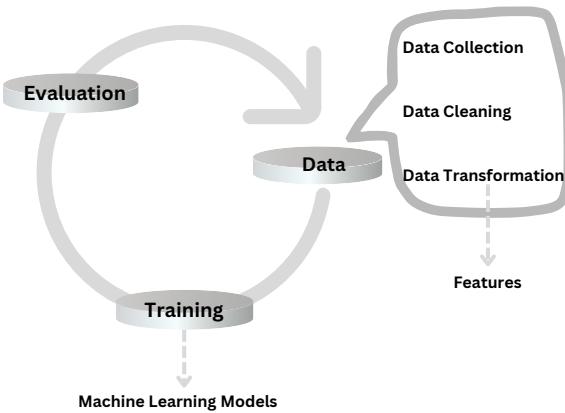


An Introduction to Audio Processing and Machine Learning Using Python

Yüksek seviye makine öğrenmesi 3 tip görevde ayrılabilir:

- **data tasks** : veri toplama, veri temizleme, özellik oluşturma
- **training** : veri özelliklerini kullanarak makine öğrenmesi modeli oluşturma
- **evaluation** : modeli değerlendirmeye



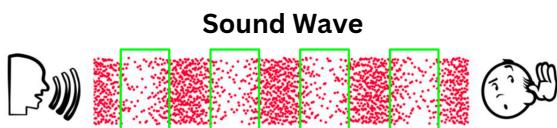
Bir makine öğrenmesi modelinde özellik çıkarmak modelin veriyi anlaması, kategorilere sınıflandırmamasına veya değer tahmin etmesine yardımcı olur.

Farklı veri tiplerinde farklı işleme teknikleri kullanılır. Bu nedenle sinyal verileriyle iyi çalışan sinyal verisi türüne özgü işleme teknikleri vardır.

Sound and audio waveforms?

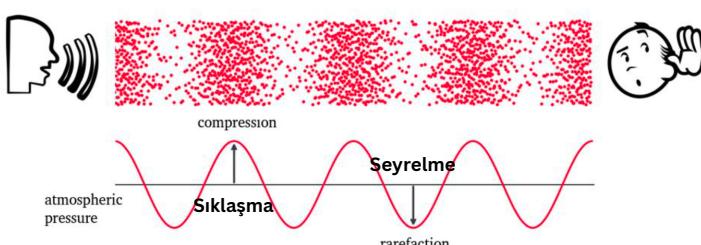
Ses : Bir cisim titreşimiyle üretilir. Titreşimler hava moleküllerinin de titreşmesine (salınımına) sebep olur. Hava basıncındaki değişiklik (sıklaşma-seyrelme) bir dalga oluşturur.

Mekanik Dalga : Uzayda dolaşan salınımdır. Enerji bir noktadan diğerine aktarılır. Ortam deform olmuştur.



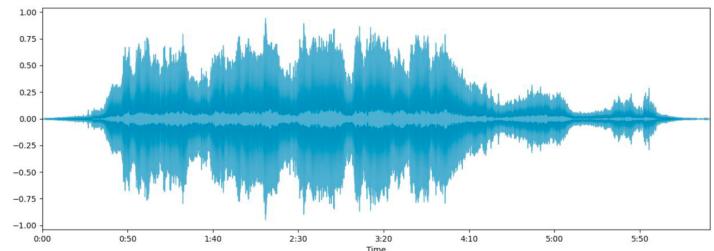
Ses Sinyali Nedir?

Ses sinyalleri, işitilebilir frekans aralığında titreten sinyallerdir. Birisi konuşduğunda hava basıncı sinyalleri üretir; kulak bu hava basıncı farklarını alır ve beyin ile iletişim kurar. Beyin bir kişinin sinyalin konuşma olduğunu anlamasına ve birinin ne söylediğini anlamasına bu şekilde yardımcı olur.



Waveform : Çok faktörlü bilgi taşırlar.

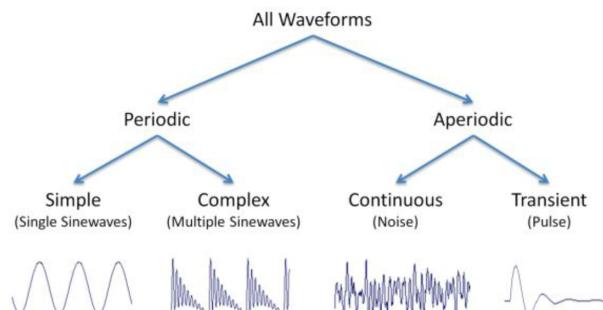
- Frekans (Frequency)
- Intensity (Yoğunluk)
- Timbre (Tını)



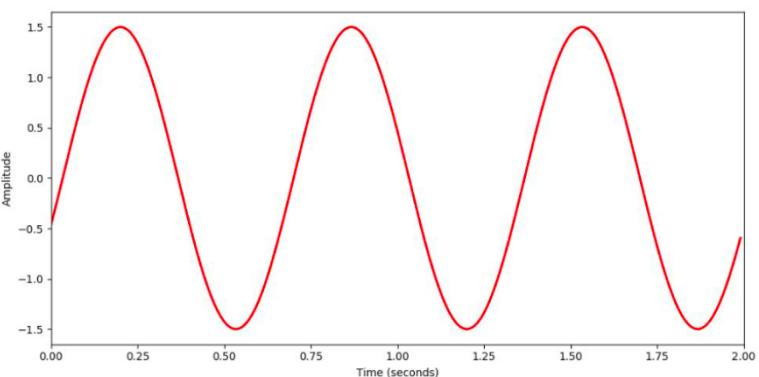
Waveform : Çok faktörlü bilgi taşırlar.

- Frekans (Frequency)
- Intensity (Yoğunluk)
- Timbre (Tını)

Periyodik ve Periyodik Olmayan Sesler

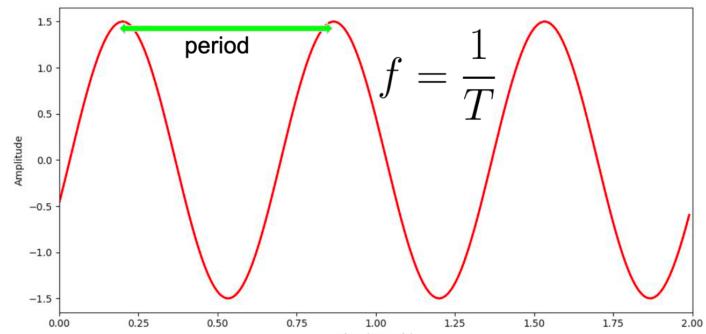


Waveform

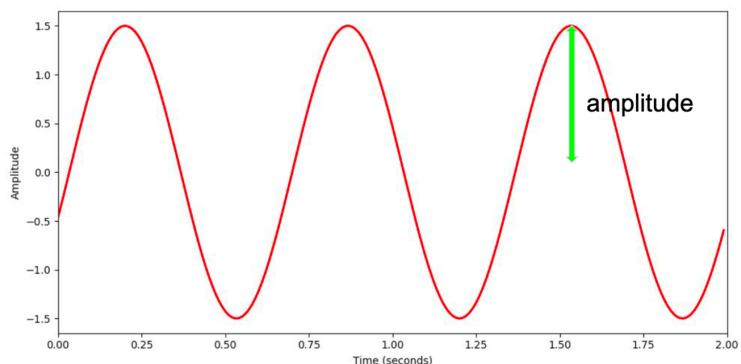


$$y(t) = A \sin(2\pi ft + \varphi)$$

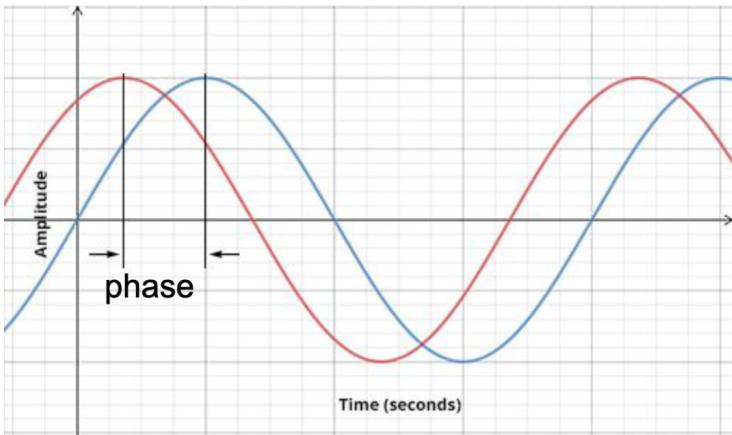
Frequency



Amplitude

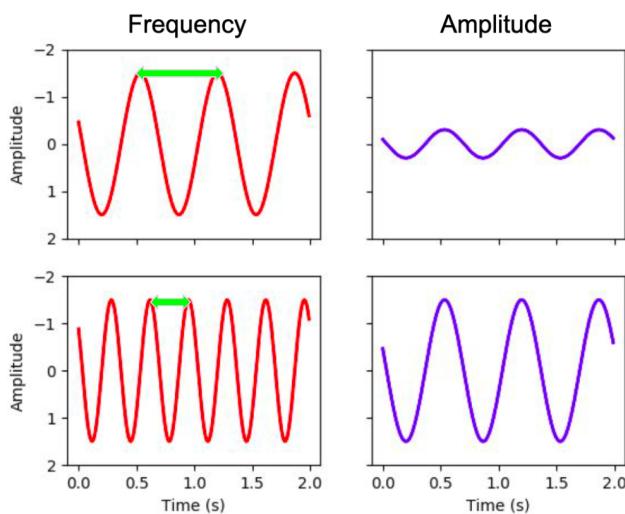


Phase

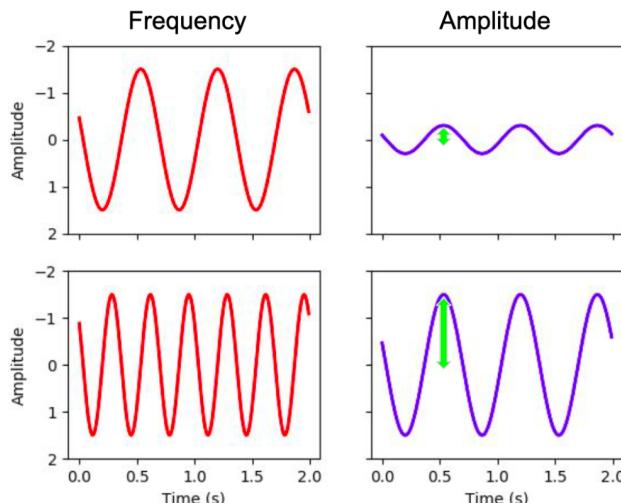


Frequency and Amplitude

- higher frequency -> higher sound



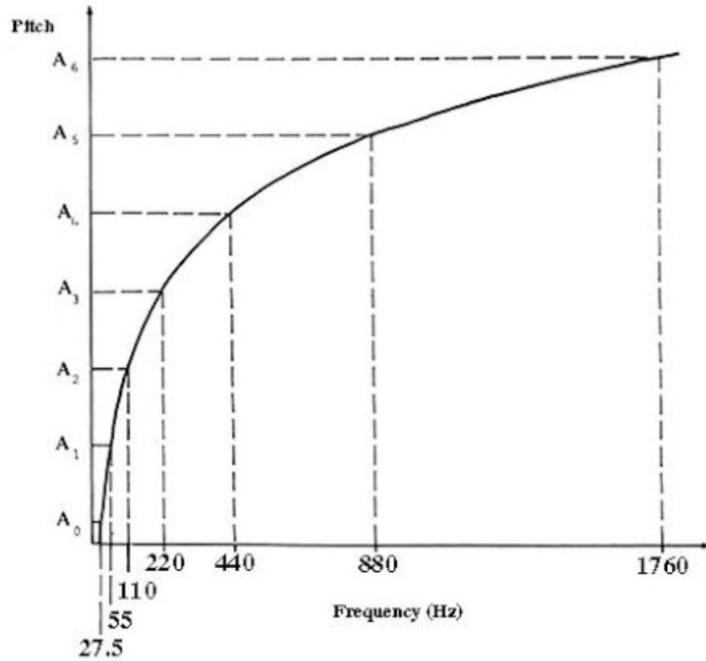
- larger amplitude -> louder



Pitch

- Logarithmic perception
- 2 frekans, 2'lük bir güçle farklılık gösteriyorsa benzer şekilde algılanır.

Pitch - Frequency Chart



Mapping pitch to frequency

$$F(p) = 2^{\frac{p-69}{12}} \cdot 440$$

$$F(60) = 2^{\frac{60-69}{12}} \cdot 440 = 261.6$$

$$F(p+1)/F(p) = 2^{1/12} = 1.059$$

Intensity (yoğunluk), loudness (ses yüksekliği) and timbre (tını)

Sound power

- Enerjinin aktarılma hızı
- Bir ses kaynağı tarafından tüm yönlerde yayılan birim zaman başına enerji
- Watt (W) cinsinden ölçülür

Sound intensity

- Birim alan başına ses gücü
- W/m² cinsinden ölçülür.

