Statistical musicology with an application in SVM based instrument classification

Uwe Ligges and Sebastian Krey

Department of Statistics, TU Dortmund

Vienna, June 2009

Introduction

We have worked on several (Music) Signal Analysis tasks in the past, among them:

- ‘classification and clustering of vocal performances’ (Weihs et al., 2003)
    - objective criteria for the assessment of the quality of vocal performance
    - **timbre differences** of voices and instruments
    - properties related to performance quality aspects of single tones like solidity / softness / brilliance of tones

Introduction

- ‘Vowel Classification by a Perceptually Motivated Neurophysiologically Parameterized Auditory Model’ (Szepannek et al., 2006)
  - **speech recognition**
  - perception analysis (e.g. for audio compression, **hearing aids** etc.)
- ‘Detection of Chattering and Spiralling and BTA Deep Hole Drilling’
  - project in collaborative research center SFB 475
  - huge multivariate time series with high sampling rate
  - prediction of chattering / spiralling
  - **controlling the process**

Introduction

- **automatic transcription of music**
  - of interest for music publishers, music amateurs, and scientists (particularly those working in music psychology)
  - part of transcription algorithms heavily used in music recommender systems
- **classification of instruments (timbre analysis)**
  - useful as a tool for music transcription tasks
  - useful for singing teachers and students who try to improve voices
  - useful to identify if audio compression (like in hearing aids) works sufficiently well

Simplify further research of audio or vibration time series: Need for a toolset which allows easy access to (at least) the standard methods.

Software

It is inconvenient to switch frequently between different software products such as wave editors, spectral analysis software, statistical programming languages, which means exporting / importing the data again and again ...

Software

It is inconvenient to switch frequently between different software products such as wave editors, spectral analysis software, statistical programming languages, which means exporting / importing the data again and again ...

R (R Development Core Team, 2009) is the statistical programming language of our choice, hence the development of an R package is our preferred solution:

Software

It is inconvenient to switch frequently between different software products such as wave editors, spectral analysis software, statistical programming languages, which means exporting / importing the data again and again ...

R (R Development Core Team, 2009) is the statistical programming language of our choice, hence the development of an R package is our preferred solution:

tuneR

Talk about both, software and ideas (methods?) to solve our tasks.

Handling Waves

An object of class *Wave* representing the data in a typical Wave file (e.g. sound from a CD) is fundamental for further analyses.

Information contained in a *Wave* object are:

- data in left / right channel
- sampling rate
- resolution (bit)
- stereo / mono

Handling Waves

Required functions to handle Waves are (among others):

- Access: readWave(), writeWave()
- Get info: show(), summary()
- Visualize: plot()
- Listen: play()
- Extract: channel(), mono(), subset(), extract()
- Merge: stereo(), bind()
- Change: downsample(), bit()

Example

- Along the example of 'automatic transcription of singing performances', we will present which additional **functions** and what kind of **classes** are required for the convenient analysis of audio time series.
- Let's play around with the Wave of a performance of the song 'Tochter Zion' by Händel sung by a professional bass (cp. Weihs et al., 2001).

Steps of Transcription

- **Separation** of the singing voice from any other sound.
- **Segmentation** of sound into segments presumably corresponding to notes, silence, or noise, as well as **pitch estimation** and **classification** of the corresponding note (Ligges et al., 2002; Ligges, 2006).
- **Quantization** is the derivation of relative lengths of notes (quavers, quarter notes, etc.) from estimated absolute lengths.
- **Final transcription** into music notation, sheet music.

Details on the segmentation step

- **Passing through** the vocal time series by sections of given size.
- **Pitch estimation** for each section.
- **Note classification** using estimated fundamental frequencies
- **Smoothing** of classified notes because of vibrato (what we really need is a *good* model for vibrato).
- **Segmentation**, if a change in the smoothed list of notes occurs.

