Spraakverwerking per computer

Instituut voor Fonetische Wetenschappen
Universiteit van Amsterdam

email: David.Weenink@hum.uva.nl

Gebruikte hulpbronnen:

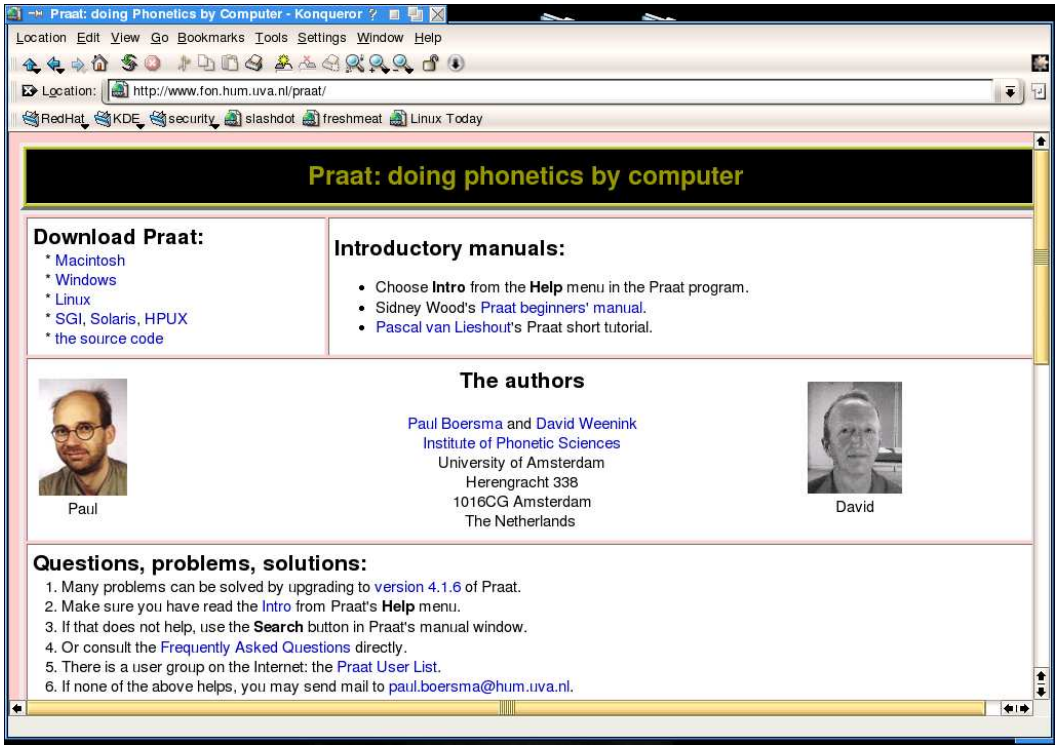
David Huber & Robert Runstein (1997), *Modern Recording Techniques*, Focal Press.

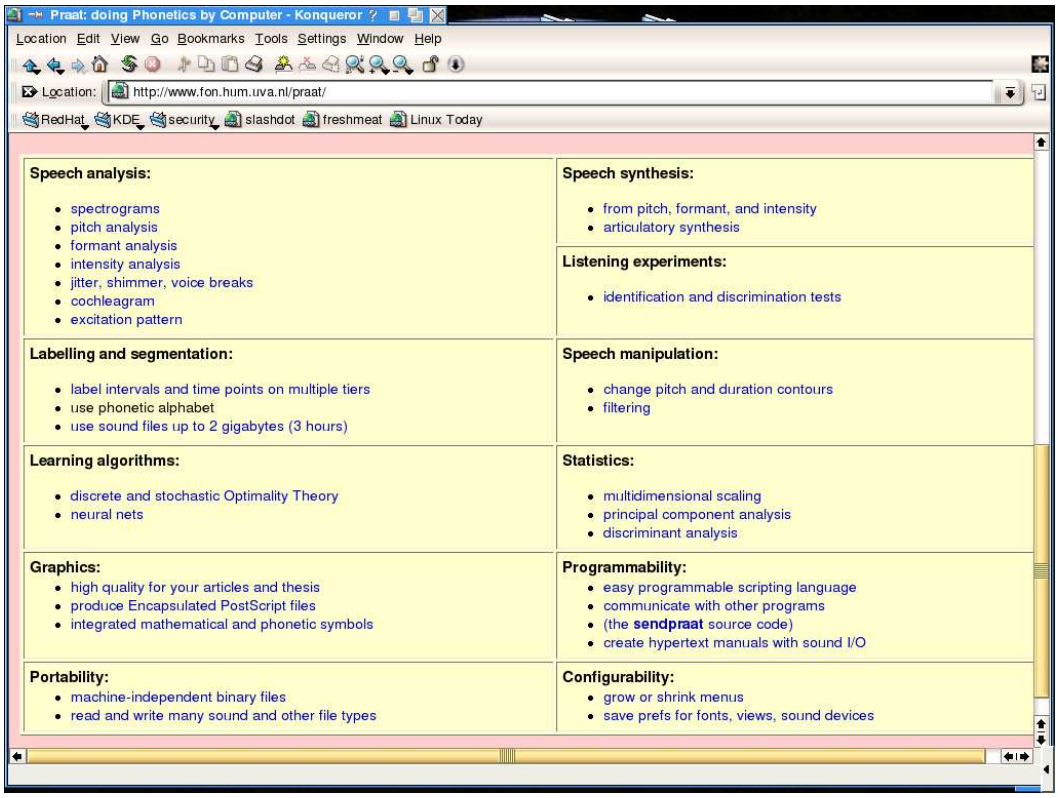
Peter Ladefoged (2003), *Phonetic Data Analysis*, Blackwell.

Keith Johnson (1997), *Acoustic and Auditory Phonetics*, Blackwell.

David Weenink (2003), *Spraaksignaalanalyse per computer*,
URL: http://www.fon.hum.uva.nl/david/spc/spc.pdf

Paul Boersma & David Weenink, PRAAT,
URL: http://www.fon.hum.uva.nl/praat


Praat: doing Phonetics by Computer - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: http://www.fon.hum.uva.nl/praat/
RedHat KDE security slashdot freshmeat Linux Today
Praat: doing phonetics by computer
Download Praat:
* Macintosh
* Windows
* Linux
* SGI, Solaris, HPUX
* the source code
Introductory manuals:
Choose Intro from the Help menu in the Praat program.
Sidney Wood's Praat beginners' manual.
Pascal van Lieshout's Praat short tutorial.
The authors
Paul
Paul Boersma and David Weenink
Institute of Phonetic Sciences
University of Amsterdam
Herengracht 338
1016CG Amsterdam
The Netherlands
David
Questions, problems, solutions:
1. Many problems can be solved by upgrading to version 4.1.6 of Praat.
2. Make sure you have read the Intro from Praat's Help menu.
3. If that does not help, use the Search button in Praat's manual window.
4. Or consult the Frequently Asked Questions directly.
5. There is a user group on the Internet: the Praat User List.
6. If none of the above helps, you may send mail to paul.boersma@hum.uva.nl.


Praat: doing Phonetics by Computer - Konqueror
Location Edit View Go Bookmarks Tools Settings Window Help
Location: http://www.fon.hum.uva.nl/praat/
RedHat KDE security slashdot freshmeat Linux Today
Speech analysis:
spectrograms
pitch analysis
formant analysis
intensity analysis
jitter, shimmer, voice breaks
cochleagram
excitation pattern
Speech synthesis:
from pitch, formant, and intensity
articulatory synthesis
Listening experiments:
identification and discrimination tests
Labelling and segmentation:
label intervals and time points on multiple tiers
use phonetic alphabet
use sound files up to 2 gigabytes (3 hours)
Speech manipulation:
change pitch and duration contours
filtering
Learning algorithms:
discrete and stochastic Optimality Theory
neural nets
Statistics:
multidimensional scaling
principal component analysis
discriminant analysis
Graphics:
high quality for your articles and thesis
produce Encapsulated PostScript files
integrated mathematical and phonetic symbols
Programmability:
easy programmable scripting language
communicate with other programs
(the sendpraat source code)
create hypertext manuals with sound I/O
Portability:
machine-independent binary files
read and write many sound and other file types
Configurability:
grow or shrink menus
save prefs for fonts, views, sound devices

Wat gaan we doen in deze colleges?

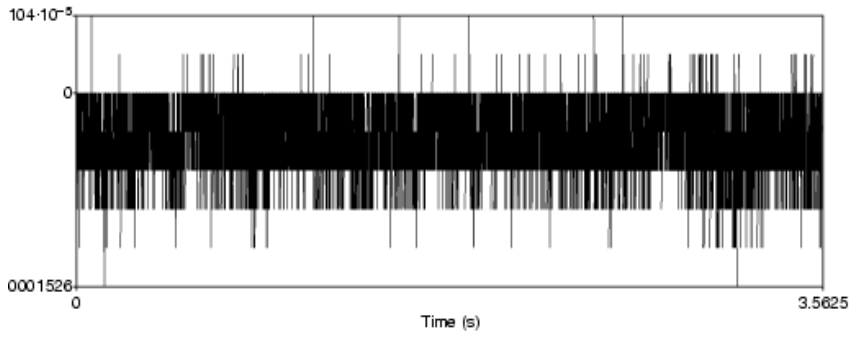
- Opnametechniek (analoog + digitaal)
- Maken van digitale opnames
- Opnames invoeren in de computer
- Analyse van opnames met PRAAT
- Spectrum, Spectrogram, LPC, Pitch, Formant...

Het maken van opnames

Grootste probleem: er is altijd ruis (achtergrondruis, interne ruis apparatuur)

Daarom: in een opnamestudio

Achter je PC kan wel maar....



De onderdelen van de opnameketen

- De spreker
- De opnametechnicus
- De opnameruimte
- De opnameapparatuur: de microfoon, de kabels, de versterkers, het mengpaneel, het opnameapparaat

Zelf opnames maken

- Waar? Opnameruimte (2-de verdieping)
- Hoe? Direct op audioCD
- Waar zijn audioCD rewriteables: bij Ton Wempe (sousterrain)
- Opnames maak je met z'n tweeën
- Inschrijvingslijst hangt op de deur
- *Handleiding opnemen van spraak en invoeren in de computer*
- *Aanwijzingen voor het opnemen/kopiëren van geluid*
- Van audioCD naar bestanden op de harde schijf

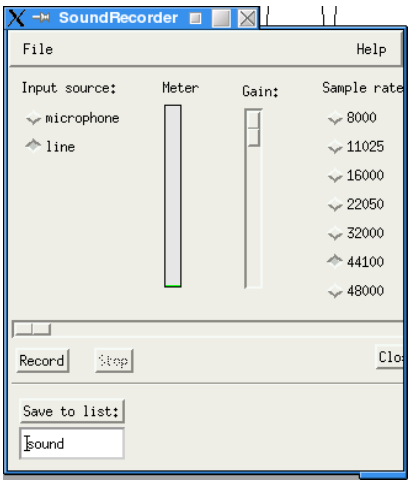
Waar moet je op letten bij het opnemen?

- Maak van te voren het protocol (op papier)
- Zorg dat spreker comfortabel zit zodat
  - hij/zij goed kan lezen van het papier
  - hij/zij niet hoeft te bewegen tijdens de opname
  - De afstand tot de microfoon konstant blijft (30 cm)
- Zorg dat de spreker storende bijgeluiden zoals
  het knisperen van het papier en lipgeluiden vermijdt
  (evt glas water)
- Laat de spreker zo natuurlijk mogelijk spreken:
  - bed de zinnen in tussen dragerzinnen!
- Test het geluidsnivo voor je gaat opnemen
- Begin een opname altijd met *datum* en *naam van de spreker*
- Maak optimaal gebruik van het opneembereik
  (stuur optimaal uit)
- De tracknummering van de CD-recorder is je vriend!

Invoeren in de computer

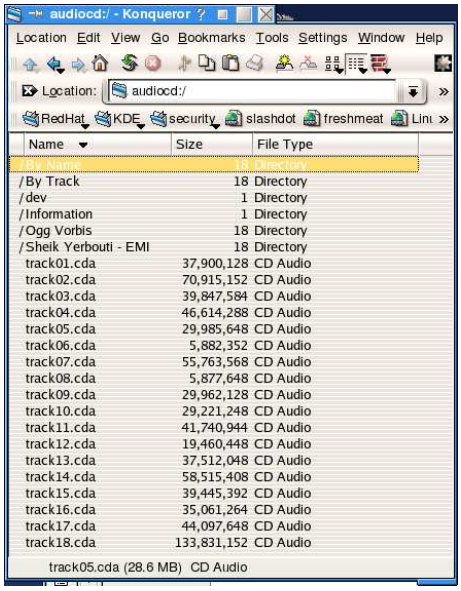
- Analoog met gebruikmaking van de Analoog Digitaal Converter (ADC) van de computer en praat
  - Audiocd digitaal (DAC) -> analoog
  - Soundcard analog -> digitaal
  - Naar bestand op schijf (wav, aiff...)
- Digitaal (via een audioripper) die rechtstreeks naar file schrijft (“zonder” kwaliteitsverlies)

Analoge invoer met PRAAT



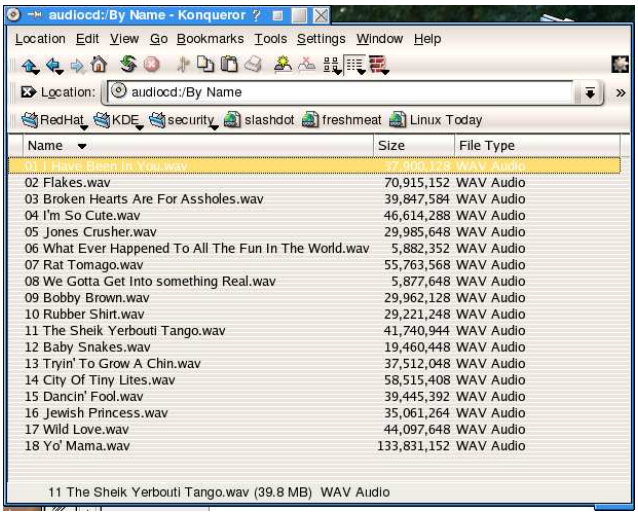
- Zorg dat de mixer goed staat
- Praat:New/Record mono Sound...
  - Line input
  - Regel gain af
  - Sample rate 44100 Hz
  - Record
  - Save to list

Digitale invoer van audioCD (Linux)

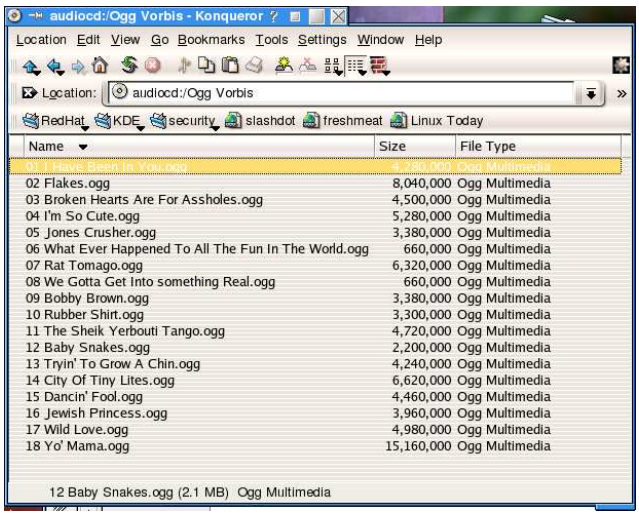


- Audio cd in cdrom drive
- audiocd:/
- Diverse uitvoermogelijk heden
- Echter: dubbele opslag (L+R)
- Daarom in *praat*:
  - Read two Sounds from stereo file...
  - Remove... left
  - Write to WAV file...

- Naar wav bestand
- audiocd:/By Name
- Bij muziek CD worden de titels opgezocht op internet



- Naar Ogg Vorbis bestand
- Audiocd:/Ogg Vorbis
- Vergelijkbaar met MP3 maar open standaard



Over bits en bytes

- De bit (**b**inary dig**it**)is de basisinformatie-eenheid
  - Interne representatie via *twee* spanningsnivo's
  - Externe reps: 0 of 1, True of False, Yes or No ...
- Een *byte* is een groepje van 8 bits
  - 2^8 (= 256) mogelijkheden
  - Bit nummering r->l: 76543210
  - (0)0110001 == 'A' (0)0110010 == 'B' (ASCII)
  - 00110001 == 49 00110010 == 50
- Combinatie van *N* bits geven 2^N mogelijkheden
  - 2^{16} = 65536 2^{32} = 4294967296 $\approx$ 4 10^9
  - 2^{64} = 18446744073709551616 $\approx$ 1.8 10^{19}

Bits en bytes (2)

- 32 bit processors
  - Intel i386, .., i686, Pentium4, IBM Power
  - Data registers 32 bit en/of
  - 32 bit geheugenadressen (deze wijzen bytes aan)
- 64 bit processors (Itanium, Power, MIPS, SPARC...)
- Graphics
  - 8 bit kleur: 256 verschillende kleuren
  - 24 of 42 bit kleur: true color

Bytes en veelvouden hiervan

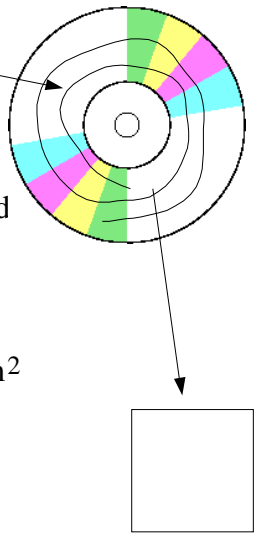
- k (kilo) == 10^3 = 1000 (duizend)
  - K (Kilo) == 1024 = 2^{10} byte
- M (mega) == 10^6 = 1.000.000 (miljoen)
  - 1 Mbyte == 1024^2 = 2^{20} = 1.048.576 byte
- G (giga) == 10^9 = 1.000.000.000 (miljard, billion)
  - 1 Gbyte = 1024^3 = 2^{30} = 1.073.741.824 byte
- T (tera) == 10^{12} = 1.000.000.000.000 (biljoen)
  - 1 Tbyte == 1024^4 = 2^{40} = 1.099.511.627.776 byte
  -
- P (peta) == 10^{15} (biljard)
  - 1 Pbyte == 1024^5 = 2^{50} = 1.125.899.906.842.624 byte

CD standaarden

- Audio (CD-DA: standaard *red* book)
- Gegevensopslag (CD-ROM: *yellow* book)
  - CD-I (Interactief *green* book)
  - MPEG-1 video CD (*white* book)
  - CD extra (audio+data) (*blue* book)
  - CD-R -RW (*orange* book, 3 delen:)
    - 1. CD-MO (niet populair)
    - 2. CD-R
    - 3. CD-RW

- "Kopieerbeveiligde" audio CD's voldoen NIET aan de red book spe
  (en zouden dus eigenlijk verboden moeten worden)

CD (audio)



- *Program area* tussen 12.5mm en 58 mm van midd
  - CD oppervlakte ongeveer 100 cm^2
  - 74 minuten stereo (≈ 6.300.000.000 bits)
  - ≈ 63.000.000 bits / cm^2
  - Maximale oppervlakte van 1 bit ≈ 1.5 10^{-8} cm^2
    Vierkantje met lengte 1.2 10^{-4} cm
    = 1.2 10^{-3} mm = 1.2 μm
- Opslag als spiraal vanaf het midden
  - constante hoeveelheid bits / lengte
  - spoorafstand1.6 μm ± 0.1 μm
  - rotatiesnelheid?

