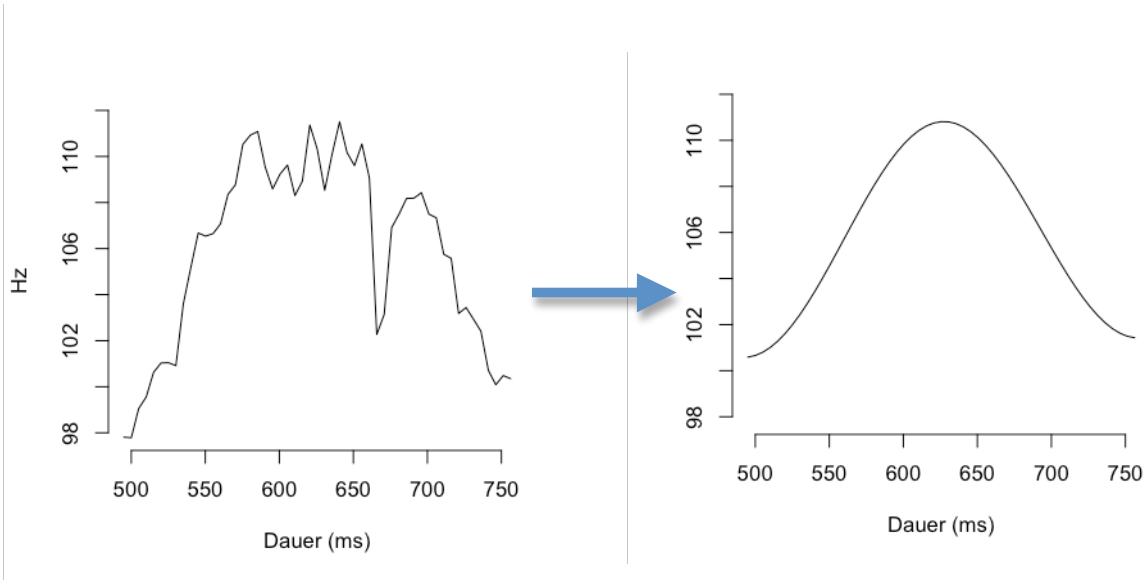
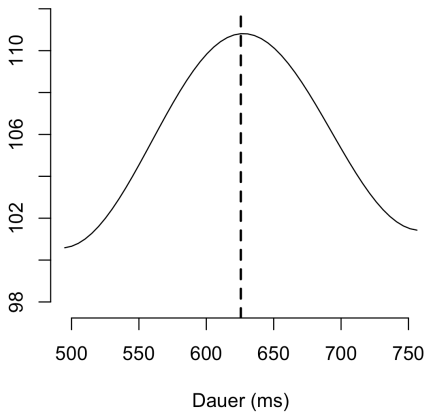
Datenbank lvn

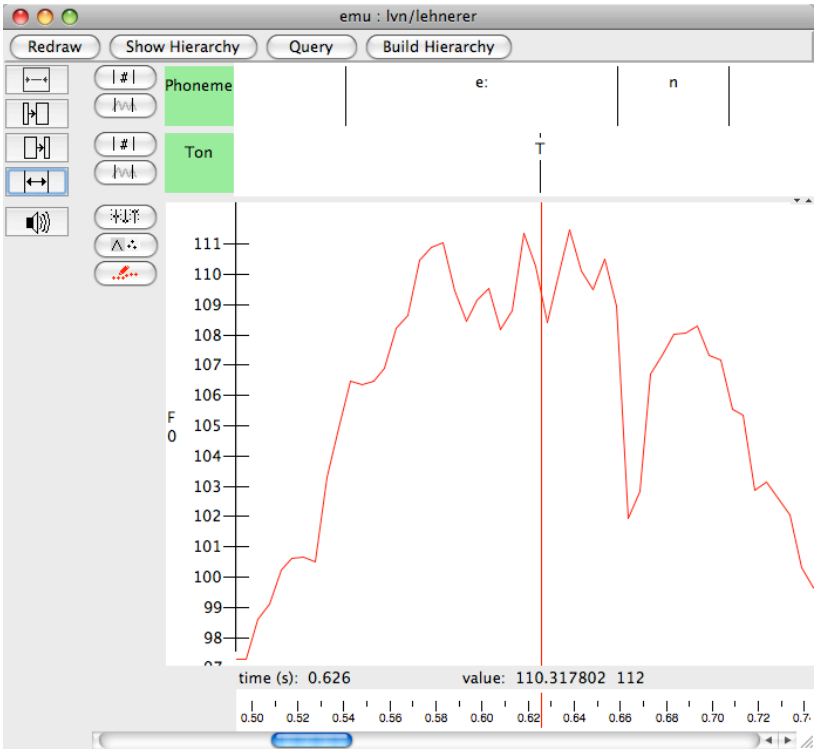
1. Formanten in R glätten (eine Parabel Ordnung 2 anpassen).



