Fonem => Bil dilde kelimeyi oluşturan ve daha küçük parçalara ayrılamayan ses öğelerinin her birine fonem denir.

Sinyal işleme ses sinyallerinin yapısını incelemektedir.

Ses dili oluşturan en küçük birimdir.

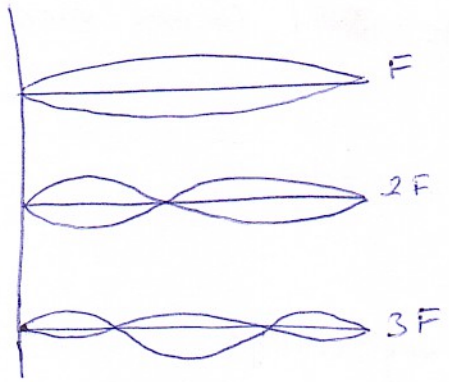
15°C'lik hava içerisinde sesin hızı 340 m/s olmaktadır.

Sesin oluşumu

Edels → insan sesinin oluşumunda başlıca dört olayın yer aldığını belirtmiştir. Sırayla

1) İnisiyasyon (Patlama)
2) Vibrasyon (titreşim)
3) Rezonans
4) Artikülasyon

Ses bir bileşik titreşim hareketidir. Periyot ve genlikleri farklı.



Temel ses dalgaları armonikler.

Bir titreşimin frekansı, hareketin kendi kendine özdeş olarak, bir saniyedeki yineleme sayısıdır.

Saniyedeki bu titreşim sayısı Hz (hertz)'le ölçülmektedir.

Benzer tekrarlanır titreşimlerle izlenen uyumlu eğriler oluşturan basit ses dalgalarına ton denir.

Frekans ve tınıdan sonra sesin ölçülen özelliği şiddetidir.

Sesin şiddeti onun enerjisi ile ilişkilidir.

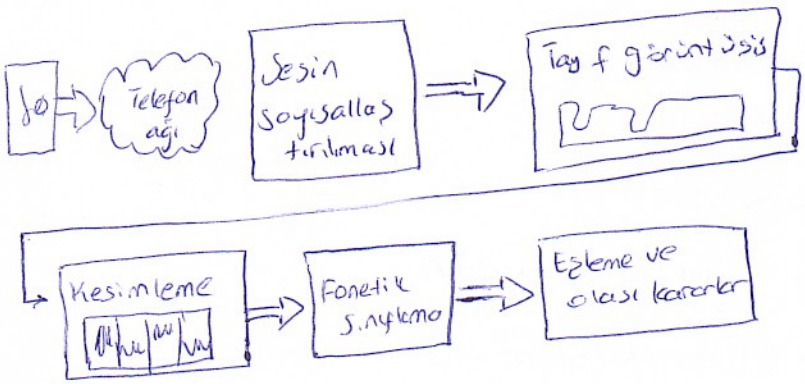
Bir sesin enerjisi ise onun watt cinsinden ölçülen gücüyle etkide bulunduğu yüzey alanı arasındaki ilişkiye bağlıdır. Dolayısıyla enerji $1cm^2$'ye gelen [illegible] watt cinsinden (w/cm^2) ölçülür.

Aksan => Temel tonun frekansının değişmesine denir.

Sert sesler yüksek, yumuşak sesler ise düşük yoğunluktadır.

* Konuşmanın dinamik aralığı 35-45 dB'dir.
* Ünlü seslerin sürekliliği ortalama 0.15s
* ünsüzlerin ise 0.08s'dir.

Sesin tanınması =>



Konuşmacı Tanıma => Gelen ses sinyaline göre konuşanın kimliği tanımaya çalışılır.

Anahtar sözcük yakalama => Bu sistemlerde kalıp kelimeler aranmaktadır.

Ses bilgi mikrofon gibi dönüştürücüler yardımıyla elektriksel işarete çevrilebilir. Sonra analog bir bilgi olan ses, sayısal tekniklerin daha kolay gerçekleştirebilmesi ve hızlı gerçekleneler sağlayabilmeleridir.

Mono => Tek kanal yardımıyla gerçekleştirilen kayıtlar mono, iki kanal kullananlar ise stereo olarak adlandırılır.

Hızlı Fourier Dönüşümü (Fast Fourier Transform FFT)

mikrofona gelen ses gürültüler içermektedir. Beyaz gürültülere daha çok görülür. Beyaz gürültünün frekansı ana bileşene göre çok yüksektir, ve iki komşu değerin salınım yapan eksene olan uzaklıkları farkı sıfır olmaktadır.

