Bijlage 1 Worldbet voor het Fries

IPA-symbool	*Worldbet*	*voorbeeld*	*transcriptie*
p	p	part	p a t
b	b	bal	b > l
t	t	ta	t a
d	d	daam	d a: m
k	k	klear	k l I& r
g	g	goed	g u& t
f	f	fol	f o l
v	v	skevel	s k e: v & l
s	s	stil	s t I l
z	z	wêze	w E: z &
x	x	rûch	r u x
ɣ	G	drage	d r a: G &
h	h	heech	h e: x
m	m	laam	l a: m
n	n	noch	n > x
ŋ	N	ring	r I N
l	l	slikje	s l I k j &
r	r	raar	r a: r
ʋ	V	wetter	V E t & r
w	w	wûn	w u n
j	j	jas	j > s
vocalen			
i	i	dyk	d i k
ɪ	I	ik	I k
ɛ	E	let	l E t
y	y	nút	n y t
ʏ	Y	nut	n Y t
a	a	ta	t a
u	u	rûch	r u x
ʊ	U	rom	r U m
ɔ	>	kat	k > t
ə	&	de	d &
i:	i:	tiid	t i: t
e:	e:	reed	r e: t
ɛ:	E:	bêd	b E: t
y:	y:	drúf	d r y: f
ø:	7:	deun	d 7: n
a:	a:	baas	b a: s
u:	u:	sûch	s u: x
ʊ:	U:	rook	r U: k
ɔ:	>:	sâlt	s >: t
diftongen			
ɪ:ə	i&	biede	b i& d &
iu	iu	ieu	iu
ɪ:ə	I&	hea	h I&

ɛi	Ei	rij	r Ei
ɔi ai	>i ai	laitsje	l >i t S j & l ai t S j &
au	Au	gau	g Au
y:ə	y&	flues	f l y& s
ʏ:ə	Y&	gleon	g l Y& n
œy	8y	bui	b 8y
u:ə	u&	goed	g u& t
ui	ui	bloei	b l ui
ʊi	Ui	floite	f l Ui t &
ʊ:ə	U&	boat	b U& t
ʊəi	U&i	moai	m U&_i
stijgende difongen:			
Jɪ	jI	triedden	t r j I d n
Jɛ	jE	beammen	b j E m n
Jʏ	jY	sljurkje	s l j Y r k j &
jo	jU	rjocht	r j U x t
jɔ	j>	ljocht	l j > x t
wʊ	wU	fuotten	f w U t n
wa	wa	boarje	b w a r j &
wa:	wa:	moarns	m w a:_~ s
klanken uit leenwoorden			
œ:	8:	freule	f r 8: l &
ʃ	S	lunch	l Y n S
ʒ	Z	rûzje	r u: Z j &

Bijlage 2 an_mbrola.scm

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;                                                                                                                          ;;
;;              Centre for Speech Technology Research                                                         ;;
;;                  University of Edinburgh, UK                                                                       ;;
;;                   Copyright (c) 1996,1997                                                                       ;;
;;                    All Rights Reserved.                                                                            ;;
;;                                                                                                                          ;;
;;  Permission is hereby granted, free of charge, to use and distribute                                      ;;
;;  this software and its documentation without restriction, including                                         ;;
;;  without limitation the rights to use, copy, modify, merge, publish,                                        ;;
;;  distribute, sublicense, and/or sell copies of this work, and to                                              ;;
;;  permit persons to whom this work is furnished to do so, subject to                                       ;;
;;  the following conditions:                                                                                         ;;
;;   1. The code must retain the above copyright notice, this list of                                          ;;
;;      conditions and the following disclaimer.                                                                    ;;
;;   2. Any modifications must be clearly marked as such.                                                       ;;
;;   3. Original authors' names are not deleted.                                                                 ;;
;;   4. The authors' names are not used to endorse or promote products                                      ;;
;;      derived from this software without specific prior written                                               ;;
;;      permission.                                                                                                     ;;
;;                                                                                                                          ;;
;;  THE UNIVERSITY OF EDINBURGH AND THE CONTRIBUTORS TO THIS WORK               ;;
;;  DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING              ;;
;;  ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT           ;;
;;  SHALL THE UNIVERSITY OF EDINBURGH NOR THE CONTRIBUTORS BE LIABLE           ;;
;;  FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES       ;;
;;  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN      ;;
;;  AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION,                 ;;
;;  ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF            ;;
;;  THIS SOFTWARE.                                                                                                  ;;
;;                                                                                                                          ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;


;;;  Add the directory contains general frisian stuff to load-path
(defvar an_mbrola_dir (cdr (assoc 'an_mbrola voice-locations)))
(set! load-path (cons (path-append an_mbrola_dir "festvox/") load-path))

;;; other files we need
(require '/u/jdijkstra/festival/lib/mbrola)
(require '/u/jdijkstra/festival/lib/voices/frisian/an_diphone/festvox/frislex)
(require '/u/jdijkstra/festival/lib/voices/frisian/an_diphone/festvox/frisint)
(require '/u/jdijkstra/festival/lib/voices/frisian/an_diphone/festvox/fristoken)
(require '/u/jdijkstra/festival/lib/mrpa_phones)
;; (require '/u/jdijkstra/festival/lib/mrpa_durs)
;; (require '/u/jdijkstra/festival/lib/gswdurtreeZ)

;;; Part of speech down by crude lookup using gpos
(set! frisian_guess_pos
  '((yn fan foar op dat dy""" t mei by """ut as
   at tsjin oer foar omdat om"""t "^"under nei
   wylst s"^"under troch nij tusken per de neist
   noch sels want sa ear"""t sa"""t yntusken "^"undertusken
   wyllens dochs hoewol"""t al wannear"""t dan oan tichtby
   tsjinoan tsjinoer fan"^"of om hinne fia oant boppe
   efter achter """tsein)
   (te)
   (det de it in """e """t """n gjin net guon dizze dy dat elk elts
```

```
   oar alle elke beide soad)
   (md sille sil sist meie mei meist kinne kin kinst
   soene soe soest mochten mocht mochtst koene koe koest
   moatte moat moatst moast)
   (cc en mar of as at en plus noch)
   (wp wa wat w"^"er hoe wannear)
   (pps myn dyn syn har """us jo harren
   my dy him har jim jimme jo)
   (aux bin bist is binne wie wiest wiene
   haw ha hast hawwe hie hiest hiene)
))

;;; Phrase breaks
;;; use punctuation
(set! frisian_phrase_cart_tree
'
((lisp_token_end_punc in ("?" "." ":"))
  ((BB))
  ((lisp_token_end_punc in ("'" "\"" "," ";"))
    ((B))
    ((n.name is 0)
     ((BB))
     ((NB))))))

;;; Intonation
(set! frisian_accent_cart_tree
  '
  (
   (R:SylStructure.parent.gpos is content)
    ( (stress is 1)
       ((Accented))
       ((NONE))
    )
  )
)

;;; Duration
(set! frisian_dur_tree
'
   ((R:SylStructure.parent.R:Syllable.p.syl_break > 1 ) ;; clause initial
    ((R:SylStructure.parent.stress is 1)
     ((1.5))
     ((1.2)))
    ((R:SylStructure.parent.syl_break > 1)   ;; clause final
     ((R:SylStructure.parent.stress is 1)
      ((1.5))
      ((1.2)))
     ((R:SylStructure.parent.stress is 1)
      ((ph_vc is +)
       ((1.2))
       ((1.0)))
      ((1.0))))))

(set! frisian_an_phone_data
'(

   (# 0.0 0.250)
   (a 0.0 0.040)
   (E 0.0 0.050)
   (> 0.0 0.040)
```

```
(& 0.0 0.040)
(I 0.0 0.040)
(Y 0.0 0.040)
(U 0.0 0.040)
(i 0.0 0.050)
(y 0.0 0.040)
(u 0.0 0.040)

(a: 0.0 0.080)
(E: 0.0 0.090)
(8: 0.0 0.080)
(>: 0.0 0.090)
(e: 0.0 0.080)
(7: 0.0 0.080)
(o: 0.0 0.080)
(i: 0.0 0.070)
(y: 0.0 0.080)
(u: 0.0 0.070)

(a1 0.0 0.050)
(E1 0.0 0.060)
(>1 0.0 0.050)
(&1 0.0 0.050)
(I1 0.0 0.050)
(Y1 0.0 0.050)
(U1 0.0 0.050)
(i1 0.0 0.060)
(y1 0.0 0.050)
(u1 0.0 0.050)

(i& 0.0 0.090)
(iu 0.0 0.090)
(I& 0.0 0.090)
(Ei 0.0 0.090)
(>i 0.0 0.090)
(Au 0.0 0.090)
(y& 0.0 0.090)
(Y& 0.0 0.090)
(8y 0.0 0.090)
(u& 0.0 0.090)
(ui 0.0 0.090)
(oi 0.0 0.090)
(U& 0.0 0.090)
(U&_i 0.0 0.100)

(b 0.0 0.065)
(d 0.0 0.060)
(f 0.0 0.100)
(g 0.0 0.080)
(j 0.0 0.100)
(k 0.0 0.100)
(l 0.0 0.080)
(m 0.0 0.070)
(n 0.0 0.080)
(N 0.0 0.110)
(p 0.0 0.100)
(r 0.0 0.080)
(s 0.0 0.110)
(t 0.0 0.085)
(x 0.0 0.130)
```

```
   (v 0.0 0.100)
   (z 0.0 0.110)
   (G 0.0 0.130)
   (V[ 0.0 0.100)
   (S 0.0 0.110)
   (Z 0.0 0.110)
   (h 0.0 0.080)
   (w 0.0 0.100)
   (j 0.0 0.100)
))

;; Go ahead and set up the diphone db
(define (frisian_voice_reset)
  "(frisian_voice_reset)
Reset global variables back to previous voice."
  (set! token.prepunctuation frisian_previous_tok_prepunc)
)

;;;  Full voice definition
(define (voice_an_mbrola)
"(voice_frisian_an)
Set up synthesis for Male Frisian speaker: an"
  (voice_reset)
  (Parameter.set 'Language 'frisian)
  ;; Phone set
  (Parameter.set 'PhoneSet 'frisian)
  (PhoneSet.select 'frisian)

  ;; numeric expansion
  (Parameter.set 'Token_Method 'Token_Any)
  (set! token_to_words frisian_token_to_words)

  ;; Because of use of ' for accents remove it from prepunctuation
  (set! frisian_previous_tok_prepunc token.prepunctuation)
  (set! token.prepunctuation "\"`(){}[]")

  ;; No pos prediction (get it from lexicon)
  (set! pos_lex_name nil)
  ;; Phrase break prediction by punctuation
  (set! pos_supported nil) ;; well not real pos anyhow

  ;; Lexicon selection
  (lex.select "frisian")

;; Phrase prediction - uit en1_mbrola.scm, echter werkt niet onder deze file
;;  (Parameter.set 'Phrase_Method 'prob_models)
;;  (set! phr_break_params frisian_phr_break_params)

;; Accent and tone prediction
;; uit en1_mbrola.scm; hiermee klinkt stem natuurlijker
(set! int_tone_cart_tree f2b_int_tone_cart_tree)
(set! int_accent_cart_tree f2b_int_accent_cart_tree)
;; F0 prediction
(set! f0_lr_start f2b_f0_lr_start)
(set! f0_lr_mid f2b_f0_lr_mid)
(set! f0_lr_end f2b_f0_lr_end)
(Parameter.set 'Int_Method Intonation_Tree)
(set! int_lr_params
          ((target_f0_mean 150) (target_f0_std 15)
            (model_f0_mean 180) (model_f0_std 34)))
```

```
(Parameter.set 'Int_Target_Method Int_Targets_LR)

;; Duration prediction -- use gsw durations
;; uit en1_mbrola.scm; hiermee ook snellere spraak mogelijk
;;  (set! duration_cart_tree gsw_duration_cart_tree)
;;  (set! duration_ph_info gsw_durs)
;;  (Parameter.set 'Duration_Method Duration_Tree_ZScores)
;;  (Parameter.set 'Duration_Stretch 0.95)

  ;; Phrasing
  (set! phrase_cart_tree frisian_phrase_cart_tree)
  (Parameter.set 'Phrase_Method 'cart_tree)
  ;; Lexicon selection
  (lex.select "frisian")

;; Accent and tone prediction - uit an_diphone.scm
;; stem klinkt te laag
;;  (set! int_accent_cart_tree frisian_accent_cart_tree)
;;
;;  (Parameter.set 'Int_Target_Method 'Simple)

;;  (Parameter.set 'Int_Method 'General)
;;  (set! int_general_params (list (list 'targ_func targ_func1)))
;;  (set! guess_pos frisian_guess_pos)

  ;; Duration prediction
  (set! duration_cart_tree frisian_dur_tree)
  (set! duration_ph_info frisian_an_phone_data)
  (Parameter.set 'Duration_Method 'Tree_ZScores)


(Parameter.set 'Synth_Method MBROLA_Synth)
;; Because we need an extra parameter in the new version of mbrola
;; we add that parameter to the database "name"
(set! mbrola_progname
"/u/jdijkstra/festival/lib/voices/frisian/an_diphone/mbrola/mbrola-linux-i386")

(set! mbrola_database
        (format nil
        "-I %s%s %s%s "
        "/u/jdijkstra/festival/lib/voices/frisian/an_diphone/mbrola/" "anmrpa"
        "/u/jdijkstra/festival/lib/voices/frisian/an_diphone/mbrola/"
"fy1/temp.dat"
        ))
;; set callback to restore some original values changed by the frisian voice
(set! current_voice_reset frisian_voice_reset)

(set! current-voice 'an_mbrola)
)

(proclaim_voice
'an_mbrola
'((language frisian)
   (gender female)
   (dialect Kleifries)
   (description
    "This voice provides a Frisian male voice using a
     residual excited LPC diphone synthesis method. The lexicon
     is provived by a set of letter to sound rules producing pronunciation
     accents and syllabification.  The durations, intonation and
```

```
    prosodic phrasing are minimal but are acceptable for simple
    examples.")))