CD (data)

- Opslag in sectoren (≥ 330.000 geen spiraal)
- 75 minuten data: 75 x 60 x 75 x 2048 = 691.200.000
    - 75 minuten
    - 60 seconden
    - 75 frames / s
    - 2048 byte / frame

Getallenstelsels

Positioneel stelsel met basis *b* met *b* symbolen

Waarde van een symbolenreeks $s_n s_{n-1} \ldots . s_2 s_1 s_0$?

- De positie van een symbool in de reeks bepaalt zijn waarde
  de 1 in 12 en 1234 heeft waardes 10 en 1000

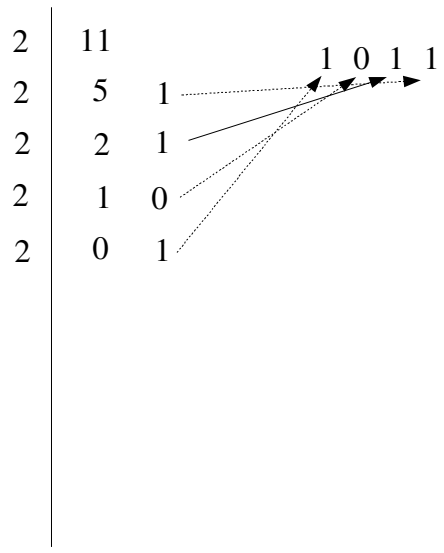
- Decimaal 1234 = $1.1000 + 2.100 + 3.10 + 4 =$
  $1.10^3 + 2.10^2 + 3.10^1 + 4.10^0$

- In het algemeen:
  de waarde van de reeks $s_n s_{n-1} \ldots . s_2 s_1 s_0$ met basis *b*
  $= s_n.b^n + s_{n-1}b^{n-1} + \ldots + s_2.b^2 + s_1.b^1 + s_0.b^0$

Getallenstelsels (voorbeelden)

- Decimaal, basis 10
  - Symbolen 0 1 2 3 4 5 6 7 8 9
  - 10 + 20 = 30
- Binair, basis b = 2
  - Symbolen 0 1
  - 1010 + 10100 = 11110 *(10 + 20 = 30)*
- Octaal, basis 8
  - Symbolen 0 1 2 3 4 5 6 7
  - 12 + 24 = 36
- Hexadecimaal, basis 16
  - Symbolen 0 1 2 3 4 5 6 7 8 9 A B C D E F
  - A + 14 = 1E

Van decimaal naar binair



De logaritme

Niet linear (rechte lijn)
Monotoon stijgend
Laag “gevoeliger” dan hoog:

- y-as opgerekt
- 0-10 == 1
- 10-100 == 1
- 100 – 1000 == 1
- ...

Gevoeligheid oor:

- Basilair membraan (y-as)
- 1 – 10.000.000.000.000

De logaritme (vervolg)

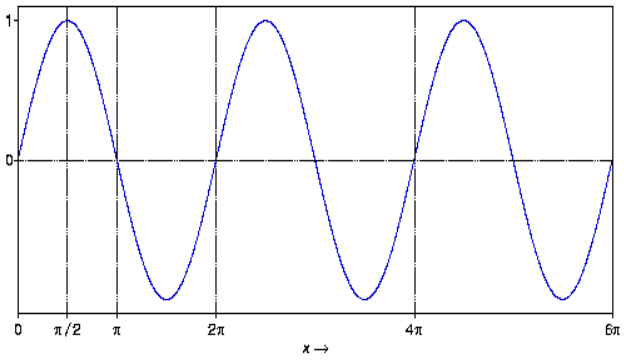
- log 1 = 0
- log 2 ≈ 0.3 (let op: ipv 0,3 !)
- log 10 = 1
- log (a x b) = log a + log b
- log (a/b) = log a – log b
- log a^x = x log a
- log 100 = log (10x10) = log 10 + log 10 = 1 + 1 = 2
- log 100 = log 10^2 = 2 log 10 = 2
- log 20 = log (2x10) = log 2 + log 10 ≈ 1.3
- log 200 = log (2x100) = log 2 + log 100 ≈ 2.3
- log 1000 = log (10x10x10) = log (10x100) = 3
- log 10000 = 4
- log 10.000.000.000.000 = log 10^{13} = 13
- log 0.1 = log (1/10) = log 1 – log 10 = -1
- Log 0.5 = log ½ = log 1 – log 2 = -0.3

Decibel (dB)

Logaritme van een vermogensverhouding:

- 10 log (P/P_{ref})
- Vermogen = amplitude^2
- 20 log ($Amplitude/Amplitude_{ref}$)

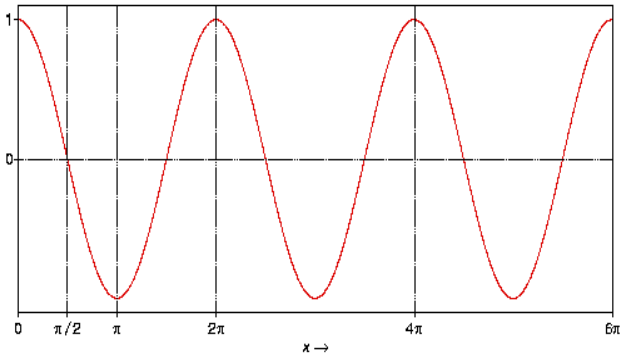
sinus



- sin (x)
- $\pi \approx 3.14152...$
- Periodiek
- Ook voor x < 0

Create Sound... sinus 0 6*pi 1000 sin (x)

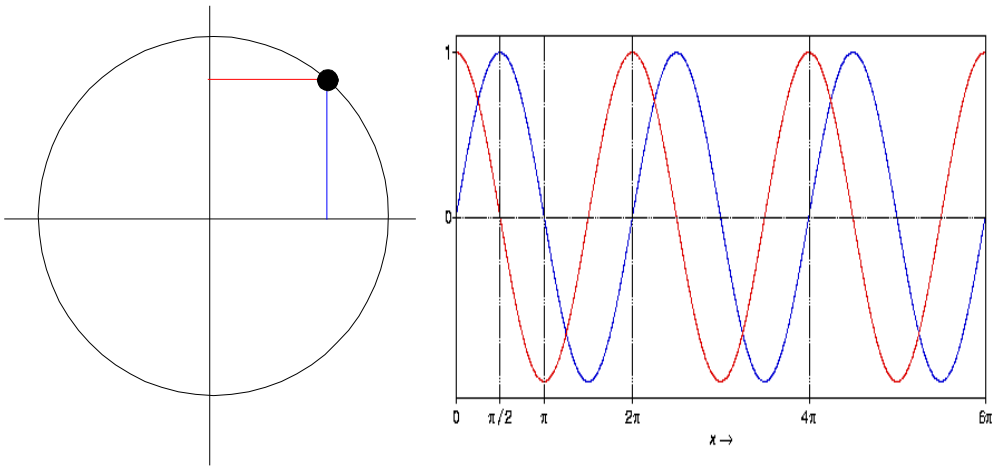
cosinus



- cos (x)
- $\pi \approx 3.14152...$
- Periodiek
- Ook voor x < 0

Create Sound... cosinus 0 6*pi 1000 cos (x)

Sinus en cosinus



- Circelbeweging met “constante” snelheid
- Faseverschil tussen sin en cos is $\pi/2$ (90°)

Wat doet de geluidskaart (soundcard)

- Opgenomen geluid afspelen (CD, wav, DVD, spelletjes)
- Geluid opnemen van externe bronnen (microfoon, tape)
- Geluid bewerken
- (Geluid synthetiseren)

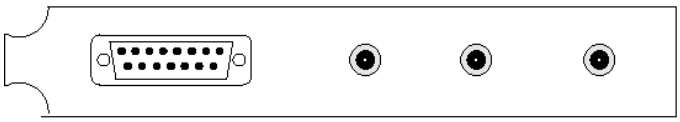
- Componenten o.a. DSP, DAC, ADC

Geluidskaartcomponenten

- ADC : Analoog naar Digitaal Converter
  - Maakt een reeks getallen van analoog geluid
- DAC : Digitaal naar Analoog Converter
  - Zet reeks getallen om in analoog geluid
- DSP : Digitale Signaal Processor
  - Bewerkingen op het digitale signaal

Sound Blaster 16 PCI

Een mogelijke geluidskaart (achterkant computer)



- Line in: voor cassettedeck, taperecorder etc
- Line out: voor luidsprekers of koptelefoon
- Mic: voor microfoon
- (MIDI/GAME: voor MIDI apparaten en spelletjes)

Geluidskaart (over spanningen en zo)

- Line (in+out) bereik ongeveer 1000 mV
  - Line out kan op Line in aangesloten worden
- Microfoon ingang normaal (1 mV) op PC-kaarten anders
  -
- Let op bij een pick-up.
  - element levert enkel mV.
  - speciale (niet-lineaire) voorversterker nodig (RIAA).
  - *phono*-ingang van versterkers heeft RIAA karakteristiek
  - anders lage tonen te zacht, hoge te sterk

De geluidskaart van de computer (sound card)

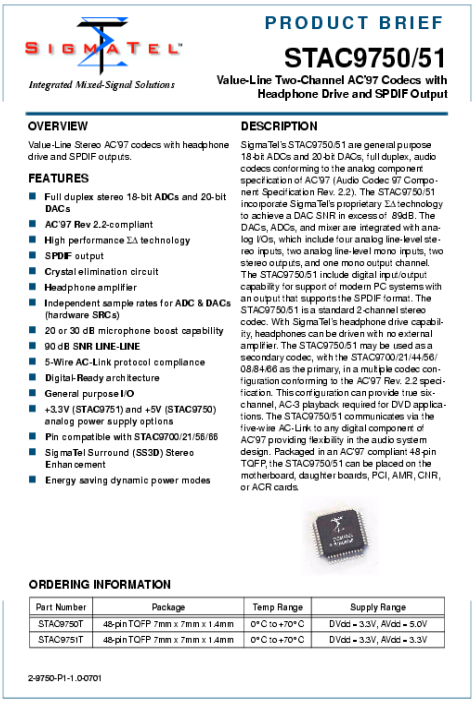
Wat zit er op een geluidskaart:

- Invoer
  - Analoog
    - Line in
    - Microfoon
    - Audio-cd (digitaal/analoog)
  - Digitaal
    - Bytes van het computergeheugen
- Uitvoer
  - Analoog
    - Line uit
  - Digitaal
    - Bytes naar het computergeheugen
    - (SPDIFF)
- (MIDI/GAME: Musical Instrument Digital Interface)

Het maken van een opname via de microfoon op de geluidskaart

Van geluid naar file:

- Het analoge signaal gaat via de microfoonplug naar de kaart
- Een computerprogramma heeft al geselecteerd welke geluidsbronnen er gebruikt gaan worden
- De uitvoer van de mixer gaat naar de ADC
- De digitale uitvoer van de ADC gaat naar de DSP
  - Voor verdere bewerking
  - DSP is onafhankelijk van de CPU
- De uitvoer van de DSP gaat via een digitale bus (lijn) naar de computer
- De processor van de computer zendt de gegevens naar de schijfcontrolle die het naar de harde schijf zend.



SIGMATEL
Integrated Mixed-Signal Solutions

PRODUCT BRIEF

STAC9750/51

Value-Line Two-Channel AC'97 Codecs with Headphone Drive and SPDIF Output

OVERVIEW

Value-Line Stereo AC'97 codecs with headphone drive and SPDIF outputs.

FEATURES

- Full duplex stereo 18-bit ADCs and 20-bit DACs
- AC'97 Rev 2.2-compliant
- High performance ΣΔ technology
- SPDIF output
- Crystal elimination circuit
- Headphone amplifier
- Independent sample rates for ADC & DACs (hardware SRCs)
- 20 or 30 dB microphone boost capability
- 90 dB SNR LINE-LINE
- 5-Wire AC-Link protocol compliance
- Digital-Ready architecture
- General purpose I/O
- +3.3V (STAC9751) and +5V (STAC9750) analog power supply options
- Pin compatible with STAC9700/21/58/88
- SigmaTel Surround (SS3D) Stereo Enhancement
- Energy saving dynamic power modes

DESCRIPTION

SigmaTel's STAC9750/51 are general purpose 18-bit ADCs and 20-bit DACs, full duplex, audio codecs conforming to the analog component specification of AC'97 (Audio Codec '97 Component Specification Rev. 2.2). The STAC9750/51 incorporate SigmaTel's proprietary ΣΔ technology to achieve a DAC SNR in excess of 89dB. The DACs, ADCs, and mixer are integrated with analog I/Os, which include four analog line-level stereo inputs, two analog line-level mono inputs, two stereo outputs, and one mono output channel. The STAC9750/51 include digital input/output capability for support of modern PC systems with an output that supports the SPDIF format. The STAC9750/51 is a standard 2-channel stereo codec. With SigmaTel's headphone drive capability, headphones can be driven with no external amplifier. The STAC9750/51 may be used as a secondary codec, with the STAC9700/21/44/56/08/84/66 as the primary, in a multiple codec configuration conforming to the AC'97 Rev. 2.2 specification. This configuration can provide true six-channel, AC-3 playback required for DVD applications. The STAC9750/51 communicates via the five-wire AC-Link to any digital component of AC'97 providing flexibility in the audio system design. Packaged in an AC'97 compliant 48-pin TQFP, the STAC9750/51 can be placed on the motherboard, daughter boards, PCI, AMR, CNR, or ACR cards.

ORDERING INFORMATION

Part Number	Package	Temp Range	Supply Range
STAC9750T	48-pin TQFP 7mm x 7mm x 1.4mm	0°C to +70°C	DVdd = 3.3V, AVdd = 5.0V
STAC9751T	48-pin TQFP 7mm x 7mm x 1.4mm	0°C to +70°C	DVdd = 3.3V, AVdd = 3.3V

2-9750-P1-1.00701

DESCRIPTION (CONTINUED)

The STAC9750/51 block diagram is illustrated on page 3. It provides variable sample rate Digital-to-Analog (DA) and Analog-to-Digital (AD) conversion, mixing, and analog processing. Supported audio sample rates include 48 kHz, 44.1 kHz, 32 kHz, 22.05 kHz, 16 kHz, 11.025 kHz, and 8 kHz; additional rates are supported in the STAC9750/51 soft audio drivers. The digital interface communicates with the AC'97 controller via the five-wire AC-Link and contains the 64-word by 16-bit registers. The two DACs convert the digital stereo PCM-out content to audio. The MIXER block combines the PCM_OUT with any analog sources, to drive the LINE_OUT and HP_OUT outputs. The MONO_OUT delivers either mic only, or a mono mix of sources from the MIXER. The stereo variable sample rate ADC's provide record capability for any mix of mono or stereo sources, and deliver a digital stereo PCM-in signal back to the AC-Link. The microphone input and mono input can be recorded simultaneously, thus allowing for an all digital output in support of the digital ready initiative. All ADC's operate at 18-bit resolution and DAC's at 20-bit resolution. For a digital ready record path, the microphone is connected to the left channel ADC while the mono output of the stereo mixer is connected to right channel ADC. Make sure the microphone input is not connected to the stereo mixer when in this mode.

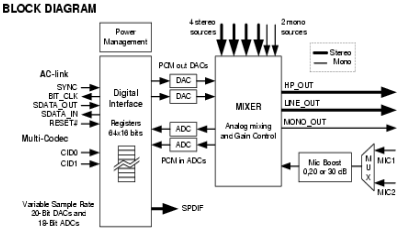
The STAC9750/51 supports General Purpose Input/Output (GPIO), as well as SPDIF output. These digital I/O options provide for a number of advance architectural implementations, with volume controls and digital mixing capabilities built directly into the codec.

The STAC9750/51 is designed primarily to support stereo (2-speaker) audio. True AC-3 playback can be achieved for 6-speaker applications by taking advantage of the multi-codec option available in the STAC9750/51 to support multiple codecs in an AC'97 architecture. Additionally, the STAC9750/51 provides for a stereo enhancement feature, SigmaTel Surround 3D (SS3D). SS3D provides the listener with several options for improved speaker separation beyond the normal 2/4-speaker arrangements.

Together with the logic component (controller or advanced core logic chip-set) of AC'97, STAC9750/51 can be SoundBlaster® and Windows Sound System® compatible with SigmaTel's WDM driver for WIN 98/2K/ME/XP. SoundBlaster is a registered trademark of Creative Labs. Windows is a registered trademark of Microsoft Corporation.

Copyright © 2001 SigmaTel, Inc. All rights reserved.
All contents of this document are protected by copyright law and may not be reproduced without the express written consent of SigmaTel, Inc.
SigmaTel, the SigmaTel logo, and combinations thereof are trademarks of SigmaTel, Inc. Other product names used in this publication are for identification purposes only and may be trademarks or registered trademarks of their respective companies. The contents of this document are provided in connection with SigmaTel, Inc. products. SigmaTel, Inc. has made best efforts to ensure that the information contained herein is accurate and reliable. However, SigmaTel, Inc. makes no warranties, express or implied, as to the accuracy or completeness of the contents of this publication and is providing this publication "AS IS". SigmaTel, Inc. reserves the right to make changes to specifications and product descriptions at any time without notice, and to discontinue or make changes to its products at any time without notice. SigmaTel, Inc. does not assume any liability arising out of the application or use of any product or circuit, and specifically disclaims any and all liability, including without limitation special, consequential, or incidental damages.