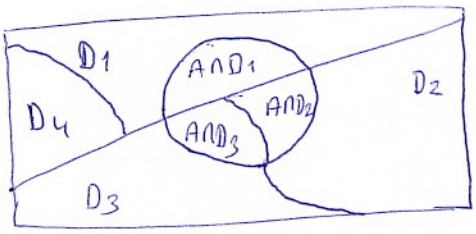
$$x(x) = C_0 + \sum_{k=0}^{\infty} \left(a_k \cos(k\omega_1 t) + b_k \sin(k\omega_1 t) \right)$$

ayrık işaret (zaman)

C_0, a_k ve b_k ise Fourier katsayılarını ifade etmektedir.

$\omega_1 = 2\pi f_0$ $\quad f_0 \rightarrow$ temel frekans

Bayes Kuralı



$A = (A \cap D_1) \cup (A \cap D_2) \cup \dots \cup (A \cap D_n)$

A olayının gerçekleşmesi olasılığı

$P(A) = P(A \& D_1) + P(A \& D_2) + \dots + P(A \& D_n)$

$P(A \& D) = P(A|D)P(D)$ olduğunda;

$$P(A) = P(A|D_1)P(D_1) + P(A|D_2)P(D_2) + \dots + P(A|D_n)P(D_n) = \sum_{i=1}^{n} P(A|D_i)P(D_i)$$

$\Rightarrow$ Bu sonuca toplam olasılık teoremi denir.

$P(A \& D) = P(A|D)P(D)$ olduğundan her hanki bir k değeri için.

$$P(D_k|A) = P(D_k \& A)/P(A)$$

$$P(D_k|A) = P(D_k)P(A|D_k)/P(A)$$

P(A) için toplam olasılık teoreminden faydalanarak;

$$P(D_k|A) = \frac{P(D_k)P(A|D_k)}{\sum_{i=1}^{n} P(D_i)P(A|D_i)}$$ $\Rightarrow$ Bayes teoremi

$y \Rightarrow$ yarın yağmurun olması
$B \Rightarrow$ Bugünkü hava
} Bayes kuralı $P(y|B) = P(B|y)P(y) / P(B|y)P(y) + P(B|-y)P(-y)$

Eğer $P(y|B)$ biliniyorsa Bayes kuralı kullanılır.

Bayes teoreminin, tüm kombinasyonların hesaplanmasına dayalı yöntemden farkı değerler bir biri ile korelasyon oluşturduğundan tüm durumları denemeden karar verme imkanının olmasıdır.

Veriler temelinde bilgisayarın kendini "geliştirerek" yeni bilgiler kazanması bilgisayarlı öğrenme (machine learning) denilir.

Saklı Markov Modeli (HMM) Baum ve meslektaşları tarafından 1970'li yılların başlarında geliştirilmiştir.

Yöntemin temelini (t-1) anındaki durum bilindiğinden t anındaki durumun değerlendirilmesi mantığı oluşturmaktadır.

$$P(t) = A * P(t-1)$$

$\Rightarrow$ Eğitme aşamasında ise kelimenin modeli oluşturan M ilişkisi bulunur

$$M = [N, P(1), A, B]$$

N $\Rightarrow$ Durum sayısı P(1) $\Rightarrow$ Başlangıç durum A $\Rightarrow$ geçiş matrisi B $\Rightarrow$ istenilen duru

Think DSP

Digital signal Processing in Python

! Nump - SciPy

Extensively → yoğun olarak

Assume ⇒ varsaymak / sanmak

* Think Python
* Mark Lutz's Learning Python

* Numpy for basic numerical computation
* Scipy for scientific computation
* Matplotlib for visualization

Sounds and signals

A signal represents a quantity that varies in time.

Sound is variation in air pressure. A sound signal represents variations in air pressure over time.

→ The frequency of a signal is the number of cycles per second, which is the inverse of the periodic.

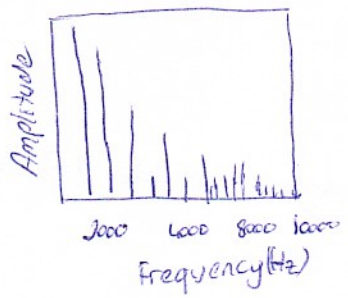
The units of Frequency are cycles per second, or Hertz. (kısaltılan) abbreviated 'Hz'

The shape of periodic signal is called the waveform.

Discrete Fourier Transform (DFT) (Ayrık Fourier dönüşümü)

Bir sinyali alır ve spektrumunu üretir.