(provide '/u/jdijkstra/festival/lib/voices/frisian/an_diphone/festvox/an_mbrola)
(voice_an_mbrola)
```

Bijlage 3 frislex.scm

```
;;; Lexicon
(lex.create "frisian")
(lex.set.phoneset "frisian")
(lex.set.lts.method 'frisian_lts )
(lex.set.lts.ruleset 'frisian )

;;; This which just have to be in the lexicon
(lex.add.entry '("a" nn (((a:) 0))))
(lex.add.entry '("b" nn (((b e:) 0))))
(lex.add.entry '("c" nn (((c e:) 0))))        ;;; bestaat niet in Fries, maar
                                              ;;; misschien als letter in bedrijfsnaam?
(lex.add.entry '("d" nn (((d e:) 0))))
(lex.add.entry '("e" nn (((e:) 0))))
(lex.add.entry '("f" nn (((E f) 0))))
(lex.add.entry '("g" nn (((x e:) 0))))
(lex.add.entry '("h" nn (((h a:) 0))))
(lex.add.entry '("i" nn (((i) 0))))
(lex.add.entry '("j" nn (((j e:) 0))))
(lex.add.entry '("k" nn (((k a:) 0))))
(lex.add.entry '("l" nn (((E l) 0))))
(lex.add.entry '("m" nn (((E m) 0))))
(lex.add.entry '("n" nn (((E n) 0))))
(lex.add.entry '("o" nn (((o:) 0))))
(lex.add.entry '("p" nn (((p e:) 0))))
(lex.add.entry '("q" nn (((k y) 0))))  ;;; bestaat niet in Fries
(lex.add.entry '("r" nn (((E r) 0))))
(lex.add.entry '("s" nn (((E s) 0))))
(lex.add.entry '("t" nn (((t e:) 0))))
(lex.add.entry '("u" nn (((y) 0))))
(lex.add.entry '("v" nn (((v e:) 0))))
(lex.add.entry '("w" nn (((V| e:) 0))))
(lex.add.entry '("x" nn (((I k s) 0))))
(lex.add.entry '("y" nn (((i) 0) ((g r E1) 1)) (k) 0))
(lex.add.entry '("z" nn (((z E1 t) 0))))

(lex.add.entry
'("*" n (((a s) 0) ((t e) 0) ((r i1 s) 1)  ((k) 0))))
(lex.add.entry
'("%" n (((p r o:) 0) ((s I n t) 1))))
(lex.add.entry
'("&" n (((a1 m) 1) ((p e r) 0) ((s a n) 0))))
(lex.add.entry
'("$" n (((d >1) 1) ((I a r) 0))))
(lex.add.entry
'("#" n (((h E k) 1) ((j &) 0))))
(lex.add.entry
'("@" n (((a: p) 1) ((k &) 0) )))
(lex.add.entry
'("+" n (((p I Y s) 0))))
(lex.add.entry
'("^" n (((d a1 k) 1) ((j &) 0)) ((pos "K6$"))))
(lex.add.entry
'("~" n (((g >1) 1) ((I y f ) 0) ((k e) 0)) ((pos "K6$"))))
(lex.add.entry
'("=" n ((((I s) 1)))))
(lex.add.entry
'("/" n (((e1 n ) 1) ((t r e) 0))))  ;; $$$division, etc.
(lex.add.entry
```

```
'("\\" n (((b a1) 1) ((rr a) 1))))
(lex.add.entry
'("_" n (((u n) 1) ((d & r) 0) ((s t r e: k) 1) ((j &) 0)) ))
(lex.add.entry
'("|" n (((b a1) 1) ((rr a) 0))))
(lex.add.entry
'(">" n ((( g r Y1 ) 1) ((t & r) 0) ((# > s) 1))))
(lex.add.entry
'("<" n ((( l i1 t ) 1) ((s & r) 0) ((# > s) 1))))
(lex.add.entry
'("[" n ((( a) 0) ((b r i1 r) 1) ((k o r) 0)((ch e1) 1)((t e) 0))))
(lex.add.entry
'("]" n (((th e) 0) ((rr a1 r) 1) ((k o r) 0)((ch e1) 1)((t e) 0))))
(lex.add.entry
'("\"" n (((k l a1 ) 1)((m & r) 0)((k & s) 0))))
(lex.add.entry
'("\t" n (((t a1 b) 1))))
(lex.add.entry
'("\n" n (((n Ei)1) (( j &) 0) ((a) 0)((l i) 1) ((n e a) 0))))

(lex.add.entry '("." punc nil))
(lex.add.entry '("." nn (((p Y1 n t) 1))))
(lex.add.entry '("'" punc nil))
(lex.add.entry '(":" punc nil))
(lex.add.entry '(";" punc nil))
(lex.add.entry '("," punc nil))
(lex.add.entry '("," nn (((k o1) 1) ((m a) 0))))
(lex.add.entry '("-" punc nil))
(lex.add.entry '("\"" punc nil))
(lex.add.entry '("`" punc nil))
(lex.add.entry '("?" punc nil))
(lex.add.entry '("!" punc nil))

;; (lex.add.entry '("bjusterbaarlik" n (((b j Y s)0) ((t & r)0) ((b a: r)1) ((l &
k)0))))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;;  Down cases with accents
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(lts.ruleset
frisian_downcase
( nil )
(

  ( [ a ] = a )
  ( [ e ] = e )
  ( [ o ] = o )
  ( [ i ] = i )
  ( [ u ] = u )
  ( [ y ] = y )

  ( [ b ] = b )
  ( [ c ] = c )      ;; bestaat niet in Fries, alleen in /ch/ (ch = c h) geeft
                     ;; foutmelding
  ( [ d ] = d )
  ( [ f ] = f )
  ( [ g ] = g )
  ( [ h ] = h )
  ( [ j ] = j )
```

```
  ( [ k ] = k )
  ( [ l ] = l )
  ( [ m ] = m )
  ( [ n ] = n )
  ( [ p ] = p )
  ( [ r ] = r )
  ( [ s ] = s )
  ( [ t ] = t )
  ( [ v ] = v )
  ( [ w ] = w )
  ( [ x ] = x )      ;; bestaat eigenlijk niet in Fries
  ( [ z ] = z )


  ( [ "\"" ] = "\"" )
  ( [ \: ] = "\:" )
  ( [ ~ ] = ~ )
  ( [ "\"" ] = "\"" )

  ( [ A ] = a )
  ( [ E ] = e )
  ( [ I ] = i )
  ( [ O ] = o )
  ( [ U ] = u )
  ( [ Y ] = y )
  ( [ B ] = b )
  ( [ D ] = d )
  ( [ F ] = f )
  ( [ G ] = g )
  ( [ H ] = h )
  ( [ J ] = j )
  ( [ K ] = k )
  ( [ L ] = l )
  ( [ M ] = m )
  ( [ N ] = n )
  ( [ P ] = p )
  ( [ R ] = r )
  ( [ S ] = s )
  ( [ T ] = t )
  ( [ V ] = v )
  ( [ W ] = V| )
  ( [ Z ] = z )
))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;   Main letter to sound rules
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;; In het Fries komen veel dakjes voor op klinkers. Deze heb ik aangemerkt als
;;; "^"V, Borja deed dit ook zo voor het Spaans bij ":" u. Ik hoop dat Festival
;;; deze code accepteert. Andere accenten: ""e en ""u en ":"V. Hieronder volgt
;;; de uitspraak van (bijna) alle mogelijke vocaal- en consonantcombinaties.

(lts.ruleset
;;;  Name of rule set
frisian
;;;  Sets used in the rules
(
  (SLDTN s l d t n)
  (STEMC b d j l m n r v w z)
  (BDG b d g )
```

```
  (VOC a e o i u y "^"a "^"e "^"o "^"u)
  (C b d f g h j k l m n ng p r s t v w z)
  (D i& iu I& Ei >i Au y& Y& 8y u& ui oi U& )
)
;  Rules
(
  ( [ # ] = # )
  ( [ - ] = - )

   ;;; einde -en? dan syllabificatie als woord groter dan 1 syllabe
    ( - [ y e n ] # = i j & n )
    ( - [ m e n ] # = m m )
    ( - [ k e n ] # = k N )
    ( - C [ e n ] # = n )

    ( - [ l i k ] # = l & k )

   ;;; e = & en woord is groter dan 1 syllabe
   ( - [ s u m ] # = s & m )
   ( - C [ e l ] # = & l )
   ( - C [ e m ] # = & m )
   ( - C [ e r ] # = & r )
   ( - C [ e t ] # = & t )

   ;;; begin e = & en woord is groter dan 1 syllabe
    ( # [ b e ] - VC * = b & )
    ( # [ g e ] - VC * = g & )
    ( # [ f e r ] - VC * = f & r )
    ( VC * - [ d e r ] - = d & r )
    ( - [ t e r ] - = t & r )

;;; lange klinker als lettergreep open, behalve bij e-finaal (woord)
;;; y komt alleen in gesloten lettergrepen voor, i en u niet woordfinaal
    ( [ a ] - = a: )
    ( [ e ] - = e: )
    ( [ i ] - = i: )
    ( [ o ] - = U: )
    ( [ u ] - = u: )
    ( [ a ] # = a: )
    ( [ e ] # = & )
    ( [ o ] # = U: )

   ;;; korte klinker als lettergreep gesloten
    ( [ a ] C * - = a )
    ( [ e ] C * - = E )
    ( [ i ] C * - = I )
    ( [ o ] C * - = U )
    ( [ u ] C * - = Y )
    ( [ a ] C * # = a )
    ( [ e ] C * # = E )
    ( [ i ] C * # = I )
    ( [ o ] C * # = U )
    ( [ u ] C * # = Y )

  ;; triphtongs
    ( [ o a i ] = U>_i )
    ( [ u o i ] = w > i )
    ( [ a a i ] = >i )          ;;; geen triftong, maar wel drie lettercombinaties
    ( [ o e i ] = ui )          ;;; geen triftong, maar wel drie lettercombinaties
    ( [ i u w ] = j U: u)
```

```
   ( [ e a u ] = j U: u)

;;; nasaliteit
  ( [ o a n s ] = w a:~ s )
  ( [ o a r n s ] = w a:~ s )
  ;; bij diftongen in D
  ( D [ n j ] = D~ j )        ;; doet het niet, uitschrijven voor elke D als ( [ i e n
                              ;; j ] = i&~ j ), misschien later?
  ( D [ r n s ] = D~ s ) ;; idem
  ( D [ n s ] = D~ s )  ;; idem
  ( D [ n z ] = D~ z ) ;; idem
  ( [ i r n s ] = e:~ s )
  ( [ i r n z ] = e:~ z )
  ( [ i n s ] = e:~ )
  ( [ i n z ] = e:~ z )
  ( VOC [ r n s ] = VOC:~ s )     ;; doet het niet, uitschrijven voor elke VOC, zie
                                  ;; hierboven bij D
  ( VOC [ n j ] = VOC:~ j )
  ( VOC [ n s ] = VOC:~ s )
  ( VOC [ n z ] = VOC:~ z )

  ( [ a a i ] = >i )
  ( [ a u ] = Au )
  ( [ a i ] = >i )
  ( [ a a ] = a: )
  ( [ a ] SLDTN  = > )
  ( [ a ] = a )

  ( [ "^" a l t ] = >: t )
  ( [ "^" a l d ] = >: t )
  ( [ "^"a ] = >: )

  ( [ o u w e ] = Au w & )
  ( [ o w e ] = U: w & )
  ( [ i u w e ] = j U: w & )
  ( [ i j e ] = Ei j & )
  ( [ i u w ] = j U: u )
  ( [ i c h ] = & x ) ;; is dit voor alle woorden toepasbaar of alleen als ich-?
  ( [ i e u ] = iu )
  ( [ i e ] = i&  )
  ( [ i i ] = i: )
  ( [ i j ] = Ei)
  ( [ i ] = i )

  ( [ e a u w e ] = j U: w & )
  ( [ e a u ] = j U: u)
  ( [ e u r ] = Y& r )  ;;; voor r klinkt eu als Y&
  ( [ e a ] = I& )
  ( [ e e ] = e: )
  ( [ e i ] = >i )
  ( [ e o ] = Y& )
  ( [ e u ] = 7: )
  ( [ e ] = & ) ;; let op voor E, wanneer die?
  ( [ e ] # = & ) ;; doet het niet, zie d #

  ( [ """ e ] = e: )
  ( [ "^"e ] = E: )

  ( [ o u w i ] = Au I )
  ( [ o a i ] = U>_i )
```

```
( [ o e d ] = u& d )
( [ o e i ] = ui )
( [ o a ] = U& )
( [ o e ] = u& )
( [ o i ] = >i )
( [ o o ] = U: )
( [ o u ] = Au )
( [ o ] = o )
( [ o ] # = U: ) ;; doet het niet, #-finaal probleem
( [ "^"o ] = >: )

( [ u e ] = y& )
( [ u i ] = 8y )
( [ u o ] = wU )
( [ u u ] = y j & )      ;;; alleen in leenwoorden
( [ u ] = y )
( [ "" u ] = y )
( [ "^" u ] = u )

( [ y ] = i )

( [ > ] = > )
( [ E ] = E )
( [ & ] = & )

( [ b - b ] = - b )
( [ b ] = b )
( [ c h ] = x )
( [ d - d ] = - d )
( [ d # ]  = t ) ;; doet het niet # doet het finaal niet, initiaal wel.
( [ d ] = d )
( [ f - f ] = - f )
( [ f ] = f )
( [ g - g ] = - g )
( VOC [ g ] VOC = G )
( STEMC [ g ] STEMC = G )
( STEMC [ g ] VOC = G )
( VOC [ g ] STEMC = G )
( [ g ] = g )
( [ g ] # = x ) ;;  doet het niet
( # [ h j ] = j )
( [ h ] = h )
( [ j ] = j )
( [ k - k ] = - k )
( [ k ] = k )
( [ l - l ] = - l )
( [ l ] = l )
( [ m - m ] = - m )
( [ m ] = m )