STAC9750/51 BLOCK DIAGRAM



KEY SPECIFICATIONS

- Analog **LINE_OUT SNR:** 90 dB
- **Digital DAC SNR:** 89 dB
- **Digital ADC SNR:** 85 dB
- **Full-scale Total Harmonic Distortion:** 0.005%
- **Crosstalk between Input Channels:** -70 dB
- **Spurious Tone Rejection:** 100 dB

RELATED MATERIALS

- **Data Sheet**
- **Reference Designs for ME, CNR, ACR and PCI applications**
- **Audio Precision Performance Plots**

STAC9750/51 MIXER

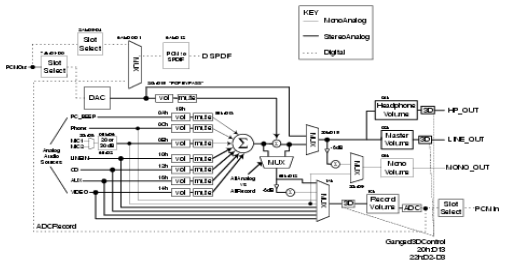


Figure 1. STAC9750 2-Channel Mixer Functional Diagram

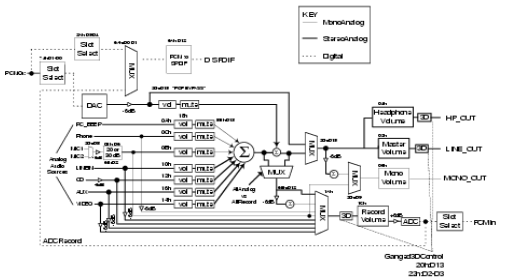


Figure 2. STAC9751 2-Channel Mixer Functional Diagram

PIN DESCRIPTION

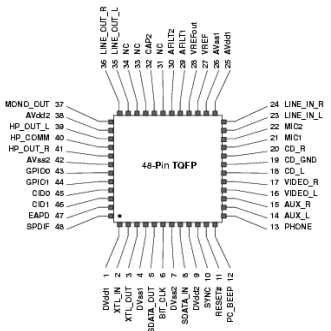


Figure 3. STAC9750/51 Pin Description Drawing

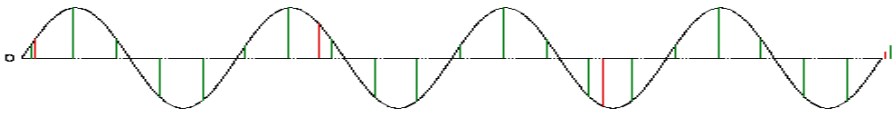
ADDITIONAL SUPPORT

Additional product and company information can be obtained by going to the SigmaTel website at:
www.sigmatel.com

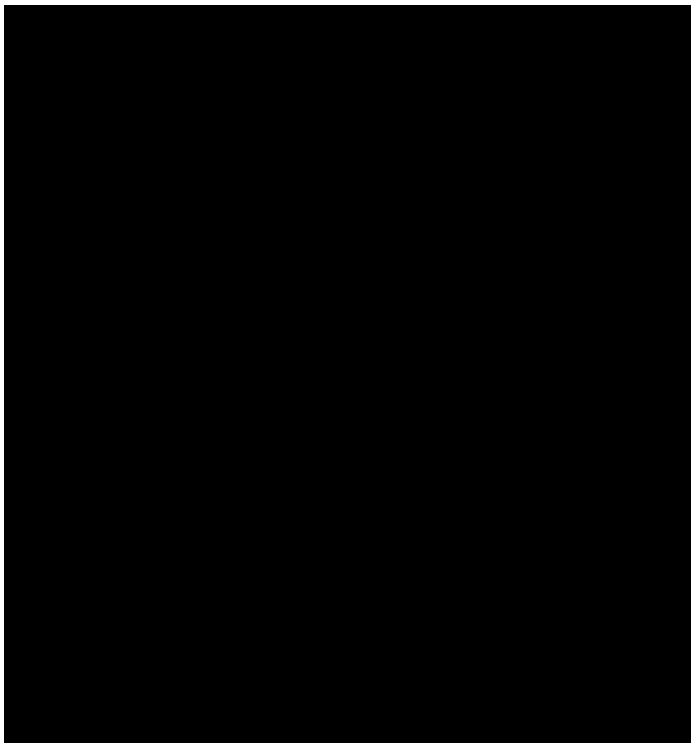
Analoog naar digitaal conversie

- Geluiden om ons heen zijn analoog
  - Continue
  - Op alle 'tussenliggende' tijdstippen gedefiniëerd
- Computer kent alleen discrete eenheden
  - Bits en bytes
- ADC zet analoog om naar een rijtje getallen
  - Bemonstering (hoe vaak per seconde)
  - Kwantisatie (hoe precies)

ADC : bemonstering (*eng*: sampling)



- Bemonsteren
  - Op vaste tijdsafstand T :
    - Fs : bemonsteringsfrequentie $1 / T$
      (*eng*: sample rate, sampling frequency)
    - T : bemonsteringstijd (*eng*: sampling time)
    - s_n : Monsterwaardes (*eng*: samples)
  - rood te weinig, groen genoeg bemonsterd



Vouwvervorming
(*eng*: aliasing)

- *Fs* = 8 Hz
- (a) *f* = 1 Hz
- (b) is (a) sampled
- (c) *f* = 8 Hz
- (d) is (c) sampled
- (e) = (a) + (c)
- (f) is (e) sampled

Vouwvervorming hoorbaar gemaakt (*praat* script)

```
fs = 11025
f1 = 1000
f2 = 4 * fs + 500                                   ! 44600 Hz
Create Sound... s1 0 1 'fs' sin (2*pi *'f1' * x)
Play
Create Sound... s2 0 1 'fs' sin (2*pi *'f2' * x)
Play
Create Sound... s3 0 1 'fs' (sin (2*pi *'f1' *x) +
   ... sin (2 * pi * 'f2' * x)) / 2
Play
```

Bemonsteringstheorema van Nyquist-Shannon:

Echte versie:
Als de bemonsteringsfrequentie *minstens twee maal zo hoog* is als de *bandbreedte* van het signaal dan kan het signaal *exact* uit zijn monsterwaardes *gereconstrueerd worden.*

Populaire versie:
Als de bemonsteringsfrequentie *minstens twee maal zo hoog* is als de *hoogste frequentie* in het signaal dan kan het signaal *exact* uit zijn monsterwaardes *gereconstrueerd worden.*

Bemonsteren (sampling)

Belangrijk:

Zorg dat de bemonsteringsfrequentie minstens twee maal zo hoog is als de hoogste frequentiecomponent van het signaal.

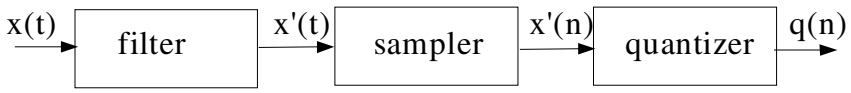
Als de bemonsteringsfrequentie niet vrij gekozen kan worden, dan moeten, met behulp van een filter, de frequentiecomponenten die hoger zijn dan de helft van de te gebruiken bemonsterings-frequentie weggefilterd worden.

ADC : kwantisatie (*eng*: quantisation)

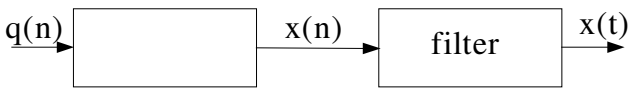
Eenheid: aantal bits / monsterwaarde

- 8 bit / 8 kHz : Alaw en μlaw
- 12 bit / 10..16 kHz : oudere opnames
- 16 bit / 44.1 kHz : CD's
- 16 bit / 48 kHz : DAT
- 24 bit / 48 en 192 kHz : (nu nog) professionele geluidskaarten
- 16, 20, 24 bit / 44.1, 48, 88.2, 96, 176.4, 192 kHz : DVD-audio

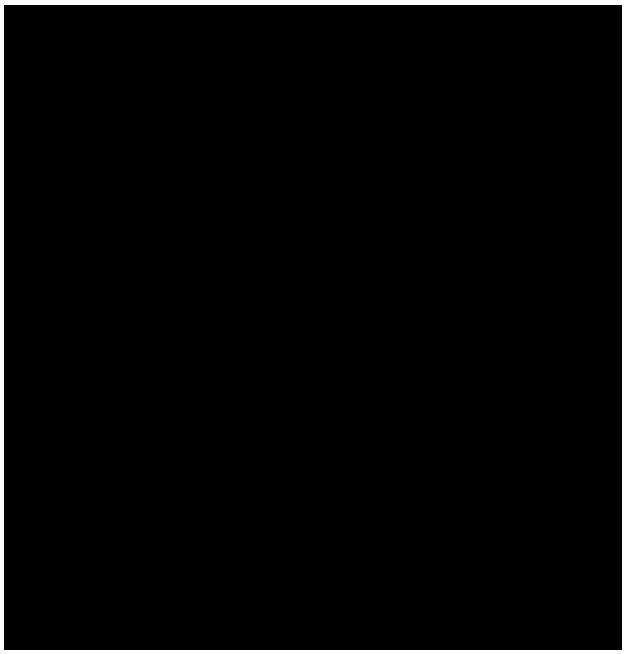
ADC en DAC



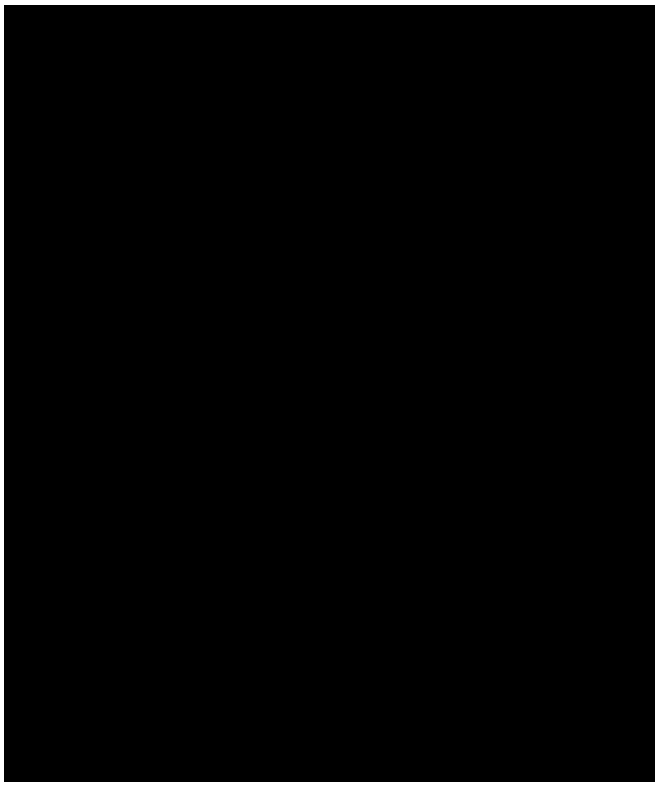
- Kwantisatiefout e(t) = x'(t) - q(n)



Kwantisatie nivo's (3 bit)



- 3 bit ⇒ 8 nivo's
- 0 nivo: sym / asym
- Ruis ⇒ asym
- Voor 3 bit
  - Laag: -4
  - Hoog: +3
- Voor 16 bit:
  - -32768
  - +32767



Kwantisaties

1, 2, 4, 8 en 16 bit

Kwantisaties

- PCM (Puls Code Modulation)
  Kwantiseert de *grootte* van de amplitude met *N* bits.
- ADPCM (Adaptive Differential Pulse Code Modulation)
  Kwantiseert de *verandering* van de amplitude met *n* bits
  Het aantal bits kan adaptief aangepast worden.
- In het algemeen is het aantal bits $n << N$
  en de bemonsteringsfrequentie veel hoger
- Extreem geval van ADPCM is DSD (Super Audio-CD)
  - 1 bit kwantisatie
  - Bemonstering 64 x 44.100 Hz = 2.822.400 Hz

Super Audio CD en DVD parameters

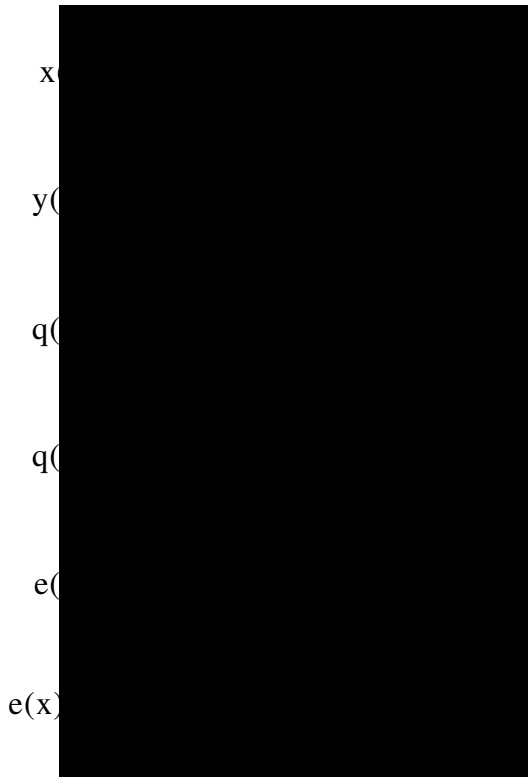
Parameter	**DVD-Audio**	**SACD**	**CD**
Codering	16,20,24 bit LPCM	1 bit DSD	16 bit LPCM
Bem.freq.	44.1, 48, 88.2, 96, 176.4, 192 kHz	2.822,4 kHz	44.1kHz
Kanalen	1-6	2-6	2
Compressie	ja (MLP)	ja (DST)	nee
Foutcorrectie	ja	ja	CIRC
Vercijfering	ja	ja	nee
Speeltijd	62-843 min	70-80 min	74 min
Bandbreedte	0-96 kHz	0-100 kHz	0-20kHz
Dynamiek	tot 144 dB	> 120 dB	96 dB

Ref: Konstantinides (2003), *An introduction to Super Audio CD and DVD-Audio*, IEEE Signal Processing, vol 20, pag 71.

Codering: μlaw kwantisatie (8 bit)

$$y(x) = sign(x)\, x_{max} \frac{\log(1+\mu|x/x_{max}|}{\log(1+\mu)}$$

- sign(x)> 0 als x>0 en sign(x)<0 als x<0
- x_{max} : absolute waarde van maximale amplitude van x-signaal
- $0 \leq |x/x_{max}| \leq 1$
- $x = 0 \Rightarrow y = 0$
- $-x_{max} <= y(x) <= x_{max}$
- Kwantisatie met 8 bit
- $\mu = 100$ of 255



μlaw vs linear

- μ = 100
- 3 bit kwantisatie
- q(z) kwantisatie
- e(x) = x – q(x)
- Van q(y) naar q(x)
  - Expansie: exp(..)

Shannon's interpolatieformule

$$s(t) = T \sum_{n=-\infty}^{n=+\infty} s_n \frac{\sin 2\pi f_N (t - nT)}{\pi (t - nT)}$$

- Dit is een *interpolatie*formule
- Geeft signaalwaarde *s(t)* op elk tijdstip *t* uit monsters s_n die op afstanden *T* van elkaar liggen
- Nyquistfrequentie f_N is helft van bemonsteringsfrequentie $1/T$
- We kunnen herbemonsteren met nieuwe bemonsterings freq. $1/T'$ (filteren als 1/T' < 1/T)

$$s(kT') = T \sum_{n=-\infty}^{n=+\infty} s_n \frac{\sin 2\pi f_N (kT' - nT)}{\pi (kT' - nT)}$$

- Sound: Resample... (n is interpolation depth)

Audiobestandsformaten (audiofiles)

Audiobestanden zijn <u>zelfbeschrijvend</u> of niet:

- ik ben audiofile type “X”
- gegevens over de audio in het bestand

In de audio-file moet minimaal info staan over:

- Bemonsteringsfrekwentie
- Aantal bits / monster
- Aantal kanalen (mono, stereo, ...)
- (Uitwisselbaarheid: little endian, big endian)

Audiofiles (2)

- .aiff .aifc Audio Interchange File Format,
  ontwikkeld door Apple.
- .wav een (slechtere) kopie van AIFC,
  "ontwikkeld" door Microsoft en IBM
- .au Sun's audioformaat. Ondersteund een groot aantal types zoals
  A-law, μlaw...
- .mp3 : MPEG audiolayer 3, compressieformaat (tot 10:1,
  gepatenteerd)
- tientallen andere formaten
  Zie o.a. http://www.euro.net/ecompany/audioformats.html

Ogg Vorbis audiofiles

- .ogg : open compressieformaat

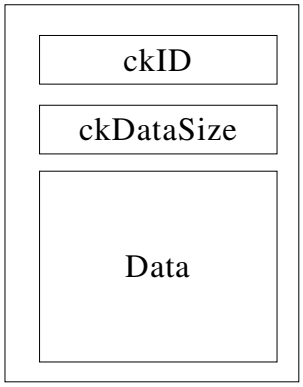
  Uit de folder:
  "Ogg Vorbis is a new audio compression format. It is roughly comparable to other formats used to store and play digital music, such as MP3, VQF, AAC, and other digital audio formats.
  It is different from these other formats because it is completely free, open, and unpatented."

  "... Vorbis doesn't just try to sound better, it tries to do things fundamentally better in all the ways that it can."

  (From: http://www.vorbis.com/)

Audiofiles (voorbeeld: AIFC)

- Bestanden bestaan uit zelfbeschrijvende brokken (chunks) en een container brok
- Elke brok bevat info om naar de volgende brok te springen (ckDataSize)
- Een bestand met brokken is uitbreidbaar (multimedia!)



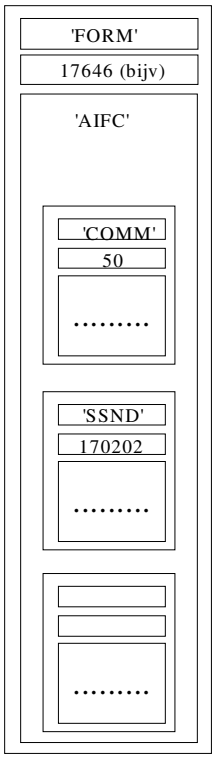
brok-kenmerk

#bytes in brok

de gegevens...