Threshold of hearing : İnsan çok küçük yoğunlıklardaki sesleri algılayabilir.

$$TOH = 10^{-12} W/m^2$$

Threshold of pain

$$TOP = 10 \cdot W/m^2$$

Intensity Level

- Logaritmik ölçek
- Desibel (dB) cinsinden ölçülür
- İki yoğunluk değeri arasındaki oran
- Bir referans yoğunluğu kullanılır. (TOH)
- Her ~3 dB'de bir, yoğunluk iki katına çıkar.

$$dB(I) = 10 \cdot \log_{10}\left(\frac{I}{I_{TOH}}\right)$$

$$dB(I_{TOH}) = 10 \cdot \log_{10}\left(\frac{I_{TOH}}{I_{TOH}}\right) = 0$$

$$dB(I_{TOH}) = 10 \cdot \log_{10}\left(\frac{I_{TOH}}{I_{TOH}}\right) = 0$$

$\log(1) = 0$

Source	Intensity	Intensity level	\times TOH
Threshold of hearing (TOH)	10^{-12}	0 dB	1
Whisper	10^{-10}	20 dB	10^2
Pianissimo	10^{-8}	40 dB	10^4
Normal conversation	10^{-6}	60 dB	10^6
Fortissimo	10^{-2}	100 dB	10^{10}
Threshold of pain	10	130 dB	10^{13}
Jet take-off	10^2	140 dB	10^{14}
Instant perforation of eardrum	10^4	160 dB	10^{16}

Loudness (Ses Yüksekliği)

- Ses yoğunluğunun öznel algısı
- Bir sesin süresine / frekansına bağlıdır
- Yaşa bağlıdır
- phons cinsinden ölçülür.

Timbre (Tını)

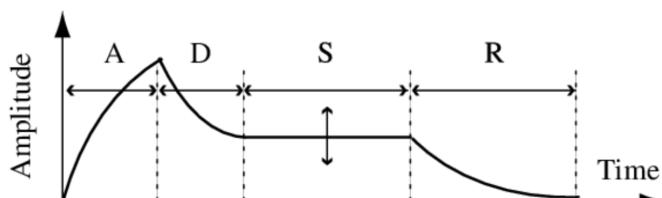
- Sesin rengi
- Aynı yoğunluk, frekans ve süreye sahip iki ses arasındaki fark
- Parlak (bright), karanlık (dark), donuk (dull), sert (harsh), sıcak (warm) gibi kelimelerle tanımlanır.

Features of timbre

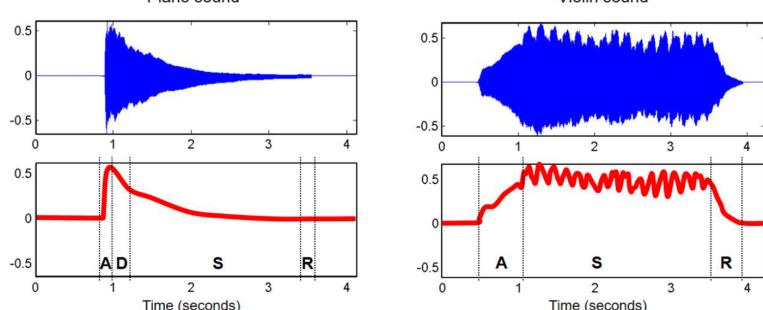
- Timbre is multidimensional
- Sound envelope
- Harmonic content
- Amplitude / frequency modulation

Sound envelope

- Attack-Decay-Sustain-Release Model



Piano sound



Complex Sound

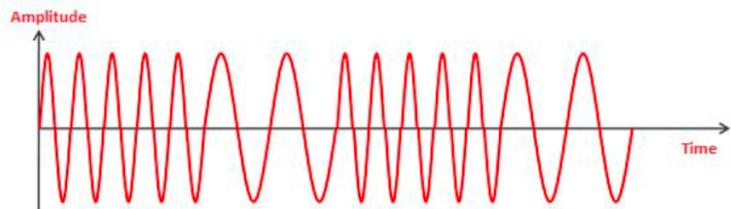
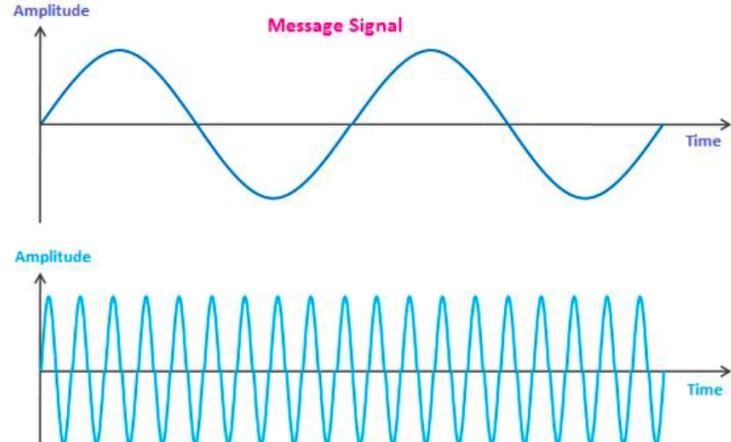
- Sinüzoidlerin süperpozisyonu
- Kısıtlı, bir sesi tanımlamak için kullanılan bir sinüzoiddir
- En düşük kısmı temel frekans olarak adlandırılır
- Harmonik kısmı, temel frekansın katları olan bir frekanştır

$$f_1 = 440, f_2 = 2 \cdot 440 = 880, f_3 = 3 \cdot 440 = 1320, \dots$$

- Uyumsuzluk, harmonik bir kısmı değerden sapmayı gösterir.

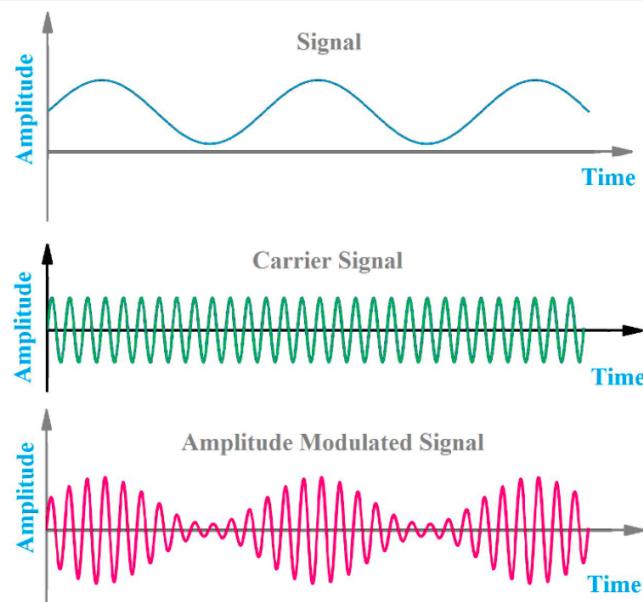
Frequency modulation

- AKA vibrato
- Frekansa periyodik değişim
- Müzikte, ifade amaçlı kullanılır



Amplitude modulation

- AKA tremolo
- Genlikte periyodik değişim
- Müzikte, ifade amaçlı kullanılır



Timbre recap

- Multifactorial sound dimension
- Amplitude envelope
- Distribution of energy across partials
- Signal modulation (frequency / amplitude)

Sound recap

- Sound is a wave
- Frequency, intensity, timbre
- Pitch, loudness, timbre

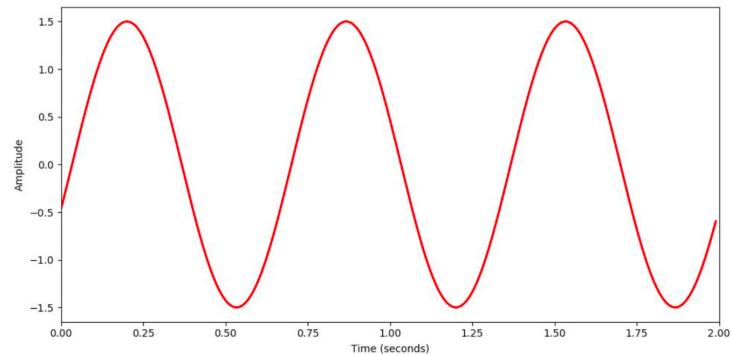
Understanding Audio Signals for ML

Audio Signal

- Sesin temsili
- Sesi yeniden üretmek için ihtiyaç duyduğumuz tüm bilgileri kodlar

Analog Signal

- Continuous values for time
- Continuous values for amplitude



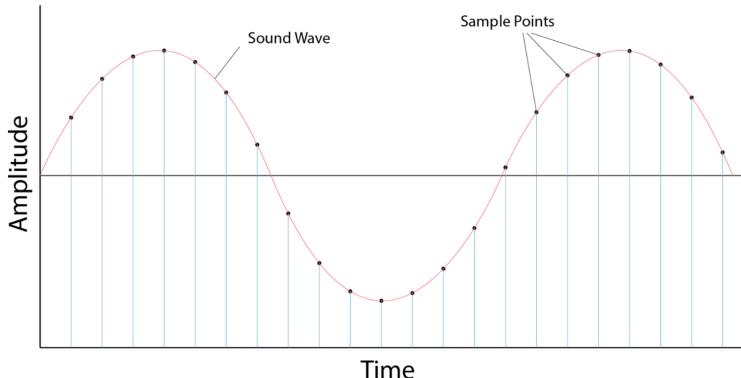
Digital Signal

- Ayrık değerler dizisi
- Veri noktaları yalnızca sonlu sayıda değer alabilir

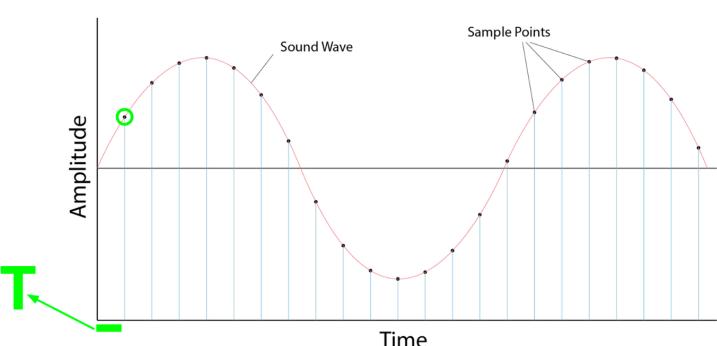
Analog to digital conversion

- Sampling
- Quantization

Sampling



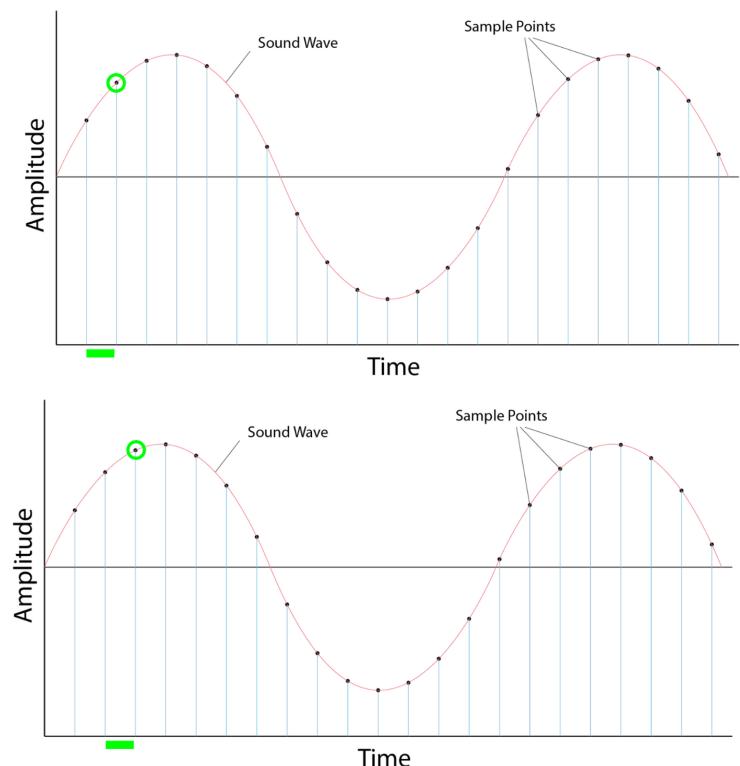
Sampling Period



Pulse - code modulation: Pulse-code modulation (PCM), örneklenmiş analog sinyalleri dijital olarak temsil etmek için kullanılan bir yöntemdir. Bilgisayarlarda, kompakt disklerde, dijital telefonlarda ve diğer dijital ses uygulamalarında standart dijital ses biçimidir. Bir PCM akışında, analog sinyalin genliği düzgün aralıklarla örneklenir ve her örnek bir dijital adım aralığı içinde en yakın değere nicerlenir.

Linear pulse-code modulation (LPCM), niceleme seviyelerinin doğrusal olarak tekdeğer olduğu özel bir PCM türüdür. Bu, niceleme seviyelerinin genliğin bir fonksiyonu olarak değiştiği PCM kodlamalarının aksine (A-law algoritması veya μ -law algoritmasında olduğu gibi). PCM daha genel bir terim olmasına rağmen, genellikle LPCM olarak kodlanmış verileri tanımlamak için kullanılır.

Bir PCM akışı, akışın orijinal analog sinyale olan sadakatini belirleyen iki temel özelliğe sahiptir: örneklerin saniyede kaç kez alındığını gösteren örneklemme oranı ve her bir örneği temsil etmek için kullanılan olası dijital değerlerin sayısını belirleyen bit derinliği.