( # [ "^"u n - g e ] - = u N - g & )      ;; doet het niet, ook niet meer naar
                                          ;; boven (u N)
( [ n - g ] = N - )
( [ n - k ] = N - k )
( [ n g ] = N )
( [ n k ] = N k )
( [ n - m ] = - m )
( [ n - n ] = - n )
( [ n ] = n )
```

```
  ( [ p - p ] = - p )
  ( [ p ] = p )
  ( [ r z ] = z )
  ( [ r - r ] = - r )
  ( [ r s ] - = s ) ;; deze regel doet het niet door - , verplaatsten naar syl?
  ( [ r l ] - = l ) ;; idem
  ( [ r d ] - = d ) ;; idem
  ( [ r t ] - = t ) ;; idem
  ( [ r n ] - = n ) ;; idem
  ( [ r ] = r )
  ( [ s - s ] = s )
  ( [ s - j ] = S - j )
  ( [ s ] BDG = z )
  ( [ s ] = s )
  ( [ t - t ] = t )
  ( [ t ] = t )
  ( [ v - v ] = v )
  ( [ v ] = v )
  ( [ w - w ] = - V| )
  ( [ w ] = V| )
  ( [ x ] = k s )
  ( [ z - j ] = Z - j )
  ( [ z - z ] = - z )
  ( [ z ] = z )

  ;;; quotes are used for vowel accents in foreign keyboards (i.e. cami'on).
  ;;; remove those that were not before a vowel. same with other signs.
( [ "'" ] = )
( [ ":" ] = )
( [ "\"" ] = )
))

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;  Frisian sylabification by rewrite rules
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(lts.ruleset
   frisian_syl
   (  (V a e i o u y "^"a "^"e "^"i "^"o "^"u "'"e )
      (C b d f g x h j k l m n N p r s t v w z )
   )
   ;; Rules will add - at syllable boundary
   (
     ;; valid CC groups
     ( V C * [ b l ] V C * = - b l )
     ( V C * [ b r ] V C * = - b r )
     ( V C * [ k j ] V C * = - k j ) ;; maar als xxxxk-je
     ( V C * [ k l ] V C * = - k l )
     ( V C * [ k n ] V C * = - k n )
     ( V C * [ k r ] V C * = - k r )
     ( V C * [ k w ] V C * = - k w )
     ( V C * [ d j ] V C * = - d j ) ;; maar xxxxd-je
     ( V C * [ d r ] V C * = - d r )
     ( V C * [ d w ] V C * = - d w )
     ( V C * [ f j ] V C * = - f j )
     ( V C * [ f l ] V C * = - f l )
     ( V C * [ f r ] V C * = - f r )
     ( V C * [ g j ] V C * = - g j )
     ( V C * [ g l ] V C * = - g l )
     ( V C * [ g r ] V C * = - g r )
```

```
    ( V C * [ l j ] V C * = - l j ) ;; maar xxxxl-je
    ( V C * [ p l ] V C * = - p l )
    ( V C * [ p r ] V C * = - p r )
    ( V C * [ p s ] V C * = - p s ) ;; maar soms ook p-s
    ( V C * [ r b ] V C * = r - b )
    ( V C * [ r j ] V C * = - r j ) ;; maar xxxxr-je
    ( V C * [ s j ] V C * = s - j ) ;; maar xxxxs-je
    ( V C * [ s k ] V C * = - s k ) ;; maar soms ook sk- of s-k
    ( V C * [ s l ] V C * = - s l )
    ( V C * [ s n ] V C * = - s n )
    ( V C * [ s m ] V C * = - s m )
    ( V C * [ s p ] V C * = - s p )
    ( V C * [ s t r ] V C * = - s t r )
    ( V [ s t ] V C * = s - t )
    ( V C * [ s t ] V C * = - s t )
    ( V C * [ s w ] V C * = - s w )
    ( V C * [ t j ] V C * = - t j ) ;; maar xxxxt-je
    ( V C * [ t r ] V C * = - t r )
    ( V C * [ t s ] V C * = - t s ) ;; maar soms ook ts- of t-s
    ( V C * [ t w ] V C * = - t w )
    ( V C * [ w j ] V C * = - w j ) ;; maar xxxxw-je
    ( V C * [ w r ] V C * = - w r )
    ( V C * [ z j ] V C * = z - j )

;; break invalid triptongs, in tekst altijd aangegeven door trema of platte
;; streep
  ( [ a a i ":" i ] = >i - I )
  ( [ a ":" a ] = a - a )
  ( [ a ":" i ] = a - i )
  ( [ e e ":" e ] = e: - & )
  ( [ e i ":" i ] = >i - & )
  ( [ e ":" a ] = e: - a )
  ( [ e ":" e e ] = & - e: )
  ( [ e ":" e ] = & - e: )
  ( [ e ":" i c h ] = i - & )
  ( [ e ":" i e ] = & - i& )
  ( [ e ":" i ] = & - i )
  ( [ e ":" u ] = e: - y )
  ( [ i ":" e ] = i - e: )
  ( [ i ":" i ] = i - I )
  ( [ o a i ":" i ] = U>_i - I )
  ( [ o ":" a ] = U - a )
  ( [ o ":" e ] = U: - e: )
  ( [ o ":" o ] = U: - U )
  ( [ u ":" i ] = y - i )

  ( [ a - e ] = a - e )
  ( [ a - a ] = a - a )
  ( [ e - a ] = e - a )

    ;; If any consonant is followed by a vowel and there is a vowel
    ;; before it, its a syl break
    ;; the consonant cluster are dealt with above
    ( V C * [ b ] V = - b )
    ( V C * [ d ] V = - d )
    ( V C * [ f ] V = - f )
    ( V C * [ g ] V = - g )
    ( V C * [ x ] V = - x )
    ( V C * [ j ] V = - j )
    ( V C * [ k ] V = - k )
```

```
    ( V C * [ l ] V = - l )
    ( V C * [ m ] V = - m )
    ( V C * [ n ] V = - n )
    ( V C * [ p ] V = - p )
    ( V C * [ r ] V = - r )
    ( V C * [ s ] V = - s )
    ( V C * [ t ] V = - t )
    ( V C * [ v ] V = - v )
    ( V C * [ w ] V = - w )
    ( V C * [ z ] V = - z )

   ;;; Catch all consonants on their own (at end of word)
   ;; and vowels not preceded by vowels are just written as it -> default
  ( [ "^"a ] = "^"a )
  ( [ "^"e ] = "^"e )
  ( [ """e ] = """e )
  ( [ "^"u ] = "^"u )
  ( [ """u ] = """u )
  ( [ a ] = a )
  ( [ e ] = e )
  ( [ o ] = o )
  ( [ i ] = i )
  ( [ u ] = u )
  ( [ y ] = y )

  ( [ b ] = b )
  ( [ c ] = c )      ;; bestaat niet in Fries, alleen in /ch/ (ch = c h) geeft
                     ;; foutmelding
  ( [ d ] = d )
  ( [ f ] = f )
  ( [ g ] = g )
  ( [ h ] = h )
  ( [ j ] = j )
  ( [ k ] = k )
  ( [ l ] = l )
  ( [ m ] = m )
  ( [ n ] = n )
  ( [ p ] = p )
  ( [ r ] = r )
  ( [ s ] = s )
  ( [ t ] = t )
  ( [ v ] = v )
  ( [ w ] = w )
  ( [ x ] = x )      ;; bestaat eigenlijk niet in Fries
  ( [ z ] = z )
   )
)

;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;  Stress assignment in unstress words by rewrite rules
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;

(lts.ruleset
;; Assign stress to a vowel when non-exists
frisian.stress
(

  ;;; frisian:
  (V a e i o u y "^"a "^"e "^"o "^"u )
  (V1 a1 E1 >1 I1 Y1 U1 i1 y1 u1)
```

```
  (C b d f g x G j k l m n N p r s t v V| w z)
  (VC b d f g x G j k l m n N p r s t v V| w z a e i o u y "^"a "^"e "^"o "^"u)
  (NS b g f j t )

  )
(

  ;;; als woord begint met voorvoegsel dan stress op 2e syllabe
  ;;; ( # NOT - C * [V] = V1 )
  ;;; ( # NOT - C * C * [V] = V1 )
  ;;; ( # NOT - C * C * C * [V] = V1)

  ;;; anders stress op 1e syllabe
  ;;; ( # C * [V] = V1 )
  ;;; ( # C * C * [V] = V1 )
  ;;; ( # C * C * C * [V] = V1 )

  ;; consonants to themselves
    ( [ b ] = b )
    ( [ d ] = d )
    ( [ f ] = f )
    ( [ g ] = g )
    ( [ G ] = G )
    ( [ x ] = x )
    ( [ h ] = h )
    ( [ j ] = j )
    ( [ k ] = k )
    ( [ l ] = l )
    ( [ m ] = m )
    ( [ n ] = n )
    ( [ N ] = N )
    ( [ p ] = p )
    ( [ r ] = r )
    ( [ S ] = S )
    ( [ s ] = s )
    ( [ t ] = t )
    ( [ v ] = v )
    ( [ V| ] = V| )
    ( [ w ] = w )
    ( [ Z ] = Z )
    ( [ z ] = z )
    ( [ - ] = - )

    ; all vowels to themselves
    ( [ a ]  =  a )
    ( [ i ]  =  i )
    ( [ I ] = I )
    ( [ & ] = & )
    ( [ E ]  =  E )
    ( [ > ] = > )
    ( [ U ] = U )
    ( [ u ]  =  u )
    ( [ y ] = y )
    ( [ Y ] = Y )
    ( [ a: ] = a: )
    ( [ e: ] = e: )
    ( [ E: ] = E: )
    ( [ i: ] = i: )
    ( [ y: ] = y: )
    ( [ 7: ] = 7: )
```

```
    ( [ u: ] = u: )
    ( [ o: ] = U: )
    ( [ >: ] = >: )
    ( [ 8: ] = 8: )

    ( [ Au ] = Au )
    ( [ I& ] = I& )
    ( [ Y& ] = Y& )
    ( [ i& ] = i& )
    ( [ Ei ] = Ei )
    ( [ iu ] = iu )
    ( [ U& ] = U& )
    ( [ u& ] = u& )
    ( [ Ui ] = Ui )
    ( [ y& ] = y& )
    ( [ 8y ] = 8y )
    ( [ >i ] = >i )
    ( [ ui ] = ui )
    ( [ U>_i ] = U>_i )

    ( [ a:~ ] = a:~ ) ;; nasale vocalen en diftongen
    ( [ E:~ ] = E:~ )
    ( [ 8:~ ] = 8:~ )
    ( [ >:~ ] = >:~ )
    ( [ e:~ ] = e:~ )
    ( [ 7:~ ] = 7:~ )
    ( [ U:~ ] = U:~ )
    ( [ i:~ ] = i:~ )
    ( [ y:~ ] = y:~ )
    ( [ u:~ ] = u:~ )

    ( [ i&~ ] = i&~ ) ;; alle mogelijke nasale difongen?
    ( [ I&~ ] = I&~ )
    ( [ >i~ ] = >i~ )
    ( [ Au~ ] = Au~ )
    ( [ y&~ ] = y&~ )
    ( [ Y&~ ] = Y&~ )
    ( [ 8y~ ] = 8y~ )
    ( [ u&~ ] = u&~ )
    ( [ U&~ ] = U&~ )


))

(lts.ruleset
  frisian_weak_vowels
(
    (V a e i o u y "^"a "^"e "^"i "^"o "^"u """e )
    (C b d f g x h j k l m n N p r s t v w z )
)
(
;; all consonant to themselves
    ( [ b ] = b )
    ( [ d ] = d )
    ( [ f ] = f )
    ( [ g ] = g )
    ( [ G ] = G )
    ( [ x ] = x )
    ( [ h ] = h )
    ( [ j ] = j )
```

```
( [ k ] = k )
( [ l ] = l )
( [ m ] = m )
( [ n ] = n )
( [ N ] = N )
( [ p ] = p )
( [ r ] = r )
( [ S ] = S )
( [ s ] = s )
( [ t ] = t )
( [ v ] = v )
( [ V| ] = V| )
( [ w ] = w )
( [ Z ] = Z )
( [ z ] = z )
( [ - ] = - )

; all vowels to themselves
( [ a ]  =  a )
( [ i ]  =  i )
( [ I ] = I )
( [ & ] = & )
( [ E ]  =  E )
( [ > ] = > )
( [ U ] = U )
( [ u ]  =  u )
( [ y ] = y )
( [ Y ] = Y )
( [ a: ] = a: )
( [ e: ] = e: )
( [ E: ] = E: )
( [ i: ] = i: )
( [ y: ] = y: )
( [ 7: ] = 7: )
( [ u: ] = u: )
( [ o: ] = U: )
( [ >: ] = >: )
( [ 8: ] = 8: )

( [ Au ] = Au )
( [ I& ] = I& )
( [ Y& ] = Y& )
( [ i& ] = i& )
( [ Ei ] = Ei )
( [ iu ] = iu )
( [ U& ] = U& )
( [ u& ] = u& )
( [ Ui ] = Ui )
( [ y& ] = y& )
( [ 8y ] = 8y )
( [ >i ] = >i )
( [ ui ] = ui )
( [ U>_i ] = U>_i )

( [ a:~ ] = a:~ ) ;; nasale vocalen en diftongen
( [ E:~ ] = E:~ )
( [ 8:~ ] = 8:~ )
( [ >:~ ] = >:~ )
( [ e:~ ] = e:~ )
( [ 7:~ ] = 7:~ )
```

```
    ( [ U:~ ] = U:~ )
    ( [ i:~ ] = i:~ )
    ( [ y:~ ] = y:~ )
    ( [ u:~ ] = u:~ )