Regels voor lezen en schrijven van brokken

- Ordening van de brokken ligt niet vast
- Sla onbekende brokken over
- Schrijf onbekende brokken niet weg als er veranderingen in het bestand zijn aangebracht!
- Brokkenhiërarchie
  - FVER Format version
  - INST Instrument (MIDI)
  - COMT Comments
  - MARK Markers
  - APPL Application specific
  - ...



- identificatie aiff & aifc
- aantal bytes
- identificatie aifc

- COMMON brok met o.a.
  - lengte
  - bemonsteringsfrequentie

- Sound brok met audiomonsters
- ... andere brokken

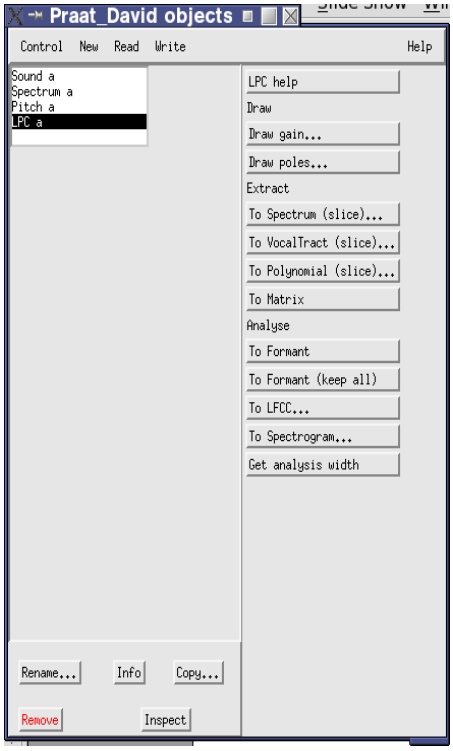
Hexadecimale dump van het begin van een aifc file:

```
[david@pc2 audio]$ od  -t xz iuay.aifc |more
4d524f46 70690200 43464941 52455646
>FORM..ipAIFCFVER<
04000000 405180a2 4d4d4f43 18000000  >....¢.
Q@COMM....<
01000100 10009834 44ac0d40 00000000
>....4...@.¬D....<
4f4e0000 0000454e 444e5353 38690200
>..NONE..SSND..i8<
```

PRAAT : intro

PRAAT is een computerprogramma voor analyse en synthese van spraaksignalen.

- Aanroep van PRAAT : dubbelklik op het praat-icoon
  - Er verschijnen twee vensters op het scherm:
    - Object-venster
    - Picture-venster
- Verlaten van PRAAT : Kies optie `Quit` uit het Control menu
- Hulp bij het programma:
  - Kies `Help` menu-optie en tik een zoekterm in
  - `Help`-knop op een formulier



PRAAT objecten

- Benoemde objecten zijn zichtbaar in de lijst van objecten
- Mogelijke acties zie je door een (of meer) object(en) te selecteren:
  - acties zonder formulier
    `To Matrix`
  - acties met formulier (altijd ...)
    `Draw gain...`
- Links onder: vaste menu met
  `Rename...`, `Info`, ...
- Rechts: dynamische menu

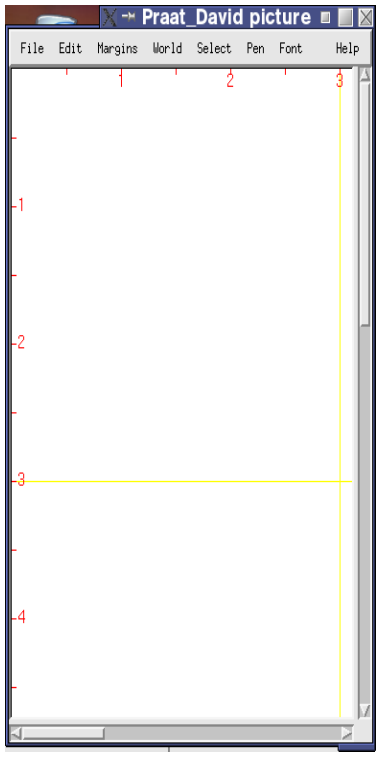
PRAAT objecten (vervolg)

- Objecten bestaan alleen in het RAM-geheugen
  Bij het beëindigen van PRAAT zijn ze weg!
- Bewaren van objecten: via `Write`-menu
  - `Write to short text file...`
  - `Write to binary file...`

  Voor `Sound`-objecten speciale formaten
- Creatie van objecten:
  - Via het `Read`-menu
    `Read from file...`
  - Via het `New`-menu
    `Create Sound...`
  - Via objectconversies
    `To Matrix`
  - Via analyses
    `To Formant`

PRAAT objecten (vervolg 2)

- `Query`-commando's veranderen het object niet
  - `Get number of samples`
  - `Get time form index...`
- `Modify`-commando's veranderen het object
  - `Formula...`
  - `Subtract mean`
  - `Override sample rate...`
- `Analyse`- en `Draw`-commando's veranderen ook niets aan het objec
  - `Draw...`
  - `To Spectrum`



PRAAT: Tekenen en het `Picture`-venster

- Selecteer de plaats waar je wilt tekenen
- Selecteer een tekenbaar object
- `Draw...`

- Verfijnen van tekening met menu-opties
  - `Margins`-menu
    `One mark left...`
    `Text top...`
  - `Pen`-menu
    - Lijn: dikte, type, kleur
- Bewaren als PostScript-bestand
  - File-menu:
    `PostScript Settings...`
    `Write to EPS file...`

PRAAT: Formules

Numerieke expressies (formules) mogelijk op veel plaatsen in PRAAT:

- in de `Calculator` in het `File/Goodies`-menu
  - `Calculator... 2^10`
- in de numerieke velden van de meeste formulieren
  - `Scale... 32767/32768`
    (`Sound`-object)
- in scripts
  - a = b + c
  - ...

Voor meer info zie de tutorial in PRAAT: `Help/Formulas`

PRAAT: `Formula...`

Voor sommige types objecten zoals `Sound` en `Matrix` kun je met `Formula...` een formule loslaten op elk element van de inhoud van het object:

- Modificatie van de inhoud van een `Sound`-object (via `self`)
  - `Formula... self*2`
    (vermenigvuldig elke monsterwaarde met 2)
- Vervanging van de inhoud van een `Sound`-object (geen `self`)
  - `Formula... 0.9*sin(2*pi*1000*x)`
    (maak een toontje van 1000 Hz)
- Referentie aan andere posities
  - `Formula... (self+self[col-1])/2`
    (plus een over één hokje verschoven versie)
- Referentie aan ander object
  - `Formula... Sound_s[nx+1-col]`
    (Keer geluid `s` om in de tijd: net als `Modify/Reverse`)

PRAAT: scripts (1)

Een script is een textfile met commando's voor PRAAT

- Een nieuw script via `Control/New Praat script`
  - `Run` of `Run selection` in de ScriptEditor
  - `Paste history`
- Geen nieuwe commando's: de scriptcommando's zijn gelijk aan de menucommando's (inclusief de ... voor de interactieve).
- Alle menucommando's beginnen dus met een Hoofdletter!
- Een regel kan verlengd worden door ... aan het begin volgende regel

```
Create Sound... sinus_met_ruis 0 1 22050
...  0.9*sin(2*pi*600*x) + randomGauss (0,0.1
```

- Commentaarregels beginnen met !

```
! deze regel kan in elk script, hij doet nix
```

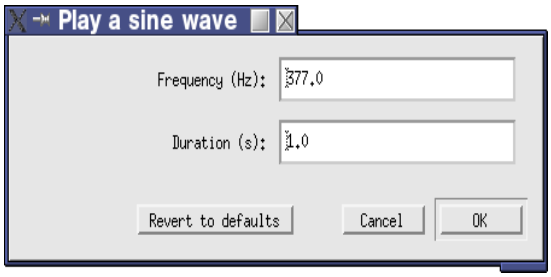
- Objectselectie via kleine-letter commando's
  - `select <objecttype> <naam>`
  - `plus  <objecttype> <naam>`
  - `minus <objecttype> <naam>`

PRAAT: scripts (2: Taal elementen)

- Variabelen (numeriek, string)
- Formules (numeriek, string)
- Jumps (for/endfor, while/endwhile, repeat/until)
- Procedures (procedure/endproc, call)
- Arrays
- Including other scripts
- Quit (exit)

PRAAT: scripts (3: voorbeeld)

```
form Play a sine wave
   positive Frequency_(Hz) 377.0
   positive Duration_(s) 1.0
endform
Create Sound... sine_'frequency' 0 duration
   ... 22050 0.9*sin(2*pi*frequency*x)
Play
Remove
```



PRAAT: scripts (4)

- Scripts kunnen andere scripts aanroepen
- Scripts kunnen andere scripts `include`ren
  (include wordt als eerste uitgevoerd, voor 'var' substitutie)
  `include "some_script.praat"`
- Gebruik system shell commando's
  `system cd /u/david; rm kanweg`
- Procedures

```
procedure play freq
    Create Sound... kanweg 0 1 22050
    ... sin(2*pi*'freq'*x)
    Play
    Remove
endproc
call play 500
call play 600
```

Opgave 2

a. Maak een sinustoontje van 440 Hz dat een halve seconde duurt bij een bemonsteringsfrequentie van 22050 Hz. Maak de maximale amplitude gelijk aan 0.99.
b. Bekijk en beluister het toontje in de `SoundEditor`. Maak evenlange selecties op twee verschillende manieren: begin- en eindcursor op nuldoorgangen en begin- en eindcursor op lokale maxima. Waarom klinkt het anders?
c. Zoom in en probeer de frequentie te meten.
d. Maak nu met behulp van `Formula...` alle negatieve amplitudes gelij aan nul. Je houdt dan een signaal over met alleen "bulten". Luister er naar.
e. We gaan uit van het oorspronkelijke toontje en gaan nu de negatieve amplitudes positief maken. Luister er naar.
f. Wat er gebeurt als een signaal geclipt wordt kunnen we simuleren met een formule als `if self < 'min' then 'min' else ... if self>'max' then 'max' else self fi fi` Neem `min=-0.5` en `max=0.5` en beluister het effect.

Fouriersynthese en Fourieranalyse

- Genoemd naar Baron Jean-Baptiste Fourier (1768-1830).
  Wiskundige, bestuurder en egyptoloog:
  Fourier et al., *Description de l'Égypte*, 21 vol. 1808-25.
  Na de val van Napoleon alleen een academisch leven (1815).
- 1822: *Théorie analytique de la chaleur*: een analyse van de warmtegeleiding in vaste lichamen in termen van sinussen en cosinussen.
- Nu één van de belangrijkste methodes:
  - synthese: een methode om spraaksignalen te *maken* door sinussen en cosinussen op te tellen.
  - analyse: een methode om spraaksignalen te *ontleden* in sinussen en cosinussen.

Fouriersynthese

We synthetiseren door het optellen van sinussen en cosinussen

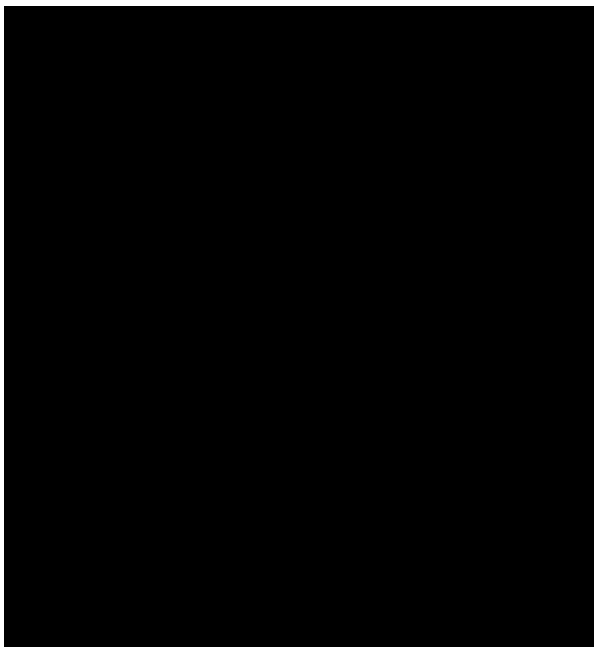
$$s(x)=\sum_{k=0}^{N}(a_k\cos(2\pi k f_0 x)+b_k\sin(2\pi k f_0 x))$$

Speciale gevallen:

$$\forall a_k=0\rightarrow s(x)=\sum_{k=0}^{N} b_k\sin(2\pi k f_0 x)$$

$$\forall b_k=0\rightarrow s(x)=\sum_{k=0}^{N} a_k\cos(2\pi k f_0 x)$$

Component Somsignaal Spectrum



$\sin (2\pi \bullet 1 \bullet 100 \bullet x) / 1$

+

$\sin (2\pi \bullet 3 \bullet 100 \bullet x) / 3$

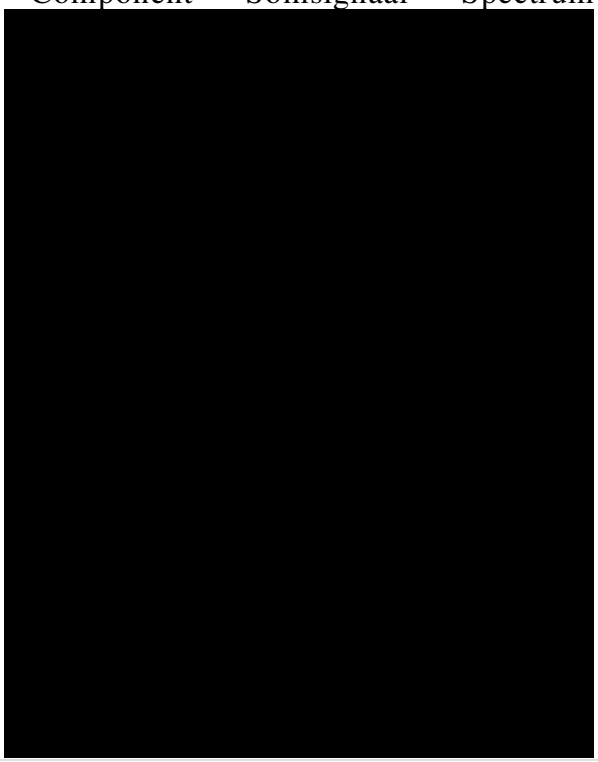
+

$\sin (2\pi \bullet 5 \bullet 100 \bullet x) / 5$

+

$\sin (2\pi \bullet 7 \bullet 100 \bullet x) / 7$

Component Somsignaal Spectrum



$\cos (2\pi \bullet 1 \bullet 100 \bullet x) / (1 \bullet 1)$

+

$\cos (2\pi \bullet 3 \bullet 100 \bullet x) / (3 \bullet 3)$

+

$\cos (2\pi \bullet 5 \bullet 100 \bullet x) / (5 \bullet 5)$

+

$\cos (2\pi \bullet 7 \bullet 100 \bullet x) / (7 \bullet 7)$

Somsignaal Spectrum

Som van 5 sinussen:

$$\sum_{k=1}^{5} b_k \sin(2\pi f_k x)$$

- $f_k = k \cdot 100$
- b_k = randomUniform in interval (0,1)

→ Periodiek

- f_k = randomUniform in interval (10,1000)
- b_k = als boven

→ Niet periodiek

Somsignaal Spectrum

Som van 5 cosinussen:

$$\sum_{k=1}^{5} a_k \cos(2\pi f_k x)$$

- $f_k = k \cdot 100$
- a_k = randomUniform in interval (0,1)

→ Periodiek

- f_k = randomUniform in interval (10,1000)
- a_k = als boven

→ Niet periodiek

Somsignaal

Spectrum

Som van 5 componenten:

$$\sum_{k=1}^{5} a_k \sin(2\pi f_k x) + \sum_{k=1}^{5} b_k \cos(2\pi f_k x)$$

- $f_k = k \cdot 100$ (harmonisch)
- $a_k = b_k$: random uniform in interval (0,1)
- Periodiek

- f_k = randomUniform in interval (10,1000)
- $a_k = b_k$ en als boven
- Niet periodiek

Alternatieve schrijfwijze voor som van sinussen en cosinussen

$a\cos x + b\sin x = c\sin(x+\phi)$, met fase ϕ

waarbij: $c = \sqrt{a^2+b^2}$

$\tan(\phi) = a/b$

Afleiding:

We kunnen altijd een r en een ϕ vinden zodat

$a = r\sin\phi$

$b = r\cos\phi$

Want dan is: $a^2+b^2 = r^2\sin^2\phi + r^2\cos^2\phi = r^2(\cos^2\phi+\sin^2\phi) =$

Dan krijgen we: $r = \sqrt{a^2+b^2}$, $\tan\phi = \sin\phi/\cos\phi = a/$

Herschrijven:

$a\cos x + b\sin x = r\sin\phi\cos x + r\cos\phi\sin x$

r buiten haakjes geeft: $r(\sin\phi\cos x + \cos\phi\sin x) = r\sin(x+$

Dit kunnen we toepassen op Fourier:

$$\sum_{k=0}^{N} (a_k \cos(2\pi k f_0 x) + b_k \sin(2\pi k f_0 x)) = \sum_{k=0}^{N} c_k \sin(2\pi k f_0 x + \phi_k)$$

$$c_k = \sqrt{a_k^2 + b_k^2}$$

$$\phi_k = \tan^{-1}(a_k / b_k)$$

Voor de synthese betekent dit dat we signalen die verschoven sinussen zijn, kunnen maken van "zuivere" sinussen en cosinussen door geschikt gekozen coefficienten a_k en b_k.

Samenvatting en "conclusies"

- Periodiciteit:
  - Som van sinussen met *harmonisch* gerelateerde frequenties is een *periodiek* signaal met grondfrequentie gelijk aan de afstand tussen de harmonischen (missing fundamental!)
  - Voor som van cosinussen geldt hetzelfde
  - En ook voor een som van sinussen en cosinussen
  - Som van niet harmonisch gerelateerde componenten is i.h.a. niet periodiek
- Symmetrie om de nul:
  - som van sinussen houdt sinussymmetrie (antisymmetrisch)
  - som van cosinussen houdt cosinussymmetrie (symmetrisch)

- Door geschikte keuze van frequenties en amplitudes lijken we elk signaal te kunnen maken uit een som van cosinussen en sinussen.

