be able to:

- account for how feature extraction works and explain why it is needed:

E: can explain windowing and computing statistics of windowed data

C: also, can illustrate procedure

A: also, can choose appropriate parameter settings for a given music informatics application

Assessed in Labs and Project Report

Intended Learning Outcomes

After passing the course, the student should be able to:

- summarise and explain which distinctive features can be extracted from a music signal, based on time, frequency and time-frequency:

E: can name some features in each domain

C: also, can describe how the features are computed

A: also, can choose appropriate features for a given music informatics application

Assessed in Labs and Project Report

Intended Learning Outcomes

After passing the course, the student should be able to:

- use existing software libraries for feature extraction and interpret distinctive features that have been extracted from a music signal:

E: can extract features and display them for a given signal

C: also, can explain and interpret these features

A: also, can integrate these features into a music informatics tool

Assessed in Labs and Project

Intended Learning Outcomes

After passing the course, the student should be able to:

- recommend methods for comparing and modelling of music data:

E: can name some modeling approaches in music informatics

C: also, can explain and illustrate these methods

A: also, can compare and contrast these methods

Assessed in Project Report

Intended Learning Outcomes

After passing the course, the student should be able to:

- design and implement own methods for modelling of music data:

E: can describe a modeling approach as an algorithm

C: also, can implement a method in computer code

A: also, can explain how such methods can be evaluated

Assessed in Labs and Project

Intended Learning Outcomes

After passing the course, the student should be able to:

- describe how information on different abstraction levels can be extracted from music data (acoustic as well as symbolic) and be used in many applications (e.g., search, retrieval, synthesis):

E: can describe the music informatics pipeline

C: also, can identify where limitations are encountered

A: also, can choose appropriate evaluation methodologies

Assessed in Labs and Project Report

Intended Learning Outcomes

After passing the course, the student should be able to:

- design algorithms for handling and modelling of music data as well as evaluate their performance:

E: use existing software libraries to extract a few basic features, model them, and evaluate them using basic approaches

C: also use existing software libraries to extract a variety of relevant features, model them with relevant methods, and evaluate them using many approaches

A: also improve the performance of developed models

Assessed in Labs and Project

Intended Learning Outcomes

After passing the course, the student should be able to:

- be able to appreciate the latest technology in music informatics and build on it:

E: name a few companies that use music informatics

C: also explain what aspects of music informatics might be employed at said companies

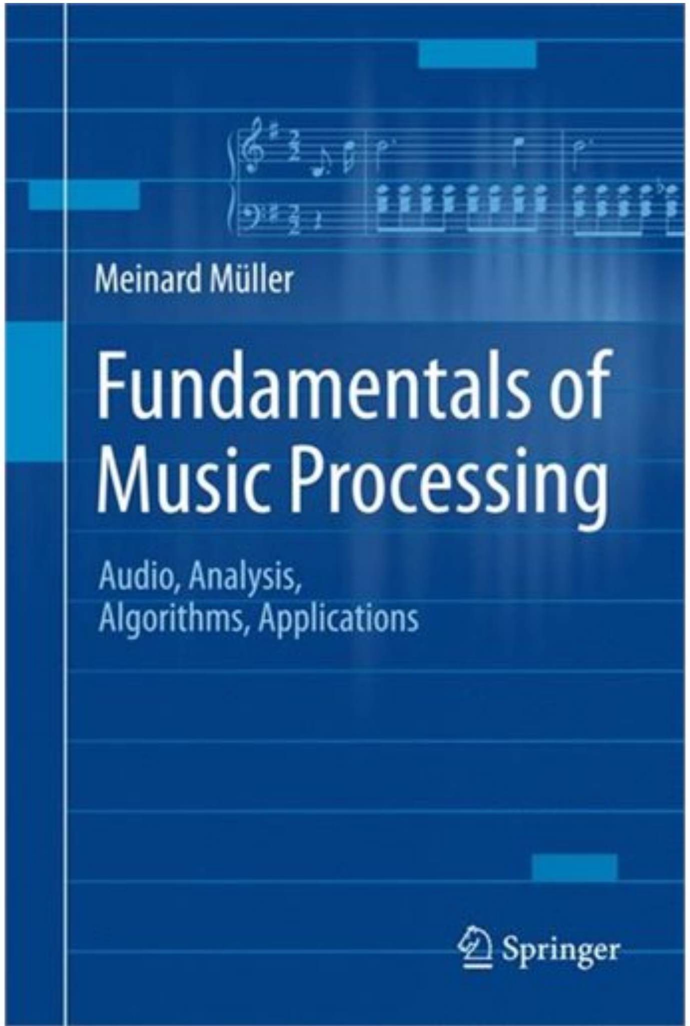
A: also elaborate upon novel applications of music informatics

Assessed in Project Report

Grading

1. Laboratory work, 3.0 credits, Grading scale: P, F
2. Project (implementation and evaluation), 3.0 credits, Grading scale: A–F
3. Written report and presentation, 1.5 credits, Grading scale: A–F

For Thursday



Read Chapter 1!

Install:

- Python environment and IDE, e.g., Anaconda and Spyder
- Sonic Visualiser https://www.sonicvisualiser.org/ and VST plugins