    ( [ i&~ ] = i&~ ) ;; alle mogelijke nasale difongen?
    ( [ I&~ ] = I&~ )
    ( [ >i~ ] = >i~ )
    ( [ Au~ ] = Au~ )
    ( [ y&~ ] = y&~ )
    ( [ Y&~ ] = Y&~ )
    ( [ 8y~ ] = 8y~ )
    ( [ u&~ ] = u&~ )
    ( [ U&~ ] = U&~ )

    ( [ # ] = # )
))

;;;  Function to turn word into lexical entry for Frisian
;;;
;;;  First uses lts to get phoneme string then assigns stress if
;;;  there is no stress and then uses a third set of rules to
;;;  mark syllable boundaries, finally converting that list
;;;  to the bracket structure festival requires

(define (frisian_lowercase lword)
  "(frisian_lowercase WORD)
Downs case word by letter to sound rules becuase or accented form
this can't use the builtin downcase function."
  (lts.apply lword 'frisian_downcase))

(define ( frisian_lts word features )
  "(frisian_lts word FEATURES)
Using various letter to sound rules build a Frisian pronunciation of
WORD."
  (let (phones syl stresssyl dword weakened)
    ( if (lts.in.alphabet word 'frisian_downcase)
         (set! dword (frisian_lowercase word )))
;;       (set! dword (frisian_lowercase "LYK" ))) ! wat doet deze regel? 1 haakje bij
;; vorige regel
;; oorspronkelijk onderstaand, echter frisian_syl en frisian omgedraaid om
;; syllabificatie te krijgen
;   (set! phones (lts.apply dword 'frisian))
;   (set! syl (lts.apply phones 'frisian_syl))
   (set! syl (lts.apply dword 'frisian_syl))
   (set! phones (lts.apply syl 'frisian))
   (if (frisian_is_a_content_word
         (apply string-append dword)
         frisian_guess_pos)
        (set! stresssyl (lts.apply phones 'frisian.stress))
        (set! stresssyl phones))  ;; function words leave as is
    (set! weakened (lts.apply stresssyl 'frisian_weak_vowels))
    (list word
          nil
          (frisian_tosyl_brackets weakened)        )))

(define (frisian_is_a_content_word word poslist)
  "(frisian_is_a_content_word WORD POSLIST)
Check explicit list of function words and return t if this is not
listed."
```

```
  (cond
   ((null poslist)
    t)
   ((member_string word (cdr (car poslist)))
    nil)
   (t
    (frisian_is_a_content_word word (cdr poslist)))))

(define (frisian_tosyl_brackets phones)
  "(frisian_tosyl_brackets phones)
Takes a list of phones containing - as syllable boundary.  Construct the
Festival bracket structure."
(let ((syl nil) (syls nil) (p phones) (stress 0))
    (while p
     (set! syl nil)
     (set! stress 0)
     (while (and p (not (eq? '- (car p))))
       (set! syl (cons (car p) syl))
       (if (string-matches (car p) ".*1")
           (set! stress 1))
       (set! p (cdr p)))
     (set! p (cdr p))  ;; skip the syllable separator
     (set! syls (cons (list (reverse syl) stress) syls)))
   (reverse syls)))

(provide '/u/jdijkstra/festival/lib/voices/frisian/an_diphone/festvox/frislex)
```

Bijlage 4 fristoken.scm

```
;;;  Tokenization rules for frisian
;;;
;;;  Particularly numbers and symbols.
;;;

(define (frisian_number name)
"(frisian_number name)
Convert a string of digits into a list of words saying the number."
  (if (string-matches name "0")
      (list "nul")
      (frisian_number_from_digits (symbolexplode name))))

(define (just_zeros digits)
"(just_zeros digits)
If this only contains 0s then we just do something different."
(cond
  ((not digits) t)
  ((string-equal "0" (car digits))
   (just_zeros (cdr digits)))
  (t nil)))

(define (frisian_number_from_digits digits)
  "(frisian_number_from_digits digits)
Takes a list of digits and converts it to a list of words
saying the number."
  (let ((l (length digits)))
    (cond
     ((equal? l 0)
      nil)
     ((string-equal (car digits) "0")
      (frisian_number_from_digits (cdr digits)))
     ((equal? l 1);; single digit
      (cond
       ((string-equal (car digits) "0") (list "nul"))
       ((string-equal (car digits) "1") (list "ien"))
       ((string-equal (car digits) "2") (list "twa"))
       ((string-equal (car digits) "3") (list "trije"))
       ((string-equal (car digits) "4") (list "fjouwer"))
       ((string-equal (car digits) "5") (list "fiif"))
       ((string-equal (car digits) "6") (list "seis"))
       ((string-equal (car digits) "7") (list "s^an"))
       ((string-equal (car digits) "8") (list "acht"))
       ((string-equal (car digits) "9") (list "njoggen"))
       ;; fill in the rest
       (t (list "equis"))));; $$$ what should say?
     ((equal? l 2);; less than 100
      (cond
       ((string-equal (car digits) "0");; 0x
            (frisian_number_from_digits (cdr digits)))

       ((string-equal (car digits) "1");; 1x
            (cond
             ((string-equal (car (cdr digits)) "0") (list "tsien"))
             ((string-equal (car (cdr digits)) "1") (list "alve"))
             ((string-equal (car (cdr digits)) "2") (list "tolve"))
             ((string-equal (car (cdr digits)) "3") (list "trettjin"))
             ((string-equal (car (cdr digits)) "4") (list "fjirtjin"))
             ((string-equal (car (cdr digits)) "5") (list "fyftjin"))
```

```
        ((string-equal (car (cdr digits)) "6") (list "sechtjin))
        ((string-equal (car (cdr digits)) "7") (list "santjin))
        (t
         (cons (frisian_number_from_digits (cdr digits)) "tjin" ))))

  ((string-equal (car digits) "2");; 2x
       (if (string-equal (car (cdr digits)) "0")
          (list "tweintich")
          (cons (frisian_number_from_digits (cdr digits)) "entweintich")))

  ((string-equal (car digits) "3");; 3x
       (if (string-equal (car (cdr digits)) "0")
          (list "tritich")
          (cons (frisian_number_from_digits (cdr digits)) "entritich")))

  ((string-equal (car digits) "4");; 4x
       (if (string-equal (car (cdr digits)) "0")
          (list "fjirtich")
          (cons (frisian_number_from_digits (cdr digits)) "enfjirtich")))

  ((string-equal (car digits) "5");; 5x
       (if (string-equal (car (cdr digits)) "0")
          (list "fyftich")
          (cons (frisian_number_from_digits (cdr digits)) "enfyftich")))

  ((string-equal (car digits) "6");; 6x
       (if (string-equal (car (cdr digits)) "0")
          (list "sechstich")
          (cons (frisian_number_from_digits (cdr digits)) "ensechstich")))

  ((string-equal (car digits) "7");; 7x
       (if (string-equal (car (cdr digits)) "0")
          (list "santich")
          (cons (frisian_number_from_digits (cdr digits)) "ensantich")))

  ((string-equal (car digits) "8");; 8x
       (if (string-equal (car (cdr digits)) "0")
          (list "tachtich")
          (cons (frisian_number_from_digits (cdr digits)) "entachtich")))

  ((string-equal (car digits) "9");; 9x
       (if (string-equal (car (cdr digits)) "0")
          (list "njoggentich")
          (cons (frisian_number_from_digits (cdr digits))"ennjoggentich")))

  ))

 ((equal? l 3);; in the hundreds
  (cond

  ((string-equal (car digits) "1");; 1xx
       (if (just_zeros (cdr digits)) (list "h^undert")
          (cons "h^undert" (frisian_number_from_digits (cdr digits)))))

  ((string-equal (car digits) "2");; 2xx
        (cons "twah^undert" (frisian_number_from_digits (cdr digits))))
  ((string-equal (car digits) "3");; 3xx
       (cons "trijeh^undert" (frisian_number_from_digits (cdr digits))))
  ((string-equal (car digits) "4");; 4xx
       (cons "fjouwerh^undert" (frisian_number_from_digits (cdr digits))))
```

```
     ((string-equal (car digits) "5");; 5xx
          (cons "fiifh^undert" (frisian_number_from_digits (cdr digits))))
     ((string-equal (car digits) "6");; 6xx
          (cons "seish^undert" (frisian_number_from_digits (cdr digits))))
     ((string-equal (car digits) "7");; 7xx
          (cons "s^anh^undert" (frisian_number_from_digits (cdr digits))))
     ((string-equal (car digits) "8");; 8xx
          (cons "achth^undert" (frisian_number_from_digits (cdr digits))))
     ((string-equal (car digits) "9");; 9xx
          (cons "njoggenh^undert" (frisian_number_from_digits (cdr digits))))

      (t;; ?xx
          (append (frisian_number_from_digits (list (car digits)))
                  (list "h^undert")
                  (frisian_number_from_digits (cdr digits))))
      ))

    ((< l 7)
     (let ((sub_thousands
              (list
               (car (cdr (cdr (reverse digits))))
               (car (cdr (reverse digits)))
               (car (reverse digits))))
             (thousands (reverse (cdr (cdr (cdr (reverse digits)))))))
          (set! x (frisian_number_from_digits thousands))
          (append
           (if (string-equal (car x) "ien") nil x)
           (list "t^uzen")
           (frisian_number_from_digits sub_thousands))))

    ((< l 13)
     (let ((sub_million
              (list
               (car (cdr (cdr (cdr (cdr (cdr(reverse digits)))))))
               (car (cdr (cdr (cdr (cdr (reverse digits))))))
               (car (cdr (cdr (cdr (reverse digits)))))
               (car (cdr (cdr (reverse digits))))
               (car (cdr (reverse digits)))
               (car (reverse digits))
               ))
             (millions (reverse (cdr (cdr (cdr (cdr (cdr (cdr (reverse digits)))))))))
          (set! x (frisian_number_from_digits millions))
          (append
           (if (string-equal (car x) "ien")
               (list "ien" "miljoen")
               (append x (list "miljoen")))
           (frisian_number_from_digits sub_million))))

    (t
     (list "in" "hiel" "grut" "getal")))))


(define (frisian_token_to_words token name)
  "(frisian_token_to_words TOKEN NAME)
Returns a list of words for the NAME from TOKEN.  This primarily
allows the treatment of numbers, money etc."
  (cond
   ((string-matches name "[1-9][0-9]+")
    (frisian_number name))
   ((not (lts.in.alphabet name 'frisian_downcase))
```

```
    ;; It contains some other than the lts can deal with
    (let ((subwords))
      (item.set_feat token "pos" "nn")
      (mapcar
       (lambda (letter)
            ;; might be symbols or digits
            (set! subwords
                 (append
                      subwords
                      (cond
                       ((string-matches letter "[0-9]")
                        (frisian_number letter))
                       ((string-matches letter "[A-ZÁÉÍÓÚÜÑ]")
                          (frisian_downcase letter))
                       (t
                        (list letter))))))
       (symbolexplode name))
      subwords))
    (t
     (list name))))

(provide '/u/jdijkstra/festival/lib/voices/frisian/an_diphone/festvox/fristoken)
```

Bijlage 5 frisint.scm

```
;;;  Using the general intonation module add flattened hat accents
;;;  of Accented syllables

(define (targ_func1 utt syl )
"funzioi bat"
  (let ((start (item.feat syl 'syllable_start))
        (end (item.feat syl 'syllable_end))
        (ulen (item.feat (utt.relation.last utt 'Segment ) 'segment_end))
        nstart nend fustart fuend fuend fstart fend)
        (set! nstart (/ start ulen))
        (set! nend (/ end ulen))
        (set! fustart '130)
        (set! fuend   '110)
        (set! fstart  (+ (* (- fuend fustart) nstart) fustart))
        (set! fend    (+ (* (- fuend fustart) nend) fustart))

    (cond
      ((equal? (item.feat syl "R:Intonation.daughter1.name") "Accented")
       (list
        (list start fstart)
        (list (+ start 0.010) (+ fstart 10 ))
        (list (- end   0.010) (+ fstart 8 ))
        (list end   fend)
            ))
      ((not (item.next syl))
       (list
            (list end fuend)))
      ((not (item.prev syl))
       (list
            (list start fustart)))
      (t
       nil))))