Opgave 3

Maak een praatscript waarmee je drie periodes van de driehoeksfunctie d genereert:

$$d(x) = \sin(2\pi 1 f_0 x)/(1.1) - \sin(2\pi 3 f_0 x)/(3.3) + \sin(2\pi 5 f_0 x)/(5.5) - \ldots$$

De k-de term in deze serie is:

$$(-1)^{k+1} \sin(2\pi(2k-1) f_0 x)/(2k-1)^2$$

Maak een serie van 8 grafieken op één pagina (2 kolommen van 4, zoals in het college, het spectrum hoeft niet).
Laat *k* varieren van 1 tot 4. Linker kolom toont de component, rechter kolom het signaal *d(x)*.
Hint: definieer variabelen `lx` en `ly` die de grootte van het gebied voor all grafieken definiëren. Bereken hieruit telkens de variabelen `vx1`, `vx2`, `vy` en `vy2` waarmee je de `Viewport` zet.

Fourieranalyse

- Via Fourieranalyse proberen we de frequentiecomponenten van een onbekend signaal te weten te komen. We proberen nu van een signaal te berekenen welke componenten, met welke sterkte, we nodig hebben om dit signaal zo goed mogelijk na te maken.

$$s(x) \approx \sum_{k=0}^{N} (a_k \cos(2\pi k f_0 x) + b_k \sin(2\pi k f_0 x))$$

- De componenten in de Fourieranalyse zijn de sinussen en cosinusser van de vorm $\sin(2\pi k f_0 x)$ en $\cos(2\pi k f_0 x)$ die we al bij de Fouriersynthese zijn tegengekomen.
- De sterktes van de componenten, de ak's en de bk's, kunnen we interpreteren als: "hoeveel lijkt het signaal op deze component".

"Lijken op" is dan zoiets als: leg de signalen over elkaar en kijk hoeveel ze overlappen. Hoe meer ze overlappen, hoe meer ze op elkaar lijken!

Bepaling van *overlap (s(x), b(x))*.

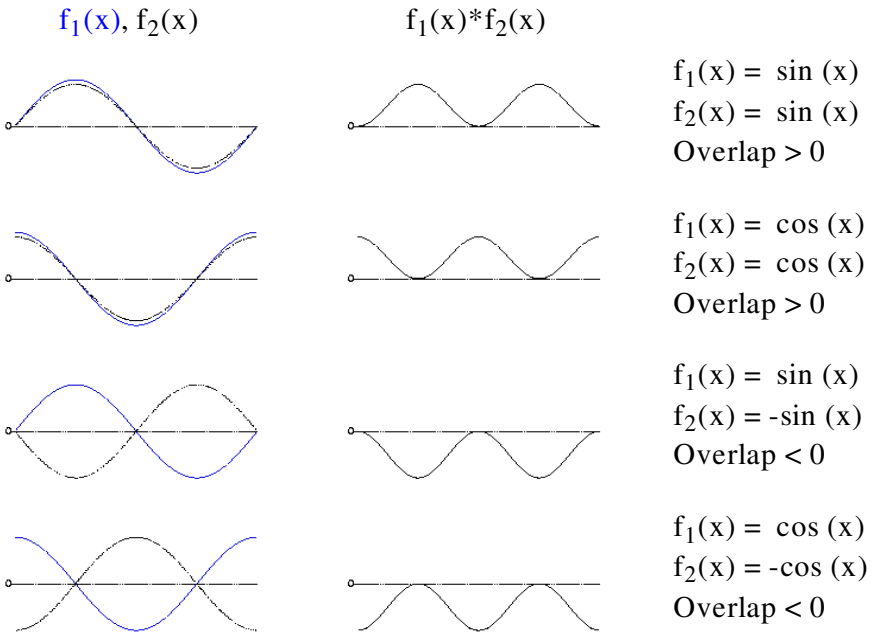
- Vermenigvuldig de twee signalen met elkaar
- Bepaal de oppervlakte boven en onder de nul-as
- Trek deze oppervlaktes van elkaar af
- Het verschil bepaalt de mate van overlap (lijken op)

Wiskundigen hebben een betere notatie bedacht dan *overlap (s,b)*: We schrijven voor de sterkte *b* waarmee basiscomponent *b(x)* aanwezig is in het signaal *s(x)* als

$$b = \int_{begin}^{end} s(x)\,b(x)\,dx$$

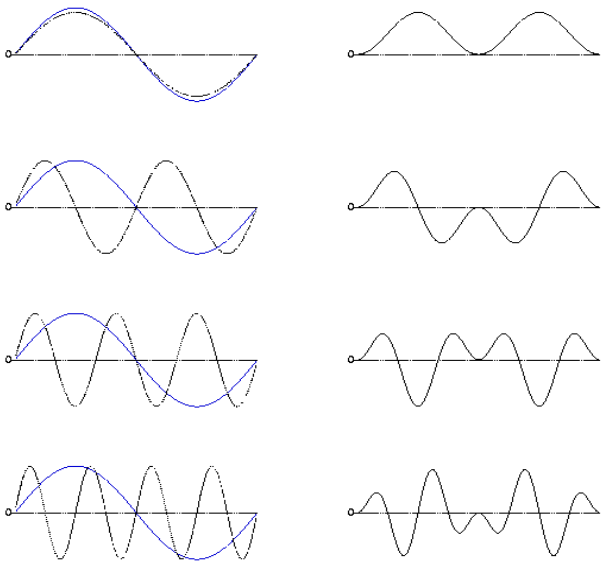
Hier staat niks anders dan: *vermenigvuldig s(x) en b(x) en bepaal de oppervlakte onder de productfunctie!*

Overlap sin(x) met sin(x) en cos(x) met cos(x)



Overlap sin(x) met sin(kx)

$f_1(x)$, $f_2(x)$ | $f_1(x)*f_2(x)$



$f_1(x) = \sin(x)$
$f_2(x) = \sin(x)$
Overlap > 0

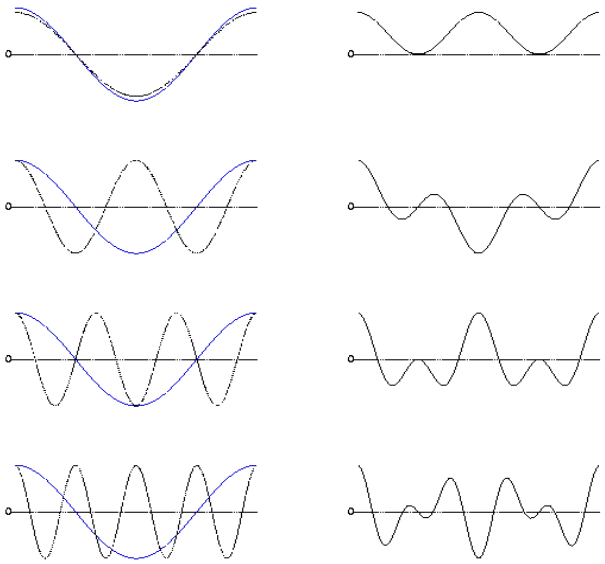
$f_1(x) = \sin(x)$
$f_2(x) = \sin(2x)$
Overlap = 0

$f_1(x) = \sin(x)$
$f_2(x) = \sin(3x)$
Overlap = 0

$f_1(x) = \sin(x)$
$f_2(x) = \sin(4x)$
Overlap = 0

Overlap cos(x) met cos(kx)

$f_1(x)$, $f_2(x)$ | $f_1(x)*f_2(x)$



$f_1(x) = \cos(x)$
$f_2(x) = \cos(x)$
Overlap > 0

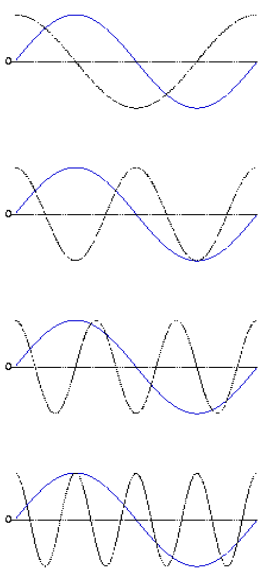
$f_1(x) = \cos(x)$
$f_2(x) = \cos(2x)$
Overlap = 0

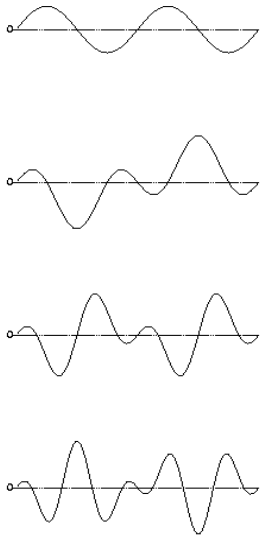
$f_1(x) = \cos(x)$
$f_2(x) = \cos(3x)$
Overlap = 0

$f_1(x) = \cos(x)$
$f_2(x) = \cos(4x)$
Overlap = 0

Overlap sin(x) met cos(kx)

$f_1(x)$, $f_2(x)$ | $f_1(x)*f_2(x)$





$f_1(x) = \sin(x)$
$f_2(x) = \cos(x)$
Overlap = 0

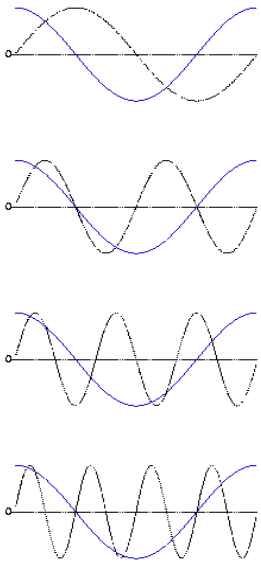
$f_1(x) = \sin(x)$
$f_2(x) = \cos(2x)$
Overlap = 0

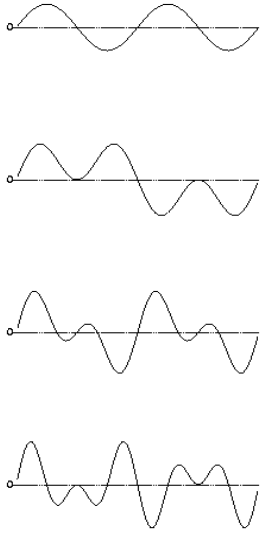
$f_1(x) = \sin(x)$
$f_2(x) = \cos(3x)$
Overlap = 0

$f_1(x) = \sin(x)$
$f_2(x) = \cos(4x)$
Overlap = 0

Overlap cos(x) met sin(kx)

$f_1(x)$, $f_2(x)$ | $f_1(x)*f_2(x)$





$f_1(x) = \cos(x)$
$f_2(x) = \sin(x)$
Overlap = 0

$f_1(x) = \cos(x)$
$f_2(x) = \sin(2x)$
Overlap = 0

$f_1(x) = \cos(x)$
$f_2(x) = \sin(3x)$
Overlap = 0

$f_1(x) = \cos(x)$
$f_2(x) = \sin(4x)$
Overlap = 0

Conclusie

$$\int_0^{2\pi} \sin(mx)\sin(nx)\,dx = 0, \text{als } n \neq m, \text{anders} > 0$$

$$\int_0^{2\pi} \cos(mx)\cos(nx)\,dx = 0, \text{als } n \neq m, \text{anders} > 0$$

$$\int_0^{2\pi} \sin(mx)\cos(nx)\,dx = 0$$

Voor n=1,2,... en m = 1, 2, ...
- de functies *sin(nx)* een orthogonale verzameling (⇒ sin ($2\pi nf_0x$).
- de functies *cos(nx)* ook.
- de functies *sin(nx)* en *cos(mx)* zijn onderling orthogonal.

Hoe vinden we de a_k's en b_k's in:

$$s(x) \approx \sum_{k=0}^{N} (a_k \cos(2\pi k f_0 x) + b_k \sin(2\pi k f_0 x))$$

Vermenigvuldig, integreer en gebruik de orthogonaliteit

$$\int_{x_1}^{x_2} s(x)\cos(2\pi k f_0 x)\,dx \approx a_k$$

$$\int_{x_1}^{x_2} s(x)\sin(2\pi k f_0 x)\,dx \approx b_k$$

Het spectrum is dan een plaatje van frequentie tegen de (logaritme van de) amplitudes: $\log\sqrt{a_k^2 + b_k^2}$

Het uitrekenen van een integraal

Voor het bepalen van de sterkte van een component kwamen we een uitdrukking tegen van de vorm:

$$b = \int_{begin}^{end} s(x)\, b(x)\, dx$$

Deze uitdrukking betekent: bepaal de oppervlakte van het product van s(x) met b(x) over het interval [begin, end]. Stel we noemen dit product g(x), dan kunnen we noteren:

Een integraal bepaalt de oppervlakte onder een functiecurve.
Van een bemonsterd signaal kunnen we de integraal benaderen:

$$\int_a^b s(x)\, dx \approx \sum_{i=1}^{N} s(x_i)\, dx = T \sum_{i=1}^{N} s(x_i),$$

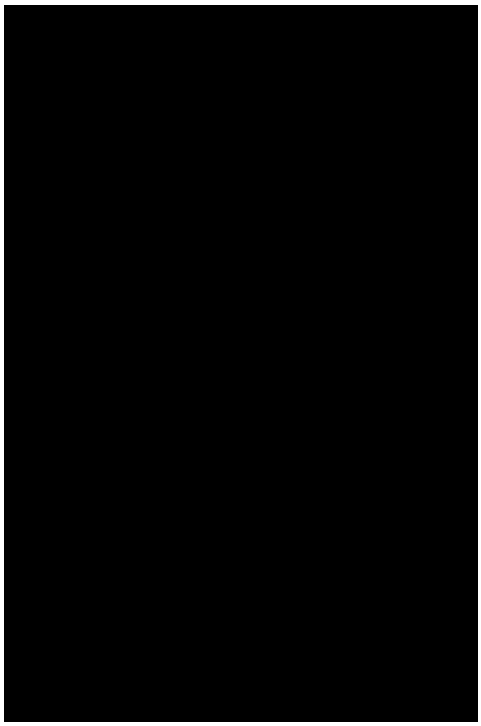
bij voldoende kleine dx (= T, de bemonsteringstijd)

Het gemiddelde van een bemonsterd signaal *f* is:

gemiddelde = (som van alle monsterwaardes) / (aantal monsterwaardes)

$$= \sum_{i=1}^{N} f(x_i)/N$$

Op een schalingsfactor na is de integraal van een signaal over een interv
gelijk aan het gemiddelde van het signaal over het interval!



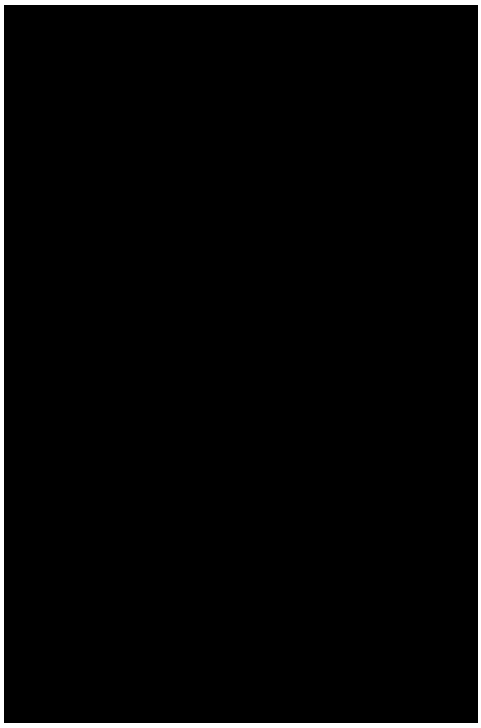
FA voorbeeld 1

sin $(2\pi x)$

Linker kolom: *cos (kx)
Rechter kolom: *sin (kx)

Alleen component bij 1:
sin: 0.5

Schaalfactor: 2



FA voorbeeld 2

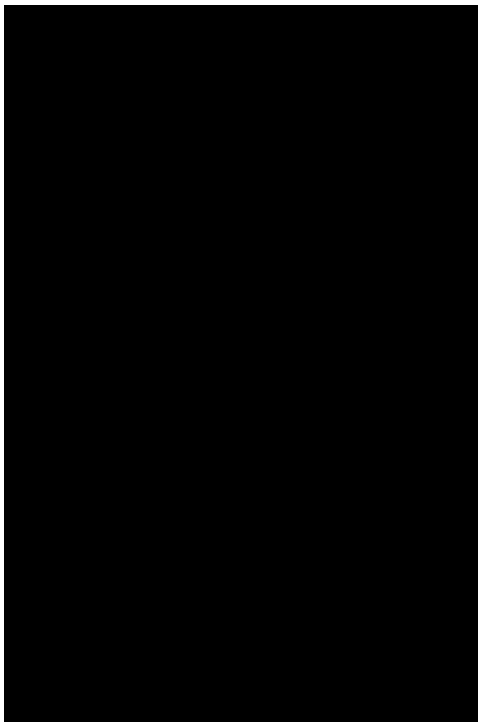
$\sin(2\pi x+\pi/3)$

Linker kolom: *cos (kx)
Rechter kolom: *sin (kx)

Alleen component bij 1:
cos: 0.433
sin: 0.25

$$c=\sqrt{0.433^2+0.25^2}\approx 0.5$$
$$\tan\phi=0.433/0.25\rightarrow\phi=\pi/3$$

Schaalfactor: 2



FA voorbeeld 3

$\sin(2\pi x+\pi/3)+4\sin(2\pi 3x)$

Linker kolom: *cos (kx)
Rechter kolom: *sin (kx)

Alleen component bij 1:
cos: 0.433
sin: 0.25
en bij 3:
sin: 2

$$c=\sqrt{0.433^2+0.25^2}\approx 0.5$$
$$\tan\phi=0.433/0.25\rightarrow\phi=\pi/3$$

Schaalfactor: 2

Fourieranalyse: conclusies

- Om de sterktes van de componenten te krijgen de overlap bepalen met basisfuncties sin(kx) en cos (kx)
- Het spectrum is dan de log (amplitude)
- Fourier transformatie paar (signaalanalyse definitie!):

$$H(f) = \int_{-\infty}^{+\infty} h(t)\, e^{-2\pi i f t}\, dt$$

$$h(t) = \int_{-\infty}^{+\infty} H(f)\, e^{2\pi i f t}\, df$$

Where,

$$e^{2\pi i f t} = \cos 2\pi f t + i \sin 2\pi f t$$

- Kortere complexe notatie in plaats van aparte sin en cos componenten.
- H(f) bestaat uit twee componenten (re:cos, im:sin)