Fast Fourier transform or FFT which is an efficient way to compute the DFT



A wave represent a signal evaluated at a sequence of points in time

Each point in time is called a frame

```
wave = mix.make_wave(duration=0.5, start=0, framerate=11025)
```

Duration is the length of the wave in seconds.

Frame rate is the (integer) number of frames per second, which is also the number of samples per second.

_ Signal objects _

Sinusoid is a child class of Signal, with this definition

```
class Sinusoid(Signal):
    def __init__(self, freq=440, amp=1.0, offset=0, func=np.sin):
        Signal.__init__(self)
        self.freq = freq
        self.amp = amp
        self.offset = offset
        self.func = func
```

The parameters of __init__ are:

freq => frequency in cycles per second or Hz.

Amp => Amplitude. The units of amplitude are arbitrary, usually chosen so 1.0 corresponds to the maximum input from a microphone or maximum output to speaker.

offset: indicates where in its period the signal ~~at a particular point in time~~ starts.

func => A python function used to evaluate the signal at a particular point in time. it is usually either np.sin or np.cos yielding a sine or cosine signal.

Framerat is the number of frames (samples) per second.

```
def make_wave(self, duration=1, start=0, framerate=11025):
    n = round(duration * framerate)        # n number of samples
    ts = start + np.arange(n)/framerate    # ts is an Numpy array of sample times.
    ys = self.evaluate(ts)
    return Wave(ys, ts, framerate=framerate)

def _evaluate(self, ts):
    phases = PI2 * self.freq * ts + self.offset
    ys = self.amp * self.func(phases)
    return ys
```

self.freq → saniye başına döngü cinsinden frekanstır. frequency in cycles per second

self.func => is np.sin or np.cos, the result is a value between -1 +1.

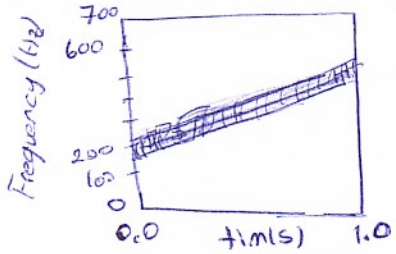
PI2 → is a constant that stores 2π

self.offset => is the phase when t=0

Spectrogram

To recover the relationship between frequency and time, we can break the chirp into segments and plot the spectrum of each segment.

The result is called a short-time Fourier transform (STFT)



Spectogram shows clearly that frequency increases linearly over time.

Noise

In English "noise" means an unwanted or unpleasand sound, if two signals interfere with each other, each signal would consider the other to be noise.

Uncorrelated Noise

The simple way to understand noise is to generate it, and the simlest kind to generate is uncorrelated uniform noise (UU noise)

Uniform means the signal contains random values from a uniform distribution.

Every value in the range is equally likely.

! quencies is called white noise by analogy with light, because an equal mixture of light at all visible frequencies is white.

Pink Noise

$P = K/f^{\beta}$

$\beta = 0$, power is constant at all frequencies, so the result is white noise.

when $\beta = 2$ the result is red noise.

when β is between 0 and 2 the result's between white and red noise, so it is called pink noise.

There are several ways to generate pink noise, The simplest is to generate white noise and then apply a low-pass filter with the desired exponent.

Like noise as "uncorrelated" which means that each value is independent of the others.

- Correlation as dot Product -

In signal processing, we are often working with unbiased signals, where the mean is 0, and normalized signals, where the standard deviation is 1,

ρ simplifies to;

$$\rho = \frac{1}{N} \sum_i x_i y_i$$

- Using Numpy -

```
corrs2 = np.correlate(segment.ys, segment.ys, mode='same')
```

- Discrete cosine Transform -

which is used in MP3 and related formats for compressing music.

DCT is similar in many ways to the Discrete Fourier Transform (DFT) which we have been using for spectral analysis.

$$M = \cos(2\pi t \otimes f)$$

```
def analyze(ys, fs, ts):
    args = np.outer(ts, fs)
    M = np.cos(PI2 * args)
    amps = np.linalg.solve(M, ys)
    return amps
```

- Orthogonal matrices -

One way to solve linear systems is by inverting metrices. The inverse of a matrix M is written M^{-1}, and it has the property that $M^{-1}M = I$ I is the identity matrix, which has the value 1 on all diagonal elements and 0 everywhere else.

$$M^{-1}y = \underbrace{M^{-1}M}_{I}a$$

$$M^{-1}y = Ia$$

If we multiply I by any vector a, the result is a, so

$$M^{-1}y = a \qquad M^T = M^{-1}$$

$$M^T M = I$$