(provide '/u/jdijkstra/festival/lib/voices/frisian/an_diphone/festvox/frisint)
```

Bijlage 6 mrpa_phones.scm

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;;                Centre for Speech Technology Research                  ;;
;;                     University of Edinburgh, UK                       ;;
;;                       Copyright (c) 1996,1997                         ;;
;;                        All Rights Reserved.                           ;;
;;                                                                       ;;
;;  Permission is hereby granted, free of charge, to use and distribute  ;;
;;  this software and its documentation without restriction, including   ;;
;;  without limitation the rights to use, copy, modify, merge, publish,  ;;
;;  distribute, sublicense, and/or sell copies of this work, and to      ;;
;;  permit persons to whom this work is furnished to do so, subject to   ;;
;;  the following conditions:                                            ;;
;;   1. The code must retain the above copyright notice, this list of    ;;
;;      conditions and the following disclaimer.                         ;;
;;   2. Any modifications must be clearly marked as such.                ;;
;;   3. Original authors' names are not deleted.                         ;;
;;   4. The authors' names are not used to endorse or promote products   ;;
;;      derived from this software without specific prior written        ;;
;;      permission.                                                      ;;
;;                                                                       ;;
;;  THE UNIVERSITY OF EDINBURGH AND THE CONTRIBUTORS TO THIS WORK        ;;
;;  DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING      ;;
;;  ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT   ;;
;;  SHALL THE UNIVERSITY OF EDINBURGH NOR THE CONTRIBUTORS BE LIABLE     ;;
;;  FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES    ;;
;;  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN   ;;
;;  AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION,          ;;
;;  ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF       ;;
;;  THIS SOFTWARE.                                                       ;;
;;                                                                       ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;
;;  A definition of the mrpa phone set
;;

(defPhoneSet
frisian
;;;  Phone Features
(;; vowel or consonant
  (vc + -)
  ;; vowel length: short long dipthong schwa
  (vlng s l d a 0)
  ;; vowel height: high high-mid low-mid low
  (vheight 1 2 3 4 -)
  ;; vowel frontness: front mid back
  (vfront 1 2 3 -)
  ;; lip rounding
  (vrnd + -)
  ;; consonant type: stop fricative affricative nasal liquid
  (ctype s f a n l 0)
  ;; place of articulation: labial alveolar palatal labio-dental
  ;;                         dental velar
  (cplace l a p b d v 0)
  ;; consonant voicing
  (cvox + -)
  )
```

```
(
  (#  - 0 - - - 0 0 -)        ;; silence phoneme

  (a  + s 4 1 - 0 0 -)        ;; ta
  (E  + s 3 1 - 0 0 -)        ;; let
  (>  + s 3 3 + 0 0 -)        ;; kat
  (&  + s 2 2 - 0 0 -)        ;; de
  (I  + s 2 1 - 0 0 -)        ;; ik
  (Y  + s 2 1 + 0 0 -)        ;; nut
  (U  + s 2 3 + 0 0 -)        ;; rom
  (i  + s 1 1 - 0 0 -)        ;; dyk
  (y  + s 1 1 + 0 0 -)        ;; nút
  (u  + s 1 3 + 0 0 -)        ;; rûch

  (a: + l 4 1 - 0 0 -)        ;; baas
  (E: + l 3 1 - 0 0 -)        ;; bêd
  (8: + l 3 1 + 0 0 -)        ;; freule
  (>: + l 3 3 + 0 0 -)        ;; sâlt
  (e: + l 2 1 - 0 0 -)        ;; reed
  (7: + l 2 1 + 0 0 -)        ;; deun
  (U: + l 2 3 + 0 0 -)        ;; rook
  (i: + l 1 1 - 0 0 -)        ;; tiid
  (y: + l 1 1 + 0 0 -)        ;; drúf
  (u: + l 1 3 + 0 0 -)        ;; sûch

  (i& + d 1 1 - 0 0 -)        ;; biede
  (iu + d 1 2 - 0 0 -)        ;; ieu
  (I& + d 2 2 - 0 0 -)        ;; hea
  (Ei + d 2 1 - 0 0 -)        ;; rij
  (>i + d 3 2 - 0 0 -)        ;; laitsje
  (Au + d 4 1 + 0 0 -)        ;; gau
  (y& + d 1 1 + 0 0 -)        ;; flues
  (Y& + d 2 1 + 0 0 -)        ;; gleon
  (8y + d 3 1 + 0 0 -)        ;; bui
  (u& + d 1 3 + 0 0 -)        ;; goed
  (ui + d 1 2 + 0 0 -)        ;; bloei
  (Ui + d 2 2 + 0 0 -)        ;; floite
  (U& + d 2 3 + 0 0 -)        ;; boat
  (U>_i + d 3 2 + 0 0 -)      ;; moai

  (a1  + s 4 1 - 0 0 -)
  (E1  + s 3 1 - 0 0 -)
  (>1  + s 3 3 + 0 0 -)
  (I1  + s 2 1 - 0 0 -)
  (Y1  + s 2 1 + 0 0 -)
  (U1  + s 2 3 + 0 0 -)
  (i1  + s 1 1 - 0 0 -)
  (y1  + s 1 1 + 0 0 -)
  (u1  + s 1 3 + 0 0 -)

  (a:~ + l 4 1 - n 0 -)       ;; nasale vocalen
  (E:~ + l 3 1 - n 0 -)
  (8:~ + l 3 1 + n 0 -)
  (>:~ + l 3 3 + n 0 -)
  (e:~ + l 2 1 - n 0 -)
  (7:~ + l 2 1 + n 0 -)
  (U:~ + l 2 3 + n 0 -)
  (i:~ + l 1 1 - n 0 -)
  (y:~ + l 1 1 + n 0 -)
  (u:~ + l 1 3 + n 0 -)
```

```
  (i&~ + d 1 1 - n 0 -)
  (I&~ + d 2 2 - n 0 -)
  (>i~ + d 3 2 - n 0 -)
  (Au~ + d 4 1 + n 0 -)
  (y&~ + d 1 1 + n 0 -)
  (Y&~ + d 2 1 + n 0 -)
  (8y~ + d 3 1 + n 0 -)
  (u&~ + d 1 3 + n 0 -)
  (U&~ + d 2 3 + n 0 -)

  (p  - 0 - - - s l -)          ;; part
  (t  - 0 - - - s d -)          ;; ta
  (k  - 0 - - - s v -)          ;; klear
  (b  - 0 - - - s l +)          ;; bal
  (d  - 0 - - - s d +)          ;; daam
  (g  - 0 - - - s v +)          ;; goed

  (f  - 0 - - - f b -)          ;; fol
  (s  - 0 - - - f a -)          ;; stil
  (x  - 0 - - - f v -)          ;; rûch
  (v  - 0 - - - f b +)          ;; skevel
  (z  - 0 - - - f a +)          ;; wêze
  (G  - 0 - - - f v +)          ;; drage
  (V| - 0 - - - f b +)          ;; wetter
  (S  - 0 - - - f p -)          ;; lunch
  (Z  - 0 - - - f p +)          ;; rûzje

  (h  - 0 - - - a v -)          ;; heech

  (m  - 0 - - - n l +)          ;; laam
  (n  - 0 - - - n a +)          ;; noch
  (N  - 0 - - - n v +)          ;; bang

  (r  - 0 - - - l a +)          ;; raar
  (l  - 0 - - - l a +)          ;; slikje

  (w  - 0 - - - l l +)          ;; woansdei
  (j  - 0 - - - l p +)          ;; jas
)
)
(PhoneSet.silences '(#))

(provide '/u/jdijkstra/festival/lib/mrpa_phones)
```

Bijlage 7 mbrola.scm

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;                                                                       ;;
;;;                Centre for Speech Technology Research                  ;;
;;;                     University of Edinburgh, UK                       ;;
;;;                       Copyright (c) 1996,1997                         ;;
;;;                        All Rights Reserved.                           ;;
;;;                                                                       ;;
;;;  Permission is hereby granted, free of charge, to use and distribute  ;;
;;;  this software and its documentation without restriction, including  ;;
;;;  without limitation the rights to use, copy, modify, merge, publish,  ;;
;;;  distribute, sublicense, and/or sell copies of this work, and to      ;;
;;;  permit persons to whom this work is furnished to do so, subject to   ;;
;;;  the following conditions:                                            ;;
;;;   1. The code must retain the above copyright notice, this list of    ;;
;;;      conditions and the following disclaimer.                         ;;
;;;   2. Any modifications must be clearly marked as such.                ;;
;;;   3. Original authors' names are not deleted.                         ;;
;;;   4. The authors' names are not used to endorse or promote products   ;;
;;;      derived from this software without specific prior written        ;;
;;;      permission.                                                      ;;
;;;                                                                       ;;
;;;  THE UNIVERSITY OF EDINBURGH AND THE CONTRIBUTORS TO THIS WORK        ;;
;;;  DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING      ;;
;;;  ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT   ;;
;;;  SHALL THE UNIVERSITY OF EDINBURGH NOR THE CONTRIBUTORS BE LIABLE     ;;
;;;  FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES    ;;
;;;  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN   ;;
;;;  AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION,          ;;
;;;  ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF       ;;
;;;  THIS SOFTWARE.                                                       ;;
;;;                                                                       ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;  Support for MBROLA as an external module.
;;;

;;; You might want to set this in your sitevars.scm
;; (defvar mbrola_progname "/cstr/external/mbrola/mbrola"
;; "mbrola_progname
;;  The program name for mbrola.")

(defvar mbrola_progname
"/u/jdijkstra/festival/lib/voices/frisian/an_diphone/mbrola/mbrola-linux-i386"
        "mbrola_progname
        The program name for mbrola.")

(defvar mbrola_database
"/u/jdijkstra/festival/lib/voices/frisian/an_diphone/mbrola/fy1/temp.dat"
  "mbrola_database
The name of the MBROLA database to use during MBROLA Synthesis.")

(define (MBROLA_Synth utt)
  "(MBROLA_Synth UTT)
  Synthesize using MBROLA as external module.  Basically dump the info
  from this utterance. Call MBROLA and reload the waveform into utt.
  [see MBROLA]"
  (let ((filename (make_tmp_filename)))
    (save_segments_mbrola utt filename)
    (system (string-append mbrola_progname " "
```

```
                                mbrola_database " "
                                filename " "
                                filename ".au"))
    (utt.import.wave utt (string-append filename ".au"))
    (apply_hooks after_synth_hooks utt)
    (delete-file filename)
    (delete-file (string-append filename ".au"))
    utt))

(define (save_segments_mbrola utt filename)
  "(save_segments_mbrola UTT FILENAME)
  Save segment information in MBROLA format in filename.  The format is
  phone duration (ms) [% position F0 target]*. [see MBROLA]"
  (let ((fd (fopen filename "w")))
    (mapcar
     (lambda (segment)
       (save_seg_mbrola_entry
            (item.feat segment 'name)
            (item.feat segment 'segment_start)
            (item.feat segment 'segment_duration)
            (mapcar
             (lambda (targ_item)
               (list
                (item.feat targ_item "pos")
                (item.feat targ_item "f0")))
             (item.relation.daughters segment 'Target)) ;; list of targets
            fd))
     (utt.relation.items utt 'Segment))
    (fclose fd)))

(define (save_seg_mbrola_entry name start dur targs fd)
  "(save_seg_mbrola_entry ENTRY NAME START DUR TARGS FD)
  Entry contains, (name duration num_targs start 1st_targ_pos 1st_targ_val)."
  (format fd "%s %d " name (nint (* dur 1000)))
  (if targs     ;; if there are any targets
      (mapcar
       (lambda (targ) ;; targ_pos and targ_val
            (let ((targ_pos (car targ))
                  (targ_val (car (cdr targ))))

              (format fd "%d %d "
                      (nint (* 100 (/ (- targ_pos start) dur))) ;; % pos of target
                      (nint (parse-number targ_val)))           ;; target value
              ))
       targs))
  (terpri fd)
  (terpri fd)
)

(provide '/u/jdijkstra/festival/lib/mbrola)
```

Bijlage 8 mrpa_durs.scm

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;                                                                                                    ;;
;;;              Centre for Speech Technology Research                                                  ;;
;;;                   University of Edinburgh, UK                                                       ;;
;;;                     Copyright (c) 1996,1997                                                         ;;
;;;                      All Rights Reserved.                                                           ;;
;;;                                                                                                    ;;
;;;  Permission is hereby granted, free of charge, to use and distribute                               ;;
;;;  this software and its documentation without restriction, including                                ;;
;;;  without limitation the rights to use, copy, modify, merge, publish,                               ;;
;;;  distribute, sublicense, and/or sell copies of this work, and to                                   ;;
;;;  permit persons to whom this work is furnished to do so, subject to                                ;;
;;;  the following conditions:                                                                         ;;
;;;   1. The code must retain the above copyright notice, this list of                                 ;;
;;;      conditions and the following disclaimer.                                                      ;;
;;;   2. Any modifications must be clearly marked as such.                                             ;;
;;;   3. Original authors' names are not deleted.                                                      ;;
;;;   4. The authors' names are not used to endorse or promote products                                ;;
;;;      derived from this software without specific prior written                                     ;;
;;;      permission.                                                                                   ;;
;;;                                                                                                    ;;
;;;  THE UNIVERSITY OF EDINBURGH AND THE CONTRIBUTORS TO THIS WORK                                     ;;
;;;  DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING                                   ;;
;;;  ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT                                ;;
;;;  SHALL THE UNIVERSITY OF EDINBURGH NOR THE CONTRIBUTORS BE LIABLE                                  ;;
;;;  FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES                                 ;;
;;;  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN                                ;;
;;;  AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION,                                       ;;
;;;  ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF                                    ;;
;;;  THIS SOFTWARE.                                                                                    ;;
;;;                                                                                                    ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;;  mrpa average phoneme durations from gsw 450
;;;

(set! phoneme_durations
'(

 (# 0.250)
 (a 0.040)
 (E 0.050)
 (> 0.040)
 (& 0.040)
 (I 0.040)
 (Y 0.040)
 (U 0.040)
 (i 0.050)
 (y 0.040)
 (u 0.040)

 (a: 0.080)
 (E: 0.090)
 (8: 0.080)
 (>: 0.090)
 (e: 0.080)
 (7: 0.080)
 (U: 0.080)
```

```
  (i: 0.070)
  (y: 0.080)
  (u: 0.070)

  (a1 0.050)
  (E1 0.060)
  (>1 0.050)
  (&1 0.050)
  (I1 0.050)
  (Y1 0.050)
  (U1 0.050)
  (i1 0.060)
  (y1 0.050)
  (u1 0.050)

  (i& 0.090)
  (iu 0.090)
  (I& 0.090)
  (Ei 0.090)
  (>i 0.090)
  (Au 0.090)
  (y& 0.090)
  (Y& 0.090)
  (8y 0.090)
  (u& 0.090)
  (ui 0.090)
  (Ui 0.090)
  (U& 0.090)
  (U&_i 0.100)

  (b 0.065)
  (d 0.060)
  (f 0.100)
  (g 0.080)
  (j 0.100)
  (k 0.100)
  (l 0.080)
  (m 0.070)
  (n 0.080)
  (N 0.110)
  (p 0.100)
  (r 0.080)
  (s 0.110)
  (t 0.085)
  (x 0.130)
  (v 0.100)
  (z 0.110)
  (G 0.130)
  (V[ 0.100)
  (S 0.110)
  (Z 0.110)
  (h 0.080)
  (w 0.100)
  (j 0.100)
))

(set! gsw_durs
'(
  (# 0.225 0.130)
  (a 0.045 0.024)
```

```
  (E 0.055 0.030)
  (> 0.045 0.030)
  (& 0.045 0.030)
  (I 0.045 0.030)
  (Y 0.045 0.030)
  (U 0.045 0.030)
  (i 0.055 0.040)
  (y 0.045 0.030)
  (u 0.045 0.030)

  (a: 0.085 0.070)
  (E: 0.095 0.080)
  (8: 0.085 0.070)
  (>: 0.095 0.080)
  (e: 0.085 0.070)
  (7: 0.085 0.070)
  (U: 0.085 0.070)
  (i: 0.075 0.060)
  (y: 0.085 0.070)
  (u: 0.075 0.060)

  (b 0.070 0.055)
  (d 0.065 0.050)
  (f 0.105 0.090)
  (g 0.085 0.070)
  (j 0.105 0.090)
  (k 0.105 0.090)
  (l 0.085 0.070)
  (m 0.075 0.060)
  (n 0.085 0.070)
  (N 0.115 0.100)
  (p 0.105 0.090)
  (r 0.085 0.070)
  (s 0.115 0.100)
  (t 0.090 0.075)
  (x 0.135 0.120)
  (v 0.105 0.090)
  (z 0.115 0.100)
  (G 0.135 0.120)
  (V[ 0.105 0.090)
  (S 0.115 0.100)
  (Z 0.115 0.100)
  (h 0.085 0.070)
  (w 0.105 0.090)
  (j 0.105 0.090)

))

(provide 'mrpa_durs)
```