Complex numbers

Euler: $e^{i\phi} = \cos\phi + i \sin\phi$

Two dimensions

If $h(t)$ is...	then...
real	$H(-f)=H^*(f)$
imaginary	$H(-f)=-H^*(f)$
even	$H(-f)=H(f)$ $H(f)$ is even
odd	$H(-f)=-H(f)$ $H(f)$ is odd
real and even	$H(f)$ is real and even
real and odd	$H(f)$ is imaginary and odd
imaginary and even	$H(f)$ is imaginary and even
imaginary and odd	$H(f)$ is real and odd

Fouriertransformatie paren

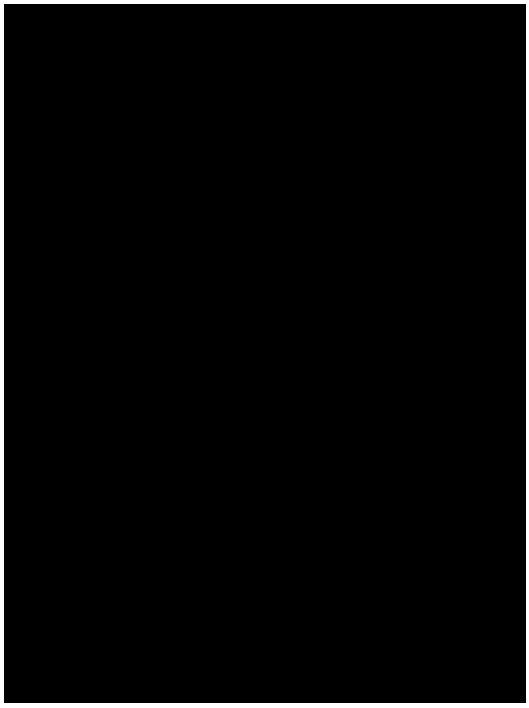
$h(t) \Leftrightarrow H(f)$ het basis paar

$h(at) \Leftrightarrow \frac{1}{|a|} H(\frac{f}{a})$ tijd schaling

$\frac{1}{|b|} h(\frac{t}{b}) \Leftrightarrow H(bf)$ frequentie schaling

$h(t-t_0) \Leftrightarrow H(f) e^{-2\pi i f t_0}$ tijd translatie

$h(t) e^{2\pi i f t_0} \Leftrightarrow H(f-f_0)$ frequentie modulatie



```
Create Sound... s 0 0.01
  ... 22050
  ... 0.5*sin(2*pi*1000*x
```

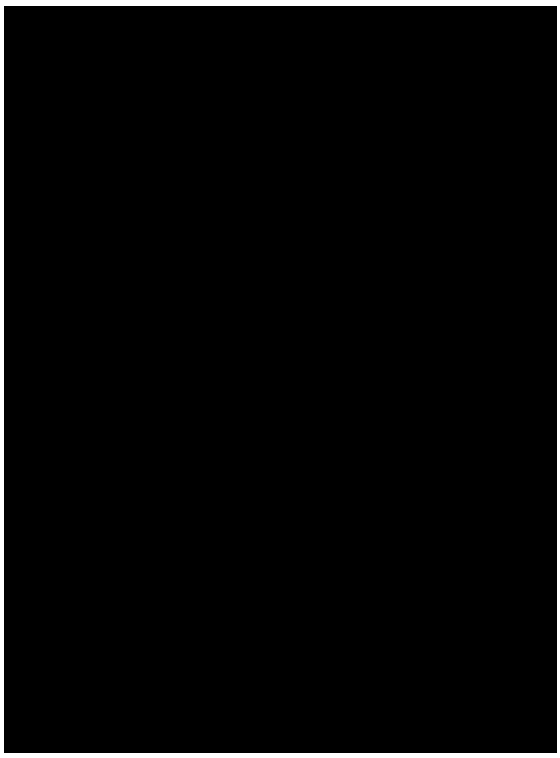
- Draw 0 0 -1 1 y
- in 0.01s 10 periodes

```
To Spectrum (fft)
```

- Draw... 0 0 0 70 y
- One mark bottom...
  ... 1000 y y y

spectrum heeft piek bij 1000 Hz
en deze verwachten we.

- waar komen de andere pieken vandaan?



Frequentie resolutie versus (eindige) signaalduur

- Links : signaal (sin 1000 Hz)
  - 1, 2, 4, 8, 16, 20.000 periodes
- Rechts: spectrum (0-2000Hz)
  - sync -vorm: sin(x)/x
  - Ho e korter het signaal hoe "breder" het spectrum
  - In de limiet (1 puls) dan recht spectrum met alle frequenties even sterk aanwezig.
  - In de andere limiet (heel lange meettijd) één puls in het spectrum
- Zekerheid over frequentie kost tijd!

Fourier analyse van een bemonsterd signaal: DFT

Discrete Fourier Transformatie geeft Spectrum

$$H_n = \sum_{k=0}^{N-1} h_k e^{-2\pi i n/N} \Leftrightarrow h_n = 1/N \sum_{k=0}^{N-1} H_k e^{2\pi i n/N}$$

Hoeveel berekeningen zijn nodig om het spectrum te bepalen?

Elke spectrale component H_n gebruikt O(N) vermenigvuldigingen
(#vermenigvuldigingen ~ N)
N spectrale componenten gebruiken dan O(N^2) vermenigvuldigingen
(#vermenigvuldigingen ~ N^2)

Totale aantal berekeningen O(N^2)
Als N verdubbelt, dan wordt het aantal berekeningen 4x zo veel,
als N verdrievoudigt, dan aantal berekeningen 9x zoveel, etc...

De FFT (Fast Fourier Transform)

- Een snellere manier om het spectrum te berekenen
- Nadeel: aantal samples moet macht-van-2 zijn
- Aantal berekeningen dan: O($N\ ^2\log N$)
  ($^2\log x = y \ \ 2^y = x$)
- Maakt dat wat uit vergeleken met DFT's O(N^2) ?

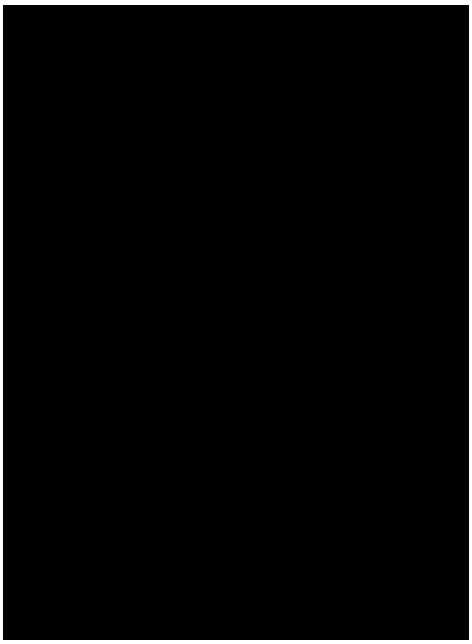
1 µs per elementaire berekening (1990), N = 10^6
O(N^2): $10^6 \bullet 10^6 \bullet 1$ µs = 10^6 s ≈ 11.5 dag
O($N\ ^2\log N$): $10^6 \bullet 20 \bullet 1$ µs = 20 s

1 ns per elementaire berekening(2003), N = 10^6
O(N^2): $10^6 \bullet 10^6 \bullet 1$ ns = 10^3 s ≈ 17 minuten
O($N\ ^2\log N$): $10^6 \bullet 20 \bullet 1$ ns = 0.02 s

Sound: To Spectrum (fft)

- Copiëer Sound.nx samples naar FFT-buffer
- Als aantal samples geen macht-van-2 is maak lengte FFT-buffer dan gelijk volgende macht-van-2
  (bufferlengte is nu macht-van-2, stel 2^p. Dan:
  Sound.nx $\leq 2^p < 2 \bullet$Sound.nx)
- Bereken spectrum volgens FFT methode
- Copiëer $2^p/2 + 1$ cosinus en sinus delen naar het Spectrum (z[1] en z[2])
- Het Spectrum-object heeft dan:

xmin	0
xmax	F_c = 1 / Sound.dx / 2 (Nyquist frequency)
nx	$2^p/2 + 1 = 2^{p-1} + 1$
dx	$F_c / 2^{p-1}$
z[1]	Cosinus deel (real)
z[2]	Sinus deel (imaginary)



- Dit is het signaal
- Zo is het in de FFT-buffer
- Een basis functie met dezelfde frequentie past wel maar niet aan het eind (extra's nodig!)
- Het benodigde "venster"

Excercise 5

Random noise

a. low pass 1000 Hz
b. high pass 1000 Hz
c. bandpass filter 330-3000
d. bandstop filter 330-3000

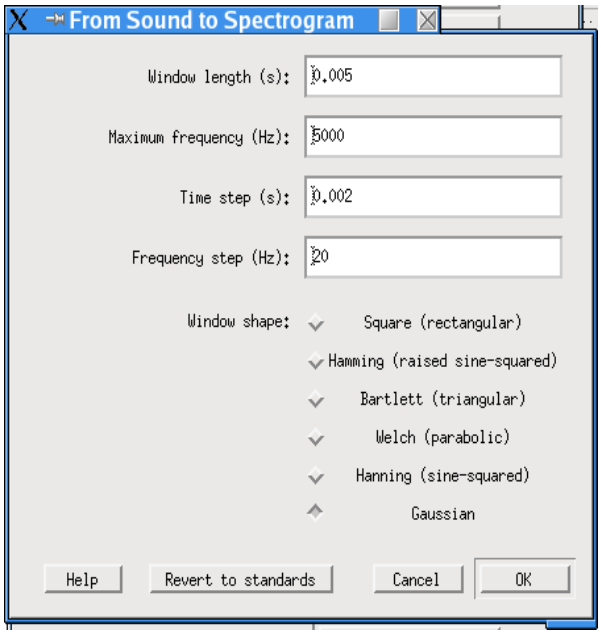
Het spectrogram(-object)

Het Spectrogram-object representeert een acoustische tijd-frequentie representatie van een geluid: de *power spectral density P* (*f*, *t*), uitgedrukt in Pa^2/Hz.

Het is bemonsterd in punten die op gelijke afstanden van elkaar liggen, en gecentreerd zijn om tijdstippen t_i en frequenties f_j.

Populair: "de sterkte van frequenties als functie van de tijd".
Deze "sterkte" wordt aangegeven via zwarting: hoe zwarter hoe sterker

Een doorsnede van een analyse-object in de tijd wordt een *frame* genoemd.



Sound: To Spectrogram...

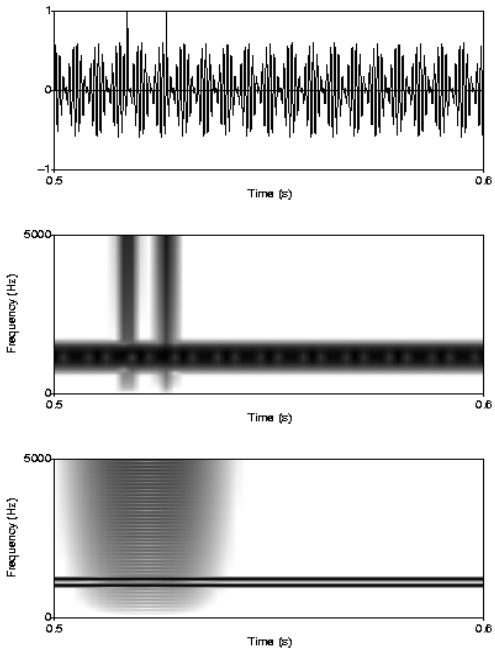
Help / Spectral analysis

Sound: To Spectrogram...

- *Window length* (s)
  the duration of the analysis window. If this is 0.005 seconds, PRAAT uses for each frame the part of the sound that lies between 0.0025 seconds before and 0.0025 seconds after the centre of that frame (for Gaussian windows, Praat actually uses a bit more than that).
  The window length determines the *bandwidth* of the spectral analysis.
  To get a broad-band spectrogram (bandwidth 260 Hz), set *Window length* to 5 milliseconds; to get a narrow-band spectrogram (bandwidth 43 Hz), set it to 30 milliseconds.

Sound: To Spectrogram...

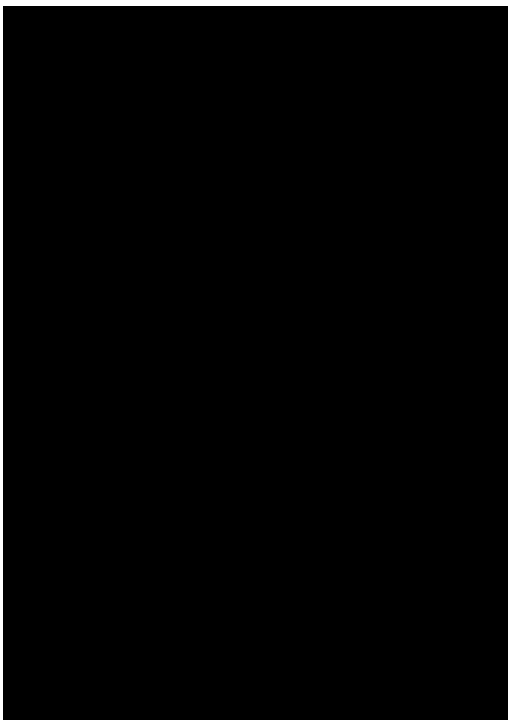
- *Maximum frequency* (Hz)
  the maximum frequency subject to analysis, e.g. 5000 Hertz. If it is higher than the Nyquist frequency of the Sound (which is half its sampling frequency), some values in the result will be zero (and will be drawn in white by **Spectrogram: Paint...**
- *Time step* (s)
  the distance between the centres of subsequent frames, e.g. 0.002 seconds. This determines the number of frames of the resulting Spectrogram. For instance, if the Sound is 1 second long, and the time step is 2 milliseconds, the Spectrogram will consist of almost 500 frames (not *exactly* 500, because no reliable spectrum can be measured near the beginning and end of the sound).



Create Sound... bn 0 1 11025
... 0.3*(sin(2*pi*1000*x) +
... sin(2*pi*1200*x)) +
... (col=5700) + (col=5800)

"Broad band"
Window length 0.005 s

"Narrow band"
Window length 0.030 s



Rectangular

Gaussian **

Hamming

Hanning

Triangular

Parabolic

Opgave 6: spectrale meting aan Nederlandse klinkers (1)

We gaan aan de klinkers van het Nederlands meten aan de hand van de gemaakte opnames. Het werkt het beste als je de klinkers in één of twee bestanden hebt staan (gescheiden door stilte natuurlijk).
We beginnen met het selecteren van de segmenten waarop we analyses willen uitvoeren. De selectie gebeurt door *labelen*. Het object dat we hiervoor gebruiken heet `TextGrid`. Door een `TextGrid`- en een `Sound`-object samen te selecteren en `Edit` te kiezen kunnen we gaan labelen (op een interval-tier):

```
select Sound <je-geluid>
To TextGrid... klinkers
plus Sound <je-geluid>
Edit
```

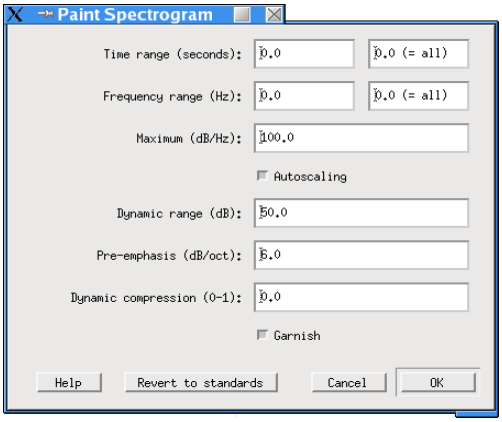
Lees de hulp in het `TextGridEditor`-object. We markeren van elk van de 12 Nederlandse klinkers een interval van ongeveer 30 ms in het midden van de klinker. Bewaar de TextGrid-objecten als bestand op je harde schijf!

Opgave 6: spectrale meting aan Nederlandse klinkers (2)

Maak een Spectrum (fft) van de 12 segmenten en teken deze op twee bladzijdes (6 per pagina). Zet de klinkernaam er boven.

Opmerkingen:
• Omdat het TextGrid-object opgelijnd moet kunnen worden met het Sound-object om te kunnen editten, hebben we het vanuit het geluid gemaakt.
• Het TextGrid-object krijgt maar één tier, namelijk een interval-tier om te labelen. Een interval-tier bestaat uit *aaneensluitende* intervallen.
• Het labelen zelf kun je niet automatiseren. Je kunt er over denken om het maken en tekenen van de spectra te automatiseren door een script.

Spectrogram: Paint... (1)



Spectrogram: Paint... (2)

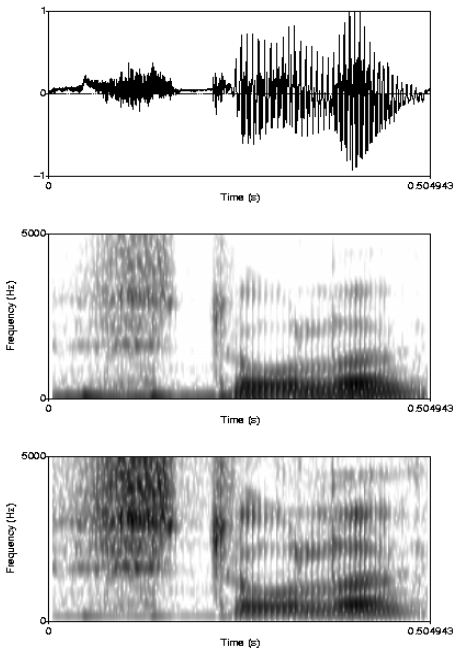
- *From time*, *To time* (seconds)
  the time domain along the *x* axis.

- *From frequency*, *To frequency* (Hertz)
  the frequency domain along the *y* axis.

- *Dynamic range* (dB)
  The global maximum of the spectrogram (after preemphasis) will always be drawn in black; all values that are more than *Dynamic range* dB below this maximum (after dynamic compression) will be drawn in white. Values in-between have appropriate shades of grey.