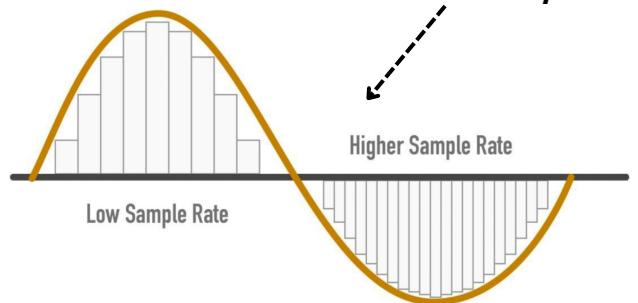


Locating Samples

$$t_n = n \cdot T$$

Sampling Rate

$$s_r = \frac{1}{T}$$



İnsanların işitme aralığı 20-20K Hz arasındadır ve Ses Sinyali İşleme kullanan çoğu uygulama örneklemme hızını bu aralıktan tutmalıdır. Analog sinyalleri Dijital sinyallere dönüştürme yöntemi, ses dalgalarının sürekli doğası nedeniyle gerekli olan Sampling olarak bilinir. Bilgisayarlar ve diğer dijital makinelerin çoğu sürekli sinyalleri işleyemez, yalnızca ayrık dijital sinyalleri işleyebilir. Ancak çoğu ses sinyali, birçok bireysel ses dalgasının birleşimidir. Bu sinyalleri analiz etmek için, her bir sinyalin boyutlarının daha iyi anlaşılabilmesi amacıyla ayırtılması gereklidir. Bunu başarmak için Hızlı Fourier Dönüşümü (FFT) kullanılır. Yukarıdaki şekil iki farklı örneklemme hızının nasıl görüneceğini göstermektedir. Dalganın sol tarafı düşük örneklemme hızına, sağ tarafı ise daha yüksek örneklemme hızına sahiptir.

Nyquist Frequency Formula and Implementing for CD

Nyquist frekansı dijital sinyal işlemede temel bir kavramdır. Ayrık bir sinyalin örnekleme hızının yarısı veya dijital bir sistemde doğru bir şekilde temsil edilebilen maksimum frekans olarak tanımlanır.

Harry Nyquist, Nyquist frekansı kavramını ilk kez ortaya attığında, dijital sinyal işleme alanında devrim yaratmıştır. Onun çalışmalarından önce mühendisler analog sinyalleri dijital formatta doğru bir şekilde temsil etmek için mücadele ediyorlardı. Nyquist'in atılımı Nyquist-Shannon örnekleme teoremi şeklinde geldi.

Nyquist-Shannon örnekleme teoremine göre, bir sinyalin doğru bir şekilde yeniden yapılandırılabilmesi için en yüksek frekans bileşeninin iki katı hızda örneklemesi gereklidir. Bu ilke modern dijital ses ve video teknolojilerinin temelini oluşturmuştur.

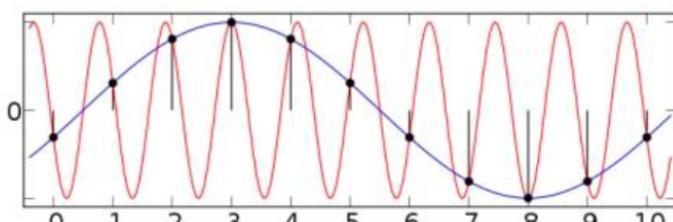
Nyquist-Shannon örnekleme teoreminin çeşitli endüstriler üzerinde derin bir etkisi olmuştur. Yüksek kaliteli ses ve video codec'lerinin geliştirilmesini sağlayarak multimedya içeriğinin minimum doğruluk kaybıyla iletilmesine ve depolanmasına olanak tanımlıdır. Teorem ayrıca telekomünikasyon, radar sistemleri ve tıbbi görüntüleme gibi alanlarda da uygulama alanı bulmuştur.

Sonuç olarak, adını Harry Nyquist'ten alan Nyquist frekansı, dijital sinyal işlemede çok önemli bir kavramdır. Dijital bir sistemde doğru bir şekilde temsil edilebilen maksimum frekansı temsil eder ve sinyalleri yakalama, iletme ve işleme şeklimizde devrim yaratmıştır. Harry Nyquist'in telekomünikasyon alanına yaptığı katkılar modern teknolojileri şekillendirmeye devam etmekte ve dijital çağda sayısız ilerlemenin önünü açmaktadır.

$$f_N = \frac{s_r}{2}$$

$$f_N = \frac{44100}{2} = 22050$$

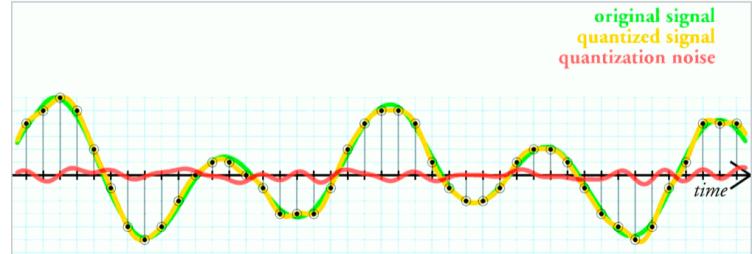
Aliasing : Örtüşme, bir sinyali Nyquist hızından daha düşük bir hızda örneklediğinizde ortaya çıkan bozulmadır. Bu durumda, sinyaldeki Nyquist frekansından daha yüksek frekanslar daha düşük olanlardan ayırt edilemez ve örnekleşen sinyalde yanlış veya örtüşen frekanslar olarak görünürler. Bu durum istenmeyen gürültülere, yapaylıklara ve sinyal işlemenizde hatalara yol açabilir. Örtüşmeyi önlemek için, sinyali örneklemeden önce Nyquist frekansının üzerindeki frekansları filtrelemeniz gereklidir. Buna örtüşme önleyici filtreleme denir ve genellikle yüksek frekansları zayıflatın ve düşük frekansları geçiren bir alçak geçiren filtre kullanılarak yapılır.



Quantization : Matematikte ve dijital sinyal işlemede niceleme (quantization), büyük bir kümedeki (genellikle sürekli bir küme) giriş değerlerini, genellikle sonlu sayıda elemanı olan (sayılabilir) daha küçük bir kümedeki çıkış değerlerine eşleme işlemidir. Yuvarlama ve kesme (rounding and truncation), niceleme işlemlerinin tipik örnekleridir. Bir sinyali dijital formda temsil etme süreci normalde yuvarlamayı içerdiğinden, kuantizasyon neredeyse tüm dijital sinyal işlemlerinde bir dereceye kadar yer alır. Niceleme ayrıca esasen tüm kayıplı sıkıştırma algoritmalarının çekirdeğini oluşturur. (Quantization also forms the core of essentially all lossy compression algorithms.)

Bir giriş değeri ile niceleme arasındaki fark (yuvarlama hatası / round-off error gibi) niceleme hatası olarak adlandırılır. Niceleme yapan bir cihaza veya algoritmik işleve niceleyici denir. Analogdan dijitalde dönüştürücü bir nice(m)leyici örneğidir.

Bir sinyali kuantize etmenin en basit yolu, orijinal analog genlige en yakın dijital genlik değerini seçmektir. Aşağıdaki örnekte orijinal analog sinyal (yeşil), kuantize edilmiş sinyal (siyah noktalar), kuantize edilmiş sinyalden yeniden yapılandırılmış sinyal (sarı) ve orijinal sinyal ile yeniden yapılandırılmış sinyal arasındaki fark (kırmızı) gösterilmektedir. Orijinal sinyal ile yeniden yapılandırılmış sinyal arasındaki fark kuantalama hatasıdır (**quantization error**) ve bu basit kuantalama şemasında giriş sinyalinin deterministik bir fonksiyonudur.



Dynamic Range

Dinamik aralık, belirli bir niceliğin alabileceği en büyük ve en küçük değerler arasındaki orandır. Genellikle ses ve ışık gibi sinyaller bağlamında kullanılır. Ya bir oran olarak ya da en küçük ve en büyük sinyal değerleri arasındaki farkın baz-10 (desibel) veya baz-2 (ikiye katlama, bit veya durak) logaritmik değeri olarak ölçülür.

Elektronik olarak yeniden üretilen ses ve video genellikle geniş bir dinamik aralığı sahip orijinal materyali daha kolay depolanabilen ve yeniden üretilebilen daha dar bir kaydedilmiş dinamik aralığı sıkıtmak için işlenir; bu işleme dinamik aralık sıkıştırma denir.

resolution ↑ dynamic range ↑

Signal-to-quantization-noise ratio (SQNR veya SNqR), pulse-code modulation (PCM) gibi saysısallaştırma şemalarının analizinde yaygın olarak kullanılan bir kalite ölçütüdür. SQNR, maksimum nominal sinyal gücü ile analogdan dijital dönüşümde ortaya çıkan niceleme hatası (niceleme gürültüsü olarak da bilinir) arasındaki ilişkiye yansır.

- Maksimum sinyal gücü (signal strength) ve niceleme hatası (quantization error) arasındaki ilişkidir.
- Dinamik aralık (Dynamic range) ile ilişkilidir.

$$SQNR \approx 6.02 \cdot Q$$

$$SQNR(16) \approx 96dB$$

How do we record sound?



How do we reproduce sound?

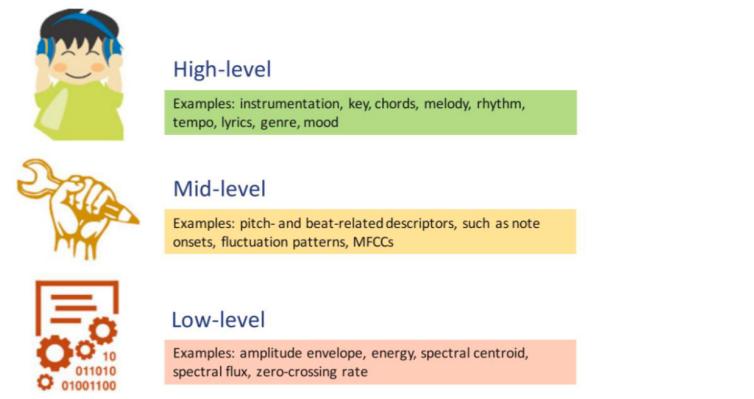


Why audio features?

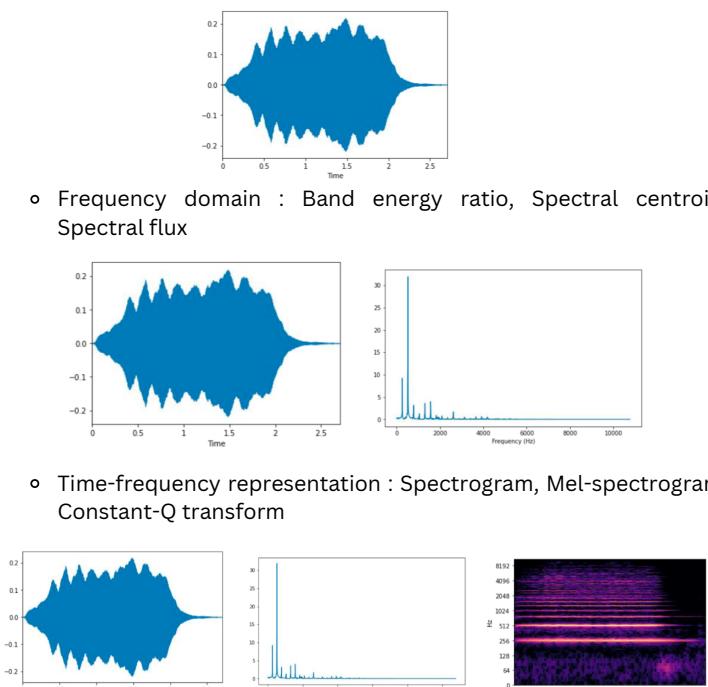
- Sesin tanımı
- Farklı özellikler sesin farklı yönlerini yakalar
- Akıllı ses sistemleri oluşturma

Ses özelliği kategorizasyonu (Audio feature categorisation)

- Soyutlama düzeyi (Level of abstraction)



- Zamansal kapsam (Temporal scope)
 - Instantaneous (~50 ms)
 - Segment-level (seconds)
 - Global
- Müzik yönü (Music aspect)
 - Beat
 - Timbre
 - Pitch
 - Harmony
- Sinyal alanı (Signal domain)
 - Time domain : Amplitude envelope, Root-mean square energy, zero crossing rate
 - Frequency domain : Band energy ratio, Spectral centroid, Spectral flux
 - Time-frequency representation : Spectrogram, Mel-spectrogram, Constant-Q transform



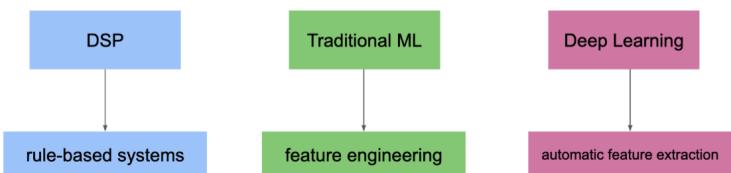
- Makine öğrenimi yaklaşımı (ML approach)
 - Traditional Machine Learning



- Deep Learning

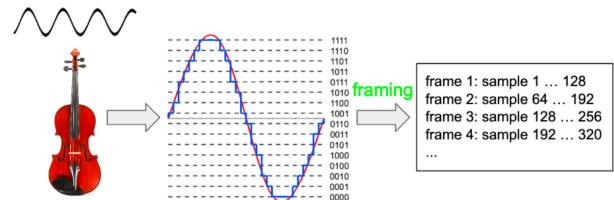


Types of intelligent audio systems



Time-domain feature pipeline

Quantization işlemi ile seslerin dijitalleştirilmiş versiyonu alındıktan sonra Framing işlemine geçilir. Başka bir deyişle bir grup örnek bir araya getirilir. (Bundle together a bunch of samples)



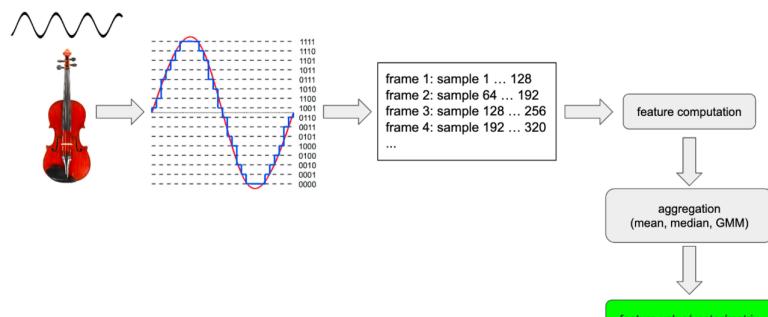
Özellikleri çıkarmadan önce neden framing kullanılır?