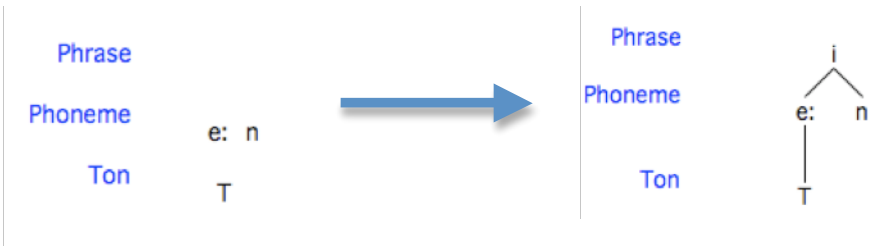
2. Den Zeitpunkt des f0-Maximums finden



3. Diese Zeitpunkte als Textdateien exportieren und in Emu als Etiketierung `T` der Ebene Ton einlesen



4. Ein Tcl-Skript anwenden, um die Etikettierungsstruktur automatisch zu bauen.



0. Vorarbeiten

- Template Datei lvn.tpl in S:\IPSK\dbtemplates nach H:\templates kopieren, angenommen dass H:\templates existiert und von Emu (Emu-conf Editor) sichtbar ist).
- Ein Verzeichnis erzeugen, wo hlb-Dateien gespeichert werden
- Dementsprechend `path for hlb files` in der Template-Datei (`Levels` Registerkarte) ändern

```
# Segmentliste von beiden Segmenten der Ebene Phonem
rhyme = emu.query("lvn", "*", "[Phoneme!=x -> Phoneme!=x]")

# ggf. die Segmentliste nach links und rechts um zB 50 ms verlängern
rhyme[,2] = start(rhyme)-50
rhyme[,3] = end(rhyme)+50

# f0-Daten
rhyme.f0 = emu.track(rhyme, "F0")
```

1. Glättung

```
# Die Frames vom zweiten Segment
daten = rhyme.f0[2]$data

# Anpassung einer Parabel
p = plafit(daten, fit=T)

# Parabel anpassen auf alle Segmente und eine Trackdatei bauen
glatt = trapply(rhyme.f0, plafit, fit=T, buildtrack=T)
```

2. Funktion um den Zeitpunkt vom f0-Maximum oder -minimum zu finden

```
mfun <- function(daten, fun=max)
{
zeiten = tracktimes(daten)
temp = daten == fun(daten)
zeiten[temp][1]
}

# zB
m = mfun(glatt[2,]$data)
plot(glatt[2,])
abline(v=m)

# Zeiten vom Maximum in allen Segmenten?
```

```
maxzeiten = trapply(

# Abbilden pro Segment
lab = c("rauh", "glatt")

for(j in 1:5){
newt = rbind(rhyme.f0[j,], glatt[j,])
dplot(newt, lab)
abline(v = maxzeiten[j] - start(glatt[j,]), col=3, lwd=2, lty=2)
locator(1)
}
```

3. exportieren

```
path = "H:"
makelab(maxzeiten, utt(rhyme), path, "ton", "T")
```

4. Tcl-Skript.

Siehe auch: http://emu.sourceforge.net/manual/ch07.html

4.1 In der Console ausprobieren

Console aufmachen: `File -> Console`

```
#  Zugang zur Emu-Tcl Library
package require emu::autobuild

#  Mit der emutemplate Funktion
# wird die Template Datei lvn als t laden (t wird zur Funktion)
# Übrigens: emutemplate ohne Argumente macht eine Liste aller zugänglichen
# Sprachdatenbanken
emutemplate t lvn

# Mit der hierarchy Funktion wird die Etikettierungsstruktur der Äußerung lehnerer
# dieser Datenbank als h laden - h wird zur Funktion
t hierarchy h lehnerer

# h segments E findet die Segmente der Ebene E.
# set x y speichert y als x Die Segmente der Ton-Ebene
set ton [h segments Ton]

# Die Segmente der Phoneme-Ebene
set phon [h segments Phoneme]

# Das erste davon lindex $x 4 findet das dritte Element von x
set phonerstes [lindex $phon 0]

# Mit h seginfo $x children E $y werden alle Elemente in y der Ebene E Kinder
# von x (x darf nur aus einem Element bestehen; y aus mehreren)
# Das Segment der Ton-Ebene wird Kind
```