Bijlage 9 gswdurtreeZ.scm

```
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;                                                                       ;;
;;;                Centre for Speech Technology Research                  ;;
;;;                     University of Edinburgh, UK                       ;;
;;;                       Copyright (c) 1996,1997                         ;;
;;;                        All Rights Reserved.                           ;;
;;;                                                                       ;;
;;;  Permission is hereby granted, free of charge, to use and distribute  ;;
;;;  this software and its documentation without restriction, including   ;;
;;;  without limitation the rights to use, copy, modify, merge, publish,  ;;
;;;  distribute, sublicense, and/or sell copies of this work, and to      ;;
;;;  permit persons to whom this work is furnished to do so, subject to   ;;
;;;  the following conditions:                                            ;;
;;;   1. The code must retain the above copyright notice, this list of    ;;
;;;      conditions and the following disclaimer.                         ;;
;;;   2. Any modifications must be clearly marked as such.                ;;
;;;   3. Original authors' names are not deleted.                         ;;
;;;   4. The authors' names are not used to endorse or promote products   ;;
;;;      derived from this software without specific prior written        ;;
;;;      permission.                                                      ;;
;;;                                                                       ;;
;;;  THE UNIVERSITY OF EDINBURGH AND THE CONTRIBUTORS TO THIS WORK        ;;
;;;  DISCLAIM ALL WARRANTIES WITH REGARD TO THIS SOFTWARE, INCLUDING      ;;
;;;  ALL IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS, IN NO EVENT   ;;
;;;  SHALL THE UNIVERSITY OF EDINBURGH NOR THE CONTRIBUTORS BE LIABLE     ;;
;;;  FOR ANY SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES OR ANY DAMAGES    ;;
;;;  WHATSOEVER RESULTING FROM LOSS OF USE, DATA OR PROFITS, WHETHER IN   ;;
;;;  AN ACTION OF CONTRACT, NEGLIGENCE OR OTHER TORTIOUS ACTION,          ;;
;;;  ARISING OUT OF OR IN CONNECTION WITH THE USE OR PERFORMANCE OF       ;;
;;;  THIS SOFTWARE.                                                       ;;
;;;                                                                       ;;
;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;;
;;;
;;;   A tree to predict zcore durations build from gsw 450 (timit)
;;;   doesn't use actual phonemes so it can have better generalizations
;;;

;; pre Sue's changes to mrpa_phones (on traing data)
;; RMSE 0.79102 Correlation is 0.610184 Mean (abs) Error 0.605081 (0.509517)
;; Post with balance
;; train test spit --stop 19 --balance 16
;; RMSE 0.841861 Correlation is 0.526064 Mean (abs) Error 0.646614 (0.539288)
;; on training data
;; RMSE 0.784032 Correlation is 0.619165 Mean (abs) Error 0.602819 (0.501332)
;;
;; Oct 29th 1997
;; stepwise (but its over trained)
;; RMSE 0.8322 Correlation is 0.5286 Mean (abs) Error 0.6375 (0.5350)
;;
;; May 11th 1998
;; new architecture, full new train on f2b on test data
;; in zscore domain
;; RMSE 0.8076 Correlation is 0.5307 Mean (abs) Error 0.6113 (0.5278)
;; in absolute domain
;; RMSE 0.0276 Correlation 0.7468 Mean (abs) error 0.0203 (0.0187)
;;
;; May 18th 1998
;; various corrections f2bdur.bbz.H0.S50.tree no names zscore
```

```
;; in zscore domain
;; RMSE 0.8049 Correlation is 0.6003 Mean (abs) Error 0.6008 (0.5357)
;; in absolute domain
;; RMSE 0.0268 Correlation 0.7766 Mean (abs) error 0.0196 (0.0183)

(set! gsw_duration_cart_tree
'
((name is #)
((emph_sil is +)
((0.0 -0.5))
((p.R:SylStructure.parent.parent.pbreak is BB)
  ((0.0 2.0))
  ((0.0 0.0))))

((R:SylStructure.parent.accented is 0)
((n.ph_ctype is 0)
((p.ph_vlng is 0)
  ((R:SylStructure.parent.syl_codasize < 1.5)
   ((p.ph_ctype is n)
    ((ph_ctype is f)
     ((0.559208 -0.783163))
     ((1.05215 -0.222704)))
    ((ph_ctype is s)
     ((R:SylStructure.parent.syl_break is 2)
      ((0.589948 0.764459))
      ((R:SylStructure.parent.asyl_in < 0.7)
       ((1.06385 0.567944))
       ((0.691943 0.0530272))))
     ((ph_vlng is l)
      ((pp.ph_vfront is 1)
       ((1.06991 0.766486))
       ((R:SylStructure.parent.syl_break is 1)
        ((0.69665 0.279248))
        ((0.670353 0.0567774))))
      ((p.ph_ctype is s)
       ((seg_onsetcoda is coda)
        ((0.828638 -0.038356))
        ((ph_ctype is f)
         ((0.7631 -0.545853))
         ((0.49329 -0.765994))))
       ((R:SylStructure.parent.parent.gpos is det)
        ((R:SylStructure.parent.last_accent < 0.3)
         ((R:SylStructure.parent.sub_phrases < 1)
          ((0.811686 0.160195))
          ((0.799015 0.713958)))
         ((0.731599 -0.215472)))
        ((ph_ctype is r)
         ((0.673487 0.092772))
         ((R:SylStructure.parent.asyl_in < 1)
          ((0.745273 0.00132813))
          ((0.75457 -0.334898)))))))))
   ((pos_in_syl < 0.5)
    ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
     ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
      ((0.902446 -0.041618))
      ((R:SylStructure.parent.sub_phrases < 2.3)
       ((0.900629 0.262952))
       ((1.18474 0.594794))))
     ((seg_onset_stop is 0)
      ((R:SylStructure.parent.position_type is mid)
```

```
      ((0.512323 -0.760444))
      ((R:SylStructure.parent.syl_out < 6.8)
       ((pp.ph_vlng is a)
        ((0.640575 -0.450449))
        ((ph_ctype is f)
         ((R:SylStructure.parent.sub_phrases < 1.3)
          ((0.862876 -0.296956))
          ((R:SylStructure.parent.syl_out < 2.4)
           ((0.803215 0.0422868))
           ((0.877856 -0.154465))))
         ((R:SylStructure.parent.syl_out < 3.6)
          ((R:SylStructure.parent.syl_out < 1.2)
           ((0.567081 -0.264199))
           ((0.598043 -0.541738)))
          ((0.676843 -0.166623)))))
       ((0.691678 -0.57173))))
     ((R:SylStructure.parent.parent.gpos is cc)
      ((1.15995 0.313289))
      ((pp.ph_vfront is 1)
       ((0.555993 0.0695819))
       ((R:SylStructure.parent.asyl_in < 1.2)
        ((R:SylStructure.parent.sub_phrases < 2.7)
         ((0.721635 -0.367088))
         ((0.71919 -0.194887)))
        ((0.547052 -0.0637491)))))))
   ((ph_ctype is s)
    ((R:SylStructure.parent.syl_break is 0)
     ((R:SylStructure.parent.R:Syllable.p.syl_break is 1)
      ((0.650007 -0.333421))
      ((0.846301 -0.165383)))
     ((0.527756 -0.516332)))
    ((R:SylStructure.parent.syl_break is 0)
     ((p.ph_ctype is s)
      ((0.504414 -0.779112))
      ((0.812498 -0.337611)))
     ((pos_in_syl < 1.4)
      ((0.513041 -0.745807))
      ((p.ph_ctype is s)
       ((0.350582 -1.04907))
       ((0.362 -0.914974)))))))
 ((R:SylStructure.parent.syl_break is 0)
  ((ph_ctype is n)
   ((R:SylStructure.parent.position_type is initial)
    ((pos_in_syl < 1.2)
     ((0.580485 0.172658))
     ((0.630973 -0.101423)))
    ((0.577937 -0.360092)))
   ((R:SylStructure.parent.syl_out < 2.9)
    ((R:SylStructure.parent.syl_out < 1.1)
     ((R:SylStructure.parent.position_type is initial)
      ((0.896092 0.764189))
      ((R:SylStructure.parent.sub_phrases < 3.6)
       ((ph_ctype is s)
        ((0.877362 0.555132))
        ((0.604511 0.369882)))
       ((0.799982 0.666966))))
     ((seg_onsetcoda is coda)
      ((p.ph_vlng is a)
       ((R:SylStructure.parent.last_accent < 0.4)
        ((0.800736 0.240634))
```

```
          ((0.720606 0.486176)))
         ((1.18173 0.573811)))
        ((0.607147 0.194468))))
      ((ph_ctype is r)
       ((0.88377 0.499383))
       ((R:SylStructure.parent.last_accent < 0.5)
        ((R:SylStructure.parent.position_type is initial)
         ((R:SylStructure.parent.parent.word_numsyls < 2.4)
          ((0.62798 0.0737318))
          ((0.787334 0.331014)))
         ((ph_ctype is s)
          ((0.808368 0.0929299))
          ((0.527948 -0.0443271))))
        ((seg_coda_fric is 0)
         ((p.ph_vlng is a)
          ((0.679745 0.517681))
          ((R:SylStructure.parent.sub_phrases < 1.1)
           ((0.759979 0.128316))
           ((0.775233 0.361383))))
         ((R:SylStructure.parent.last_accent < 1.3)
          ((0.696255 0.054136))
          ((0.632425 0.246742)))))))
    ((pos_in_syl < 0.3)
     ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
      ((0.847602 0.621547))
      ((ph_ctype is s)
       ((0.880645 0.501679))
       ((R:SylStructure.parent.sub_phrases < 3.3)
        ((R:SylStructure.parent.sub_phrases < 0.3)
         ((0.901014 -0.042049))
         ((0.657493 0.183226)))
        ((0.680126 0.284799)))))
     ((ph_ctype is s)
      ((p.ph_vlng is s)
       ((0.670033 -0.820934))
       ((0.863306 -0.348735)))
      ((ph_ctype is n)
       ((R:SylStructure.parent.asyl_in < 1.2)
        ((0.656966 -0.40092))
        ((0.530966 -0.639366)))
       ((seg_coda_fric is 0)
        ((1.04153 0.364857))
        ((pos_in_syl < 1.2)
         ((R:SylStructure.parent.syl_out < 3.4)
          ((0.81503 -0.00768613))
          ((0.602665 -0.197753)))
         ((0.601844 -0.394632)))))))))
((n.ph_ctype is f)
  ((pos_in_syl < 1.5)
   ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
    ((pos_in_syl < 0.1)
     ((1.63863 0.938841))
     ((R:SylStructure.parent.position_type is initial)
      ((0.897722 -0.0796637))
      ((nn.ph_vheight is 0)
       ((0.781081 0.480026))
       ((0.779711 0.127175)))))
    ((ph_ctype is r)
     ((p.ph_ctype is s)
      ((0.581329 -0.708767))
```

```
    ((0.564366 -0.236212)))
   ((ph_vlng is a)
    ((p.ph_ctype is r)
     ((0.70992 -0.273389))
     ((R:SylStructure.parent.parent.gpos is in)
      ((0.764696 0.0581338))
      ((nn.ph_vheight is 0)
       ((0.977737 0.721904))
       ((R:SylStructure.parent.sub_phrases < 2.2)
        ((pp.ph_vfront is 0)
         ((0.586708 0.0161206))
         ((0.619949 0.227372)))
        ((0.707285 0.445569))))))
    ((ph_ctype is n)
     ((R:SylStructure.parent.syl_break is 1)
      ((nn.ph_vfront is 2)
       ((0.430295 -0.120097))
       ((0.741371 0.219042)))
      ((0.587492 0.321245)))
     ((p.ph_ctype is n)
      ((0.871586 0.134075))
      ((p.ph_ctype is r)
       ((0.490751 -0.466418))
       ((R:SylStructure.parent.syl_codasize < 1.3)
        ((R:SylStructure.parent.sub_phrases < 2.2)
         ((p.ph_ctype is s)
          ((0.407452 -0.425925))
          ((0.644771 -0.542809)))
         ((0.688772 -0.201899)))
        ((ph_vheight is 1)
         ((nn.ph_vheight is 0)
          ((0.692018 0.209018))
          ((0.751345 -0.178136)))
         ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.3)
          ((R:SylStructure.parent.asyl_in < 1.5)
           ((0.599633 -0.235593))
           ((0.60042 0.126118)))
          ((p.ph_vlng is a)
           ((0.7148 -0.174812))
           ((R:SylStructure.parent.parent.gpos is content)
            ((0.761296 -0.231509))
            ((0.813081 -0.536405)))))))))))))
  ((ph_ctype is n)
   ((0.898844 0.163343))
   ((p.ph_vlng is s)
    ((seg_coda_fric is 0)
     ((0.752921 -0.45528))
     ((0.890079 -0.0998025)))