Konuşma durağan olmayan bir sinyaldir, dolayısıyla istatistiksel özellikler zaman içinde sabit değildir. Bu nedenle, spektral özellikler ve diğer karakteristik özellikleri (örneğin: short-time energy, MFCC vb.) sinyalin küçük bloklarından çıkarılmalıdır. Bu, sinyalin bu küçük çerçevede (frame) durağan olduğu (yani istatistiksel özelliklerinin bu bölge içinde sabit olduğu) varsayıma dayanmaktadır. Tüm bunların ötesinde, çerçeve bloklama (framing) genellikle gerçek zamanlı sistemlerde kullanılır çünkü sabit işlem yükünü birçok örnegéye dağıtarak sistemin verimliliğini en üst düzeye çıkarır.

Frames:

- Frameler insanların algılayabildiği ses parçaları olarak düşünülebilir. (Perceivable audio chunk)
 - 1 sample @44.1 KHz = 0.0227 ms (Bu değer örnekleme oranının tersi gibidir.)
 - Duration 1 sample << Ear's time resolution (10 ms)
- 2' nin katları olan örnekler sahiptirler. (Power of 2 num. samples) Bu sayede Fourier dönüşümü yapılurken süreç hızlandırılmış olunur.
- Typical values : 256 - 8192
 - K : Frame Size / Number of Samples

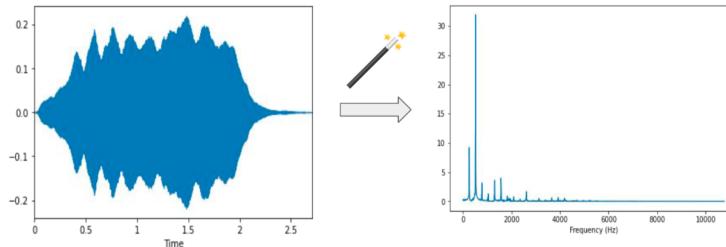
$$d_f = \frac{1}{s_r} \cdot K \quad d_f = \frac{1}{\boxed{s_r}} \cdot \boxed{K} \quad d_f = \frac{1}{\boxed{s_r}} \cdot \boxed{\frac{512}{44100}} = 11.6ms$$



Frequency-domain feature pipeline

Bu kısımda yapmak istenilen zaman domaininden frekans domainine geçmektedir. Bunu yapmanın yolu da **Fourier Transform**'dur.

From time to frequency domain (Fourier Transform)



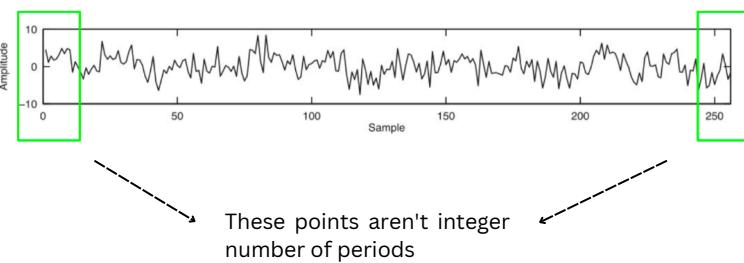
Zaman alanında temel olarak genliğin bir fonksiyonu olarak görselleştirilen sese sahibiz. Dolayısıyla ses için zaman içindeki tüm olaylar görebiliyoruz.

Burada bir sesin tüm frekans bileşenlerine bakarız ve genel sese ne kadar katkıda bulunduklarını görürüz ve aslında x ekseninde frekans veya Hertz olarak frekans alanına sahibiz, y ekseninde ise bize farklı frekans bantlarının her birinin genel sese ne kadar katkıda bulunduğuunu söyleyen büyülüğe sahibiz.

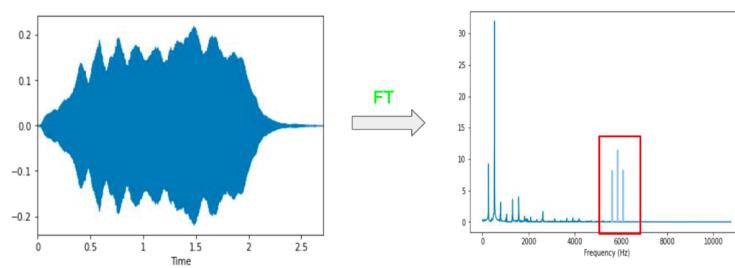
Şimdi, çıkışma işlemindeki bir sonraki adımın Fourier Dönüşümünü uygulamak olacağını ve bunun Metro K gibi olacağını, böylece zaman alanında frekans alanına geçebileceğimizi bekledik, ancak ne yazık ki **spektral sızıntı (spectral leakage)** adı verilen önemli bir sorun ortaya çıkar.

Spectral Leakage Happens...

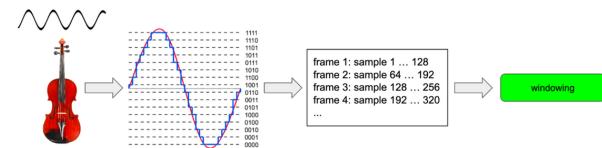
- Bir sinyali işlerken
- Tam sayı (periyotta) olmayan bir sinyalin Fourier dönüşümünü aldığımızda
- Sinyalin uç noktalarının sürekli olmadığı durumlarda



- Bu süreksizliklere sahipsek, bu süreksizlikler spektruma veya frekans alanına yüksek frekans bileşenleri olarak çevrilir, ancak bu yüksek frekans bileşenleri orijinal sinyalde gerçekten mevcut değildir, sadece Fourier dönüşümü ile işaretlediğimiz sinyalin üç noktalarındaki süreksizlikler nedeniyle ortaya çıkan bir tür yapaylık vardır, başka bir deyişle olan şey, süreksizliklerdeki bu süreksiz frekansların bazılarının diğer yüksek frekanslara sızmasıdır, dolayısıyla spektral sızıntı olarak adlandırılır.



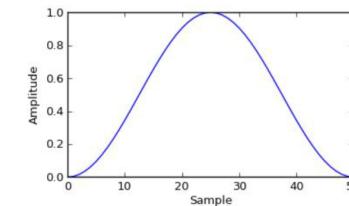
Spectral leakage'ın çözümü windowing' tir.



Windowing fikrinin temel amacı, her bir frame'i Fourier Dönüşümü'ne beslemeden önce her frame'e bir framing işlemi uygulamaktır ve bu sayede çerçeveyenin üç noktalarındaki (endpoints) örneklemeleri tamamen kaldırırız, diğer bir deyişle üç noktalarındaki bilgiyi tamamen ortadan kaldırırız ve bu, spectral leakage en azı indiren periyodik bir sinyal oluşturur. Bu işlem için yaygın olarak kullanılan bir fonksiyon vardır:

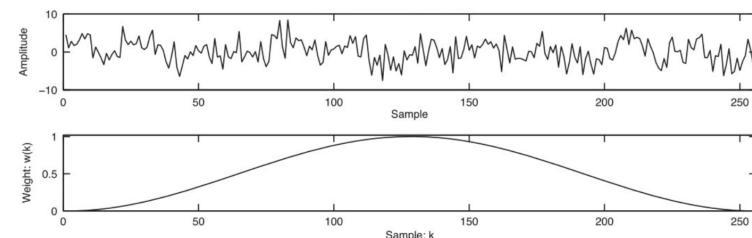
Hann Window : $k \rightarrow$ sample, başlangıç ve bitiş noktalarının sıfır gitme eğiliminde olduğu çan şeklindeki bir eğridir.

$$w(k) = 0.5 \cdot (1 - \cos(\frac{2\pi k}{K-1})), k = 1 \dots K$$



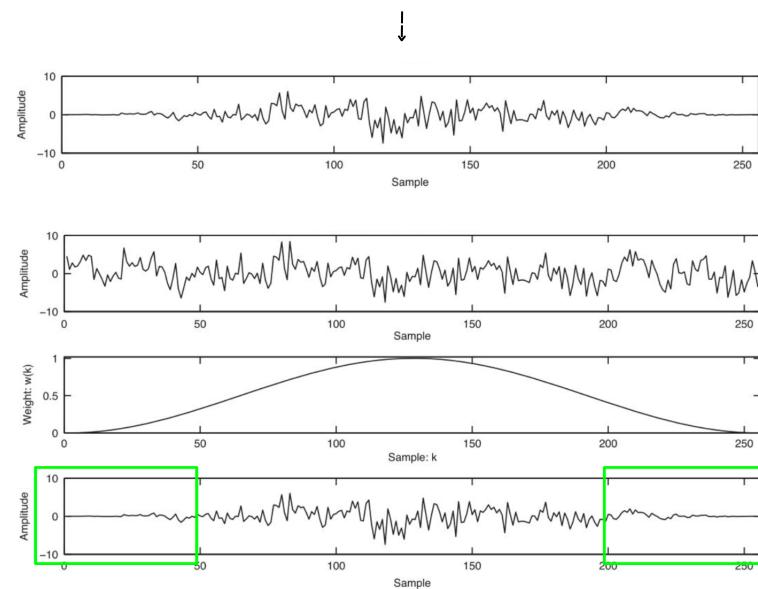
Applying Hann Window

Aşağıda 256 örnek içeren bir frame vardır.



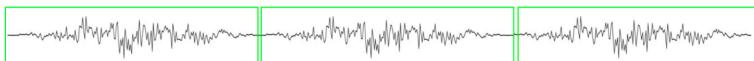
$$s_w(k) = s(k) \cdot w(k), k = 1 \dots K$$

Yukarıdaki formülü orijinal sinyali Hann window'a sokabilmek / orijinal sinyala Hann window uygulayabilmek için kullanıyoruz. Başka bir deyişle, yaptığımız şey orijinal sinyali her bir karşılık gelen örnekte hann penceresi ile çarpmaktır ve elde ettiğimiz şey aşağıdakine benzer.



Burada görebileceğiniz gibi hann penceresini uyguladığımız için şimdi üç noktaları tamamen yumuşattık ve böylece artık bu süreksizliklere sahip değiliz çünkü sinyal, kullandığımız pencereleme sayesinde doğal olarak sıfır gidiyor.

Şimdi ise başka bir problemimiz var!