Spectrogram: Paint... (3)

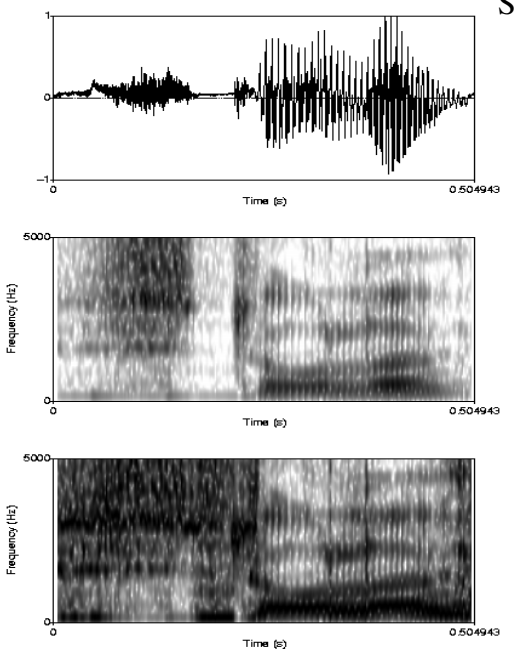
- *Preemphasis* (dB/octave)
  determines the steepness of a high-pass filter, i.e., how much the power of higher frequencies will be raised before drawing, as compared to lower frequencies. Since the spectral slope of human vowels is approximately -6 dB per octave, the standard value for this parameter is +6 dB per octave, so that the spectrum is flattened and the higher formants look as strong as the lower ones.
- *Dynamic compression*
  determines how much stronger weak time frames should be made before drawing. Normally, this parameter is between 0 and 1. If it is 0, there is no dynamic compression. If it is 1, all time frames (vertical bands) will be drawn equally strong, i.e., all of them will contain frequencies that are drawn in black. If this parameter is 0.4 and the global maximum is at 80 dB, then a frame with a maximum at 20 dB (which will normally be drawn all white if the dynamic range is 50 dB), will be raised by 0.4 * (80 - 20) = 24 dB, so that its maximum will be seen at 44 dB (thus making this frame visible).



Spectrogram: Paint / Preemphasis

Sound is het Volendams 'stoelen'.
To Spectrogram... 0.005 5000 0.02 ... 20 Gaussian
Paint... 0 0 0 0 100 yes 50 *x* 0 yes

Zonder en met preemphasis *x*:
x = 0, +6 dB/octave



Spectrogram: Paint /dyn. compression

Sound is het Volendams 'stoelen'
To Spectrogram... 0.005 5000 0.02 ... 20 Gaussian
Paint... 0 0 0 0 100 yes 50 6 *z* yes

Voor het middelste plaatje is de dynamic compression *z*=0, voor het onderste plaatje is de compressie maximaal: *z*=1.

Opgave 7

Maak smalleband en breedband spectrogrammen van je klinkers.

Pitch

- Inconsistentie: term pitch voor *stimulus* parameter (frequentie) en voor *perceptie*.
- Psychoacoustic ANSI definitie:
  Pitch is that auditory attribute of sound according to which sounds can be ordered on a scale from low to high.
- Pitch perceptie is gecompliceerd.
- Alleen een sinustoon heeft steeds dezelfde pitchsensatie tot gevolg bij normaal-horende luisteraars.
- Experimentele verificatie: presenteer twee opeenvolgende toontjes, een vast, de ander variabel. Laat veel luisteraars de frequentie van het tweede toontje aanpassen. Herhalen tot vervelens toe.
  Het resultaat is een distributie bestaande uit *één* piek.
- Complexe signalen geven *meer*-piekige distributies.

Meer info via Terhart:
http://www.mmk.ei.tum.de/persons/ter/top/defpitch.html

Pitch analyse.

Veel verschillende analysemethodes. Alle goede methodes geven altijd meerdere kandidaten.

- Analyses in het *frequentie-domein*: vanwege de harmonische structuur in het spectrum: harmonische zeef, cepstrum, kam-filters, ...
- Analyses die de *auditieve periferie modelleren*: gehoorgang, auditieve filterbank, neurale transductie, spike-generatie, interval-detectie, generatie van kandidaten...
- Analyses in the *tijd-domein*: Auto-correlation, inverse filtering...
  Boersma, P. (1993), "Accurate short-term analysis of the fundamental frequency and the harmonics-to-noise ratio of a sampled sound", IFA Proceedings 17, 97-110.

We zullen ons bezig houden met Boersma's *autocorrelatie* methode omdat deze zo voortreffelijk werkt.

Periodiciteit: het vinden van de kandidaten

De periodiciteitsdetectie methode bestaat uit twee delen:

1. Het vinden van de *frequenties* en de *sterktes* van de kandidaten voor pitch in elk analyseframe.

2. Het vinden van de optimale kandidaat in elk frame. Dit komt neer op het vinden van een optimaal pad in de kandidatenruimte.

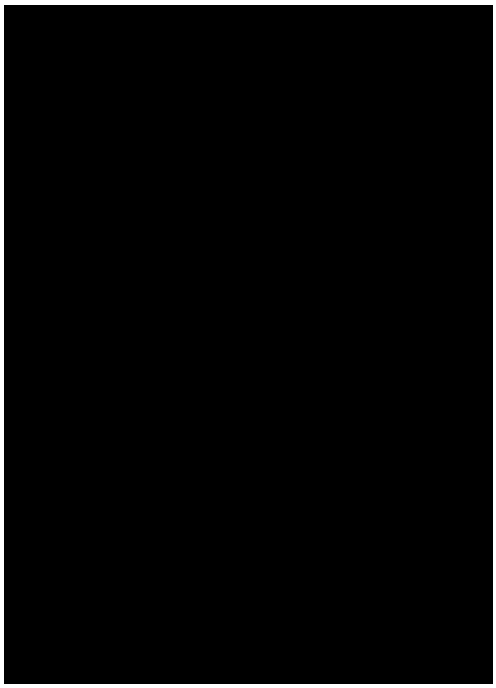
Autocorrelatie

$r(\tau)=\int x(t)\,x(t+\tau)\,dt$ autocorrelatie

$r'(\tau)=\frac{r(\tau)}{r(0)}$ genormaliseerd

Bepaalt de overlap van het signaal met een in de tijd verplaatste versie van zichzelf. Deze tijdsverplaatsing τ heet in het Engels *lag*.
De functie $r(\tau)$ kan elke waarde tussen $-\infty$ en $+\infty$ aannenemen.
De autocorrelatie is maximaal als de lag τ=0.
Dan geldt voor de genormaliseerde autocorrelatie: $-1 \leq r'(\tau) \leq 1$.

Als het signaal $x(t)$ periodiek is met periode T dan heeft $r'(\tau)$ pieken die op lags $\tau = kT$ (k = ..., -2, -1, 0, 1, 2, ...)



Autocorrelatie van sin 100 Hz:

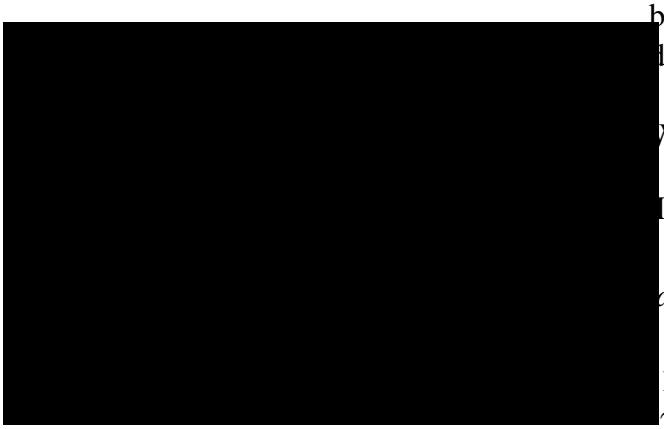
```
Create Sound... sin100 0 6.5536
    ... 10000 0.9*sin(2*pi*100*x)
Copy... sin100c
plus Sound sin100
Cross-correlate...  -0.1 0.1 yes
```

Alle pieken dezelfde hoogte!

Pieken in de autocor functie op intervallen van 0.01 s correspond-eren met frequenties van 100, 50, 33.33, 25, ... Hz (f_k = 100 / k)

Negatieve pieken: negatieve correlaties

Harmonics to Noise Ratio (HNR)



Als er ruis aanwezig is dan bereiken de pieken in r' niet de waarde 1.

We definiëren:

$$HNR = 10\log\left(\frac{r'(\tau_{max})}{1-r'(\tau_{max})}\right)$$

dB	r'
0	0.5
10	0.91
20	0.99
30	0.999
40	0.9999
50	0.99999

Vereist extreem nauwkeurige amplitude bepaling!

x(t) multiplied by *w(t)* gives *a(t)*

Hallo, verkeerde amplitude

Da's beter

Periodiciteit 2: De padvinder (1)

Het probleem: stel we hebben N frames, elk frame heeft M kandidaten.
Dit levert M^N mogelijke paden (combinaties).
Exponentieel probleem $O(M^N)$.

Voorbeeld: $M = 10$ kandidaten/frame in $N = 100$ frames geeft 10^{100} combinaties.
Met snelle computers anno 2003: 10^{10} instructie/s hebben we ongeveer 10^{90} s $\sim 3 \cdot 10^{82}$ jaar nodig om alle combis te testen.
(leeftijd universum $\sim 25 \cdot 10^9$ jaar of ieder deeltje 300 jaar (10^{80}))

We hebben iets beters nodig: Viterbi-algoritme $O(N \cdot M)$
We moeten dan *kosten* introduceren en het optimale pad heeft dan minimale kosten. We hebben te maken met kosten *binnen* een frame en kosten voor overgangen *tussen* frames.

Periodiciteit 2: De padvinder (*binnen* één frame)

Sterkte van een kandidaat:
Bijdrage in de kosten: - $r'(\tau_{max})$

Niet genoeg: in een perfekt periodiek signaal hebben alle kandidaten dezelfde sterkte. De *hoogste* frequentie moet winnen!

Cost parameter 1: *octaveCost*
Bijdrage: $octaveCost \cdot {}^2\log(minimumPitch \cdot \tau_{max})$
Gemoduleerde signalen $(1+d_{mod} \cdot \sin 2\pi ft) \cdot \sin 2\pi 2ft)$ hebben akoestische pitch f en perceptieve pitch $2f$ als $d_{mod} \leq 0.2, 0.3$. Zet deze grens met $octaveCost=d_{mod}^2$

Netto kosten: $octaveCost \cdot 2\log(minimumPitch \cdot \tau_{max}) - r'(\tau_{max})$

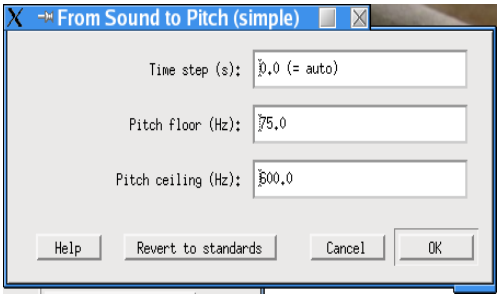
Periodiciteit 2: De padvinder (overgangen *tussen* frames)

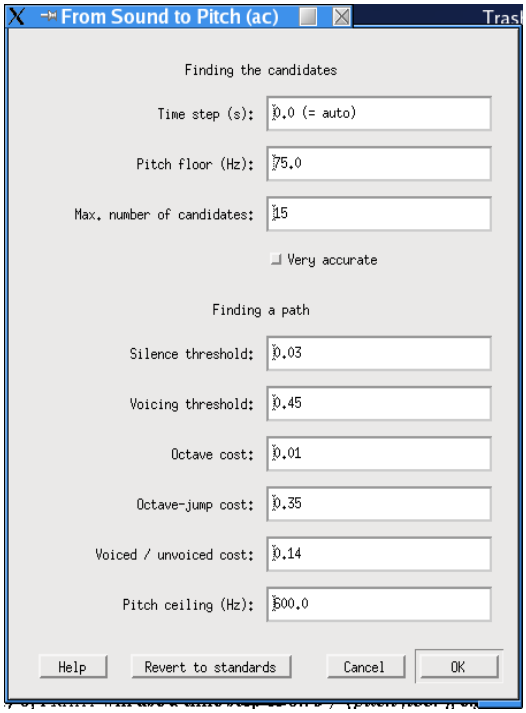
Cost parameter 2: *voicedUnvoicedCost*
Cost parameter 3: *octaveJumpCost*

$$transitionCost(f_1, f_2) = \begin{cases} 0 & \text{if } f_1 = 0 \text{ and } f_2 = 0 \\ voicedUnvoicedCost & \text{if } f_1 = 0 \text{ xor } f_2 = 0 \\ octaveJumpCost \cdot \left| {}^2\log \frac{f_1}{f_2} \right| & \text{if } f_1 \neq 0 \text{ and } f_2 \neq 0 \end{cases}$$

$f = 0$ betekend unvoiced (stemloos)!

Hoe groter *voicedUnvoicedCost*, hoe minder stemhebbend/stemloos overgangen zullen optreden (en vice versa).
Uitschakelen van de kosten tussen frames door *voicedUnvoicedCost* en *octaveJumpCost* nul te maken.





- *Time step* (standard value: 0.0)
  the measurement interval (frame duration), in seconds. If you supply 0, PRAAT will use a time step of 0.75 / (pitch floor), e.g. 0.01 s if the pitch floor is 75 Hz; in this example, PRAAT computes 100 pitch values per second.
- *Pitch floor* (standard value: 75 Hz)
  candidates below this frequency will not be recruited. This parameter determines the length of the analysis window: it will be 3 longest periods long, i.e., if the pitch floor is 75 Hz, the window will be 3/75 = 0.04 seconds long.
  Note that if you set the time step to zero, the analysis windows for consecutive measurements will overlap appreciably: PRAAT will always compute 4 pitch values within one window length, i.e., the degree of *oversampling* is 4.
  A post-processing algorithm seeks the cheapest path through the candidates. The argument that determines the cheapest path is:
- *Pitch ceiling* (standard value: 600 Hz)
  candidates above this frequency will be ignored.

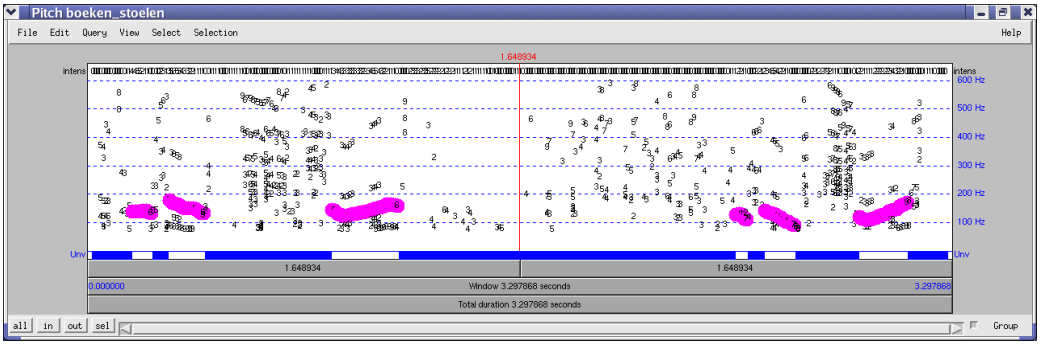
Wat kunnen we doen met een Pitch-object?

- Play
- Edit
- Query
- Draw
- Info
- To
- Pitch & TextGrid: Draw...
- Manipulation

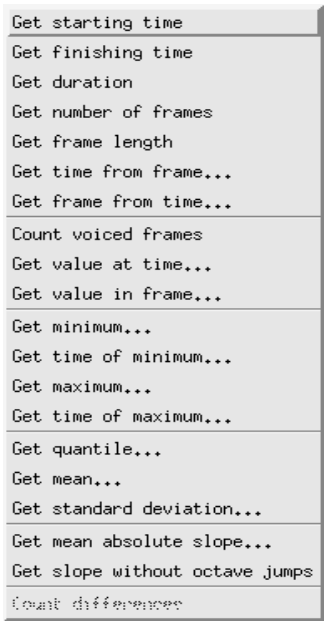
  Volendams: boeken-stoelen

Pitch: Edit

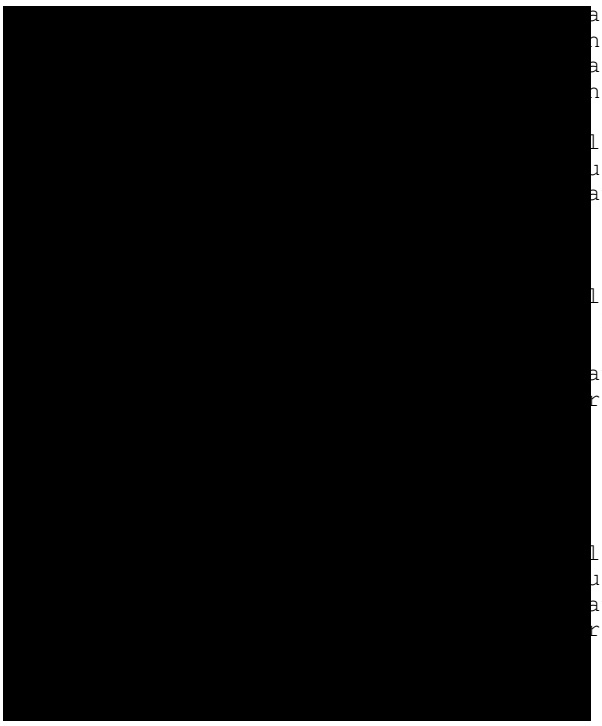
- Pitch lokaal aanpassen
- Pitch neurieën