    ((ph_ctype is f)
     ((0.729376 -0.930547))
     ((ph_ctype is s)
      ((R:SylStructure.parent.R:Syllable.p.syl_break is 0)
       ((0.745052 -0.634119))
       ((0.521502 -0.760176)))
      ((R:SylStructure.parent.syl_break is 1)
       ((0.766575 -0.121355))
       ((0.795616 -0.557509)))))))))
 ((p.ph_vlng is 0)
  ((p.ph_ctype is r)
   ((ph_vlng is 0)
```

```
 ((0.733659 -0.402734))
 ((R:SylStructure.parent.sub_phrases < 1.5)
  ((ph_vlng is s)
   ((0.326176 -0.988478))
   ((n.ph_ctype is s)
    ((0.276471 -0.802536))
    ((0.438283 -0.900628))))
  ((nn.ph_vheight is 0)
   ((ph_vheight is 2)
    ((0.521 -0.768992))
    ((0.615436 -0.574918)))
   ((ph_vheight is 1)
    ((0.387376 -0.756359))
    ((pos_in_syl < 0.3)
     ((0.417235 -0.808937))
     ((0.384043 -0.93315)))))))
((ph_vlng is a)
 ((ph_ctype is 0)
  ((n.ph_ctype is s)
   ((p.ph_ctype is f)
    ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
     ((0.415908 -0.428493))
     ((pos_in_syl < 0.1)
      ((0.790441 0.0211071))
      ((0.452465 -0.254485))))
    ((p.ph_ctype is s)
     ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
      ((0.582447 -0.389966))
      ((0.757648 0.185781)))
     ((R:SylStructure.parent.sub_phrases < 1.4)
      ((0.628965 0.422551))
      ((0.713613 0.145576)))))
   ((seg_onset_stop is 0)
    ((R:SylStructure.parent.R:Syllable.p.syl_break is 0)
     ((pp.ph_vfront is 1)
      ((0.412363 -0.62319))
      ((R:SylStructure.parent.syl_out < 3.6)
       ((0.729259 -0.317324))
       ((0.441633 -0.591051))))
     ((R:SylStructure.parent.syl_break is 1)
      ((R:SylStructure.parent.sub_phrases < 2.7)
       ((0.457728 -0.405607))
       ((0.532411 -0.313148)))
      ((R:SylStructure.parent.last_accent < 0.3)
       ((1.14175 0.159416))
       ((0.616396 -0.254651)))))
    ((R:SylStructure.parent.position_type is initial)
     ((0.264181 -0.799896))
     ((0.439801 -0.551309)))))
  ((R:SylStructure.parent.position_type is final)
   ((0.552027 -0.707084))
   ((0.585661 -0.901874))))
 ((ph_ctype is s)
  ((pos_in_syl < 1.2)
   ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
    ((pp.ph_vfront is 1)
     ((0.607449 0.196466))
     ((0.599662 0.00382414)))
    ((0.64109 -0.12859)))
   ((pp.ph_vfront is 1)
```

```
  ((0.720484 -0.219339))
  ((0.688707 -0.516734))))
 ((ph_vlng is s)
  ((n.ph_ctype is s)
   ((R:SylStructure.parent.parent.gpos is content)
    ((R:SylStructure.parent.position_type is single)
     ((0.659206 0.159445))
     ((R:SylStructure.parent.parent.word_numsyls < 3.5)
      ((R:SylStructure.parent.sub_phrases < 2)
       ((0.447186 -0.419103))
       ((0.631822 -0.0928561)))
      ((0.451623 -0.576116))))
    ((ph_vheight is 3)
     ((0.578626 -0.64583))
     ((0.56636 -0.4665))))
   ((R:SylStructure.parent.parent.gpos is in)
    ((0.771516 -0.217292))
    ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
     ((0.688571 -0.304382))
     ((R:SylStructure.parent.parent.gpos is content)
      ((R:SylStructure.parent.R:Syllable.p.syl_break is 1)
       ((n.ph_ctype is n)
        ((0.556085 -0.572203))
        ((0.820173 -0.240338)))
       ((R:SylStructure.parent.parent.word_numsyls < 2.2)
        ((0.595398 -0.588171))
        ((0.524737 -0.95797))))
      ((R:SylStructure.parent.sub_phrases < 3.9)
       ((0.371492 -0.959427))
       ((0.440479 -0.845747)))))))
  ((R:SylStructure.parent.R:Syllable.p.syl_break is 0)
   ((p.ph_ctype is f)
    ((0.524088 -0.482247))
    ((nn.ph_vheight is 1)
     ((0.587666 -0.632362))
     ((ph_vlng is l)
      ((R:SylStructure.parent.position_type is final)
       ((0.513286 -0.713117))
       ((0.604613 -0.924308)))
      ((R:SylStructure.parent.syl_codasize < 2.2)
       ((0.577997 -0.891342))
       ((0.659804 -1.15252))))))
   ((pp.ph_vlng is s)
    ((ph_ctype is f)
     ((0.813383 -0.599624))
     ((0.984027 -0.0771909)))
    ((p.ph_ctype is f)
     ((R:SylStructure.parent.parent.gpos is in)
      ((R:SylStructure.parent.R:Syllable.p.syl_break is 1)
       ((0.313572 -1.03242))
       ((0.525854 -0.542799)))
      ((R:SylStructure.parent.syl_out < 2.8)
       ((0.613007 -0.423979))
       ((0.570258 -0.766379))))
     ((R:SylStructure.parent.syl_break is 1)
      ((R:SylStructure.parent.parent.gpos is to)
       ((0.364585 -0.792895))
       ((ph_vlng is l)
        ((0.69143 -0.276816))
        ((0.65673 -0.523721))))
```

```
        ((R:SylStructure.parent.syl_out < 3.6)
         ((R:SylStructure.parent.position_type is initial)
          ((0.682096 -0.488102))
          ((0.406364 -0.731758)))
         ((0.584694 -0.822229)))))))))))
((n.ph_ctype is r)
 ((R:SylStructure.parent.position_type is initial)
  ((p.ph_vlng is a)
   ((0.797058 1.02334))
   ((ph_ctype is s)
    ((1.0548 0.536277))
    ((0.817253 0.138201))))
  ((R:SylStructure.parent.sub_phrases < 1.1)
   ((R:SylStructure.parent.syl_out < 3.3)
    ((0.884574 -0.23471))
    ((0.772063 -0.525292)))
   ((nn.ph_vfront is 1)
    ((1.25254 0.417485))
    ((0.955557 -0.0781996)))))
 ((pp.ph_vfront is 0)
  ((ph_ctype is f)
   ((n.ph_ctype is s)
    ((R:SylStructure.parent.parent.gpos is content)
     ((R:SylStructure.parent.R:Syllable.p.syl_break is 0)
      ((0.583506 -0.56941))
      ((0.525949 -0.289362)))
     ((0.749316 -0.0921038)))
    ((p.ph_vlng is s)
     ((0.734234 0.139463))
     ((0.680119 -0.0708717))))
   ((ph_vlng is s)
    ((ph_vheight is 1)
     ((0.908712 -0.618971))
     ((0.55344 -0.840495)))
    ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 1.2)
     ((pos_in_syl < 1.2)
      ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
       ((0.838715 0.00913392))
       ((R:SylStructure.parent.R:Syllable.p.syl_break is 1)
        ((ph_vheight is 2)
         ((0.555513 -0.512523))
         ((R:SylStructure.parent.position_type is initial)
          ((0.758711 0.121704))
          ((0.737555 -0.25637))))
        ((R:SylStructure.parent.syl_out < 3.1)
         ((n.ph_ctype is s)
          ((0.611756 -0.474522))
          ((1.05437 -0.247206)))
         ((R:SylStructure.parent.syl_codasize < 2.2)
          ((R:SylStructure.parent.position_type is final)
           ((0.567761 -0.597866))
           ((0.785599 -0.407765)))
          ((0.575598 -0.741256))))))
      ((ph_ctype is s)
       ((n.ph_ctype is s)
        ((0.661069 -1.08426))
        ((0.783184 -0.39789)))
       ((R:SylStructure.parent.R:Syllable.p.syl_break is 1)
        ((R:SylStructure.parent.sub_phrases < 2.6)
         ((0.511323 -0.666011))
```

```
            ((0.691878 -0.499492)))
           ((ph_ctype is r)
            ((0.482131 -0.253186))
            ((0.852955 -0.372832))))))
        ((0.854447 -0.0936489)))))
     ((R:SylStructure.parent.position_type is final)
      ((0.685939 -0.249982))
      ((R:SylStructure.parent.syl_out < 3.2)
       ((0.989843 0.18086))
       ((0.686805 -0.0402908)))))))))
((R:SylStructure.parent.syl_out < 2.4)
((R:SylStructure.parent.syl_out < 0.2)
  ((seg_onsetcoda is coda)
   ((ph_ctype is s)
    ((R:SylStructure.parent.syl_break is 4)
     ((pp.ph_vlng is 0)
      ((0.959737 1.63203))
      ((1.20714 0.994933)))
     ((n.ph_ctype is 0)
      ((R:SylStructure.parent.syl_break is 2)
       ((0.864809 0.214457))
       ((0.874278 0.730381)))
      ((pp.ph_vfront is 0)
       ((seg_coda_fric is 0)
        ((1.20844 -0.336221))
        ((1.01357 0.468302)))
       ((0.658106 -0.799121)))))
    ((n.ph_ctype is f)
     ((ph_ctype is f)
      ((1.26332 0.0300613))
      ((ph_vlng is d)
       ((1.02719 1.1649))
       ((ph_ctype is 0)
        ((R:SylStructure.parent.asyl_in < 1.2)
         ((1.14048 2.2668))
         ((ph_vheight is 1)
          ((1.15528 1.50375))
          ((1.42406 2.07927))))
        ((R:SylStructure.parent.sub_phrases < 1.1)
         ((0.955892 1.10243))
         ((R:SylStructure.parent.syl_break is 2)
          ((1.32682 1.8432))
          ((1.27582 1.59853)))))))
     ((n.ph_ctype is 0)
      ((ph_ctype is n)
       ((R:SylStructure.parent.syl_break is 2)
        ((1.45399 1.12927))
        ((1.05543 0.442376)))
       ((R:SylStructure.parent.syl_break is 4)
        ((R:SylStructure.parent.position_type is final)
         ((ph_ctype is f)
          ((1.46434 1.76508))
          ((0.978055 0.7486)))
         ((1.2395 2.30826)))
        ((ph_ctype is 0)
         ((0.935325 1.69917))
         ((nn.ph_vfront is 1)
          ((1.20456 1.31128))
          ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
           ((nn.ph_vheight is 0)
```

```
        ((1.16907 0.212421))
        ((0.952091 0.653094)))
       ((p.ph_ctype is 0)
        ((1.05502 1.25802))
        ((0.818731 0.777568))))))))
  ((ph_ctype is f)
   ((p.ph_ctype is 0)
    ((1.03918 0.163941))
    ((0.737545 -0.167063)))
   ((R:SylStructure.parent.position_type is final)
    ((n.ph_ctype is n)
     ((R:SylStructure.parent.last_accent < 0.5)
      ((R:SylStructure.parent.sub_phrases < 2.8)
       ((0.826207 -0.000859005))
       ((0.871119 0.273433)))
      ((R:SylStructure.parent.parent.word_numsyls < 2.4)
       ((1.17405 1.05694))
       ((0.858394 0.244916))))
     ((R:SylStructure.parent.syl_codasize < 2.2)
      ((p.ph_ctype is 0)
       ((1.14092 1.21187))
       ((R:SylStructure.parent.syl_break is 2)
        ((1.02653 0.59865))
        ((0.94248 1.1634))))
      ((seg_coda_fric is 0)
       ((1.07441 0.292935))
       ((1.15736 0.92574)))))
    ((ph_vlng is s)
     ((R:SylStructure.parent.syl_break is 2)
      ((1.34638 1.23484))
      ((0.951514 2.02008)))
     ((ph_ctype is 0)
      ((p.ph_ctype is r)
       ((0.806106 0.697089))
       ((R:SylStructure.parent.syl_break is 2)
        ((1.10891 0.992197))
        ((1.04657 1.51093))))
      ((1.18165 0.520952)))))))))
((p.ph_vlng is 0)
 ((pos_in_syl < 0.7)
  ((R:SylStructure.parent.position_type is final)
   ((ph_ctype is r)
    ((0.966357 0.185827))
    ((ph_ctype is s)
     ((0.647163 0.0332298))
     ((0.692972 -0.534917))))
   ((ph_ctype is s)
    ((0.881521 0.575107))
    ((p.ph_ctype is f)
     ((0.8223 -0.111275))
     ((R:SylStructure.parent.last_accent < 0.3)
      ((0.969188 0.09447))
      ((0.894438 0.381947))))))
  ((p.ph_ctype is f)
   ((0.479748 -0.490108))
   ((0.813125 -0.201268))))
 ((ph_ctype is s)
  ((0.908566 1.20397))
  ((R:SylStructure.parent.last_accent < 1.2)
   ((0.88078 0.636568))
```

```
    ((0.978087 1.07763))))))
((pos_in_syl < 1.3)
 ((R:SylStructure.parent.syl_break is 0)
  ((pos_in_syl < 0.1)
   ((R:SylStructure.parent.position_type is initial)
    ((p.ph_ctype is n)
     ((0.801651 -0.0163359))
     ((ph_ctype is s)
      ((n.ph_ctype is r)
       ((0.893307 1.07253))
       ((p.ph_vlng is 0)
        ((0.92651 0.525806))
        ((0.652444 0.952792))))
      ((p.ph_vlng is 0)
       ((seg_onsetcoda is coda)
        ((0.820151 0.469117))
        ((p.ph_ctype is f)
         ((0.747972 -0.0716448))
         ((ph_ctype is f)
          ((0.770882 0.457137))
          ((0.840905 0.102492)))))
       ((R:SylStructure.parent.syl_out < 1.1)
        ((0.667824 0.697337))
        ((0.737967 0.375114))))))
    ((ph_vheight is 1)
     ((0.624353 0.410671))