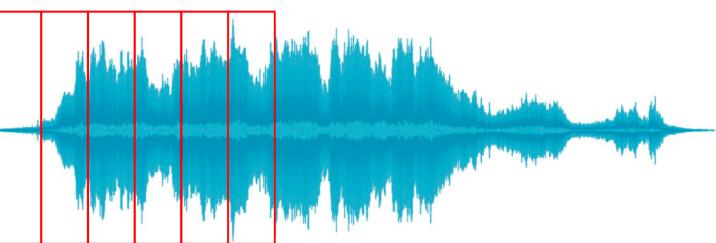
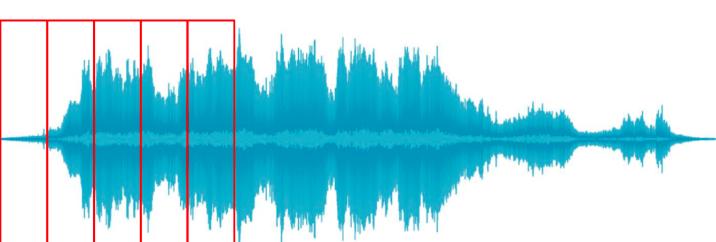
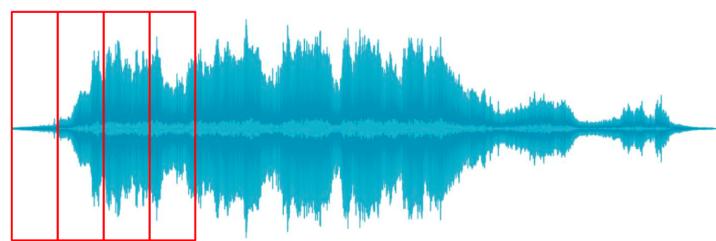
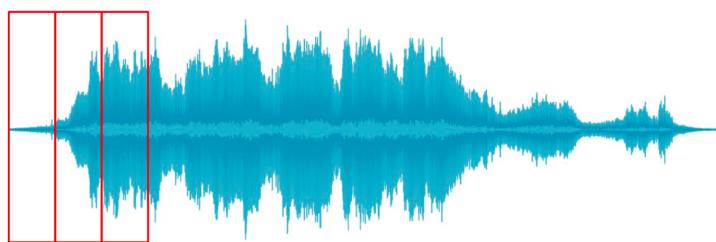
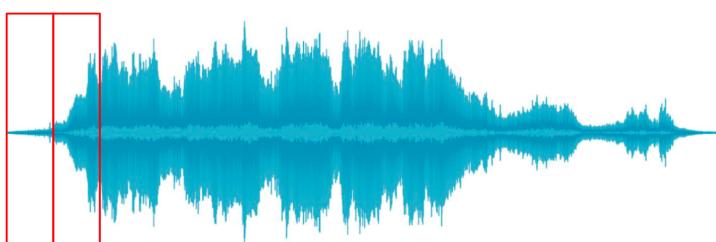
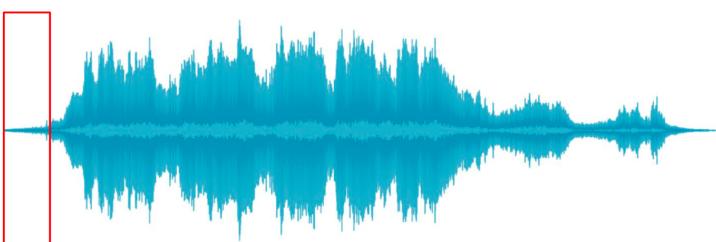


Bilgi kaybı yaşadığımız kısımlar aşağıdaki gibidir. Bunu Overlapping Frames ile çözeriz.

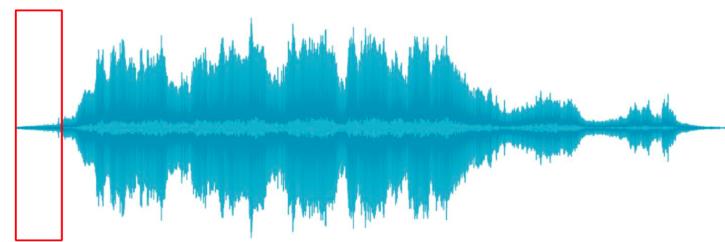
Frameleri bir araya getirdiğimizde windowing fonksiyonundan dolayı bazı kısımlarda bilgi kaybı yaşayacağız. Bilgi kaybı yaşamak istemediğimiz için bunu çözmemiz gerekiyor.



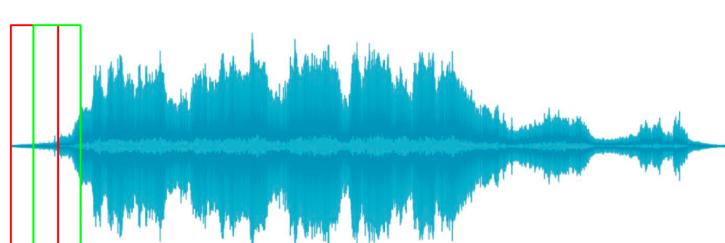
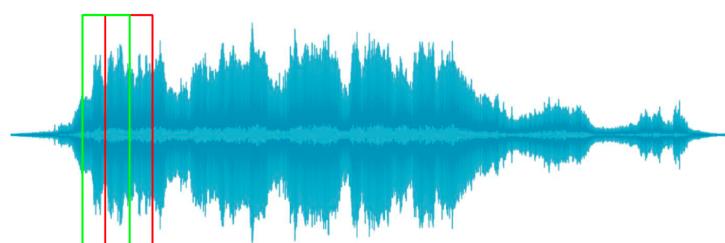
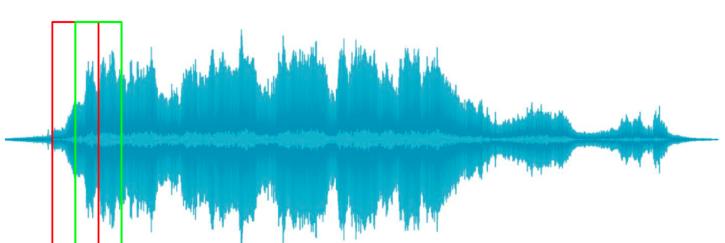
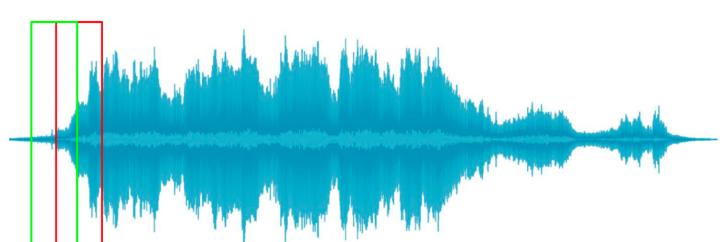
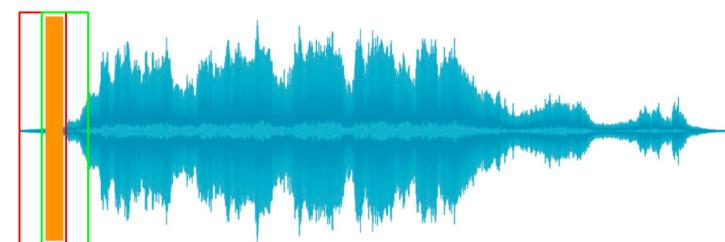
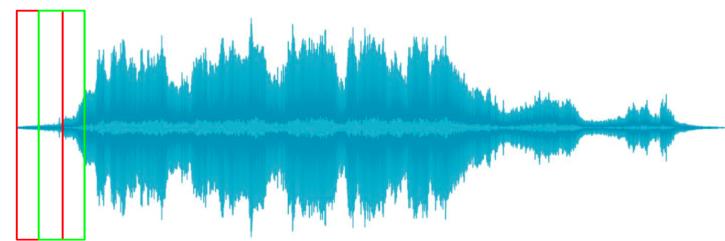
Öncelikle non-overlapping frames görelim.



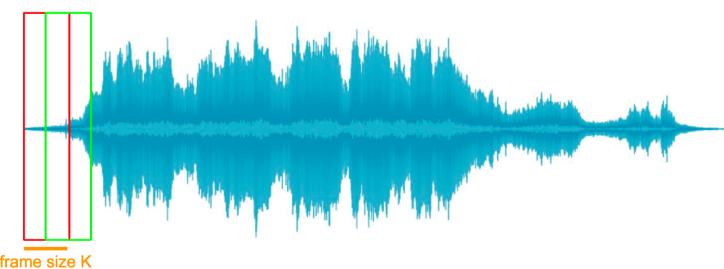
Overlapping frames



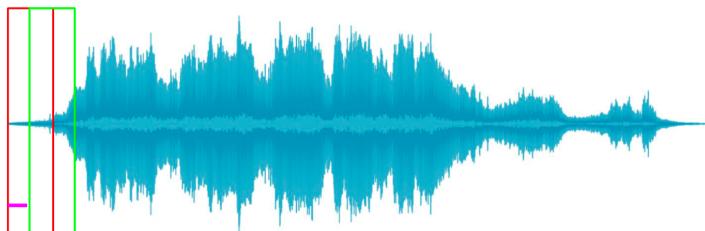
Üst üste binen kısımları nasıl bulabiliriz bunun üzerine tartışacağız.



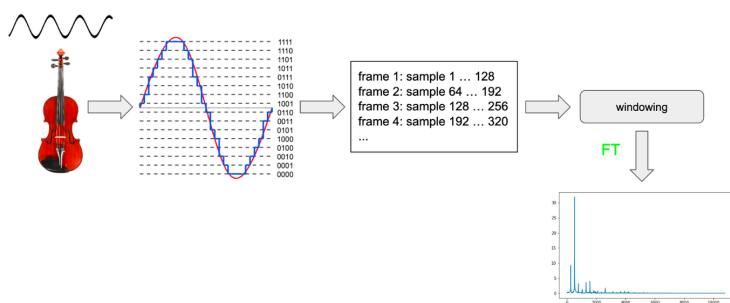
frame size (K) sample sayısını ifade eder.



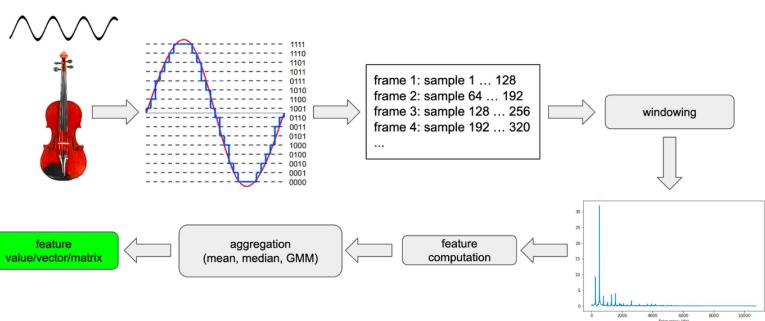
hop length atlama uzunluğunu ifade eder. hop size olarak da geçmektedir. Uzunluk temel olarak her yeni örnek aldığımda sağa kaydırduğumuz frame miktarıdır.



Artık Fourier Transform uygulayabileceğimiz noktadayız. Fourier Transform ile spectral leakage'in en aza indirildiği bir spektrum elde etmiş oluruz.



Daha sonra noktaların zaman alanı özellik çıkarma işlemi için kullandığımız aynı kümelere geri döneriz, böylece her karede frekans alanı özelliklerini hesaplarız, daha sonra bu sonuçları bazı istatistiksel araçlar kullanarak ses sinyalinin tamamı için toplarız ve son olarak bir frekans özelliği/frekans alanı özellik hacmi vektörü veya matrisi elde ederiz.



Time-domain features

- Amplitude envelope (AE)
- Root-mean-square energy (RMS)
- Zero-crossing-rate (ZCR)
-

Amplitude envelope

- Bir framedeki tüm örneklerin maksimum genlik değeri

$$AE_t = \max_{k=t \cdot K}^{(t+1) \cdot K - 1} s(k)$$

Amplitude envelope at frame t

Frame size

$$AE_t = \max_{k=t \cdot K}^{(t+1) \cdot K - 1} s(k)$$

Amplitude envelope at frame t

First sample of frame t

Amplitude of kth sample

$$AE_t = \max_{k=t \cdot K}^{(t+1) \cdot K - 1} s(k)$$

Amplitude envelope at frame t

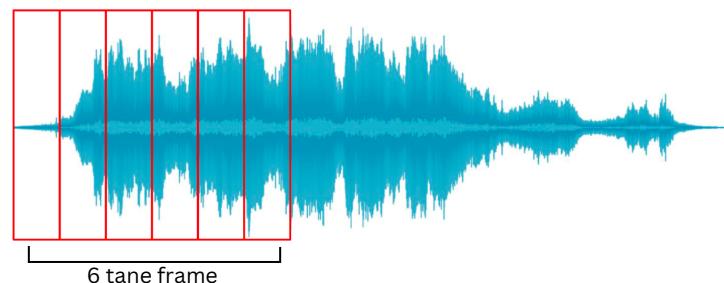
First sample of frame t

Last sample of frame t

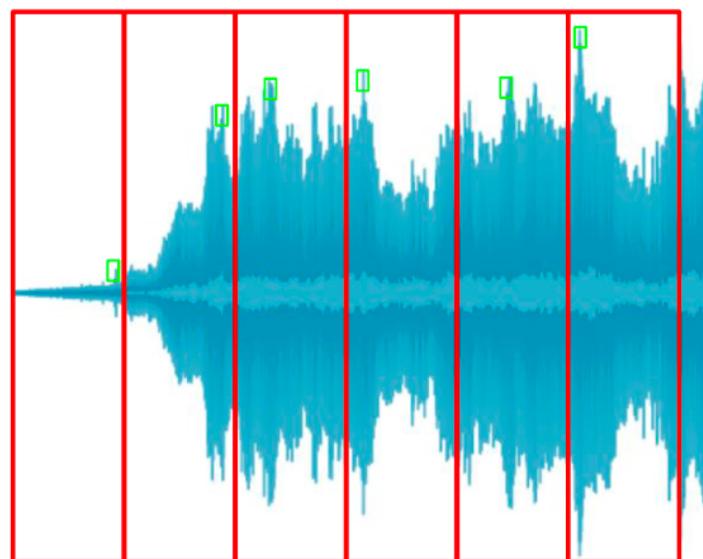
Amplitude of kth sample

$$AE_t = \max_{k=t \cdot K}^{(t+1) \cdot K - 1} s(k)$$

Calculate AE for all the frames



Her bir frame için amplitude envelope alalım.



- Ses yüksekliği hakkında kabaca fikir verir.
- Aykırı değerlere duyarlıdır.
- Onset detection (Başlangıç tespiti / algılama), music genre classification (müzik türü sınıflandırma) için kullanılabilir.