Pitch estimation

Several methods for pitch estimation (f_0 tracking, ...) have been proposed:

- in time domain (such as a model that follows shortly)
- in frequency domain (such as our heuristical proposal)
- hybrid methods
- any combinations with, e.g., HMMs

- none of them works really well on singing data
- none of them works on polyphonic data

Pitch estimation model (monophonic)

$$y_t \quad = \qquad \cos\left[2\pi t f_0 \qquad + \phi \ \right] + \epsilon_t$$

- f_0 = **fundamental frequency**, the parameter of interest
- ϵ_t = error
- $t \in \left\{\frac{0}{S}, \frac{1}{S}, \ldots, \frac{T-1}{S}\right\}$ time, no. of observations T
- ϕ = phase displacement

Pitch estimation model (monophonic)

$$y_t = \sum_{h=1}^{H} \cos\left[2\pi t f_0 (h \qquad) + \phi_h\right] + \epsilon_t$$

- f_0 = **fundamental frequency**, the parameter of interest
- ϵ_t = error
- $t \in \left\{\frac{0}{S}, \frac{1}{S}, \ldots, \frac{T-1}{S}\right\}$ time, no. of observations T
- ϕ_h = phase displacement of h-th partial
- H = no. of partials in the model

Pitch estimation model (monophonic)

$$y_t = \sum_{h=1}^{H} B_h \cos\left[2\pi t f_0 (h \qquad) + \phi_h\right] + \epsilon_t$$

- f_0 = **fundamental frequency**, the parameter of interest
- ϵ_t = error
- $t \in \left\{\frac{0}{S}, \frac{1}{S}, \ldots, \frac{T-1}{S}\right\}$ time, no. of observations T
- ϕ_h = phase displacement of h-th partial
- H = no. of partials in the model
- B_h = amplitude of h-th partial

Pitch estimation model (monophonic)

$$y_t = \sum_{h=1}^{H} B_h \cos\left[2\pi t f_0 (h + \delta_h) + \phi_h\right] + \epsilon_t$$

- f_0 = **fundamental frequency**, the parameter of interest
- ϵ_t = error
- $t \in \left\{\frac{0}{S}, \frac{1}{S}, \ldots, \frac{T-1}{S}\right\}$ time, no. of observations T
- ϕ_h = phase displacement of h-th partial
- H = no. of partials in the model
- B_h = amplitude of h-th partial
- δ_h = frequency displacement of h-th partial where $\delta_1 := 0$

Pitch estimation model (monophonic)

$$y_t = \sum_{h=1}^{H} \sum_{i=0}^{I} \Phi_i(t) B_{h,i} \cos\left[2\pi t f_0 (h + \delta_h) + \phi_h\right] + \epsilon_t$$

- $B_{h,i}$ = amplitude of h-th partial for i-th basis function
- i = index of $I + 1$ basis functions
- $\Phi_i(t) := \cos^2\left[\pi \frac{tS - i\Delta}{2\Delta}\right] \mathbf{1}_{[(i-1)\Delta,(i+1)\Delta]}(t)$ i-th basis function defined on windows with 50% overlap, $\Delta := \frac{T-1}{I}$, $\mathbf{1}$ indicator function, S sampling rate

Pitch estimation model (monophonic)

$$y_t = \sum_{h=1}^{H} \sum_{i=0}^{I} \Phi_i(t) B_{h,i} \cos\left[2\pi t f_0 (h + \delta_h) + \phi_h + (h + \delta_h) A_v \sin(2\pi f_v t + \phi_v)\right] + \epsilon_t$$

- $B_{h,i}$ = amplitude of h-th partial for i-th basis function
- i = index of $I+1$ basis functions
- $\Phi_i(t) := \cos^2\left[\pi \frac{tS - i\Delta}{2\Delta}\right] \mathbf{1}_{[(i-1)\Delta,(i+1)\Delta]}(t)$ i-th basis function defined on windows with 50% overlap, $\Delta := \frac{T-1}{I}$, $\mathbf{1}$ indicator function, S sampling rate
- f_v = frequency of vibrato
- A_v = amplitude of vibrato
- ϕ_v = phase displacement of vibrato

Pitch estimation model (monophonic)

$$y_t = \sum_{h=1}^{H} \sum_{i=0}^{I} \Phi_i(t) B_{h,i} \cos\left[2\pi t f_0 (h + \delta_h) + \phi_h + (h + \delta_h) A_v \sin(2\pi f_v t + \phi_v)\right] + \epsilon_t$$