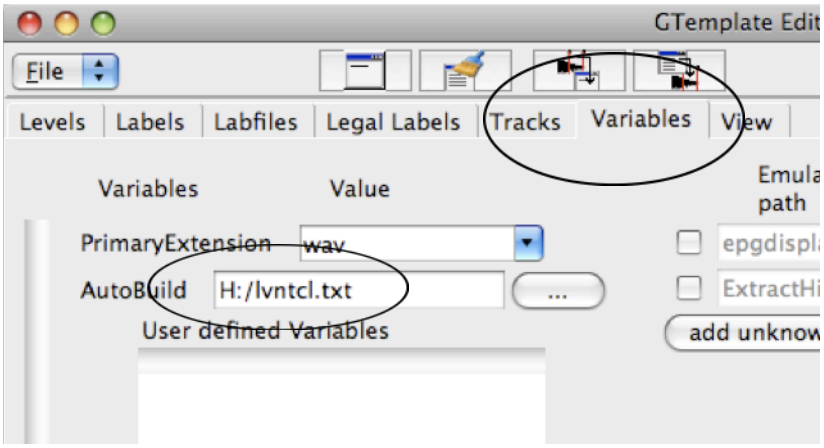
```
# vom ersten Segment der Phonem-Ebene
h seginfo $phonerstes children Ton $ton

# Ein Segment "i" der Phrase-Ebene hinzufügen
h append Phrase "i"

# Segment der Phrase-Ebene holen
set p [h segments Phrase]

# Die Segmente der Phoneme-Ebene werden
# Kinder der Phrase-Ebene
h seginfo $p children Phoneme $phon

# hlb Datei erzeugen
h write lehnerer
```

4.2 Als Funktion umsetzen.

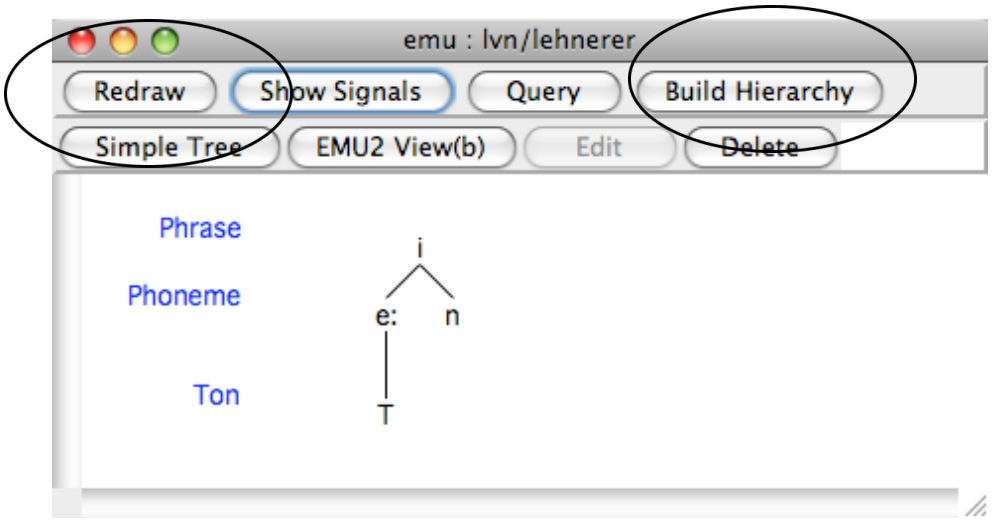
Folgendes in einer Text-Datei (zB lvntcl.txt) speichern.

```
package require emu::autobuild
proc AutoBuildInit {t} {}

proc AutoBuild {t h} {
# Die Segmente der Ton-Ebene
set ton [$h segments Ton]

# Die Segmente der Phoneme-Ebene
set phon [$h segments Phoneme]

# Das erste davon
set phonerstes [lindex $phon 0]

# Das Segment der Ton-Ebene wird Kind
# vom ersten Segment der Phonem-Ebene
$h seginfo $phonerstes children Ton $ton

# Ein Segment "i" der Phrase-Ebene erzeugen
$h append Phrase "i"

# Segment der Phrase-Ebene holen
set p [$h segments Phrase]

# Die Segmente der Phoneme-Ebene werden
# Kinder der Phrase-Ebene
$h seginfo $p children Phoneme $phon

}
```

4.3 Template-Datei editieren, um den Skript zu laden



4.4. Auf eine Äußerung mit Build Hierarchy anwenden



4.5 Skript auf alle Äußerungen anwenden

```
Database Operations -> AutoBuild Extern
```