RMS

- RMS of all samples in a frame

Sum of energy for all samples in frame t

$$RMS_t = \sqrt{\frac{1}{K} \cdot \sum_{k=t \cdot K}^{(t+1) \cdot K - 1} s(k)^2}$$

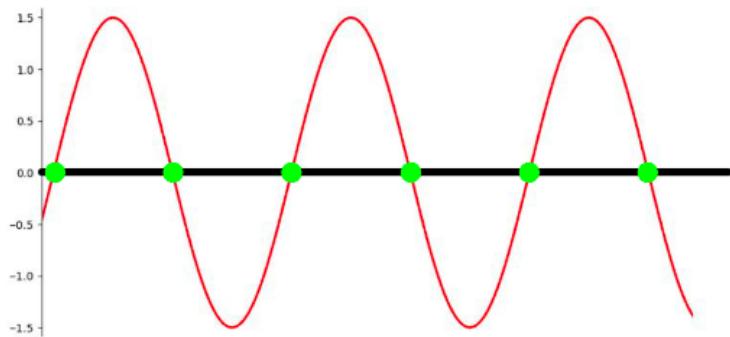
Mean of sum of energy

Energy of kth sample

- Ses yüksekliği göstergesi
- Aykırı değerlere AE'den daha az duyarlı
- Audio segmentation, music genre classification

ZCR

- Bir sinyalin yatay ekseni kaç kez kestiği



$$ZCR_t = \frac{1}{2} \cdot \sum_{k=t \cdot K}^{(t+1) \cdot K - 1} |sgn(s(k)) - sgn(s(k + 1))|$$

Sign function:
 • $s(k) > 0 \rightarrow +1$
 • $s(k) < 0 \rightarrow -1$
 • $s(k) = 0 \rightarrow 0$

$$ZCR_t = \frac{1}{2} \cdot \sum_{k=t \cdot K}^{(t+1) \cdot K - 1} |sgn(s(k)) - sgn(s(k + 1))|$$

+ +
- +
0

$$ZCR_t = \frac{1}{2} \cdot \sum_{k=t \cdot K}^{(t+1) \cdot K - 1} |sgn(s(k)) - sgn(s(k + 1))|$$

2

ZCR Uygulamaları

- Tiz ve vurgulu seslerin tanınması
- Monofonik (tek sesli) perde tahmini
- Konuşma sinyalleri için sesli / sessiz kararı

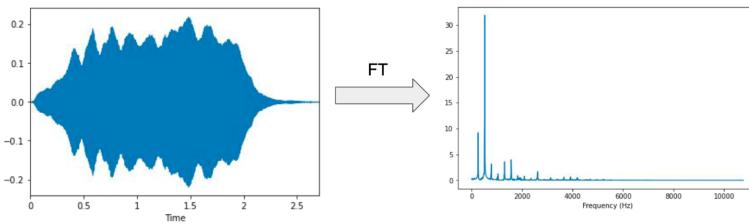
Fourier Transform

Bir ışığı prizmaya sokarsak ışığı oluşturan tüm farklı bileşenleri elde etmiş oluruz. Fourier Transform da buna benzer şekilde çalışır.



- Karmaşık bir sesi frekans bileşenlerine ayırma
- Fourier Transform kullandığımızda aslında time domainden frequency domaine geçiş yapmış oluruz.

From time to frequency domain



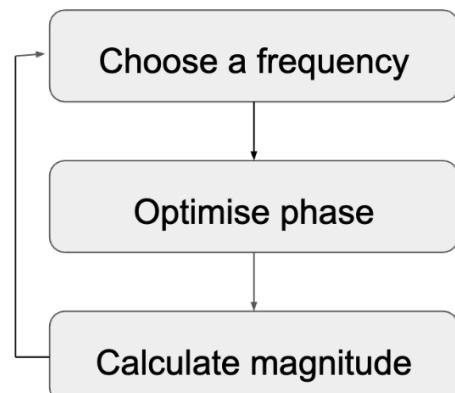
- Orijinal sinyali çeşitli frekanslardaki sinüzoidlerle karşılaştırma
- Her frekans için bir büyüklük (magnitude) ve bir faz (phase) elde ederiz
- Yüksek büyüklük (magnitude), sinyal ile sinüzoid arasında yüksek benzerlik olduğunu gösterir

Sine Wave (Sinüs Dalgası)

- $f \rightarrow$ frequency
- $t \rightarrow$ time
- $\varphi \rightarrow$ phase

$$\sin(2\pi \cdot (ft - \varphi))$$

Fourier Transform : step by step



Fourier Transform

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

Multiply signal and sinusoid

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

Calculate area

$$\varphi_f = \boxed{\operatorname{argmax}_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)}$$

Select phase in $[0, 1]$ that maximises the area

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

$$d_f = \max_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

$$d_f = \boxed{\max_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)}$$

Select max area

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\boxed{\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt} \right)$$

$t \in \mathbb{R}$

$$d_f = \max_{\varphi \in [0,1]} \left(\boxed{\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt} \right)$$

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (\boxed{ft} - \varphi)) \cdot dt \right)$$

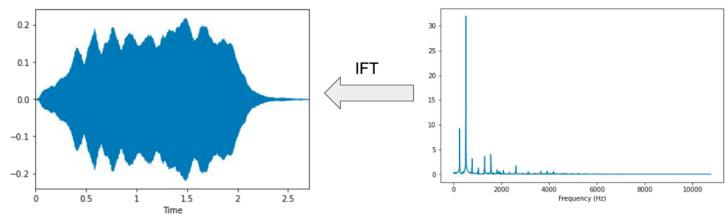
$f \in \mathbb{R}$

$$d_f = \max_{\varphi \in [0,1]} \left(\int s(t) \cdot \sin(2\pi \cdot (\boxed{ft} - \varphi)) \cdot dt \right)$$

Reconstructing a signal (Bir sinyalin yeniden yapılandırılması)

- Sinüzoidleri üst üste bindirme (Superimpose sinusoids)
- Bunları göreli büyüklüklerine göre ağırlılandırın (Weight them by the relative magnitude)
- Bağlı faz kullanın (Use relative phase)
- Orijinal sinyal ve FT'de bazı bilgiler var (Original signal and FT have some information)

Inverse Fourier Transform



Complex Numbers for Audio Signal Processing

Why bother with complex numbers?

- Fourier dönüşümüyle uğraşırken genellikle karmaşık bir sesi ayırtındığımızda saf terim ve frekansların her biri için birkaç parametre elde ederiz. Bu parametreler büyüklik ve fazdır. (Fourier transform -> magnitude and phase)
- Magnitude gerçek bir sayıdır.
- Magnitude ve phase kavramlarını bir arada kullanabileceğimiz bir şey olsaydı sorusuna cevap olarak complex numbers verilebilir.

COMPLICATED NUMBERS?

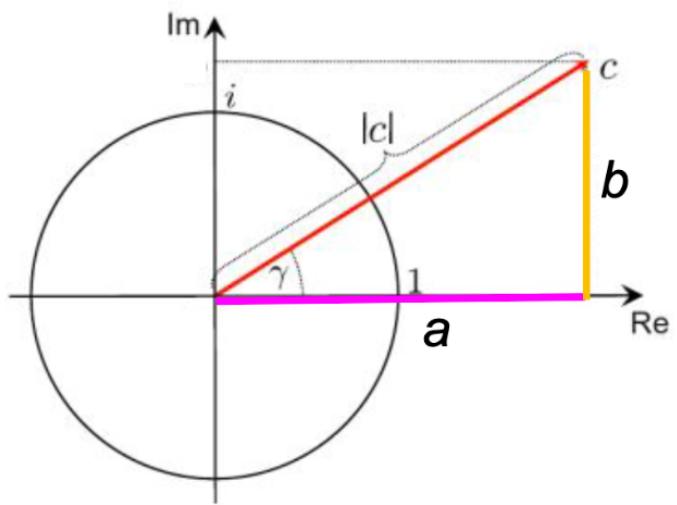


NO SIRaj! IT'S COMPLEX NUMBERS





$\sqrt{-1}$



$i^2 = -1$

$$|c|^2 = a^2 + b^2 \quad \downarrow$$

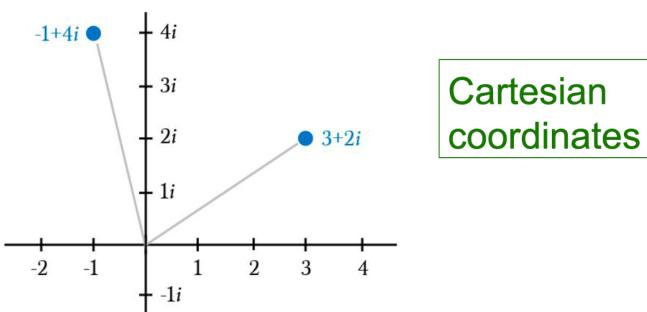
$$|c| = \sqrt{a^2 + b^2}$$

Real part Imaginary part

$$c = \boxed{a} + \boxed{i}b$$

$$a, b \in \mathbb{R}$$

Plotting complex numbers



$$\cos(\gamma) = \frac{a}{|c|} \quad \sin(\gamma) = \frac{b}{|c|}$$

$$\tan(\gamma) \quad \boxed{\frac{\sin(\gamma)}{\cos(\gamma)}} = \frac{b}{a}$$

$$\cos(\gamma) = \frac{a}{|c|} \quad \sin(\gamma) = \frac{b}{|c|}$$

$$\frac{\sin(\gamma)}{\cos(\gamma)} = \frac{b}{a}$$

$$\boxed{\gamma = \arctan\left(\frac{b}{a}\right)}$$

$$\gamma = \arctan\left(\frac{b}{a}\right)$$

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

$$\gamma = \arctan\left(\frac{b}{a}\right)$$

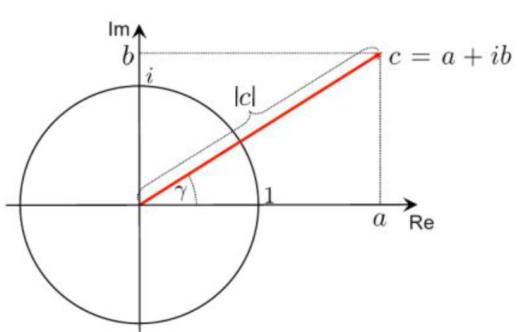
$$|c| = \sqrt{a^2 + b^2}$$

$$a = |c| \cdot \cos(\gamma) \quad b = |c| \cdot \sin(\gamma)$$

$$a = |c| \cdot \cos(\gamma) \quad b = |c| \cdot \sin(\gamma)$$

$$c = a + ib$$

Polar coordinate representation

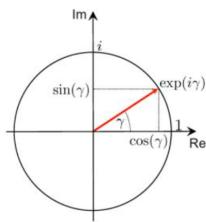


$$a = \boxed{|c| \cdot \cos(\gamma)} \quad b = \boxed{|c| \cdot \sin(\gamma)}$$

$$c = \boxed{a + ib}$$

$$c = |c| \cdot (\cos(\gamma) + i \sin(\gamma))$$

Euler Formula



$$e^{i\gamma} = \cos(\gamma) + i \sin(\gamma)$$

Euler Identity (Özdeşliği)

$e^{i\pi} + 1 = 0$

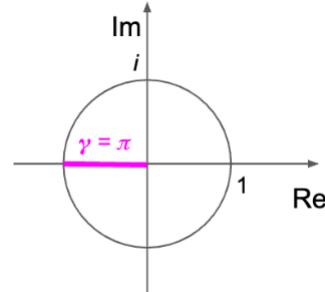
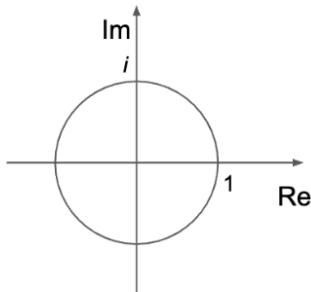
$$e^{i\pi} + 1 = 0$$

$\boxed{e^{i\pi}}$ + 1 = 0

-1



$$e^{i\gamma} = \cos(\gamma) + i \sin(\gamma)$$



Polar coordinate 2.0

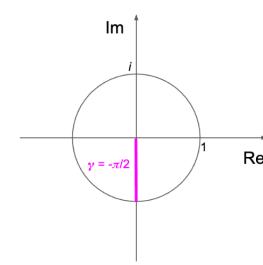
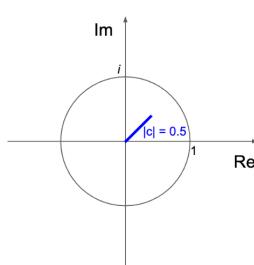
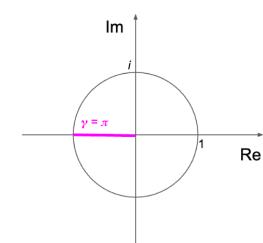
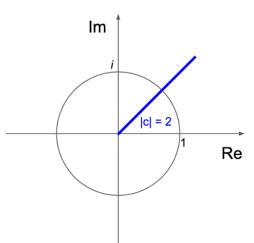
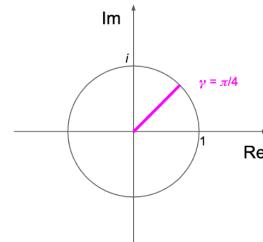
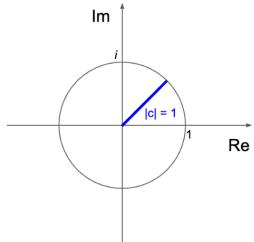
$$c = |c| \cdot (\cos(\gamma) + i \sin(\gamma))$$

$$\boxed{e^{i\gamma}} = \cos(\gamma) + i \sin(\gamma)$$

$$c = |c| \cdot e^{i\gamma}$$

Scales distance from origin

Direction of a number in the complex plane



Complex Fourier Transform Katsayılarını Elde Etme

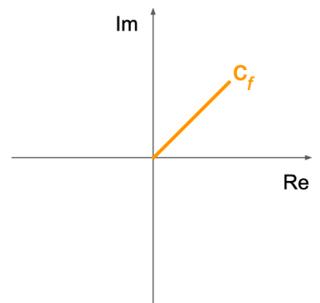
$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1)} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

$$d_f = \max_{\varphi \in [0,1)} \left(\int s(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right)$$