     ((R:SylStructure.parent.asyl_in < 0.8)
      ((0.647905 -0.331055))
      ((p.ph_ctype is s)
       ((0.629039 -0.240616))
       ((0.749277 -0.0191273))))))
    ((ph_vheight is 3)
     ((p.ph_ctype is s)
      ((0.626922 0.556537))
      ((0.789357 0.153892)))
     ((seg_onsetcoda is coda)
      ((n.ph_ctype is 0)
       ((R:SylStructure.parent.parent.word_numsyls < 3.4)
        ((0.744714 0.123242))
        ((0.742039 0.295753)))
       ((seg_coda_fric is 0)
        ((R:SylStructure.parent.parent.word_numsyls < 2.4)
         ((ph_vheight is 1)
          ((0.549715 -0.341018))
          ((0.573641 -0.00893114)))
         ((nn.ph_vfront is 2)
          ((0.67099 -0.744625))
          ((0.664438 -0.302803))))
        ((p.ph_vlng is 0)
         ((0.630028 0.113815))
         ((0.632794 -0.128733)))))
      ((ph_ctype is r)
       ((0.367169 -0.854509))
       ((0.94334 -0.216179))))))
   ((n.ph_ctype is f)
    ((ph_vlng is 0)
     ((1.3089 0.46195))
     ((R:SylStructure.parent.syl_codasize < 1.3)
      ((1.07673 0.657169))
      ((pp.ph_vlng is 0)
```

```
    ((0.972319 1.08222))
    ((1.00038 1.46257)))))
  ((p.ph_vlng is l)
   ((1.03617 0.785204))
   ((p.ph_vlng is a)
    ((R:SylStructure.parent.position_type is final)
     ((1.00681 0.321168))
     ((0.928115 0.950834)))
    ((ph_vlng is 0)
     ((pos_in_syl < 0.1)
      ((R:SylStructure.parent.position_type is final)
       ((0.863682 -0.167374))
       ((nn.ph_vheight is 0)
        ((p.ph_ctype is f)
         ((0.773591 -0.00374425))
         ((R:SylStructure.parent.syl_out < 1.1)
          ((0.951802 0.228448))
          ((1.02282 0.504252))))
        ((1.09721 0.736476))))
      ((R:SylStructure.parent.position_type is final)
       ((1.04302 0.0590974))
       ((0.589208 -0.431535))))
     ((n.ph_ctype is 0)
      ((1.27879 1.00642))
      ((ph_vlng is s)
       ((R:SylStructure.parent.asyl_in < 1.4)
        ((0.935787 0.481652))
        ((0.9887 0.749861)))
       ((R:SylStructure.parent.syl_out < 1.1)
        ((R:SylStructure.parent.position_type is final)
         ((0.921307 0.0696307))
         ((0.83675 0.552212)))
        ((0.810076 -0.0479225)))))))))
 ((ph_ctype is s)
  ((n.ph_ctype is s)
   ((0.706959 -1.0609))
   ((p.ph_ctype is n)
    ((0.850614 -0.59933))
    ((n.ph_ctype is r)
     ((0.665947 0.00698725))
     ((n.ph_ctype is 0)
      ((R:SylStructure.parent.position_type is initial)
       ((0.762889 -0.0649044))
       ((0.723956 -0.248899)))
      ((R:SylStructure.parent.sub_phrases < 1.4)
       ((0.632957 -0.601987))
       ((0.889114 -0.302401)))))))
  ((ph_ctype is f)
   ((R:SylStructure.parent.syl_codasize < 2.2)
    ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
     ((R:SylStructure.parent.syl_out < 1.1)
      ((0.865267 0.164636))
      ((0.581827 -0.0989051)))
     ((nn.ph_vfront is 2)
      ((0.684459 -0.316836))
      ((0.778854 -0.0961191))))
    ((R:SylStructure.parent.syl_out < 1.1)
     ((p.ph_ctype is s)
      ((0.837964 -0.429437))
      ((0.875304 -0.0652743)))
```

```
       ((0.611071 -0.635089))))
      ((p.ph_ctype is r)
       ((R:SylStructure.parent.syl_out < 1.1)
        ((0.762012 0.0139361))
        ((0.567983 -0.454845)))
       ((R:SylStructure.parent.syl_codasize < 2.2)
        ((ph_ctype is l)
         ((1.18845 0.809091))
         ((R:SylStructure.parent.position_type is initial)
          ((ph_ctype is n)
           ((0.773548 -0.277092))
           ((1.01586 0.281001)))
          ((p.ph_ctype is 0)
           ((1.06831 0.699145))
           ((0.924189 0.241873)))))
        ((R:SylStructure.parent.syl_break is 0)
         ((ph_ctype is n)
          ((0.592321 -0.470784))
          ((0.778688 -0.072112)))
         ((n.ph_ctype is s)
          ((1.08848 0.0733489))
          ((1.25674 0.608371)))))))))
((pos_in_syl < 0.7)
  ((p.ph_vlng is 0)
   ((R:SylStructure.parent.position_type is mid)
    ((ph_ctype is 0)
     ((ph_vheight is 2)
      ((0.456225 -0.293282))
      ((0.561529 -0.0816115)))
     ((0.6537 -0.504024)))
    ((ph_ctype is s)
     ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
      ((1.31586 0.98395))
      ((R:SylStructure.parent.position_type is single)
       ((0.816869 0.634789))
       ((R:SylStructure.parent.syl_out < 4.4)
        ((1.05578 0.479029))
        ((R:SylStructure.parent.asyl_in < 0.4)
         ((1.11813 0.143214))
         ((0.87178 0.406834))))))
     ((n.ph_ctype is n)
      ((R:SylStructure.parent.last_accent < 0.6)
       ((0.838154 -0.415599))
       ((0.924024 0.110288)))
      ((seg_onsetcoda is coda)
       ((nn.ph_vfront is 2)
        ((0.670096 0.0314187))
        ((n.ph_ctype is f)
         ((1.00363 0.693893))
         ((R:SylStructure.parent.syl_out < 6)
          ((0.772363 0.215675))
          ((0.920313 0.574068)))))
       ((R:SylStructure.parent.position_type is final)
        ((0.673837 -0.458142))
        ((R:SylStructure.parent.sub_phrases < 2.8)
         ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
          ((0.894817 0.304628))
          ((ph_ctype is n)
           ((0.787302 -0.23094))
           ((R:SylStructure.parent.asyl_in < 1.2)
```

```
          ((ph_ctype is f)
           ((R:SylStructure.parent.last_accent < 0.5)
            ((1.12278 0.326954))
            ((0.802236 -0.100616)))
           ((0.791255 -0.0919132)))
          ((0.95233 0.219053)))))
       ((R:SylStructure.parent.position_type is initial)
        ((ph_ctype is f)
         ((1.0616 0.216118))
         ((0.703216 -0.00834086)))
        ((ph_ctype is f)
         ((1.22277 0.761763))
         ((0.904811 0.332721)))))))))
 ((ph_vheight is 0)
  ((p.ph_vlng is s)
   ((0.873379 0.217178))
   ((n.ph_ctype is r)
    ((0.723915 1.29451))
    ((n.ph_ctype is 0)
     ((R:SylStructure.parent.R:Syllable.p.syl_break is 1)
      ((R:SylStructure.parent.sub_phrases < 4)
       ((seg_coda_fric is 0)
        ((p.ph_vlng is l)
         ((0.849154 0.945261))
         ((0.633261 0.687498)))
        ((0.728546 0.403076)))
       ((0.850962 1.00255)))
      ((0.957999 1.09113)))
     ((0.85771 0.209045)))))
  ((ph_vheight is 2)
   ((0.803401 -0.0544067))
   ((0.681353 0.256045)))))
((n.ph_ctype is f)
 ((ph_ctype is s)
  ((p.ph_vlng is 0)
   ((0.479307 -0.9673))
   ((0.700477 -0.351397)))
  ((ph_ctype is f)
   ((0.73467 -0.6233))
   ((R:SylStructure.parent.syl_break is 0)
    ((p.ph_ctype is s)
     ((0.56282 0.266234))
     ((p.ph_ctype is r)
      ((0.446203 -0.302281))
      ((R:SylStructure.parent.sub_phrases < 2.7)
       ((ph_ctype is 0)
        ((0.572016 -0.0102436))
        ((0.497358 -0.274514)))
       ((0.545477 0.0482177)))))
    ((ph_vlng is s)
     ((0.805269 0.888495))
     ((ph_ctype is n)
      ((0.869854 0.653018))
      ((R:SylStructure.parent.sub_phrases < 2.2)
       ((0.735031 0.0612886))
       ((0.771859 0.346637)))))))))
 ((R:SylStructure.parent.syl_codasize < 1.4)
  ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.3)
   ((R:SylStructure.parent.position_type is initial)
    ((0.743458 0.0411808))
```

```
  ((1.13068 0.613305)))
 ((pos_in_syl < 1.2)
  ((R:SylStructure.parent.R:Syllable.p.syl_break is 1)
   ((1.11481 0.175467))
   ((0.937893 -0.276407)))
  ((0.74264 -0.550878))))
((pos_in_syl < 3.4)
 ((seg_onsetcoda is coda)
  ((ph_ctype is r)
   ((n.ph_ctype is s)
    ((0.714319 -0.240328))
    ((p.ph_ctype is 0)
     ((0.976987 0.330352))
     ((1.1781 -0.0816682))))
   ((ph_ctype is l)
    ((n.ph_ctype is 0)
     ((1.39137 0.383533))
     ((0.725585 -0.324515)))
    ((ph_vheight is 3)
     ((ph_vlng is d)
      ((0.802626 -0.62487))
      ((n.ph_ctype is r)
       ((0.661091 -0.513869))
       ((R:SylStructure.parent.position_type is initial)
        ((R:SylStructure.parent.parent.word_numsyls < 2.4)
         ((0.482285 0.207874))
         ((0.401601 -0.0204711)))
        ((0.733755 0.397372)))))
     ((n.ph_ctype is r)
      ((p.ph_ctype is 0)
       ((pos_in_syl < 1.2)
        ((0.666325 0.271734))
        ((nn.ph_vheight is 0)
         ((0.642401 -0.261466))
         ((0.783684 -0.00956571))))
       ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
        ((0.692225 -0.381895))
        ((0.741921 -0.0898767))))
      ((nn.ph_vfront is 2)
       ((ph_ctype is s)
        ((0.697527 -1.12626))
        ((n.ph_ctype is s)
         ((ph_vlng is 0)
          ((R:SylStructure.parent.sub_phrases < 2.4)
           ((0.498719 -0.906926))
           ((0.635342 -0.625651)))
          ((0.45886 -0.385089)))
         ((0.848596 -0.359702))))
       ((p.ph_vlng is a)
        ((p.ph_ctype is 0)
         ((0.947278 0.216904))
         ((0.637933 -0.394349)))
        ((p.ph_ctype is r)
         ((R:SylStructure.parent.syl_break is 0)
          ((0.529903 -0.860573))
          ((0.581378 -0.510488)))
         ((ph_vlng is 0)
          ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
           ((seg_onset_stop is 0)
            ((R:SylStructure.parent.syl_break is 0)
```

```
                ((p.ph_vlng is d)
                 ((0.768363 0.0108428))
                 ((ph_ctype is s)
                  ((0.835756 -0.035054))
                  ((ph_ctype is f)
                   ((p.ph_vlng is s)
                    ((0.602016 -0.179727))
                    ((0.640126 -0.297341)))
                   ((0.674628 -0.542602)))))
                ((ph_ctype is s)
                 ((0.662261 -0.60496))
                 ((0.662088 -0.432058))))
               ((R:SylStructure.parent.syl_out < 4.4)
                ((0.582448 -0.389079))
                ((ph_ctype is s)
                 ((0.60413 -0.73564))
                 ((0.567153 -0.605444)))))
              ((R:SylStructure.parent.R:Syllable.p.syl_break is 2)
               ((0.761115 -0.827377))
               ((ph_ctype is n)
                ((0.855183 -0.275338))
                ((R:SylStructure.parent.syl_break is 0)
                 ((0.788288 -0.802801))
                 ((R:SylStructure.parent.syl_codasize < 2.2)
                  ((0.686134 -0.371234))
                  ((0.840184 -0.772883)))))))
             ((pos_in_syl < 1.2)
              ((R:SylStructure.parent.syl_break is 0)
               ((n.ph_ctype is n)
                ((0.423592 -0.655006))
                ((R:SylStructure.parent.syl_out < 4.4)
                 ((0.595269 -0.303751))
                 ((0.478433 -0.456882))))
               ((0.688133 -0.133182)))
              ((seg_onset_stop is 0)
               ((1.27464 0.114442))
               ((0.406837 -0.167545))))))))))))
     ((ph_ctype is r)
      ((0.462874 -0.87695))
      ((R:SylStructure.parent.R:Syllable.n.syl_onsetsize < 0.2)
       ((0.645442 -0.640572))
       ((0.673717 -0.321322)))))
    ((0.61008 -0.925472))))))))
;; RMSE 0.8085 Correlation is 0.5899 Mean (abs) Error 0.6024 (0.5393)

))

(provide 'gswdurtreeZ)
```

Bijlage 10 anmrpa

```
; This file enables the use of CSTR's MRPA phonetic notation system with en1.
;; RENAME _ #
;; FLUSH $
```

Bijlage 11 temp.pho

;; #bjYst&rba:rlik#

_ 84
b 80
j 67
Y 75 (30 , 145)
s 60
t 40
& 40
r 94
b 80
a: 170 (45 , 175)
r 79
l 67
& 40 (24 , 130)
k 50
_ 84

Bijlage 12 opbouw directories