- $B_{h,i}$ = amplitude of h-th partial for i-th basis function
- i = index of $I + 1$ basis functions
- $\Phi_i(t) := \cos^2\left[\pi \frac{tS - i\Delta}{2\Delta}\right] \mathbf{1}_{[(i-1)\Delta,(i+1)\Delta]}(t)$ i-th basis function defined on windows with 50% overlap, $\Delta := \frac{T-1}{I}$, $\mathbf{1}$ indicator function, S sampling rate
- f_v = frequency of vibrato
- A_v = amplitude of vibrato
- ϕ_v = phase displacement of vibrato
- $5 + 3H$ **parameters to estimate**, but H might be > 10

Pitch estimation model (POLYphonic)

$$
\begin{aligned}
y_t \; = \sum_{j=1}^{J}\sum_{h=1}^{H}\sum_{i=0}^{I} \; & \Phi_{i,j}(t) B_{h,i,j} \cos\left[2\pi t f_{0,j}(h_j + \delta_{h,j}) + \phi_{h,j} \right. \\
& \left. + (h_j + \delta_{h,j}) A_{v,j} \sin(2\pi f_{v,j} t + \phi_{v,j})\right] + \epsilon_t
\end{aligned}
$$

- Joint work in progress (?) with Katrin Sommer, Claus Weihs; cooperation with Technical University of Tampere.
- J number of polyphonic tones
- **Identifiability ?!**

The Timbre Problem

Timbre Classification

- Joint work with Sebastian Krey
- Specific task: Classification of instruments based on a given audio track of one tone
- Data: McGill Instrument Database, 38 instruments played in 59 ways (e.g. bowed vs. pizz.), each with 6-88 differently pitched tones, altogether 1976 wave files (44100 Hertz, 16 bit, 3-5 seconds each)

Let's start

- Pre-emphasis filtering to increase higher partials:

$$y_t = x_t - 0.97 x_{t-1}$$

- Short Time Fourier Transformation (on overlapping windows):

$$F(t,k) = \sum_{j=1-M}^{N-M} w(j-t) x_j \exp\left(-2i\pi j \frac{k}{N}\right)$$

- Hamming windows (width: 25ms, overlap: 10ms):

$$w(t) = \begin{cases} 0.54 - 0.46\cos\left(\frac{2\pi t}{T}\right), & -\frac{T}{2} \leq t \leq \frac{T}{2} \\ 0 & \text{otherwise} \end{cases}$$

Let's start

- Mel scale:
  Transformation of FFT frequencies to Mel scale in order to model the emotional sense of the human ear (better resolution of human ear above 1 kHz, for example):

$$Mel(hz) = 2595 \log_{10}\left(1 + \frac{hz}{700}\right)$$

Feature Extraction

Using features pretty well known from speech recognition, e.g.:

- (Perceptive) Linear Predictive Coding (LPC/PLP)
  - Filter even more in order to get a somehow uniform loudness impression on the whole frequency range (PLP)
  - Loudness compression by looking at cubic roots of amplitudes (PLP)
  - Transformation back to time domain by inverse Fourier transformation
  - Fit an autoregressive model (by Levinson Durbin recursion):

$$y_t = \sum_{j=1}^{p} a_j y_{t-j} + e_t$$

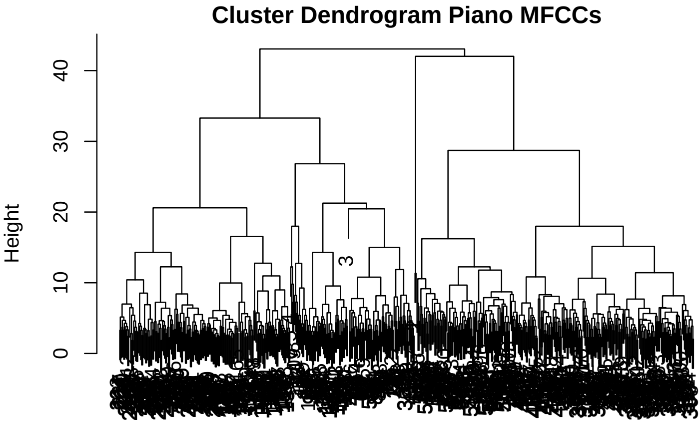
Feature Extraction

- Mel Frequency Cepstral Coefficients (MFCC)
    - Logarithm of loudness compression
    - Discrete cosine transformation (DCT)
    - Considering first p DCT coefficients