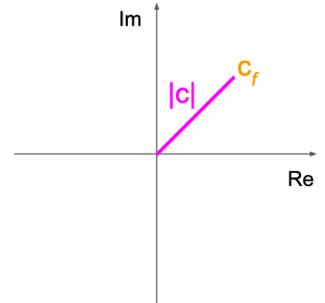
$$c = |c| \cdot e^{i\gamma}$$

$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$

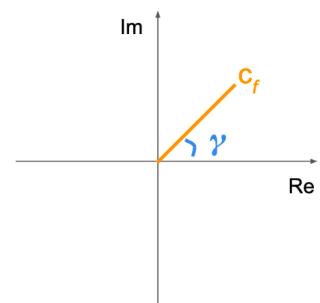
$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$



$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$

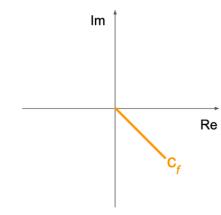
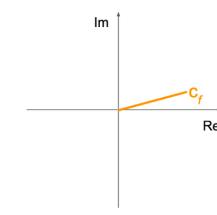
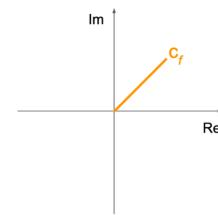


$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$



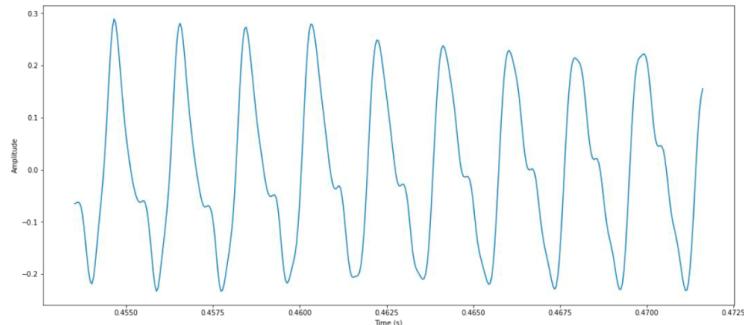
$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$

Baştaki eksiden dolayı phase'i arttırırsak saat yönünde dönmüş oluruz.



Continuous Audio Signal

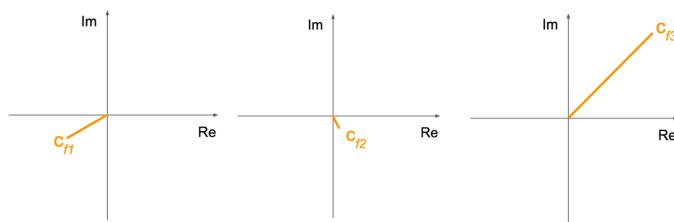
$$g(t) \quad g : \mathbb{R} \rightarrow \mathbb{R}$$



Complex Fourier Transform

$$\hat{g}(f) = c_f$$

Orijinal sinyalden biraz farklıdır çünkü bu fonksiyon giriş olarak gerçek bir sayıyı alır, bu da frekans Hertz cinsinden ifade edilir. Ancak fonksiyon sonucundan gerçek bir sayı çıkarmaz, bunun yerine karmaşık sayının çıktısını verir. Çıktarıdı complex number fourier transform katsayısidır. Buna karmaşık düzlemede bir bakanım...



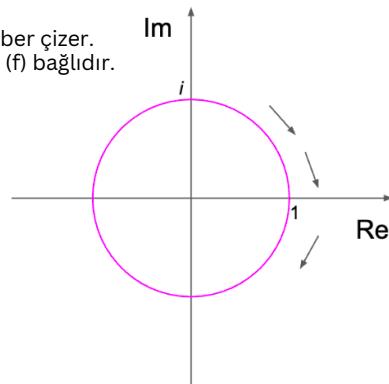
$$d_f = \max_{\varphi \in [0,1]} \left(\int g(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right) \quad (\text{Magnitude})$$

$$\varphi_f = \operatorname{argmax}_{\varphi \in [0,1]} \left(\int g(t) \cdot \sin(2\pi \cdot (ft - \varphi)) \cdot dt \right) \quad (\text{Phase})$$

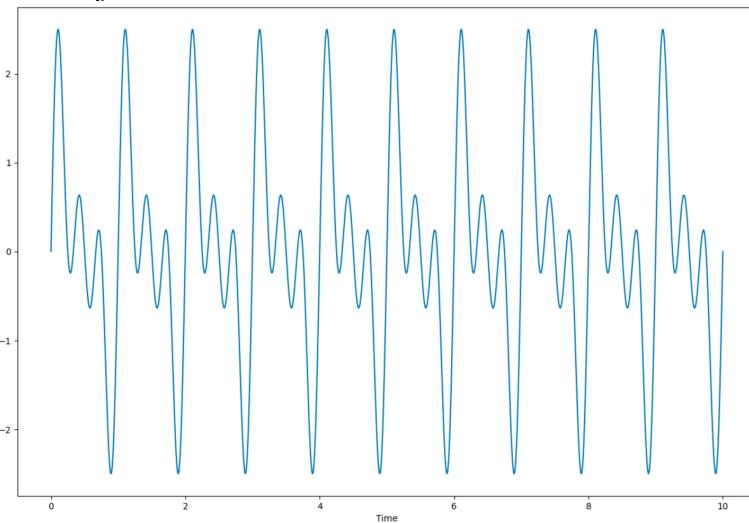
$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt \quad (\text{Complex Fourier Transform Tanımı})$$

Karmaşık düzlemede bir birim çember çizer. Bu çemberin çizilme hızı frekansa (f) bağlıdır.

$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt$$

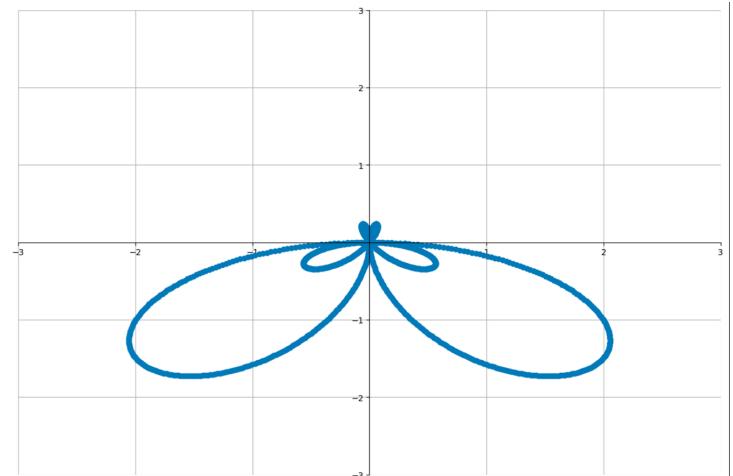


$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt$$



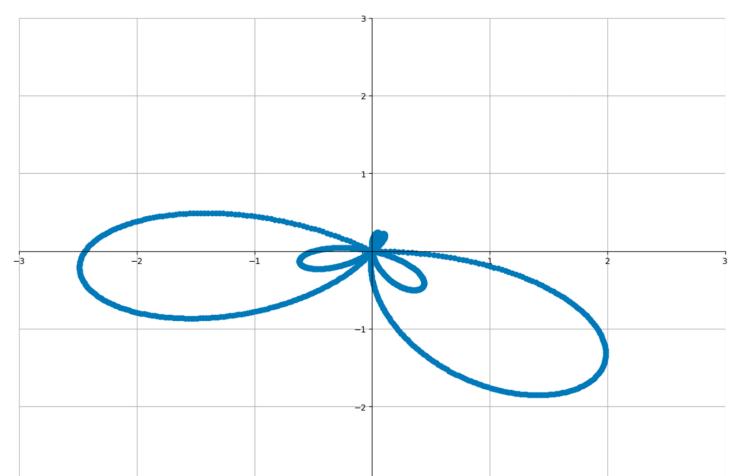
$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt$$

```
time = np.linspace(0, 10, 1000)
plot_fourier_transform(pure_tone_frequency=1,
                      signal_frequency=1,
                      time=time)
```

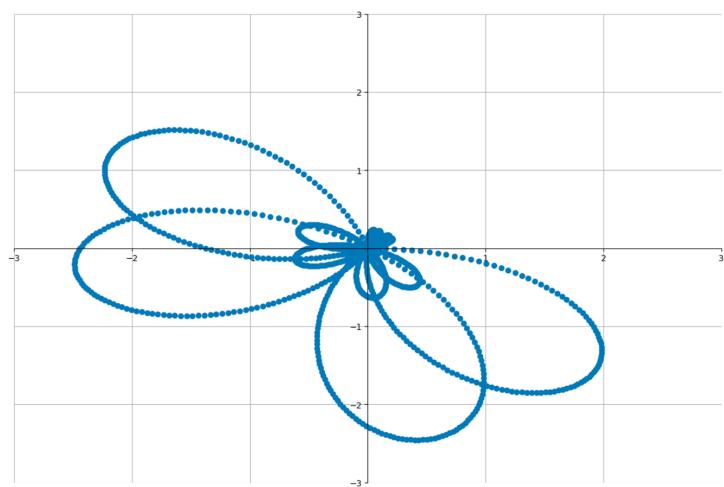


orijinal sinyali alıyoruz ve bunu karmaşık düzlemin etrafını sarıyoruz.

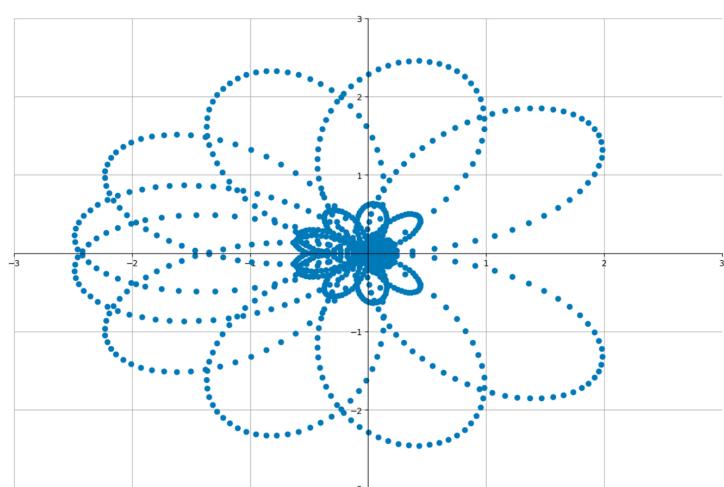
```
time = np.linspace(0, 1, 1000)
plot_fourier_transform(pure_tone_frequency=1.1,
                      signal_frequency=1,
                      time=time)
```



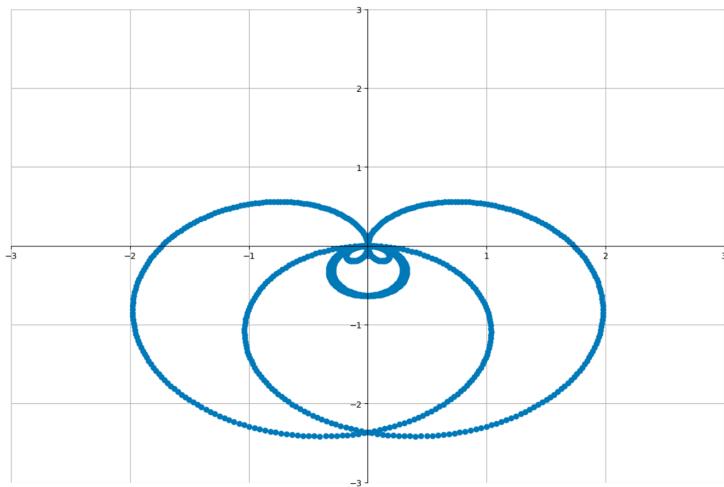
```
time = np.linspace(0, 2, 1000)
plot_fourier_transform(pure_tone_frequency=1.1,
                      signal_frequency=1,
                      time=time)
```



```
time = np.linspace(0, 5, 1000)
plot_fourier_transform(pure_tone_frequency=1.1,
                      signal_frequency=1,
                      time=time)
```