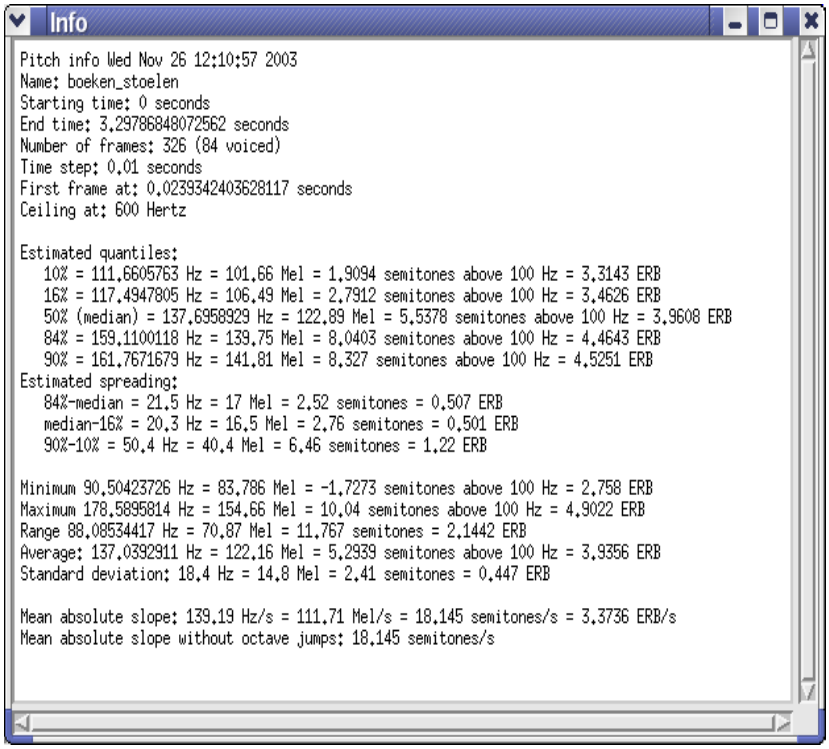
Pitch: Query



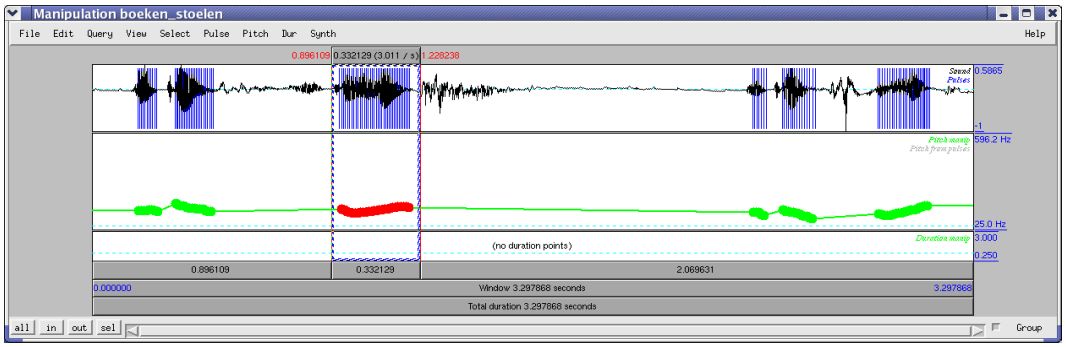
Pitch: Draw



```
ad from file... <textgrid>
name... vol
ad from file... <sound>
name... vol

lect Sound vol
us TextGrid vol
aw...  0 3.3 y y y


lect Sound vol
 Pitch... 0 75 500

aw...  0 3.3 0 300 y
rks left every... 1 50 y y y


lect Pitch vol
us TextGrid vol
aw... 1 0 3.3 0 300 14 y y
rks left every... 1 50 y y y
```

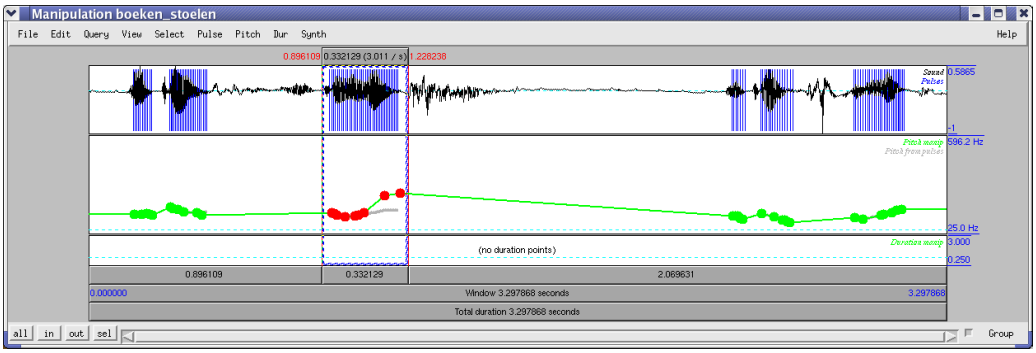


Pitch: Info




```
select Sound vol
To Manipulation... 0.01 75 600
Edit
   Select... 0.89 1.23
   Stylize pitch... 2 Hz
   <modify intercatively some pitch points>
```

Pitch manipulated in ManipulationEditor



Oefening 8: Toonhoogtemanipulatie

Probeer van een drietal opgenomen zinnetjes de toonhoogte zo te manipuleren dat ze vragend klinken.

Tips:

- Maak eerst een Manipulation-object met `Sound: To Manipulation`
- Open de ManipulationEditor en lees de Help.
- Speel met de pitchpunten in het PitchTier-venster en beluister het resultaat. Bijvoorbeeld, selecteer een woord en maak alles (shift-knop) hoger of lager in frequentie.

- Om te manipuleren is het handig om eerst te stileren: dwz dat je probeert om de individuele pitchpunten op de "pitchcontour" te vervangen door een aaneenschakeling van rechte lijnstukjes (zo min mogelijk). De manipulatie van de toonhoogte via deze schakelpunten is minder werk.

Wat zijn formanten?

- Productie: resonanties van de mond-keelholte
- Perceptie: relatieve intensiteit in het excitatie patroon
- Akoestisch: maxima in het spectrum

Wij houden ons aan de akoestische definitie en beschouwen formanten als maxima in het spectrum.

Een formant kunnen we simuleren door een exponentieel gedempte sinus

$$s(t) = e^{-\pi Bt} \sin(2\pi Ft)$$

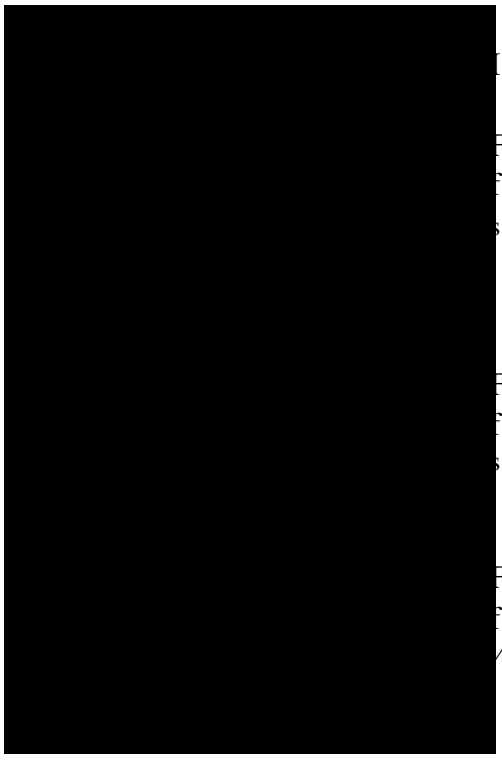
Formant frequentie F en bandbreedte B.

reate Sound... ds 0 0.05 11025
. exp(-50*pi*x)*sin(2*pi*500*x)
raw... 0 0 -1 1 y
o Spectrum (fft)
raw... 0 2000 0 0 y

ormant frequentie: 500 Hz
andbreedte: 50 Hz

Formantfrequentiemetingen

- Vanaf het signaal op het oog (Volendams)
- Vanaf het spectrum en het spectrogram op het oog
- Automatisch via lineaire predictie



Interactie patronen tussen sinussen

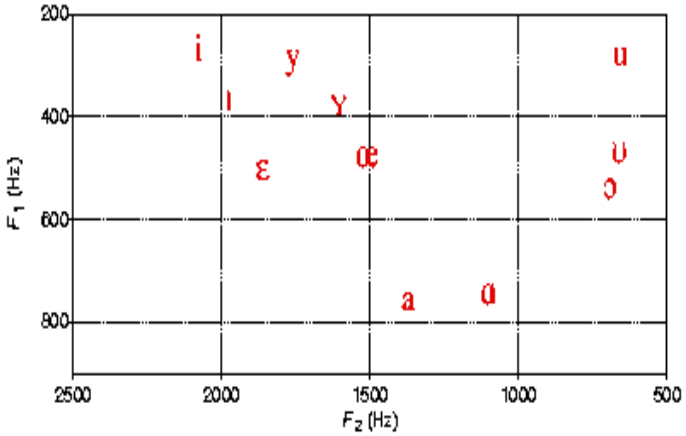
Frequentie/amplitude relatie 1:2,
fase (0 - π/2)
$\sin(2\pi 300x)+\frac{1}{2}\sin(2\pi 600x + \frac{1}{2}\pi x/xmax)$

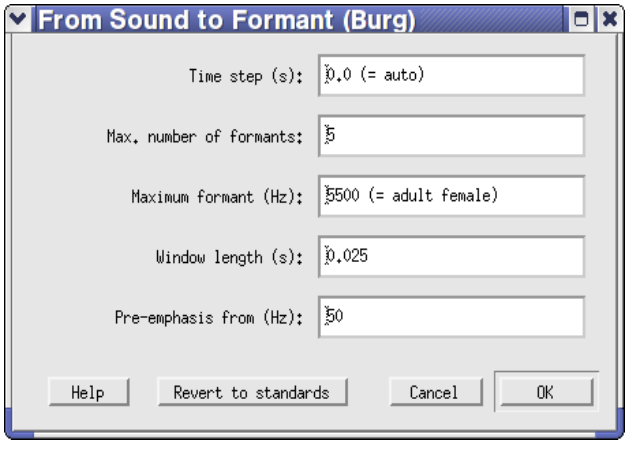
Frequentie/amplitude relatie 1:3,
fase (0 - π/2)
$\sin(2\pi 500x)+1/3\sin(2\pi 1500x + \frac{1}{2}\pi x/xmax)$

Frequentie/amplitude relatie 2:3,
fase (0 - π/2)
$\frac{1}{2}\sin(2\pi 800x)+1/3\sin(2\pi 1200x + \frac{1}{2}\pi x/xmax)$

Dialect van Volendam

select u, ɛ, ah, œ





• *Time step* (seconds)
the time between the centres of consecutive analysis frames. If the sound is 2 seconds long, and the time step is 0.01 seconds, there will be approximately 200 analysis frames.
The actual number is somewhat lower (usually 195), because we cannot measure very well near the edges. If you set the time step to 0.0 (the standard), PRAAT will use a time step that is equal to 25 percent of the analysis window length (see below).

• *Maximum number of formants*
for most analyses of human speech, you will want to extract 5 formants per frame. This, in combination with the *Maximum formant* argument, is the only way in which this procedure will give you results compatible with how people tend to interpret formants for vowels, i.e. in terms of vowel height (F1) and vowel place (F2).

• *Maximum formant* (Hertz)
the ceiling of the formant search range. It is crucial that you set this argument to a value suitable for your speaker. The standard value of 5500 Hz is suitable for an average adult female. For a male, use 5000 Hz; if you use 5500 Hz for an adult male, you may end up with too few formants in the low frequency region, e.g. analysing an [u] as having a single formant near 500 Hz whereas you want two formants at 300 and 600 Hz. For a young child, use a value much higher than 5500 Hz, for instance 8000 Hz (experiment with it on steady vowels).

• *Window length* (seconds)
the effective duration of the analysis window. The actual length is twice this value, because PRAAT uses a Gaussian-like analysis window with sidelobes below -120 dB.
For instance, if the *Window length* is 0.025 seconds, the actual Gaussian window duration is 0.050 seconds. This window has values below 4\\% outside the central 0.025 seconds.

• *Pre-emphasis from* (Hertz)
the +3 dB point for an inverted low-pass filter with a slope of +6 dB/octave.
If this value is 50 Hz, then frequencies below 50 Hz are not enhanced, frequencies around 100 Hz are amplified by 6 dB, frequencies around 200 Hz are amplified by 12 dB, and so forth.
The point of this is that vowel spectra tend to fall by 6 dB per octave; the pre-emphasis creates a flatter spectrum, which is better for formant analysis because we want our formants to match the local peaks, not the global spectral slope.

Exercise 9: Formantfrequentiemetingen

Selecteer via de Sound en TextGrid-editor een plek in de klinker waar je formantfrequenties kunt meten (gebruik ook de weergave van het spectrogram om een stabiel punt te zoeken). Markeer dit punt in een PointTier. Rapporteer de eerste drie formanten.

Voor degenen die willen: er valt heel wat te automatiseren als je eenmaal de meetposities hebt gevonden. Enkele hints:

```
Sound: Resample... <10000/11000> 50
Sound: To Formant...
TextGrid: Get number of tiers
TextGrid: Get number of intervals... <tier>
TextGrid: Get label of interval... <tier> <int>
TextGrid: Get label of point... <tier> <point>

for i to npoints
...
endfor
```

Bron-filter model voor spraakproductie

- Bron
  - Stemhebbend: stembanden (e, a, m...)
  - Stemloos: ruis via vernauwing in mond-keelholte (f, s, p, ...)
- Filter: mond-keelholte werkt als resonator.

Source Filter Lips sound

(Source: L. Vogten: Basistechniek in de signaalanalyse: LPC)

- Helling van het bronspectrum ongeveer -12 dB/octaaf

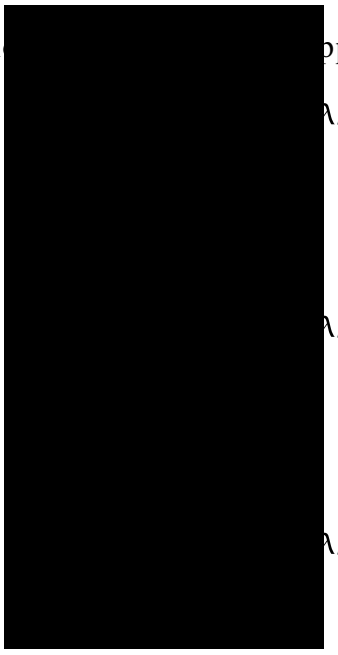
- Uitstraling bij de mond is dusdanig dat lage frequenties meer spreiden
  hoge frequenties. Hoge frequenties zijn meer gericht.
  Dit heeft als resultaat dat een microfoon een kleiner deel van het
  vermogen van de lage tonen registreert dan van de hoge tonen.
  Lage frequenties worden dus relatief verzwakt geregistreed.
  Dit (filter)effect heeft een extra helling van +6dB in het filterspectru

- De resulterende helling in het geluidsspectrum is dan -12+6= -6dB/oct

Bron-filter model (3 klinkers, 2 toonhoogtes)

(Source: L. Vogten: Basistechniek in de signaalanalyse: LPC)

De schwa-buis



$$\lambda f = c$$

$$f = (2k+1)\frac{c}{4 \cdot vtl}$$

Volwassen man:

vocal tract length = 0.17 m
c = 340 m/s

f = 500, 1500, 2500, ...

Lineair predictie analyse

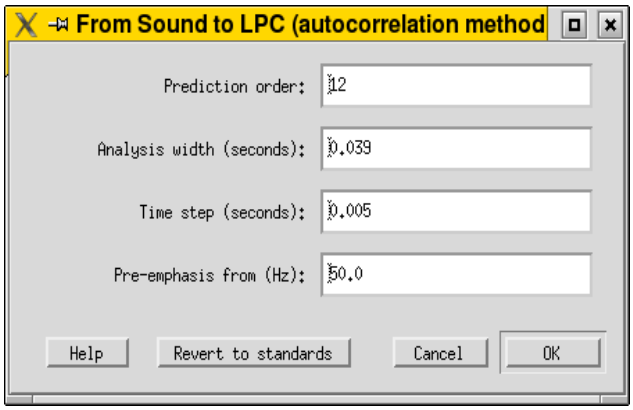
Techniek die sinds eind jaren 70 in het spraakanalysedomein wordt gebruikt.

Gebaseerd op het bron-filter model.

Ook wel: LPC-analyse (Linear Predictive Coding)

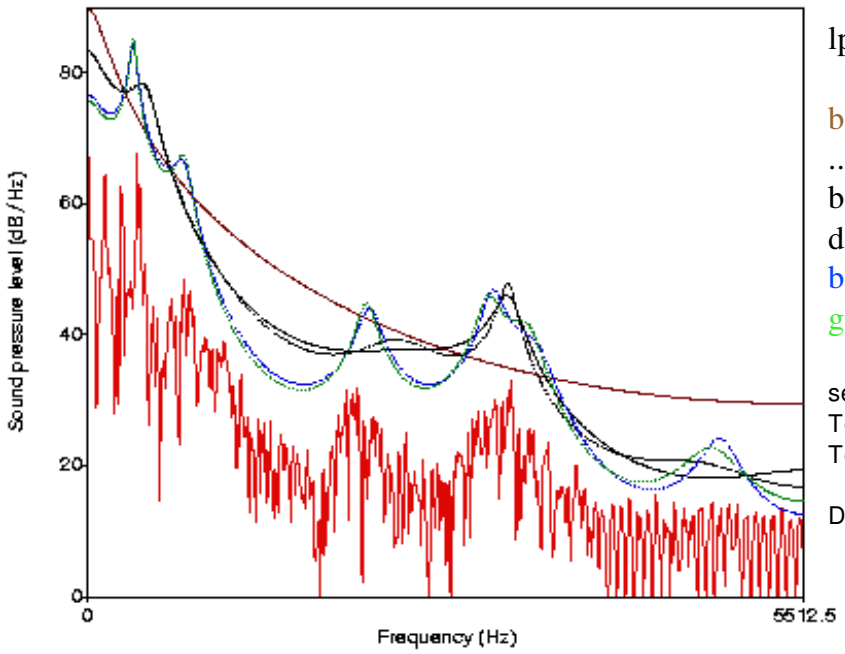
Via LPC proberen we het spectrum zo goed mogelijk te benaderen via een aantal formant-pieken.

Sound -> Formant is eigenlijk Sound -> LPC -> Formant
Het aantal formanten moet je kiezen



aantal coëfficiënten = prediction order ≥ 2 x gewenste aantal formanten !
Elke 2 coëfficiënten definiëren samen één formant.

Let op: voor LPC-analyse op spraaksignalen eerst herbemonsteren op 5000/5500 Hz (M/V) (Resample... 5000)



lpc-order:

bruin: 2
...
black: 8
dotted:10
blue: 12
green: 14

select Sound
To LPC ()...
To Spectrum
...(slice)
Draw...

Bron-filter en LPC

Spraak(f) = Filter(f) * Bron(f)

Alternatieve formuleringen:

- Filter(f) = Spraak(f) / Bron(f)
- Bron(f) = Spraak(f) / Filter(f) (Inverse filter)

LPC-object bevat filtercoëfficiënten van het "inverse filter".

Hoe komen we aan de ontbinding van Spraak(f) in Filter(f) en Bron(f)?
(Bedenk dat we bv 20 op oneindig veel manieren als een product
kunnen schrijven: 20 = 20*1, 10*2, 5*4, 6*3.66, ...)

Aanname: het spectrum van de bron, Bron(f), is zo recht mogelijk
Dit is voor twee extremen het geval: witte ruis en pulssignaal (stemloos en stemhebbend).