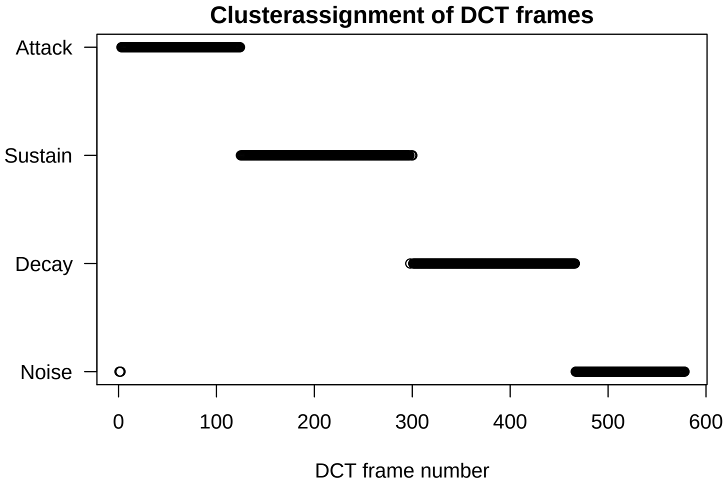
Clustering

- Different tones have different lengths, i.e. different numbers of windows are used
- Additionally, there might be silence (or noise such as breathing) at the start / end of a tone
- Hence clustering the found (vectors of) coefficients of all windows using Kmeans
- Number of clusters: 3-4, motivated by different phases of a tone: attack, (sustain), decay, silence/noise.

Hierarchical Clustering



Kmeans Clustering



Classification

- Support Vector Machines, tried kernels:

$$\begin{aligned} \text{linear} \quad & K(x_i, x_j) = x_i' x_j \\ \text{polynomial} \quad & K(x_i, x_j) = (\gamma x_i' x_j + r)^d,\ \gamma > 0 \\ \text{rbf} \quad & K(x_i, x_j) = \exp(-\gamma \|x_i - x_j\|^2) \\ \text{sigmoid} \quad & K(x_i, x_j) = \tanh(\gamma x_i' x_j + r) \text{ (extremely bad)} \end{aligned}$$

- Linear Discriminant Analysis
- Random Forests
- some more not reported

Software

- Port of functions from Matlab package `rastamat` to R
- Some more speech processing functions implemented to be published in package `tuneR`
- SVM implementation from R package `kernlab` and (the not yet published) `classifieR` for optimization and validation of classification results

Results

- all results based on a doubled 5-fold crossvalidation (inner loop for parameter optimization, outer loop for assessment)
- 59 classes
- LPC coefficients: All misclassification rates $> 85\%$.
- PLP coefficients:

classif.	parameter	error	std error
SVM-Poly	$\gamma = 1.4, \ d = 3$	0.33	0.03
SVM-RBF	$\gamma = 1.4, \ \sigma = 0.023$	0.44	0.03
SVM-Lin	$\gamma = 1.5$	0.51	0.03
RandFor	$U = 1500, \ V = 3$	0.32	0.03
LDA		0.55	0.02

Results

- MFCC + PLP

classif.	parameter	error	std error
SVM-Poly	$\gamma = 1.4,\ d = 2$	0.18	0.02
SVM-RBF	$\gamma = 1.5,\ \sigma = 0.007$	0.23	0.02
SVM-Lin	$\gamma = 0.6$	0.18	0.03
RandFor	$U = 1000,\ V = 6$	0.22	0.03
LDA		0.28	0.02

Summary

- We are working on 59 classes, i.e. guessing implies misclassification error of 0.98
- Best misclassification rate: 0.18
  (comparable to what trained humans can archive)
- It turns out that the choice and construction of appropriate variables is (as in so many other classification tasks) much more important than the particular classification method that is finally used.