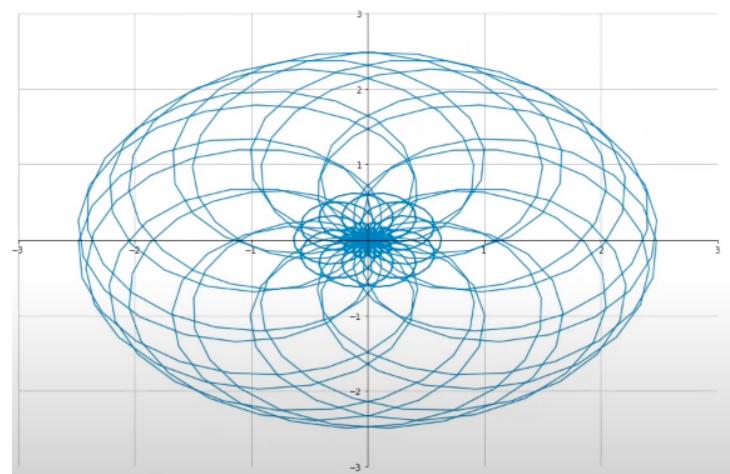


```
time = np.linspace(0, 10, 1000)
plot_fourier_transform(pure_tone_frequency=3,
                      signal_frequency=1,
                      time=time)
```



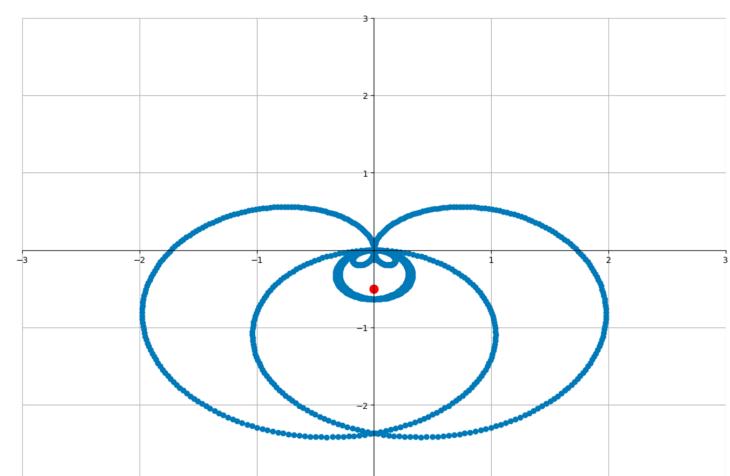
pure_tone_frequency azaldıkça daha stabil / istikrarlı bir şekil elde ederiz.

```
time = np.linspace(0, 10, 1000)
plot_fourier_transform(pure_tone_frequency=3.1,
                      signal_frequency=1,
                      time=time)
```



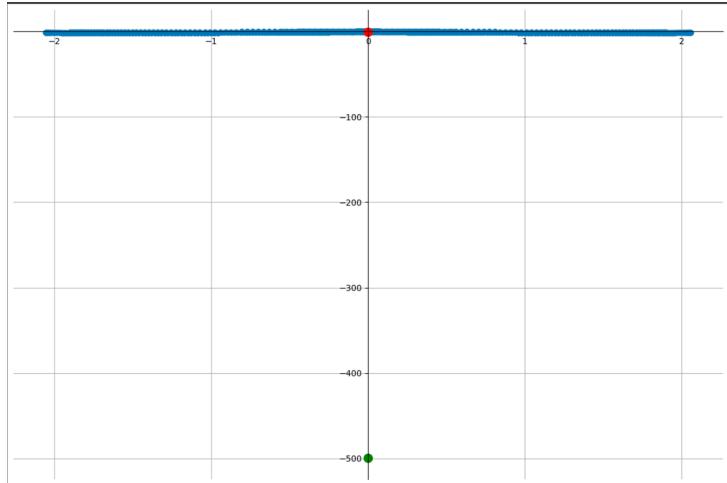
$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt$$

```
time = np.linspace(0, 10, 1000)
plot_fourier_transform(pure_tone_frequency=3,
                      signal_frequency=1,
                      time=time,
                      plot_centre_of_gravity = True)
```



Tüm bunları integralini alarak toplarsak ne olur?

```
time = np.linspace(0, 10, 1000)
plot_fourier_transform(pure_tone_frequency=1,
                      signal_frequency=1,
                      time=time,
                      plot_centre_of_gravity = True,
                      plot_sum = True)
```



$$e^{i\gamma} = \cos(\gamma) + i \sin(\gamma)$$

Real part

Imaginary part

$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt = \boxed{\int g(t) \cdot \cos(-2\pi ft) dt} + i \boxed{\int g(t) \cdot \sin(-2\pi ft) dt}$$

Magnitude Fourier Transform

$$|\hat{g}(f)|$$

Bu fonksiyonun mutlak değerini almaktır.

Magnitude and phase

$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$

$$d_f = \sqrt{2} \cdot |\hat{g}(f)|$$

$$c_f = \boxed{\frac{d_f}{\sqrt{2}}} \cdot e^{-i2\pi\varphi_f}$$

$$d_f = \sqrt{2} \cdot |\hat{g}(f)|$$

$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$

$$d_f = \sqrt{2} \cdot |\hat{g}(f)|$$

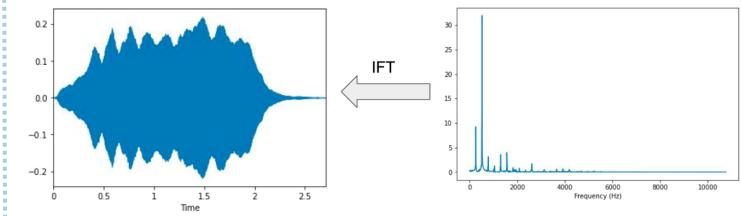
$$\varphi_f = -\frac{\gamma_f}{2\pi}$$

$$c_f = \frac{d_f}{\sqrt{2}} \cdot e^{-i2\pi\varphi_f}$$

$$d_f = \sqrt{2} \cdot |\hat{g}(f)|$$

$$\varphi_f = -\frac{\gamma_f}{2\pi}$$

Inverse Fourier Transform

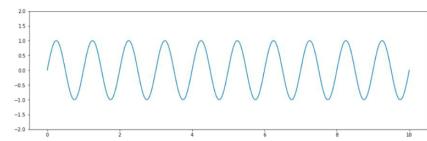


Fourier Representation

$$g(t) = \int c_f \cdot e^{i2\pi ft} df$$

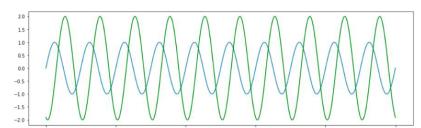
Pure tone of frequency f

$$g(t) = \int c_f \cdot \boxed{e^{i2\pi ft}} df$$



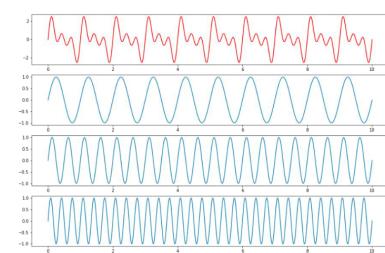
Weight pure tone with magnitude and add phase

$$g(t) = \int \boxed{c_f \cdot e^{i2\pi ft}} df$$



Add up all (weighted) sinusoids

$$g(t) = \int c_f \cdot e^{i2\pi ft} df$$



A Fourier Roundtrip

$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt$$

$$g(t) = \int c_f \cdot e^{i2\pi ft} df$$

When we digitalize a signal...

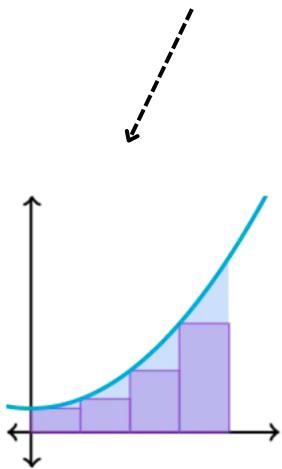
$$g(t) \mapsto x(n)$$

$$t = nT$$

Building a discrete Fourier Transform

$$\hat{g}(f) = \int g(t) \cdot e^{-i2\pi ft} dt \quad \text{Continuous Case}$$

$$\hat{x}(f) = \sum_n x(n) \cdot e^{-i2\pi fn} \quad \text{Discrete Case}$$



Discrete Fourier Transform'da birkaç hata ile karşılaşırız. Bunlar sürekli frekans (continuous frequency) ve sonsuz sayıda örnekle uğraşıldığından sonsuz zaman (infinite time) olarak söylenebilir. Formülden yola çıkarak bu durumları anlatmaya çalışalım...

$$\hat{x}(f) = \sum_n x(n) \cdot e^{-i2\pi fn}$$

Frekansla ilgili olarak, temelde bu frekansın bir fourier dönüşümünde sonsuz veya sürekli doğru gibi olması gereklidir. Bu da dijital makinelerimizle yapamayacağımız bir şeydir çünkü sonsuza benzeyen her şey hesaplama ve hafıza sorunları nedeniyle makinelerimizde çalışmaz. Bu yüzden bunu çözmeliyiz. Bunun yanı sıra sonsuz sayıda örnek oluşturmamız gerekecektir ama bu da dijital makinelerimizle yapabileceğimiz bir şey değil, bu yüzden bu iki büyük sorunu nasıl çözeceğiz?

Hack 1 : Time

- Orijinal sinyalimizi sonlu bir zaman aralığında 0 olmayan sinyallere ayırtmak istediğimiz f (frekans) değerini dikkate alıyoruz. (Consider f to be non 0 in a finite time interval)
- Yalnızca sınırlı sayıda örneğe odaklanabiliriz. $x(0), x(1), x(2), \dots, x(N-1)$

Hack 2 : Frequency

- Tüm olası frekansları almak yerine, frekansı sürekli bir değişken olarak ele alduğumuz için yalnızca bir sınırlı sayıda frekans için dönüşümü hesaplarız. (Compute transform for finite # of frequencies)
- Şimdi aklınızda "Ancak kaç frekansı dikkate almak istiyoruz?" sorusu olabilir. İşte burada güzel bir hile devreye giriyor. Temel olarak, hesaplamak istediğimiz sinyaldeki örnek sayısı n ile aynı olan frekans sayısını dikkate alırız. (# frequencies (M) = # samples (N))
- Why $M = N$?**
 - İlk neden çok önemlidir ve daha önceki bir konuda gördüğümüz bir şeydir ve temelde frekans alanından zaman alanına dönüş yapabileceğimiz bir fikirdir. Veya şu şekilde düşünebiliriz; zaman alanından başlıyoruz, frekans alanına geçiyoruz, ancak ardından ters Fourier dönüşümünü kullanarak zaman alanına geri dönebiliriz ve eğer m ile n eşitse, zaman alanından frekans alanına ve ardından frekans alanından zaman alanına oldukça kolayca geçebiliriz.
 - İkinci neden, m 'in n ile eşit olmasının hesaplamalı açıdan çok verimli olmasıdır.
 - Bu nedenlerle, sahip olduğumuz örnek sayısı ile aynı sayıda frekansı kullanırız.

Şimdi söylediklerimizi kelimelerle bağlam içinde açıklayalım ve gerçek formüllerle ne olduğunu bir göz atalım.

$$\hat{x}(f) = \sum_n x(n) \cdot e^{-i2\pi fn}$$

Discrete Fourier dönüşümünün orijinal tanımı ile başlıyoruz ve burada iki sorunumuz var, birinci sorun şu ki sonsuz bir sayıda terim içeriyor çünkü sonsuz bir örnek var. İkinci sorun ise frekansın sürekli olmasıdır, bu yüzden ilk hileyi uygularız ve sonuç olarak şunu elde ederiz.

$$\hat{x}(f) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi fn}$$

Burada sonsuz sayıda örneği toplamıyoruz, bunun yerine yalnızca N adet örneği dikkate alıyoruz ve bu harika bir şey. Şimdi bir kez daha sığır yapmamız gerekiyor, sonunda dünya genelinde ve belki de evrende kullanılan kesirli Fourier dönüşümünün tanımına ulaşıyoruz ve bu aşağıdaki şekildedir:



$$\hat{x}(k/N) = \sum_{n=0}^{N-1} x(n) \cdot e^{-i2\pi n \frac{k}{N}}$$

Görebileceğiniz gibi değiştirdiğimiz son parça şuydu: frekans f yerine k/N yazdık. Ama bu k nedir?

$$k = [0, M - 1] = [0, N - 1]$$

Peki, k'nın ne olduğunu anlamak için k'nın aralığını anlamamız gerekiyor ve k 0'dan M-1'e veya diğer bir deyişle 0'dan N-1'e kadar değişir, yani N değerine sahibiz, bu yüzden k'nın aralığının N'nin aralığıyla aynı olduğunu söyleyebiliriz.

$$F(k) = \frac{k}{NT} = \frac{ks_r}{N}$$

Bunun ne anlamına geldiğini frekans açısından anlamaya çalışalım. Verilen bir k değerinin frekansı ve bu frekans açıkça hertz cinsinden ifade edilir ve bu formül tarafından verilir, yani k bölü N büyük T ile çarpılır. T örneklem periyodu ise, örneklem hızının tersidir, diğer bir deyişle, bu formülü şu şekilde yeniden yazabiliriz: k / N ve tüm bu faktörler örneklem hızı ile çarpılır.