References I

- von Ameln, F. (2001): *Blind source separation in der Praxis*. Diploma Thesis, Fachbereich Statistik, Universität Dortmund, Germany.
- Cano, P., Loscos, A., Bonada, J. (1999): Score-Performance Matching using HMMs. In: *Proceedings of the International Computer Music Conference*. Beijing, China.
- Cemgil, T., Desain, P., Kappen, B. (2000): Rhythm Quantization for Transcription. *Computer Music Journal* 24 (2), 60–76.
- Ellis, D. P. W. (2005), *PLP and RASTA (and MFCC, and inversion) in Matlab*, URL `http://www.ee.columbia.edu/ dpwe/resources/matlab/rastamat/`
- Garczarek, U., Weihs, C., Ligges, U. (2003): Prediction of Notes from Vocal Time Series. *Technical Report 1/2003*, SFB475, Department of Statistics, University of Dortmund. `http://www.sfb475.uni-dortmund.de`.
- Hastie, T. & Tibshirani, R. & Friedman, J. (2001), *The Elements of Statistical Learning*, Springer, New York.
- Hsu, C.-W. & Chang, C.-C. & Lin, C.-J. (2008) *A Practical Guide to Support Vector Classification* National Taiwan University, Taipei, URL `http://www.csie.ntu.edu.tw/ cjlin`

References II

- Hyvärinen, A., Karhunen, J., Oja, E. (2001): *Independent Component Analysis*. Wiley, New York.
- Karatzoglou, A. & Smola, A. & Hornik, K. & Zeileis, A. (2004) *kernlab – An S4 Package for Kernel Methods in R* Journal of Statistical Software Vol. 11, No. 9, pages 1–20 URL `http://www.jstatsoft.org/v11/i09/`
- Kleber, B. (2002): *Evaluation von Stimmqualität in westlichem, klassischen Gesang*. Diploma Thesis, Fachbereich Psychologie, Universität Konstanz.
- Ligges, U. (2006): *Transkription monophoner Gesangszeitreihen*. Dissertation, Fachbereich Statistik, Universität Dortmund, `http://hdl.handle.net/2003/22521`.
- Ligges, U., Weihs, C., Hasse-Becker, P. (2002): Detection of Locally Stationary Segments in Time Series. In: W. Härdle And B. Rönz (Eds.): *CompStat2002 – Proceedings in Computational Statistics – 15th Symposium held in Berlin, Germany*. Physika Verlag, Heidelberg, 285–290.
- Nienhuys, H.-W., Nieuwenhuizen, J., et al. (2004): *GNU LilyPond – The Music Typesetter*. Free Software Foundation, `http://www.lilypond.org`, Version 2.0.3.
- Opolko, F. & Wapnick, J. (1987) *McGill University master samples (CDs)*

References III

- Preusser, A., Ligges, U., Weihs, C. (2002): Ein R Exportfilter für das Notations- und Midi-Programm LilyPond. *Arbeitsbericht* 35, Fachbereich Statistik, Universität Dortmund, Germany. `http://www.statistik.uni-dortmund.de`.
- R Development Core Team (2008): *R: A language and environment for statistical computing*. R Foundation for Statistical Computing, Vienna, ISBN 3-900051-00-3, `http://www.R-project.org`.
- Reuter, C. (2002): *Klangfarbe und Instrumentation – Geschichte – Ursachen – Wirkung*. Peter Lang, Frankfurt/M.
- Roever, C. (2003), *Musikinstrumentenerkennung mit Hilfe der Hough-Transformation*, URL `http://www.aei.mpg.de/ chroev/publications/RoeverDiplom.pdf`
- Rosell, M. (2006) *An Introduction to Front-End Processing and Acoustic Features for Automatic Speech Recognition* URL `www.nada.kth.se/ rosell/courses/rosell_acoustic_features.pdf`
- Weihs, C., Berghoff, S., Hasse-Becker, P., Ligges, U. (2001): Assessment of Purity of Intonation in Singing Presentations by Discriminant Analysis. In: J. Kunert And G. Trenkler (Eds.): *Mathematical Statistics and Biometrical Applications.* Josef Eul, Bergisch-Gladbach, Köln, 395–410.

References IV

- Weihs, C., Ligges, U. (2003): Automatic Transcription of Singing Performances. *Bulletin of the International Statistical Institute*, 54th Session, Proceedings, Volume LX, Book 2, 507–510.
- Weihs, C., Ligges, U., Güttner, J., Hasse-Becker, P., Berghoff, S. (2003): Classification and Clustering of Vocal Performances. In: M. Schader, W. Gaul and M. Vichi (Eds.): *Between Data Science and Applied Data Analysis.* Springer, Berlin, 118–127. (in print)